**МИНОБРНАУКИ РОССИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕНЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ**

**им. В.Г.Шухова»**

**(БГТУ им. В.Г.Шухова)**

Кафедра технической кибернетики

Дисциплина: Теория матриц

Практическая работа № 4

Тема: «Ортогональное преобразование матриц. Нахождение собственных чисел и собственных векторов матрицы»

Выполнил:

Студент группы МТК-233

Орлов-Куреши М. Н.

Проверил:

Кариков Е. Б.

Белгород 2023

**Цель работы:** изучить способы нахождения собственных чисел и собственных векторов. Реализовать одним из методов нахождение собственных чисел и собственных векторов на языке программирования Python.

**QR – алгоритм**

QR-алгоритм — это численный метод в линейной алгебре, предназначенный для решения полной проблемы собственных значений, то есть отыскания всех собственных чисел и собственных векторов матрицы.

Пусть A — вещественная матрица, для которой мы хотим найти собственные числа и векторы. Положим A0=A. На k-м шаге (начиная с k = 0) вычислим QR-разложение Ak=QkRk, где Qk — ортогональная матрица (то есть QkT = Qk−1), а Rk — верхняя треугольная матрица. Затем мы определяем Ak+1 = RkQk.

Пусть все диагональные миноры матрицы A не вырождены. Тогда последовательность матриц Ak при �→∞, сходится по форме к клеточному правому треугольному виду, соответствующему клеткам с одинаковыми по модулю собственными значениями.

Для того, чтобы получить собственные векторы матрицы, нужно перемножить все матрицы Qk.

Алгоритм считается вычислительно устойчивым, т. к. производится ортогональными преобразованиями подобия.

**Реализация алгоритма на языке программирования Python**

def qr\_decomposition(self):

        n = self.column

        Q = Matrix(self.row, self.column, [0 for \_ in range(self.row\*self.column)])

        R = Matrix(self.row, self.column, [0 for \_ in range(self.row\*self.column)])

        for i in range(n):

            # Begin the Gram-Schmidt process

            v = self.get\_column(i, 1)

            for j in range(i):

                q\_col = Q.get\_column(j, 1)

                temp = q\_col.dot\_sum(self.get\_column(i, 1))

                R.\_elements[j][i] = temp

                q\_col.mul\_by\_num(temp)

                v = v - q\_col

            R.\_elements[i][i] = v.norm()

            v.divide\_by\_num(R.\_elements[i][i])

            Q.set\_column(v, i)

        return Q, R

    def qr\_algorithm(self, max\_iter):

        q, r = self.qr\_decomposition()

        q\_list = []

        for \_ in range(max\_iter):

            q\_list.append(q.copy())

            A\_i = r \* q

            q, r = A\_i.qr\_decomposition()

        eigenvalues = [A\_i.\_elements[i][i] for i in range(A\_i.row)]

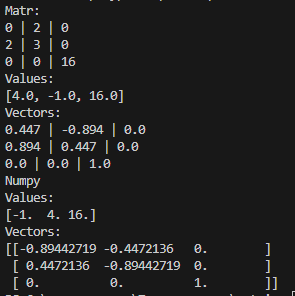
        vectors = q\_list[0]

        for i in range(1, len(q\_list)):

            vectors = vectors \* q\_list[i]

        return eigenvalues, vectors

**Скриншоты работы программы**



**Вывод:** в ходе работы был изучен и реализован QR-алгоритм для нахождения собственных значений и собственных векторов.

**Список литературы**

1. Юдин Д.А. Прикладные аспекты теории матриц: учебное пособие / Д.А. Юдин. - Белгород: Изд-во БГТУ, 2016.

2. QR-алгоритм [Электронный ресурс] // URL: https://ru.wikipedia.org/wiki/QR-алгоритм (дата обращения: 01.12.2020).

3. QR-разложение [Электронный ресурс] // URL: https://ru.wikipedia.org/wiki/QR-разложение (дата обращения: 01.12.2020).

4. Процесс\_Грама\_―\_Шмидтам [Электронный ресурс] // URL: https://ru.wikipedia.org/wiki/Процесс\_Грама\_―\_Шмидтам (дата обращения: 01.12.2020).