Проблема собственных значений. Вариант 11

Содержание

Ввод данных

Введем матрицу системы

```
A = [-0.93016,-0.2577,0.45254; -0.25770,0.65022,0.07193; 0.45254,0.07193,-0.97112]
```

```
A = 3×3
   -0.9302   -0.2577    0.4525
   -0.2577    0.6502    0.0719
    0.4525    0.0719   -0.9711
```

Метод Якоби

Алгоритм

Для нахождения собственных значений матрицы A приведем ее к виду $\Lambda = V^T A V$. Для этого построим последовательность матриц $A^{(0)} = A, A^{(1)}, \ldots, A^{(k)}, \ldots$, где $A^{(k+1)} = V^T_{i_k j_k}(\phi_k) A^{(k)} V_{i_k j_k}(\phi_k)$.

$V(\phi_k) = (v_{ij}(\phi_k))_{i,j=1}^{n}$, где

- $v_{ii} = 1, i \neq i_k \ \& \ i \neq j_k$
- $v_{i_k i_k} = c, v_{j_k j_k} = c, c = \cos(\phi_k)$
- $v_{ij} = 0, i \neq i_k \ \& \ i \neq j_k \ \& \ j \neq i_k \ \& \ j \neq j_k$
- $v_{i_k j_k} = -s, v_{j_k i_k} = s, s = \sin(\phi_k)$

i_k, j_k выбираются из соотношения $|a^{(k)}_{i_k j_k}| = \max\limits_{i,j \in 1:n,\ i<j} |a^{(k)}_{ij}|$. Значения c и s можно посчитать следующим образом:

- $$c = \sqrt{\frac{1}{2}(1 + \frac{|a_{i_k i_k} - a_{j_k j_k}|}{d})}$$
- $$s = sign(a_{i_k j_k}(a_{i_k i_k} - a_{j_k j_k}))\sqrt{\frac{1}{2}(1 - \frac{|a_{i_k i_k} - a_{j_k j_k}|}{d})}$$

где $d = \sqrt{(a_{i_k i_k} - a_{j_k j_k})^2 + 4a_{i_k j_k}^2}.$

Собственные числа будут диагональными элементами матрицы $A^{(k)}$. Собственные векторы будут столбцами матрицы $X = V_{i_0 j_0} \dots V_{i_k j_k}$.

Построение новых приближений стоит завершить, если $|a_{i_k j_k}^{(k)}| < \epsilon$, где ϵ - заранее заданная точность.

Реализация

Реализация алгоритма находится в функции jacobiMethod.m

Результат работы

```
[eigVals,X] = jacobiMethod(A)
```

```
eigVals = 3×3
   -0.5129   -0.0000   -0.0000
   -0.0000    0.6912    0.0000
   -0.0000    0.0000   -1.4293
X = 3×3
    0.6983   -0.1570   -0.6983
    0.1110    0.9876   -0.1110
    0.7071    0.0000    0.7071
```

Сравним с результатом работы встроенного метода языка Matlab.

```
[matlab_eigVectors,matlab_eigVals] = eig(A)
```

```
matlab_eigVectors = 3×3
   -0.6983    0.6983    0.1570
   -0.1110    0.1110   -0.9876
    0.7071    0.7071   -0.0000
matlab_eigVals = 3×3
   -1.4293         0         0
         0   -0.5129         0
         0         0    0.6912
```

Степенной метод

Алгоритм

Подберем вектор $Y^{(0)}$ и построим по нему последовательность $Y^{(0)}, \dots, Y^{(k)}, \dots$, где $Y^{(k+1)} = AY^{(k)},\ Y^{(k)} = (y_1^{(k)}, \dots, y_n^{(k)})^T$. Затем найдем $p : |y_p^{(k)}| = \max\limits_{1 \le i \le n} |y_i^{(k)}|$ и нормируем $Y^{(k)}$ следующим образом: $Y^{(k)}_{norm} = \frac{Y^{(k)}}{y^{(k)}_p}$. Тогда искомое максимальное по модулю собственное число $(\lambda_1^{(k,p)})_{pow} = y_p^{(k+1)}$, а соответствующий ему собственный вектор - $X = Y^{(k)}$.

Построение новых приближений стоит завершить, когда $\frac{||AY^{(k)} - \lambda Y^{(k)}||_2}{||Y^{(k)}||_2} < \epsilon$.

Реализация

Реализация алгоритма описана в функции powerMethod.m

Результат работы

```
[pow_eigVal,pow_eigVect,pow_n] = powerMethod(A)
```

```
pow_eigVal = -1.4294
pow_eigVect = 3×1
   -0.6984
   -0.1113
    0.7070
pow_n = 9
```

Метод скалярных произведений

Алгоритм

Метод скалярных произведений почти полностью совпадает с описанным выше степенным методом. Отличия заключаются в следующем:

- $(\lambda_1^{(k)})_{scal} = \frac{(Y^{(k+1)}_{norm}, Y^{(k)}_{norm})}{(Y^{(k)}_{norm}, Y^{(k)}_{norm})}$
- Построение новых приближений завершается, если $|(\lambda_1^{(k+1)})_{scal} - (\lambda_1^{(k)})_{scal}| < \epsilon$

Реализация

Реализация алгоритма описана в функции dotProdMethod.m

Результат работы

```
[dot_eigVal,dot_n] = dotProdMethod(A)
```

```
dot_eigVal = -1.4292
dot_n = 5
```

Сравнение со степенным методом

Как можно заметить, метод скалярных произведений сходится быстрее, чем степенной метод.