МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики

Практическое задание по курсу лекций «Численные методы линейной алгебры»
Задание №2
Отчет о выполненном задании

студента 303 учебной группы факультета ВМК МГУ
Курбацкого Вячеслава Константиновича

Москва
2024

Содержание

1 Прямые и итерационные методы решения СЛАУ

1.1 Постановка задачи

Дана система линейных алгебраических уравнений:

$$x + Ax = F \Leftrightarrow (A + I)x = F$$

Причём матрица $A + I$ является симметричной и положительно определенной. Напомним, что матрица $B = B^T$ называется симметричной, а положительно определенной (обозначение $B > 0$) называется такая матрица, что $(Bx, x) > 0$ $\forall x \neq \theta$, где θ - нулевой вектор соответствующей размерности. Матрица определяется вариантом задания.
Требуется решить данную систему прямым методом, реализованным в задании 1, и итерационным методом Чебышёва. Так же предлагается оценить среднеквадратическую норму ($\|\cdot\|_2$) ошибки и оценить спектр матрицы системы. Реализация алгоритмов осуществлена на языке $C++$. Вектор решений x генерируется случайным образом с компонентами $x_i \in [-1, 1]$ с равномерным распределением. ($i = \overline{1, n}$). Правая часть строится как $F = x + Ax$ - при уже построенном решении x.

1.2 Описание методов

1.2.1 Процесс ортогонализации Грама-Шмидта

Данный метод позволяет решать СЛАУ путём построения QR-разложения матрицы системы. Данный метод был подробно рассмотрен в задании 1 и реализация не претерпела никаких изменений.

1.2.2 Метод Чебышева

Как было заявлено выше, данный метод является итерационным. Отличие этих методов от прямых заключается в том, что в процессе решения СЛАУ строится итерационная последовательность векторов, сходящаяся к точному решению. Итерационные методы удобны тем, что можно повысить точность, увеличив число итераций алгоритма. В случае использования как прямых, так и итерационных методов точное решение никогда не будет найдено из-за ошибок округления и неточного представления вещественных чисел в виде чисел с плавающей точкой. Теперь подробней остановимся на одном из таких методов - методе Чебышева, рассмотренном и обоснованном на лекциях.

Пусть $A^T = A > 0$, λ_{max}, λ_{min} - соответственно максимальное и минимальное собственное значение матрицы A. В силу положительной определенности и симметричности они будут вещественны и положительны. Обозначим:

$$\tau_0 = \frac{2}{\lambda_{max} + \lambda_{min}}, \quad \kappa = cond(A) = \frac{\lambda_{max}}{\lambda_{min}}, \quad \rho_0 = \frac{1 - \kappa^{-1}}{1 + \kappa^{-1}}$$

Рассмотрим вектора y^n такие, что:

$$\begin{cases} \frac{1}{\tau_n}(y^n - y^{n-1}) + Ay^{n-1} = f, \quad n = \overline{1, m} \\ \\ y^0 \text{ - заданное начальное приближение} \end{cases}$$

где

$$\tau_n = \frac{\tau_0}{1 + \rho_0 cos\frac{\pi}{2m}(2n-1)}, \quad m - \text{число итераций метода Чебышева.}$$

Пусть также имеется точное решение u^*: $Au^* = f$. Введём вектор ошибки на каждой итерации: $z^n = y^n - u^*$. На лекции была доказана следующая теорема:

Теорема (о сходимости метода Чебышева). При всех указанных выше условиях и обозначениях:

$$\|z^m\|_2 \leq q_m \|z^0\|_2, \text{ где}$$

$$q_m = \frac{2\rho_1^m}{1 + \rho_1^{2m}}, \quad \rho_1 = \frac{1 - \kappa^{-1/2}}{1 + \kappa^{-1/2}} < 1$$

То есть гарантируется сходимость метода к точному решению, т.к. $\|z^n\|_2 \to 0$. Метод Чебышева хоть и имеет намного более высокую скорость сходимости, чем, например, метод простых итераций, где имело место оценка $\|z^m\|_2 \leq \rho_0^m \|z^0\|_2$, но в процессе реализации накладываются дополнительные трудности.

Замечание. Для вычислений по указанным выше формулам необходимо знать λ_{max} и λ_{min}. На практике явное вычисление собственных значений у матриц большого размера не производится. Вместо этого достаточно найти a, b :

$$a \leq \lambda_{min} \leq \lambda_{max} \leq b$$

и использовать эти значения вместо λ_{min}, λ_{max}. Точность от этого станет хуже (т.к. увеличится величина κ), но зато этот приём позволяет избежать явного вычисления спектра, ограничившись лишь некоторыми оценками на собственные значения матрицы.

Замечание. Метод Чебышева "заточен" под заранее выбранное число итераций m, т.к. оно напрямую используется в вычислениях параметров τ_n. Это приводит к тому, что при появлении необходимости увеличить число итераций, придется заново пересчитывать все τ_n.

Замечание. На лекции было показано, что на практике имеет место следующее явление: норма ошибки $\|z^n\|_2$ на некоторых итерациях увеличивается. При прямом порядке итераций ($n = 1, 2, \ldots m$) такие "плохие"итерации будут следовать подряд, из-за чего ошибки округления будут стремительно расти, что может привести к переполнению до завершения финальной итерации. Таким образом, хоть сходимость и гарантированна теоретически, следует модифицировать алгоритм во избежании проблем с точностью на практике. Заданием предложено выбирать число итераций, равное степени двойки, то есть $n = 2^k,\ k \in \mathbb{N}$. Для такого случая на лекции был представлен алгоритм упорядочивания итераций, который и был использован в написанной программе. Пусть:

$$J^{(1)} = \{1\}, \quad J^{(2)} = \{1, 3\}$$

$$J^{(m)} = \{j_1^{(m)}, j_2^{(m)}, \dots j_m^{(m)}\} \Rightarrow$$
$$J^{(m)} = \{j_1^{(m)}, j_2^{(2m)}, j_2^{(m)}, j_4^{(2m)}, \dots j_m^{(m)}, j_{2m}^{(2m)}\}, \text{ где}$$
$$j_{2k}^{(2m)} = 4m - j_k^{(m)}$$

То есть данное реккурентное соотношение строит следующее множество индексов по предыдущему, расставляя все индексы из предыдущего множества на нечетные места и заполняя четные в соответствии с указанным соотношением. Такой выбор индексов можно интерпретировать следующим образом: чередуются "хорошие"итерации (уменьшающие норму ошибки) и "плохие"(увеличивающие эту норму). Сходимость от этого не перестанет иметь место, однако ошибка не будет накапливаться достаточно долго, чтобы вызвать переполнение.

1.3 Оценка спектра матрицы

Из курса линейной алгебры известна следующая теорема:

Теорема (Гершгорина). Пусть $A \in \mathbb{C}^{n \times n}$, введём следующие множества (круги Гершгорина):

$$\Gamma_i = \{z : |a_{ii} - z| \le R_i\}, \quad R_i = \sum_{j=1, j\neq i}^{n} |a_{ij}|$$

Тогда для любого собственного значения матрицы λ верно:

$$\lambda \in \bigcup_{i=1}^{n} \Gamma_i$$

То есть все собственные значения лежат в объединении кругов Гершгорина. Стоит отметить, что можно считать радиусы, суммируя модули по столбцам. В данной задаче эта возможность не предоставит дополнительных оценок, т.к. матрица является симметричной. Пользуясь этой теоремой и учитывая, что элементы матрицы и её собственные значения вещественны, можно свести круги к отрезкам на вещественной прямой. В программе были найдены все круги, объединение которых даёт следующую оценку на собственные значения:

$$\lambda \in [1, 158.6] \Rightarrow 1 \le \lambda_{min} \le \lambda_{max} \le 158.6$$

То есть искомые параметры $a = 1$ и $b = 158.6$. Положив начальное приближение $y^0 = \theta$ (так предложено заданием), можно перейти к программной реализации метода Чебышева, в которой так же будет поиск наименьшего числа итераций, при которых погрешность решения на последней итерации в среднеквадратической норме не превосходит погрешность прямого метода.

1.4 Программная реализация

1.4.1 Реализация метода Чебышева

Нахождение решения осуществляется данной функцией:

```
std::vector<double>
Chebyshev_solve(const Matrix& A, const std::vector<double>& f,
const std::vector<double>& y0, double a, double b, size_t m,
const std::vector<size_t> &mask) // Chebyshev method
{
    double t0 = 2 / (a + b); // approximation for optimal tau
    double cond = b / a; // approximation for cond(A)
    double r0 = (1 - 1 / cond) / (1 + 1 / cond); // approximation for rho
    double t_k;
    std::vector<double> y(y0);
    for (size_t k = 0; k < m; k++) { // iterations
        t_k = tau(t0, r0, m, mask[k]); // count parameter
        y = (f - A * y) * t_k + y; // update y
    }
    return y;
}
```

со вспомогательной функцией tau,

```
double
tau(const double &t0, const double &r0, const size_t m, const size_t n)
{
    const double pi = 2 * std::acos(0);
    return t0 / (1 + r0 * std::cos(pi * (2n - 1) / (2 * m)));
}
```

осуществляющей вычисление по формуле:

$$\tau_n = \frac{\tau_0}{1 + \rho_0 cos\frac{\pi}{2m}(2n-1)}$$

Опишем параметры подробней:

- A - матрица системы. Класс матриц был реализован в прошлом задании. С полной реализацией можно ознакомиться в исходном коде.
- f - правая часть уравнения.
- $y0$ - начальное приближение (в данном задании кладется равным нулю).
- a - левая граница оценки спектра.
- b - правая граница оценки спектра.
- m - число итераций алгоритма.
- $mask$ - оптимально упорядоченные индексы итераций ($J^{(m)}$).

1.4.2 Результат работы прямого метода

Метод	$\|x - \tilde{x}\|_2$	$\|f - A\tilde{x}\|_2$
Грам-Шмидт	5.2142e-15	3.42049e-13

Эти данные необходимы для нахождения числа итераций. Напомним, что заданием требуется найти минимальное число итераций, при котором норма ошибки в методе Чебышева будет не больше нормы ошибки прямого метода. Нахождение нужного числа итераций осуществлено перебором степеней 2, начиная с 1 степени.

1.4.3 Нахождения числа итераций

Данный цикл реализует вышеупомянутый перебор степеней двойки. Каждый раз проверяется точность и, если она хуже точности прямого метода, пересчитывается множество $J^{(m)}$, и вычисляется решение с $m = count$ итерациями. Вектор индексов $mask$ отвечает за оптимальный порядок итераций для количества итераций, равному степени двойки. Порядок итераций соответствует оптимальному, рассмотренному выше.

```
std::vector<double> y0(SIZE, 0), y(SIZE, 0);
size_t count = 1;
std::vector<size_t> mask = {1};
while (norm(x - y) >= x_error) {
    count *= 2;
    std::vector<size_t> new_mask(count);
    for (size_t k = 0; k < count; k++) {
        if (k % 2 == 0) {
            new_mask[k] = mask[k / 2];
        } else {
            new_mask[k] = 2 * count - mask[k / 2];
        }
    }
    mask = new_mask;
    y = Chebyshev_solve(A, f, y0, a, b, count, mask);
}
```

В результате выполнения программы получилось, что минимальное число итераций равняется 256.

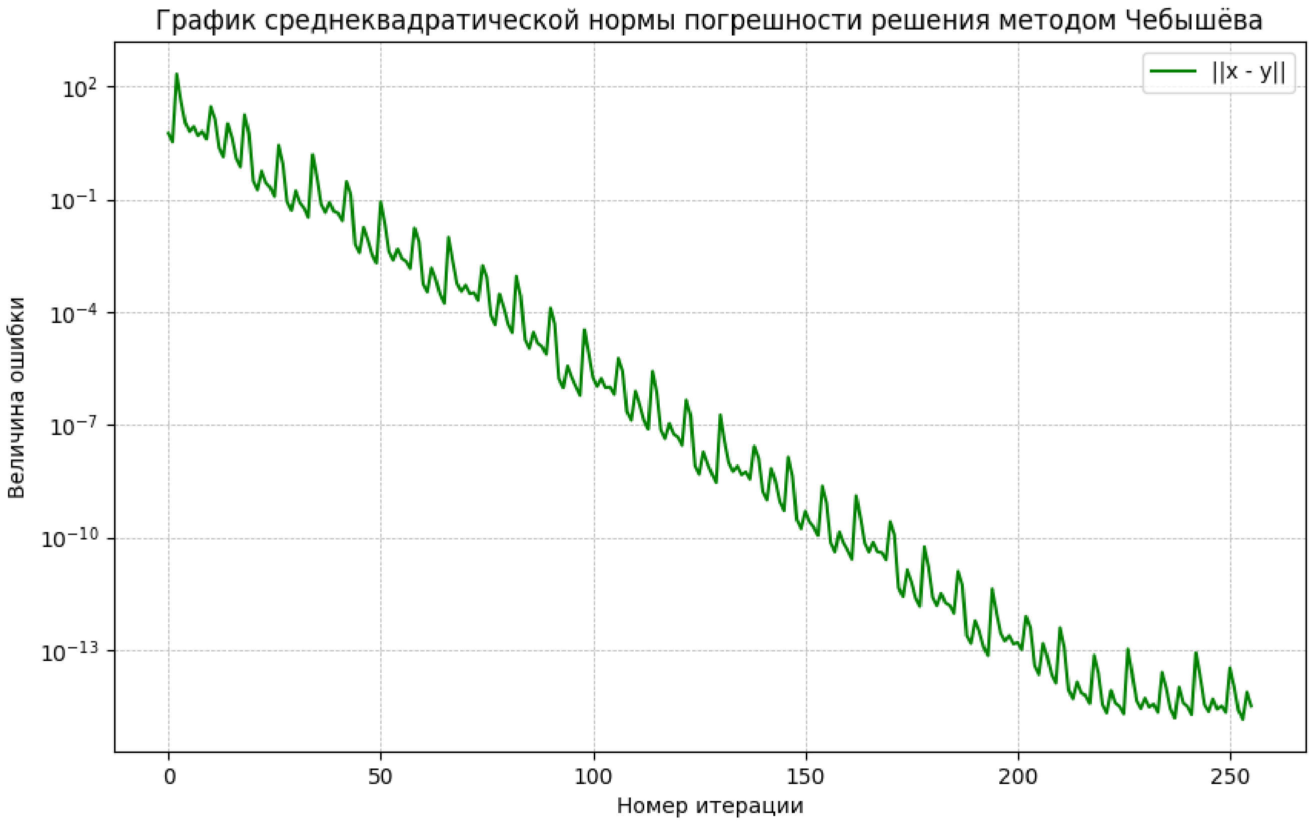
1.5 Норма ошибки метода Чебышева.

В данной таблице и далее: x - точное решение, $\tilde{x} = \tilde{x}^m$ - приближенное, $\tilde{x}^k$ - приближенное решение на k-ой итерации.

Метод	$\|x - \tilde{x}\|_2$	$\|f - A\tilde{x}\|_2$	$\frac{\|x-\tilde{x}\|_2}{\|x\|_2}$	Число итераций
Грам-Шмидт	5.2142e-15	3.42049e-13	8.3165e-16	-
Чебышев	4.11607e-15	3.16756e-13	6.56502e-16	256

Как и было заявлено, точность метода Чебышева выше, чем у прямого метода. Отметим, что точность ещё можно повысить, увеличив число итераций - в данном задании рассмотрено только наименьшее число итераций, для которого итерационный метод не хуже прямого. Так же важно уточнить, что от запуска к запуску минимальное число итераций может поменяться, т.к. вектор x каждый раз генерируется случайно. Однако в большинстве случаев значения 256 достаточно.

1.6 График $\|x - \tilde{x}^k\|_2$ на каждой итерации



Невооруженным взглядом видно, что практические результаты не противоречат теоретическим оценкам: погрешность действительно убывает немонотонно. Несмотря на периодические скачки, благодаря выбранному порядку норма не успевает вырасти достаточно, чтобы вызвать переполнение. Несмотря на немонотонность нормы ошибки, она всё равно сходится к нулю при увеличении числа итераций.

2 Выводы

Был реализован итерационный метод Чебышева решения СЛАУ. Для этого были оценены собственные значения матрицы с помощью теоремы о кругах Гершгорина. Количество итераций было подобрано минимальным таким, что итерационный метод имеет норму $\|z^m\|_2$ меньшую, чем прямой метод. Так же был построен график $\|z^k\|_2 = \|x - \tilde{x}^k\|_2$ в зависимости от номера итерации k. Задание выполнено в соответствии с поставленными требованиями. В следующем разделе указаны инструкции для компиляции и запуска написанной программы.

3 Инструкции по запуску программы.

- Запустить скрипт run.sh командой

  ```
  $ ./run.sh
  ```

  Данный скрипт создаст директорию build и внутри неё скомпилирует проект с помощью cmake и Makefile.

- находясь в директории build команда

  ```
  $ ./main 100 < ../tests/SLAU_var_6.txt
  ```

  запустит программу на матрице размера 100×100, заданной вариантом. Кроме того, предусмотрена возможность тестирования на матрицах размера 1024 и 2048:

  ```
  $ ./main 1024 < ../tests/in1024.txt
  $ ./main 2048 < ../tests/in2048.txt
  ```

  Во всех случаях вывод будет примерно такой:

  ```
  Gram-Shmidt:
  ||x - x_f|| = 5.2142e-15
  ||f - Ax_f|| = 3.42049e-13
  ||x - x_f|| / ||x|| = 8.3165e-16

  Spectrum range:
  Range: [1, 158.6]

  Chebyshev:
  ||x - y|| = 4.11607e-15
  ||f - Ay|| = 3.16756e-13
  ||x - y|| / ||x|| = 6.56502e-16
  Iterations count = 256
  ```

  Программа выводит соответственно погрешность решения, невязку и относительную погрешность решения для метода ортогонализации Грама-Шмидта, отрезок, содержащий все собственные значения матрицы системы (посчитанный с помощью теоремы о кругах Гершгорина), погрешность решения, невязку и относительную погрешность решения для итерационного метода Чебышева, а так же минимальное количество итераций, при которых погрешность решения этого метода не превосходит выше указанную погрешность прямого метода.