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.

Теоретические положения. Пусть дана функция двух аргументов $z = f(x, y)$. В области определения $x \in [x_{\min}, x_{\max}]$, $y \in [y_{\min}, y_{\max}]$ командой `[X,Y] = meshgrid(xgv,ygv)` создайте координатную сетку с заданным шагом. Теперь можно создавать саму функцию, например:

```
>>[x,y]=meshgrid(-10:0.3:10,-10:0.3:10);
>> z=(sin(x)./x).*(sin(y)./y);
>> surf(x,y,z)
```

Полученная поверхность приведена на рисунке 3.1. Можно произвольно выбрать шкалу цветовых оттенков (см. `help colormap`).

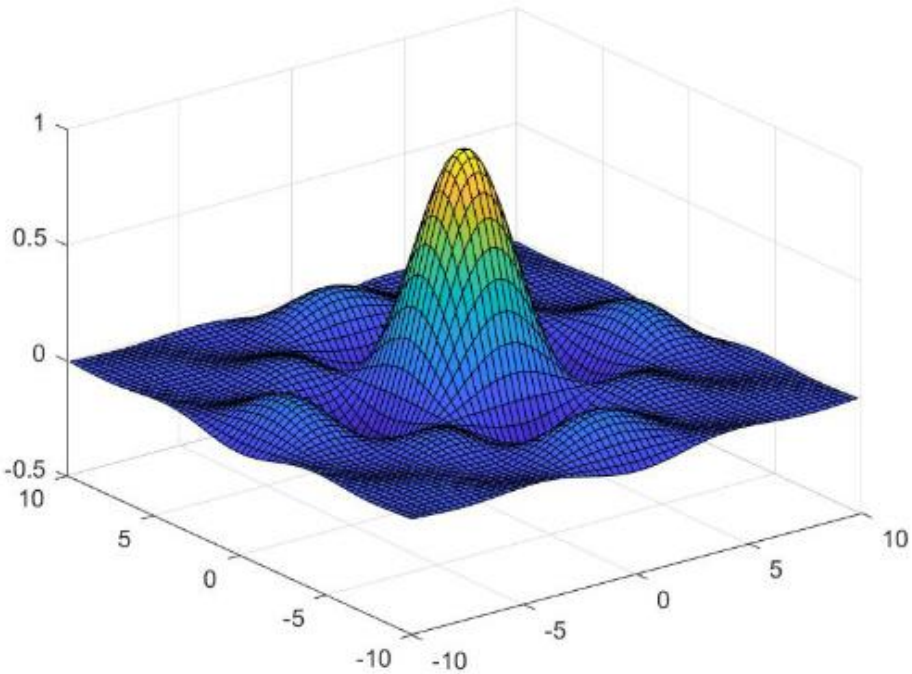

Рис. 3.1. Трехмерная поверхность

Второй способ состоит в формировании двух взаимно перпендикулярных плоскостей X и Y – аналогов двумерных осей ординат.

```
>> x=-10:10;
>> y=ones(1,21);

>> X=x'*y;
```

```
>> Y=y'*x;
```

Теперь используем координатные плоскости для построения конуса:

```
>> R1=sqrt(X.^2+Y.^2);
>> surf(X,Y,-R1);
```

или пирамиды (рис. 3.2):

```
>> R2=max(abs(X),abs(Y))
>> surf(X,Y,-R2);
```

Матричные операции, вычисление функций от матриц. Кратко напомним основные матричные операции.

Определитель Δ.

$\Delta = \sum_{i=1}^{n} a_{ij} A_{ij} \;\; j = 1 \ldots n$, A_{ij} – алгебраическое дополнение;

Обратная матрица A^{-1}.

$A^{-1} = \dfrac{A^*}{\Delta A} \;\; A^*$ – союзная матрица;

Умножение матриц $w = \alpha\beta$. Элемент результирующей матрицы w_{jk} на пересечении строки j и столбца k равен *сумме попарных произведений элементов строки* j матрицы α и столбца k матрицы β: $w_{jk} = \sum_l \alpha_{jl} \beta_{lk}$

Возведение в степень $v = w^p = w \cdot w \ldots \cdot w$ осуществляется p-кратным умножением матрицы на себя.

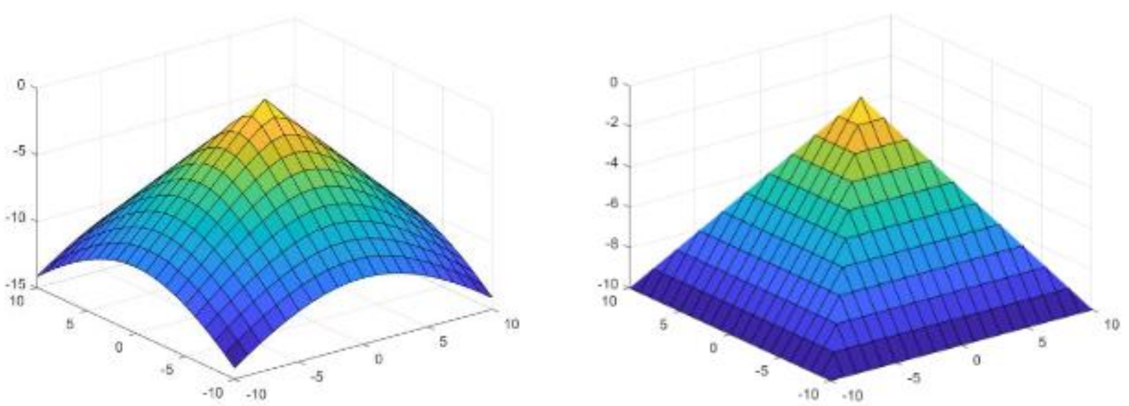

Рис. 3.2. Трехмерные фигуры

Ход работы.

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

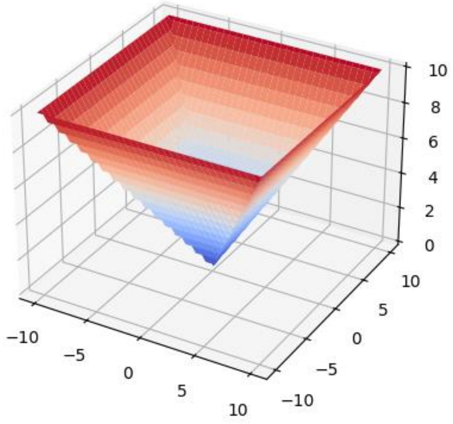

Рис. 3.3 – Пирамида

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.

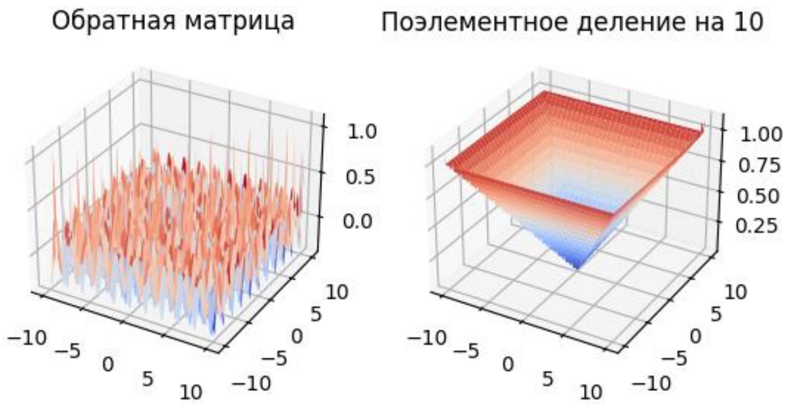

Рис. 3.4 - Результаты двух операций

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).

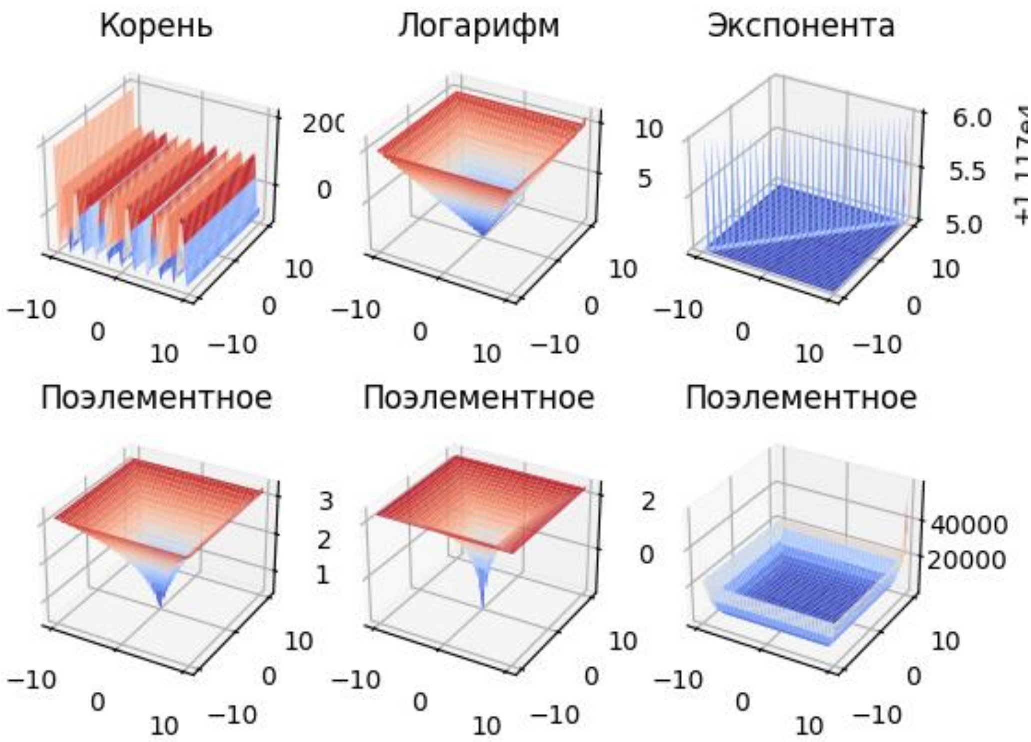

Рис. 3.5 - Результаты матричных операций

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

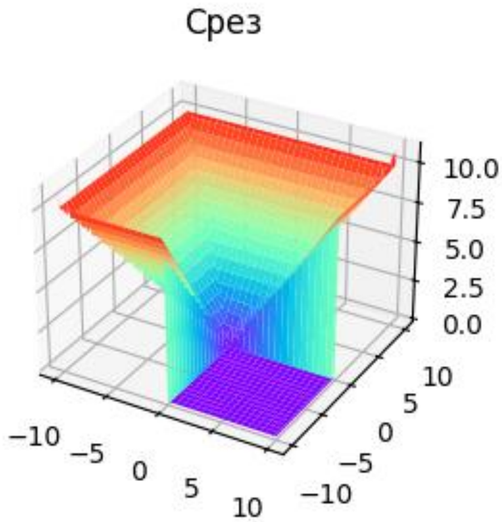

Рис. 3.6 – Срез пирамиды

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

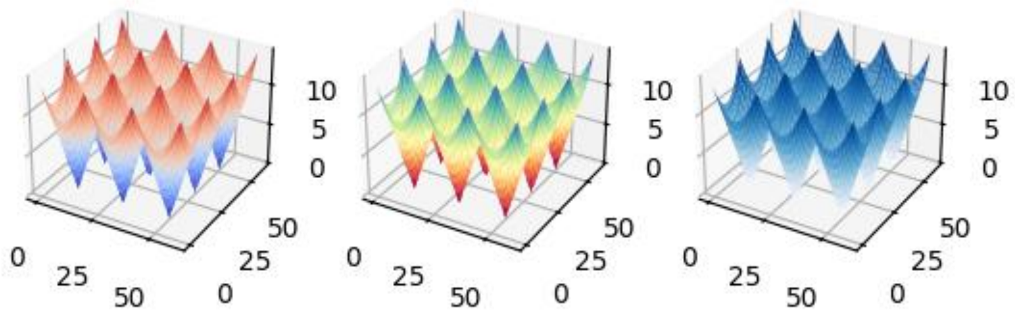

Рис. 3.6 – Результат операции размножения массивов

Выводы.

В данной лабораторной работе мы изучили основы матричных преобразований и сравнительный анализ различных форм графического отображения результатов. Мы наглядно привели примеры работ с матричными операциями. Также были рассмотрены основные методы визуализации данных, 3D графики, что позволило лучше понять влияние различных матричных операций на представление информации. В результате выполнения заданий мы смогли закрепить теоретические знания.

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('Срез')
```

```
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()
```