Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра Информатики

Дисциплина «Методы численного анализа»

**ОТЧЕТ**

к лабораторной работе №12

на тему:

**«Решение краевых задач методом разностных аппроксимаций»**

БГУИР 6-05-0612-02 005

|  |
| --- |
| Выполнила студент группы 353504  АНТОНОВА Лидия Сергеевна |
|  |
| (дата, подпись студента) |
| Проверил доцент каф. Информатики  АНИСИМОВ Владимир Яковлевич |
|  |
| (дата, подпись преподавателя) |

Минск 2025

# 1 ЦЕЛЬ ВЫПОЛНЕНИЯ ЗАДАНИЯ

1 изучить метод разностных аппроксимаций, составить алгоритм метода и программу их реализации, получить численное решение заданной краевой задачи;

2 составить алгоритм решения краевых задач указанными методами, применимыми для организации вычислений на ПЭВМ;

3 составить программу решения краевых задач по разработанному алгоритму;

4 выполнить тестовые примеры и проверить правильность работы программ.

**2 задание**

Изображение выглядит как текст, снимок экрана, Шрифт, чек

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, чек, Шрифт, белый

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, Шрифт, белый, линия

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Контент, сгенерированный ИИ, может содержать ошибки.

# 3 Выполнение работы

**Задача 1.**

**Методика решения**

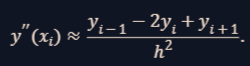
1 Дискретизация области определения. Область [−1,1] делится на N=40 равномерных отрезков с шагом:

h=2/N.

Узлы сетки задаются следующим образом:



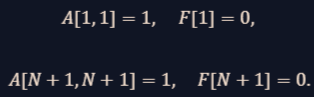
2 Составление разностной схемы. Для внутренних узлов применяется аппроксимация производной второго порядка:



Подставляя это выражение в дифференциальное уравнение, получаем:



3 Граничные условия. Граничные условия y(−1)=0 и y(1)=0 накладываются непосредственно:



4 Численное решение. Система линейных алгебраических уравнений решается с использованием метода Гаусса (процедура LinearSolve).

5 Визуализация. График полученного численного решения строится с использованием пакета plots. Точки отображаются в виде красных точек.

**Код реализации:**

with(LinearAlgebra);

with(plots);

k := 1;

a := sin(k);

b := cos(k);

eq1 := y(-1) = 0;

eq2 := y(1) = 0;

N := 40;

h := 2/N;

x := Vector(N + 1, i -> -1 + (i - 1)\*h);

A := Matrix(N + 1, N + 1, 0);

F := Vector(N + 1, -1);

A[1, 1] := 1;

A[N + 1, N + 1] := 1;

F[1] := 0;

F[N + 1] := 0;

for i from 2 to N do

xi := x[i];

A[i, i - 1] := a/h^2;

A[i, i] := -2\*a/h^2 + 1 + b\*xi^2;

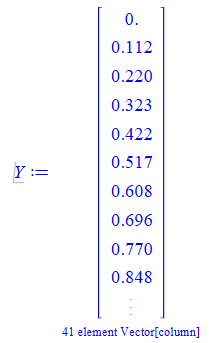
A[i, i + 1] := a/h^2;

end do;

Y := evalf[3](LinearSolve(A, F));

plot([seq([x[i], Y[i]], i = 1 .. N + 1)], style = point, symbol = solidcircle, color = red);

Протестируем данный метод на исходной функции, на рисунке 1 можно увидеть результат.



Изображение выглядит как диаграмма, График, линия, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1

**Задача 2.**

**Методика решения**

1 Дискретизация области определения. Область [0,2] разбивается на N=⌊(b−a)/h⌋ равномерных сегментов. Координаты узлов задаются:



2 Составление разностной схемы. Для внутренней точки xi разностное представление уравнения имеет вид:



После упрощения члены системы записываются в виде:

Изображение выглядит как Шрифт, текст, снимок экрана, типография

Контент, сгенерированный ИИ, может содержать ошибки.

3 Граничные условия. Граничные условия накладываются в явном виде:

A[1,1]=1, F[1]=ua,

A[N+1, N+1]=1, F[N+1]=ub.

4 Решение системы линейных уравнений. Полученная система решается с использованием функции LinearSolve, которая возвращает численные значения u(x)u(x) в узлах сетки.

5. Визуализация решения График решения строится с использованием пакета plots. Функция u(x)u(x) представлена гладкой линией.

**Код реализации:**

a := 0;

b := 2;

h := 0.02;

N := floor((b - a)/h);

p := x -> 0.5 + sin(x)^2;

q := x -> 2 + 2\*x^2;

f := x -> 10 + 10\*sin(x)^2;

u\_a := 0:

u\_b := 8:

A := Matrix(N + 1, N + 1, 0);

F := Vector(N + 1, 0);

A[1, 1] := 1;

A[N + 1, N + 1] := 1;

F[1] := u\_a;

F[N + 1] := u\_b;

for i from 2 to N do

xi := x[i];

A[i, i - 1] := 1/h^2 - p(xi)/(2\*h);

A[i, i] := -2/h^2 + q(xi);

A[i, i + 1] := 1/h^2 + p(xi)/(2\*h);

F[i] := f(xi);

end do;

Y := evalf[3](LinearSolve(A, F));

plot([seq([x[i], Y[i]], i = 1 .. N + 1)], style = line, color = purple, thickness = 2, labels = ["x", "u(x)"], title = "Численное решение краевой задачи");

Протестируем данный метод на исходной функции, на рисунке 2 можно увидеть результат.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

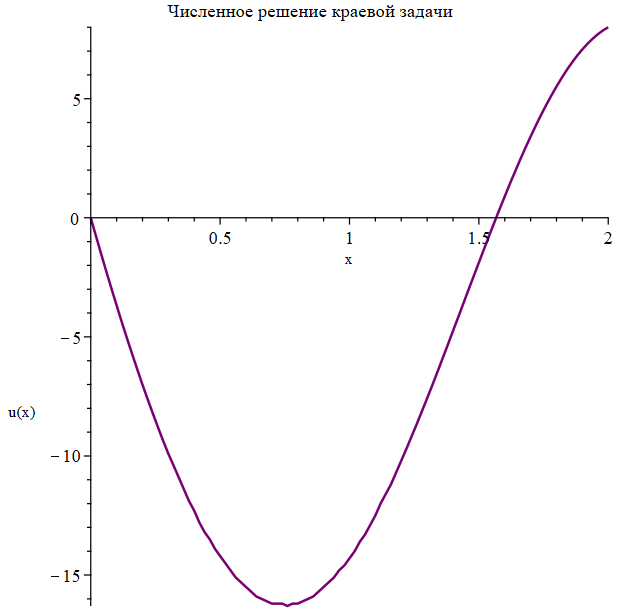


Рисунок 2

**Задача 3.**

**Методика решения**

1 Дискретизация области определения. Область [0.9,2.9] делится на N=⌊(b−a)/h⌋ равномерных сегментов с шагом h=ϵ/10. Координаты узлов вычисляются по формуле:



2 Граничные условия. Для учета граничных условий разностная схема модифицируется:

A[1,1]=1+0.2/2h, A[1,2]=−0.2/2h, F[1]=0.2,

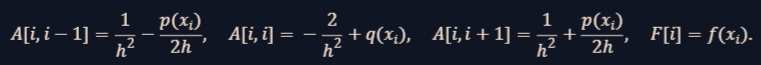
A[N+1,N+1]=1,F[N+1]=4.

3 Разностная схема для внутренних узлов. Разностное представление системы уравнений:

Изображение выглядит как Шрифт, снимок экрана, типография

Контент, сгенерированный ИИ, может содержать ошибки.

Для внутренних узлов i=2,…,N:



4 Численное решение. Полученная система уравнений решается с использованием метода прогонки (встроенной функции LinearSolve).

5 Визуализация решения. График решения строится для отображения зависимости u(x) от x.

**Код реализации:**

a := 0.9;

b := 2.9;

epsilon := 0.04;

h := epsilon/10;

N := floor((b - a)/h);

x := Vector(N + 1, i -> a + (i - 1)\*h);

A := Matrix(N + 1, N + 1, 0);

F := Vector(N + 1, 0);

A[1, 1] := 1 + 0.2/(2\*h);

A[1, 2] := -0.2/(2\*h);

F[1] := 0.2;

A[N + 1, N + 1] := 1;

F[N + 1] := 4;

for i from 2 to N do

xi := x[i];

A[i, i - 1] := 1/h^2 - p(xi)/(2\*h);

A[i, i] := -2/h^2 + q(xi);

A[i, i + 1] := 1/h^2 + p(xi)/(2\*h);

F[i] := f(xi);

end do;

Y := evalf[2](LinearSolve(A, F));

plot([seq([x[i], Y[i]], i = 1 .. N + 1)], style = line, color = green, thickness = 2, labels = ["x", "u(x)"], title = "Численное решение краевой задачи");

Протестируем данный метод на исходной функции, на рисунке 3 можно увидеть результат.

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

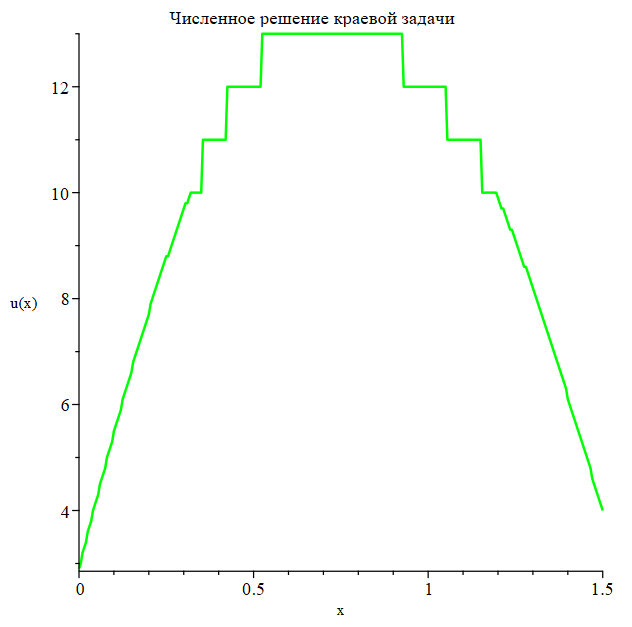


Рисунок 3

**Задача 4.**

**Методика решения**

1 Дискретизация области определения. Область [a,b]=[0,1.5] делится на N=⌊(b−a)/h⌋ равномерных отрезков с шагом h. Координаты узлов вычисляются следующим образом:



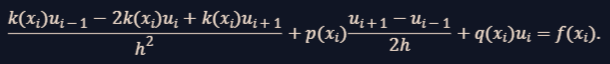
2 Учёт граничных условий (условия Ньюмана). Для левого края x=a:

A[1,1]=0.5+1/2h, A[1,2]=−1/2h, F[1]=0.

Для правого края x=b:

A[N+1,N]=−1/h, A[N+1,N+1]=0.5+1/h, F[N+1]=0.

3 Разностная схема для внутренних узлов. Для xi, i=2,…,N, разностное представление принимает вид:



Это преобразуется в коэффициенты матрицы:



Изображение выглядит как Шрифт, снимок экрана, линия, текст

Контент, сгенерированный ИИ, может содержать ошибки.

4 Решение системы линейных уравнений. Для получения решения u(x)u(x) используется метод прогонки через функцию LinearSolve.

5 Визуализация решения. Построен график численного решения u(x)u(x) для всей области определения.

**Код реализации:**

a := 0;

b := 1.5;

c := 1.125;

epsilon := 0.05;

h := epsilon/10;

N := floor((b - a)/h);

k := x -> piecewise(a < x and x < c, 0.5, c < x and x < b, 1.4);

q := x -> piecewise(a < x and x < c, 3.2, c < x and x < b, 8.5);

f := x -> 8\*x^2\*(2 - x);

x := Vector(N + 1, i -> a + (i - 1)\*h);

A := Matrix(N + 1, N + 1, 0);

F := Vector(N + 1, 0);

A[1, 1] := 0.5 + 1/(2\*h);

A[1, 2] := -1/(2\*h);

F[1] := 0;

A[N + 1, N] := -1/h;

A[N + 1, N + 1] := 0.5 + 1/h;

F[N + 1] := 0;

for i from 2 to N do

xi := x[i];

A[i, i - 1] := k(xi)/h^2 - p(xi)/(2\*h);

A[i, i] := -2\*k(xi)/h^2 + q(xi);

A[i, i + 1] := k(xi)/h^2 + p(xi)/(2\*h);

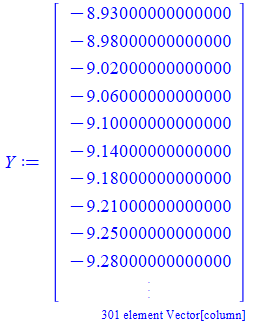
F[i] := f(xi);

end do;

Y := evalf[3](LinearSolve(A, F));

plot([seq([x[i], Y[i]], i = 1 .. N + 1)], style = line, color = blue, thickness = 2, labels = ["x", "u(x)"], title = "Численное решение краевой задачи 2.3.4 с условиями Ньюмана");

Протестируем данный метод на исходной функции, на рисунке 4 можно увидеть результат.



Изображение выглядит как текст, линия, диаграмма, График

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4

# Вывод

В рамках выполнения заданий рассмотрены численные методы решения краевых задач второго порядка, включая условия Ньюмана и граничные условия первого рода. Для каждого задания применялись метод конечных разностей и метод прогонки для численного решения систем уравнений, полученных в результате дискретизации. Также для всех задач построены графики численных решений, которые наглядно демонстрируют поведение функции u(x)u(x) в заданных областях. Это не только подтверждает правильность вычислений, но и позволяет визуально оценить сходимость и точность метода.