**Лабораторная работа 5**

**Исходный код программы**

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

import plotly.plotly as py

import plotly.graph\_objs as go

import plotly

A = 180

B = 100

R = 30

d = 2

P = 70

E = 60

v = 0.28

# In[40]:

D = 10 \* E \* d\*\*3 / (12 \* (1 - v\*\*2))

P / D

h = 2

nx = len(np.arange(0, A + h, h))

ny = len(np.arange(0, B + h, h))

# In[48]:

def plot(nx, ny, h, u):

s = np.arange(0, A + h, h)

t = np.arange(0, B + h, h)

tGrid, sGrid = np.meshgrid(t, s)

surface = go.Surface(x=sGrid, y=tGrid, z=u.T)

data = [surface]

layout = go.Layout(

title='Parametric Plot',

scene=dict(

xaxis=dict(

gridcolor='rgb(255, 255, 255)',

showbackground=True,

backgroundcolor='rgb(230, 230,230)'

),

yaxis=dict(

title='t',

gridcolor='rgb(255, 255, 255)',

showbackground=True,

backgroundcolor='rgb(230, 230,230)'

),

zaxis=dict(

title='u(x, t)',

gridcolor='rgb(255, 255, 255)',

showbackground=True,

backgroundcolor='rgb(230, 230,230)'

)

)

)

fig = go.Figure(data=data, layout=layout)

plotly.offline.plot(fig, auto\_open=True)

u = np.zeros((ny, nx))

for i in range(ny):

for j in range(nx):

if np.sqrt((A / 2 - h \* j)\*\*2 + (i \* h)\*\*2) <= R or i == 0 or j == 0 or i == ny - 1 or j == nx - 1:

u[i][j] = 1

plot(nx, ny, h, u)

def get\_u(eps):

cur\_error = 1

iters = 0

while cur\_error > eps:

cur\_error = 0

iters += 1

for i in range(1, ny - 1):

for j in range(1, nx - 1):

if np.sqrt((A / 2 - h \* j)\*\*2 + (i \* h)\*\*2) <= R:

u[i][j] = 1

else:

prev = u[i][j]

u[i][j] = (u[i, j + 1] + u[i + 1, j] + u[i - 1, j] + u[i, j - 1] - h \* h \* P / D) / 4

error = abs(prev - u[i][j])

cur\_error = max(error, cur\_error)

print(cur\_error)

return iters, u

get\_ipython().run\_cell\_magic('time', '', 'iters, u = get\_u(0.001)\nplot(nx, ny, h, u) ')

epsilons = np.linspace(0.5, 0.001, 50)

iters = []

errors = []

for i, eps in enumerate(epsilons):

it, u1 = get\_u(eps)

if i == 0:

iters.append(it)

else:

iters.append(it + iters[i - 1])

print(it)

# In[78]:

iters

# In[79]:

plt.plot(epsilons, iters)

# In[89]:

u1 = np.copy(u)

for h in [3, 4, 7, 9]:

nx = len(np.arange(0, A + h, h))

ny = len(np.arange(0, B + h, h))

u = np.zeros((ny, nx))

for i in range(ny):

for j in range(nx):

if np.sqrt((A / 2 - h \* j)\*\*2 + (i \* h)\*\*2) <= R or i == 0 or j == 0 or i == ny - 1 or j == nx - 1:

u[i][j] = 1

iters = []

for i, eps in enumerate(epsilons):

it, u = get\_u(eps)

if i == 0:

iters.append(it)

else:

iters.append(it + iters[i - 1])

print(it)

#dx, dy = int(u.shape[1] / u1.shape[1]), int(u.shape[0] / u1.shape[0])

#errors += np.mean(np.abs(u[::dx, ::dy] - u1))

plot(nx, ny, h, u)

plt.plot(epsilons, iters, label='h = {}'.format(h))

plt.xlabel('error')

plt.ylabel('steps')

plt.label()

plt.show()

plt.plot(epsilons, iters, label='h = {}'.format(h))