**Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования**

**"Уфимский государственный авиационный технический университет"**

**Кафедра** Высокопроизводительных вычислительных технологий и систем

**Дисциплина:** Теория разностных схем.

**Отчет по лабораторной работе № 4**

**Тема:** «Решение краевых задач  
для уравнений гиперболического типа»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Группа МКН-313 | Фамилия И.О. | Подпись | Дата | Оценка |
| Студент | Павлов Р.Е. |  |  |  |
| Принял | Ямилева А.М. |  |  |  |

**Уфа 2020**

**Цель работы:** получить навык численного решения краевых задач для уравнений гиперболического типа на примере начально-краевой задачи для линейного одномерного уравнения переноса и линейного одномерного неоднородного волнового уравнения.

**Теоретический материал**

**I. Начально-краевая задача для уравнения переноса**

Рассматривается простейшая линейная одномерная задача для уравнения переноса:

|  |  |
| --- | --- |
|  | (1) |
|  | (2) |
|  | (3) |

Параметры задачи выбираются в соответствии с индивидуальным заданием (Таблица 1). Начальное и граничные условия, а также функция восстанавливаются по заданному точному решению.

**Задача 1. Явная схема «левый уголок».**

При решении данной задачи (1)-(3) используется явная конечно-разностная схема с шаблоном «левый уголок» на равномерной пространственно-временной сетке:

Порядок аппроксимации данной схемы: . Схема устойчива при

**Задача 2. Неявная схема «левый уголок» (схема «бегущего счета»).**

При решении данной задачи (1)-(3) используется неявная конечно-разностная схема с шаблоном «левый уголок» («схема бегущего счета») на равномерной пространственно-временной сетке:

Значение сеточной функции на верхнем временном слое *p+1* рассчитывается по ее значениям в точках верхнего и нижнего *p* временных слоев.

Порядок аппроксимации данной схемы: . Схема устойчива при любых соотношениях между шагами сетки

**II. Начально-краевая задача для волнового уравнения.**

Рассматривается начально-краевая задача для линейного одномерного волнового уравнения с источником:

|  |  |
| --- | --- |
|  | (5) |
|  | (6) |
|  | (7) |
|  | (8) |
|  | (9) |

Параметры задачи выбираются в соответствии с индивидуальным заданием (Таблица 2). Аналитическое решение задачи строится по формуле Даламбера для всей оси , функции вычисляются по аналитическому решению для заданного в таблице вида граничных условий.

**Задача 3. Явная схема «крест».**

При решении данной задачи (5)-(9) используется явная конечно-разностная схема с шаблоном «крест» на равномерной пространственно-временной сетке:

Порядок аппроксимации данной схемы: . Схема устойчива при

**Задача 4. Неявная схема с весами.**

При решении данной задачи (5)-(9) используется неявная разностная схема с весами на равномерной пространственно-временной сетке:

чтобы все веса были неотрицательны, следует брать

Схема безусловно устойчива приЕсли то схема условно устойчива при Таким образом, при выборе веса неявная схема безусловно сходится с точностью .

**Задача 5. Схема повышенного порядка аппроксимации.**

При решении данной задачи (5)-(9) аналогично *Задаче 4* используется неявная разностная схема с весами на равномерной пространственно-временной сетке:

чтобы все веса были неотрицательны, следует брать Для повышения порядка точности следует брать

**Задания на лабораторную работу**

***I. Начально-краевая задача для уравнения переноса***

Рассматривается простейшая линейная одномерная задача для уравнения переноса:

|  |  |
| --- | --- |
|  | (1) |
|  | (2) |
|  | (3) |

Параметры задачи выбираются в соответствии с индивидуальным заданием (Таблица 1). Начальное и граничные условия, а также функция восстанавливаются по заданному точному решению.

***Задача 1 (2 балла).***

1. Написать вычислительную программу на языке программирования C++ решения задачи (1)-(3) с использованием явной конечно-разностной схемы с шаблоном «левый уголок» на равномерной пространственно-временной сетке.
2. Непосредственными расчетами продемонстрировать условную устойчивость схемы и справедливость условия устойчивости.
3. Исследовать зависимость решения от величины шагов сетки по пространственной и временной переменным посредством сравнения с построенным аналитическим решением. Построить графики зависимости погрешности, оцениваемой в равномерной норме по пространственной переменной, от времени и шагов сетки.

***Задача 2 (2 балла).***

1. Написать вычислительную программу на языке программирования C++ решения задачи (1)-(3) с использованием неявной конечно-разностной схемы с шаблоном «левый уголок» (схема «бегущего счета») на равномерной пространственно-временной сетке.
2. Выполнить сравнение точности получаемого решения по двум схемам с использованием точного решения. Построить графики погрешностей как функций координат и времени, а также графики норм погрешностей как функций шагов сетки.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант |  |  |  |  | Точное решение |
| 7 | 0 | 2 | 10 | 1 |  |

***II. Начально-краевая задача для волнового уравнения***

Рассматривается начально-краевая задача для линейного одномерного волнового уравнения с источником:

|  |  |
| --- | --- |
|  | (5) |
|  | (6) |
|  | (7) |
|  | (8) |
|  | (9) |

Параметры задачи выбираются в соответствии с индивидуальным заданием (Таблица 2). Аналитическое решение задачи строится по формуле Даламбера для всей оси , функции вычисляются по аналитическому решению для заданного в таблице вида граничных условий.

***Задача 3 (2 балла).***

1. Написать вычислительную программу на языке программирования C++ решения задачи (5)-(9) с использованием явной разностной схемы (шаблон «крест») на равномерной пространственно-временной сетке.
2. Непосредственными расчетами продемонстрировать условную устойчивость схемы и справедливость условия устойчивости.
3. Исследовать зависимость решения от величины шагов сетки по пространственной и временной переменным посредством сравнения с построенным аналитическим решением. Построить графики погрешностей как функций координат и времени, а также графики норм погрешностей как функций шагов сетки.

***Задача 4 (2 балла).***

1. Написать вычислительную программу на языке программирования C++ решения задачи (5)-(9) с использованием неявной разностной схемы (T-образный шаблон) на равномерной пространственно-временной сетке.
2. Непосредственными расчетами продемонстрировать абсолютную устойчивость схемы (сравнением с явной схемой).
3. Исследовать зависимость решения от величины шагов сетки по пространственной и временной переменным посредством сравнения с построенным аналитическим решением. Построить графики погрешностей как функций координат и времени, а также графики норм погрешностей как функций шагов сетки.

***Задача 5 (2 балла).***

1. Написать вычислительную программу на языке программирования C++ решения задачи (5)-(9) с использованием неявной разностной схемы с весами (9-точечный шаблон) на равномерной пространственно-временной сетке.
2. Непосредственными расчетами продемонстрировать условную устойчивость схемы (сравнением с явной схемой).
3. Провести расчеты для веса, соответствующего повышенному порядку аппроксимации. Выполнить сравнение точности получаемого решения с использованием точного решения. Построить графики погрешностей как функций координат и времени, а также графики норм погрешностей как функций шагов сетки.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант |  |  |  |  | Граничные условия |
| 7 | *1* |  |  |  |  |

**Практическая реализация**

**Задача 1: Явная схема «левый уголок»**

Для решения задачи используется явная конечно-разностная схема с шаблоном «левый уголок» на равномерной пространственно-временной сетке.

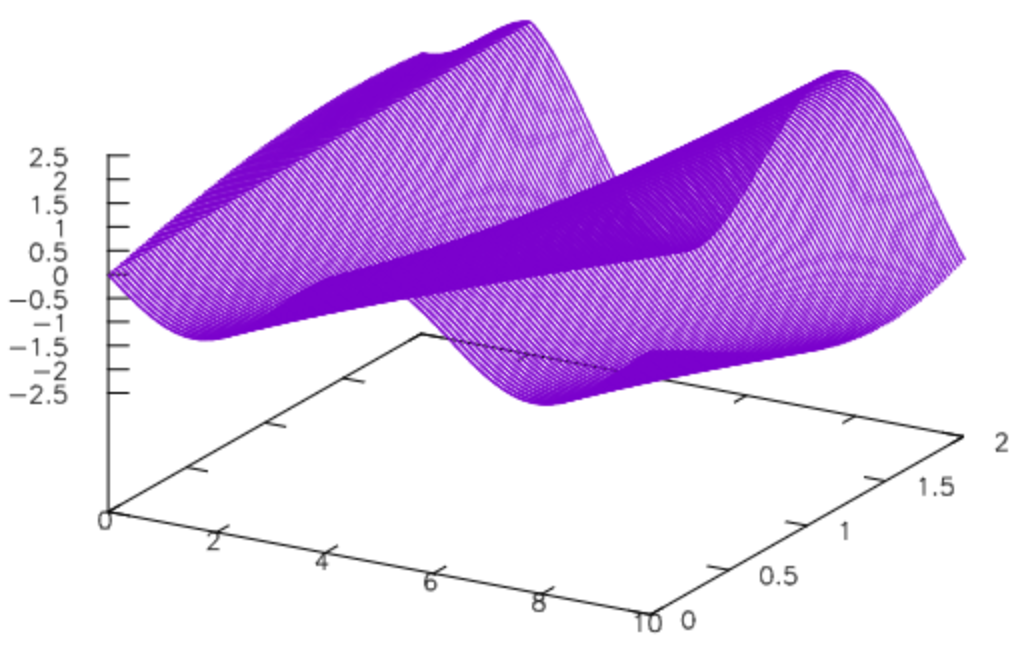


Рисунок 1. График численного решения при N=100, M=500

Построим график погрешности решения как разность между аналитическим и численным решением:

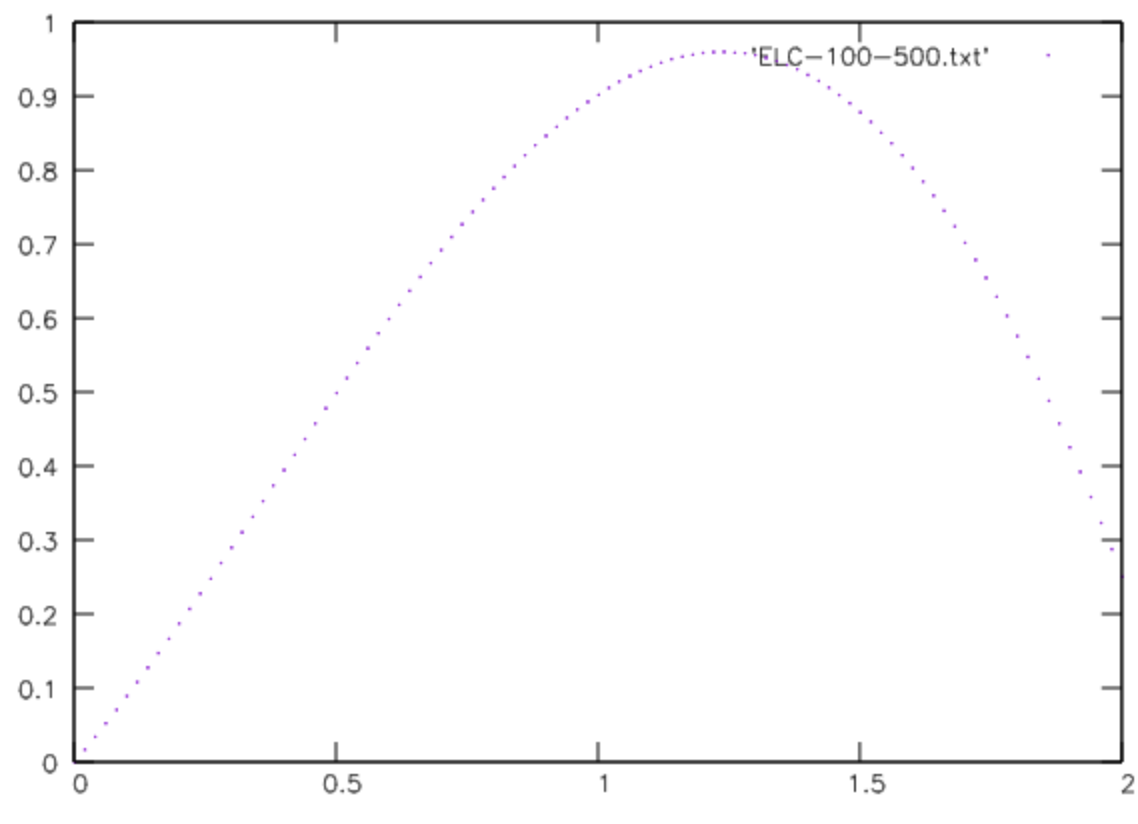


Рисунок 2. График погрешности численного решения

Нарушим условие устойчивости, приняв N=M=100, результат показан на рисунке 3.

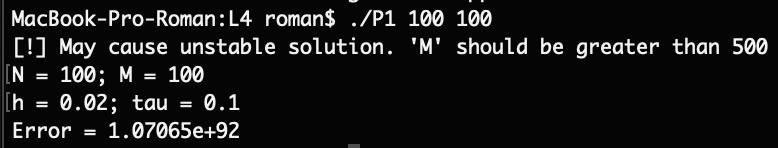


Рисунок 3. Пример работы программы при нарушении условия устойчивости

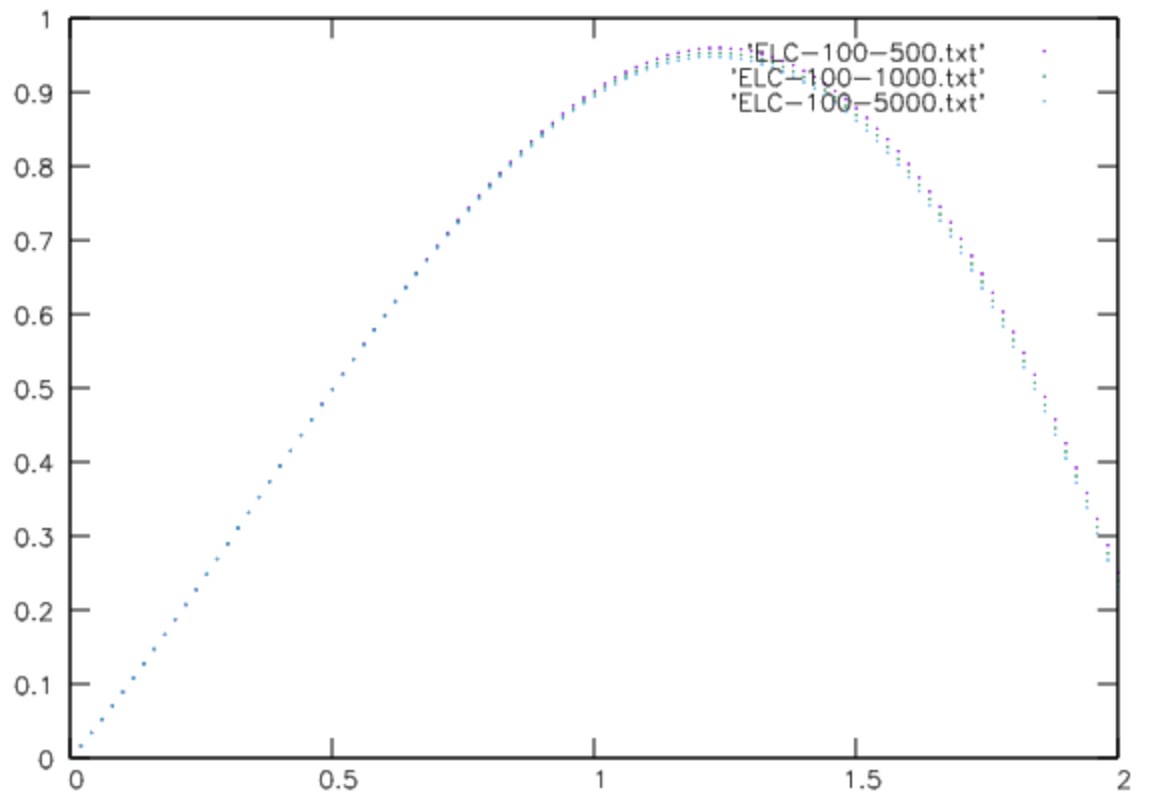


Рисунок 4. Зависимость погрешности от шага временной сетки

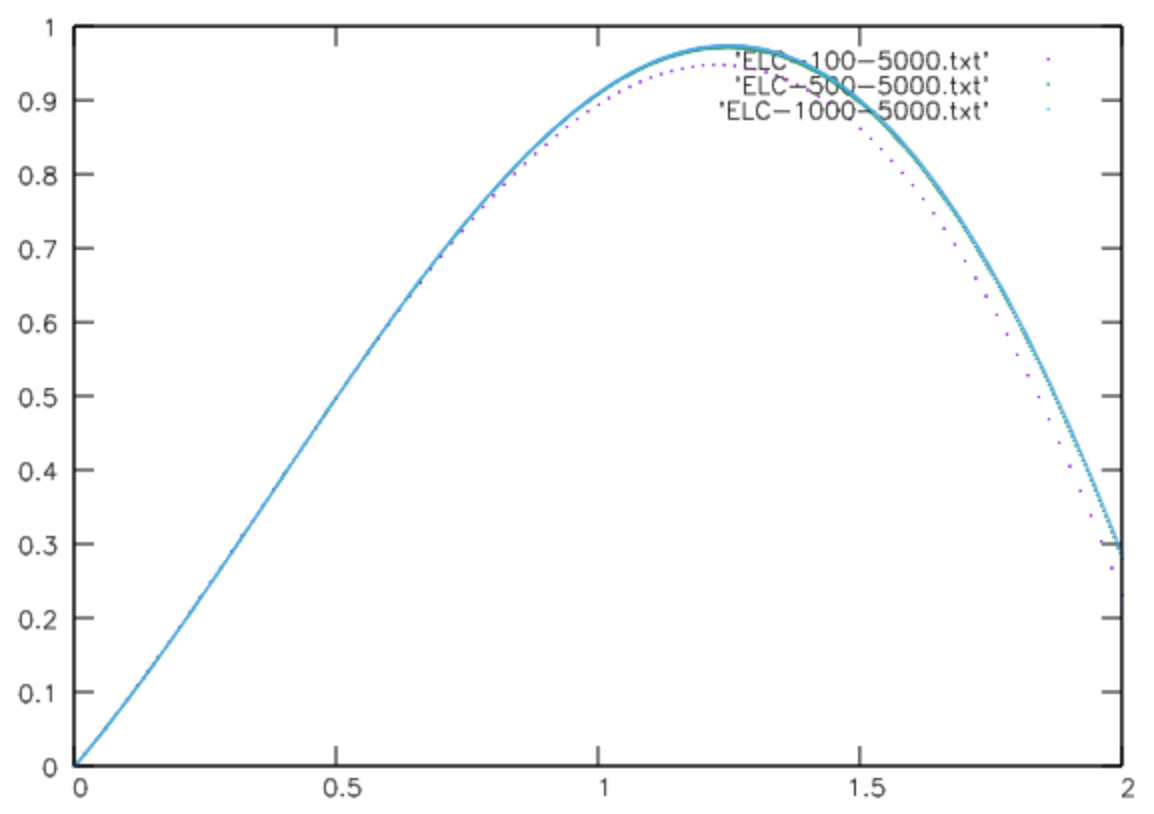


Рисунок 5. Зависимость погрешности решения от шага пространственной сетки

По графику можно сделать вывод, что с увеличением количества узлом сетки точность решения немного возрастает.

**Задача 2: Неявная схема «левый уголок»**

Для решения поставленной задачи используется неявная конечно-разностная схема с шаблоном «левый уголок» на равномерной пространственно-временной сетке.

Результат вычислений представлен на рисунке 6, по которому можно отметить, что результат близок к явной конечно-разностной схеме с таким же шаблоном.

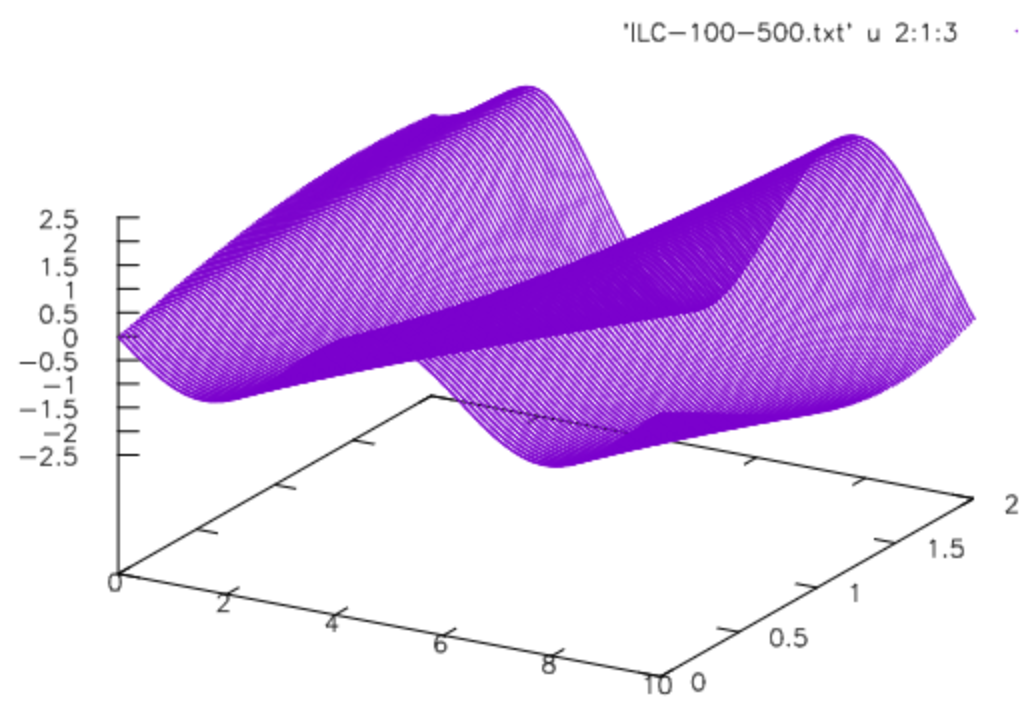


Рисунок 6. График численного решения при N=100, M=500

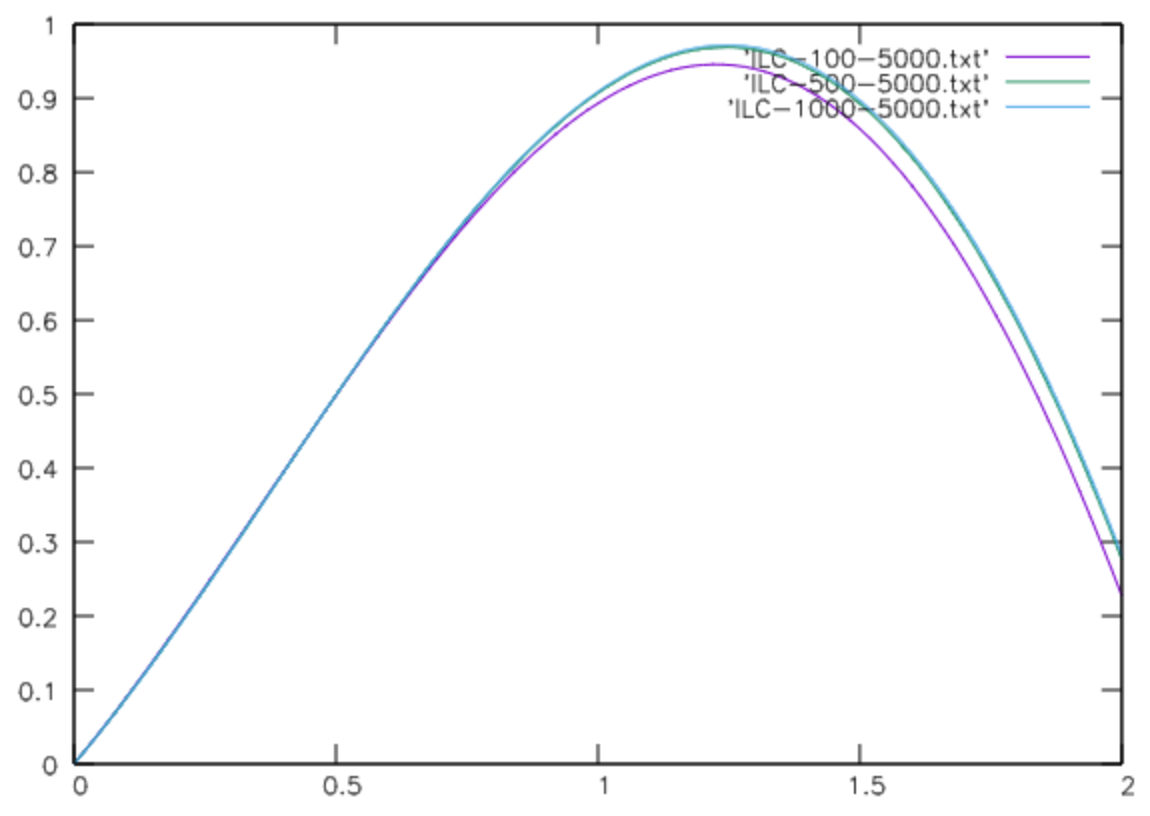


Рисунок 7. Зависимость погрешности решения от шага пространственной сетки

По графику зависимости погрешности от количества узлов пространственно-временной сетки можно сделать вывод, что с увеличением количества узлов точность немного повышается.

**Задача 3: Явная схема «крест»**

В этой задаче используется явная конечно-разностная схема с шаблоном «крест». Результат работы программы показан на рисунке 8.

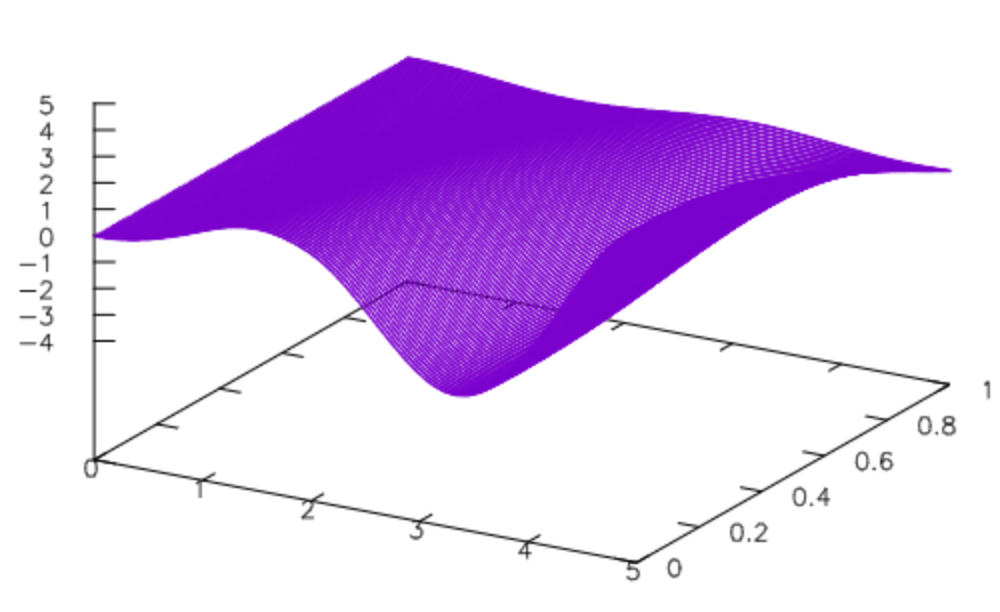


Рисунок 8. График численного решения при N=100, M=500

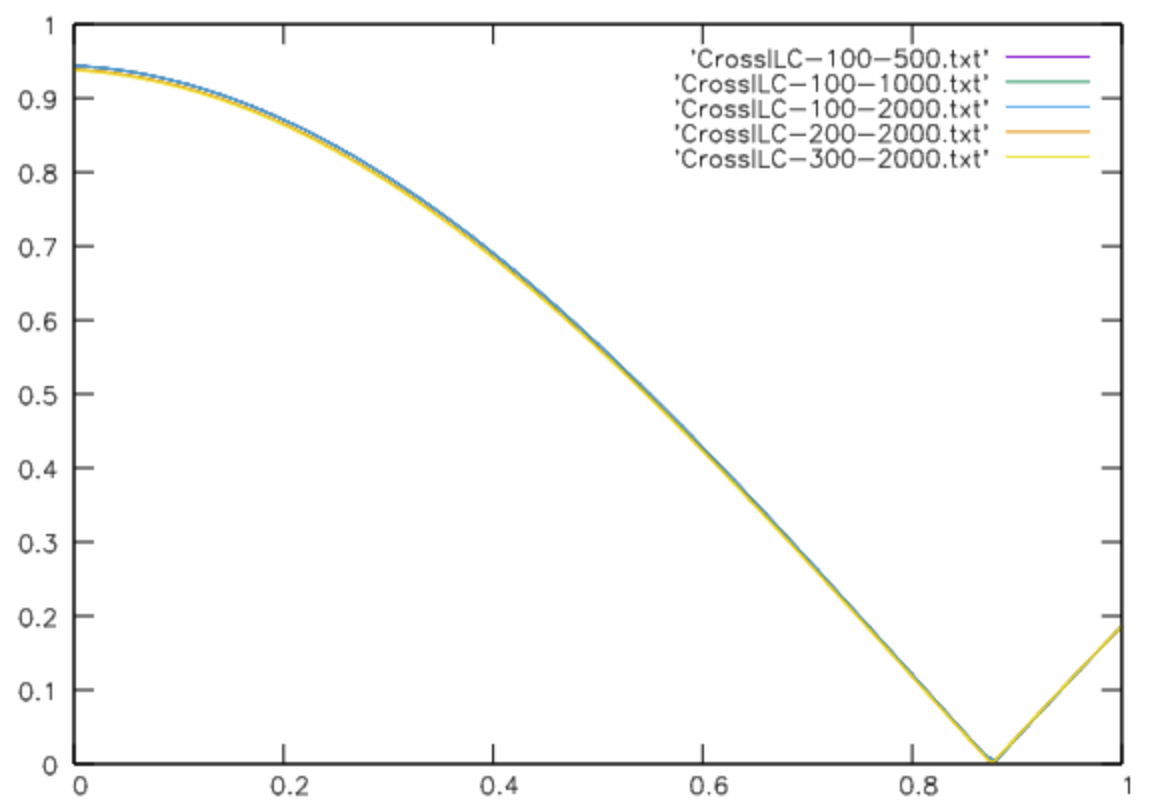


Рисунок 9. Зависимость погрешности от количества узлов сетки.

По графику на рисунке 9 можно сделать вывод, что с увеличением количества узлов как в пространственной, так и во временной сетке, точность повышается незначительно.

**Вывод**

В ходе лабораторной работы получен навык численного решения краевых задач для уравнений гиперболического типа на примере начально-краевой задачи для линейного одномерного уравнения переноса и линейного одномерного неоднородного волнового уравнения.

Для каждой поставленной задачи написана вычислительная программа на языке программирования С++, выполняющая необходимые построения и расчеты.

**Приложение**

Задача 1-2.

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <stdio.h>

#include <fstream>

#include <math.h>

#include <iomanip>

using namespace std;

const double C = 1.;

double phi(double x, double t) {

return (1/6) \* sin(5\*x) + sin(x);

}

double F(double x, double t) {

return (1/6) \* (cos(t+5\*x) - 6\*cos(t-x)) + cos(t-x) + (5/6)\*cos(t+5\*x);

}

double psi(double x, double t) {

return (1/6) \* sin(t) + sin(-t);

}

double ExplicitLeftCorner(int N, int M, double h, double tau) {

double \*X = new double[N + 1];

double \*T = new double[M + 1];

double \*\*u = new double\*[M + 1];

double gamma = tau / h;

for (int i = 0; i < N + 1; i++)

{

X[i] = i \* h;

}

for (int i = 0; i < M + 1; i++)

{

T[i] = i \* tau;

}

for (int t = 0; t < M + 1; t++) {

u[t] = new double[N + 1];

}

for (int i = 0; i < M + 1; i++)

{

for (int j = 0; j < N + 1; j++)

{

u[i][j] = 0.0;

}

}

for (int j = 0; j < N + 1; j++)

{

u[0][j] = phi(j\*h, 0);

}

for (int t = 0; t < M; t++)

{

for (int x = 1; x < N+1; x++)

{

u[t + 1][x] = (1. - gamma)\*u[t][x] + gamma \* u[t][x - 1] + F(X[x], T[t])\*tau;

}

u[t + 1][0] = psi(0, (t+1)\*tau);

}

double pogr = 0.;

ofstream fout("ELC-"+to\_string(N)+"-"+to\_string(M)+".txt");

for (int t = 0; t < M + 1; t++)

{

for (int j = 0; j < N + 1; j++)

{

double exact;

exact = (1/6)\*sin(T[t]+5\*X[j]) + sin(X[j]-T[t]);

double delta = abs(exact - u[t][j]);

if (delta >= pogr) {

pogr = delta;

}

if (t == M)

fout << X[j] << " " << delta << endl;

}

}

fout.close();

cout << "Error = " << pogr << endl;

return 0;

}

double ImplicitLeftCorner(int N, int M, double h, double tau) {

double \*X = new double[N + 1];

double \*T = new double[M + 1];

double \*\*u = new double\*[M + 1];

double gamma = tau / h;

for (int i = 0; i < N + 1; i++)

{

X[i] = i \* h;

}

for (int i = 0; i < M + 1; i++)

{

T[i] = i \* tau;

}

for (int t = 0; t < M + 1; t++) {

u[t] = new double[N + 1];

}

for (int i = 0; i < M + 1; i++)

{

for (int j = 0; j < N + 1; j++)

{

u[i][j] = 0.0;

}

}

for (int j = 0; j < N + 1; j++)

{

u[0][j] = phi(j\*h, 0);

}

double Func;

for (int t = 0; t < M; t++)

{

u[t + 1][0] = psi(0, (t + 1)\*tau);

for (int x = 1; x < N+1; x++)

{

u[t + 1][x] = (u[t][x] + (tau / h) \* u[t + 1][x - 1] + F(X[x], T[t+1])\*tau) / (1. + tau / h);

}

u[t + 1][0] = psi(N, (t+1)\*tau);

}

double pogr = 0;

ofstream fout("ILC-"+to\_string(N)+"-"+to\_string(M)+".txt");

for (int t = 0; t < M + 1; t++)

{

for (int j = 0; j < N + 1; j++)

{

double exact;

exact = (1/6)\*sin(T[t]+5\*X[j]) + sin(X[j]-T[t]);

double delta = abs(exact - u[t][j]);

if (delta >= pogr) {

pogr = delta;

}

if (t == M)

fout << X[j] << " " << T[t] << " " << u[t][j] << endl;

}

}

fout.close();

cout << "Error = " << pogr << endl;

return 0;

}

int main(int argc, char\* argv[])

{

int N, M;

double h, tau;

double a = 0., b = 2., T = 10.;

if (argc == 3)

{

N = atoi(argv[1]);

M = atoi(argv[2]);

}

else

{

N = 100;

M = 500;

}

h = (double)((b - a) / double(N));

tau = (double)T / (double)M;

if (tau/h > 1) {

cout << "[!] May cause unstable solution. 'M' should be greater than " << (int)(T/h) << endl;

//return 0;

}

cout << "N = " << N << "; M = " << M << endl;

cout << "h = " << h << "; tau = " << tau << endl;

//ExplicitLeftCorner(N, M, h, tau);

ImplicitLeftCorner(N, M, h, tau);

return 0;

}

Задача 3

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <stdio.h>

#include <fstream>

#include <math.h>

#include <iomanip>

#define eps 0.01

using namespace std;

double phi(double x, double t) {

return 0.;

}

double F(double x, double t) {

return sin(2\*t);

}

double uExact(double x, double t)

{

return -(sin(2\*x + 2\*t) + sin(2\*x - 2\*t) - 2\*sin(2\*x)) / 8;

}

double psi0(double x, double t)

{

return -sin(2\*t);

}

double psi1(double x, double t)

{

return cos(2) \* sin(t) \* sin(t);

}

double ExplicitCross(int N, int M, double h, double tau) {

double \*X = new double[N + 1];

double \*T = new double[M + 1];

double \*\*u = new double\*[M + 1];

for (int i = 0; i < N + 1; i++)

{

X[i] = i \* h;

}

for (int i = 0; i < M + 1; i++)

{

T[i] = i \* tau;

u[i] = new double[N + 1];

}

for (int i = 0; i < M + 1; i++)

{

for (int j = 0; j < N + 1; j++)

{

u[i][j] = 0.0;

}

}

for (int j = 0; j < N + 1; j++)

{

u[0][j] = phi(X[j], 0);

u[M][j] = phi(X[j], 0);

}

for (int t = 1; t < M; t++)

{

for (int x = 1; x < N+1; x++)

{

u[t+1][x] = 2\*u[t][x] - u[t-1][x] + tau\*tau \* (u[t][x-1]/h/h - 2 \* u[t][x]/h/h + u[t][x+1]/h/h + F(X[x], T[t]));

}

u[t + 1][0] = -h \* psi0(0, (t + 1)\*tau) + u[t+1][1];

u[t + 1][N] = psi1(N, (t + 1)\*tau);

}

double pogr = 0;

ofstream fout("CrossILC-"+to\_string(N)+"-"+to\_string(M)+".txt");

for (int t = 0; t < M + 1; t++)

{

for (int j = 0; j < N + 1; j++)

{

fout << X[j] << " " << T[t] << " " << u[t][j] << endl;

}

}

fout.close();

return 0;

}

int main(int argc, char\* argv[])

{

int N, M;

double h, tau;

double a = 0., b = 1., T = 5.;

if (argc == 3)

{

N = atoi(argv[1]);

M = atoi(argv[2]);

}

else

{

N = 100;

M = 500;

}

h = (double)((b - a) / double(N));

tau = (double)T / (double)M;

if (tau/h > 1) {

cout << "[!] May cause unstable solution. 'M' should be greater than " << (int)(T/h) << endl;

//return 0;

}

cout << "N = " << N << "; M = " << M << endl;

cout << "h = " << h << "; tau = " << tau << endl;

ExplicitCross(N, M, h, tau);

return 0;

}