



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ _____ «Информатика и системы управления»

КАФЕДРА _____ «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 4.2
по курсу «Численные методы линейной алгебры»
«Вычисление собственных значений и собственных векторов
симметричной матрицы методом А.Н. Крылова»

Студент группы ИУ9-71Б Баев Д.А

Преподаватель Посевин Д. П.

Москва 2023

1 Задание

1. Реализовать метод поиска собственных значений действительной симметричной матрицы A размером 4×4 .
2. Проверить корректность вычисления собственных значений по теореме Виета.
3. Проверить выполнение условий теоремы Гершгорина о принадлежности собственных значений соответствующим объединениям кругов Гершгорина.
4. Вычислить собственные вектора и проверить выполнение условия ортогональности собственных векторов.
5. Проверить решение на матрице приведенной в презентации.
6. Продемонстрировать работу приложения для произвольных симметричных матриц размером $n \times n$ с учетом выполнения пунктов приведенных выше.

2 Исходный код

Исходный код программы представлен в листингах 1–4.

Листинг 1 — Вычисление собственных значений и векторов со всеми проверками

```
1 def eig(A):
2     assert len(A) == len(A[0])
3     assert is_sym_matrix(A)
4     n = len(A)
5     intervals = gerchgoin_intervals(A)
6     trace = sum(A[i][i] for i in range(n))
7     coefs, B = danilevskiy(A)
8     coefs = list(map(lambda x: x * -1, coefs))
9     f = lambda x: np.polyval([1] + coefs, x)
10    search_intervals = binary_search_intervals(intervals, f)
11    eigs = binary_search_roots(search_intervals, f)
12    check_gerchgoin(intervals, eigs)
13    print(f"Viet theorem:\nSum = {sum(eigs)}\nTrace = {trace}")
14    eig_vectors = []
15    for eig in eigs:
16        y_vector = [eig ** i for i in range(n - 1, -1, -1)]
17        x_vector = np.array(mul_matrix_by_vector(B, y_vector))
18        eig_vectors.append(x_vector / norm(x_vector))
19    print("Ortnorm eig vectors:")
20    print('Norms:')
21    for vector in eig_vectors:
22        print(norm(vector))
23    print("Scalar prods:")
24    for i in range(n - 1):
25        for j in range(i + 1, n):
26            print(scalar(eig_vectors[i], eig_vectors[j]))
27    return np.array(eigs), eig_vectors
28
```

Листинг 2 — Вычисление коэффициентов характеристического уравнения и собственных векторов методом А.Н. Крылова

```
1 def krylov_eig_vectors(A, Y, coefs, eigs):
2     n = len(A)
3     Q = np.zeros(shape=(n, n))
4     for i in range(n):
5         for j in range(n):
6             Q[j][i] = 1 if j == 0 else eigs[i] * Q[j - 1][i] + coefs[
7 j - 1]
8     vectors = []
9     for i in range(n):
10        vector = deepcopy(Y[0])
11        for j in range(1, n):
12            vector += Q[j][i] * Y[j]
13        vectors.append(vector / norm(vector))
14    return vectors
15
16 def krylov(A):
17     n = len(A)
18     y = np.random.uniform(-10, 10, n)
19     Y = [y]
20     for i in range(n - 1):
21         y = mul_matrix_by_vector(A, y)
22         Y.append(y)
23     b = mul_matrix_by_vector(A, y)
24     Y = np.array(list(reversed(Y)))
25     P = Y.T
26     coefs = np.linalg.solve(P, b)
27     return coefs, Y
28
```

Листинг 3 — Вычисление и объединение кругов Гершгорина

```
1 def gerchgoin_intervals(A):
2     intervals = []
3     for i in range(len(A)):
4         center = A[i][i]
5         radius = sum(abs(A[i][j]) for j in range(len(A)) if j != i)
6         intervals.append([center - radius, center + radius])
7
8     intervals.sort(key=lambda x:x[0])
9     merged = [intervals[0]]
10    for i in range(1, len(intervals)):
11        current_interval = intervals[i]
12        previous_interval = merged[-1]
13        if current_interval[0] <= previous_interval[1]:
14            merged[-1] = [previous_interval[0], max(
previous_interval[1], current_interval[1])]
15        else:
16            merged.append(current_interval)
17
18    return merged
19
20 def check_gerschgoin(intervals, eigs):
21     for eig in eigs:
22         interval_found = False
23         for interval in intervals:
24             if interval[0] <= eig <= interval[1]:
25                 interval_found = True
26                 break
27         if not interval_found:
28             print(f"Eig: {eig} not in gerschgoin circles")
29         return
30     print("All eigs in gerschgoin circles")
31
```

Листинг 4 — Решение характеристического уравнения методом бинарного поиска

```
1  def binary_search_roots(intervals , f, delta=1e-3):
2      lambdas = []
3      for interval in intervals:
4          left = interval[0]
5          right = interval[1]
6          f_left = f(left)
7          assert f_left * f(right) < 0
8          x = (left + right) / 2
9          f_x = f(x)
10         while abs(f_x) > delta:
11             if f_left * f_x < 0:
12                 right = x
13             else:
14                 left = x
15                 f_left = f(left)
16             x = (left + right) / 2
17             if x == left or x == right:
18                 break
19             f_x = f(x)
20         lambdas.append(x)
21     return lambdas
22
23 def binary_search_intervals(intervals , f, delta=0.1):
24     search_intervals = []
25     for interval in intervals:
26         left = interval[0]
27         right = interval[1]
28         delta_x = left + delta
29         while delta_x <= right:
30             f_left = f(left)
31             f_delta = f(delta_x)
32             if f_left * f_delta < 0:
33                 search_intervals.append([left , delta_x])
34                 left = delta_x
35                 delta_x = left + delta
36             else:
37                 delta_x += delta
38     return search_intervals
39
```

3 Результаты

Результат поиска собственных значений и векторов и все необходимые проверки представлены на рисунках 1- 3.

```

[-1.4200744628906299, 0.22260742187499594, 1.5453979492187466, 5.652032089233383]
[[-3.5999999999999996, 6.6]]
All eigs in gerchgojn circles
Viet theorem:
Sum = 5.999962997436495
Trace = 6.0
Ortnorm eig vectors:
Norms:
1.0
1.0
1.0
0.9999999999999999
Scalar prods:
2.281237853890561e-05
-1.0639576947926721e-05
1.5253399213718495e-06
-1.5775523831695892e-06
3.2854732530118724e-06
-3.5110161853563193e-06
(array([-1.42007446, 0.22260742, 1.54539795, 5.65203209]), [array([ 0.22204419, -0.51590363, 0.75727447, -0.33327947]), array([ 0.52190966,
0.45487211, -0.15342945, -0.7050965 ]), array([ 0.62893951, -0.57256172, -0.48565936, 0.20184935]), array([-0.53173605, -0.44619422,
-0.40881553, -0.59248405])])

```

Рис. 1 — Результат поиска собственных значений и векторов для матрицы 4x4 (из презентации)

```

[-44.863550868814244, -30.443938002251883, -21.002865804308573, -14.07909580591684, -2.8756463962222, 4.137744704540937, 21.436605088920075,
23.081645659570057, 30.222445798499336, 32.425845260444504]
[[-95.02167170882848, 76.6219343623794]]
All eigs in gerchgojn circles
Viet theorem:
Sum = -1.9608103655388192
Trace = -1.960810365562466
Ortnorm eig vectors:
Norms:
1.0
1.0
1.0
1.0
1.0
1.0
0.9999999999999999
0.9999999999999999
1.0
Scalar prods:
-1.0644263248593688e-14
-1.5182299861749016e-14
-2.0777823905859805e-13
-1.1672433852805142e-13
-1.6819878823071122e-14
-2.26554885962571e-13
-5.830691832131585e-13
-2.771394225220547e-14
-1.5404344466674047e-15
2.958744360626042e-14
4.641287354445467e-13
2.9493768538557674e-14
4.746203430272544e-15
3.4416913763379805e-14
2.7236806560138005e-13

```

Рис. 2 — Результат поиска собственных значений и векторов для произвольной матрицы 10x10

```

[-69.385778689284, -62.08422232763795, -53.87445780174059, -41.77966435803846, -31.517114465035625, -28.624218578056862, -20.25121858722426,
-15.02795382026489, -4.98465463338699, 0.49691931838805337, 9.413542069090838, 11.583761471874631, 22.114662442273335, 25.708133614217584,
31.945712128470987, 36.395484514408736, 45.67433124842184, 48.99992801064862, 51.9629727891474, 64.12826283366968]
[[-178.34186926834576, 176.0018877705777]]
All eigs in gerchgoiin circles
Viet theorem:
Sum = 20.066427171941456
Trace = 20.0664214996477
Ortnorm eig vectors:
Norms:
0.9999999999999999
1.0000000000000002
0.9999999999999999
0.9999999999999998
0.9999999999999999
1.0
1.0
0.9999999999999999
1.0
1.0
1.0
0.9999999999999998
0.9999999999999999
0.9999999999999999
1.0
1.0
1.0
1.0
1.0
1.0
Scalar prods:
1.2963692619383238e-12
2.6799258992515895e-11
1.0513458159611133e-10
1.6948458331222938e-09
1.407581423813753e-09

```

Рис. 3 — Результат поиска собственных значений и векторов для произвольной матрицы 20x20

4 Выводы

В результате выполнения лабораторной работы был реализован метод для поиска собственных значений и нормированной системы собственных векторов произвольной квадратной действительной симметричной матрицы. Для поиска собственных значений метод использует метод бинарного поиска, локализуя область поиска с помощью теоремы Гершгорина. Коэффициенты характеристического уравнения и собственные вектора вычисляются при помощи метода А.Н. Крылова.