Санкт-Петербургский государственный политехнический университет

Курсовая работа

«Предобусловленный метод сопряженных градиентов для разреженных матриц»

Студент: Марков Михаил Денисович

Преподаватель: Добрецова Светлана Борисовна

2023

1. Задача: запрограммировать метод сопряженных градиентов с предобуславливателем на основе неполного разложения Холецкого. Учесть, что метод будет применяться к разреженным матрицам (в частности, к 5-ти диагональным). Для заданного уравнения Пуассона построить и решить СЛАУ. Исследовать метод по вариантам.

   Этапы решения

   1.1. Способ хранения разреженной матрицы. Реализация действий над матрицами и векторами

   1.2. Разложение Холецкого.

   А) Разложение Холецкого для разреженной матрицы

   Б) Неполное разложение Холецкого с нулевым порогом - факторизация, которая содержит ненулевые элементы только в тех позициях, что и исходная матрица.

   В) Неполное разложение Холецкого с ненулевым порогом – отсекаются только те элементы, которые меньше порогового значения, исключая диагональные

   1.3. Метод сопряженных градиентов

   А) Без предобуславливателя

   Б) С предобуславливателем

   4. Построить СЛАУ для модельной задачи по вариантам

   Вариант 9в, cos(x)cos(y) $[0,\pi]x[0,\pi]$ и Фактическая ошибка от заданной точности.

2. Используемые методы будут приводиться попунктно в соответствии с этапами решения. Реализация алгоритмов и построение графиков были реализованы на языке C++ и MATLAB соответственно.

   2.1. Способом хранения разреженных матриц был выбран массив списков. Список позволяет хранить только ненулевые элементы матрицы, в отличие от массива. Это большое преимущество в данной работе, так как матрицы разреженные – с большим количеством нулевых элементов.

   2.2. Разложение Холецкого было реализовано на языке C++. Оно представляет собой представление исходной матрицы A в виде произведение нижнетреугольной матрицы L и верхнетреугольной L^T : A = L * L^T .Для его применения необходима симметричная, положительно-определенная матрица. Ниже приведены формулы для разложения Холецкого, а именно пункта А:

   2.3.

   $l_{11} = sqrt(a_{11})$

   $l_{ji} = a_{ji} / l_{11}$, j = [2,n];

   $l_{ii} = sqrt\ (a_{ii} - \sum_p^{i-1} l_{ip} * l_{ip})$, i = [2, n];

   $l_{ji} = (a_{ji} - \sum_p^{i-1} l_{ip} * l_{jp}) / l_{ji}$, i = [2, n-1], j = [i+1 , n];

   Пункты Б и В отличаются лишь условиями при присвоении значений, описанными выше в этапах решения.

   2.4. Предобуславливание с использованием вышеописанного разложения Холецкого. Рассмотрим на примере решения СЛАУ A*x = b:

   2.3.1. Раскладываем исходную матрицу A на L и L^T

   2.3.2. Решаем систему L*q = b: q- вектор промежуточных значений

   2.3.3. Решаем систему L^T *x = q; x - искомый

Примечание – системы в пунктах 2.3.2. и 2.3.3. можно решать с использованием простого Гауссового прохода, так как матрицы являеются нижнетреугольной и верхнетреугольной соответственно.

2.5. Построение СЛАУ происходит на основе формулы об аппроксимации 2-й производной. Задача – численное решение задачи Дирихле для уравнения Пуассона, определяемую как задачу нахождения функции u = u(x,y), удовлетворяющей в области определения D уравнению

$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y)$, (x,y) = D^0

u(x,y) = g(x,y), (x,y) = D^0

и принимающей значения g(x,y) на границе D^0 области D (f и g являются функциями, задаваемыми при постановке задачи).

Для аппроксимации используется метод конечных разностей. Следуя этому подходу, область решения D представляется в виде дискретного (как правило, равномерного) набора (сетки) точек (узлов). Так, например, прямоугольная сетка в области D может быть задана в виде,

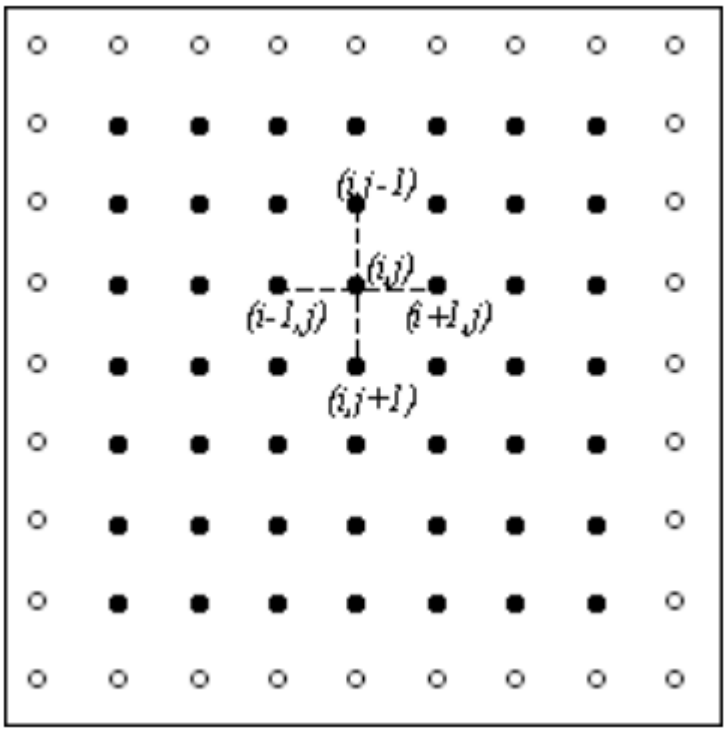
$D_h = \{(x_i, y_j): x_i = i*h, y_j = j*h, 0<=i,j<=N+1\}$
$h = 1/(N+1)$

Где величина N задает количество узлов по каждой из координат области D.

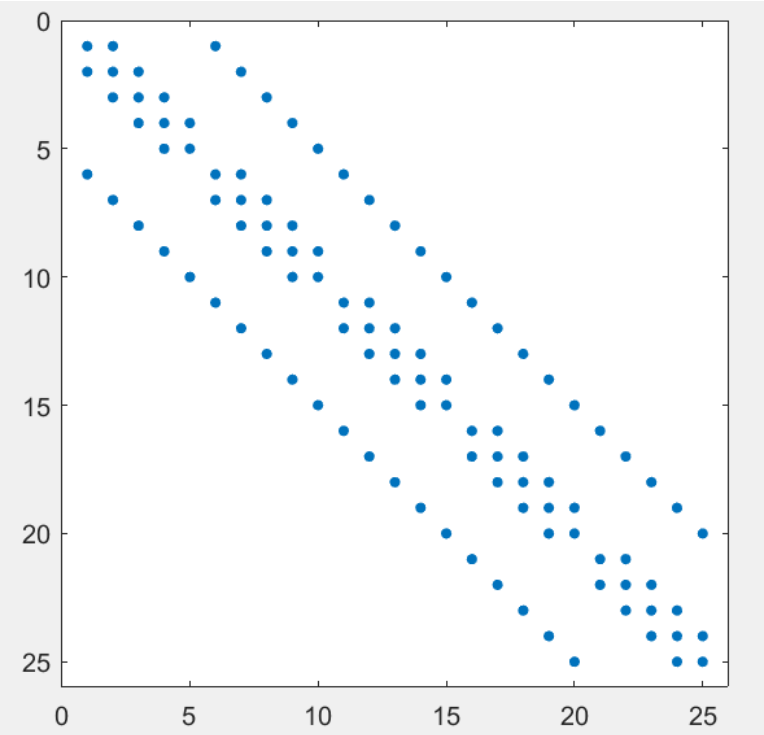
Тогда, используя пятиточечный шаблон для вычисления значений производных, уравнение Пуассона может быть представлено в конечно-разностной форме

$u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j-1} - 4u_{i,j} = h*h * f_{i,j}$

Разностное уравнение, записанное в подобной форме, позволяет определять значение u_{ij} по известным значениям функции u(x,y) в соседних узлах используемого шаблона.



По коэффициентам уравнения, разрешенного относительно u_{ij} , строится матрица. Она является пяти диагональной, так как количество точек, используемых для нахождения равно 5. Размер матрицы равен N^2 – количеству точек в квадрате. Блок в этой матрице имеет размер N на N. Пример представлен для N = 5:



Диагонали строятся таким образом. Матрица разбивается на блоки N на N. В тех блоках, диагональ которых совпадает с главной диагональю матрицы, выбираются их главная диагональ и 2 соседних – выше и ниже. На главной записывается 4, а на побочных -1. Это связано с коэффициентами уравнения,

разрешенного относительно u_{ij}. Далее относительно этого блока выбираются 2 соседних, диагональ которых уже не совпадает с главной. На их главной диагонали тоже записываются -1.
Вектор решений – “развернутая” сетка значений искомой функции u = u(x,y) .
Под разверткой я понимаю растягивание матрицы в строку построчно – сначала первая строка слева направо, затем вторая и т.д. Вектор правой части – значения заданной функции f(x,y), взятые с противоположным знаком. Однако, для тех точек, которые лежат на границе области, добавляются граничные значения, не учтенные в пяти диагональной матрице. Далее это СЛАУ решается вышеописанным методом, сопряженным градиентов, а также им с предобуславливанием на основе вышеописанных 3-х разложений Холецкого. По рассчитанной сетке значений строятся графики зависимости фактической точности от заданной.

3.1. Алгоритмы

3.1.1. Разложение Холецкого

На псевдо-алгоритмическом языке код разложения выглядит так:

```
DO  I = 1, N
        A(I,I) = SQRT (A(I, I))
        DO  J = I+1, N
                A(J,I) = A(J,I)/A(I,I)
        END DO
        DO  K=I+1,N
                DO J = K, N
                        A(J,K) = A(J,K) - A(J,I)*A(K,I)
                END DO
        END DO
END DO
```

Матрица строится на основе заданной. Здесь A – заданная матрица N на N. Здесь представлена версия для обычных матриц, но суть алгоритма неизменна в зависимости от типа матрицы.

3.1.2. Неполное разложение Холецкого

Принципиальное отличие состоит в том, что происходит проверка на нулевое значение в обычной матрице или на существование узла в строке в случае разреженной матрицы. В случае нахождения 0 или несуществования узла, значение не записывается – тем самым достигается условие о существовании элементов только на тех позициях, где есть ненулевое значение или существует узел.

3.1.3. Неполное разложение Холецкого с ненулевым порогом

Принципиальное отличие от разложения Холецкого заключается в проверке значения элемента. Если его значение меньше заданного, то значение в такой элемент не записывается. q

3.2.1. Метод сопряженных градиентов

Используемые переменные:

- I. Метод сопряженных градиентов
  1.1. Задать x0 и e > 0
  1.2. Вычислить вектор r(0) = b – Ax.
  1.3. Положить p(0) = r(0), k = 0.
  2.1. Вычислить вектор q(k) = Ap(k).
  2.2. Вычислить скаляр a(k) = (r(k), p(k)) / (q(k), p(k)).
  2.3. Вычислить вектор x(k + 1) = x(k) + a(k)p(k).
  2.4. Вычислить вектор r(k + 1) = r(k) – a(k)q(k).
  2.5. Проверить выполнение неравенства ||r(k+1)||<= e; если «да», остановить работы алгоритма и вывести результаты.
  3.1. Вычислить скаляр B(k) = (r(k+1), q(k)) / (p(k), q(k)).
  3.2. Вычислить вектор p(k + 1) = r(k+1) – B(k)p(k).
  3.3. Положить k:= k+1 и вернуться к шагу 2.1.
  II. Метод сопряженных градиентов с предобуславливателем.
  P – предобуславливатель.
  k=0: x0 – начальное приближение, r(0) = b – Ax0, Pz(0) = r(0), p(0) = r(0)
  k>=0: Пока ||r(k)||/||r(0)|| > e
  1. q(k) = Ap(k), a(k) = (z(k), z(k)) / (p(k), q(k)).
  2. x(k+1) = x(k) + a(k)p(k), r(k+1) = r(k) – a(k)q(k).
  3. Pz(k+1) = r(k+1)
  4. B(k) = (z(k+1), r(k+1)) / (z(k), r(k)), p(k+1) = r(k+1) + B(k)p(k).

3.3.1 Построение СЛАУ для модельной задачи

Построение правой части уравнения

```
for (int i = 1; i < N+1; i++)
	{
		x = A + i * h;
		for (int j = 1; j < N+1; j++)
		{
			value = 0;
			y = A + j * h;
			value-= h*h* f(x, y);
			if (i == 1)//проверка на границы и соотвуствующее изменение правой
части
				value += u(x - h, y );
			if (i == N)
				value += u(x + h, y );
			if (j == 1)
				value += u(x, y - h);
			if (j == N)
				value += u(x , y + h);
			f[i - 1][j - 1] = value;
		}

	}
```

Построение пяти диагональной матрицы по вышеописанному способу

Примечание: Sparse – тип разреженной матрицы, Add – метод, добавляющий элемент по заданной позиции и с выбранным значением.

```
int size = pow(N, 2);
Sparse* S = new Sparse(size);
for (int k = 0; k < size; k++)
{
	S->Add(k, k, 4);
	if (k < size - 1 && (k / N) == ((k+1) / N))
	{
		S->Add(k, k + 1, -1);
		S->Add(k + 1, k, -1);
	}
	if (k + N < size)
	{
		S->Add(k, k + N, -1);
		S->Add(k + N, k, -1);
	}

}
```

4. Модульная структура программы

Sparse.h

```
typedef struct List // структура, определяющая элемент разреженной матрицы - список
{
public:
	int column;
	double value;
	List* next;
	List();
	List(int column, double value);
} List;

typedef struct Sparse // массив списков – разреженная матрица
{
public:
	int n;
	vector<List*> rows;
	Sparse();
	Sparse(int n);
	Sparse(const Sparse& A1);

} Sparse;
```

Chol.h

Матрица A1 – исходная матрицу, которую необходимо разложить

```
Sparse iChol(Sparse& A1) // неполное разложение Холецкого

Sparse Chol(Sparse& A1) // разложение Холецкого

Sparse Chol(Sparse& A1, double tol) //неполное разложение Холецкого с ненулевым порогом
```

HeaderPCG.h

```
vector<double> Gauss1(Sparse& A, vector<double> b)// Решение СЛАУ Гауссовым проходом для нижнетреугольной матрицы

vector<double> Gauss2(Sparse& A, vector<double> b)// Решение СЛАУ Гауссовым проходом для верхнетреугольной матрицы
```

```
vector<double> Gauss(Sparse& L, Sparse& Lt, vector<double>b)// применение предобуславливания к вектору b посредством 2-х последовательных Гауссовых проходов

vector<double> PCG_Pred(Sparse& A, vector<double>& b, vector<double>& x_0, double epsilon, Sparse& L)// применение метода сопряженных градиентов с предобуславливанием к матрице с правой частью b, начальным приближением x_0, заданной точностью и матрицей из Разложения матрицы A по разложению Холецкого. Возвращает вектор значений x.

vector<double> PCG(Sparse& A, vector<double>& b, vector<double>& x_0, double epsilon) )// применение метода сопряженных градиентов к матрице с правой частью b, начальным приближением x_0, заданной точностью
```

Poisson.h

```
double Function_Value(double x, double y)// расчет значения функции f(x,y)

double U_Acc(double x, double y)// расчет значения функции u(x,y)

vector<vector<double>> F(double A, double B, int N)//создание правой части СЛАУ в виде матрицы

vector<double> f(double A, double B, int N)// представление матрицы из функции F в виде векторы для правой части СЛАУ

Sparse Diag_5(int N)// создание пяти диагональной матрицы
```

5. Тестовый пример:

Матрица A:

1.395 1.794 1.228 0.5212 1.296
1.794 2.965 1.892 1.309 2.079
1.228 1.892 1.936 1.036 1.768
0.5212 1.309 1.036 1.062 1.07
1.296 2.079 1.768 1.07 1.729

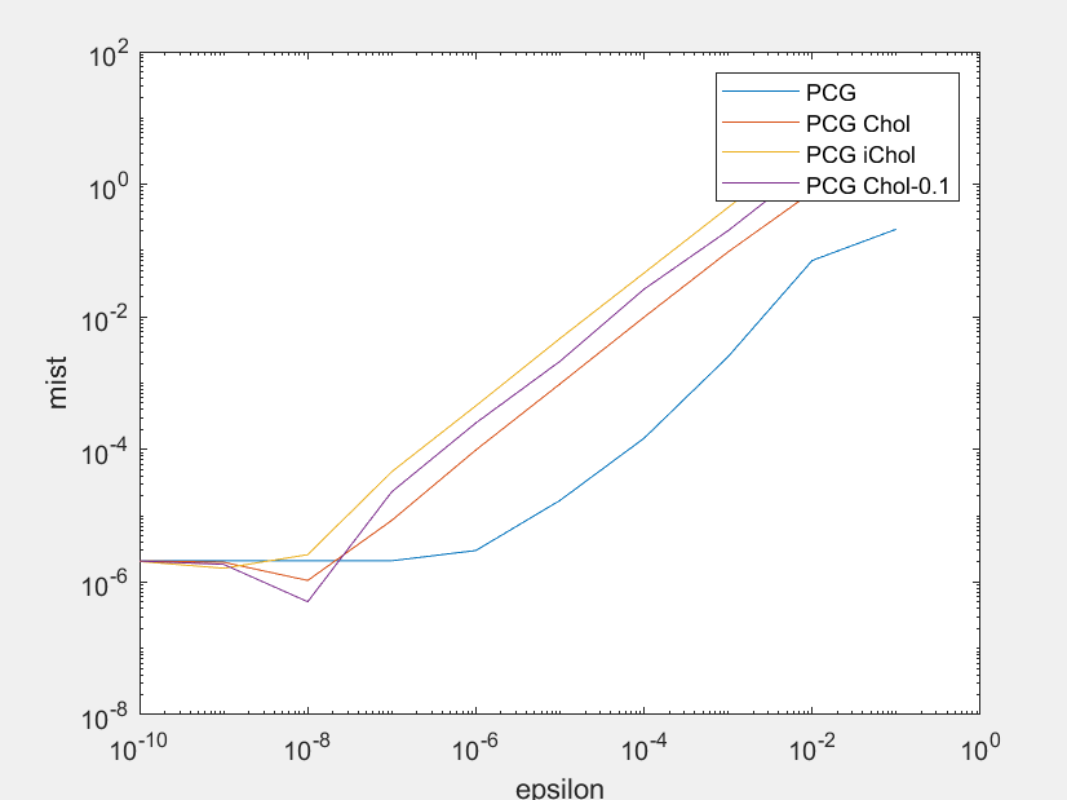
Матрица из разложения Холецкого:

1.181 0 0 0 0
1.519 0.8115 0 0 0
1.039 0.3869 0.8401 0 0
0.4412 0.7869 0.3246 0.378 0
1.097 0.5087 0.513 0.05079 0.005279

X_0: 1 0 0 0 0
z_0: 1 1 1 1 1
eps0: 4.839 8.245 6.633 4.477 6.646
p_0: 1 1 1 1 1
k: 0
q: 6.234 10.04 7.86 4.998 7.943
alpha: 1
x: 2 1 1 1 1
eps: -1.395 -1.794 -1.228 -0.5212 -1.296
z: -1 1.585e-15 -3.289e-16 -7.771e-16 0

mistake: 0.2079
beta: 0.04525
p: -1.35 -1.749 -1.183 -0.4759 -1.251
k: 1
q: -8.342 -13.07 -9.962 -6.061 -10.15
alpha: 0.1357
x: 1.817 0.7627 0.8395 0.9354 0.8302
eps: -0.2633 -0.02052 0.124 0.3013 0.08102
z: -0.8168 0.2373 0.1605 0.06459 0.1698
mistake: 0.03024
beta: 0.1887
p:-0.518 -0.3505 -0.0992 0.2115 -0.155
количество итераций :2
Вектор решений : 1.817 0.7627 0.8395 0.9354 0.8302
ошибка: 0.88446

6. Анализ графиков:



Это график зависимости фактической ошибки от заданной точности на промежутках x[0,pi/10], y[0,pi/10] для 30 разбиений - график ошибки сетки рассчитанной функции от эталонной и заданной точности для метода сопряженных градиентов. Как видно из графика, методы сходятся к одному значению. Я думаю, что это связано с сеточным разбиением - а именно аппроксимацией производных.

Фиксированным разбиением и длиной промежутка мы "задаем" шаг разбиения, который и определяет фактическую точность, а методом сопряженных градиентов мы лишь стремимся к нему и уточняем получаемые значения. Нет возможности посчитать сетку точнее из-за фиксированного числа разбиений. Уменьшив длину промежутка в 10 раз, мы достигли уменьшения шага в 10 раз без увеличения числа разбиений.

Как мы видим на графике, метод сопряженных градиентов с предобуславливателем, полученным из разложения Холецкого, дает фактическую точность меньше, чем неполное разложение с факторизацией и без нее. Это связано с тем, что разложение Холецкого является более точным по отношению к неполным. По этой же причине фактическая точность неполного с порогом выше, чем просто неполного.

7. Выводы

Однако, все представленные методы дают фактическую точность одного порядка. Тем не менее, метод сопряженных градиентов показывает немного лучшую точность, потому что число обусловленности пяти диагональной матрицы не велико. Также влияет изменение невязки внутри самого метода. В отсутствии предобуславливания, невязка может изменяться на соседних итерациях на несколько порядков, что не может гарантировать нам получение заданной точности, в отличие от методов с предобсуславливанием – в них невязка меняется равномерно.

Не имеет смысла делать точность метода сопряженных градиентов выше, чем точность апроксимации производной. Это вызвано тем, что используется 2 метода – конечных разностей и сопряженных градиентов. Но метод конечных разностей использует фиксированное число точек для аппроксимации, так что не имеет возможности получить точность лучше, даже используя повышение метода точности сопряженных градиентов.