ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»
Институт компьютерных наук и технологий
Высшая школа искусственного интеллекта

ЛАБОРАТОРНАЯ РАБОТА №5

по дисциплине «Элементы теории вероятности и линейной алгебры»

Вариант №11

Выполнил
Студент 3540201/10301 группы Ф.М. Титов

Руководитель
доцент, к.т.н. А.В. Востров

Санкт-Петербург
2021 г.

Оглавление

Задание

Определить собственные значения и собственные векторы матрицы A, затем найти максимальное по модулю собственное число и соответствующий ему собственный вектор с помощью реализованных самостоятельно подпрограмм $\lambda(x0, A)$ и $evect(\lambda 1, A)$.

Матрица A:

$$11.\ A = \begin{pmatrix} 1.2 & 0.5 & 2 & 1 \\ 0.5 & 1 & 0.6 & 2 \\ 2 & 0.6 & 1 & 1 \\ 1 & 2 & 1 & 2 \end{pmatrix}$$

Теоретическая часть

Собственный вектор - ненулевой вектор X, переводящий X в коллинеарный ему вектор, то есть AX = λX, где λ - собственное значение или собственное число оператора A.

Собственным числом матрицы A называется такое число λ, которое обращает в ноль определитель: $A - \lambda E$, где E – единичная матрица.

Геометрический смысл собственного вектора заключается в том, что при преобразовании он дает коллинеарный (параллельный) вектор, т.е. тот же вектор, умноженный на некоторое скалярное значение, которое называется собственным числом. Любой вектор, параллельный данному, также будет собственным вектором, соответствующим тому же собственному значению. Множество таких собственных векторов (вместе с нулевым) образуют собственное подпространство. Пример: ось вращения твердого тела – собственный вектор.

Для нахождения максимальных по модулю собственных чисел будем использовать степенной метод. Нам потребуется алгоритм, представленный на рис.1.

```
λ(x0, A) := | n ← rows(A)
            | norm1 ← |x0|
            | x ← x0
            | kol ← 0
            | while norm1 > TOL
            |   | y ← A·x
            |   | λ1 ← y·x
            |   | norm ← |y|
            |   | for i ∈ 1..n
            |   |    y_i ← y_i / norm
            |   | norm1 ← |y - x|
            |   | x ← y
            |   | kol ← kol + 1
            |   | break if kol > 100
            | λ1
```

Рис.1. Алгоритм нахождения максимального по модулю собственного числа матрицы.

На рис. 2 представлен алгоритм нахождения собственного вектора матрицы.

```
evect(λ, A) := | n ← rows(A)
               | for i ∈ 1.. n
               |    x0_i ← 1
               | E ← identity(n)
               | A1 ← A - λ·E
               | norm ← |x0|
               |
               | while norm > TOL
               |    | y ← lsolve(A1, x0)
               |    | norm ← |y|
               |    | for i ∈ 1.. n
               |    |    | y_i ← y_i / norm
               |    |    | x0_i ← |y_i| - |x0_i|
               |    | norm ← √|x0|
               |    | x0 ← y
               | x0
```

Рис.2. Алгоритм нахождения собственного вектора матрицы.

Степенной метод – итерационный алгоритм поиска собственного значения с максимальной абсолютной величиной и одного из соответствующих собственных векторов для произвольной матрицы. Алгоритм прост и сходится со скоростью геометрической прогрессии если все максимальные по модулю собственные значения совпадают, в противном случае сходимости нет. При близких по модулю собственных значениях сходимость может оказаться медленной.

Реализация

Приведенный алгоритм на рис.1 был реализован. В результате была разработана функция get_lam (рис.3).

```
def get_lam(x0, A, eps):
    norm = get_norm(A)
    x = np.array(x0[:])
    lam = 0
    n_iter = 0

    while norm >= eps:
        n_iter += 1
        y = np.dot(A, x)
        lam = np.dot(np.transpose(y), x)
        y = y / get_norm(y)
        norm = get_norm(y - x)
        x = y[:]

        if n_iter > 100:
            break

    return lam
```

Рис.3. Функция нахождения наибольшего по модулю собственного числа матрицы.

Далее был реализован алгоритм нахождения собственного вектора матрицы (рис.2). В результате была разработана функция get_lam_vector (рис.4). На вход она принимает наибольшее по модулю собственное число, исходную матрицу и точность, с которой необходимо определить значение собственного вектора.

```
def get_lam_vector(lam_1, A, eps):
    x_0 = np.ones(len(A))
    E = np.eye(len(A))
    A_1 = A - lam_1 * E
    norm = get_norm(np.abs(x_0))
    n_iter = 0

    while norm > eps:
        n_iter += 1
        y = np.linalg.solve(A_1, x_0)
        norm = get_norm(np.abs(y))
        y = y / norm
        x_0 = np.abs(y) - np.abs(x_0)
        norm = math.sqrt(get_norm(np.abs(x_0)))
        x_0 = y[:]

    return x_0
```

Рис.4. Функция нахождения собственного вектора матрицы.

Результаты

Исходная матрица *A* была подана на вход функции get_lam дважды. В результате были получены два первых максимальных по модулю собственных числа (рис. 5).

```
A: [[1.2 0.5 2.  1. ]
 [0.5 1.  0.6 2. ]
 [2.  0.6 1.  1. ]
 [1.  2.  1.  2. ]]
lam_1:  4.946131547396801
lam_2:  -0.9104832538170653
```

Рис.5.Результат работы функции get_lam – полученные значения двух первых максимальных по модулю собственных чисел.

Первое собственное число было подано на вход функции нахождения собственного вектора матрицы. Результат – рис. 6.

```
lam_vec: [0.46781073 0.4400703  0.45947672 0.61349196]
```

Рис.6. Полученный собственный вектор матрицы A.

Далее была выполнена проверка (рис. 7).

```
A * lam_vec [2.31385344 2.17664563 2.27263233 3.03441199]
lam_vec * lam_1 [2.31385341 2.17664561 2.27263231 3.03441196]
```

Рис.7. Проверка полученных значений.

Для того, чтобы убедиться в правильности работы алгоритма, он был проверен на данных, представленных в методичке. Результат – рис.8.

```
Пример из методички:
A: [[ 1.  1.  2.  3.]
 [ 0.  2.  2.  4.]
 [ 0.  0.  1. -2.]
 [ 0.  0.  0.  2.]]
lam_1:  2.0008470455928165
lam_2:  1.0000000000000002
[ 0.63245553  0.73786479 -0.21081851  0.10540926]
A * lam_vec [ 1.26491106  1.47572957 -0.42163702  0.21081851]
lam_vec * lam_1 [ 1.26544678  1.47635458 -0.42181559  0.2109078 ]
```

Рис.8. Пример из методички.

Имеем возможность сравнить с результатом, полученным в методичке (рис. 9).

$$x0 := \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \lambda 1 := \lambda(x0, A) \quad \lambda 1 = 2 \quad A1 := A - \lambda 1 \cdot E \quad \lambda 2 := \lambda(x0, A1) + \lambda 1 \quad \lambda 2 = 1$$

$$v1 := \mathrm{evect}(\lambda 1, A) \quad v1 = \begin{pmatrix} 0.632 \\ 0.738 \\ -0.211 \\ 0.105 \end{pmatrix} \quad A \cdot v1 = \begin{pmatrix} 1.265 \\ 1.476 \\ -0.422 \\ 0.211 \end{pmatrix} \quad v1 \cdot \lambda 1 = \begin{pmatrix} 1.265 \\ 1.476 \\ -0.422 \\ 0.211 \end{pmatrix}$$

Рис. 9. Результаты, представленные в методичке.

Заключение

В ходе работы были реализованы алгоритмы нахождения наибольшего по модулю собственного числа матрицы, а также собственного вектора. Далее они были протестированы на заданном наборе значений A, а также на примере из методички. В результате было найдено наибольшее по модулю собственное число, а также собственный вектор матрицы A. Кроме того, полученный результат на примере из методички совпал с результатом, представленном в методичке для того же примера, до 3-го знака после запятой (что обусловлено заданным ε).

Из недостатков степенного метода можно отметить медленную сходимость, пропорциональную $\left|\frac{\lambda_2}{\lambda_1}\right|$. При близких по модулю собственных значениях сходимость будет медленной, или метод будет не сходиться. Кроме того, данный способ находит лишь два максимальных по модулю собственных числа.

Из достоинств степенного метода можно выделить, что так как алгоритм сводится к последовательному умножению заданной матрицы на вектор, он хорошо работает для больших разреженных матриц.