**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет прикладной математики и физики

Кафедра вычислительной математики и программирования

**Лабораторная работа № 1**

по курсу «Численные методы»

Студент: Стрыгин Д.Д.

Группа: 80-306Б

Преподаватель: Ревизников Д.Л.

Вариант: 25

Оценка:

Москва, 2022

1. **Постановка задачи**

1.1. Реализовать алгоритм LU - разложения матриц (с выбором главного элемента) в виде программы. Используя разработанное программное обеспечение, решить систему линейных алгебраических уравнений (СЛАУ). Для матрицы СЛАУ вычислить определитель и обратную матрицу.



1.2. Реализовать метод прогонки в виде программы, задавая в качестве входных данных ненулевые элементы матрицы системы и вектор правых частей. Используя разработанное программное обеспечение, решить СЛАУ с трехдиагональной матрицей.



1.3. Реализовать метод простых итераций и метод Зейделя в виде программ, задавая в качестве входных данных матрицу системы, вектор правых частей и точность вычислений. Используя разработанное программное обеспечение, решить СЛАУ. Проанализировать количество итераций, необходимое для достижения заданной точности.



1.4. Реализовать метод вращений в виде программы, задавая в качестве входных данных матрицу и точность вычислений. Используя разработанное программное обеспечение, найти собственные значения и собственные векторы симметрических матриц. Проанализировать зависимость погрешности вычислений от числа итераций.



1.5. Реализовать алгоритм QR – разложения матриц в виде программы. На его основе разработать программу, реализующую QR – алгоритм решения полной проблемы собственных значений произвольных матриц, задавая в качестве входных данных матрицу и точность вычислений. С использованием разработанного программного обеспечения найти собственные значения матрицы.



1. **Описание программы**

Программа работает, используя функции считывания матрицы, вывода результата и непосредственно реализованных методов в виде одноимённых функций.

Также программа обладает выводом справки по использованию, если была вызвана с неправильными аргументами.

Реализовано два вида ввода входных данных: с консоли и из файла по специализированным ключам.

1. **Примеры работы программы**
2. Решение СЛАУ с помощью LU разложения.

Входные данные:

|  |
| --- |
| -5x^1 - 6x^2 + 4x^3 - 2x^4 = 64  3x^2 - 4x^3 - 6x^4 = -55  2x^1 + 4x^2 - 4x^3 + 2x^4 = -48  x^1 - 8x^2 + 2x^3 + 8x^4 = 68 |

Вывод:

|  |
| --- |
| Матрица A:  [-5.0, -6.0, 4.0, -2.0]  [0.0, 3.0, -4.0, -6.0]  [2.0, 4.0, -4.0, 2.0]  [1.0, -8.0, 2.0, 8.0]  Вектор b:  [64.0, -55.0, -48.0, 68.0]  Матрица A(после перестановок):  [-5.0, -6.0, 4.0, -2.0]  [1.0, -8.0, 2.0, 8.0]  [0.0, 3.0, -4.0, -6.0]  [2.0, 4.0, -4.0, 2.0]  Вектор b(после перестановок):  [64.0, 68.0, -55.0, -48.0]  Матрица U:  [-5.0, -6.0, 4.0, -2.0]  [0.0, -9.2, 2.8, 7.6]  [0.0, 0.0, -3.087, -3.522]  [0.0, 0.0, 0.0, 4.704]  Матрица L:  [1.0, 0.0, 0.0, 0.0]  [-0.2, 1.0, 0.0, 0.0]  [-0.0, -0.326, 1.0, 0.0]  [-0.4, -0.174, 0.62, 1.0]  Определитель А: -667.977408  Обратная матрица к A:  [-0.341, 0.096, 0.108, -0.401]  [0.084, -0.006, -0.132, 0.213]  [0.012, -0.144, -0.162, 0.102]  [-0.117, -0.099, -0.174, -0.243]  Решение СЛАУ:  [-2.007112718446908, 2.0042312925170065, -4.9973625763822245, 6.99718088362653] |

1. Решение СЛАУ методом прогонки

Входные данные:

|  |
| --- |
| 12x^1 - 5x^2 = 148  -3x^1 - 18x^2 - 8x^3 = 45  -2x^2 - 16x^3 - 9x^4 = -155  -4x^3 + 18x^4 - 7x^5 = 11  4x^4 - 9x^5 = 3 |

Вывод:

|  |
| --- |
| Матрица A:  [12.0, -5.0, 0.0, 0.0, 0.0]  [-3.0, -18.0, -8.0, 0.0, 0.0]  [0.0, -2.0, -16.0, -9.0, 0.0]  [0.0, 0.0, -4.0, 18.0, -7.0]  [0.0, 0.0, 0.0, 4.0, -9.0]  Решение СЛАУ:  [9.0, -8.0, 9.0, 3.0, 1.0] |

1. Решение СЛАУ методом простых итераций и Зейделя

Входные данные:

|  |
| --- |
| 15x^1 - 4x^2 - 6x^3 + 5x^4 = 104  4x^1 - 14x^2 - x^3 +4x^4 = 70  7x^1 - 7x^2 + 27x^3 - 8x^4 = 170  -3x^1 - 3x^2 + 2x^3 - 14x^4 = 48 |

Вывод (метод простых итераций):

|  |
| --- |
| Задайте точность: 0.01  Матрица A:  [15.0, -4.0, -6.0, 5.0]  [4.0, -14.0, -1.0, 4.0]  [7.0, -7.0, 27.0, -8.0]  [-3.0, -3.0, 2.0, -14.0]  Вектор b:  [104.0, 70.0, 170.0, 48.0]  Матрица alpha:  [0.0, 0.267, 0.4, -0.333]  [0.286, 0.0, -0.071, 0.286]  [-0.259, 0.259, 0.0, 0.296]  [-0.214, -0.214, 0.143, 0.0]  Вектор betta:  [6.933333333333334, -5.0, 6.296296296296297, -3.4285714285714284]  Решение СЛАУ методом простых итераций:  [8.001925308093929, -4.001838300018997, 2.0006412044425472, -3.99907802759158]  Количество итераций: 9  Решение СЛАУ методом Зейделя:  [8.000132184830234, -3.9980467886037667, 2.005118576944259, -3.998285746860863]  Количество итераций: 6 |

1. Отыскание собственных значений и векторов методом вращений.

Входные данные:

|  |
| --- |
| 5 -4 7  -4 -3 4  7 4 1 |

Вывод:

|  |
| --- |
| Задайте точность: 0.01  Матрица A:  [10.396, 0.009, -0.027]  [0.009, 1.989, 0.032]  [-0.027, 0.032, -9.407]  Количество итераций: 50  Собственные значения: [10.396, 1.989, -9.407]  Матрица собственных векторов:  [0.815, -0.346, -0.475]  [-0.073, 0.75, -0.668]  [0.58, 0.578, 0.583] |

1. Отыскание собственных значений методом QR-разложения.

Входные данные:

|  |
| --- |
| -9 9 -7  -7 5 -1  -4 3 4 |

Вывод:

|  |
| --- |
| Задайте точность: 0.01  Матрица A:  [-9.0, 9.0, -7.0]  [-7.0, 5.0, -1.0]  [-4.0, 3.0, 4.0]  Матрица A(19):  [6.115, 2.013, -2.981]  [-0.001, -0.753, 15.713]  [-0.002, -0.659, -5.362]  Собственные значения:  [6.115, '-3.0575 + 2.245917796803792i', '-3.0575 - 2.245917796803792i'] |

1. **Часть текста программы**

Основные методы, выполняющие поставленные задачи:

|  |
| --- |
| def LU(A, L, U, n):  for i in range(n):  j = i  while j < n:  U[i][j] = A[i][j]  L[j][i] = float(A[j][i]) / float(A[i][i])  j += 1  k = i  while k < n - 1:  j = i  koef = float(A[k + 1][i]) / float(A[i][i])  while j < n:  A[k + 1][j] = float(A[k + 1][j]) - koef \* float(A[i][j])  j += 1  #print\_matrix(A) пошаговая проверка  k += 1  def solveLU(L, U, n, b):  Y = []  Y.append(float(b[0]))  X = []  for i in range(1, n):  sum = 0  for j in range(i):  sum += float(Y[j]) \* float(L[i][j])  Y.append(float(b[i]) - sum)  #print(Y)  #Зануляем вектор  for i in range(n):  X.append(float(0))  i = n - 1  while i > -1:  sum = 0  for j in range(i + 1, n):  sum += U[i][j] \* X[j]  X[i] = float(1 / U[i][i] \* (Y[i] - sum))  i -= 1  return X  def progonka(A, b, n):  P, Q, X = [], [], []  nul\_vector(P, n)  nul\_vector(Q, n)  nul\_vector(X, n)  P[0], Q[0] = -A[0][1] / A[0][0], b[0] / A[0][0]  for i in range(1, n - 1):  P[i] = -A[i][i + 1] / (A[i][i] + A[i][i - 1] \* P[i - 1])  Q[i] = (b[i] - A[i][i - 1] \* Q[i - 1]) / (A[i][i] + A[i][i - 1] \* P[i - 1])  Q[n - 1] = (b[n - 1] - A[n - 1][n - 2] \* Q[n - 2]) / (A[n - 1][n - 1] + A[n - 1][n - 2] \* P[n - 2])  X[n - 1] = Q[n - 1]  i = n - 2  '''  print("Вектор b:\n", b, "\n")  print("Вектор P:\n", P, "\n")  print("Вектор Q:\n", Q, "\n")  '''  while i > -1:  X[i] = P[i] \* X[i + 1] + Q[i]  i -= 1  return X  def zeydel(A, b, n, e):  x0, x1, iter = [], [], 0  for i in b:  x1.append(i)  bibl2.nul\_vector(x0, n)  C = get\_C(A)  epsil = epsilon(x0, x1, C, A, n)  while epsil > e and iter < max\_iter:  for i in range(n):  x0[i] = x1[i]  for i in range(n):  sum = 0  for j in range(n):  sum += A[i][j] \* x1[j]  x1[i] = b[i] + sum  epsil = epsilon(x0, x1, C, A, n)  iter += 1  return [x1, iter]  def common\_iterations(A, b, n, e):  iter, x1, x0 = 0, b, []  bibl2.nul\_vector(x0, n)  epsil = epsilon(x0, x1, A, A, n)  while epsil > e and iter < max\_iter:  x0 = x1  x1 = vector\_plus(b, multiplication(A, x0, n))  epsil = epsilon(x0, x1, A, A, n)  iter += 1  return [x1, iter]  def matrix\_U(A, max\_i, max\_j, fi, n):  U = []  bibl.nul\_matrix(U, n)  for i in range(n):  U[i][i] = 1  U[max\_i][max\_i] = round(math.cos(fi), 2)  U[max\_j][max\_j] = round(math.cos(fi), 2)  U[max\_i][max\_j] = round(-math.sin(fi), 2)  U[max\_j][max\_i] = round(math.sin(fi), 2)  return U  def get\_QR(R, n):  k = 0  Q = get\_E(n)  while k < n - 1:  v = get\_v(R, n, k)  H = get\_haus(v, n)  R = bibl4.matrix\_multiplication(H, R, n, n, n)  Q = bibl4.matrix\_multiplication(Q, H, n, n, n)  k += 1  return [Q, R] |

1. **Вывод**

В ходе выполнения этой лабораторной работы я получил опыт разработки комплексных, обширных программ на python, а также в реализации интерфейса взаимодействия с пользователем. Помимо этого, я реализовал большое количество методов обработки матриц и векторов, а также, решения СЛАУ.