**Лабораторная работа №8**

«Метод Крылова»

выполнил Пажитных Иван, 2-й курс, 1-я группа

**1) Постановка задачи**

Необходимо найти собственные значения и собственные векторы матрицы А:

,

Для этого требуется:

1. Построить Pn(() – собственный (характеристический) многочлен матрицы А.
2. Решить уравнение Pn( и найти
3. Найти собственные векторы:

**2)Алгоритм решения**

Метод Крылова является прямым методом решения полной задачи собственных значений (т.е. позволяет весь спектр). Согласно теорема Гамильтона-Кэли матрица А удовлетворяет своему характеристическому многочлену, поэтому: Pn(() = 0

Возьмём произвольный, ненулевой вектор *c0 =* ( согласованный с исходной матрицей и рекуррентно вычислим *ci* = *Aci-1,. cn* будет являться линейно-независимой комбинацией векторов *c0, c1,…, cn-1*:

…

Покоординатно расписывая это равенство, придём к системе из n линейных уравнений от n неизвестных …. В матричном виде: . Решая её найдём коэффициенты характеристического многочлена Pn(.

Из уравнения Pn( находим Далее найдём собственный вектор *x* соответствующий собственному значению Для этого распишем *x* в базисе (*c0, c1,…, cn-1):*  …, найдём по формулам:

**3) Листинг программы**

a = np.**array**(A)

At = a.**transpose**() *# находим Аt*

a = np.**dot**(At, a) *# перемножаем А на Аt, теперь А - симметрическая*

c = []

c.**append**([1, 0, 0, 0, 0]) *# начальный сo*

**for** i **in** range(1, n + 1):

c.**append**(np.**dot**(a, c[i - 1])) *# рекурсивно вычисляем ci*

C = np.**array**(c) *# копия С*

cn = c.**pop**() *# выделяем столбец свободных членов cn*

c = np.**array**(c).**transpose**()

**for** i **in** range(n): *# транспонируем матрицу коэффициентов C*

c[i] = **list**(**reversed**(c[i]))

p = np.linalg.**solve**(c, cn) *# решаем систему C\*p = cn*

x = **Symbol**('x') *# вычисляем собственные значения*

Lambda = **solve**(x\*\*5 - p[0] \* x\*\*4 - p[1] \* x\*\*3 - p[2] \* x\*\*2 - p[3] \* x - p[4], x)

l = **max**(Lambda) *# максимальное собственное*

b = ones(n)

**for** i **in** range(1, n): *# находим коэффиценты ß*

b[i] = b[i - 1] \* l - p[i - 1]

x = np.**sum**([b[i] \* C[n - i - 1] **for** i **in** range(n)], axis=0) *# вычисляем собственный вектор x*

r = np.**dot**(a, x) - l \* x *# находим вектор невязки*

rnorm = np.linalg.**norm**(r, 1) *# находим норму невязки*

**4) Результат и его анализ**

Симметрическая *АAT*:

[[ 0.70536135 0.01441237 0.13398766 -0.08030921 0.5676231 ]  
 [ 0.01441237 1.22673234 -0.00165256 0.11340719 0.05855825]  
 [ 0.13398766 -0.00165256 0.77976056 -0.21682262 0.2943685 ]  
 [-0.08030921 0.11340719 -0.21682262 0.79926611 -0.0500214 ]  
 [ 0.5676231 0.05855825 0.2943685 -0.0500214 1.07689486]]

Векторы *Сi*:

[[ 1. 0. 0. 0. 0. ]

[ 0.70536135 0.01441237 0.13398766 -0.08030921 0.5676231 ]

[ 0.8443406 0.051756 0.38346741 -0.17664583 1.05595268]

[ 1.26126047 0.11682784 0.76119847 -0.33909049 1.74116478]

[ 2.0088768 0.22374058 1.34841937 -0.6112058 2.83884785]

[ 3.26136131 0.39811696 2.28842983 -1.05884457 4.63803295]]

Коэффициенты собственного многочлена *P(*:

[ 4.58801522 -7.82119475 6.11344651 -2.15665219 0.2685558 ]

Собственные значения :

[0.274152549963745, 0.550498737833298, 0.857845170065991, 1.26253883985411, 1.64297992228287]

Максимальное собственное

1.64297992228287

Коэффициенты *ßi*:

[ 1. -2.9450353 2.98256088 -1.21315887 0.16345653]

Собственный вектор матрицы *А —*

[ 0.12045857 0.0165594 0.08782958 -0.04200146 0.17188228]

Вектор невязки *r*:

[3.57769369685457e-14 4.51028103753970e-17 1.38777878078145e-16

-5.55111512312578e-17 -3.88578058618805e-16]

Норма *||r||*:

3.64049068668493E-14

С помощью метода Крылова мы нашли все собственные значения и по ним можем построить полный спектр. Для найденного собственного вектора *x* матрицы *А* соответствующего максимальному собственному значению норма вектора невязки получилась порядка 10-14, что говорит о том, что собственный вектор найден с достаточной точностью. Сравнивая с методом Данилевского, метод Крылова является более точным (в МД невязка порядка 10-10), однако и более трудоёмким.