МИНИСТЕРСТВО НАУКИ
И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Новосибирский государственный технический университет»

НГТУ НЭТИ

Кафедра прикладной математики

Курсовой проект
по дисциплине «Численные методы»

ФПМИ

ГРУППА	ПМ-92
ВАРИАНТ	21
СТУДЕНТ	ГЛУШКО ВЛАДИСЛАВ
ПРЕПОДАВАТЕЛЬ	СОЛОВЕЙЧИК ЮРИЙ ГРИГОРЬЕВИЧ

Новосибирск

1 Условие задачи

Формулировка задачи

МКЭ для двумерной краевой задачи для эллиптического уравнения в декартовой системе координат. Базисные функции линейные на треугольниках. Краевые условия всех ти-пов. Коэффициент разложить по линейным базисным функциям. Матрицу СЛАУ генери-ровать в разреженном строчном формате. Для решения СЛАУ использовать МСГ или ЛОС с неполной факторизацией.

Постановка задачи

Эллиптическая краевая задача для функции *u* определяется дифференциальным уравнением

$$-div(\lambda gradu) + \gamma u = f$$

заданным в некоторой области Ω с границей $S = S_1 \cup S_2 \cup S_3$ и краевыми условиями:

$$u|_{S_1} = u_g$$

$$\lambda \frac{\partial u}{\partial n}\bigg|_{S_2} = \theta$$

$$\lambda \frac{\partial u}{\partial n}\bigg|_{S_3} + \beta(u|_{S_3} - u_\beta) = 0$$

В декартовой системе координат x,y это уравнение может быть записано в виде

$$-\frac{\partial}{\partial x}\left(\lambda \frac{\partial u}{\partial x}\right) - \frac{\partial}{\partial y}\left(\lambda \frac{\partial u}{\partial y}\right) + \gamma u = f$$

Конечноэлементная дискретизация

Так как для решения задачи используются линейные базисные функции, то на каждом конечном элементе Ω_k - треугольнике эти функции будут совпадать с функциями $L_1(x,y), L_2(x,y), L_3(x,y)$, такими, что $L_1(x,y)$ равна единице в вершине (x_1, y_1) и нулю во всех остальных вершинах, $L_2(x,y)$ равна единице в вершине (x_2, y_2) и нулю во всех остальных вершинах, $L_3(x,y)$ равна единице в вершине (x_3, y_3) и нулю во всех остальных вершинах. Любая линейная на Ω_k функция представима в виде линейной комбинации этих базисных линейных функций, коэффициентами будут значения функции в каждой из вершин треугольника Ω_k. Таким образом, на каждом конечном элементе нам понадобятся три узла – вершины треугольника.

$$\psi_1 = L_1(x,y)$$

$$\psi_2 = L_2(x,y)$$

$$\psi_3 = L_3(x,y)$$

Учитывая построение *L-функций*, получаем следующие соотношения:

$$\begin{cases} L_1 + L_2 + L_3 = 1 \\ L_1x_1 + L_2x_2 + L_3x_3 = x \\ L_1y_1 + L_2y_2 + L_3y_3 = y \end{cases}$$

Т.е. имеем систему:

$$\begin{pmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix} \cdot \begin{pmatrix} L_1 \\ L_2 \\ L_3 \end{pmatrix} = \begin{pmatrix} 1 \\ x \\ y \end{pmatrix}$$

Отсюда находим коэффициенты линейных функций $L_1(x,y), L_2(x,y), L_3(x,y)$

$$L_i = a_0^i + a_1^i x + a_2^i y, i = \overline{1,3}$$

$$\begin{pmatrix} \alpha_0^1 & \alpha_1^1 & \alpha_2^1 \\ \alpha_0^2 & \alpha_1^2 & \alpha_2^2 \\ \alpha_0^3 & \alpha_1^3 & \alpha_2^3 \end{pmatrix} = D^{-1} = \begin{pmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{pmatrix}^{-1}$$

$$D^{-1} = \frac{1}{|detD|} \begin{pmatrix} x_2y_3 - x_3y_2 & y_2 - y_3 & x_3 - x_2 \\ x_3y_1 - x_1y_3 & y_3 - y_1 & x_1 - x_3 \\ x_1y_2 - x_2y_1 & y_1 - y_2 & x_2 - x_1 \end{pmatrix}$$

Переход к локальным матрицам

Чтобы получить выражения для локальных матриц жёсткости G и массы M каждого конечного элемента Ω_K, перейдём к решению локальной задачи на каждом конечном элементе. Полученное уравнение для области Ω представим в виде суммы интегралов по областям Ω_k без учёта краевых условий. Тогда на каждом конечном элементе будем решать локальную задачу построения матриц жёсткости, массы и вектора правой части.

$$\int\limits_{\Omega_k} \lambda \left(\frac{\partial \psi_j}{\partial x} \frac{\partial \psi_i}{\partial x} + \frac{\partial \psi_j}{\partial y} \frac{\partial \psi_i}{\partial y} \right) dxdy + \int\limits_{\Omega_k} \gamma \psi_j \psi_i dsdy = \int\limits_{\Omega_k} f \psi_i dxdy$$

Локальная матрица будет представлять собой сумму матриц жёсткости и массы и будет иметь размерность 3×3 (по числу узлов на конечном элементе)

Построение матрицы массы

$$\begin{aligned} M_{ij} = \int\limits_{\Omega_m} \gamma Y_i Y_j d\Omega_m = \left| \gamma = Y_1\gamma_1 + Y_2\gamma_2 + Y_3\gamma_3 \right| = \int\limits_{\Omega_m} \left(Y_1\gamma_1 + Y_2\gamma_2 + Y_3\gamma_3 \right) Y_i Y_j d\Omega_m = \\ = \gamma_1 \int\limits_{\Omega_m} Y_1 Y_i Y_j d\Omega_m + \gamma_2 \int\limits_{\Omega_m} Y_2 Y_i Y_j d\Omega_m + \gamma_3 \int\limits_{\Omega_m} Y_3 Y_i Y_j d\Omega_m = \\ = \gamma_1 \int\limits_{\Omega_m} L_1 L_i L_j d\Omega_m + \gamma_2 \int\limits_{\Omega_m} L_2 L_i L_j d\Omega_m + \gamma_3 \int\limits_{\Omega_m} L_3 L_i L_j d\Omega_m \end{aligned}$$

Построение матрицы жёсткости

Рассмотрим первый член в выражении для k-го конечного эдемента:

$$\int\limits_{\Omega_k} \lambda \left(\frac{\partial \psi_j}{\partial x} \frac{\partial \psi_i}{\partial x} + \frac{\partial \psi_j}{\partial y} \frac{\partial \psi_i}{\partial y} dxdy \right)$$

$$B_{i,j} = (\alpha_1^i \alpha_1^j + \alpha_2^i \alpha_2^j) \frac{\left| detD \right|}{2} \qquad i,j = \overline{0,2}$$

Построение вектора правой части

Рассмотрим правую часть выражения для k-го конечного элемента:

$$\int\limits_{\Omega_k} f \psi_i dxdy$$

представим f в виде $f_1 L_1 + f_2 L_2 + f_3 L_3$, где f_i - значения в вершинах треугольника. Получим:

$$\int\limits_{\Omega_k} f_q L_q L_i dxdy = f_q \int\limits_{\Omega_k} L_q L_i d\Omega_k$$

Таким образом:

$$G_i = \sum_{q=1}^{3} f_q \int\limits_{\Omega_k} L_q L_i d\Omega_k \qquad i = \overline{0,2}$$

Сборка глобальной матрицы и глобального вектора

При формировании глобальной матрицы из локальных, полученных суммированием соответствующих матриц массы и жесткости, учитываем соответствие локальной и глобальной нумераций каждого конечного элемента. Глобальная нумерация каждого конечного элемента однозначно определяет позиции вклада его локальной м атрицы в глобальную. Поэтому, зная глобальные номера соответствующих узлов конечного элемента, определяем и то, какие элементы глобальной матрицы изменятся при учете текущего конечного элемента. Аналогичным образом определяется вклад локального вектора правой части в глобальный. При учете текущего локального вектора изменятся те элементы глобального вектора правой части, номера которых совпадают с глобальными номерами узлов, присутствующих в этом конечном элементе.

Учёт первых краевых условий

Для учета первых краевых условий, в глобальной матрице и глобальном векторе находим соответствующую глобальному номеру краевого узла строку и зануляем всё кроме диагонального элемента, которому присваиваем 1, а вместо элемента с таким номером в векторе правой части - значение краевого условия, заданное в исходной задаче.

Учёт вторых и третьих краевых условий

Рассмотрим краевые условия второго и третьего рода:

$$\lambda \frac{\partial u}{\partial n}\bigg|_{S_2} = \theta$$

$$\lambda \frac{\partial u}{\partial n}\bigg|_{S_3} + \beta(u|_{S_3} - u_\beta) = 0$$

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

$$\int\limits_{S_2} \theta\psi_j dxdy, \qquad \int\limits_{S_3} \beta u_\beta \psi_j dxdy, \qquad \int\limits_{S_3} \beta\psi_i\psi_j dxdy$$

Краевые условия второго и третьего рода задаются на рёбрах, т.е. определяются двумя узлами, лежащими на ребру. Будем считать, что параметр β на S_3 постоянен, тогда параметр $_\beta$ будем раскладывать по двум базисным функциям, определённым на этом ребре:

$$u_\beta = u_{\beta 1}\phi_1 + u_{\beta 2}\phi_2$$

где $\phi_i, \ i = \overline{0,1}$ - локально занумерованные линейные базисные функции, которые имеют также свои глобальные номера во всей расчетной области, а $u_{\beta i}$ - значение функции u_β в узлах ребра.

Аналогично поступаем и при учете вторых краевых условий, раскладывая по базису ребра функцию $\theta = \theta_0\phi_0 + theta_1\phi_1$.
Тогда приведенные выше интегралы примут вид:

$$I_1 = \int\limits_{S_2} (\theta_0\phi_0 + \theta_1\phi_1)\phi_i dxdy$$

$$I_2 = \beta \int\limits_{S_3} (u_{\beta 1}\phi_0 + u_{\beta 2}\phi_1)\phi_i dxdy$$

$$I_3 = \beta \int\limits_{S_3} \phi_i\phi_j dxdy$$

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 $\phi_i, \ i = \overline{0,1}$ конечного элемента.

Для учёта вклада вторых и третьих краевых условий рассчитываются 2 матрицы 2×2.

Иттегралы I_1, I_2, I_3 будем вычислять по формуле:

$$\int (L_i)^{v_i}(L_j)^{v_j} dS = \frac{v_i! v_j!}{(v_i + v_j + 1)!} mes\mathit{\Gamma}, \ i \neq j$$

где $mes\Gamma$ длина ребра. При этом независимо от того, что на каждом из ребер присутствуют свои функции, интегралы, посчитанные по приведенным выше формулам, будут равны.

$$I_1 = \begin{pmatrix} \int\limits_{S_2} L_1 L_1 dxdy & \int\limits_{S_2} L_1 L_2 dxdy \\ \int\limits_{S_2} L_2 L_1 dxdy & \int\limits_{S_2} L_2 L_2 dxdy \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} = \frac{1}{6} mesS_2 \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix}$$

Этот вектор поправок в правую часть позволяет учесть не только вторые краевые условия, но и часть βu_β из третьих. Осталось рассмотреть матрицу поправок в левую часть:

$$I_3 = \beta \int\limits_{S_3} \phi_i \phi_j dxdy$$

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

$$I_3 = \frac{1}{6} mesS_3 \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

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

2 Текст программы

main.cpp

```cpp
#include "argparse/argparse.hpp"
#include "timer/cxxtimer.hpp"
#include "LOS/LOS.hpp"
#include "FEM.hpp"

#include <iostream>
#include <optional>
#include <fstream>

int main(int argc, char* argv[]) {
    using namespace    ::Log;
    using ::std::chrono::milliseconds;

    argparse::ArgumentParser _program("FEM", "1.0.0");
    _program.add_argument("-i", "--input")
        .help("path to input files" )
        .required();

    _program.add_argument("-o", "--output")
        .help("path to output files");
```

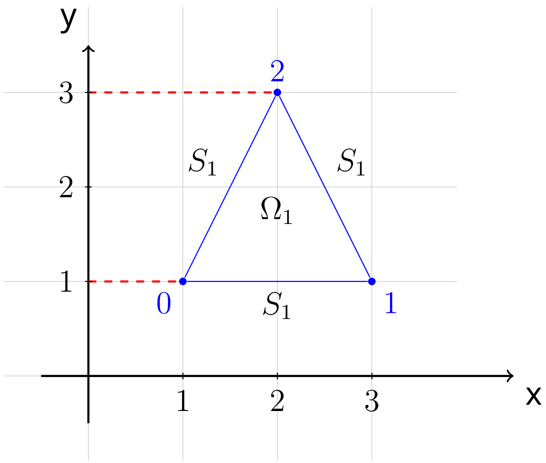
```cpp

    try {
        _program.parse_args(argc, argv);

        std::optional _opt             = _program.present("-o");
        std::filesystem::path _input   = _program.get<std::string>("-i");
        std::filesystem::path _output =
            _opt.has_value() ?
                _program.get<std::string>("-o") :
                _input / "sparse";

        Function::setFunction(_input.string());

        cxxtimer::Timer _timer(true);       /// start timer
        FEM _FEM(_input);                   /// start FEM
        LOS<double> _LOS (
            _FEM.takeDate(),                /// data
            _FEM.getNodes(),                /// count nodes
            1E-16, 1000);                   /// epsilon and max iteration
        _LOS.solve(Cond::HOLLESKY, true);   /// solve LOS + DIAGONAL
        _timer.stop();                      /// stop timer

        #if DEBUG != 0
        _FEM.printAll();                    /// print input FEM data
        _FEM.printSparse();                 /// print sparse format
        _LOS.printX(14);                    /// print solution vector
        _FEM.printAnalitics();              /// print analicals solve
        #endif

        std::cout << std::scientific << 0.1341234 << std::endl;
        std::cout << "Milliseconds: " << _timer.count<milliseconds>();

    } catch(const std::runtime_error& err) {
        Logger::append(getLog("argc != 2 (FEM --input ./input)"));
        std::cerr << err.what();
        std::cerr << _program;
        std::exit(1);                       /// program error
    }
    return 0;
}
```

FEM.hpp

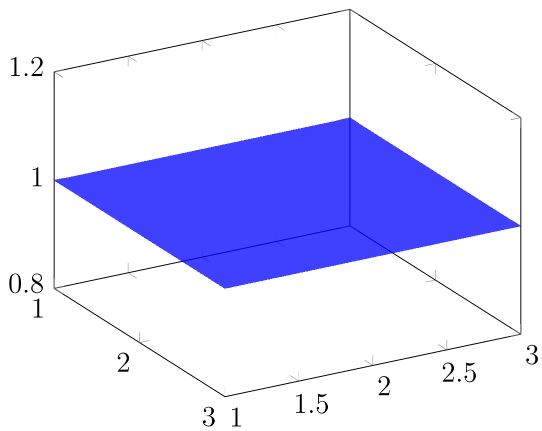
3 Тестирование

Тест №1

$$u(x, y) = 1$$
$$f(x, y) = 0$$
$$\lambda = 1$$
$$\gamma = 0$$
$$\beta = 0$$



$$I_0 = 1$$



nodes	elems	area	bords
1 1	0 1 2	0	0 0 1 1 0
3 1			0 1 2 1 0
2 3			0 2 0 1 0

x	x^*	$x^* - x$	$\|x^* - x\|$
1.000	1.000	0.00E+00	
1.000	1.000	0.00E+00	0.00E+00
1.000	1.000	0.00E+00	

Тест №2

Тест №3

Тест №4

Тест №5

4 Выводы

Выводы