Методы математической физики

Семинар № 7. Метод конечных элементов на примере первой краевой задачи для уравнения Пуассона.

Мих. Дмитр. Малых

Физический факультет МГУ

2012/13 уч. г., версия от 20 июня 2012 г.

По каким функциям разлагать решение в произвольной области?

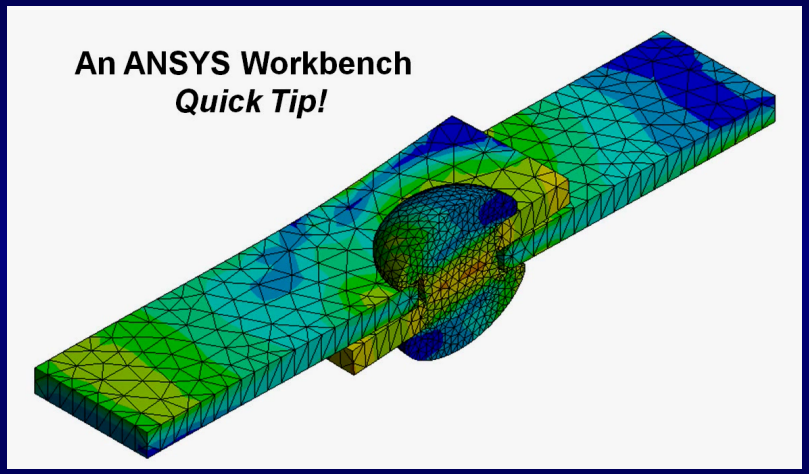
В случае круга и прямоугольника было вполне понятно, по каким функциям вести разложение. Для произвольной области указать подходящий набор функций удалось лишь в 1940-х годах, когда появился абрисс того, что теперь называют *методом конечных элементов* (МКЭ, FEM) и открытие которого обычно связывают с именем *Рихарда Куранта* (Courant, 1943). Название – метод конечных элементов – было предложено Клафом в середине 1960-х годов [1].

FEA Software

Работа Куранта не была замечена, но в конце 1950-х годах инженеры, занимавшиеся сопроматом, переоткрыли МКЭ. Рост популярности МКЭ во многом объясняется удобством его реализации на персональном компьютере. В настоящее время имеется великое множество компьютерных систем конечно-элементного анализа (FEA Software).

ANSYS, Inc.

В 1970 г. Джон Свенсон (John A. Swanson) в Пенсельвании основал Swanson Analysis Systems, Inc. (SASI), ставящую перед собой цель развитие программного обеспечение, основанного на МКЭ. В 1994 г. компания была продана и переименована в ANSYS, Inc.; одноименный продукт до сих пор существует на рынке.

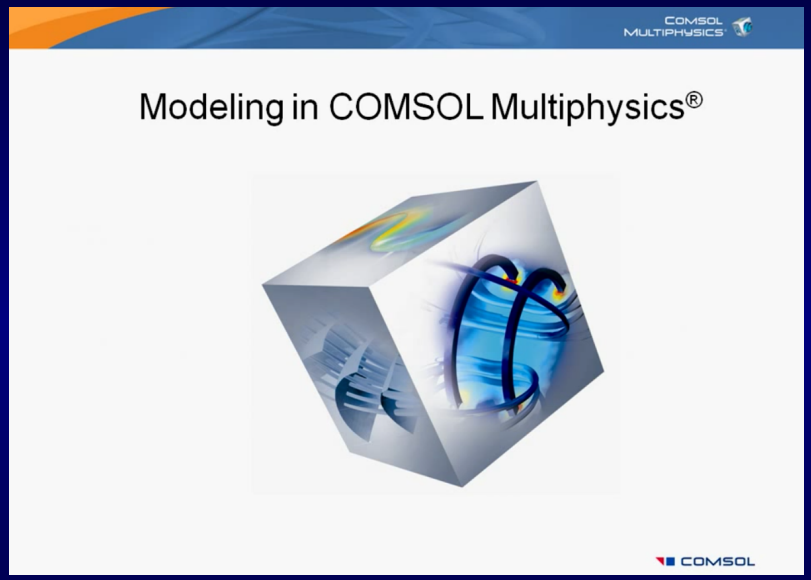


▸ YouTube

http://www.ansis.com

MatLab

В состав стандартной поставки MatLab'а входит Pde Toolbox, умеющий решать МКЭ основные линейные задачи мат. физики в плоских областях.
Хорошо интегрируется в MatLab пакет, созданный Comsol Multiphysics; его обычно называют FEMLab'ом.



▸ YouTube

http://www.comsol.com

FreeFem++

Мы будем использовать бесплатную программу FreeFem++, созданную в Лаборатории Ж.-Л. Лионса в Париже и поддерживающую большое число ОС: Linux, Mac OS X, Microsoft Windows и Solaris. Входит в репозиторий Ubuntu, начиная с версии 12.04.
Для справок см.:

- *Hecht F.* FreeFem++. Third Edition, Version 3.19. Laboratoire Jacques-Louis Lions, Université Pierre et Marie Curie, Paris, 2012.
- *Жуков М.Ю., Ширяева Е.В.* Пакет конечных элементов FreeFem++. Ростов-на-Дону, 2010

http://www.freefem.org

Задача

Первая краевая задача для уравнения Пуассона.

Простейшая краевая задача в 2d

Простейшей с точки зрения МКЭ является первая краевая задача для уравнения Пуассона, а именно

$$\begin{cases} \Delta u = f, & (x, y) \in \Omega \\ u|_{\partial\Omega} = 0. \end{cases} \tag{1}$$

Будем считать, что Ω – односвязная область на плоскости, ограниченная замкнутой кривой.

Классическое решение

Определение

Примем, что классическое решение дважды непрерывно дифференцируемо в области Ω и на ее границе.

Единственность такого решения очевидна: разность двух классических решений является решением однородной первой краевой задачи для уравнения Лапласа, и следовательно, равно нулю.
Вопрос о существовании классического решения оставим на потом, сосредоточившись на конструктивном построении такового.

Метод конечных элементов

Применение МКЭ к любой линейной краевой задачи мат. физики удобно разбить на несколько шагов:

1. триангуляция области,
2. построение системы базисных функций,
3. дискретизация дифференциального уравнения,
4. решение системы линейных алгебраически уравнений.

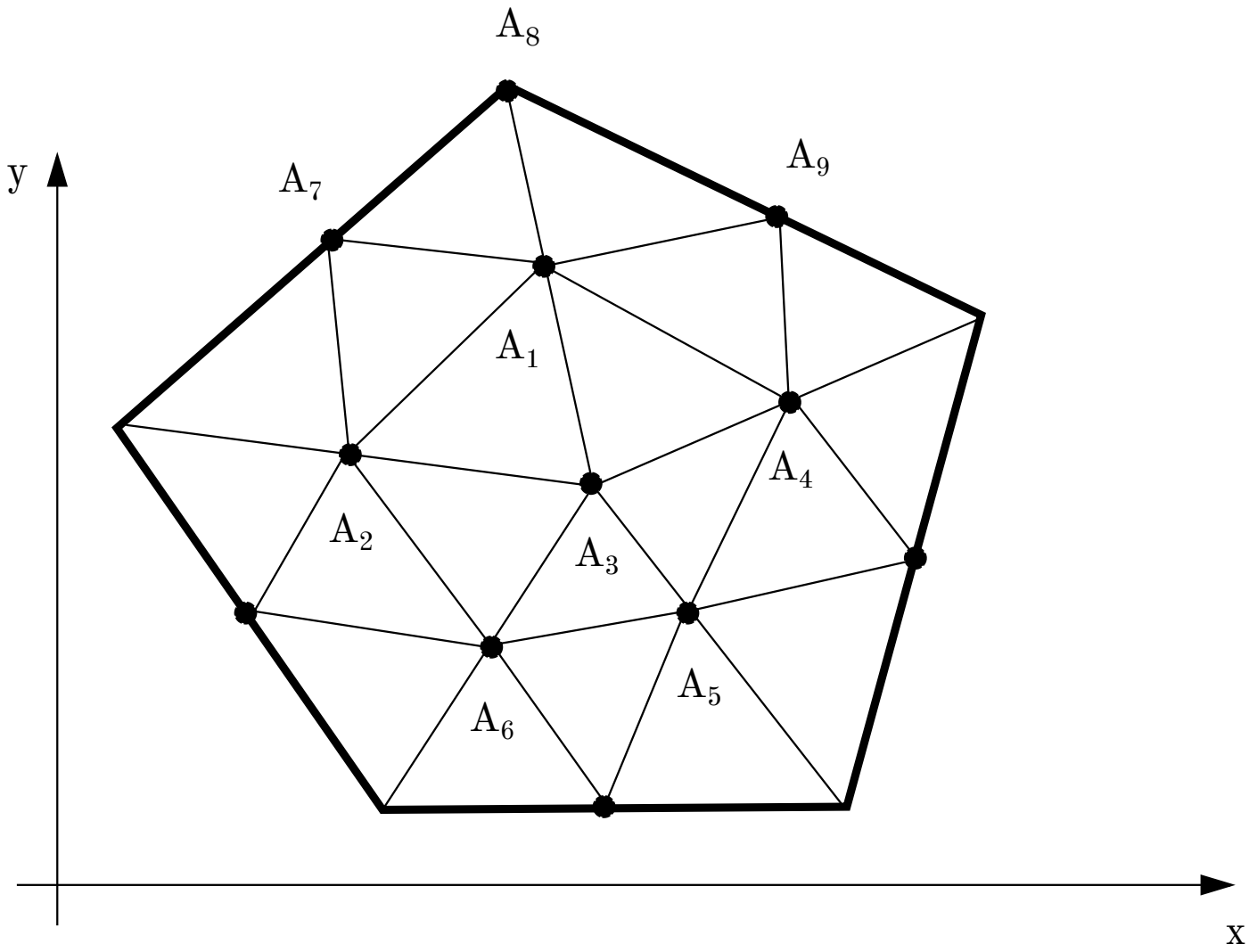
Шаг № 1

Триангуляция области

Триангуляция области

Разделим область Ω на треугольники, имеющие сходные размеры и пропорции. Линейные размеры треугольников сети характеризуют числом h, равным максимальной длине сторон треугольников.
В ручную заданную на клетчатой бумаге область поделить на небольшое число примерно равносторонних треугольников совсем не трудно. Если же треугольников много, применяют специальные алгоритмы триангуляции, восходящие к работе *Б.Н. Делоне* 1934 г. [4]

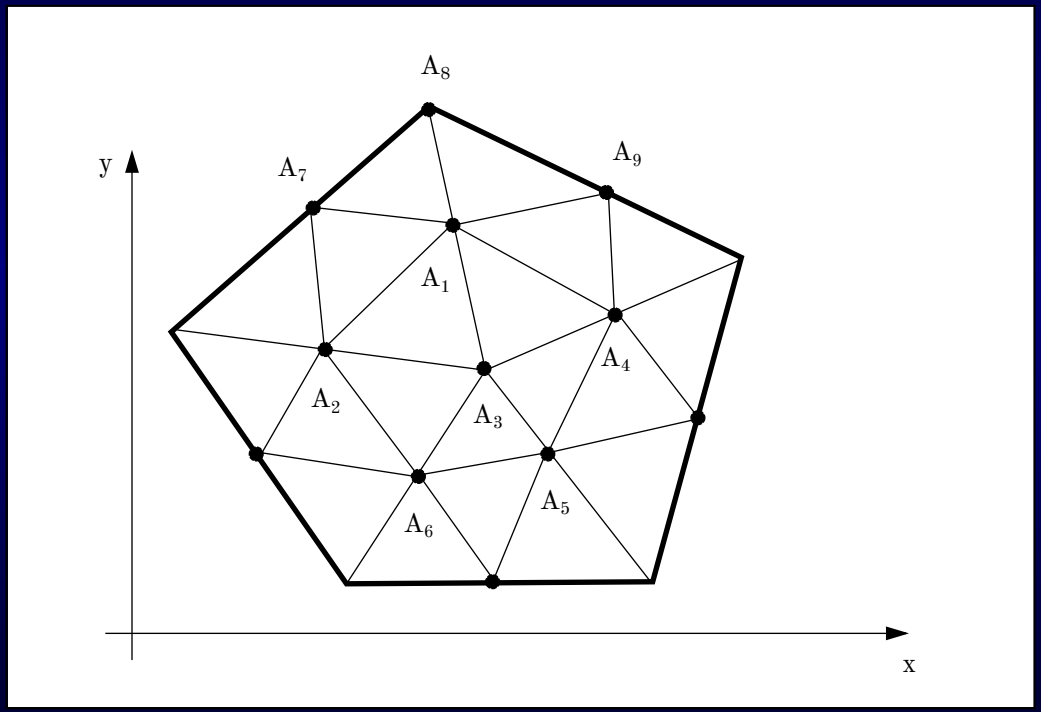
Триангуляция области



Терминология

Сами треугольники и называют конечными элементами, их вершины делят на внутренние, т.е. лежащие внутри области Ω, и граничные, то есть лежащие на границе этой области.
Две вершины будем называть соседними, если они являются вершинами одного треугольника.
Перенумеруем вершины в таком порядке, чтобы первыми шли внутренние вершины, а затем граничные. Число внутренних вершин будем обозначать как N.

Вершины



- Внутренние: $A_1, \ldots A_6$,
- граничные: $A_7, A_8, \ldots,$
- соседние с A_1: A_2, A_3, A_4 и A_7, A_8, A_9.

Пример: триангуляция эллипса

Для триангуляции эллипса создадим файл след. содержания.

Файл Poisson-1.edp

```
border C(t=0,2*pi){x=1.5*cos(t); y=sin(t);}
mesh Th = buildmesh (C(20));
plot(Th, ps="Poisson-1.eps",value=true,fill=false);
```

Здесь в первой строке задается граница C эллипса в параметрической форме, во второй задается сеть T_h с 20-тью вершинами на границе C, в третьей указано, что результат надо записать в графический файл `Poisson-1.eps`.

Пример: триангуляция эллипса

Теперь откроем терминал, перейдем в директорию, в которой находится файл `Poisson-1.edp`, и выполним команду

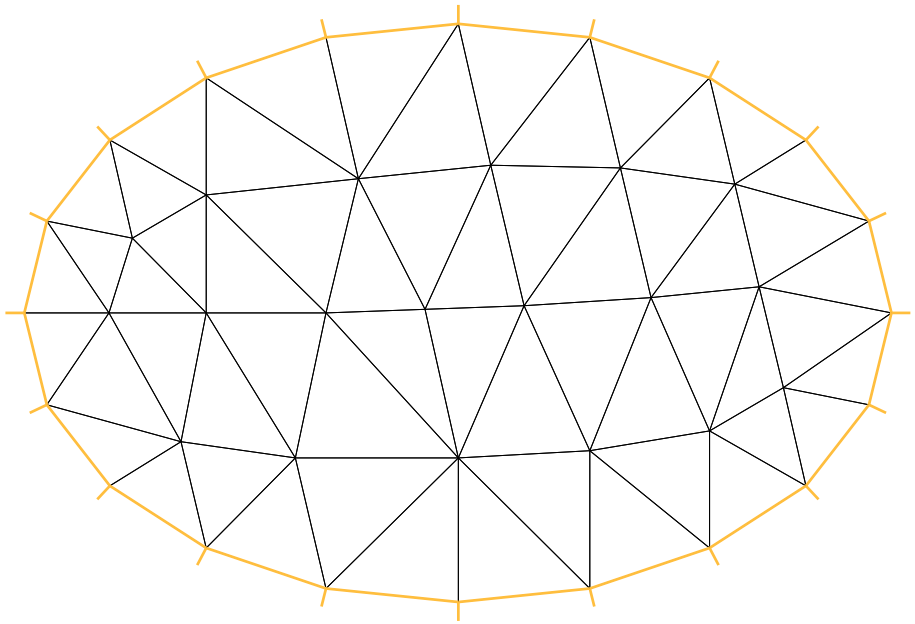
Терминал.

```
FreeFem++-nw Poisson-1.edp
```

В результате в той же директории будет создан файл `Poisson-1.eps`.

Замечание. – Использование в пути к файлу русских букв может стать источником проблем.

Результат: файл Poisson-1.eps



Шаг № 2

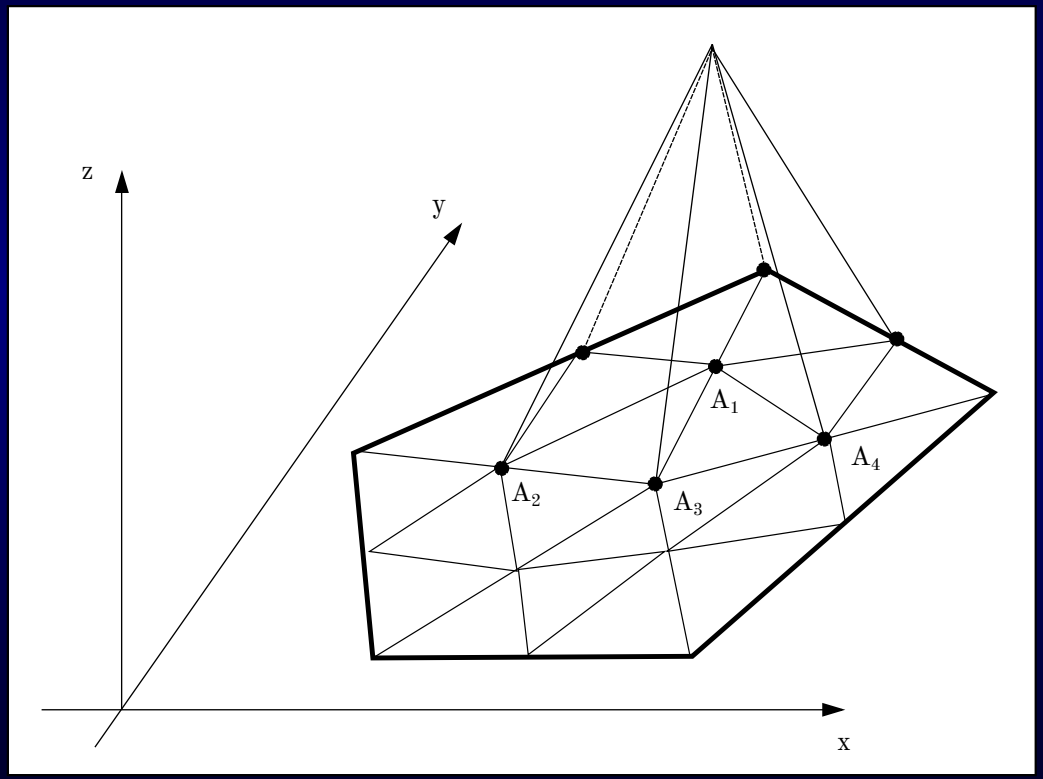
Построение системы базисных функций

Базисные функции

С каждой вершиной A_i свяжем функцию φ_i, которая линейна во всех конечных элементах и $\varphi_i(A_j) = \delta_{ij}$. Эта функция будет отлична от нуля только в тех треугольниках, одной из вершин которых является точка A_i.

График базисной функции φ_i

График функции φ_i представляет собой пирамиду высоты 1, основанием которой служит полигон с вершинами в точках, соседних с точкой A_i, а над самой точкой A_i размещается вершина пирамиды.



Линейная комбинация базисных функций

Линейная комбинация этих функций

$$\sum_i g_i \varphi_i(x, y)$$

является кусочно-линейной функцией, причем коэффициенты этой комбинации имеют очень простой геометрический смысл: g_i – значение этой функции в вершине A_i.

Разложение по базису

Для произвольной функции g разность

$$g(x, y) - \sum_i g(x_i, y_i)\varphi_i(x, y)$$

равна нулю во всех вершинах триангуляции.
Поскольку вершины триангуляции расположены друг от друга на расстоянии, примерно равном h, при надлежащих ограничениях на рост производной этой функции, эта функция всюду не больше h.

Разложение по базису

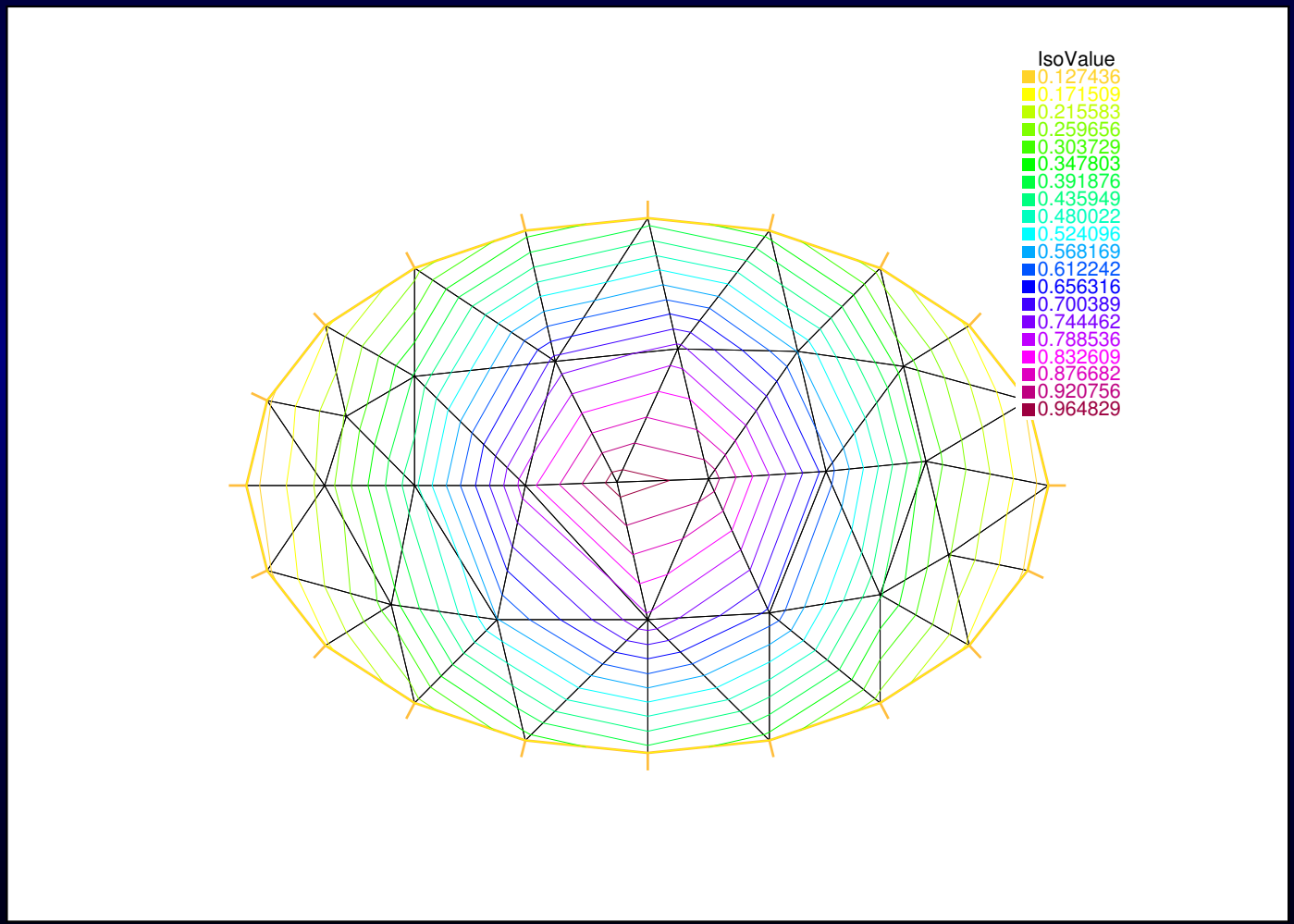
Гипотеза.

Любую функцию g можно аппроксимировать линейной комбинацией базисных функций, то есть с ошибкой порядка h верно

$$g(x, y) \approx \sum_i g(x_i, y_i)\varphi_i(x, y).$$

Справедливость этой гипотезы подтверждают многочисленные примеры.

Пример: аппроксимация функции $g = \exp(-x^2 - y^2)$.



Пример

Для построения этого рисунка в FreeFem++ нужно создать и выполнить файл след. содержания.

Файл Poisson-2.edp

```
border C(t=0,2*pi){x=1.5*cos(t); y=sin(t);}
mesh Th = buildmesh (C(20));
fespace Vh(Th,P1);
Vh gh=exp(-x^2-y^2);
plot(Th,gh, ps="Poisson-2.eps",value=true,fill=false);
```

Здесь в третьей строке задается линейная оболочка $\mathfrak{V}_h$, натянутая на базисные функции, т.н. пространство конечных элементов (f. e. space), в четвертой вычисляется аппроксимация g_h для функция $\exp(-x^2 - y^2)$.

Замечание

При изменении в 3-ей строке параметра P1, скажем, на P2, вместо аппроксимации линейными функциями получится аппроксимация полиномами второго порядка.

Шаг № 3

Дискретизация уравнения Пуассона.

Приближенное решение

Будем искать решение краевой задачи (1) в виде линейной комбинации базисных функций, отвечающих внутренним вершинам:

$$u = \sum_{i=1}^{N} u_i \varphi_i(x, y);$$

число N при этом именуют числом свободных параметров. Мы берем в этой сумме только базисные функции, отвечающие внутренним вершинам, поскольку на границе решение должно обращаться в нуль.
Мы не можем подставить это выражение в уравнение Пуассона $\Delta u = f$, поскольку базисные функции не имеют второй производной.

Слабая постановка задачи

Умножим уравнение Пуассона $\Delta u = f$ на произвольную функцию v и проинтегрируем по области. В силу формулы Гаусса-Остроградского тождество

$$\int\limits_{\Omega} dxdy(\nabla u, \nabla v) + \int\limits_{\Omega} dxdy f v = 0 \tag{2}$$

верно для любой функции v, лишь бы она была непрерывна в $\overline{\Omega}$, кусочно-дифференцируема функций в Ω и равна нулю на границе.

СЛАУ

В частности при $v = \varphi_j$ для точного решения u верно

$$\int\limits_\Omega dxdy(\nabla u, \nabla \varphi_j) + \int\limits_\Omega dxdy f\varphi_j = 0. \quad (j = 1, 2, \dots N)$$

Подставляя сюда

$$u = \sum_{i=1}^{N} u_i \varphi_i(x, y)$$

получим систему из N алгебраических уравнений для отыскания N коэффициентов u_i, именно

$$\sum_{i=1}^{N} \int\limits_\Omega dxdy(\nabla \varphi_i, \nabla \varphi_j)u_i + \int\limits_\Omega dxdy f\varphi_j = 0 \quad (j = 1, 2, \dots N)$$

Приближенное решение

Определение

Мы принимаем за приближенное решение u_h исходной краевой задачи линейную комбинацию

$$u_h = \sum_{i=1}^{N} u_i \varphi_i(x, y),$$

которая точно удовлетворят соотношению (2), т.е.

$$\int\limits_{\Omega} dxdy(\nabla u, \nabla v) + \int\limits_{\Omega} dxdy f v = 0$$

для всех функций v, которые сами являются линейной комбинацией базисных функций.

Шаг № 4

Решение СЛАУ.

СЛАУ в матричной форме

Исходная краевая задача свелась к системе алгебраических уравнений

$$\mathfrak{Au} + \mathfrak{f} = 0,$$

где $\mathfrak{A}$ – матрица $N \times N$ с элементами

$$a_{i,j} = \int\limits_{\Omega} dxdy(\nabla\varphi_i, \nabla\varphi_j),$$

$\mathfrak{f}$ – столбец длины N с элементами

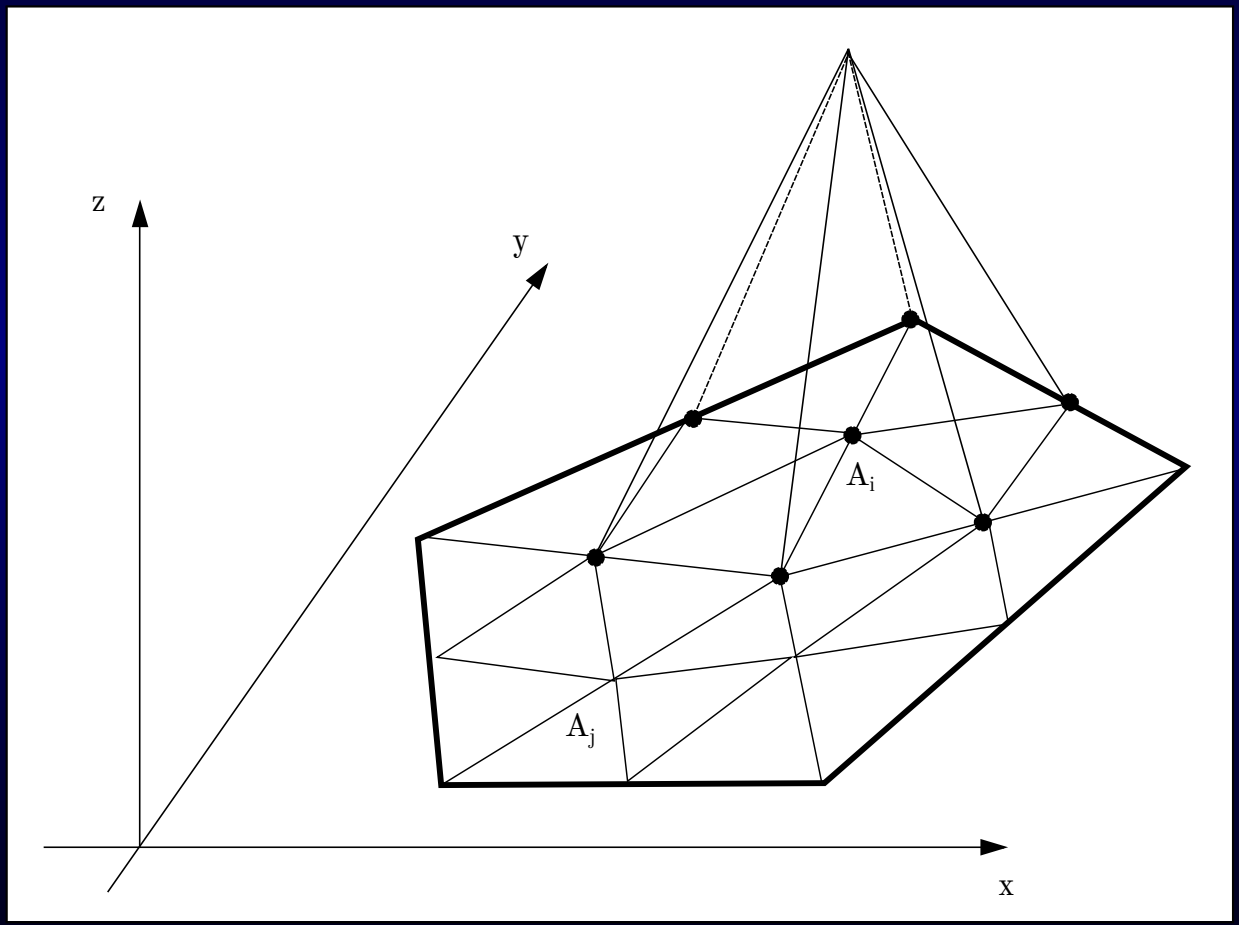
$$f_i = \int\limits_{\Omega} dxdy f\varphi_i,$$

а $\mathfrak{u}$ – столбец длины N, элементами которого служат искомые коэффициенты u_i.

СЛАУ большого порядка

Остается получившуюся систему линейных уравнений решить средствами линейной алгебры. Хотя на практике число свободных параметров N может иметь порядок сотни или даже тысячи, современные пакеты для работы с матрицами вычислят u за доли секунды. Однако не следует воображать себе, что при этом применяется правило *Крамера*, бывшее основным способом решения линейных уравнений в учебниках по Линейной алгебре в XIX веке.
`FreeFem++` использует при работе со СЛАУ сторонний пакет – UMFPACK (unsymmetric multifrontal sparse LU factorization package) Тима Девиса (Timothy A. Davis, 1994).

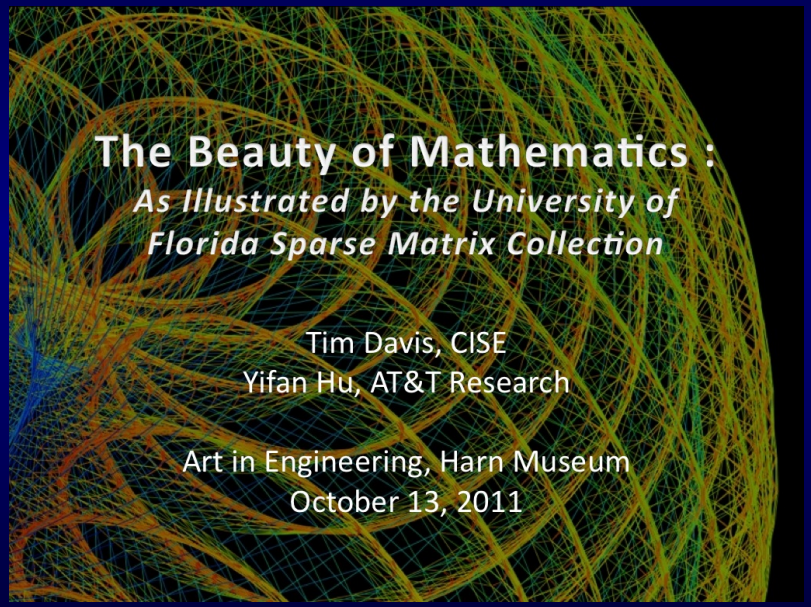
Структура матрицы и вершины сетки



Если i-ая и j-ая вершины не являются соседними, то $a_{ij} = a_{ji} = 0$.

Разреженность матрицы

Если каждая вершина соседствует не более чем с m другими вершинами, то среди N^2 элементов матрицы имеется лишь mN ненулевых элементов. Такие матрицы называются *разреженными* и последние лет 40 развития линейной алгебры были направлены на изобретение особых алгоритмов работы с такими матрицами [5].



▸ Показать

Ответ

Остается написать ответ или в виде суммы

$$u = \sum_{i=1}^{N} u_i \varphi_i(x, y),$$

или приняв u_i за найденные значения решения в вершинах $\{A_i\}$.

Замечание. – Как отмечалось выше, часто интересно знать не саму функцию u, а ее максимум. Для его отыскания больше не надо ничего дифференцировать, просто

$$u_{\max} = \max_{i=1..N} u_i.$$

Пример.

Краевая задача в эллипсе.

Постановка задачи.

Рассмотрим краевую задачу

$$\left\{ \Delta u + e^{-x^2-y^2} = 0, u|_C = 0 \right.$$

в уже знакомом нам эллипсе C. Специфической особенностью `FreeFem++` является необходимость задавать уравнение в слабой форме (2), т.е. в данном случае

$$\int\limits_\Omega dxdy(\nabla u, \nabla v) - \int\limits_\Omega dxdy e^{-x^2-y^2} v = 0$$

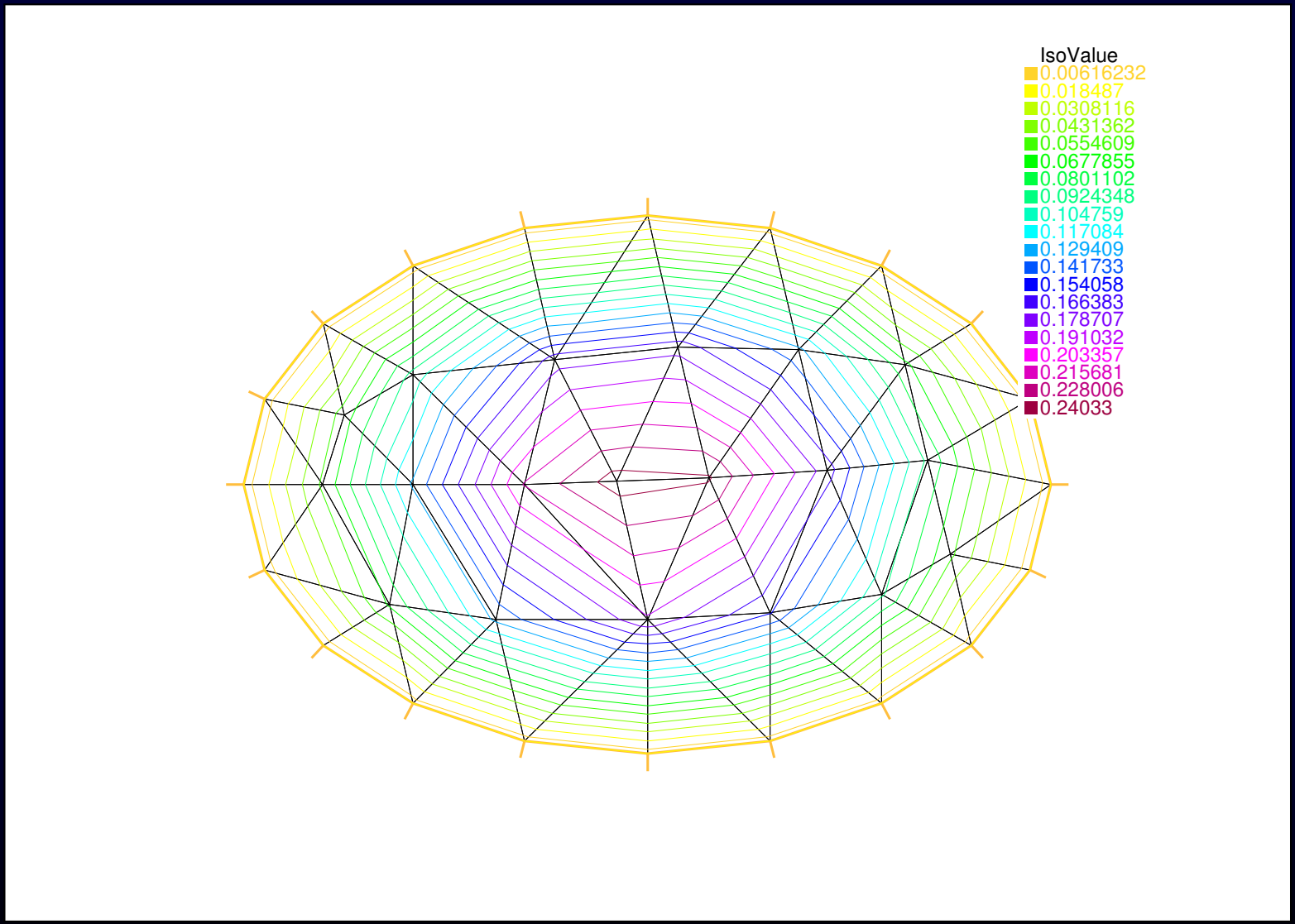
для любой функции v, обращающейся в нуль на границе C.

Решение задачи в FreeFem++.

Файл Poisson-3.edp

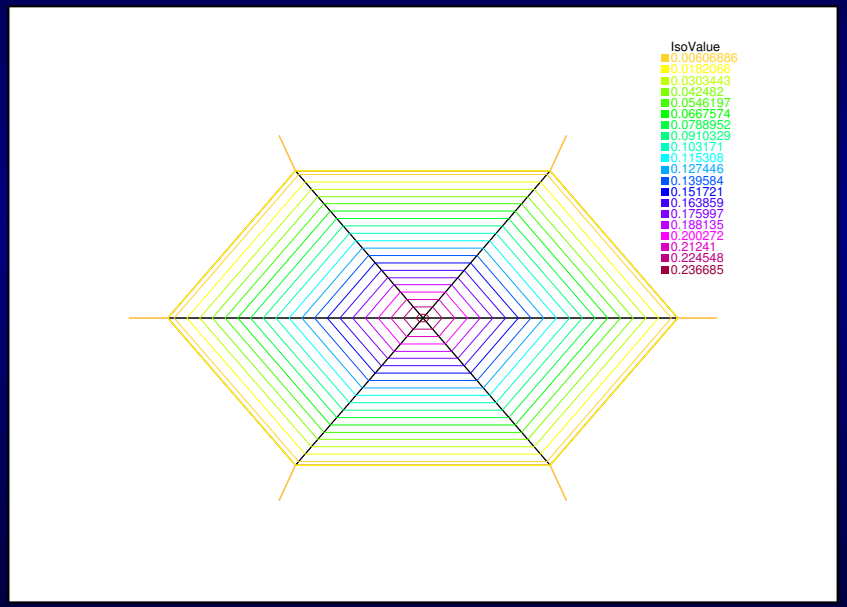
```
border C(t=0,2*pi){x=1.5*cos(t); y=sin(t);}
mesh Th = buildmesh (C(20));
fespace Vh(Th,P1);
Vh u,v;
solve Poisson(u,v,solver=LU) =
 int2d(Th)(dx(u)*dx(v) + dy(u)*dy(v))
 + int2d(Th)( -exp(-x^2-y^2)*v)
 + on(C,u=0);
plot(Th,u,ps="Poisson-3.eps",value=true,fill=false);
```

Ответ: файл Poisson-3.eps



Какую СЛАУ решает FreeFem++ на самом деле?

Для обозримости возьмем не 20, а всего 6 точек на границе. По идеи, в этом случае имеется 1 базовая функция и, следовательно, СЛАУ сводится к одному уравнению.



Вывод СЛАУ на печать

Файл Poisson-4.edp

```
border C(t=0,2*pi){x=1.5*cos(t); y=sin(t);}
mesh Th = buildmesh (C(6));
fespace Vh(Th,P1);
Vh u,v;
varf a(u,v) = int2d(Th)( dx(u)*dx(v) + dy(u)*dy(v))
+ on(C,u=0) ;
matrix A=a(Vh,Vh);
varf l(unused,v) = int2d(Th)(exp(-x^2-y^2)*v)+on(C,unused=0);
Vh f; f[] = l(0,Vh);
u[]=A^-1*f[];
cout << "A = " << A << endl;
cout << "f = " << f[] << endl;
cout << "u = " << u[] << endl;
plot(Th,u,ps="Poisson-4.eps",value=true,fill=false);
```

СЛАУ в FreeFem++.

$$\begin{pmatrix} 10^{30} & -0.4 & -0.4 & -0.1 & & & \\ -0.4 & 10^{30} & 0 & -0.9 & -0.1 & & \\ -0.4 & 0 & 10^{30} & -0.9 & 0 & -0.1 & \\ -0.1 & -0.9 & -0.9 & 3.7 & -0.9 & -0.9 & -0.1 \\ 0 & -0.1 & 0 & -0.9 & 10^{30} & 0 & -0.4 \\ 0 & 0 & -0.1 & -0.9 & 0 & 10^{30} & -0.4 \\ 0 & 0 & 0 & -0.1 & -0.4 & -0.4 & 10^{30} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0.9 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Реализация условий Дирихле в FreeFem++.

FreeFem++ ищет решение в виде

$$u = \sum u_i \varphi_i,$$

распространив сумму и на грачиные вершины с № 1-3, 5-7, но увидев +on(C,u=0), для этих номеров заполняет диагональ матрицы $\mathfrak{A}$ очень большим числом 10^{30}, а $\mathfrak{f}$ – нулями. Разумеется 6 уравнений (при $i \neq 4$) дают

$$10^{30} u_i + \cdots = 0 \quad \Rightarrow \quad u_i = 0,$$

а оставшееся

$$3.7 u_4 = 0.9 \quad \Rightarrow \quad u_4 = 0.24.$$

tgv в FreeFem++.

В документации ([2], стр. 16) это очень большое число так и называется tgv (très grande valeur), по умолчанию оно равно 10^{30}, но при желании его можно переопределить.
Такой неожиданный способ реализации условий Дирихле был избран из-за того, что разработчики не захотели писать два разных блока для условий Дирихле и условий Неймана. При решении задач с небольшим числом элементов он, конечно, весьма неэкономичен, а в задачах на собственные значения и вовсе приводит к появлению лишних собственных значений.

Домашняя работа

Выполненную домашнюю работу следует собрать в один pdf-файл и послать по адресу mmph@narod.ru, указав в теме письма номер группы. В ответ придут замечания и комментарии. После одной такой итерации работа будет *опубликована* на сайте http://mmph.narod.ru.

Домашняя задача № 1

Используя любую понравившуюся систему конечно-элементного анализа, решите краевую задачу в круге:

$$\begin{cases} \Delta u = \sin x \sin y, \quad (r < 1) \\ u|_{r=1} = 0 \end{cases}$$

Ответ представьте в виде графика. Что меняется с увеличением числа треугольников?

Домашняя задача № 2

Докажите след. утверждение: если u дважды непрерывно дифференцируема в замыкании области Ω и удовлетворяет условию (2) при всех гладких v, обращающихся в нуль на границе Ω, то она удовлетворяет и уравнению Пуассона $\Delta u = f$.

Ссылки

- *Деклу*. Метод конечных элементов. М., 1976.
- *Hecht F.* `FreeFem++`. Third Edition, Version 3.19. Laboratoire Jacques-Louis Lions, Université Pierre et Marie Curie, Paris, 2012.
- *Жуков М.Ю., Ширяева Е.В.* Пакет конечных элементов `FreeFem++`. Ростов-на-Дону, 2010
- *Скворцов А.В.* Триангуляция Делоне и ее применение. Томск: Изд-во Том. ун-та, 2002.
- *Yousef Saad*. Iterative Methods for Sparse Linear Systems. Самиздат, 2000.

Конец семинара № 7