Федеральное государственное автономное образовательное учреждение высшего образования«Московский физико-технический институт (национальный исследовательский университет)»

Решение краевой задачи методом Шварца.

Выполнили студенты:
Сериков Василий Романович
Сериков Алексей Романович
Данилов Иван Владимирович
группа: Б03-102

Москва, 2024 г.

Цель работы:

Решить методом Шварца краевую задачу, описывающую стационарный процесс.

Теория:

Имеем краевую задачу:

$$Lu = -\operatorname{div}(k \operatorname{grad} u) + qu = \nabla\kappa\nabla u + qu = f$$

С граничными условиями Неймана и Дирихле:

$$u|_{\Gamma_D} = \Phi$$
$$-\vec{n}\cdot\kappa\nabla u + \alpha u|_{\Gamma_\kappa} = \Psi$$

Где, заданная область $\Omega = \Omega_1 + \Omega_2 = (-1;0)\text{x}(1;-2) + (0;1)\text{x}(0;1)$, f$-$ непрерывна в Ω, $\kappa = \kappa^T > 0$, решение ищем, как непрерывную функцию в $C^2(\Omega)$.
А для граничных условий выполняется следующие:

$$\Gamma_D = \overline{\Gamma_D} \subset \partial\Omega,$$
$$\Gamma_N = \partial\Omega - \Gamma_D;$$

Метод Шварца заключается в разбиении исходной области на подобласти и решении задачи в каждой подобласти с граничными условиями, полученными из предыдущей итерации.

На общей границе Γ между Ω_1 и Ω_2 мы будем использовать фиктивные граничные условия, которые будут обновляться на каждой итерации.

Задаем начальное приближение для u на Γ, например, $u^0 = 0$

Решаем задачу в Ω_1 с граничными условиями Дирихле на $\partial\Omega_1 \setminus \Gamma$ и условием $u = u^{k-1}$ на Γ. Решаем задачу в Ω_2 с граничными условиями Дирихле на $\partial\Omega_2 \setminus \Gamma$ и условием $u = u^{k-1}$ на Γ.

Обновляем фиктивные граничные условия на Γ используя решения в Ω_1 и Ω_2. Например, можно взять среднее значение: $u^k = \frac{1}{2}(u_{\Omega_1} + u_{\Omega_2})$.

Проверка сходимости: Проверяем, насколько изменилось решение на текущей итерации. Если изменение меньше заданной точности, то алгоритм завершается. В противном случае, повторяем снова итерацию.

Общий вид решения на каждой итерации будет представлять собой комбинацию решений в Ω_1 и Ω_2 с учетом граничных условий на Γ.

Решим упрощенную задачу, используя язык программирования FreeFem++.

$$\begin{cases} -\Delta u = f \text{ в } \Omega = \Omega_1 \cup \Omega_2 \\ u|_\Gamma = 0 \end{cases}$$

Введем Γ_i - общую границу Ω_1 и Ω_2, а $\Gamma_e^i = \partial\Omega_i \setminus \Gamma_i$. Задача состоит в том, чтобы найти такое λ, что $(u_1|\Gamma_i = u_2|\Gamma_i)$, где u_i - решение следующей задачи Лапласа:

$$\begin{cases} -\Delta u_i = f \text{ в } \Omega_i \\ u_i|\Gamma_i = \lambda \\ u_i|\Gamma_e^i = 0 \end{cases}$$

Чтобы решить эту задачу, мы просто создаем цикл с обновлением λ с помощью:

$$\lambda = \lambda \pm \frac{(u_1 - u_2)}{2}$$

где знак + или − выбирается для обеспечения сходимости. Примечания: $\partial\Omega_i$ обозначает границу области Ω_i. $u_i|_{\Gamma_i}$ означает значение функции u_i на границе Γ_i. Δ - оператор Лапласа.

```
// Schwarz without overlapping (Shur complenement Neumann -> Dirichet)

verbosity=2;
real cpu=clock();
int inside = 2;
int outside = 1;
border Gamma1(t=1,2){x=t;y=0;label=outside;};
border Gamma2(t=0,1){x=2;y=t;label=outside;};
border Gamma3(t=2,0){x=t ;y=1;label=outside;};

border GammaInside(t=1,0){x = 1-t; y = t;label=inside;};

border GammaArc1(t=1,-2){ x=0; y=t;label=outside;};
border GammaArc2(t=0, 1){ x=t; y=-2;label=outside;};
border GammaArc3(t=-2,0){ x=1; y=t;label=outside;};
int n=4;
mesh Th1 = buildmesh( Gamma1(5*n) + Gamma2(5*n) + GammaInside(5*n) + Gamma3(5*n));
mesh Th2 = buildmesh ( GammaInside(-5*n) + GammaArc1(5*n) +  GammaArc2(5*n) +
   GammaArc3(5*n));

plot(Th1,Th2);

fespace Vh1(Th1,P1),      Vh2(Th2,P1);
Vh1 u1=0,v1;              Vh2 u2,v2;
Vh1 lambda=0;  // take $\lambda \in V_{h1}$

int i=0;  // for factorization optimization


problem Pb2(u2,v2,init=i,solver=Cholesky) =
    int2d(Th2)( dx(u2)*dx(v2)+dy(u2)*dy(v2) )
  + int2d(Th2)( -v2)
  + int1d(Th2,inside)(-lambda*v2)
  + on(outside,u2= 0 ) ;
problem Pb1(u1,v1,init=i,solver=Cholesky) =
    int2d(Th1)( dx(u1)*dx(v1)+dy(u1)*dy(v1) )
```

```
  + int2d(Th1)( -v1)
  + int1d(Th1,inside)(+lambda*v1) +     on(outside,u1 = 0 ) ;

varf b(u2,v2,solver=CG) =int1d(Th1,inside)(u2*v2);
matrix B= b(Vh1,Vh1,solver=CG);

//  $\lambda \longrightarrow  \int_{\Gamma_i }(u_1-u_2) v_{1}$
func real[int] BoundaryProblem(real[int] &l)
{
   lambda[]=l;
   Pb1;
   Pb2;
   i++;
   v1=-(u1-u2);
   lambda[]=B*v1[];
   return lambda[] ;
};

Vh1 p=0,q=0;

//  solve the problem with Conjugue Gradient
LinearCG(BoundaryProblem,p[],q[],eps=1.e-6,nbiter=100);

//  compute the final solution, because CG works with increment
BoundaryProblem(p[]); // solve again  to have right u1,u2
cout << " -- CPU time  schwarz-gc:" <<  clock()-cpu << endl;

plot(u1,u2);
```

Получили следующие результаты:

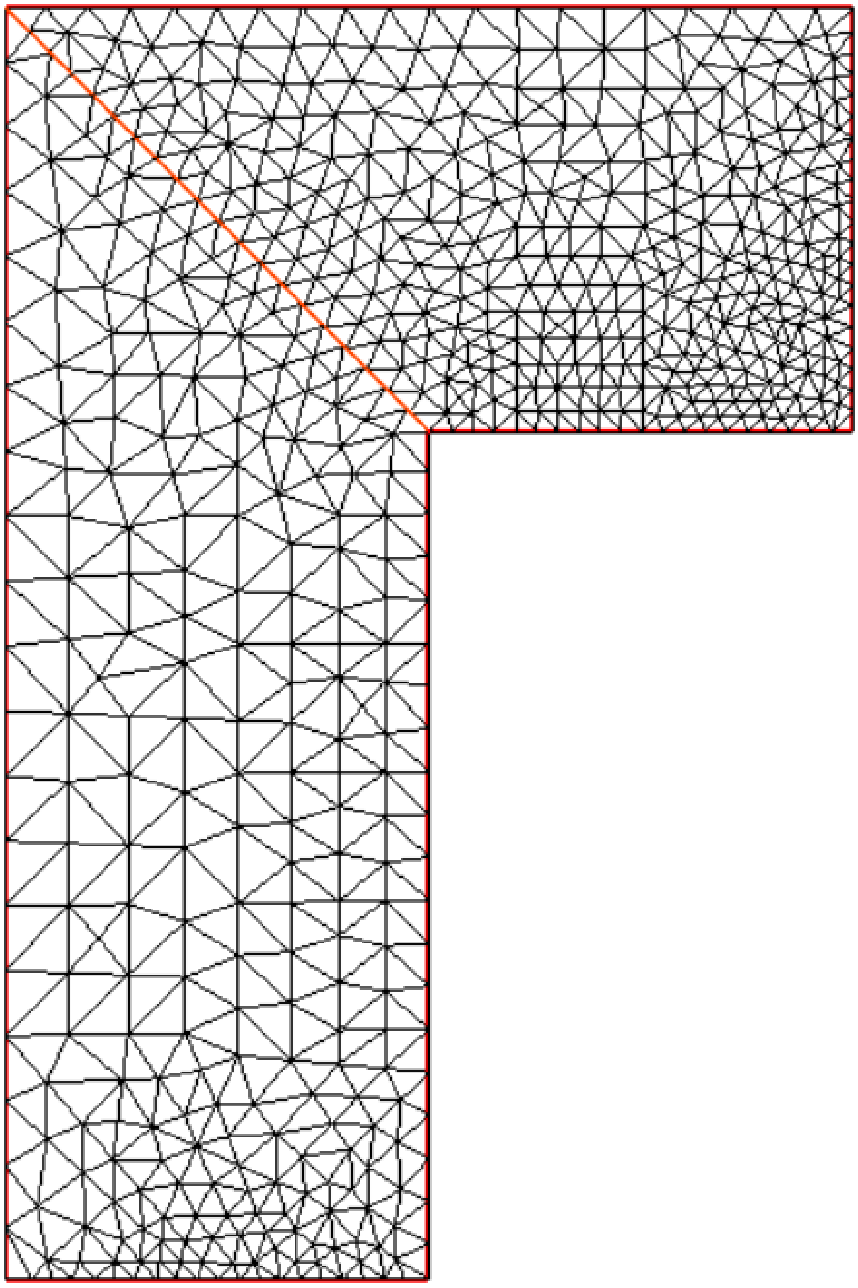

Рис. 1: Сгенерированная сетка для исходной области

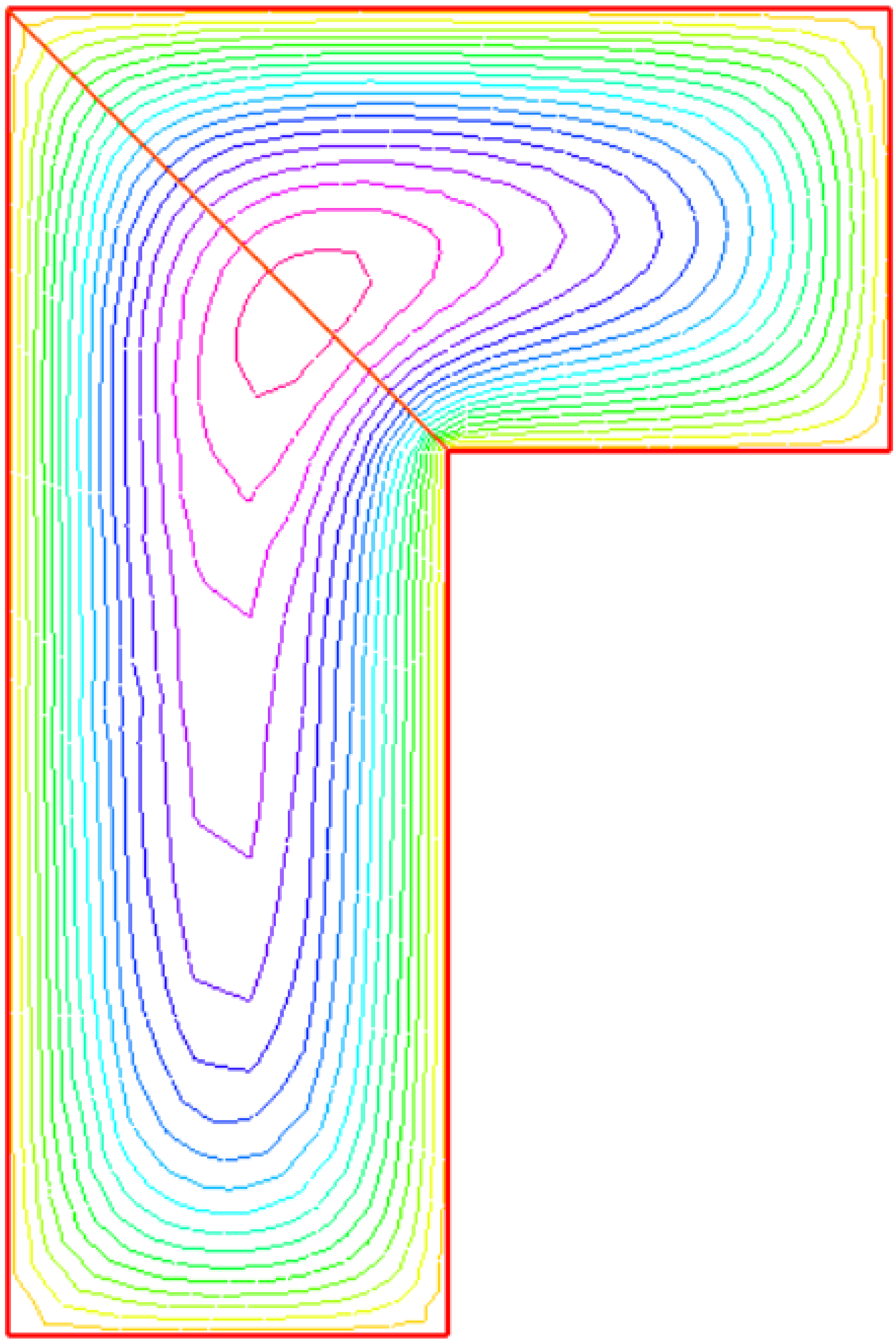

Рис. 2: Решение задачи на полученной сетке

Вывод:

В ходе работы мы познакомились с методом Шварца для решения краевой задачи и решили упрощенную задачу на области "сапог"с помощью языка программирования FreeFem++.