Министерство образования Республики Беларусь

Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Факультет компьютерных систем и сетей

Кафедра информатики

Лабораторная работа №16

«Метод сеток решения волнового уравнения»

Выполнил:

cтудент гр. 953506

Кондрашов И.Д.

Руководитель:

доцент

Анисимов В.Я.

Минск 2021

**Цель работы**

* изучить метод разностных аппроксимаций для волнового уравнения,

составить алгоритмы решения волнового уравнения методом сеток,

применимым для организации вычислений на ПЭВМ;

* составить программы волнового уравнения по разработанным

алгоритмам;

* получить численное решение волнового уравнения.

1. **Краткие теоретические сведения**

Рассмотрим смешанную задачу для волнового уравнения

где – заданные достаточно гладкие функции, причём

Будем предполагать, что задача имеет единственное решение – замкнутый прямоугольник.

Используем сетки, построенные на замкнутом прямоугольнике из ЛР №14. Получаем разностное уравнение

Это уравнение можно разрешить явно относительно Но для того, чтобы находить значения разностного решения на -м слое, требуется иметь уже вычисленные значения искомого решения на двух предыдущих слоях. Используя первое начальное условие, задаём

Кроме того, полагаем при

Правая часть формулы выше аппроксимирует многочлен Тейлора поскольку а из дифференциального уравнения вытекает связь Наконец, согласно краевым условиям

Если в данной задаче ввести дискретность только по , то мы придём к системе линейных ОДУ

с начальными условиями

причём Данный метод называется методом прямых.

1. **Программная реализация**

Для выполнения лабораторной работы были использованы язык программирования *Python* и среда разработки *PyCharm*.

Были разработаны следующие программные модули:

* **Задание 1.** *task1.py (численное решение и построение соответствующих графиков).*
* **Задание 2.** *task2.py (численное решение и построение соответствующих графиков).*

**Вариант 10**

**Задание 1 (набор параметров 4).**

Продольные колебания тяги описываются уравнением

где модуль упругости, плотность материала стержня. Тяга имеет длину и закреплена на концах. Захватив тягу в центре, её деформируют так, что продольное перемещение становится равным

Затем тяга освобождается.

Численное решение:

Chart, line chart

Description automatically generated

**Задание 2 (набор параметров 4).**

Колебания тонкой пластины без учёта потерь на трение описываются нормированным волновым уравнением вида

где – деформация пластины, координаты, время.

Колебания пластины в момент времени

Chart, surface chart

Description automatically generated

Колебания пластины в момент времени

Chart, surface chart

Description automatically generated

Колебания пластины в момент времени

Chart, surface chart

Description automatically generated

Колебания пластины в момент времени

Chart, surface chart

Description automatically generated

Колебания пластины в момент времени

Chart, surface chart

Description automatically generated

**Полный листинг кода:**

Задание №1

import numpy as np  
from matplotlib import pyplot as plt  
  
L = 0.15  
du = 0.001  
E = 86e9  
rho = 85e2  
T = 0.001  
  
def solve(N, M, build\_plot=False, count=0):  
 f = lambda x: (-4 \* du / L \*\* 2) \* x \*\* 2 + (4 \* du / L) \* x  
  
 tau = T / M  
 h = L / N  
 C = np.sqrt(E / rho) \* tau / h  
   
 x\_values = np.linspace(0, L, N)  
 matrix = np.zeros((M, N))  
   
 matrix[0] = [f(x) for x in x\_values]  
 matrix[1][1:-1] = [f(x\_values[i]) \* (1 - tau \*\* 2 / 2) for i in range(1, N - 1)]  
  
 for i in range(1, M - 1):  
 matrix[i + 1][1:-1] = C \*\* 2 \* (matrix[i][2:] - 2 \* matrix[i][1:-1] + matrix[i][:-2])  
 matrix[i + 1][1:-1] += 2 \* matrix[i][1:-1] - matrix[i-1][1:-1]  
   
 if build\_plot:  
 for i in range(0, M, int(M / count)):  
 plt.plot(x\_values, matrix[i])  
 plt.xlabel('x')  
 plt.ylabel('u(x, t)')  
  
 plt.grid()  
 plt.show()  
  
N = 100  
M = 10000  
A = solve(N, M, build\_plot=True, count=15)

Задание №2

import math  
import numpy as np  
import chart\_studio.plotly as py  
import plotly.graph\_objs as graph\_objs  
import plotly  
import time  
  
a = 2  
b = 3  
T = 4  
  
p = lambda x, y: np.tan(np.cos(np.pi \* y / b))  
q = lambda x, y: np.exp(np.sin(np.pi \* x / a)) \* np.sin(2 \* np.pi \* y / b)  
  
  
def solve(NX, NY, M, build\_plot=False):  
 tau = T / M  
 hx = a / NX  
 hy = b / NY  
 C = tau / hx + tau / hy  
  
 x\_values = np.linspace(-a / 2, a / 2, NX)  
 y\_values = np.linspace(-b / 2, b / 2, NY)  
 matrix = np.zeros((M, NX, NY))  
  
 for i in range(NX):  
 for j in range(NY):  
 matrix[0][i][j] = p(x\_values[i], y\_values[j])  
  
 for i in range(1, NX - 1):  
 for j in range(1, NY - 1):  
 matrix[1][i][j] = p(x\_values[i], y\_values[j]) + q(x\_values[i], y\_values[j]) \* tau  
 matrix[1][i][j] += tau \*\* 2 / (2 \* hx \*\* 2) \* (  
 matrix[0][i + 1][j] - 2 \* matrix[0][i][j] + matrix[0][i - 1][j])  
 matrix[1][i][j] += tau \*\* 2 / (2 \* hy \*\* 2) \* (  
 matrix[0][i][j + 1] - 2 \* matrix[0][i][j] + matrix[0][i][j - 1])  
  
 matrix[1, 1:-1, 0] = matrix[1, 1:-1, 1]  
 matrix[1, 1:-1, -1] = matrix[1, 1:-1, -2]  
  
 for t in range(1, M - 1):  
 matrix[t + 1, 1:-1, 1:-1] = 2 \* matrix[t, 1:-1, 1:-1] - matrix[t - 1, 1:-1, 1:-1]  
 matrix[t + 1, 1:-1, 1:-1] += tau \*\* 2 / hx \*\* 2 \* (  
 matrix[t, :-2, 1:-1] - 2 \* matrix[t, 1:-1, 1:-1] + matrix[t, 2:, 1:-1])  
 matrix[t + 1, 1:-1, 1:-1] += tau \*\* 2 / hy \*\* 2 \* (  
 matrix[t, 1:-1, :-2] - 2 \* matrix[t, 1:-1, 1:-1] + matrix[t, 1:-1, 2:])  
 matrix[t + 1, 1:-1, 0] = matrix[t + 1, 1:-1, 1]  
 matrix[t + 1, 1:-1, -1] = matrix[t + 1, 1:-1, -2]  
  
 for i in range(0, M, 999):  
 x\_grid, y\_grid = np.meshgrid(x\_values, y\_values)  
 surface = graph\_objs.Surface(x=x\_grid, y=y\_grid, z=matrix[i])  
  
 fig = graph\_objs.Figure([surface])  
 plotly.offline.plot(fig, auto\_open=True)  
  
 time.sleep(0.5)  
  
  
solve(100, 100, 4000, build\_plot=True)

1. **Выводы**

Таким образом, в ходе выполнения лабораторной работы, был изучен метод разностных аппроксимаций для волнового уравнения методом сеток, составлен алгоритмы (и соответствующие реализации) для решения волнового уравнения методом сеток, применимым для организации вычислений на ПЭВМ. Также данные методы были закреплены на примере решения задач по нахождению колебаний тяги и тонкой пластины из волнового уравнения.