Санкт-Петербургский политехнический университет
Высшая школа прикладной математики
и вычислительной физики, ФизМех

Направление подготовки
«Прикладная математика и информатика»

Отчет по лабораторной работе №4
«Решение алгебраической проблемы собственных значений итерационными методами»

Выполнил студент гр. 5030102/00003 Петрошенко А.В.

Преподаватель: Курц В.В.

Санкт-Петербург
2021

Формулировка задачи и ее формализация

Собственные числа и собственные вектора(далее СЧ и СВ) - основные характеристики матрицы, поэтому их нахождение является важной задачей вычислительной математики. Методы нахождения можно разделить на те, которые решают полную задачу, то есть находят все СЧ и СВ, и на те, коорые решают эту задачу частично, то есть находят, например, только максимальное СЧ.
В данной работе будет реализован метод, который ищет максимальное СЧ и соответствующий ему СВ.

Постановка задачи:
Дана матрица $A \in \mathbb{R}^{n \times n}$, требуется найти такой λ_{max} и такой x, что $A\lambda_{max} = \lambda_{max}x$, где λ_{max} - не равен 0 и x - не нулевой вектор.
Матрица размерности $n \times n$ имеет n СЧ и СВ.
В данной работе будет реализован степенной метод с нормировкой и со сдвигами для улучшения его сходимости.

Алгоритм метода и условия его применимости

Величина $|\frac{\lambda_2}{\lambda_1}|$ определяет скорость сходимости: чем она меньше, тем быстрее сходится. Поэтому для уменьшения данной величины сдвинем все собственные числа на $\mu = \frac{\lambda_2+\lambda_n}{2}$ - это оптимальный сдвиг
Далее реализуем сам степенной метод с нормировкой для данного сдвига, то есть для матрицы $B = A - \mu E$
Алгоритм метода:

1. Берем произвольный вектор $y_0 \in \mathbb{R}^n$ и нормируем его:
   $\overline{y}_0 = \frac{y_0}{\mu_0}$, $\mu_0 = y_s^0$, $y_s^0 = ||y_0||_\infty$
2. Реализуем итерационную последовательность:
   $y_k = A\overline{y}_{k-1}$, $\overline{y}_k = \frac{y_k}{\mu_k}$, $\mu_k = y_s^k$, $y_s^k = ||y_k||_\infty$

Для окончания итерационного процесса будем использовать апостериорную оценку:

$$\frac{||A\overline{y}_k - \mu_k\overline{y}_k||_2}{||\overline{y}_k||_2} \geq \epsilon$$

Получаем, что $\mu_k \to \lambda_1$, $\overline{y}_k \to \alpha w_1$
Мы получили СЧ и СВ матрицы B, СЧ матрицы A будет равен найденному СЧ плюс выбранный нами сдвиг, а собственные вектора матриц A и B равны. Условия применимости метода:
A - вещественная положительно определенная матрица, у которой 2 максимальных по модулю собственных числа различны.

Предварительный анализ задачи

АПСЗ наиболее устойчива в случае, когда $A = A^T$. Таким образом, нам нужно задать положительно определенную симметричную матрицу: $A = QDQ^T$, где Q - ортогональная матрица, а D - диагональная. Теперь также удобно задавать отделимость собсвенных чисел.

Тестовый пример для задач малой размерности

Возьмем ортогональную матрицу: $Q = \begin{pmatrix} -0.9812 & 0.1729 & 0.0854 \\ -0.1762 & -0.6237 & -0.7615 \\ -0.0784 & -0.7623 & 0.6425 \end{pmatrix}$, $D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix}$
Тогда $A = \begin{pmatrix} 1.0444 & -0.2379 & -0.0221 \\ -0.2379 & 2.5487 & -0.5031 \\ -0.0221 & -0.5031 & 2.4068 \end{pmatrix}$

Матрица $B = A - \mu E = \begin{pmatrix} -0.4556 & -0.2379 & -0.0221 \\ -0.2379 & 1.0487 & -0.5031 \\ -0.0221 & -0.5031 & 0.9068 \end{pmatrix}$ Возьмем $y_0 0 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ и найдем решение с точностью $\epsilon = 0.1$:

В таком случае $\overline{y}_0 = \frac{y_0}{\mu_0} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

1. $y_1 = \begin{pmatrix} -0.7156 \\ 0.3077 \\ 0.3816 \end{pmatrix}$, $\mu_1 = -0.7156$, $\overline{y}_1 = \begin{pmatrix} 1 \\ -0.4230 \\ -0.5333 \end{pmatrix}$
   Апостериорная оценка: $0.8733 > \epsilon$
2. $y_2 = \begin{pmatrix} -0.3415 \\ -0.4205 \\ -0.2893 \end{pmatrix}$, $\mu_2 = -0.4205$, $\overline{y}_2 = \begin{pmatrix} 0.8121 \\ 1 \\ 0.6880 \end{pmatrix}$
   Апостериорная оценка: $0.7175 > \epsilon$
3. $y_3 = \begin{pmatrix} -0.6231 \\ 0.5094 \\ 0.1028 \end{pmatrix}$, $\mu_3 = -0.6231$, $\overline{y}_3 = \begin{pmatrix} 1 \\ -0.8175 \\ -0.1650 \end{pmatrix}$
   Апостериорная оценка: $1.2064 > \epsilon$
4. $y_4 = \begin{pmatrix} -0.2575 \\ -1.0122 \\ 0.2395 \end{pmatrix}$, $\mu_4 = -1.0122$, $\overline{y}_4 = \begin{pmatrix} 0.2544 \\ 1 \\ -0.2367 \end{pmatrix}$
   Апостериорная оценка: $2.2007 > \epsilon$
5. $y_5 = \begin{pmatrix} -0.3486 \\ 1.1072 \\ -0.7233 \end{pmatrix}$, $\mu_5 = 1.1072$, $\overline{y}_5 = \begin{pmatrix} -0.3148 \\ 1 \\ -0.6533 \end{pmatrix}$
   Апостериорная оценка: $0.4611 > \epsilon$
6. $y_6 = \begin{pmatrix} -0.0800 \\ 1.4522 \\ -1.0885 \end{pmatrix}$, $\mu_6 = 1.4522$, $\overline{y}_6 = \begin{pmatrix} -0.0551 \\ 1 \\ -0.7495 \end{pmatrix}$
   Апостериорная оценка: $0.1194 > \epsilon$
7. $y_7 = \begin{pmatrix} -0.1962 \\ 1.4389 \\ -1.1816 \end{pmatrix}$, $\mu_7 = 1.4389$, $\overline{y}_7 = \begin{pmatrix} -0.1364 \\ 1 \\ -0.8212 \end{pmatrix}$
   Апостериорная оценка: $0.0710 < \epsilon$

Таким образом мы получили СЧ равное 1.4389 и СВ равный $\begin{pmatrix} -0.1364 \\ 1 \\ -0.8212 \end{pmatrix}$, теперь нам нужно прибавить сдвиг к СЧ и мы получим окончательные собсвтенные значения для матрицы A: $\lambda_A = 1.4389 + 1.5 = 2.9389 \approx 3$

Контрольные тесты

1. Создадим матрицу 10×10 и будем менять сдвиг от λ_n до λ_1 с шагом в 0.001 от $\lambda_1 - \lambda_n$.
2. Создадим 2 матрицы 10×10 с хорошей отделимостью и с плохой, и будем менять точность поиска собственных значений от 10^{-5} до 10^{-14}.
3. Создадим 2 матрицы 10×10 с хорошей отделимостью и с плохой, и будем вносить различные возмущения меняя их порядок от 10 до 10^{-8}.

Модульная структура программы

```
typedef struct{
     vector<vector<double>> A;
     int rang;
}matrix_t;
```
- структура данных, имеющая 2 поля: двумерный массив для значений матрицы и целое число для хранения ранга матрицы.

`int GetNum(ifstream *F) int fGetNum(ifstream *F)`
- функции для получения одного числа из файла. Использовалась для получения из файла ранга матриц, их количества и СЧ.

`matrix_t ImportMatrix(ifstream *F, int rang)`
- функция преобразующая матрицу, полученные из файла в удобный вид.

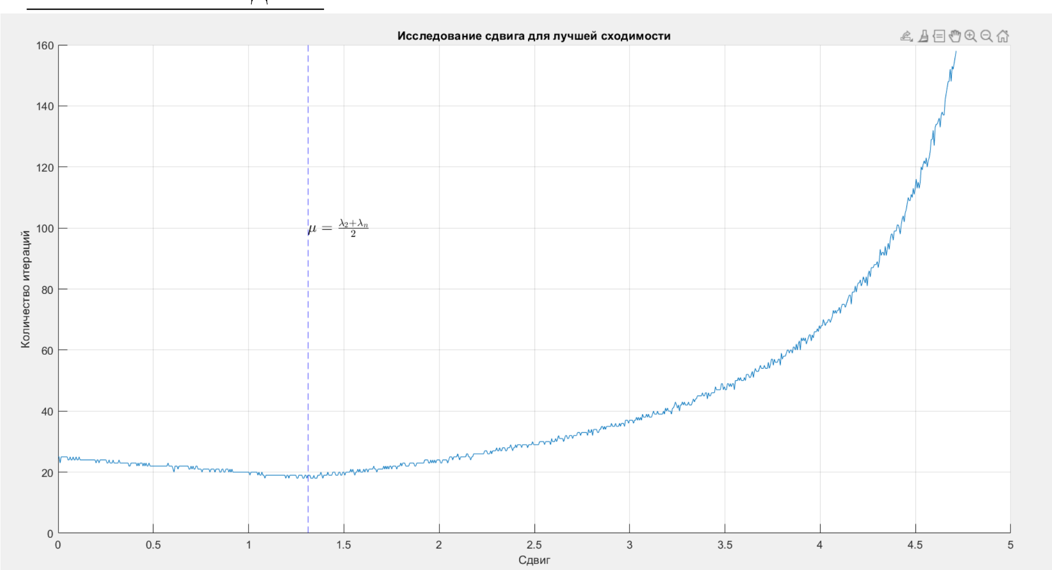
```
vector<double> RandomVector(int size)
vector<double> VectorSubstract(vector<double> vec1, vector<double> vec2)
vector<double> MatrixMulVector(matrix_t matrix, vector<double> vec)
vector<double> VectorMulNumber(vector<double> vec, double num)
double Norm(vector<double> vec)
double MaxComponent(vector<double> vec)
```
- вспомогательные функции для степенного метода(названия функций передают их смысл).

```
vector<double> PowerMethod(matrix_t A, double eps)
matrix_t Shift(matrix_t A, double mu)
vector<double> ShiftPowerMethod(matrix_t A, double mu, double eps)
```
- реализация степенного метода со сдвигом для улучшения сходимости.

`void OutputVector(vector<double> vec, ofstream *F)`
- функция записи нужных для дальнейшего анализа данных в файл.

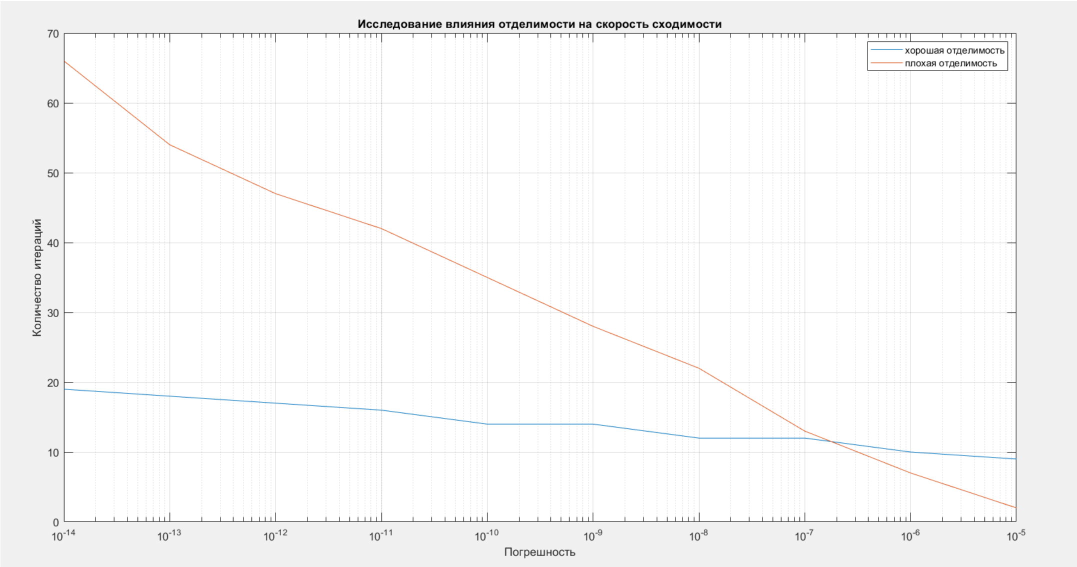
Численный анализ

▷ Оптимальный сдвиг:



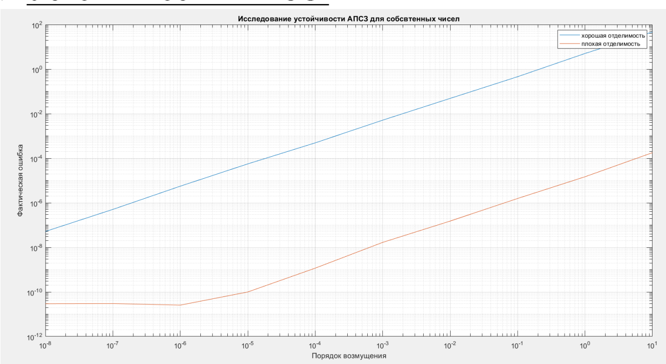
Из графика видно, что при оптимальном сдвиге, наблюдается минимум количества итераций, что вполне логично и ожидаемо, ведь сдвиг оптимальный.

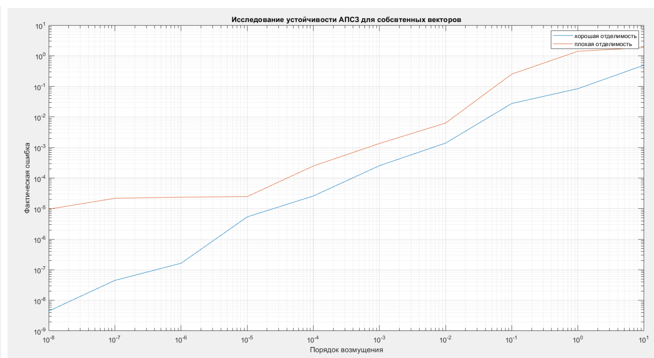
▷ <u>Отделимость:</u>



На графике мы видим, что при увеличении точности нужно совершить больше итераций для плохо отделимой матрицы, чтобы найти ее СЗ, для маленькой точности плохо отделимой матрице нужно всего 2 итерации, что объясняется тем, что беря произвольный исходный вектор, мы можем случайно близко попасть к искомому значению.

▷ <u>Устойчивость АПСЗ:</u>





Из графиков становится очевидно, что при внесении достаточно большого возмущения, погрешность при нахождении СЗ так же возрастает, а промежуток между графиками возникает из-за сильного различия заданной отделимости.

Общие выводы:

В данной лабораторной работе мы научились находить собственные значения матриц с помощью степенного метода. Мы увидели, что оптимальный сдвиг действительно оптимальный и что при преобладании максимального СЧ над другими, в частности, особенно над вторым максимальным СЧ, происходит ускорение сходимости метода.