**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики и информатики**

**Отчет**

**По методам вычислений**

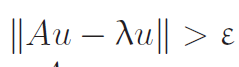
Выполнил студент группы № 13

*Веренич Владислав Николаевич*

Минск 2020

**№1 Степенной метод**

## Критерий остановки:



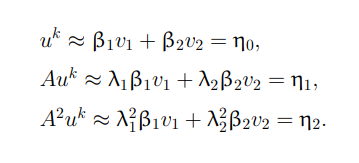
Данный критерий вытекает из основного свойства собственного значения и собственного вектора.



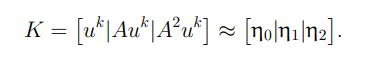
Вообще, я искал всё время два максимальных по модулю собственных значения. Однако критерий остановки проверялся только для одного из найденных собственных значений.

В качестве вектора x выступал вектор u. Я также использовал и хранил три значения вектора массиве xk, где xk[0] = u, xk[1] = A\*u, xk[2] = A\*A\*u.

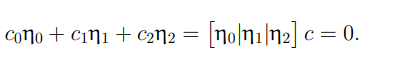
Дело в том, что при большом кол-ве итераций вектора в массиве будут равны соответственно:



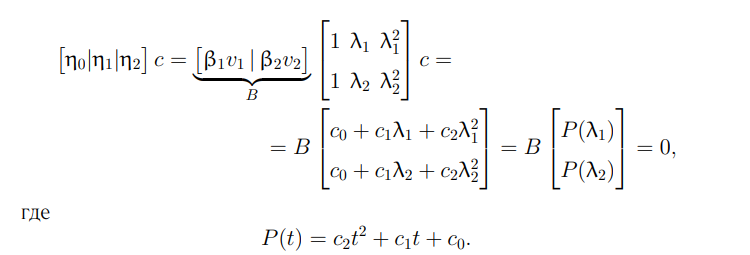
После чего я составлял матрицу K вида:

****

Векторы η𝑖 линейно зависимы. Это означает, что существует вектор 𝑐 = (𝑐0, 𝑐1, 𝑐2)^𝑇 != 0 такой, что

****

Учитывая определение η𝑖 , запишем это тождество в виде

****

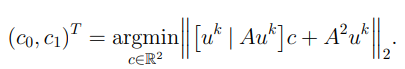
Так как столбцы матрицы 𝐵 линейно независимы, отсюда следует, что

****

то есть искомые собственные значения можно найти, решив квадратное уравнение 𝑃(λ) = 0. Осталось найти коэффициенты многочлена 𝑃. Они определены с точностью до постоянного множителя, поэтому можно положить 𝑐2 = 1. Остальные коэффициенты найдем, используя исходное тождество (3.6):

****

Отсюда можно приближенно найти коэффициенты многочлена 𝑃 путем решения задачи наименьших квадратов

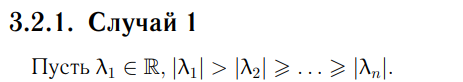
****

После этого, решая квадратное уравнение 𝑃(𝑡) = 0, находятся приближения к искомым собственным значениям λ1 и λ2.

Однако ещё раз подчеркну, что в условии остановки итераций использовалось лишь одно из собственных значений(первое) и именно для него пересчитывались значения векторов из массива xk.

Собственные вектора я находил следующим образом. Когда итерации прекращаются(то есть достигнута требуемая точность) я имею в своем распоряжении два собственных значения. Возможны следующие случаи:

1. Наибольшее по модулю собственное значение вещественное



Собственные вектор один и находится следующим образом:

v1 = Au

1. Два найденных собственных значения являются вещественными числами, равными и, соответственно, наибольшими по модулю



Собственные вектор один и находится следующим образом:

v1 = Au

1. Два найденных собственных значения являются вещественными числами, равными по модулю, но имеют разные знаки.



Собственные векторы находятся следующим образом:

v1 = Au + -(-lambda1\* u);

v1 = Au + -(lambda1\* u);

1. Два найденных собственных значения являются комплексными числами, равными по модулю.

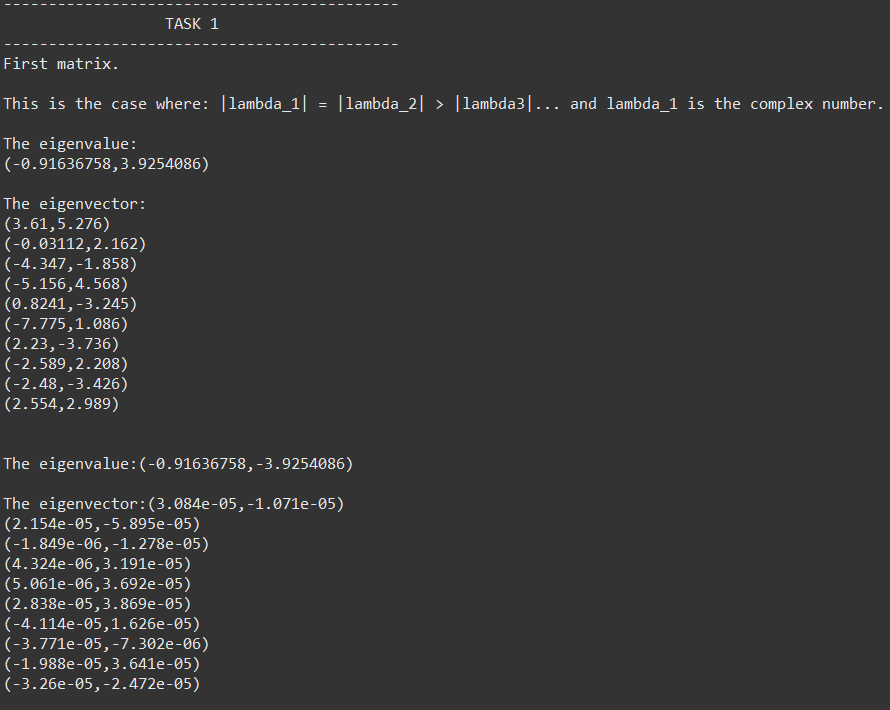
Собственные векторы находятся следующим образом:

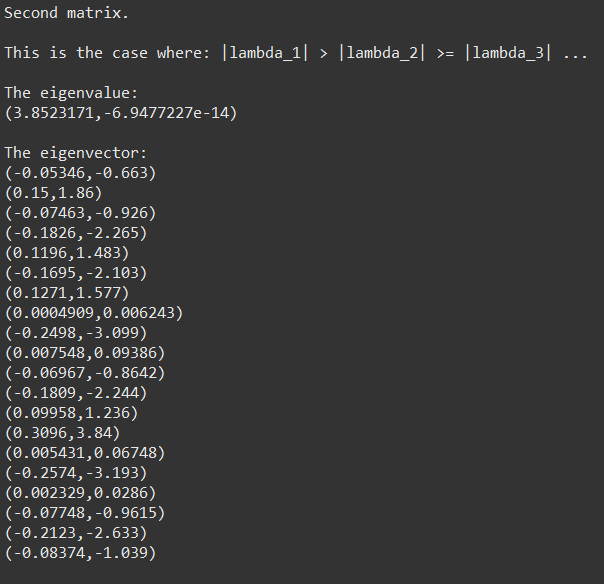
v1 = Au + -(lambda2 \* u);

v2 = Au + -(lambda1 \* u);

Результат работы программы:

(собственные значения вычислялись с точностью 1e-3)





Также я провел экспериментальное исследование скорости работы степенного алгоритма в зависимости от размерности матрицы. В качестве тестовых матриц брались матрицы от 2х2 до 30х30. График зависимости времени выполнения алгоритма от размерности матрицы можете видеть ниже:

****

Асимптотика реализованного мной алгоритма:

1. На каждой итерации я преобразую мой собственный вектор x следующим образом: xk+1 += A \* xk. Трудоемкость умножения матрицы на вектор O(n^2). После чего я получаю матрицу A1\* размерности n x 2, состоящую из найденных векторов. Я умножаю эту матрицу слева на её транспонированную и получаю матрицу 2x2. Трудоемкость O(n). Также получаю вектор b размерности 2, умножением транспонированной матрицы A\* на вектор xk. Трудоемкость O(n).

В итоге для одной итерации получаем тркдоемкость O(n^2).

Всего совершается i итераций. В итоге получаем, что трудоемкость алгоритма равную O(n^2 \* i)

**№2 QR-алгоритм**

Для нахождения всех собственных значений одновременно используется QR алгоритм.

Алгоритм состоит в следующем:

1) строится 𝑄𝑅-разложение для матрицы 𝐴𝑘: 𝐴𝑘 = 𝑄𝑘 \* 𝑅𝑘;

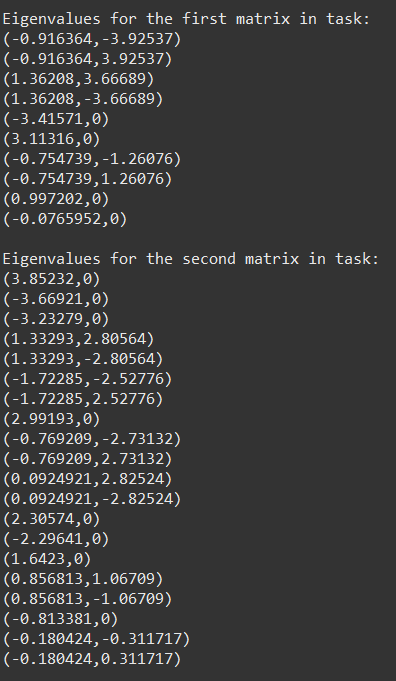
2) вычисляется 𝐴𝑘+1 = 𝑅𝑘 \* 𝑄𝑘.

При этом QR-разложение я получаю методом отражений.

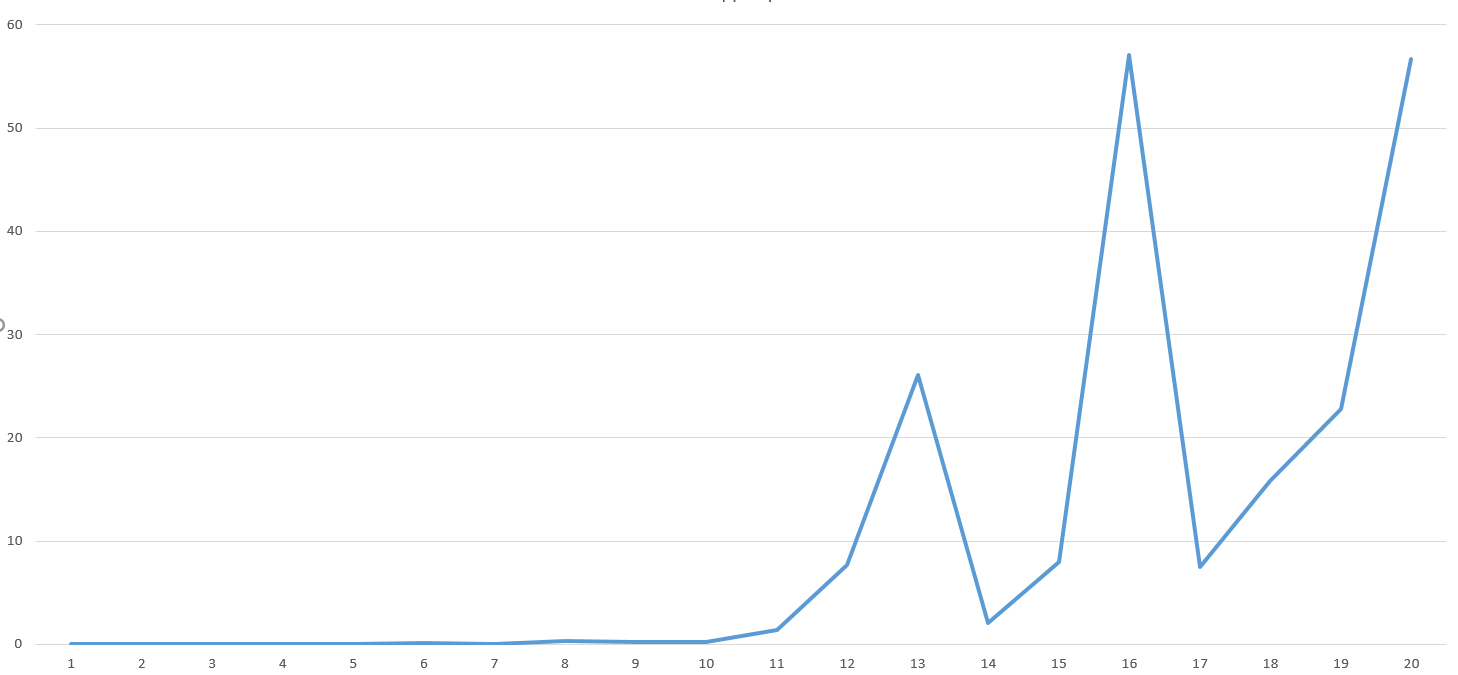
Так как QR-алгоритм никогда не применяется к неподготовленной матрице(так как это сл ишком трудоемко), я привожу исходную матрицу преобразованиями подобия к форме Хессенберга 𝐻 = 𝑄^𝑇 \* 𝐴 \* 𝑄 и лишь потом начинаю итерации алгоритма. Таким образом я снижаю трудоемкость одной итерации с O(n^3) до O(n^2).

Результат применения QR-алгоритма к матрицам в задании:

(собственные значения вычислялись с точностью 1e-3)

****

Также я провел экспериментальное исследование скорости работы QR-алгоритма в зависимости от размерности матрицы. В качестве тестовых матриц брались матрицы от 2х2 до 20х20. График зависимости времени выполнения QR-алгоритма от размерности матрицы можете видеть ниже:

****

Асимптотика реализованного мной алгоритма:

1. Приведение к форме Хессенберга методом отражений

На каждой итерации я получаю матрицу H и умножаю дважды на матрицу A(исходную). Это O(n^3) на каждой итерации. На каждой итерации таким образом я добиваюсь того, что соответствующий столбец ниже диагонали зануляется. То есть всего таких итераций я должен совершить n - 1. Значит, привожу к матрице Хессенберга я за O(n^4)

1. Как уже было сказано сам QR-алгоритм выполняется за O(n^2). Причем кол-во итераций неограниченно, пусть i – кол-во итераций, совершенных в QR-алгоритме. Тогда итоговая сложность O(n^2 \* i).

В итоге получаем O(n^3 + n^2 \* i).

# Выводы по полученным результатам

На основании результатов полученных в задании 1 и 2 делаю следующие выводы.

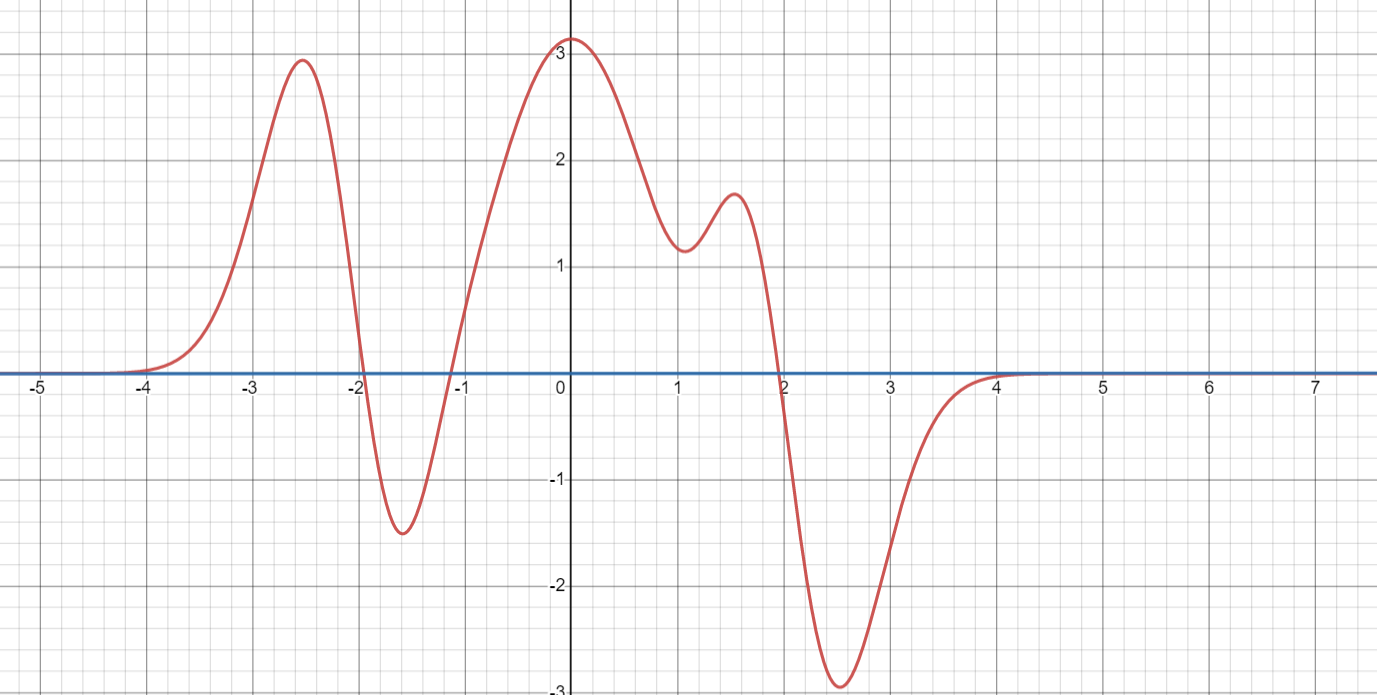
1. Среднее время нахождения двух максимальных по модулю корней степенным методом примерно в 50 раз меньше чем время нахождения всех корней QR алгоритмом.
2. Как видим из графиков для QR-алгоритма, может быть такая ситуация: для каких-то двух случайных соседних матриц алгоритм нахождения всех собственных значений для матрицы меньшей размерности может выполняться дольше, чем для матрицы большей размерности(это связано с тем, что при запуске алгоритма на диагонали будут присутствовать элементы очень близкие к действительным собственным значениям исходной матрицы)
3. Минимальное и максимальное время нахождения собственных значений степенным методом сильно разнится, хотя среднее время намного ближе к минимальному. Это значит, что крайне редко у сгенерированной матрицы собственные значения почти равны.

**№3 Решение нелинейного ур-ния**

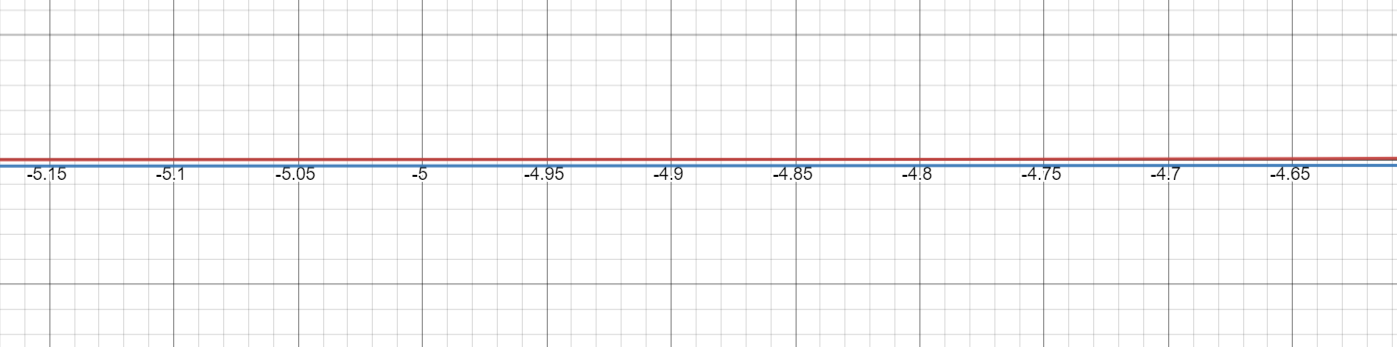
Для начала построим график(проект под названием Plot)

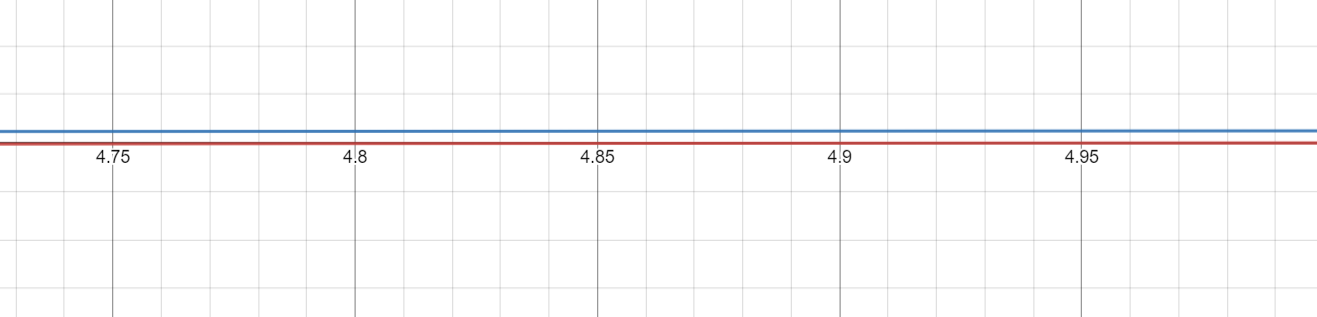
Построим отдельно график для функции f1 = (x^9 + pi) \* cos(ln(x^2 + 1)) / e^(x^2)

И для ф-ции f2 = x / 2022



Видим, что графики ф-ций персекаются точнг в 3 точках. Для того чтобы доказать, что больше точек пересечения нет, приблизим график у краев:

****

****

Отсюда видно, что на самом деле графики ф-ций не пересекаются вдоль осей, а расходятся.

Не трудно определить примерные границы корней ур-ния:

x1 – [-3.0; -1.5]

x2 – [-1.5; 0]

x3 – [1.5; 2.5]

Сужать данные промежутки будем методом бисекции до размера не более 10^(-4) . После вся работа будет вестись с приближениями выбранными с правых краев полученных отрезков. Для проверки на сходимость при определенном начальном приближении я проверял условие:

f(x0) \* f’’(x0) > 0

Все три приближения соответствуют условию выше, значит, метод Ньютона сходится при данных приближениях.

В качестве критерия остановки использовал следующее условие:

|xCurrent - xPrevious | < = EPSILON

Результат получаем с точностью 10 ^ (-8).

Полученные результаты:

