Санкт-Петербургский политехнический университет Петра Великого

Физико-механический институт

Высшая школа теоретической механики и математической физики

**РАБОТА №4**

**Нахождение функций форм**

по дисциплине «Вычислительная механика»

Выполнил

студент гр. 5030103/10301 <*подпись*> А.Г. Фёдоров

Руководитель

Доцент, к.ф.-м.н. <*подпись*> Е.Ю. Витохин

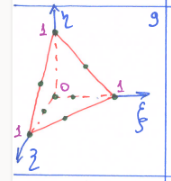
«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2023 г.

Санкт-Петербург

2023

Постановка задачи

В данной работе необходимо найти функции форм для квадратичного тетраэдра (согласно моему варианту №9).

  
Рис.1. Квадратичный тетраэдр

Интерполяционный полином квадратичного тетраэдра выглядит так:

Рассматриваемый элемент имеет узлы в следующих координатах: (0, 0, 0),

(1, 0, 0),

(0, 1, 0),

(0, 0, 1)

(0.5, 0, 0)

(0, 0.5, 0)

(0, 0, 0.5)

(0.5, 0.5, 0)

(0.5, 0, 0.5)

(0, 0.5, 0.5)

**Описание метода решения**

Функции формы для квадратичного тетраэдра представляют собой плоскости, область определения которых ограничена:

То есть необходимо найти 100 коэффициентов. Для их определения используем основные свойства функций форм. В своем узле функция формы равна 1, а в остальных узлах равна 0.

Можно записать аналогичные уравнения для - и получится всего 100 уравнений.

Если записать уравнения в матричном виде:

Либо:

- матрица с координатами

– матрица с коэффициентами функций формы

Последнее соотношение можно представить как:

Представим интерполяционный полином в виде вектора-столбца:

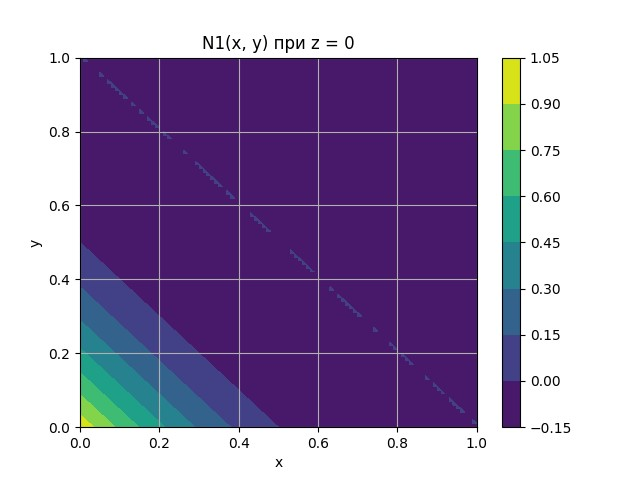
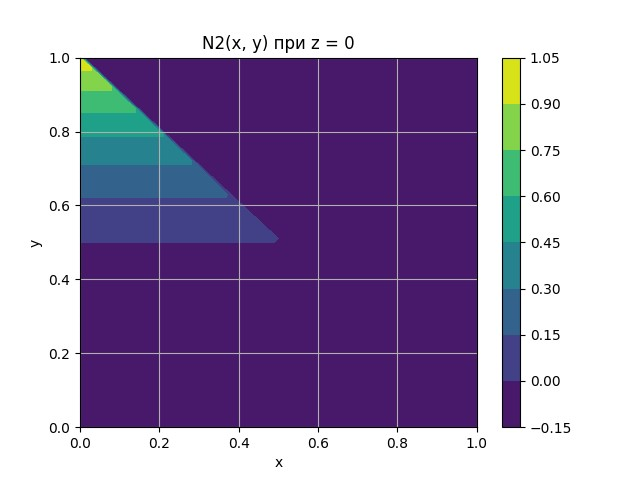
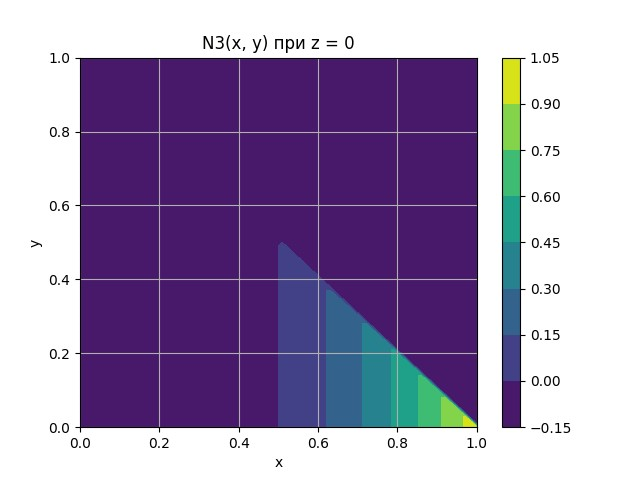
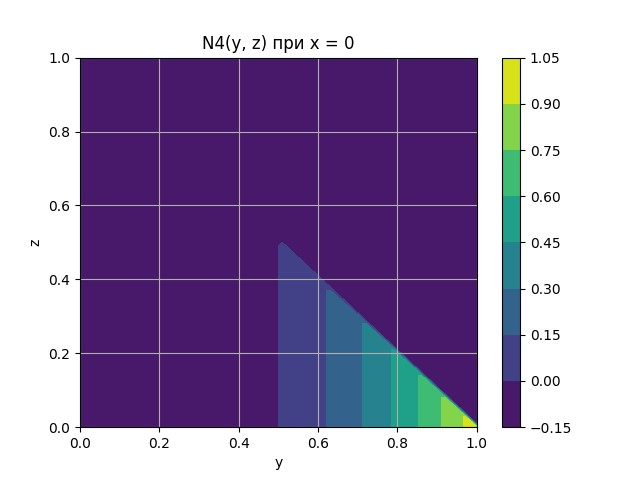
Теперь матрицу с функциями форм вида

Можно получить таким образом:

**Результаты**

Были получены следующие функции форм:

Были получены графики распределения функций форм для каждого узла в проекциях на плоскости. На рисунках 2-11 представлено распределение функций форм для десяти узлов.

Рис.2. Проекция N1 на OXY Рис.3. Проекция N2 на OXY

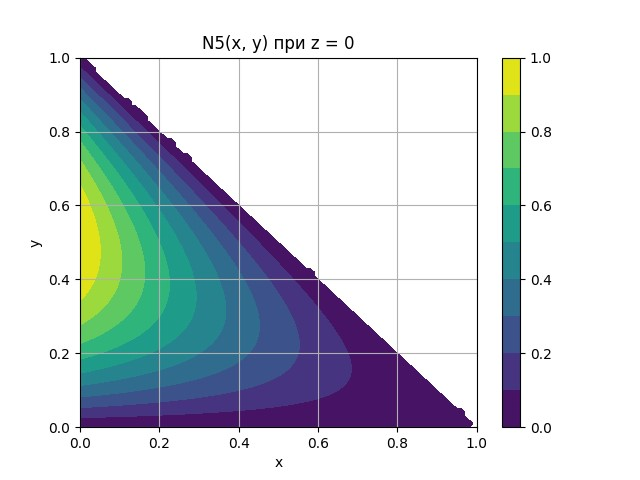
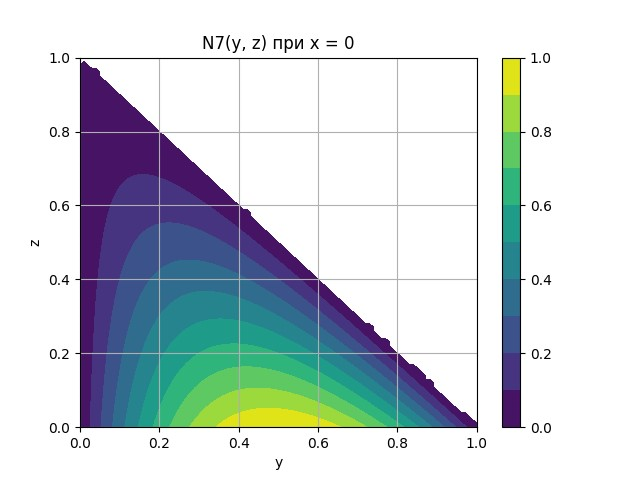
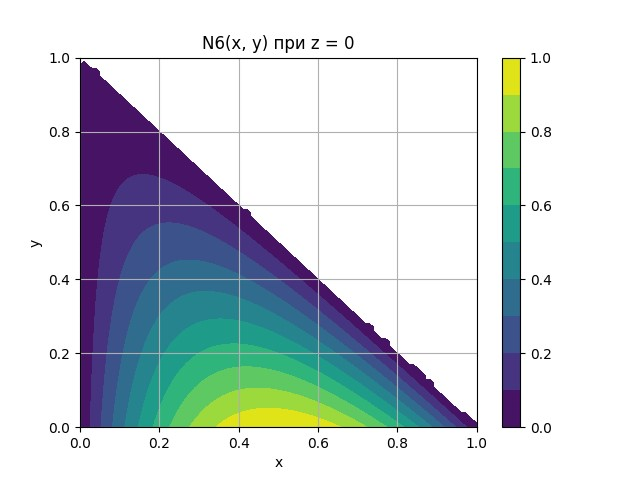
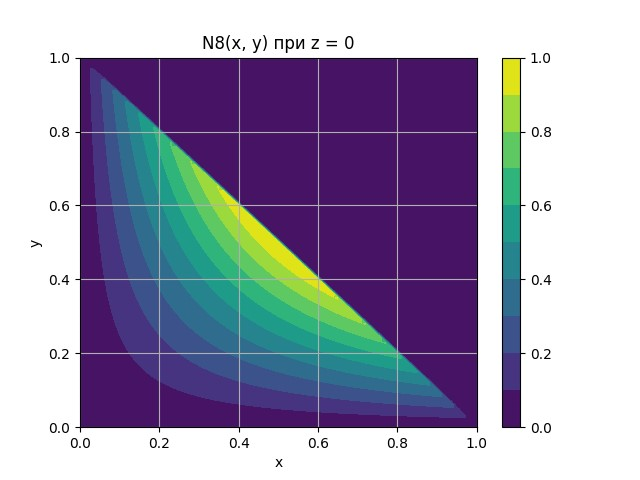
Рис.4. Проекция N3 на OYZ Рис.5. Проекция N4 на OXY

Рис.6. Проекция N5 на OXY Рис.7. Проекция N6 на OXY

Рис.8. Проекция N7 на OYZ Рис.9. Проекция N8 на OXY

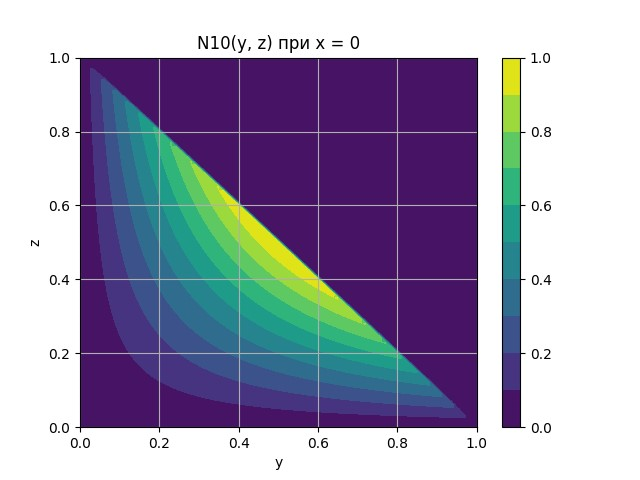
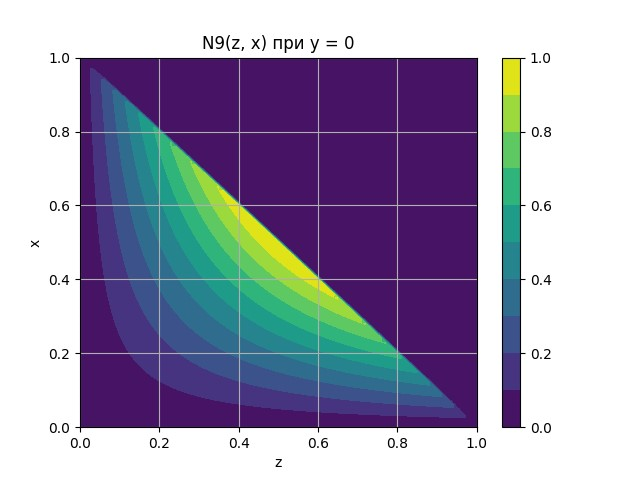


Рис.10. Проекция N9 на OZX Рис.11. Проекция N10 на OYZ

**Код программы**

**Файл interpolation\_polynom.py**

from ctypes import sizeof

import numpy as np

import matplotlib.pyplot as plt

import element as f

import sympy as sp

def get\_x\_matrix(variables):

dimension = len(variables)

X = np.zeros( (dimension, dimension) )

for i in range(0, dimension):

current\_set = variables[i, :]

X[i, : ] = f.fill\_form\_vector(current\_set[0], current\_set[1], current\_set[2])

X = np.linalg.inv(X)

X = X.T

return X

def graph(F, step, title, xlabel, ylabel, projection):

xl = np.linspace(0, 1, step)

xgrid, ygrid = np.meshgrid(xl, xl)

fig, ax = plt.subplots()

cs = ax.contourf(xgrid, ygrid, F, levels = 10)

cbar = plt.colorbar(cs)

ax.set\_xlabel("{}".format(xlabel))

ax.set\_ylabel("{}".format(ylabel))

ax.set\_title("{0}({1}, {2}) при {3} = 0" .format(title, xlabel, ylabel, projection))

ax.grid()

plt.savefig("fileout/Функция формы({}).png".format(title))

def main():

varis = f.getFigure()

X = get\_x\_matrix(varis)

print(X)

step = 101

step1 = 101

N1 = np.zeros((step, step))

N2 = np.zeros((step, step))

N3 = np.zeros((step, step))

N4 = np.zeros((step, step))

N5 = np.zeros((step, step))

N6 = np.zeros((step, step))

N7 = np.zeros((step, step))

N8 = np.zeros((step, step))

N9 = np.zeros((step, step))

N10 = np.zeros((step, step))

xcoord = np.linspace(0, 1, step)

ycoord = np.linspace(0, 1, step)

zcoord = np.linspace(0, 1, step)

for i in range(step):

for j in range(step1):

N1[i][j] = 1 - 3 \* xcoord[i] - 3 \* ycoord[j] + 2 \* xcoord[i] \*\* 2 + 2 \* ycoord[j] \*\* 2 + 4 \* xcoord[i] \* ycoord[j]

N2[i][j] = -xcoord[i] + 2 \* xcoord[i] \*\* 2

N3[i][j] = -ycoord[j] + 2 \* ycoord[j] \*\* 2

N4[i][j] = -zcoord[j] + 2 \* zcoord[j] \*\* 2

N5[i][j] = 4 \* xcoord[i] - 4 \* xcoord[i] \*\* 2 - 4 \* xcoord[i] \* ycoord[j]

N6[i][j] = 4 \* ycoord[j] - 4 \* ycoord[j] \*\* 2 - 4 \* xcoord[i] \* ycoord[j]

N7[i][j] = 4 \* zcoord[j] - 4 \* zcoord[j] \*\* 2 - 4 \* ycoord[i] \* zcoord[j]

N8[i][j] = 4 \* xcoord[i] \* ycoord[j]

N9[i][j] = 4 \* zcoord[i] \* xcoord[j]

N10[i][j] = 4 \* ycoord[i] \* zcoord[j]

step1 = step1 - 1

graph(N1, step,'N1', 'x', 'y', 'z')

graph(N2, step,'N2', 'x', 'y', 'z')

graph(N3, step,'N3', 'x', 'y', 'z')

graph(N4, step,'N4', 'y', 'z', 'x')

graph(N5, step,'N5', 'x', 'y', 'z')

graph(N6, step,'N6', 'x', 'y', 'z')

graph(N7, step,'N7', 'y', 'z', 'x')

graph(N8, step,'N8', 'x', 'y', 'z')

graph(N9, step,'N9', 'z', 'x', 'y')

graph(N10, step,'N10', 'y', 'z', 'x')

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Файл element.py**

import numpy as np

def getFigure():

X= [[0, 0, 0],

[1, 0, 0],

[0, 1, 0],

[0, 0, 1],

[0.5, 0, 0],

[0, 0.5, 0],

[0, 0, 0.5],

[0.5, 0.5, 0],

[0.5, 0, 0.5],

[0, 0.5, 0.5]]

return np.array( X )

def fill\_form\_vector(x, y, z):

return [1+0\*x, x, y, z, x \*\* 2, y \*\* 2, z \*\* 2, x \* y, x \* z, y \* z]