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 $A_0=A$. На k-м шаге (начиная с k = 0) вычислим QR-разложение $A_k=Q_kR_k$, где Q_k — ортогональная матрица (то есть $Q_k^T = Q_k^{-1}$), а R_k — верхняя треугольная матрица. Затем мы определяем $A_{k+1} = R_kQ_k$.

Пусть все диагональные миноры матрицы A не вырождены. Тогда последовательность матриц A_k 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 Q_k.

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

```
Matr:
0 | 2 | 0
2 | 3 | 0
0 | 0 | 16
Values:
[4.0, -1.0, 16.0]
Vectors:
0.447 | -0.894 | 0.0
0.894 | 0.447 | 0.0
0.0 | 0.0 | 1.0
Numpy
Values:
[-1.  4. 16.]
Vectors:
[[-0.89442719 -0.4472136   0.        ]
 [ 0.4472136  -0.89442719  0.        ]
 [ 0.          0.          1.        ]]
```

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).