Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Методы численного анализа»

**ОТЧЕТ**

к лабораторной работе №5

на тему:

**«НАХОЖДЕНИЕ СОБСТВЕННЫХ ЗНАЧЕНИЙ**

**И СОБСТВЕННЫХ ВЕКТОРОВ»**

БГУИР 1-40-04-01

|  |
| --- |
| Выполнил студент группы 253504  Дмитрук Богдан Ярославович |
| 10.11.2023 |
| (дата, подпись студента) |
| Проверил  Анисимов Владимир Яковлевич |
|  |
| (дата, подпись преподавателя) |

Минск 2023

**СОДЕРЖАНИЕ**

[1. Цели выполнения задания 3](#_Toc147528122)

[2. Краткие теоретические сведения 4](#_Toc147528123)

[3. Задание 11](#_Toc147528124)

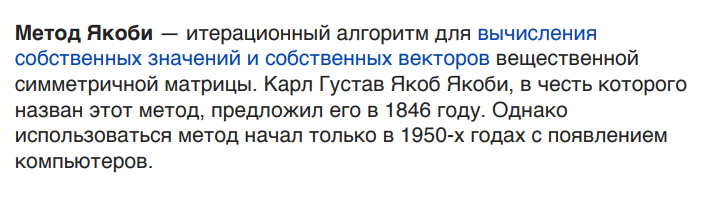
[4. Программная реализация 12](#_Toc147528125)

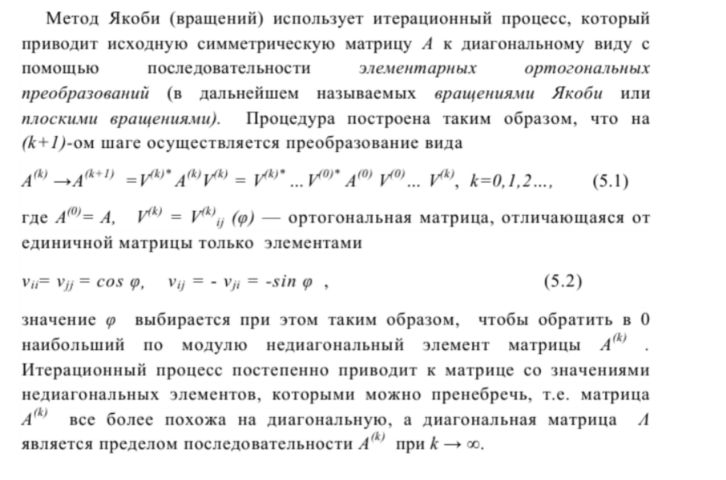
[5. Полученные результаты 16](#_Toc147528126)

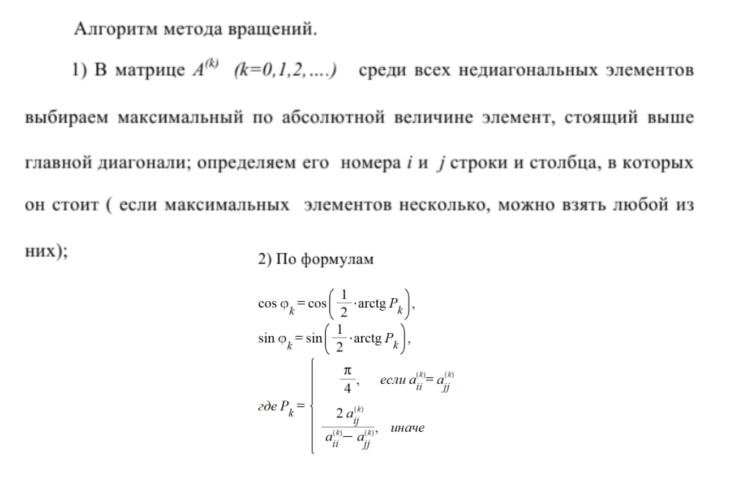
[6. Выводы 18](#_Toc147528127)

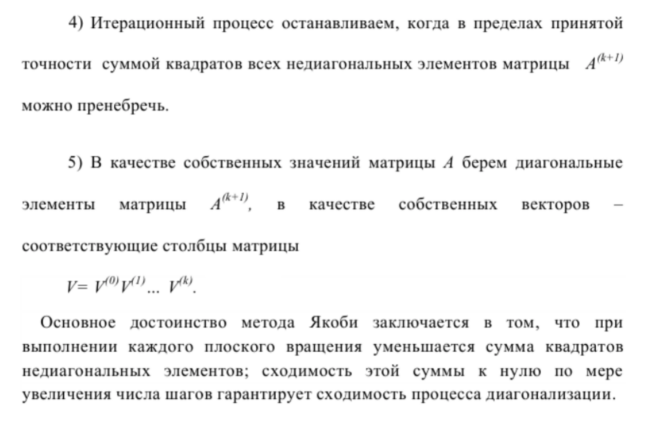
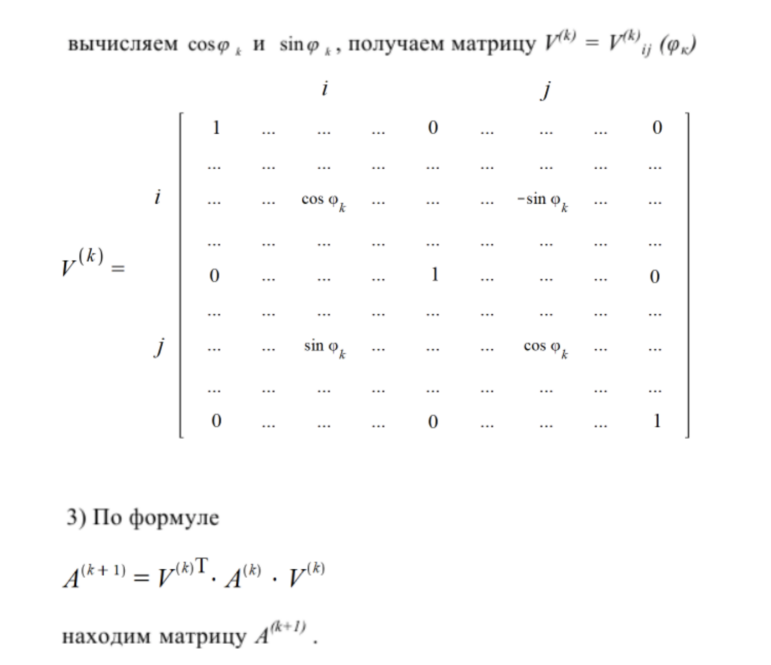
1. ЦЕЛИ ВЫПОЛНЕНИЯ ЗАДАНИЯ
2. Изучить методы вычисления собственных значений и векторов.
3. Разработать программный продукт, реализующий численный метод нахождения собственных значений и векторов методом Якоби.
4. Проверить правильность работы программы на тестовых примерах.
5. Численно найти собственные значения и собственные вектора представленной матрицы.

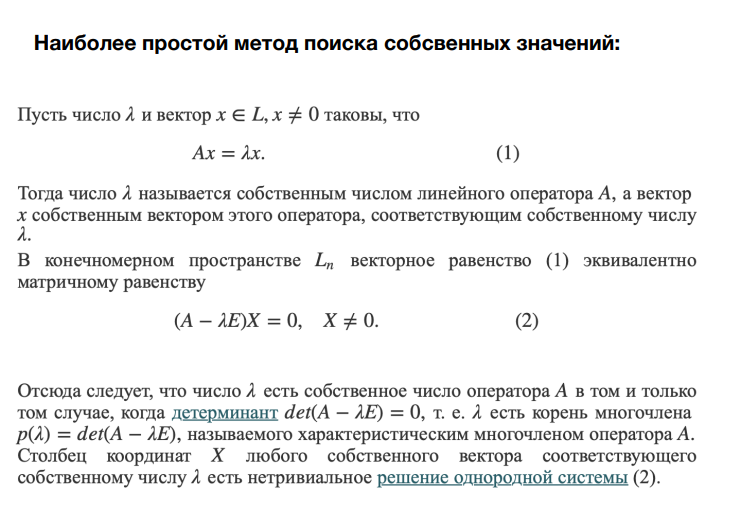
1. **КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**



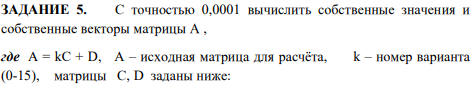


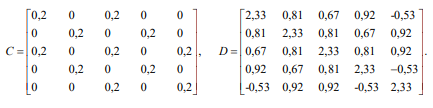






1. **ЗАДАНИЕ**



Вариант 5

1. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

**Метод простых итераций**

def find\_eigen(matrix, tol, verbose=0):

if not check\_equal\_dim(matrix):

raise ValueError("Matrix isn't n, n dimension")

A = matrix.copy()

n = A.shape[0]

rotate\_matrix = np.eye(n)

eig\_vec = np.zeros(shape=A.shape)

iteration = 0

while calc\_non\_diag(A) > tol:

off\_A = off(A)

max\_el, p, q = max\_no\_diag(A)

if A[p, p] == A[q, q]:

if max\_el > 0:

teta = np.pi / 4

else:

teta = -1 \* np.pi / 4

else:

teta = np.arctan((2 \* max\_el) / (A[p, p] - A[q, q])) / 2

for i in range(n):

for j in range(n):

if i == j and i == p:

rotate\_matrix[i, j] = np.cos(teta)

elif i == j and i == q:

rotate\_matrix[i, j] = np.cos(teta)

elif i == p and j == q:

rotate\_matrix[i, j] = np.sin(teta) \* -1

elif i == q and j == p:

rotate\_matrix[i, j] = np.sin(teta)

elif i == j:

rotate\_matrix[i, j] = 1

else:

rotate\_matrix[i, j] = 0

A = rotate\_matrix.T @ A @ rotate\_matrix

off\_A\_new = off(A)

if iteration != 0:

eig\_vec = eig\_vec @ rotate\_matrix

else:

eig\_vec = rotate\_matrix.copy()

iteration += 1

if verbose == 1:

print('Stopping computing...\n')

return A, eig\_vec, iteration

1. ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

Первый тестовый пример:

Вывод:

raise ValueError("Matrix isn't square")

ValueError: Matrix isn't square

Второй тестовый пример:

Вывод:

raise ValueError("Matrix isn't symmetrice")

ValueError: Matrix isn't symmetrice

Третий тестовый пример:

Вывод:

raise ValueError("Matrix is degenerate")

ValueError: Matrix is degenerate

Вывод по исходным данным:

Default Matrix:

[[ 2.93 0.81 1.27 0.92 -0.53]

[ 0.81 2.93 0.81 1.27 0.92]

[ 1.27 0.81 2.93 0.81 1.52]

[ 0.92 1.27 0.81 2.93 -0.53]

[-0.53 0.92 1.52 -0.53 2.93]]

Matrix of Eigenvalues:

[[ 2.50483257e+00 -1.41867366e-08 -9.18896780e-13 1.31912386e-10

-3.22873898e-21]

[-1.41867368e-08 6.06292667e+00 -1.26973473e-13 2.13376925e-06

-3.90280128e-13]

[-9.18690758e-13 -1.26968471e-13 4.08407651e+00 4.03649040e-07

9.04353702e-09]

[ 1.31912298e-10 2.13376925e-06 4.03649040e-07 1.60464917e+00

-1.85391234e-06]

[-7.27657123e-17 -3.90170092e-13 9.04353684e-09 -1.85391234e-06

3.93515080e-01]]

Eigenvalues:

[2.50483256 6.06292879 4.08407692 1.60464985 0.39351324]

Eigenvectors:

[[ 0.56083006 0.43336724 -0.39299836 -0.44297467 0.38339042]

[-0.58446532 0.50628749 0.02367897 -0.54517534 -0.32294955]

[ 0.43480583 0.54770447 0.26768296 0.37703634 -0.54511754]

[-0.3866919 0.42866593 -0.4471151 0.60356929 0.3201673 ]

[-0.0726856 0.26857242 0.75724865 0.0102471 0.59080889]]

Verification:

Eigenvalues:

[6.06292667 4.08407651 0.39351508 2.50483257 1.60464917]

Eigenvectors:

[[ 0.43336702 0.39299843 -0.38338974 -0.56083006 -0.4429754 ]

[ 0.50628724 -0.02367888 0.32295039 0.58446532 -0.5451751 ]

[ 0.54770465 -0.26768302 0.54511696 -0.43480583 0.37703687]

[ 0.42866622 0.447115 -0.32016823 0.3866919 0.60356866]

[ 0.26857242 -0.75724865 -0.5908089 0.0726856 0.01024595]]

1. ВЫВОДЫ

Таким образом, в ходе выполнения лабораторной работы был разработан программный продукт, реализующий численный метод нахождения собственных значений и векторов методом Якоби, позволяющий находить собственные значения и собственные вектора невырожденных симметричных матриц. Реализованы тестовые примеры, проверяющие корректную работу программного продукта и осуществлена проверка при помощи библиотечных функций numpy