**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра АПУ**

отчет

**по лабораторной работе №1**

**по дисциплине «Математические основы теории систем»**

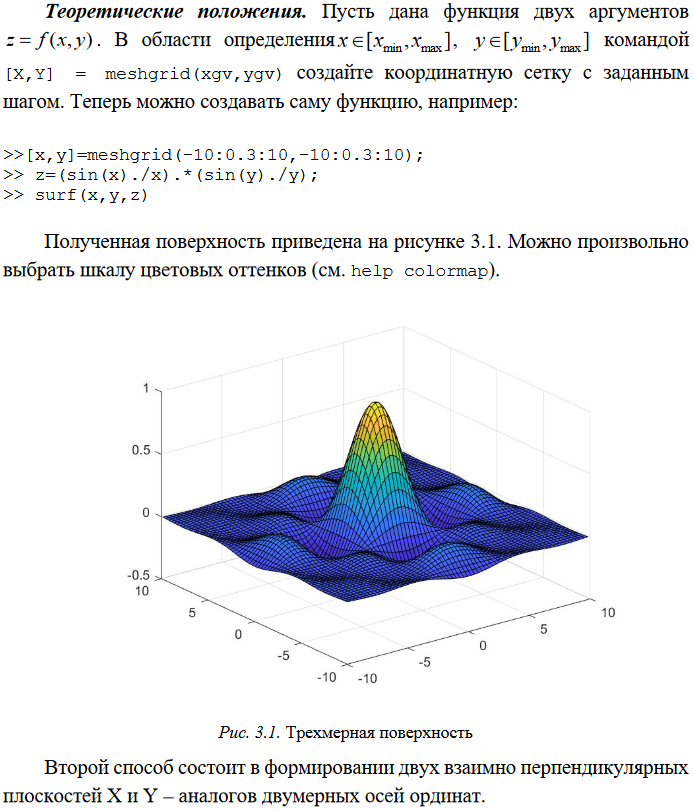
Тема: Матричные преобразования и трехмерная графика

|  |  |  |
| --- | --- | --- |
| Студент гр. 2392 |  | Жук Ф.П. |
| Преподаватель |  | Каплун Д.И. |

Санкт-Петербург

2024

**Цель работы:** освоение специфики матричных преобразований MATLAB и сравнительный анализ различных форм графического отображения результатов.

****





Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Изображение выглядит как Красочность, линия, дизайн, оригами

Автоматически созданное описание

**Ход работы.**

1. В качестве исходной фигуры, на которой будем изучать матричные преобразования, была выбрана пирамида (рис 3.3).

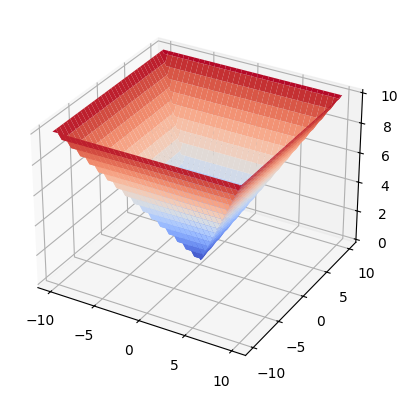


Рис. 3.3 – Пирамида

Вычисление координат пирамиды:

x, y = [np.arange(-10, 10.2, 0.2, dtype=np.float64)]\*2 # два вектора для точек

x, y = np.meshgrid(x, y) # превратим векторы в плоскоть

iter = lambda item:[

                        np.max(data)

                        for data in list(zip(item[0], item[1]))

                    ]

z = np.matrix(

    [

        iter(data)

        for data in list(zip(np.abs(x), np.abs(y)))

    ]

) # Вычислим z по функции f

z.astype(np.float64) # Приведём данные к типу float

Вывод графика:

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

ax.plot\_surface(x, y, z, cmap='coolwarm')

fig.title = 'Пирамида'

fig.show()

2. Сравним по рис. 3.4 результаты двух операций – обращения матрицы командой inv и поэлементного деления матрицы ones(n,n) на Z.

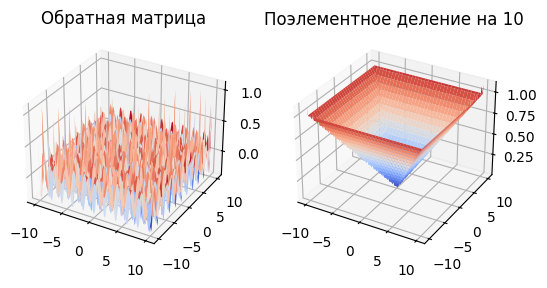


Рис. 3.4 - Результаты двух операций

При поэлементном делении высота пирамиды уменьшилась в 10 раз, а при взятии обратной матрицы была получена матрица такая матрица, которая при умножении на исходную даст единичную.

3. Сравним матричные операции sqrtm(A), logm(A), expm(A) с аналогичными операциями, выполняемыми поэлементно (рис 3.5).

Изображение выглядит как снимок экрана, дизайн

Автоматически созданное описание

Рис. 3.5 - Результаты матричных операций

4. Преобразуем пирамиду R операциями врезки. «Отрежем» какой-нибудь из углов, приравняв нулю выбранные элементы (рис. 3.6).

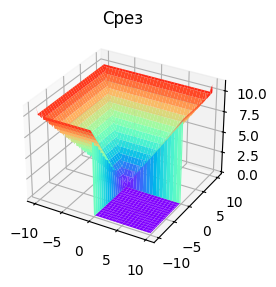


Рис. 3.6 – Срез пирамиды

5. Выполним операцию размножения массивов (мультиплицирования) на примере конусов и используем несколько видов heatmap (рис. 3.7).

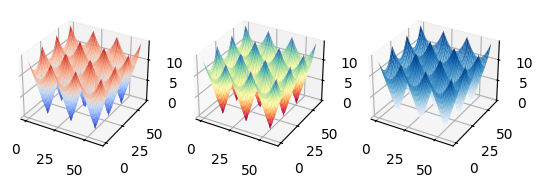


Рис. 3.6 – Результат операции размножения массивов

**Выводы.**

Оценивается степень соответствия полученных результатов расчетов и экспериментов с теоретическими данными.

Дается объяснение полученных в ходе работы зависимостей и результатов.

Приложение

Код на python:

# %% [markdown]

# Импортируем библиотеки

# %%

import numpy as np

import matplotlib.pyplot as plt

import pylab

# %% [markdown]

# Зададим точки:

# %%

x, y = [np.arange(-10, 10.2, 0.2, dtype=np.float64), np.arange(-10, 10.2, 0.2, dtype=np.float64)] # два вектора для точек

x, y = np.meshgrid(x, y) # превратим векторы в плоскоть

# %%

iter = lambda item:[

                        np.max(data)

                        for data in list(zip(item[0], item[1]))

                    ]

z = np.matrix(

    [

        iter(data)

        for data in list(zip(np.abs(x), np.abs(y)))

    ]

) # Вычислим z по функции f

z.astype(np.float64) # Приведём данные к типу float

z

# %% [markdown]

# Выведем фигуру:

# %%

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

ax.plot\_surface(x, y, z, cmap='coolwarm')

fig.title = 'Пирамида'

fig.show()

# %%

print('Определитель Матрицы Z:', np.linalg.det(z))

# %% [markdown]

# Добавим к матрице еденичную:

# %%

Z = z + np.eye(len(z))

Z

# %% [markdown]

# Обратная матрица:

# %%

fig = pylab.figure()

ax = fig.add\_subplot(121, projection='3d')

ax.plot\_surface(x, y, np.linalg.inv(Z), cmap='coolwarm')

pylab.title('Обратная матрица')

axx = fig.add\_subplot(122, projection='3d')

axx.plot\_surface(x, y, Z/10, cmap='coolwarm')

pylab.title('Поэлементное деление на 10')

pylab.show()

# %%

fig = pylab.figure()

ax = fig.add\_subplot(231, projection='3d')

ax.plot\_surface(x, y, Z@np.sqrt(np.diag(np.linalg.eig(Z)[0]))@np.linalg.inv(Z), cmap='coolwarm')

pylab.title('Корень')

axx = fig.add\_subplot(234, projection='3d')

axx.plot\_surface(x, y, np.sqrt(Z), cmap='coolwarm')

pylab.title('Поэлементное')

def logm(data):

    out = data[0]

    for i in range(1,len(data)):

        out+=i

    return(out)

axx = fig.add\_subplot(232, projection='3d')

axx.plot\_surface(x, y, logm(np.array([(-1)\*\*(i+1)\*np.matrix((Z-np.ones(len(Z)))\*\*i/i) for i in range(1,15000)])), cmap='coolwarm')

pylab.title('Логарифм')

axx = fig.add\_subplot(235, projection='3d')

axx.plot\_surface(x, y, np.log(Z), cmap='coolwarm')

pylab.title('Поэлементное')

def expm(data):

    out = data[0]

    for i in range(1,len(data)):

        out+=i

    return(out)

axx = fig.add\_subplot(233, projection='3d')

axx.plot\_surface(x, y, expm(np.array([np.matrix(Z\*\*i/np.math.factorial(i)) for i in range(150)])), cmap='coolwarm')

pylab.title('Экспонента')

axx = fig.add\_subplot(236, projection='3d')

axx.plot\_surface(x, y, np.exp(Z), cmap='coolwarm')

pylab.title('Поэлементное')

pylab.show()

# %% [markdown]

# Срез

# %%

Z[:50,50:]=0

print(Z.shape)

fig = pylab.figure()

ax = fig.add\_subplot(121, projection='3d')

ax.plot\_surface(x, y, Z, cmap='rainbow')

pylab.title('Cрез')

pylab.show()

# %% [markdown]

# 9 конусов

# %%

x, y = [np.arange(-10, 10.2, 0.2)]\*2

x, y = np.meshgrid(x, y)

z = np.sqrt(x\*\*2+y\*\*2)

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

ax.plot\_surface(x, y, z, cmap='coolwarm')

fig.title = '1 конус'

fig.show()

# %%

print(z)

# %%

x, y = [np.arange(0, 60.6, 0.2)]\*2

x, y = np.meshgrid(x, y)

z = np.kron(np.ones((3,3)), z)

# %%

fig = plt.figure()

ax = fig.add\_subplot(131, projection='3d')

ax.plot\_surface(x, y, z, cmap='coolwarm')

fig.title = '1 конус'

ax = fig.add\_subplot(132, projection='3d')

ax.plot\_surface(x, y, z, cmap='Spectral')

fig.title = '1 конус'

ax = fig.add\_subplot(133, projection='3d')

ax.plot\_surface(x, y, z, cmap='Blues')

fig.title = '1 конус'

fig.show()