Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский политехнический университет Петра Великого»

Институт прикладной математики и механики

Кафедра прикладной математики

Курсовой проект «Реализация численных методов»

Выполнил студент группы 23631/1 Цветков А.Д.

Преподаватель Павлова Л.В.

Санкт-Петербург, 2018

Оглавление

[Часть 1. Решение алгебраических и трансцендентных уравнений (вариант 20) 3](#_Toc533387314)

[Формулировка задачи и ее формализация 3](#_Toc533387315)

[Алгоритмы методов и условия их применимости 3](#_Toc533387316)

[Метод половинного деления. 3](#_Toc533387317)

[Метод Ньютона. 3](#_Toc533387318)

[Предварительный анализ задачи 4](#_Toc533387319)

[Проверка условий применимости метода 5](#_Toc533387320)

[Тестовый пример с детальными расчетами для задачи малой размерности 6](#_Toc533387321)

[Перечень контрольных тестов для иллюстрации метода средствами пакета MATLAB 7](#_Toc533387322)

[Модульная структура программы 7](#_Toc533387323)

[Анализ численного решения задач 8](#_Toc533387324)

[Краткие выводы 9](#_Toc533387325)

[Часть 2. Решение СЛАУ прямыми методами (LU-разложение) 10](#_Toc533387326)

[Формулировка задачи и ее формализация 10](#_Toc533387327)

[Алгоритм метода и условие его применимости 10](#_Toc533387328)

[Алгоритм LU-разложения 10](#_Toc533387330)

[Ly = b (прямая подстановка) 10](#_Toc533387331)

[Ux = y (обратная подстановка) 11](#_Toc533387332)

[Условия применимости 11](#_Toc533387333)

[Предварительный анализ задачи и условий применимости метода 11](#_Toc533387334)

[Хорошо обусловленная матрица 11](#_Toc533387335)

[Плохо обусловленная матрица 11](#_Toc533387336)

[Условия применимости 11](#_Toc533387337)

[Тестовый пример с детальными расчетами для задачи малой размерности 11](#_Toc533387338)

[Модульная структура программы 12](#_Toc533387339)

[Анализ численного решения задачи 13](#_Toc533387340)

[Для хорошо обусловленной матрицы 13](#_Toc533387341)

[Для плохо обусловленной матрицы 14](#_Toc533387342)

[Краткие выводы 15](#_Toc533387343)

[Часть 3. Решение СЛАУ итерационными методами (метод Зейделя) 16](#_Toc533387344)

[Формулировка задачи и ее формализация 16](#_Toc533387345)

[Алгоритм метода и условие его применимости 16](#_Toc533387346)

[Итерационный метод 16](#_Toc533387347)

[Условия сходимости 16](#_Toc533387348)

[Описание метода 16](#_Toc533387349)

[Алгоритм метода Зейделя 16](#_Toc533387350)

[Условия применимости 17](#_Toc533387351)

[Предварительный анализ задачи и условий применимости метода 17](#_Toc533387352)

[Тестовый пример с детальными расчетами для задачи малой размерности 17](#_Toc533387354)

[Модульная структура программы 18](#_Toc533387355)

[Численный анализ решения задачи 18](#_Toc533387356)

[Краткие выводы 19](#_Toc533387357)

[Часть 4. Решение алгебраической проблемы собственных значений (метод Якоби) 20](#_Toc533387358)

[Формулировка задачи и ее формализация 20](#_Toc533387359)

[Алгоритм метода и условие его применимости 20](#_Toc533387360)

[Алгоритм 20](#_Toc533387361)

[Условия применимости 21](#_Toc533387362)

[Предварительный анализ задачи и условий применимости метода 21](#_Toc533387363)

[Тестовый пример с детальными расчетами для задачи малой размерности 21](#_Toc533387364)

[Модульная структура программы 22](#_Toc533387365)

[Численный анализ решения задачи 22](#_Toc533387366)

[Краткие выводы 22](#_Toc533387367)

[Заключение по итогам выполнения курсового проекта 23](#_Toc533387368)

# Часть 1. Решение алгебраических и трансцендентных уравнений (вариант 20)

## Формулировка задачи и ее формализация

Необходимо решить задачу нахождения корней уравнения вида , где – алгебраическая или трансцендентная функции.   
Задача решается в 3 этапа:   
1) нахождение промежутков, в которых лежат корни (например, с помощью теоремы о верхней границе положительных корней полинома);  
2) отделение корней (нахождение отрезков, содержащих единственный корень)  
3) уточнение корней с некоторой задаваемой точностью ϵ.  
Методы применяются к тем отрезкам, на которых функция имеет ровно один корень.

## Алгоритмы методов и условия их применимости

Метод половинного деления.

Функция должна быть определена и непрерывна при всех x на отрезке [a, b], а также . Тогда, согласно уравнению Больцано-Коши, уравнение будет иметь хотя бы один корень.

Алгоритм:

1. Находим середину отрезка .
2. Если , то полагаем, что корень равен и заканчиваем вычисления, выходим из цикла. Если , то полагаем, иначе .
3. Условие выхода из цикла:. Тогда – искомый корень.

### Метод Ньютона.

Теорема о достаточных условиях сходимости метода Ньютона:

Если уравнение https://helpiks.org/helpiksorg/baza6/73360050953.files/image095.png имеет один корень https://helpiks.org/helpiksorg/baza6/73360050953.files/image173.png , функции https://helpiks.org/helpiksorg/baza6/73360050953.files/image175.png и https://helpiks.org/helpiksorg/baza6/73360050953.files/image177.png – знакопостоянны на этом отрезке, начальное приближение https://helpiks.org/helpiksorg/baza6/73360050953.files/image120.png выбрано из условия

Тогда последовательность https://helpiks.org/helpiksorg/baza6/73360050953.files/image182.png , построенная по методу Ньютона, сходится к точному решению https://helpiks.org/helpiksorg/baza6/73360050953.files/image097.png , когда https://helpiks.org/helpiksorg/baza6/73360050953.files/image185.png. При этом справедлива следующая оценка погрешности:

 ,

где https://helpiks.org/helpiksorg/baza6/73360050953.files/image189.png .   
Новое приближение вычисляется по формуле:

Алгоритм:

1. Задаем начальное приближение . Изначально , если начальное приближение не соответствует условию теоремы о достаточных условиях сходимости метода Ньютона, выбираем правый конец отрезка ().
2. Вычисляем новое приближение по формуле .
3. Апостериорная оценка для выхода из цикла: . В качестве корня берем и заканчиваем вычисления, в противном случае переходим к пункту 2.

## Предварительный анализ задачи

Исследуем функцию полинома . Область определения D(y)ϵ (-∞;∞), область значений E(y)ϵ[-385.85; +∞). Функция, как мы можем видеть по графику и области значений, непрерывна, не имеет точек разрыва. Она не имеет асимптот и является функцией общего вида. Первая производная равна , вторая производная равна , обе функции определены и непрерывны на всей области определения полинома. У полинома есть два корня, в окрестностях точек -4 и 10.   
*Для метода половинного деления* функция должна быть определена и непрерывна при всех x на отрезке [a, b], что выполняется на отрезках [0.8205, 437.6491] и [-21.8962, -0.8245] (данные границы мы нашли с помощью теоремы о верхней границы положительных корней полинома: см. рисунки 1,2 приложения).   
Также выполняется условие :   
Для положительного корня: Для отрицательного корня:   
*Для метода Ньютона* функция дважды дифференцируема. Чтобы выполнялись условия знакопостоянства и монотонности первой производной для положительного корня полинома, необходимо левую границу сдвинуть до 7.45 (узнали, построив график; условия для метода половинного деления все еще выполняются). Для отрицательного корня полинома нужно сдвинуть правую границу до -1.55, тогда производная не будет менять знак и равняться 0 при любых х, принадлежащих отрезкам.   
Для положительного корня: Для отрицательного корня:   
Все условия выполняются.

Исследуем функцию трансцендентного уравнения .   
Область определения D(y)ϵ (0;+∞), область значений E(y)ϵ(-∞; +∞). Функция, как мы можем видеть по графику и области значений, непрерывна, не имеет точек разрыва. Все аргументы положительны, асимптотой является прямая . Первая производная равна вторая производная равна , обе эти функции определены и непрерывны на всей области определения полинома. У трансцендентного уравнения один корень в окрестности точки 2.   
*Для метода половинного деления*: функция определена и непрерывна при всех x на отрезке [1, 2] (границы нашли с помощью графика).   
Выполняется условие : .  
*Для метода Ньютона* функция также дважды дифференцируема. Знакопостоянство первой производной и неравенство нулю выполняется при всех х, принадлежащих отрезку.   
Проверим .  
Все условия выполняются.

## Проверка условий применимости метода

Произведем проверку условий метода половинного деления:

1. Нарушим условие непрерывности. Рассмотрим функцию на промежутке [-2; 2]. Метод дал результат -0.0000000018626. Полученное решение соответствует точке разрыва функции.
2. Нарушим условие и наличие корней. Для этого возьмем функцию для на промежутке [-2; 2]. Данная функция определена и непрерывна при всех х, принадлежащих отрезку [-2, 2], а также не имеет корней не только на этом отрезке, но и на всей области определения. В этом случае метод не даст никакого ответа, поскольку программа зациклится и будет выполнять подсчеты с изначальным .
3. Нарушим условие единственности корня на отрезке.   
   Рассмотрим функцию на отрезке [-2, 2] (условие выполняется). Метод даст результат 1.2470, что является одним из трех корней уравнения на данном отрезке.

Произведем проверку условий метода Ньютона:

1. Нарушим условие существования производных. Рассмотрим на промежутке [-1; 1], программа зациклилась и не дала результата.
2. Нарушим знакопостоянство первой производной. Рассмотрим на промежутке [-1; 0.5]. Метод даст правильный ответ -0.2587.
3. Нарушим знакопостоянство второй производной. Рассмотрим на промежутке [-0.6; 0.5]. Метод даст правильный ответ -0.2587.
4. Нарушим единственность корня.   
   Возьмем функцию на отрезке (условие не выполняется). Программа даст ответ 0, то есть мы нашли один правильный корень из двух существующих.   
   Рассмотрим на отрезке (условие выполняется). Метод даст результат 0, что является одним из двух корней уравнения на данном отрезке.

## Тестовый пример с детальными расчетами для задачи малой размерности

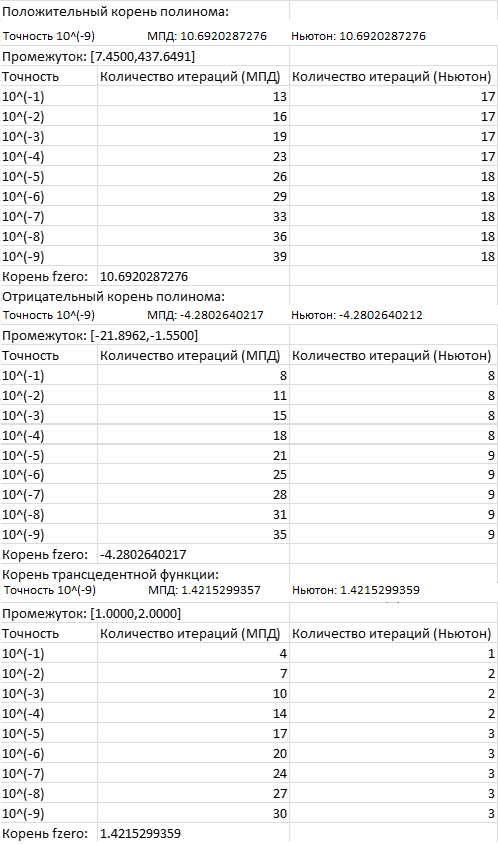
Рассмотрим, как работает метод половинного деления на примере трансцендентной функции на отрезке [1, 2].

1. . , полагаем, что
2. , полагаем, что .
3. полагаем, что
4. , полагаем, что .
5. Продолжаем алгоритм пока не дойдем до корня, удовлетворяющего заданной точности.

Рассмотрим, как работает метод Ньютона на примере трансцендентной функции на отрезке [1, 2].

1. , так как ;Следующее .
2. Второе значение: .
3. Третье значение: .
4. Продолжаем алгоритм пока не дойдем до корня, удовлетворяющего заданной точности.

## Перечень контрольных тестов для иллюстрации метода средствами пакета MATLAB



Таблица

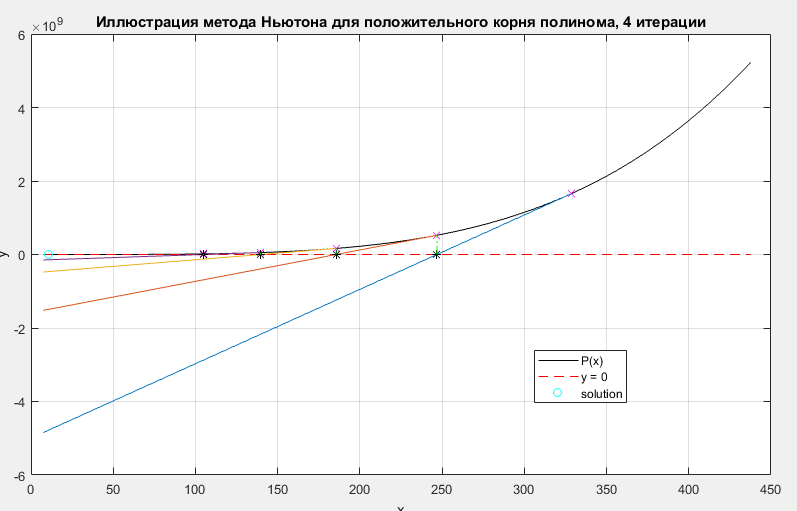
Расширенные данные исследования решения см. в Приложении в таблицах 1, 2.

## Модульная структура программы

Программа состоит из 8 модулей:

1. «poly.m» – алгебраическое уравнение (полином);
2. «func.m» – трансцендентное уравнение (функция);
3. «Bisection\_method.m» – функция, реализующая метод половинного деления. Принимает на вход границы поиска корня, уравнение, строку для записи в текстовый файл. Записывает в файл lab.txt точность, количество итераций при заданной точности, разность корней (найденного с помощью метода половинного деления и fzero), найденный корень, значение в функции в точке, относительную и абсолютные погрешности при вычислении. Также в конце подсчетов записывает корень, найденный с помощью стандартной функции fzero.
4. «Newthon\_method.m» - функция, реализующая метод Ньютона. Принимает на вход границы поиска корня, уравнение, первую и вторую его производные, строку для записи в текстовый файл. Записывает в файл lab.txt точность, количество итераций при заданной точности, разность корней (найденного с помощью метода половинного деления и fzero), найденный корень, значение в функции в точке, относительную и абсолютные погрешности при вычислении. Также в конце подсчетов записывает корень, найденный с помощью стандартной функции fzero.
5. «Check.m» - проверяет промежуток и функцию на соответствие условиям применения метода Ньютона. Принимает на вход уравнение, первую и вторую производные, границы поиска корня, строковый файл для подписи графика. Проверяет условие , а также знакопостоянство первой и второй производных – рисует их графики. По графикам можем подвинуть границы поиска корня, если условие знакопостоянства производных не выполняется на всем изначальном отрезке.
6. «Draw\_newthon.m» – рисует первые 4 итерации метода Ньютона с подниманием корня и проведением касательных. Отображает на рисунке корень. Принимает на вход функцию, первую и вторую производные, начало и конец промежутка, где содержится корень, строку для подписи графика.
7. «Function.m» – модуль, работающий с трансцендентной функцией. Рисует ее график. позволяя найти примерный промежуток для поиска корня, вызывает функцию поиска корня методом половинного деления и Ньютона, проверяет, соответствует ли промежуток условиям для метода Ньютона.
8. «Polynom.m»– модуль, работающий с алгебраической функцией. Рисует ее график. позволяя найти примерный промежуток для поиска корня, вызывает функцию поиска корня методом половинного деления и Ньютона, проверяет, соответствует ли промежуток условиям для метода Ньютона (для положительного и отрицательного корней). Вызывает функцию отрисовки метода Ньютона для положительного корня полинома.

## Анализ численного решения задач

Была поставлена задача найти корни трансцендентного и алгебраического уравнений вида двумя разными методами. Первым был метод половинного деления, который достаточно легко реализуется и является наиболее универсальным среди итерационных методов уточнения корней. Его применение гарантирует получение решения для любой непрерывной функции , если найден интервал, на котором она изменяет знак. В том случае, когда корни не отделены, будет найден один из корней уравнения. Метод достаточно медленный, корень с точностью достигается с помощью 30 итераций для трансцендентной функции, 39 итераций для положительного корня полинома и 35 для отрицательного. Метод Ньютона обладает высокой скоростью сходимости, с его помощью удалось найти корень с точностью до всего за 3 итерации для трансцендентной функции. Поскольку положительный корень полинома находится близко к нулю, а функция возрастает очень медленно (см. рисунок 1), найти корень с заданной точностью получилось только за 18 итераций. Отрицательный корень полинома был найден за 9 итераций с максимальной точностью.

Рисунок

По сравнению с методом половинного деления недостатком метода Ньютона является необходимость вычисления на каждой итерации не только f(x), но и ее производной. Таким образом, мы изучили алгоритмы этих методов для нахождения корней.  
Подробные иллюстрации решения см. в Приложении.

## Краткие выводы

В данной работе мы изучили два метода для нахождения корней уравнения, посмотрели на скорость их сходимости и реализовали графическую интерпретацию метода Ньютона.

# Часть 2. Решение СЛАУ прямыми методами (LU-разложение)

## Формулировка задачи и ее формализация

Найти решение СЛАУ методом LU-разложения, проверить вычислительную ошибку (найти вектор невязки) для матриц с разными числами обусловленности. Для исследования выбираем квадратную матрицу при

Определение. Число обусловленности матрицы показывает насколько матрица близка к матрице неполного ранга (для квадратных матриц - к вырожденности). Число обусловленности квадратной матрицы A определяется, как cond(A) = .

## Алгоритм метода и условие его применимости

А = LU, где

Так, задача сводится к тому, чтобы решить уравнение вида LUx=b. Эта система может быть решена в два шага. На первом шаге решается система Ly =b (непосредственно прямой подстановкой). На втором шаге решается система Ux=y (непосредственно обратной подстановкой).

### Алгоритм LU-разложения:

И так далее. Получим формулы:

### Ly = b (прямая подстановка):

### Ux = y (обратная подстановка):

;

### Условия применимости:

Все главные миноры матрицы А отличны от нуля.

Для нахождения главных миноров вычеркиваем (n – j) столбцы и (n – j) строки из матрицы А:

## Предварительный анализ задачи и условий применимости метода

Поскольку задачей стоит исследовать матрицы с разными числами обусловленности, рассмотрим работу метода на примерах двух матриц – хорошо и плохо обусловленной.

Подготовим матрицы с большим и маленьким числом обусловленностей по следующим алгоритмам.

### Хорошо обусловленная матрица:

*;*

где:

, где W – вектор размера , состоящий из случайных чисел;

*–* треугольная или диагональная матрица размера , состоящая из случайных чисел.

### Плохо обусловленная матрица:

В качестве плохо обусловленной матрицы выбираем матрицу Гильберта:

*;*

### Условия применимости:

## Тестовый пример с детальными расчетами для задачи малой размерности

1)

2)

3)

4)

5) ;

;

,

## Модульная структура программы

Программа состоит из 10 модулей:

1. “bad\_generate.m“ – создает плохо обусловленную матрицу Гильберта, считает число обусловленности, записывает данные в файл.

2. “good\_random.m” - создает хорошо обусловленную матрицу, считает число обусловленности, записывает данные в файл.

3. void readB (double \*\*b, char \*string1, char \*string2) – считывает вектор b в массив b из файла с названием string1, записывает вектор b в файл с названием string2.

4. void readA (double \*\*\*matr, char \*string) – считывает матрицу в двумерный массив matr из файла с названием string.

5. void LU (double \*\*\*L, double \*\*\*U, double \*\*\*A) – раскладывает матрицу А на матрицы L и U, такие, что LU=A.

6. void solve (double \*\*\*L, double \*\*\*U, double \*\*b, double \*\*\*A, char \*string) – решает СЛАУ, записывает результаты в файл с именем string.

7. void nevyazka (double \*\*\*A, double \*\*x, double \*\*b) – считает вектор невязки, выводит на экран.

8. void errors (double \*\*\*A, double \*\*b, double \*\*\*L, double \*\*\*U, char \*string) – вносит возмущения в матрицу А, считает новое разложение, записывает результаты в файл с названием string.

9. “workspace.m“ – вызывает функцию вычисления коэффициентов для каждой из матриц.

10. “errors.m” – считывает данные из матрицы, считает коэффициенты для возмущений в b и в A.

## Анализ численного решения задачи

Известно, что справедливы следующие неравенства:

; . (\*)

Решим СЛАУ Ax = b методом LU-разложения для двух матриц: хорошо и плохо обусловленной. Найдем векторы невязки, а также реальные коэффициенты K1 и K2, связывающие правые и левые части неравенств (\*).

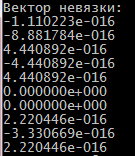
Коэффициент К1 получается при внесении возмущений в столбец b и находится из формулы:

Коэффициент К2 получается при внесении возмущений в матрицу А и находится из формулы: =𝑘2∗ .

### Для хорошо обусловленной матрицы:

Cond(A) = 11,9671;

Внесение возмущений в b.



Рисунок

.

Внесение возмущений в A.

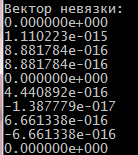
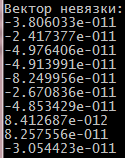


Рисунок .

### Для плохо обусловленной матрицы:

Cond(A) = 2.6791e+13;

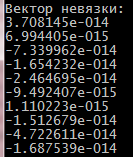
Внесение возмущений в b.



Рисунок

.

Внесение возмущений в A.



Рисунок

.

Пример подробных расчетов для задачи малой размерности см. в Приложении.

## Краткие выводы

В ходе выполнения работы были вычислены коэффициенты при внесении изменений в вектор b (К1) и в матрицу A (К2). K1 изменений (порядка 1%) в вектор b не зависит от числа обусловленности матрицы – он всегда получался порядка единицы. K2 прямо пропорционален числу обусловленности матрицы. Если число обусловленности матрицы маленькое, то при внесении возмущений в матрицу А коэффициент получается небольшой. Однако, чем больше число обусловленности матрицы, тем больше возмущения на выходе при тех же внесенных изменениях (не больше 1%). Коэффициенты всегда не превосходят число обусловленности матрицы. K1 всегда меньше K2.

Метод LU-разложения удобен в реализации, а также является устойчивым методом, поскольку при маленьком числе обусловленностей матрицы малые возмущения на входе дают малые возмущения на выходе.

# Часть 3. Решение СЛАУ итерационными методами (метод Зейделя)

## Формулировка задачи и ее формализация

Найти корни СЛАУ вида Ax=b методом Зейделя. Исследовать точность решения, когда определитель матрицы близок к 0.

## Алгоритм метода и условие его применимости

### Итерационный метод:

Суть такого метода заключается в нахождении по приближённому значению величины следующего приближения (являющегося более точным при условии сходимости метода). Итерационные методы позволяют получить решение с наперед заданной точностью, если доказана сходимость метода. Строго точного решения итерационные методы не дают, поскольку оно достигается как предел последовательности векторов.

### Условия сходимости:

Для того, чтобы применить метод Зейделя, нужно, чтобы система была представима в виде, удобном для итераций: x = αx + β.

Достаточное условие сходимости: <1 ; достаточное и необходимое условие сходимости: | (α)| < 1.

Метод Зейделя всегда сходится для систем, в которых матрица А симметричная и положительно определенная. Если матрица не является симметричной, всегда можно свести систему к подходящей, домножив обе части равенства слева на , получив систему \*А\*х = \*b, которая является нормальной. При таком преобразовании квадратично возрастает число обусловленностей, уменьшается точность решения, падает скорость сходимости (при числах обусловленности, порядок которых больше , метод сходится на бесконечности) и эффективность метода.

### Описание метода:

Метод является модификацией метода простых итераций. При нахождении i-й компоненты (k+1)-го приближения сразу используются уже найденные компоненты (к +1) -го приближения с меньшими номерами 1,2,…,i−1.  
Записывая метод в матричной форме:

= L\* + U\* + β, L – нижняя треугольная матрица, U – верхняя треугольная матрица, являющиеся LU-разложением исходной матрицы А.

Считается, что решение получено с заданной точностью ɛ, если: || - ||< ε.

### Алгоритм метода Зейделя для нормальной системы:

1. Преобразовать \*А\*х = \*b, если система изначально не является нормальной.
2. Преобразовать систему к виду x = αx + β: = , = .
3. Задать начальное приближение решение , малое положительное число , положить k=0.
4. Вычислить вектор по формуле:
5. Если выполнено условие завершения || - || < ε, процесс прекратить, иначе положить k=k+1 и перейти к пункту (4).

### Условия применимости:

.  
C помощью пакета Matlab задаем симметричную и положительно определенную матрицу.

## Предварительный анализ задачи и условий применимости метода

Для исследования работы метода выберем матрицы с большими и маленькими числами обусловленности, а также с разными определителями со значениями: (проверим с помощью пакета Matlab).

## Тестовый пример с детальными расчетами для задачи малой размерности

А = ; b =

1. Выразим столбец х:
2. α = , = β = ;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| k |  |  |  | |||| |
| 0 | 2 | 3 | -5 | - |
| 1 | 1,72 | 2,6984 | -5,036768 | 0,3016 |
| 2 | 1,737361 | 2,696041 | -5,036547 | 0,017361 |
| 3 | 1,737507 | 2,696047 | -5,036546 | 0,000146 |
| 4 | 1,737506 | 2,696048 | -5,036546 | 3,730986e-007 |

Таблица

С точностью корни уравнения: х = .

## Модульная структура программы

Программа состоит из 8 модулей:

1. “solve.m” – создает положительно определенную матрицу A и вектор b для дальнейшего исследования.
2. “edit.m” – изменяет матрицу, если она не является симметричной, домножая обе части уравнения Ax=b слева на транспонированную матрицу A.
3. “check.m” – проверяет матрицу на соответствие условиям сходимости метода Зейделя.
4. “workspace.m” – вызывает функции solve, edit и check для большого числа разных матриц.
5. void alphabeta (double \*\*\*matrnew, double \*\*bnew, double \*\*\*alpha, double \*\*beta) – приводит систему к виду x = αx + β. Принимает на вход исходную матрицу A, вектор b, а также пустые матрицы alpha и beta, которые заполняет в процессе работы.
6. void solve (double \*\*beta, double \*\*\*alpha, double \*\*\*matrnew, double \*\*bnew) – решает систему уравнений, выводит на экран ответ. Получает на вход исходную матрицу, столбец b, а так же beta и alpha разложение.
7. void nevyazka (double \*\*\*A, double \*\*x, double \*\*b) – ищет норму вектора невязки для решений СЛАУ: Ax-b.
8. double normir (double \*\*x) – ищет наибольший элемент в столбце х и возвращает его модуль, используется для условия цикла для проверки точности решения.

## Численный анализ решения задачи

Для исследования были выбраны матрицы размерности , точность .

|  |  |  |  |
| --- | --- | --- | --- |
| Определитель | Число обусловленности | Кол-во итераций | Норма вектора невязки |
| 4.000000e-020 | 40,033223 | 17 | 1.711403e-009 |
| 3.628800e-010 | 14,506673 | 14 | 2.000595e-009 |
| 0,000040 | 16,244374 | 26 | 3.822785е-008 |
| 0,004890 | 54,628695 | 24 | 9.160417e-008 |
| 37,209563 | 2,411739 | 7 | 3.705432e-009 |
| 6,000000e-020 | 1,077590+007 | 1066217 | 2.205160e-004 |
| 3,628800e-010 | 4,896994e+003 | 2377 | 1.119103e-006 |
| 3,700000e-005 | 2,637221e+004 | 12614 | 1.825174e-005 |
| 1,397000e-003 | 4,444521e+004 | 17721 | 2.621260e-005 |
| 1822,255852 | 4,884314e+003 | 2672 | 4.973386e-006 |

Таблица

## Краткие выводы

Метод Зейделя является модификацией метода простых итераций, которая обычно существенно увеличивает скорость сходимости. Условия применимости метода позволяют работать с симметричными матрицами, для обычных матриц приведение системы к нормальному виду увеличивает число обусловленности.  
Метод хорошо сходится для матриц, у которых маленькое число обусловленности. Для матриц с определителем, близким к нулю, метод работает так же, как и для матриц, определитель которых не близок к нулю, что позволяет судить о его высокой эффективности. При увеличении числа обусловленности количество итераций сильно возрастает, вектор невязки увеличивается, метод сходится на бесконечности, что делает его применение неоправданным.

# Часть 4. Решение алгебраической проблемы собственных значений (метод Якоби)

## Формулировка задачи и ее формализация

Для заданной матрицы А требуется найти собственные числа λ и соответствующие им собственные векторы X методом Якоби.

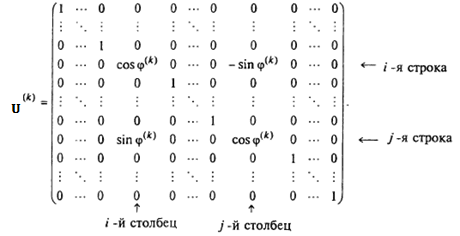
Определение: собственным числом (с.ч.) матрицы А называется такое число λ , при котором выполняется равенство для ненулевого вектора X; вектор X называется собственным вектором матрицы А, соответствующим с.ч. λ .

## Алгоритм метода и условие его применимости

Метод Якоби – итерационный алгоритм для вычисления собственных значений и собственных векторов вещественной симметричной матрицы .Он решает полную проблему собственных значений и собственных векторов таких матриц. Метод основан на отыскании с помощью итерационных процедур матрицы в преобразовании подобия , а поскольку для симметрических матриц матрица вращения U является ортогональной , то , где - диагональная матрица с собственными значениями на главной диагонали. Идея алгоритма заключается в выборе матрицы вращения U на каждой итерации таким образом, чтобы максимально свести к нулю значения внедиагональных элементов матрицы А.

### Алгоритм:

1. || = - выбираем максимальный по модулю элемент в наддиагональной матрице
2. , из условия
3. : – построение матрицы вращения

**

1. =
2. Выполняем пункты 1-4, пока > ε, (хотя бы один внедиагональный элемент)
3. , , … ,
4. U = … – матрица собственных векторов

= (,

= (,

= (.

### Условия применимости:

А = , строим подходящую матрицу с помощью пакета Matab.

## Предварительный анализ задачи и условий применимости метода

Для исследования работы метода выберем матрицы с большими и маленькими числами обусловленности, а также с плохой и хорошей отделимостью собственных чисел (проверим с помощью пакета Matlab). Матрицы выберем вещественные.

## Тестовый пример с детальными расчетами для задачи малой размерности

А =

1. = -0, 585278
2. =
3. =
4. = 0.482177
5. =
6. =
7. Повторяем процесс, пока точность не будет соответствовать заданной. В нашем случае требуется 5 итераций для вычисления собственных чисел и собственных векторов при точности .
8. Получили следующие результаты:

## Модульная структура программы

Программа состоит из 8 модулей:

1. “generate.m” – составляет матрицы для исследования по заданным собственным числам.
2. void vectorfill(double \*\*\*vector) – функция, инициализирующая матрицу собственных векторов, делает входную матрицу vector единичной.
3. void solve (double \*\*\*matr, double \*\*\*g, double \*\*\*gt, double \*\*\*vector) – функция решения задачи нахождения собственных значений. Получает на вход изначальную матрицу, матрицу собственных векторов, а также массивы для обычной и транспонированной матриц поворота.
4. void turn (double \*\*\*g, double \*\*\*gt, double \*\*\*matr) – вычисляет и заполняет матрицу поворота.
5. void multiply (double \*\*\*matr, double \*\*\*matr1, double \*\*\*res) – вспомогательная функция умножения матриц. Умножает matr на matr1, записывает результат в массив res.
6. double find (double \*\*\*matr) – находит в матрице matr наибольший внедиагональный элемент и возвращает его. Используется для проверки условия выхода из цикла в функции solve решения.
7. void trans (double \*\*\*matr1, double \*\*\*matr2) – транспонирует матрицу matr2, записывает результат в матрицу matr1.
8. void nevyazka(double \*\*\*matr, double \*\*\*vector) – ищет норму невязки для всех собственных векторов матрицы:

## Численный анализ решения задачи

Для исследования были выбраны матрицы размерности , точность .

|  |  |  |  |
| --- | --- | --- | --- |
| Число обусловленностей | Отделимость собственных чисел | max | Кол-во итераций |
| 27.335209 | Хорошая | 2.089021e-005 | 38 |
| 1.001312 | Плохая | 5.964048e-005 | 7 |
| 48489.934404 | Хорошая | 3.116041e-005 | 114 |
| 44936.092636 | Плохая | 7.990759e-005 | 35 |

Таблица

## Краткие выводы

Метод Якоби эффективен как для матриц с маленьким, так и с большим числом обусловленности. Количество итераций возрастает для матриц с хорошей отделимостью собственных чисел по сравнению с матрицами с плохой отделимостью.   
Метод Якоби нахождения собственных чисел и соответствующих им собственных векторов решает полную проблемы собственных значений и собственных векторов для задачи малой размерности (n=10).

# Заключение по итогам выполнения курсового проекта

В результате выполнения курсового проекта были рассмотрены методы решения алгебраических и трансцендентных уравнений, СЛАУ прямыми итерационными методами, алгебраической системы собственных значений.

Для решения алгебраических и трансцендентных уравнений были рассмотрены метод половинного деления и метод Ньютона. Первый является наиболее универсальным и легко реализуемым, а второй более быстрым, но в то же время более трудным в реализации.

В ходе выполнения работы были изучены два метода решения СЛАУ – прямые и итерационные. Прямые методы решения СЛАУ в общем случае универсальны и используются для широкого класса систем. В предположении, что вычисления ведутся без округлений, прямые методы позволяют получить точные значения неизвестных. Однако в связи с особенностями работы системы компьютера вычисления округляются (при решении систем больших порядков возникают большие погрешности), а также при вычислениях требуется хранить большое количество данных, что указывает на неэффективное использование памяти. Так, прямые методы подходят для решения СЛАУ для систем малой размерности. В отличие от прямых методов, итерационные хорошо реализуются с помощью компьютера, поскольку на каждой итерации происходит уточнение результатов, пока они не будут соответствовать заданной допустимой погрешности. Так, итерационные методы позволяют работать с системами большой размерности. Главным недостатком итерационных методов является необходимость проверки большого количества условий сходимости метода.

Решение алгебраической проблемы собственных значений методом Якоби показало хорошие результаты для различных матриц, однако наилучшая сходимость достигается при маленьком числе обусловленностей матрицы и плохой отделимости собственных чисел.