МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Курсовая работа
по курсу «Численные методы»

Тема: «Нахождение собственных значений и собственных векторов симметричных разреженных матриц большой размерности. Метод Ланцоша».

Выполнил: Янтиков К.А.
Группа: М8О-402Б-20
Преподаватель: Пивоваров Д.Е.
Дата:
Оценка:
Подпись:

Москва, 2023

Условие

Написать программу для нахождение собственных значений и собственных векторов симметричных разреженных матриц большой размерности методом Ланцоша

Описание алгоритма

Алгоритм Ланцоша соединяет в себе метод Ланцоша для построения крыловского подпространства с процедурой Рэлея-Ритца. Входными данными алгоритма служат квадратная матрица $A=A^T$ и вектор начального приближения b. Необходимо найти трехдиагональную симметричную матрицу $T_{k\ k}= Q^TAQ_k$, собственные значения которой приближают собственные значения матрицы A. Иными словами, на k-м шаге из ортонормированных векторов Ланцоша строится матрица $Q_k = [q_1,q_2,...,q_k,]$ и в качестве приближенных собственных значений матрицы A принимаются числа Ритца.

Пусть $T_k = V \Lambda V^T$ – есть спектральное разложение матрицы T_k, столбцы матрицы Q_kV рассматриваются как приближения к соответствующим собственным векторам матрицы A. Диагональные элементы обозначены как $\alpha_j = t_{jj}$, а элементы побочной диагонали $\beta_j = t_j - 1, j = t_j, j - 1$. После каждой итерации мы вычисляем α_j, β_j, из которых строится матрица T.

Алгоритм:

1) Заполняются начальные значения
   $q_1 = b/||b||,$
   $\beta_1 = 0,$
   $q_0 = 0,$
   где b - произвольный вектор, для всех j = 1..k
2) Пусть $z = Aq_j$
3) Вычисляется элемент на позиции t_{jj} матрицы Tk: $\alpha_j \underset{J}{=} q^Tz$
4) Ортогонализация Грамма-Шмидта:
   $z = z - \sum_{i=1}^{j-1}(z^T q_i)q_i$
   $z = z - \sum_{i=1}^{j-1}(z^T q_i)q_i$
5) Обновление: $z = z - \alpha_j q_j - \beta_j q_{j-1}$
6) Вычисляются элементы на позициях $t_{j,j+1}$и $t_{j+1,j}$: $\beta_{j+1} = ||z||$
7) Если $\beta_{j+1} = 0$, то алгоритм завершается

Код программы:

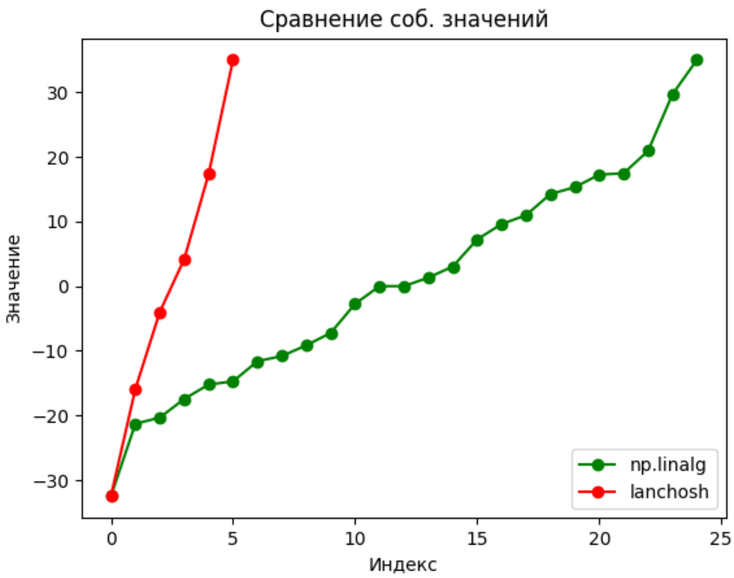
```
import numpy as np
import matplotlib.pyplot as plt

def matrixCreate():
    A = np.zeros((25, 25))
    n = np.random.randint(20, size=40)
    for k in range(len(n)):
            i = np.random.randint(25)
            j = np.random.randint(25)
            A[i][j] = n[k]
            A[j][i] = n[k]
    return A

def maxElem(A):
    i_max, j_max, max_elem = 0, 0, 0
    for i in range(A[0].size):
        for j in range(i+1, A[0].size):
            if (abs(A[i][j])>max_elem):
                max_elem = abs(A[i][j])
                i_max = i
                j_max = j
    return i_max, j_max, max_elem

def rotation(A, eps):
    Ak = np.copy(A)
    eigen_vectors =  np.eye(A[0].size)
    i_max, j_max, max_elem = maxElem(Ak)
    count = 0
    while (max_elem>eps):
        phi = 0.5*np.arctan(2*Ak[i_max][j_max]/(Ak[i_max][i_max]-
Ak[j_max][j_max]))
        U = np.eye(Ak.shape[0])
        U[i_max][j_max] = -np.sin(phi)
        U[j_max][i_max] = np.sin(phi)
        U[i_max][i_max] = np.cos(phi)
        U[j_max][j_max] = np.cos(phi)
        Ak = U.T @ Ak @ U
        eigen_vectors = eigen_vectors @ U
        count += 1
        i_max, j_max, max_elem = maxElem(Ak)
    eigen_values = np.array([Ak[i][i] for i in range(A[0].size)])
    return eigen_vectors, eigen_values, count

def lanchosh(A, b, iters, EPSILON):
    Q = np.zeros((A.shape[0], iters + 1))
    alpha = np.zeros((iters))
    beta = np.zeros((iters))
    Q[:,0] = b/np.linalg.norm(b)
    for m in range(iters):
```

```
        if np.linalg.norm(b) <= EPSILON:
            break
        v = np.dot(A, Q[:,m])
        alpha[m] = np.dot(Q[:,m],v)
        if m == 0:
            v = v - alpha[m]*Q[:,m]
        else:
            v = v - alpha[m]*Q[:,m] - beta[m-1]*Q[:,m-1]
        beta[m] = np.linalg.norm(v)
        Q[:,m+1] = v/beta[m]

    T = np.dot(np.dot(Q.T,A), Q)
    Vec_T, Val, _ = rotation(T, 1e-16)
    Vec = Q@Vec_T
    return Vec, Val

A = matrixCreate()
np.set_printoptions(suppress=True)
EPSILON = 0.00000000000000001
b = np.random.rand(A.shape[0])

iters = 25
Vec, Val = lanchosh(A, b, iters, EPSILON)
linalg_eigenvalues = np.sort(np.real(np.linalg.eigvals(A)))
kp_eigenvalues = np.sort(Val)
print("np.linalg:", linalg_eigenvalues)
print("lanchosh:", kp_eigenvalues)

plt.plot(linalg_eigenvalues, 'o-', label='np.linalg',color='green')
plt.plot(kp_eigenvalues, 'o-', label='lanchosh',color='red')
plt.title("Сравнение соб. значений")
plt.xlabel("Индекс")
plt.ylabel("Значение")
plt.legend()
plt.show()
```

Вывод программы

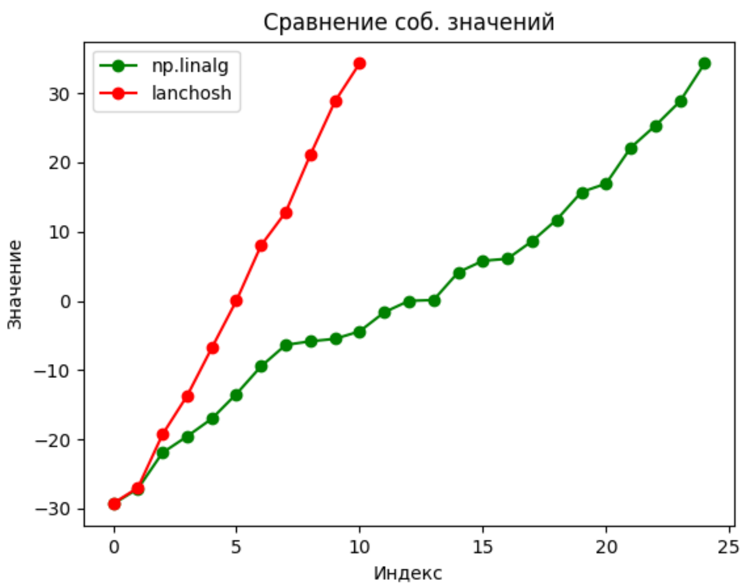
5 итераций

```
np.linalg: [-32.4686225  -21.27869597 -20.32903461 -17.44494819 -15.21308399
 -14.72311938 -11.61263662 -10.8018017   -9.14930438  -7.2322524
  -2.72858013  -0.00712853   0.           1.31879625   3.03621896
   7.23156748   9.60122758  10.97022433  14.25344508  15.29606693
  17.25264031  17.45243352  20.91684781  29.67476208  34.98497808]
lanchosh: [-32.44808593 -15.88946671  -3.98789887   4.17117145  17.3288117
  34.98358847]
```



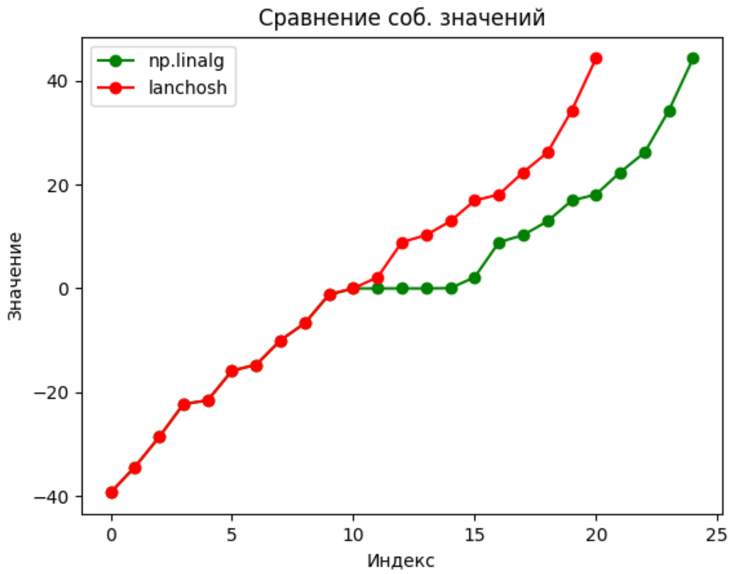
10 итераций

```
np.linalg: [-29.35597374 -27.16006948 -21.95015805 -19.58670994 -17.03271272
 -13.4583041   -9.46249854  -6.36865354  -5.84430702  -5.49421067
  -4.40686336  -1.65327964  -0.           0.11840469   4.12290146
   5.78555312   6.08295835   8.63624989  11.7407459   15.71676586
  16.94463804  22.14286873  25.32109911  28.83629747  34.32525819]
lanchosh: [-29.26445955 -27.0215379  -19.31190796 -13.71580939  -6.80234904
   0.03528754   8.01683011  12.82243086  21.10215286  28.83512171
  34.32525169]
```



20 итераций

```
np.linalg: [-39.30094859 -34.37275324 -28.61749064 -22.26802173 -21.55300315
 -15.83972576 -14.68692916  -9.9618968   -6.70364972  -1.23253137
  -0.          0.           0.           0.           0.05614305
   2.17437155   8.89527664  10.28557961  12.94619411  16.92938676
  18.10095678  22.38835207  26.17776141  34.25972121  44.32320697]
lanchosh: [-39.30094859 -34.37275324 -28.61749064 -22.26802173 -21.55300306
 -15.83972438 -14.68692827  -9.96188772  -6.70364286  -1.12105439
   0.00649961   2.17413945   8.89524575  10.28510655  12.94618592
  16.92938465  18.1009557   22.38835207  26.17776141  34.25972121
  44.32320697]
```



Вывод

В ходе исследования метода Ланцоша, основанного на использовании подпространств Крылова, для анализа собственных значений и векторов разреженной несимметричной матрицы. Я приобрел глубокое понимание этого эффективного численного метода. В процессе работы над курсовой работой, я освоил ключевые принципы формирования подпространств Крылова, их взаимодействия с матрицей, а также специфику работы с разреженными несимметричными матрицами.