БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

**Лабораторная работа №2**

**Решение краевой задачи для уравнения колебания**

**Вариант К-1**

Выполнил: Белоушко Степан

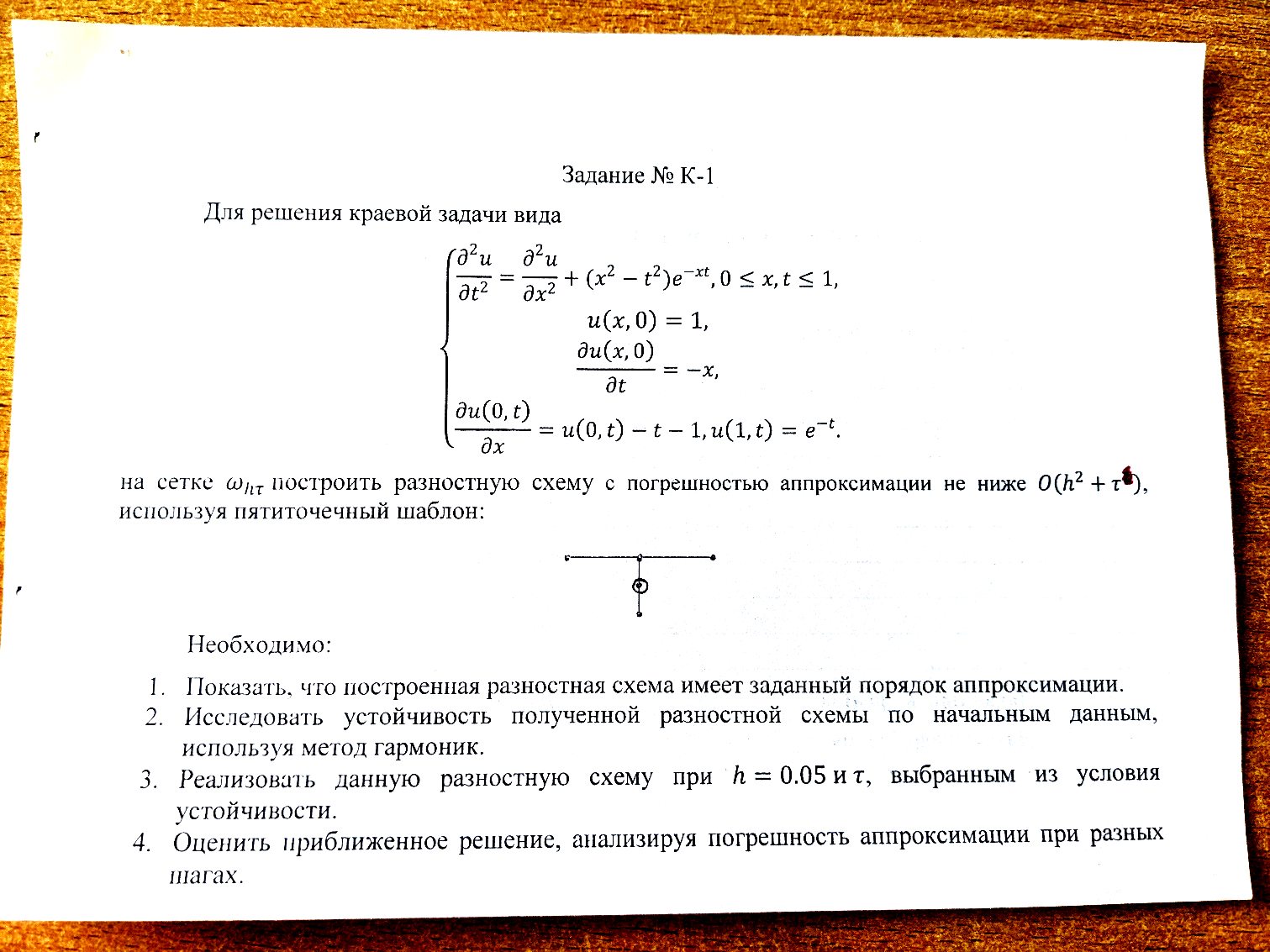
3 курс 7 группа

Преподаватель:

Будник Анатолий Михайлович

Минск, 2023

# Постановка задачи



# Теоретические сведения

Так как разностная производная по оси Ox возможна только на верхнем уровне шаблона, то получим разносную схему

# Листинг программы

import numpy as np

from warnings import warn

import matplotlib.pyplot as plt

def sweep\_method(e: np.ndarray, d: np.ndarray, c: np.ndarray, b: np.ndarray) -> np.ndarray:

'''

Метод прогонки для СЛАУ для трехдиагональных матриц, где:

d - элементы главной диагонали,

e - элементы над главной диагональю и последний 0,

c - элементы под главной диагональю и первый 0,

b - неоднородность.

'''

n = d.shape[0]

x = np.zeros\_like(d)

for k in range(1, n):

d[k] = d[k] - e[k-1]\*c[k]/d[k-1]

b[k] = b[k] - b[k-1]\*c[k]/d[k-1]

x[n-1] = b[n-1]/d[n-1]

for k in reversed(range(n-1)):

x[k] = (b[k] - e[k]\*x[k+1])/d[k]

return x

def f(x, t):

return (x\*x-t\*t)\*np.exp(-x\*t)

def u0():

return 1

def u1(x):

return -x

def mu0(t):

return t+1

def mu1(t):

return np.exp(-t)

h = 0.001

tau = 0.001

X = np.arange(0, 1+h/2, h)

T = np.arange(0, 1+tau/2, tau)

N1 = T.size

N2 = X.size

y = np.zeros(shape=(N1, N2))

for i in range(N2):

y[0][i] = u0()

y[1][i] = y[0][i] + tau\*u1(X[i])

y[1][-1] = mu1(tau)

for j in range(1, N1-1):

diag\_el = np.zeros(N2)

diag\_el[0] = -1/h - 1 - h/(2\*tau\*tau)

diag\_el[-1] = 1

for i in range(1, N2-1):

diag\_el[i] = 1/(tau\*tau) + 2/(h\*h)

up\_diag = np.zeros(N2)

up\_diag[0] = 1/h

for i in range(1, N2-1):

up\_diag[i] = -1/(h\*h)

low\_diag = np.zeros(N2)

for i in range(1, N2-1):

low\_diag[i] = -1/(h\*h)

neodnorodnost = np.zeros(N2)

neodnorodnost[-1] = mu1(T[j+1])

neodnorodnost[0] = -mu0(T[j]) - h/2\*f(0, T[j]) + h/(2\*tau\*tau)\*(-2\*y[j][0] + y[j-1][0])

for i in range(1, N2-1):

neodnorodnost[i] = 1/(tau\*tau)\*(2\*y[j][i] - y[j-1][i]) + f(X[i], T[j])

y[j+1] = sweep\_method(up\_diag, diag\_el, low\_diag, neodnorodnost)

for j in range(0, T.size, 100):

for i in range(0, X.size, 100):

print('%.4f' % y[j][i], end=' ')

print()

# Результаты

При шагах

1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000

0.9991 0.9900 0.9802 0.9704 0.9607 0.9511 0.9416 0.9322 0.9228 0.9136 0.9048

0.9983 0.9794 0.9606 0.9417 0.9230 0.9046 0.8866 0.8689 0.8516 0.8350 0.8187

0.9975 0.9687 0.9409 0.9137 0.8867 0.8604 0.8348 0.8100 0.7862 0.7632 0.7408

0.9969 0.9584 0.9214 0.8860 0.8517 0.8183 0.7861 0.7554 0.7259 0.6976 0.6703

0.9963 0.9482 0.9024 0.8589 0.8177 0.7782 0.7405 0.7046 0.6703 0.6376 0.6065

0.9958 0.9382 0.8839 0.8328 0.7848 0.7399 0.6975 0.6572 0.6189 0.5829 0.5488

0.9954 0.9283 0.8658 0.8076 0.7536 0.7034 0.6567 0.6128 0.5715 0.5328 0.4966

0.9951 0.9187 0.8483 0.7835 0.7238 0.6687 0.6179 0.5711 0.5276 0.4870 0.4493

0.9949 0.9094 0.8314 0.7602 0.6952 0.6358 0.5815 0.5319 0.4866 0.4450 0.4066

0.9949 0.9004 0.8150 0.7377 0.6678 0.6045 0.5472 0.4954 0.4486 0.4063 0.3679

При шагах

1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000

0.9999 0.9900 0.9802 0.9704 0.9608 0.9512 0.9417 0.9324 0.9231 0.9139 0.9048

0.9998 0.9801 0.9608 0.9418 0.9231 0.9048 0.8869 0.8693 0.8521 0.8352 0.8187

0.9997 0.9703 0.9417 0.9139 0.8869 0.8607 0.8352 0.8105 0.7866 0.7634 0.7408

0.9997 0.9605 0.9229 0.8868 0.8521 0.8187 0.7866 0.7557 0.7261 0.6977 0.6703

0.9996 0.9509 0.9046 0.8605 0.8186 0.7787 0.7408 0.7047 0.6703 0.6376 0.6065

0.9996 0.9414 0.8866 0.8350 0.7864 0.7407 0.6977 0.6571 0.6188 0.5828 0.5488

0.9995 0.9320 0.8690 0.8103 0.7556 0.7046 0.6570 0.6127 0.5712 0.5326 0.4966

0.9995 0.9227 0.8517 0.7863 0.7259 0.6702 0.6187 0.5712 0.5273 0.4868 0.4493

0.9995 0.9135 0.8349 0.7631 0.6974 0.6374 0.5826 0.5325 0.4867 0.4449 0.4066

0.9995 0.9044 0.8184 0.7405 0.6701 0.6063 0.5486 0.4965 0.4492 0.4065 0.3679

Норма погрешности - 0.00180

# Вывод