Cанкт – Петербургский политехнический университет Петра Великого

Физико-механический институт

Отчет по лабораторной работе №2 по Численным методам

«Алгебраическая проблема собственных значений»

Метод обратных итераций (+ сдвиг), LR-алгоритм

Выполнил студент

Группы 5030301/10002 Тугай В.В.

Преподаватель: К.Н. Козлов

Санкт-Петербург 2023

# Содержание отчета:

1. Ручной расчет
2. Метод обратных итераций (+ сдвиг)
3. LR-алгоритм
4. Структура программы
5. Зависимость погрешности от числа обусловленности
6. Зависимость погрешности от возмущения
7. Проверка работы метода для симметричных и несимметричных матриц
8. Зависимость погрешности от точности
9. Зависимость числа итераций от точности
10. Создание матриц с различной отделимостью собственных чисел
11. Проверка «на ноль»
12. Работоспособность, время выполнения и погрешность для функции eig
13. Работоспособность и время выполнения для функции eigs
14. Реализация QR-алгоритма с приведением матрицы к форме Хессенберга
15. Выводы

**Вариант 21**

Метод обратных итераций (+ сдвиг), LR-алгоритм

**Ручной расчет**

**Text, letter

Description automatically generated**

**A picture containing text, building, people, receipt

Description automatically generated**

**Метод обратных итераций (+ сдвиг)**

Алгоритм метода обратных итераций со сдвигом:

Пусть имеем собственное значение λ матрицы А. Запишем уравнение и построим по нему итерационный процесс: . На каждой итерации будем вычислять отношение скалярных произведений , и это отношение будет стремиться к собственному числу матрицы , и если у матрицы А собственное значение , то собственные значения матрицы будут Тогда у матрицы собственные значения будут Отсюда можем записать , и если мы зададим начальное приближение , близкое к λ, то сможем записать итерационный процесс Тогда получим , т. к. вместо в итерационном процессе на 1 итерации будет использоваться , а на последующих итерациях вместо будут использоваться посчитанные на предыдущих шагах . Также для 1 итерации положим , а для упрощения решения системы будет использоваться LU-разложение матрицы , и на каждой итерации будем находить вектор , который является собственным вектором матрицы А.

Алгоритм метода обратных итераций без сдвига отличается тем, что выражение и LU-разложение этой матрицы будет происходить перед итерационным процессом, и вместо там будет , будет находиться как , и по завершению итерационного процесса собственное найдем как

**LR-алгоритм**

LR-алгоритм основан на разложении матрицы в произведение треугольных A=LR, где L – единичная нижняя треугольная, а R – верхняя треугольная. Предположим, что мы составим преобразование подобия L-1AL матрицы А, тогда L-1AL= L-1(LR)L=RL. Следовательно, если разложить А в произведение треугольных, а затем перемножить сомножители в обратном порядке, то получится матрица, подобная исходной. Алгоритм же определяется уравнениями Здесь подобна , а значит подобна . При больших s матрица cтремится к виду верхней треугольной матрицы, на диагонали которой лежат собственные числа исходной матрицы А.

**Структура программы**

Программа matrix.m нужна для создания файлов, в которых будут записаны матрицы A 10x10 с отделимостями от 0.01 до 10, и в дальнейшем именно эти файлы будет считывать программа lab2.f95. Программа lab2.f95 считывает эти файлы и задает переменную и массивы: переменная содержит значения отделимости, а массивы – и есть матрицы A.

Программа lab2.f95 состоит из основного тела и двух модулей:

matrix\_fluct.mod - модуль, содержащий функции для внесения случайных возмущений в строку матрицы в пределах процента, заданного пользователем.

ort\_givens\_\_reverse\_iter\_\_lr.mod - модуль, содержащий подпрограмму с реализацией методов, подпрограмму с LU-факторизацией, а также функцию для перевода натуральных чисел в строку для более удобной записи данных в файлы, которые будут созданы по окончанию программы.

Программа lab2.m реализует решение алгебраической проблемы собственных значений с помощью средств Matlab. После выполнения в папке будут созданы рисунки, все они представлены далее в отчете.

**Зависимость погрешности от отделимости**

На рисунках 1 и 2 изображены графики зависимости погрешности от отделимости в логарифмических осях. Как можно увидеть по рисунку, с увеличением числа обусловленности растет и ошибка. Наименьшая ошибка у метода обратных итераций без сдвига, наибольшая же у LR-алгоритма

**Chart, line chart

Description automatically generated**

Рисунок 1

Зависимость погрешности от отделимости

для собственных значений

**Chart, line chart

Description automatically generated**

Рисунок 2

Зависимость погрешности от отделимости

для собственных векторов

**Зависимость относительной погрешности от возмущения**

На рисунках 3 и 4 изображены графики зависимости относительной погрешности от возмущения элементов строки матрицы А. Как видно из рисунков, ошибка растет вместе с увеличением возмущения, вносимого пользователем, для всех трех методов, самых чувствительных к возмущениям оказался метод обратных итераций со сдвигом.

**Chart, line chart

Description automatically generated**

Рисунок 3

Зависимость погрешности от отделимости

для собственных значений

**Chart, line chart

Description automatically generated**

Рисунок 4

Зависимость погрешности от отделимости

для собственных векторов

**Проверка работы метода для симметричных и несимметричных матриц**

На рисунках 5 и 6 изображены графики зависимости погрешности от отделимости в логарифмических осях. Как можно увидеть по рисунку, все три метода исправно работают как с симметричными, так и несимметричными матрицами, и при отделимости 10 ошибка меньше, чем при отделимости 0.1.

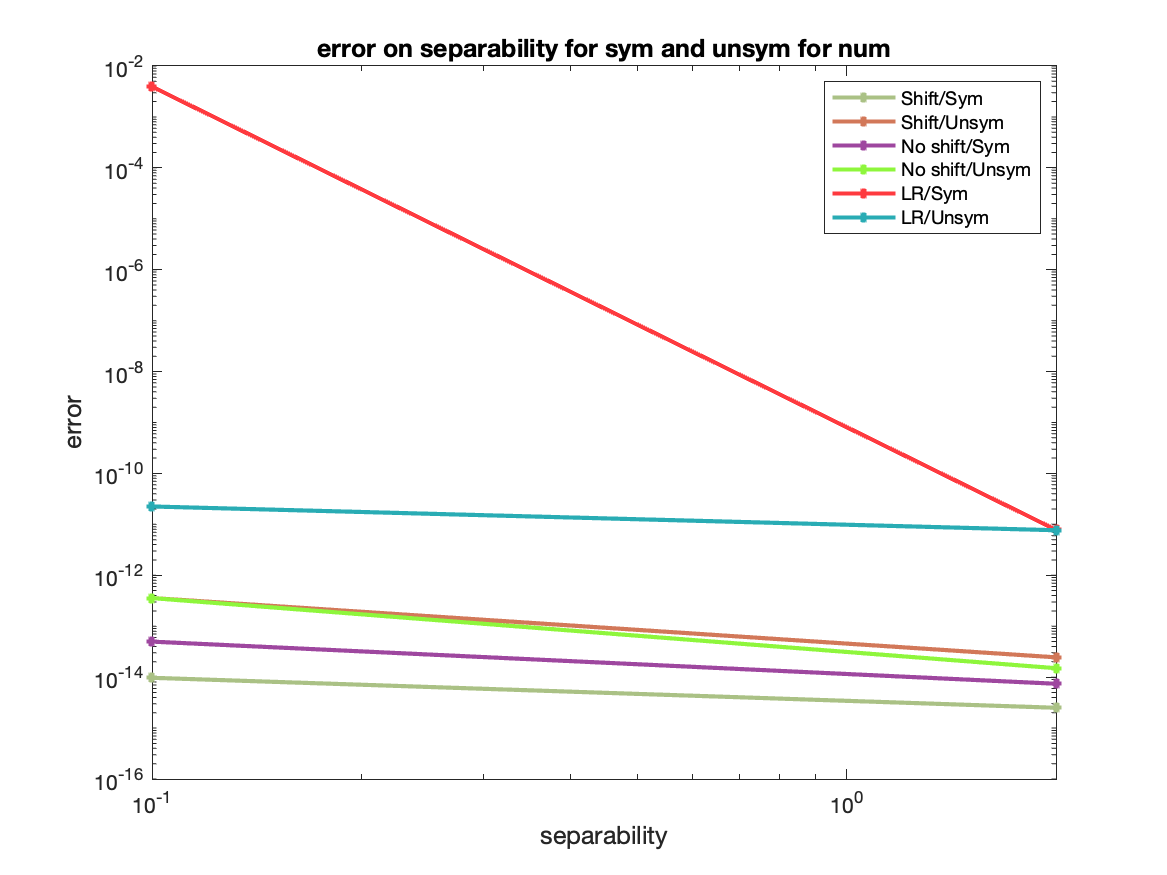


Рисунок 5

Зависимость погрешности от отделимости

для собственных значений

**Chart, line chart

Description automatically generated**

Рисунок 6

Зависимость погрешности от отделимости

для собственных векторов

**Зависимость погрешности от точности**

На рисунках 7 и 8 изображены графики зависимости погрешности от точности в логарифмических осях. Как можно увидеть по рисункам, с увеличением точности ошибка уменьшается, что соответствует смыслу данного параметра.

Chart, line chart

Description automatically generated

Рисунок 7

Зависимость погрешности от точности

для собственных значений

**Chart, line chart

Description automatically generated**

Рисунок 8

Зависимость погрешности от точности

для собственных векторов

**Зависимость числа итераций от точности**

На рисунках 9 и 10 изображены графики зависимости числа итераций от точности в логарифмических осях. Как можно увидеть по рисункам, с увеличением точности увеличивается и число итераций, необходимое для того, чтобы метод сошелся, что является логичным для итерационного процесса.

**Chart, line chart

Description automatically generated**

Рисунок 9

Зависимость погрешности от точности

для обратных итераций без сдвига

**Chart, line chart

Description automatically generated**

Рисунок 10

Зависимость погрешности от точности

для обратных итераций со сдвига

**Создание матриц с различной отделимостью СЧ**

На рисунках 11-24 приведены матрицы через функцию spy, исходя из рисунков можно сделать вывод, что у заданных матриц нет нулевых элементов.

Shape

Description automatically generatedShape

Description automatically generated

Рисунок 11 Рисунок 12

Все СЧ хорошо отделимы, вещественны, 100х100 Все СЧ хорошо отделимы, вещественны, 500х500

Shape

Description automatically generatedShape

Description automatically generated

Рисунок 13 Рисунок 14

Все СЧ хорошо отделимы, вещественны, 500х500, Все СЧ хорошо отделимы, вещественны, 500х500,

нулевые элементы заменены на EPS=1e-16 нулевые элементы заменены на EPS=1e-100

Shape

Description automatically generatedShape

Description automatically generated

Рисунок 15 Рисунок 16

Все СЧ плохо отделимы, вещественны, 100х100 Все СЧ плохо отделимы, вещественны, 200х200

Shape

Description automatically generatedShape

Description automatically generated

Рисунок 17 Рисунок 18

Макс. СЧ хорошо отделимы, остальные - нет, Макс. СЧ хорошо отделимы, остальные - нет,

вещественны, 100х100 вещественны, 500х500

Shape

Description automatically generatedShape

Description automatically generated

Рисунок 19 Рисунок 20

Макс. СЧ хорошо отделимы, остальные - нет, Макс. СЧ хорошо отделимы, остальные - нет,

вещественны, 100х100 вещественны, 500х500

нулевые элементы заменены на EPS=1e-16 нулевые элементы заменены на EPS=1e-100

Shape

Description automatically generatedShape

Description automatically generated

Рисунок 21 Рисунок 22

Мин. СЧ хорошо отделимы, остальные - нет, Мин. СЧ хорошо отделимы, остальные - нет,

вещественны, 100х100 вещественны, 500х500

Shape

Description automatically generatedShape

Description automatically generated

Рисунок 23 Рисунок 24

Все СЧ хорошо отделимы, несколько пар Все СЧ хорошо отделимы, несколько пар

комплексно-сопряженных, 100х100 комплексно-сопряженных, 500х500

**Проверка «на ноль»**

Проверка «на ноль» происходила следующим образом :

[V,D] = eig(B);

disp(i)

zero\_check=norm(B\*V-V\*D)

Вывод в консоль:

1

zero\_check =

1.9394e-12

2

zero\_check =

2.1045e-11

3

zero\_check =

2.8586e-11

4

zero\_check =

2.5905e-11

5

zero\_check =

1.8621e-13

6

zero\_check =

1.4219e-13

7

zero\_check =

8.1719e-14

8

zero\_check =

6.1313e-13

9

zero\_check =

3.9771e-13

10

zero\_check =

4.0156e-13

11

zero\_check =

1.2392e-13

12

zero\_check =

3.8751e-13

13

zero\_check =

1.8290e-12

14

zero\_check =

2.3372e-11

Значения не превышают 10-11, что говорит об успешной проверки «на ноль» для всех 14 матриц

**Работоспособность, время выполнения и погрешность для функции eig**

Для всех 14 матриц были проверены возможные вызовы функции eig, такие как:

3.1) d = eig(B);

3.2) d = eig(B,C);

3.3) [V,D] = eig(B);

3.4) [V,D] = eig(B, 'nobalance');

3.5) [V,D] = eig(B,C);

3.6) [V,D] = eig(B,C, 'chol');

Результаты предоставлены на рисунке 25

**Graphical user interface, text

Description automatically generated**

Рисунок 25

Время вычислений и погрешность функции eig

**Работоспособность и время выполнения для функции eigs**

Для всех 14 матриц были проверены возможные вызовы функции eigs в 5 вариациях: 'largestabs', 'smallestabs', 'largestreal', 'smallestreal', 'bothendsreal'. Время вычисления предоставлено на рисунке 26

**Text

Description automatically generated**

Рисунок 26

Время вычислений функции eigs

**Реализация QR-алгоритма с приведением матрицы к форме Хессенберга**

Реализация классического QR-алгоритм для нахождения всех собственных значений с предварительным приведением матрицы к форме Хессенберга в MATLAB была осуществлена следующим образом:

H = hess(B);

err = inf;

k = 0;

while err > 1e-2

k = k+1;

[Q1, R1] = qr(H);

H = R1 \* Q1;

err = norm(tril(H, -1), 'fro');

if (k>=5000)

break

end

end

Время вычисления и погрешность предоставлено на рисунке 27

**Text

Description automatically generated**

Рисунок 27

Время вычислений и погрешность QR-алгоритма

**Выводы**

В ходе проделанной лабораторной работы нами были исследованы метод обратных итераций без сдвига и со сдвигом и LR-алгоритм, а также проведено исследование решения алгебраической проблемы собственных значений средствами пакета Matlab. Метод и алгоритм были реализованы на языке Fortran, затем по полученным результатам с помощью пакета Matlab были построены графики зависимостей. Исследование же (вместе с построением графиков) было проведено с помощью пакета Matlab.