Глава 3

Решение задач для уравнения Лапласа

Цель настоящей главы — демонстрация использования FreeFem++ для решения 2D стационарной краевой задачи, не приводя подробные объяснения. Наиболее просто это сделать для неоднородного уравнения Лапласа с различными краевыми условиями и в областях различной формы. Выбор для рассмотрения именно уравнения Лапласа не случаен. Во-первых, стационарная краевая задача для уравнения Лапласа является одной из наиболее часто встречающихся в курсе уравнений математической физики. Во-вторых, при помощи этой задачи описывается множество различных стационарных физических процессов, таких как стационарное распределение температуры, распределение концентрации, распределение электрического потенциала, распределение скорости для потенциального течения несжимаемой жидкости и т. п. В-третьих, на примере этой задачи легко показать простейшие возможности FreeFem++ — запись задачи в слабой формулировке, способы задания различных краевых условий, способы задания двумерной области, выбор конечных элементов, способы визуализации решения.

3.1 Задача о стационарном распределении температуры

3.1.1 Постановка задачи

Рассмотрим задачу о стационарном распределении температуры $u(x,y)$ в некоторой двумерной области D, на фрагментах границы которой задана температура g_1, поток тепла g_2 и условие теплопередачи Ньютона (краевое условие третьего рода).

Пусть дано неоднородное уравнение Лапласа

$$-\Delta u = f(x,y), \quad (x,y) \in D, \qquad \Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}. \tag{3.1}$$

Поставим следующие краевые условия:

$$u\big|_{\Gamma_1} = g_1, \tag{3.2}$$

$$\left.\frac{\partial u}{\partial n}\right|_{\Gamma_2} = g_2, \tag{3.3}$$

$$\left.\left(\frac{\partial u}{\partial n} + \beta u\right)\right|_{\Gamma_3} = g_3. \tag{3.4}$$

Здесь $f(x,y)$ — функция, заданная в области D; $g_1(x,y)$, $g_2(x,y)$, $g_3(x,y)$ — функции, заданные на границе области D, β — заданный параметр.

Напомним, что (3.2) называется неоднородным условием Дирихле, (3.3) — неоднородным условием Неймана, а (3.4) называется неоднородным условием Робина (или Фурье, или Ньютона, или смешанным условием).

Форма области D может быть любой, однако предполагается, что граница области $\partial D = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$ (возможно несвязная) является достаточно гладкой. Для определенности, считаем, что область $D = (0,a) \times (0,b)$ и фрагмент границы Γ_1 является несвязным: $\Gamma_1 = \Gamma_{11} \cup \Gamma_{12}$ (см. рис. 3.1).

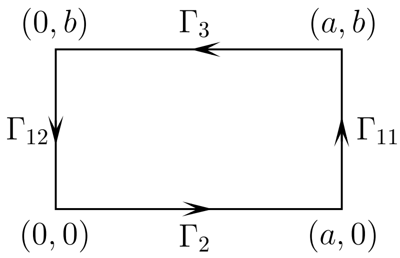

Рис. 3.1. Область $\overline{D}$. Прямоугольник $[0,a] \times [0,b]$

3.1.2 Слабая формулировка задачи

Основная причина, по которой следует переходить от сильной к слабой формулировке задачи — язык FreeFem++ (и метод конечных элементов) ориентирован на решение задач именно в слабой формулировке. Достаточно подробно способ такого перехода для задачи, аналогичной рассматриваемой, описан в гл. 2, где также имеются определения, вводится используемая терминология и т. п. Однако, для дальнейшего изучения материала знакомство с гл. 2 необязательно и приводимые ниже преобразования можно рассматривать как набор формальных действий.

Запишем задачу (3.1)–(3.4) в слабой (вариационной) формулировке. Для этого умножим (3.1) на тестовую (пробную) функцию $v(x,y)$ и проинтегрируем по области D

$$-\iint\limits_D v\Delta u\,dx\,dy = \iint\limits_D vf\,dx\,dy.$$

Используя формулу Грина, получим слабую формулировку исходной задачи (3.1)–(3.4)

$$\iint\limits_D \nabla v \cdot \nabla u\,dx\,dy - \int\limits_\Gamma v\frac{\partial u}{\partial n}\,ds - \iint\limits_D fv\,dx\,dy = 0, \quad \forall\, v(x,y). \tag{3.5}$$

Здесь $\partial u/\partial n$ — производная по внешней нормали к границе области D (по поводу обозначений см., в частности, (2.3)–(2.5)).

Ввиду того, что краевые условия (3.2)–(3.4) на фрагментах контура Γ заданы различными (см. также рис. 3.1), перепишем (3.5) в виде

$$\iint\limits_D \nabla v \cdot \nabla u \, dx \, dy - \int\limits_{\Gamma_1} v \frac{\partial u}{\partial n} \, ds - \int\limits_{\Gamma_2} v \frac{\partial u}{\partial n} \, ds - \int\limits_{\Gamma_3} v \frac{\partial u}{\partial n} \, ds - \iint\limits_D f v \, dx \, dy = 0. \quad (3.6)$$

Преобразуем интегралы по контурам Γ_2, Γ_3, исключая производные с учетом краевых условий (3.3)–(3.4). Для контура Γ_3 с учетом краевого условия (3.4) выводим

$$-\int\limits_{\Gamma_3} v \frac{\partial u}{\partial n} \, ds = -\int\limits_{\Gamma_3} v \left(g_3 - \beta u \right) \, ds. \quad (3.7)$$

Аналогично, для контура Γ_2 с учетом краевого условия (3.3) имеем

$$-\int\limits_{\Gamma_2} v \frac{\partial u}{\partial n} \, ds = -\int\limits_{\Gamma_2} v g_2 \, ds. \quad (3.8)$$

По терминологии метода конечных элементов (см., в частности, п. 1.4) краевые условия (3.3), (3.4) являются *естественными краевыми условиями*. Грубо говоря, эти краевые условия могут быть выполнены за счет требования обращения в нуль интегралов по контуру (после исключения производных) для *всех произвольных* тестовых функций $v(x, y)$.

Интеграл по контуру Γ_1 не может быть преобразован подобным образом, т. к. условие (3.2) не содержит производных функции $u(x, y)$. Краевое условие (3.2) является *главным краевым условием*. Для того, чтобы краевое условие было выполнено, необходимо накладывать дополнительные требования на функции $v(x, y)$.

Дополнительное ограничение на функции $v(x, y)$ в данном случае аналогично *однородному* условию (3.2) и имеет вид

$$v\big|_{\Gamma_1} = 0. \quad (3.9)$$

Окончательно, исходная задача (3.1)–(3.4) в слабой формулировке с учетом (3.6)–(3.8) принимает вид

$$\iint\limits_D \nabla u \cdot \nabla v \, dx \, dy \ - \ \iint\limits_D f v \, dx \, dy -$$

$$- \int\limits_{\Gamma_1} v \frac{\partial u}{\partial n} \, ds \ - \ \int\limits_{\Gamma_2} g_2 v \, ds - \int\limits_{\Gamma_3} \left(g_3 - \beta u \right) v \, ds = 0, \quad \forall \, v$$

или с учетом (3.9)

$$\iint\limits_D \nabla u \cdot \nabla v \, dx \, dy - \int\limits_{\Gamma_2} g_2 v \, ds - \int\limits_{\Gamma_3} (g_3 - \beta u) \, v \, ds - \iint\limits_D f v \, dx \, dy = 0, \quad (3.10)$$

$$\forall \, v\big|_{\Gamma_1} = 0.$$

3.1.3 Слабая формулировка задачи на языке FreeFem++

Прежде чем приводить коды на языке FreeFem++, напомним покоординатную форму записи следующего интеграла

$$\iint\limits_D \nabla v \cdot \nabla u \, dx\, dy = \iint\limits_D \left(\frac{\partial v}{\partial x}\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\frac{\partial u}{\partial y} \right) dx\, dy. \tag{3.11}$$

Удобно также использовать обозначения

$$\frac{\partial}{\partial x} = \partial_x, \quad \frac{\partial}{\partial y} = \partial_y. \tag{3.12}$$

С учетом (3.12) формула (3.11) запишется в виде

$$\iint\limits_D \nabla v \cdot \nabla u \, dx\, dy = \iint\limits_D (\partial_x v\, \partial_x u + \partial_y v\, \partial_y u)\, dx\, dy. \tag{3.13}$$

Приведем схему, при помощи которой задача в слабой формулировке (3.10) записывается на языке FreeFem++

$\iint\limits_D \nabla v \cdot \nabla u \, dx\, dy \quad \to$ `int2d(Th)( dx(u)*dx(v) + dy(u)*dy(v) )`

$-\int\limits_{\Gamma_2} g_2 v\, ds \quad \to$ `-int1d(Th,Gamma2)( g2 * v )`

$-\int\limits_{\Gamma_3} (g_3 - \beta u) v\, ds \quad \to$ `+int1d(Th,Gamma3)(beta*u*v)-int1d(Th,Gamma3)(g3*v)`

$-\iint\limits_D f v\, dx\, dy \quad \to$ `-int2d(Th)(f * v)`

$-\int\limits_{\Gamma_1} v \frac{\partial u}{\partial n}\, ds \quad \to$ `on(Gamma1, u = g1)`

Таким образом, интегралы, содержащиеся в выражении (3.10), практически дословно переписываются в кодах языка FreeFem++. Может быть следующая схема соответствий является даже более удобной

$\iint\limits_D (\dots)\, dx\, dy \quad \to$ `int2d(Th)( ... )`

$\int\limits_{\Gamma_i} (\dots)\, ds \quad \to$ `int1d(Th,GammaI)( ... )`

$\partial_x(\dots)\, \partial_x(\dots) \quad \to$ `dx( ... )*dx( ... )`

$\partial_y(\dots)\, \partial_y(\dots) \quad \to$ `dy( ... )*dy( ... )`

Интуитивно понятно, что `Th` в выражении `int2d(Th)` обозначает область интегрирования (более точно, `Th` — триангуляция области D, см.

ниже), `(Th, GammaI)` в выражении `int1d(Th,GammaI)` указывает область и фрагмент ее границы, по которой проводится интегрирование.

По поводу записи интегралов по границе сделаем важное замечание.

✓. *Интегралы* `int1d` *в случае фрагмента границы* Γ_3 *нельзя записывать в виде одного интеграла. Дело в том, что один из них, содержащий произведение* v *на известную функцию, является линейной формой относительно тестовой функции* v*, а другой, содержащий произведение* uv*, является билинейной формой относительно функций* u, v*. Язык FreeFem++ не позволяет смешивать эти понятия (выражения).*

Также должно быть ясно, что краевое условие (3.2) требует специальной формы записи. В подынтегральном выражении для интеграла по границе Γ_1 производная $\partial u/\partial n$ не может быть исключена при помощи каких-либо преобразований. Именно поэтому краевое условие (3.2) задается непосредственно в том виде, как оно записано, при помощи ключевого слова `on` в форме: `on(Gamma1, u = g1)`.

Фрагмент программы для решения задачи на языке FreeFem++ выглядит следующим образом (см. также полный текст программы, приведенный ниже на с. 43)

```
solve Poisson(u,v) = int2d(Th)( dx(u)*dx(v) + dy(u)*dy(v) )
                   - int1d(Th,Gamma2)( g2 * v )
                   + int1d(Th,Gamma3)( beta/alpha * u*v )
                   - int1d(Th,Gamma3)( g3/alpha * v )
                   - int2d(Th)( f * v )
                   + on( Gamma11, u = g1 )
                   + on( Gamma12, u = g1 ) ;
```

Обратим внимание, что нумерация строк использована для удобства ссылок и при использовании кода должна быть удалена, т. к. FreeFem++ ***не требует*** нумерации строк.

3.1.4 Задание области D на языке FreeFem++

Для того, чтобы определить область D в FreeFem++ достаточно указать границу области D. Это возможно сделать при помощи параметрического задания отдельных частей границы с использованием ключевого слова `border`. Приведем пример задания области D в случае прямоугольника $[0, a] \times [0, b]$ (см. рис. 3.1)

```
border Gamma2  (t=0,1) { x=a*t;     y=0; };
border Gamma11 (t=0,1) { x=a;       y=b*t; };
border Gamma3  (t=0,1) { x=a*(1-t); y=b; };
border Gamma12 (t=0,1) { x=0;       y=b*(1-t); };
```

Граница $\Gamma_1 = \Gamma_{11} \cup \Gamma_{12}$ предполагается несвязной. Именно поэтому для каждого ее связного фрагмента использованы различные идентификаторы `Gamma11` и `Gamma12`.

Интуитивно понятно, что, например, строка 6 соответствует параметрической форме записи для отрезка прямой линии $x \in [0, a]$, $y = 0$

$$x = at, \quad y = 0, \quad 0 \leqslant t \leqslant 1.$$

Заметим, что порядок определения фрагментов границы не имеет значения, т. е. строчки с операторами `border` в приведенном выше фрагменте программы могут быть переставлены. Однако важно, чтобы при возрастании параметра t обход области D совершался в направлении ***против часовой стрелки***.

3.1.5 Полный код программы на языке FreeFem++

Зададим для определенности следующие функции и параметры для исходной задачи (3.1)–(3.4)

$$\overline{D} = \big\{(x, y)\colon [0, a] \times [0, b]\big\}, \quad a = 1, \quad b = 1,$$

$$f(x, y) = \sin 2\pi x \, \sin 2\pi y, \quad g_1 = 0, \quad g_2 = 1, \quad g_3 = 1, \quad \beta = 1. \tag{3.14}$$

Приведем полный код программы для построения численного решения задачи

```
real a=1.0;  // ширина области D
real b=1.0;  // высота области D
int n=4;     // вспомогательный параметр для триангуляции области D
// задание границ области D (прямоугольник [0,a]x[0,b])
// сохраняем ориентацию контура -- против часовой стрелки
border Gamma2(t=0,1){ x=a*t; y=0; };          // bottom
border Gamma11(t=0,1){ x=a; y=b*t; };         // right
border Gamma3(t=0,1){ x=a*(1-t); y=b; };      // top
border Gamma12(t=0,1){ x=0; y=b*(1-t); };     // left
// построение сетки Th, каждый фрагмент границы разбит на 5*n отрезков
mesh Th = buildmesh(Gamma2(5*n)+Gamma11(5*n)+Gamma3(5*n)+Gamma12(5*n));
// plot(Th,wait=1);  // для визуализации построенной сетки
// задание пространства конечных элементов
fespace Vh(Th,P2);
// задание на пространстве Vh искомой функции u и пробной функции v
Vh u,v;
// определение функций и параметров исходной задачи
func f = sin(2*pi*x)*sin(2*pi*y);
func g1 = 0;
func g2 = 1;
func g3 = 1;
real beta=1;
// запись слабой (вариационной) формулировки задачи и ее решение
solve Poisson(u,v) = int2d(Th)(  dx(u)*dx(v) + dy(u)*dy(v) )
                     - int1d(Th,Gamma2)( g2 * v )
                     + int1d(Th,Gamma3)( beta * u*v )
                     - int1d(Th,Gamma3)( g3 * v )
                     - int2d(Th)( f * v )
```

```
29                      + on( Gamma11, u = g1 )
30                      + on( Gamma12, u = g1 ) ;
31 plot(u);  // графическое представление решения
```

Результат решения задачи (3.1)–(3.4) представлен на рис. 3.2, на котором изображены линии уровня (изолинии или изотермы, если u температура) функции $u(x, y)$. Заметим, что на экране дисплея изолинии будут цветными. Записав вместо 31-ой строки `plot(u, value=1)`, можно получить таблицу соответствия цвета изолинии и значения функции $u(x, y)$ на этой изолинии. Для сохранения результатов расчетов в файл следует использовать строку `plot(u, value=1, ps="FileName")`. При этом рисунок будет сохранен в postcript-формате (.ps).

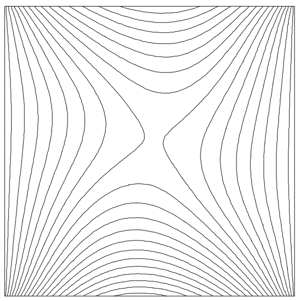

Рис. 3.2. Изолинии функции $u(x, y)$ (температуры) — решение задачи (3.1)–(3.4)

Сделаем некоторые пояснения:

1) `Th` — идентификатор генерируемой сетки. Можно задавать и другое имя, например, `Th2`, однако, в этом случае в `int2d(Th)`, `int1d(Th,GammaI)` и т. п. следует заменить `Th` на `Th2`.

2) `Vh(Th,P2)` — идентификатор пространства конечных элементов. Имя `Vh` может быть любым, `Th` — должен совпадать с именем генерируемой сетки, `P2` — наименование типа конечных элементов (зарезервированное слово в FreeFem++, подробнее см. гл. 18).

3) `solve Poisson(u,v)` — имя `Poisson` задается пользователем и может быть любым. Ключевое слово `solve` означает, что задача формулируется и одновременно решается (см. также о ключевом слове `problem` в п. 19.1).

На этом этапе изучения языка FreeFem++ можно не задумываться над строками 10–14 программы. Интуитивно понятно, что при помощи этих строк задается триангуляция области D и выбирается способ аппроксимации решения некоторыми конечноэлементными функциями. Заметим только, что использованный тип конечных элементов (`P2`) соответствует аппроксимации кусочно-непрерывными квадратичными функциями (подробнее см. гл. 18).

3.2 Решение задачи о распределении температуры в областях сложной формы

Одной из важных особенностей метода конечных элементов и, в частности, языка FreeFem++ является возможность решать задачу в области