|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| **Федеральное государственное бюджетное  образовательное учреждение высшего образования**  **«Новосибирский государственный технический университет»** | | | |
|  | | | |
| **Кафедра прикладной математики** | | | |
|  | | | |
|  | | | |
| **Курсовой проект по курсу** | | | |
| **«Численные методы»** | | | |
|  | | | |
|  | **Группа** | **ПМ-24** |  |
|  |  |  |
| **Студент** | **Марченко В.В.** |  |
|  |  |  |
|  |  |  |
|  |  |  |  |
| **Новосибирск** | | | |

**2024**

1. Формулировка задания

МКЭ для параболического уравнения в декартовой системе координат, трехслойная неявная схема для аппроксимации по времени, базисные функции линейные на треугольниках.

1. **Постановка задачи**

Параболическая краевая задача для функции  определяется дифференциальным уравнением

.

заданным в некоторой области  с границей  и краевыми условиями







в которых  - значение искомой функции  на границе , а  - значение на  производной функции  по направлению внешненормали к поверхности , .

1. Теоретическая часть
2. Дискретизация по времени и вариационная постановка:

Представим искомое решение  на интервале  с помощью аппроксимации по времени через квадратичные полиномы Лагранжа:



Здесь функции пространственных координат ,  и  являются значениями искомой функции  на временных слоях ,  и  соответственно.

Функции ,  и  – это базисные квадратичные полиномы Лагранжа (с двумя корнями из набора значений времен ,  и ), которые могут быть записаны в виде







где , , .

Подставим выражение, аппроксимирующее, для аппроксимации производной по времени параболического уравнения на временном слое :



Вычислим производные по  функций  при :







С учетом этого перепишем уравнение в виде



Выполним конечноэлементную аппроксимацию краевой задачи для получившегося уравнения. При выполнении вариационной постановки Галёркина получим:





где принимая ,  и  получаем



где  - матрица массы,  - матрица жесткости,  - матрица вкладов от краевых условий третьего рода, - вектор правой части, построенный по значениям функции  и параметром . и  краевых условий на текущем временном (-м) слое,  и  - решение на двух предыдущих слоях по времени,  - искомое решение на текущем слое по времени.

1. Переход к локальным матрицам:

Введем локальную нумерацию узлов на каждом треугольнике от 1 до 3, а также линейные базисные функции, которые будут равны единице в соответствующем узле и нулю во всех других:



При дальнейшем решении задачи будем использовать соотношения:

 (\*)

где - это площадь треугольника,  - матрица координат его вершин.

Учитывая построение - функций, получаем следующие соотношения:



Т.е. имеем систему:

Отсюда находим коэффициенты линейных функций 

Матрица жесткости

Разложили по линейным базисам

Матрица массы

=

Глобальная матрица собирается путем добавления к ней локальных матриц в соответствующие места элементов. Так как ij-ый элемент матрицы не равен нулю, только если произведение градиентов и самих функций не равно нулю (или хотя бы одно из двух произведений), поэтому матрица будет содержать много нулевых элементов и будет разреженной, из-за чего хранить ее будем в разреженной формате. Профиль матрицы формируется следующим образом: , если  и  узлы находятся в одном локальном элементе, поэтому пройдясь по всем конечным элементам, собираем номера глобальных функций, произведение которых не равно нулю в списке множеств. После этого по полученному списку множеств строим массивы, которые определяют профиль разреженной матрицы.

1. Учет краевых условий
2. Учет первых краевых условий

Для учета первых краевых условий, в глобальной матрице и глобальном векторе находим соответствующую глобальному номеру краевого узла строку, и ставим вместо диагонального элемента глобальной матрицы на этой строке «большое число», а вместо элемента с таким номером в вектор правой части ­­- «большое число», умноженное на значение краевого условия, заданное в исходной задаче.

1. Учет вторых и третьих краевых условий

Рассмотрим краевые условия второго и третьего рода

Отсюда получаем, что для учета краевых условий необходимо вычислить интегралы:

Краевые условия второго и третьего рода задаются на ребрах, т.е. определяются двумя узлами, лежащими на ребре.

Будем считать, что параметр  на постоянен, тогда параметр  будем раскладывать по двум базисным функциям, определенным на этом ребре.

, где - локально занумерованные линейные базисные функции, которые имеют также свои глобальные номера во всей расчетной области, а - значения функции  в узлах ребра.

Аналогично поступаем и при учете вторых краевых условий, раскладывая по базису ребра функцию 

Тогда приведенные выше интегралы примут вид:

Фактически, решая задачу учета краевых условий второго и третьего рода, мы переходим к решению одномерной задачи на ребре для того, чтобы занести соответствующие результаты в глобальную матрицу и вектор.

Базисными функциями ребра являются две ненулевые на данном ребре базисные функции из  конечного элемента.

Для учета вклада вторых и третьих краевых условий рассчитываются 2 матрицы .

Интегралы  будем вычислять по формуле , где  длина ребра. При этом независимо от того, что на каждом из ребер присутствуют свои базисные функции, интегралы, посчитанные по приведенным выше формулам, будут равны.

Этот вектор поправок в правую часть позволяет учесть не только вторые краевые условия, но и часть  из третьих.

Осталось рассмотреть матрицу поправок в левую часть.

Очевидно, что получится та же матрица, только не умноженная на вектор констант.



Добавляя эту матрицу в левую часть, на места соответствующие номерам узлов, получаем учет третьих краевых условий.

При расчете  и должно учитываться направление нормали .

Если рассматривать нормаль к наклонной стороне области, то для каждой из двух точек ребра, в которых рассматриваются нормали, значения производных решения по обеим координатам будет ненулевыми, если, производная самой функции по какой-либо координате не будет нулевой.

1. Описание разработанных программ.

Глобальные переменные:

double\* z, \* r, \* p, \* t, \* r1, \* l, \* l1, \* f;: Векторы, используемые для хранения различных промежуточных значений и результатов.

double\*\* tch: Двумерный массив для хранения координат точек (узлов).

double\* tr: Массив для хранения треугольников.

int\* k1, \* k2, \* k3: Массивы для хранения дополнительных данных (например, границ).

double\* F: Вектор правой части системы уравнений.

int\* ig, \* jg: Массивы для представления разреженной матрицы (индексы строк и столбцов).

double\* x, \* di, \* ggl, \* ggu: Векторы для хранения данных, связанных с разреженной матрицей.

int n, n2, n3, n4, n5: Переменные для хранения размеров различных массивов.

bool kf2, kf3: Флаги для проверки наличия определенных данных.

Функция input():

Читает данные из файлов (toch.txt, tr.txt, kr\_1.txt, kr\_2.txt, kr\_3.txt).

Заполняет массивы точек, треугольников и других параметров.

Функции для вычислений:

double betta(int k): Возвращает коэффициент в зависимости от значения k.

double resh(double x, double y, int k): Вычисляет значение в зависимости от координат и параметра.

void tochnoe(): Заполняет вектор x значениями, вычисленными с помощью функции resh.

double func(...): Различные функции для вычисления значений в зависимости от координат и параметров.

void portret(): Строит разреженную матрицу на основе треугольников и точек.

Функции для работы с матрицами:

void M\_matrix(...): Формирует матрицу масс для конечных элементов.

void G\_matrix(...): Формирует матрицу жесткости для конечных элементов.

void local\_matrix(...): Собирает локальную матрицу для текущего конечного элемента.

Функции для добавления значений в матрицы:

void addToElement(int row, int col, double value): Добавляет значение к элементу разреженной матрицы.

void zeroOutRow(int row): Обнуляет значения в указанной строке разреженной матрицы.

Функции для глобальной матрицы:

void global\_matrix(): Формирует глобальную матрицу на основе локальных матриц и обновляет вектор правой части.

void LOC(): Реализует метод локальной оптимальной схемы для решения системы уравнений.

1. Тестирования программ.

**Тест №1**

|  |  |
| --- | --- |
|  | **Точки:** 0 0 2 0  4 0  1 1  3 1  0 2  2 2  4 2  1 3  3 3  0 4  2 4  4 4  Условия:  λ=2;σ=3  u(x,y,t)=x2+y2  f(x,y,t)=-8  КУ-1:1, 2, 3, 8, 13, 12, 11, 6  t∈{0,1,2,3,4,5} |

|  |  |  |
| --- | --- | --- |
| U | U\* | U-U\* |
| 2 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | 0 |
| 16 | 16 | 0 |
| 2 | 2,46023688663283 | -0,460236886632826 |
| 10 | 10,4602368866328 | -0,460236886632828 |
| 4 | 4 | 0 |
| 8 | 7,78341793570221 | 0,216582064297794 |
| 20 | 20 | 7,11E-15 |
| 10 | 10,4602368866328 | -0,460236886632828 |
| 18 | 18,4602368866328 | -0,460236886632835 |
| 16 | 16 | 0 |
| 20 | 20 | 7,11E-15 |
| 32 | 32 | 0 |
| 3 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | 0 |
| 16 | 16 | 0 |
| 2 | 2,61506924224335 | -0,615069242243353 |
| 10 | 10,6150692422434 | -0,615069242243354 |
| 4 | 4 | 0 |
| 8 | 7,84645027928803 | 0,153549720711974 |
| 20 | 20 | 7,11E-15 |
| 10 | 10,6150692422434 | -0,615069242243353 |
| 18 | 18,6150692422434 | -0,615069242243372 |
| 16 | 16 | 0 |
| 20 | 20 | 7,11E-15 |
| 32 | 32 | 0 |
| 4 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | -8,88E-16 |
| 16 | 16 | -3,55E-15 |
| 2 | 2,65052392788084 | -0,650523927880836 |
| 10 | 10,6505239278808 | -0,650523927880837 |
| 4 | 4 | -8,88E-16 |
| 8 | 7,93743519652224 | 0,0625648034777635 |
| 20 | 20 | -1,07E-14 |
| 10 | 10,6505239278808 | -0,650523927880835 |
| 18 | 18,6505239278808 | -0,650523927880837 |
| 16 | 16 | -3,55E-15 |
| 20 | 20 | -1,07E-14 |
| 32 | 32 | -7,11E-15 |
| 5 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | -1,78E-15 |
| 16 | 16 | -7,11E-15 |
| 2 | 2,66140031605635 | -0,661400316056349 |
| 10 | 10,6614003160564 | -0,661400316056351 |
| 4 | 4 | -1,78E-15 |
| 8 | 7,98337683784502 | 0,0166231621549819 |
| 20 | 20 | 3,55E-15 |
| 10 | 10,6614003160564 | -0,661400316056351 |
| 18 | 18,6614003160563 | -0,661400316056348 |
| 16 | 16 | -7,11E-15 |
| 20 | 20 | 3,55E-15 |
| 32 | 32 | -1,42E-14 |

**Тест №2**

C краевыми условиями 2 и 3 -го типа КУ-1:1, 2, 3, 8, 6 KУ-2:12, 13 КУ-3:11, 12

|  |  |  |
| --- | --- | --- |
| U | U\* | U-U\* |
| 2 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | 8,88E-16 |
| 16 | 16 | 3,55E-15 |
| 2 | 2,45043317787041 | -0,450433177870411 |
| 10 | 10,4504331778704 | -0,450433177870409 |
| 4 | 4 | 8,88E-16 |
| 8 | 7,69714529859294 | 0,302854701407059 |
| 20 | 20 | 7,11E-15 |
| 10 | 10,7330669280454 | -0,733066928045437 |
| 18 | 18,7330669280454 | -0,733066928045439 |
| 16 | 16 | 3,55E-15 |
| 20 | 22,4871770015403 | -2,48717700154026 |
| 32 | 32 | 7,11E-15 |
| 3 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | -8,88E-16 |
| 16 | 16 | -3,55E-15 |
| 2 | 2,6140285268177 | -0,614028526817704 |
| 10 | 10,6140285268177 | -0,614028526817705 |
| 4 | 4 | -8,88E-16 |
| 8 | 7,93768196126946 | 0,0623180387305453 |
| 20 | 20 | 7,11E-15 |
| 10 | 11,39725502595 | -1,39725502595003 |
| 18 | 19,39725502595 | -1,39725502595003 |
| 16 | 16 | -3,55E-15 |
| 20 | 23,9982235905721 | -3,99822359057213 |
| 32 | 32 | -7,11E-15 |
| 4 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | 8,88E-16 |
| 16 | 16 | 3,55E-15 |
| 2 | 2,70363307730349 | -0,703633077303485 |
| 10 | 10,7036330773035 | -0,703633077303486 |
| 4 | 4 | 8,88E-16 |
| 8 | 8,3100431607867 | -0,310043160786702 |
| 20 | 20 | -7,11E-15 |
| 10 | 11,7998916723838 | -1,79989167238383 |
| 18 | 19,7998916723838 | -1,79989167238383 |
| 16 | 16 | 3,55E-15 |
| 20 | 24,665714367474 | -4,66571436747397 |
| 32 | 32 | 7,11E-15 |
| 5 |  |  |
| 0 | 0 | 0 |
| 4 | 4 | -8,88E-16 |
| 16 | 16 | -3,55E-15 |
| 2 | 2,78102955778431 | -0,781029557784308 |
| 10 | 10,7810295577843 | -0,781029557784304 |
| 4 | 4 | -8,88E-16 |
| 8 | 8,5854922795428 | -0,585492279542796 |
| 20 | 20 | 0 |
| 10 | 12,0025539258764 | -2,0025539258764 |
| 18 | 20,0025539258764 | -2,0025539258764 |
| 16 | 16 | -3,55E-15 |
| 20 | 24,9350413583943 | -4,93504135839434 |
| 32 | 32 | -7,11E-15 |

***Тесты на порядок аппроксимации по времени:***

**Тест №3**

|  |  |  |  |
| --- | --- | --- | --- |
| **Первый порядок** | | λ=2;σ=3  u(x,y,t)=x2+y2+t  f(x,y,t)=-8  КУ-1:1, 2, 3, 8, 13, 12, 11, 6  t∈{0,1,2,3,4,5} | |
| U | U\* | | U-U\* |
| 2 |  | |  |
| 2 | 2 | | -8,88E-16 |
| 6 | 6 | | 4,44E-15 |
| 18 | 18 | | -1,07E-14 |
| 4 | 4,01353637901862 | | -0,0135363790186158 |
| 12 | 12,0135363790186 | | -0,0135363790186087 |
| 6 | 6 | | 4,44E-15 |
| 10 | 9,05245346869713 | | 0,947546531302875 |
| 22 | 22 | | 7,11E-15 |
| 12 | 12,0135363790186 | | -0,0135363790186087 |
| 20 | 20,0135363790186 | | -0,0135363790186211 |
| 18 | 18 | | -1,07E-14 |
| 22 | 22 | | 7,11E-15 |
| 34 | 34 | | 0 |
| 3 |  | |  |
| 3 | 3 | | -4,44E-16 |
| 7 | 7 | | -8,88E-16 |
| 19 | 19 | | 0 |
| 5 | 4,8652088146793 | | 0,134791185320702 |
| 13 | 12,8652088146793 | | 0,134791185320703 |
| 7 | 7 | | -8,88E-16 |
| 11 | 9,4618917146939 | | 1,5381082853061 |
| 23 | 23 | | 0 |
| 13 | 12,8652088146793 | | 0,134791185320703 |
| 21 | 20,8652088146793 | | 0,134791185320697 |
| 19 | 19 | | 0 |
| 23 | 23 | | 0 |
| 35 | 35 | | 0 |
| 4 |  | |  |
| 4 | 4 | | 1,78E-15 |
| 8 | 8 | | 3,55E-15 |
| 20 | 20 | | -7,11E-15 |
| 6 | 5,74512343997208 | | 0,254876560027923 |
| 14 | 13,7451234399721 | | 0,25487656002792 |
| 8 | 8 | | 3,55E-15 |
| 12 | 10,173557823313 | | 1,82644217668696 |
| 24 | 24 | | 1,42E-14 |
| 14 | 13,7451234399721 | | 0,25487656002792 |
| 22 | 21,7451234399721 | | 0,254876560027927 |
| 20 | 20 | | -7,11E-15 |
| 24 | 24 | | 1,42E-14 |
| 36 | 36 | | 0 |
| 5 |  | |  |
| 5 | 5 | | 1,78E-15 |
| 9 | 9 | | -3,55E-15 |
| 21 | 21 | | 0 |
| 7 | 6,68800089939619 | | 0,31199910060381 |
| 15 | 14,6880008993962 | | 0,311999100603812 |
| 9 | 9 | | -3,55E-15 |
| 13 | 11,0490453761409 | | 1,95095462385906 |
| 25 | 25 | | 7,11E-15 |
| 15 | 14,6880008993962 | | 0,311999100603812 |
| 23 | 22,6880008993962 | | 0,311999100603828 |
| 21 | 21 | | 0 |
| 25 | 25 | | 7,11E-15 |
| 37 | 37 | | -1,42E-14 |

**Тест №4**

|  |  |  |  |
| --- | --- | --- | --- |
| **Второго порядок** | | λ=2;σ=3  u(x,y,t)=x2+y2+t2  f(x,y,t)=6t-8  КУ-1:1, 2, 3, 8, 13, 12, 11, 6  t∈{0,1,2,3,4,5} | |
| U | U\* | | U-U\* |
| 2 |  | |  |
| 4 | 4 | | 4,44E-16 |
| 8 | 8 | | 8,88E-16 |
| 20 | 20 | | 0 |
| 6 | 6,46023688663283 | | -0,460236886632827 |
| 14 | 14,4602368866328 | | -0,460236886632824 |
| 8 | 8 | | 8,88E-16 |
| 12 | 11,7834179357022 | | 0,216582064297798 |
| 24 | 24 | | -7,11E-15 |
| 14 | 14,4602368866328 | | -0,460236886632824 |
| 22 | 22,4602368866328 | | -0,460236886632835 |
| 20 | 20 | | 0 |
| 24 | 24 | | -7,11E-15 |
| 36 | 36 | | 0 |
| 3 |  | |  |
| 9 | 9 | | 0 |
| 13 | 13 | | -3,55E-15 |
| 25 | 25 | | -3,55E-15 |
| 11 | 11,6150692422434 | | -0,61506924224336 |
| 19 | 19,6150692422434 | | -0,615069242243365 |
| 13 | 13 | | -3,55E-15 |
| 17 | 16,846450279288 | | 0,153549720711972 |
| 29 | 29 | | 1,78E-14 |
| 19 | 19,6150692422434 | | -0,615069242243365 |
| 27 | 27,6150692422434 | | -0,615069242243351 |
| 25 | 25 | | -3,55E-15 |
| 29 | 29 | | 1,78E-14 |
| 41 | 41 | | 1,42E-14 |
| 4 |  | |  |
| 16 | 16 | | 0 |
| 20 | 20 | | -7,11E-15 |
| 32 | 32 | | 0 |
| 18 | 18,6505239278808 | | -0,65052392788084 |
| 26 | 26,6505239278808 | | -0,65052392788083 |
| 20 | 20 | | -7,11E-15 |
| 24 | 23,9374351965222 | | 0,0625648034777804 |
| 36 | 36 | | 1,42E-14 |
| 26 | 26,6505239278808 | | -0,65052392788083 |
| 34 | 34,6505239278808 | | -0,650523927880812 |
| 32 | 32 | | 0 |
| 36 | 36 | | 1,42E-14 |
| 48 | 48 | | 2,13E-14 |
| 5 |  | |  |
| 25 | 25 | | 7,11E-15 |
| 29 | 29 | | 7,11E-15 |
| 41 | 41 | | 2,13E-14 |
| 27 | 27,6614003160563 | | -0,661400316056337 |
| 35 | 35,6614003160563 | | -0,661400316056323 |
| 29 | 29 | | 7,11E-15 |
| 33 | 32,983376837845 | | 0,016623162155021 |
| 45 | 45 | | -7,11E-15 |
| 35 | 35,6614003160563 | | -0,661400316056323 |
| 43 | 43,6614003160563 | | -0,661400316056344 |
| 41 | 41 | | 2,13E-14 |
| 45 | 45 | | -7,11E-15 |
| 57 | 57 | | 7,11E-15 |

**Тест №5**

|  |  |  |  |
| --- | --- | --- | --- |
| **Третьего порядок** | | λ=2;σ=3  u(x,y,t)=x2+y2+t3  ff(x,y,t)=9t2-8  КУ-1:1, 2, 3, 8, 13, 12, 11, 6  t∈{0,1,2,3,4,5} | |
| U | U\* | | U-U\* |
| 2 |  | |  |
| 8 | 8 | | 2,66E-15 |
| 12 | 12 | | -5,33E-15 |
| 24 | 24 | | -1,07E-14 |
| 10 | 11,3536379018612 | | -1,35363790186125 |
| 18 | 19,3536379018613 | | -1,35363790186126 |
| 12 | 12 | | -5,33E-15 |
| 16 | 17,2453468697124 | | -1,24534686971235 |
| 28 | 28 | | -1,42E-14 |
| 18 | 19,3536379018613 | | -1,35363790186126 |
| 26 | 27,3536379018613 | | -1,35363790186125 |
| 24 | 24 | | -1,07E-14 |
| 28 | 28 | | -1,42E-14 |
| 40 | 40 | | -7,11E-15 |
| 3 |  | |  |
| 27 | 27 | | 1,07E-14 |
| 31 | 31 | | 1,07E-14 |
| 43 | 43 | | -7,11E-15 |
| 29 | 31,1147900973715 | | -2,11479009737147 |
| 37 | 39,1147900973715 | | -2,11479009737145 |
| 31 | 31 | | 1,07E-14 |
| 35 | 37,6155674084763 | | -2,61556740847626 |
| 47 | 47 | | -1,42E-14 |
| 37 | 39,1147900973715 | | -2,11479009737145 |
| 45 | 47,1147900973715 | | -2,11479009737147 |
| 43 | 43 | | -7,11E-15 |
| 47 | 47 | | -1,42E-14 |
| 59 | 59 | | 0 |
| 4 |  | |  |
| 64 | 64 | | 0 |
| 68 | 68 | | 4,26E-14 |
| 80 | 80 | | -1,42E-14 |
| 66 | 68,4613249036983 | | -2,46132490369833 |
| 74 | 76,4613249036984 | | -2,46132490369835 |
| 68 | 68 | | 4,26E-14 |
| 72 | 75,4651899429406 | | -3,46518994294057 |
| 84 | 84 | | -1,42E-14 |
| 74 | 76,4613249036984 | | -2,46132490369835 |
| 82 | 84,4613249036984 | | -2,46132490369838 |
| 80 | 80 | | -1,42E-14 |
| 84 | 84 | | -1,42E-14 |
| 96 | 96 | | 1,42E-14 |
| 5 |  | |  |
| 125 | 125 | | 1,42E-14 |
| 129 | 129 | | 2,84E-14 |
| 141 | 141 | | -2,84E-14 |
| 127 | 129,608199149377 | | -2,60819914937662 |
| 135 | 137,608199149377 | | -2,60819914937667 |
| 129 | 129 | | 2,84E-14 |
| 133 | 136,852039761253 | | -3,85203976125308 |
| 145 | 145 | | 5,68E-14 |
| 135 | 137,608199149377 | | -2,60819914937667 |
| 143 | 145,608199149377 | | -2,6081991493767 |
| 141 | 141 | | -2,84E-14 |
| 145 | 145 | | 5,68E-14 |
| 157 | 157 | | 0 |

**Вывод:** на полиноме третьей степени по времени появилась существенная погрешность, которой не было на двух прошлых тестах, следовательно порядок аппроксимации по времени трехслойной явной схемы равен двум.

***Тесты на порядок сходимости:***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Сетка:** | | | **Условия:** | |
|  | | | λ=1;σ=1  u(x,y,t)=sin(t)  f(x,y,t)=cos(t)  КУ -I: 1, 2, 3, 8, 13, 12, 11, 6  t∈{0,…,0.01} | |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | T=5 | 1 | 0,5 | 0,25 | 0,125 | 0,0625 | | 1 | -1,11E-16 | 2,22E-16 | 2,22E-16 | 0 | 2,22E-16 | | 2 | -1,11E-16 | 2,22E-16 | 2,22E-16 | 0 | 2,22E-16 | | 3 | -1,11E-16 | 2,22E-16 | 2,22E-16 | 0 | 2,22E-16 | | 4 | -0,127542528283297 | -0,0260981794091402 | -0,00483165285363452 | -0,000955094300450599 | -0,00020602045395357 | | 5 | -0,127542528283297 | -0,0260981794091405 | -0,00483165285363452 | -0,000955094300450599 | -0,000206020453954237 | | 6 | -1,11E-16 | 2,22E-16 | 2,22E-16 | 0 | 2,22E-16 | | 7 | -0,262906590668712 | -0,0590615078050176 | -0,0117649296818712 | -0,00246790012145459 | -0,000554833921598052 | | 8 | -1,11E-16 | 2,22E-16 | 2,22E-16 | 0 | 2,22E-16 | | 9 | -0,127542528283297 | -0,0260981794091402 | -0,00483165285363429 | -0,000955094300450599 | -0,000206020453953792 | | 10 | -0,127542528283297 | -0,0260981794091405 | -0,00483165285363452 | -0,000955094300450377 | -0,000206020453955014 | | 11 | -1,11E-16 | 2,22E-16 | 2,22E-16 | 0 | 2,22E-16 | | 12 | -1,11E-16 | 2,22E-16 | 2,22E-16 | 0 | 2,22E-16 | | | | | |
| h | Среднее значение погрешности |  | | Порядок сходимости |
| 1 | -0,0595 | 4,72222222222222 | | 2,23946593469539 |
| 0,5 | -0,0126 | 5,27196652719665 | | 2,39834121040653 |
| 0,25 | -0,00239 | 4,93801652892562 | | 2,30393166559352 |
| 0,125 | -0,000484 | 4,56603773584906 | | 2,1909427827114 |
| 0,0625 | -0,000106 | - | | - |

**Вывод:** в результате тестирования получили, что при дроблении сетки по времени порядок сходимости трехслойной неявной схемы по времени стремится к двойке.

1. Текст программы

#include <iostream>

#include <fstream>

#include <iomanip>

#include <cmath>

#include <variant>

using namespace std;

double\* z, \* r, \* p, \* t, \* r1, \* l, \* l1, \* f, vr, tp, t0, t1, \* dt0v, \* dtv;//vr глобально время сделала чтобы не передавть

double\*\* tch = NULL; // Array for points

double\* tr = NULL; // Array for triangles

double\* q1 = NULL; // Array for triangles

double\* q0 = NULL; // Array for triangles

int\* k1 = NULL; // Array for k1

int\* k2 = NULL; // Array for k2

int\* k3 = NULL;

double\* T = NULL;

double\*\* M\_s3 = NULL;

double\* F = NULL;// Array for k3

int\* ig = NULL;

int\* jg = NULL;

double\* x = NULL;

double\* x22 = NULL;

double\* di3 = NULL;

double\* ggl3 = NULL;

double\* ggu3 = NULL;

double\* di = NULL;

double\* ggl = NULL;

double\* ggu = NULL;

double\* di2 = NULL;

double\* ggl2 = NULL;

double\* ggu2 = NULL;

int n, n2, n3, n4, n5, n6;

double u0(double x, double y, int k) {//только для q0 и q1

switch (k)

{

//case 0: return sin(T[0]);//

case 0: return x \* x + y \* y + T[0] ;

// case 1: return sin(T[1]); //

case 1: return x \* x + y \* y + T[1] ;

}

return 0;

}

double betta(int k) {

return 2;

}

double ooo(double x, double y, int k) {

return 4\*y ;

}

void tochnoe()//u

{

for (int i = 0; i < n; i++)

x[i] = /\*sin(vr);\*/tch[i][0] \* tch[i][0] + tch[i][1] \* tch[i][1] + vr;//lkz 1-без время ,2

}

double gamma(double x, double y) {//сигма

return 3;

}

double lambda(double x, double y, int i) {

//switch (i)

//{

//case 1: return 10;

//case 2: return 1;

//}

return 2;

}

double func(double x, double y, int i)//f

{

/\* switch (i)

{

case 1: return -40;

case 2: return -4;

}\*/

return -8;

//return-5;

//return 6 \* vr - 8;

// return cos(vr);

}

double func\_kraev1(double x, double y, int k)

{

return /\*sin(vr);\*/x \* x + y \* y + vr;

}

double func\_kraev3(double\* x, int k)

{

switch (k)

{

case 1: return ( 2\*x[0]\* x[0] +32 );

}

return 0;

}

void input() {

// Read points

ifstream file("toch.txt");

file >> n;

tch = new double\* [n];

for (int i = 0; i < n; i++) {

tch[i] = new double[2];

for (int j = 0; j < 2; j++) {

file >> tch[i][j];

}

}

file.close();

// Read triangles

file.open("tr.txt");

file >> n2;

tr = new double[n2 \* 4];

for (int i = 0; i < n2; i++) {

for (int j = 0; j < 4; j++) {

file >> tr[i \* 4 + j];

}

}

file.close();

// Read k1

file.open("kr\_1.txt");

file >> n3;

k1 = new int[n3 \* 2];

for (int i = 0; i < n3 \* 2; i++) {

file >> k1[i];

}

file.close();

// Read k2 if k2 is initialized

file.open("kr\_2.txt");

file >> n4;

k2 = new int[n4 \* 4];

for (int i = 0; i < n4 \* 4; i++) {

file >> k2[i];

}

file.close();

file.open("kr\_3.txt");

file >> n5;

k3 = new int[n5 \* 4];

for (int i = 0; i < n5 \* 4; i++) {

file >> k3[i];

}

file.close();

file.open("t.txt");

double m, ma;

file >>m>>ma>> n6;

double hag = (ma - m) / (n6-1);

T = new double[n6];

for (int i = 0; i < n6; i++) {

T[i]=i\*hag;

}

file.close();

}

int\* copyToOneDimensionalArray(int\*\* source, int n, int& newSize) {

int\* destination = new int[newSize];

newSize = 0;

for (int i = 0; i < n; i++) {

for (int j = 1; j < n; j++) { // Начинаем с 1, чтобы пропустить первый элемент

if (source[i][j] != 0) {

destination[newSize] = source[i][j];

newSize++;

}

}

}

return destination; // Возвращаем указатель на новый массив

}

void sortArray(int\*\* arr, int n) {

for (int i = 0; i < n; i++) {

for (int j = 1; j < n - 1; j++) { // Начинаем с 1, чтобы оставить первый элемент

for (int k = 1; k < n - j; k++) {

if (arr[i][k] != 0 && arr[i][k + 1] != 0 && arr[i][k] > arr[i][k + 1]) {

// Меняем местами

int temp = arr[i][k];

arr[i][k] = arr[i][k + 1];

arr[i][k + 1] = temp;

}

}

}

}

}

void portret()

{

int i = 0;

int j = 0;

int\*\* vrem;

ig = new int(n + 1);

vrem = new int\* [n];

for (int i = 0; i < n; i++) {

vrem[i] = new int[n];

}

for (int i = 0; i < n; i++) {

vrem[i][0] = (i + 1);

for (int j = 1; j < n; j++) {

vrem[i][j] = 0;

}

}

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

ig[i] = 0;

for (int i = 0; i < n2; i++)

{

for (int j = 0; j < 3; j++) {

for (int k = 0; k < 3; k++) {

if (tr[i \* 4 + k] > tr[i \* 4 + j]) {

int to = 0;

int kk = tr[i \* 4 + k];

for (int l = 1; l < n; l++) {

if (vrem[kk - 1][l] == tr[i \* 4 + j]) {

to = 1;

break;

}

if (vrem[kk - 1][l] == 0) {

break;

}

}

if (to == 0)

for (int l = 1; l < n; l++) {

if (vrem[kk - 1][l] == 0) {

vrem[kk - 1][l] = tr[i \* 4 + j];

break;

}

}

}

}

}

}

sortArray(vrem, n);

for (int i = 1; i < n + 1; i++) {

ig[i] = ig[i - 1];

for (int j = 1; j < n; j++) {

if (vrem[i - 1][j] > 0)

ig[i]++;

}

}

int newSize = ig[n];

jg = new int[newSize];

jg = copyToOneDimensionalArray(vrem, n, newSize);

}

void M\_matrix(double\* p1, double\* p2, double\* p3, double\*\* M\_matr, double\* local\_F, int num\_of\_area) {

int i = 0;

int j = 0;

double det = (p2[0] - p1[0]) \* (p3[1] - p1[1]) - (p3[0] - p1[0]) \* (p2[1] - p1[1]);

double mnoz = fabs(det) / 24;

double\* f = new double[3];

double mnoz2 = gamma(p1[0], p1[1]) \* mnoz;

f[0] = mnoz \* func(p1[0], p1[1], num\_of\_area);

f[1] = mnoz \* func(p2[0], p2[1], num\_of\_area);

f[2] = mnoz \* func(p3[0], p3[1], num\_of\_area);

local\_F[0] = 2 \* f[0] + f[1] + f[2];

local\_F[1] = f[0] + 2 \* f[1] + f[2];

local\_F[2] = f[0] + f[1] + 2 \* f[2];

for (i = 0; i < 3; i++)

for (j = 0; j < 3; j++)

if (i == j) {

M\_matr[i][j] = 2 \* mnoz2;

}

else

M\_matr[i][j] = mnoz2;

}

void G\_matrix(double\* p1, double\* p2, double\* p3, double\*\* G\_matr, int k) {

int i = 0;

int j = 0;

double ck = 0;

double det = (p2[0] - p1[0]) \* (p3[1] - p1[1]) - (p3[0] - p1[0]) \* (p2[1] - p1[1]);

double a11, a12, a21, a22, a31, a32;

a11 = (p2[1] - p3[1]) / det;//y2-y3/det

a12 = (p3[0] - p2[0]) / det;//x3-x2/det

a21 = (p3[1] - p1[1]) / det;//y3-y1/det

a22 = (p1[0] - p3[0]) / det;//x1-x3/det

a31 = (p1[1] - p2[1]) / det;//y1-y2/det

a32 = (p2[0] - p1[0]) / det;//x2-x1/det

ck = lambda(p1[0], p1[1], k) \* fabs(det) / 2;

G\_matr[0][0] = ck \* (a11 \* a11 + a12 \* a12);

G\_matr[0][1] = ck \* (a11 \* a21 + a12 \* a22);

G\_matr[0][2] = ck \* (a11 \* a31 + a12 \* a32);

G\_matr[1][0] = G\_matr[0][1];

G\_matr[1][1] = ck \* (a21 \* a21 + a22 \* a22);

G\_matr[1][2] = ck \* (a21 \* a31 + a22 \* a32);

G\_matr[2][0] = G\_matr[0][2];

G\_matr[2][1] = G\_matr[1][2];

G\_matr[2][2] = ck \* (a31 \* a31 + a32 \* a32);

}

void addToElement(int row, int col, double value) {

// Проверяем, что row и col находятся в допустимых пределах

if (row < 0 || col < 0 || row >= n || col >= n) {

cerr << "Индексы вне диапазона!" << std::endl;

return;

}

// Находим позицию для добавления значения

bool found = false;

// Обновление верхней части матрицы (или главной диагонали)

for (int i = ig[col]; i < ig[col + 1]; ++i) {

if (jg[i] - 1 == row) {

// Если элемент найден, обновляем его значение

ggu3[i] += value;

found = true;

break;

}

}

// Если не нашли в верхней части, проверяем нижнюю часть

if (!found) {

for (int i = ig[row]; i < ig[row + 1]; ++i) {

if (jg[i] - 1 == col) {

// Если элемент найден, обновляем его значение

ggl3[i] += value;

found = true;

break;

}

}

}

// Если row и col совпадают, обновляем значение на диагонали

if (row == col) {

di3[row] += value;

}

}

void zeroOutRow(int row) {

for (int i = 0; i < ig[n]; ++i) {

if (jg[i] - 1 == row) {

ggu[i] = 0; // Обнуляем элементы ggu

}

}

int k = 0;

for (int i = 0; i < n; ++i) {

if (i == row)

while (ig[i + 1] - (ig[i] + k) != 0) {

ggl[ig[i] + k] = 0; // Обнуляем элементы ggu

k++;

}

}

}

void pervoe\_kraevoe(int vertex1, int vertex2, int form1, int form2) {

int kol = 0;

int lbeg;

int lend;

// Устанавливаем значения для трех вершин

di[vertex1] = 1;

di[vertex2] = 1;

// Вычисляем значения для F

F[vertex1] = func\_kraev1(tch[vertex1][0], tch[vertex1][1], form1);

F[vertex2] = func\_kraev1(tch[vertex2][0], tch[vertex2][1], form2);

zeroOutRow(vertex1);

zeroOutRow(vertex2);

}

void tretie(int vertex1, int vertex2, int form1) {

double\*\* m = new double\* [2];

for (int i = 0; i < 2; i++) {

m[i] = new double[2];

for (int j = 0; j < 2; j++) {

if (i == j)

m[i][j] = 2;

else

m[i][j] = 1;

}

}

double a = func\_kraev3(tch[vertex1], form1);

double b = func\_kraev3(tch[vertex2], form1);

double h = betta(form1) \* sqrt(pow((tch[vertex2][0] - tch[vertex1][0]), 2) + pow((tch[vertex2][1] - tch[vertex1][1]), 2)) / 6;

F[vertex1] += h \* (m[0][0] \* a + m[0][1] \* b);

F[vertex2] += h \* (m[1][0] \* a + m[1][1] \* b);

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

m[i][j] \*= h;

}

}

addToElement(vertex1, vertex1, m[0][0]);

addToElement(vertex1, vertex2, m[0][1]);

addToElement(vertex2, vertex1, m[1][0]);

addToElement(vertex2, vertex2, m[1][1]);

}

void vtoroe(int vertex1, int vertex2, int form1) {

double\*\* m = new double\* [2];

for (int i = 0; i < 2; i++) {

m[i] = new double[2];

for (int j = 0; j < 2; j++) {

if (i == j)

m[i][j] = 2;

else

m[i][j] = 1;

}

}

double h = sqrt(pow((tch[vertex2][0] - tch[vertex1][0]), 2) + pow((tch[vertex2][1] - tch[vertex1][1]), 2)) / 6;

F[vertex1] += h \* (m[0][0] \* ooo(tch[vertex1][0], tch[vertex1][1], form1) + m[0][1] \* ooo(tch[vertex2][0], tch[vertex2][1], form1));

F[vertex2] += h \* (m[1][0] \* ooo(tch[vertex1][0], tch[vertex1][1], form1) + m[1][1] \* ooo(tch[vertex2][0], tch[vertex2][1], form1));

}

void local\_matrix(int num\_of\_finit\_element, double\*\* local\_matr, double\*\* local\_matrG, double\* local\_F, int io) {

int ko = tr[io + 0] - 1;

int l = tr[io + 1] - 1;

int m = tr[io + 2] - 1;

double\*\* M\_matr = new double\* [3];

double\*\* G\_matr = new double\* [3];

for (int i = 0; i < 3; i++) {

M\_matr[i] = new double[3]();

G\_matr[i] = new double[3]();

}

M\_matrix(tch[ko], tch[l], tch[m], M\_matr, local\_F, tr[io + 3]);

G\_matrix(tch[ko], tch[l], tch[m], G\_matr, tr[io + 3]);

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

local\_matr[i][j] = M\_matr[i][j];

local\_matrG[i][j] = G\_matr[i][j];

}

}

for (int i = 0; i < 3; i++) {

delete[] M\_matr[i];

delete[] G\_matr[i];

}

}

void Mult\_A\_Vectr(double\* xn, double\* vec)//перемножение исходной матрицы А на вектор

{

long i, j, st;

for (i = 0; i < n; i++) {

vec[i] = di[i] \* xn[i];

for (j = ig[i]; j < ig[i + 1]; j++) {

st = jg[j] - 1;

vec[i] += ggl[j] \* xn[st];

vec[st] += ggu[j] \* xn[i];

}

}

}

void global\_matrix() {

int i, j, k;

int\* L = new int[3];

double\* local\_F = new double[3];

double\*\* local\_matr = new double\* [3];

double\*\* local\_matrG = new double\* [3];

for (i = 0; i < 3; i++) {

local\_matr[i] = new double[3]();

local\_matrG[i] = new double[3]();

}

int t = 0;

for (k = 0; k < n2; k++) {

local\_matrix(k, local\_matr, local\_matrG, local\_F, k \* 4);

for (int i = 0; i < 3; i++) {

L[i] = tr[k \* 4 + i] - 1;

}

for (int i = 0; i < 3; ++i) {

int ibeg = L[i]; // Начальный индекс строки

di[ibeg] += local\_matr[i][i]; // Обновление диагонали

di2[ibeg] += local\_matrG[i][i];

for (int j = i + 1; j < 3; ++j) {

int iend = L[j]; // Конечный индекс

// Выбор подходящего диапазона и обновление

int h;

if (ibeg < iend) {

h = ig[iend];

while (jg[h] - 1 < ibeg) h++;

ggl[h] += local\_matr[i][j]; // Нижний треугольник

ggu[h] += local\_matr[j][i]; // Верхний треугольник

ggl2[h] += local\_matrG[i][j]; // Нижний треугольник

ggu2[h] += local\_matrG[j][i]; // Верхний треугольник

}

else {

h = ig[ibeg];

while (jg[h] - 1 < iend) h++;

ggl[h] += local\_matr[i][j]; // Нижний треугольник

ggu[h] += local\_matr[j][i]; // Верхний треугольник

ggl2[h] += local\_matrG[i][j]; // Нижний треугольник

ggu2[h] += local\_matrG[j][i]; // Верхний треугольник

}

}

}

// Обновление правой части

for (int i = 0; i < 3; ++i) {

F[L[i]] += local\_F[i];

}

}

for (i = 0; i < n4 \* 2; i += 4)

vtoroe(k2[i] - 1, k2[i + 2] - 1, k2[i + 1]);

for (i = 0; i < n5 \* 2; i += 4)

tretie(k3[i] - 1, k3[i + 2] - 1, k3[i + 1]);

//d

Mult\_A\_Vectr(q0, dt0v);

Mult\_A\_Vectr(q1, dtv);

for (i = 0; i < n; i++)

F[i] = F[i] - t0 \* dt0v[i] + t1 \* dtv[i];

//a=tp\_M+G//вроде верно

for (i = 0; i < ig[n]; i++) {

ggl[i] = (ggl[i] \* tp) + ggl2[i];

ggu[i] = (ggu[i] \* tp) + ggu2[i];

}

for (i = 0; i < n; i++) {

di[i] = (di[i] \* tp) + di2[i];

}

//A+=M\_s3

for (i = 0; i < ig[n]; i ++) {

ggl[i] += ggl3[i];

ggu[i] += ggu3[i];

}

for (i = 0; i < n; i++)

di[i] += di3[i];

for (i = 0; i < n3 \* 2; i = i + 4) {

pervoe\_kraevoe(k1[i] - 1, k1[i + 2] - 1, k1[i + 1], k1[i + 3]);

}

}

double norma(double\* w)//НОРМА ВЕКТОРА

{

double s = 0;

for (int i = 0; i < n; i++)

s += w[i] \* w[i];

return sqrt(s);

}

double calc(int i, int j, double\* gl, double\* gu, int kl)//по формуле это сумма которую вычитаем

{

double s = 0;

int k, J = jg[kl] - 1, p;

for (k = j; k > 0; k--)

for (p = ig[J]; p < ig[J + 1]; p++)

if (jg[p] - 1 == jg[kl - k] - 1)

s += gl[kl - k] \* gu[p];

return s;

}

double calcD(int j, double\* gl, double\* gu, int kl)//аналогично только для диагонали

{

double s = 0;

for (int k = kl - j; k < kl; k++)

s += gl[k] \* gu[k];

return s;

}

void lulu(double\* gl, double\* gu, double\* gd)

{

int i, j, kol, kl = 0, ku = 0;

for (i = 0; i < n; i++) {

kol = ig[i + 1] - ig[i];

for (j = 0; j < kol; j++, kl++)

gl[kl] = (ggl[kl] - calc(i, j, gl, gu, kl)) / gd[jg[kl] - 1];

for (j = 0; j < kol; j++, ku++)

gu[ku] = (ggu[ku] - calc(i, j, gu, gl, ku)) / gd[jg[ku] - 1];

gd[i] = sqrt(di[i] - calcD(j, gu, gl, kl));

}

}

void Mult\_A\_Vect(double\* xn)//перемножение исходной матрицы А на вектор

{

long i, j, st;

for (i = 0; i < n; i++) {

f[i] = di[i] \* xn[i];

for (j = ig[i]; j < ig[i + 1]; j++) {

st = jg[j] - 1;

f[i] += ggl[j] \* xn[st];

f[st] += ggu[j] \* xn[i];

}

}

}//на выходе вектор f который является глобальным

double sk\_pr(double\* a, double\* b)//скалярное произведение векторов.

{

double s = 0;

for (int i = 0; i < n; i++)

s += a[i] \* b[i];

return s;

}

void LOC()

{

double nvzk, alfa, beta, skp, eps = 9.999999682655226e-030;

int i;

double lastnvzk;

Mult\_A\_Vect(x);

for (i = 0; i < n; i++)

z[i] = r[i] = F[i] - f[i];

Mult\_A\_Vect(z);

for (i = 0; i < n; i++)

p[i] = f[i];

nvzk = sqrt(sk\_pr(r, r)) / sqrt(sk\_pr(F, F));

for (int k = 1; k < 10000 && nvzk > eps; k++)

{

lastnvzk = nvzk;

skp = sk\_pr(p, p);

alfa = sk\_pr(p, r) / skp;

for (i = 0; i < n; i++) {

x[i] += alfa \* z[i];

r[i] -= alfa \* p[i];

}

Mult\_A\_Vect(r);

beta = -sk\_pr(p, f) / skp;

for (i = 0; i < n; i++) {

z[i] = r[i] + beta \* z[i];

p[i] = f[i] + beta \* p[i];

}

nvzk = sqrt(sk\_pr(r, r)) / sqrt(sk\_pr(F, F));

}

}

int main() {

double\* tr;

ofstream file11("1.1.txt");

ofstream error\_log("error\_log.txt"); // Файл для записи ошибок

input();

x = new double[n];

q0 = new double[n];

q1 = new double[n];

di = new double[n];

di2 = new double[n];

di3 = new double[n];

F = new double[n];

z = new double[n];

r = new double[n];

p = new double[n];

l = new double[n];

l1 = new double[n];

f = new double[n];

dtv = new double[n];

dt0v = new double[n];

tr = new double[n];

x22 = new double[n];

// Вычисление начальных условий

for (int i = 0; i < n; i++) {

q0[i] = u0(tch[i][0], tch[i][1], 0);

q1[i] = u0(tch[i][0], tch[i][1], 1);

}

// Основной цикл по временным слоям

for (int hi = 2; hi < n6; hi++) {

vr = T[hi];

double dt = T[hi] - T[hi - 2];

double dt0 = T[hi] - T[hi - 1];

double dt1 = T[hi - 1] - T[hi - 2];

tp = (dt + dt0) / (dt \* dt0);

t0 = (dt0) / (dt \* dt1);

t1 = (dt) / (dt1 \* dt0);

portret();

cout << endl;

// Инициализация массивов

F = new double[n];

for (int i = 0; i < n; i++) {

di[i] = di3[i] = di2[i] = F[i] = x[i] = 0;

}

// Инициализация матриц

ggu = new double[ig[n] - 1];

ggl = new double[ig[n] - 1];

ggu2 = new double[ig[n] - 1];

ggl2 = new double[ig[n] - 1];

ggu3 = new double[ig[n] - 1];

ggl3 = new double[ig[n] - 1];

for (int i = 0; i < ig[n]; i++) {

ggu3[i] = ggl3[i] = ggu2[i] = ggl2[i] = ggu[i] = ggl[i] = 0;

}

global\_matrix();

LOC();

// Сохранение решения

for (int i = 0; i < n; i++)

x22[i] = x[i];

tochnoe();

// Запись результатов

file11 << vr << endl;

for (int i = 0; i < n; i++)

file11 << setprecision(20) << x[i] << " " << x22[i] << " " << x[i] - x22[i] << endl;

// Вычисление погрешностей для последнего слоя

// Обновление значений для следующего шага

for (int i = 0; i < n; i++) {

q0[i] = q1[i];

q1[i] = x22[i];

}

}

file11.close();

error\_log.close();

return 0;

}