|  |
| --- |
| Санкт-Петербургский Политехнический университет Петра Великого |
| Отчет по курсу «Численные методы» |
| Выполнила Гареева Милена, студент группы 23631/1 |

# Часть 1. Решение алгебраических и трансцендентных уравнений

## Задача и формализация

В данной работе необходимо решить задачу нахождения корней уравнения вида , где – алгебраическая и трансцендентная функции. Данная задача решается в 2 этапа: 1) отделение корней, т.е. нахождение промежутков, в которых лежат эти корни (например, с помощью теоремы о верхней границе положительных корней полинома); 2) уточнение корней с некоторой задаваемой точностью ϵ.

## Алгоритм методов и условия их применимости

Первым методом является метод половинного деления. Функция должна быть определена и непрерывна при всех x на отрезке [a, b], а также она должна менять знак, т.е. . Тогда согласно с теоремой Больцано-Коши уравнение будет иметь хотя бы один корень.

Алгоритм (как он реализован в приложении MATLAB, показано в приложении 1):

1. Находим середину отрезка . Вычисляем .
2. Если , то полагаем, что корень равен и заканчиваем вычисления, выходим из цикла. Если , то полагаем, иначе .
3. Если модуль разности между границами a и b меньше задаваемого ϵ (), то в качестве корня берем последний подсчитанный нами и заканчиваем вычисления, в противном случае переходим к пункту 1.

Вторым методом является метод Ньютона. Достаточные условия метода Ньютона:

1. Функция должна быть определена и дважды дифференцируема на отрезке [a, b].
2. Первая производная не равна 0 для любых x, принадлежащих отрезку отрезка.
3. Первая и вторая производные не меняют знак на отрезке [a, b] при любом x из него.

Также существует теорема о достаточных условиях сходимости метода Ньютона:

Если выполняются следующие условия:

1. Функция должна быть определена и дважды дифференцируема на отрезке [a, b].
2. Отрезку [a, b] принадлежит только один простой корень x, так что .
3. Первая и вторая производные не меняют знак на отрезке [a, b], первая производная отлична от 0 при любом х отрезка.
4. Начальное приближение удовлетворяет условию (то есть знаки этих функций совпадают).

Тогда с помощью метода Ньютона можно вычислить корень уравнения с любой точностью.

Алгоритм (как он выглядит в приложении MATLAB показано в приложении 2):

1. Задаем начальное приближение x0, в данном случае по формуле .
2. Вычисляем новое приближение по формуле .
3. Если модуль разности между и меньше апостериорной оценки, которая рассчитывается по формуле ass = , где m-минимальное значение первой производной на отрезке, на котором ищем корень, а M – максимальное значение второй производной на отрезке. Тогда в качестве корня берем последний подсчитанный нами и заканчиваем вычисления, в противном случае переходим к пункту 1.

## Предварительный анализ задачи

Исследуем функцию полинома . Ее график представлен на рисунке в приложении 3. Область определения D(y)ϵ (-∞;∞), область значений E(y)ϵ[-85.2; +∞). 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 эти функции определены и непрерывны на всей области определения полинома. У полинома есть два корня, в окрестностях точек -1 и 3. Для метода половинного деления функция должна быть определена и непрерывна при всех x на отрезке [a, b], что выполняется на отрезках [0.5, 9.3] и [-3.3, -0.3] (данные границы мы нашли с помощью теоремы о верхней границы положительных корней полинома). Также выполняется условие : . Для метода Ньютона наша функция дважды дифференцируема. Чтобы выполнялись условия с первой производной, необходимо левую границу сдвинуть до 2.5(условия для половинного деления также выполняются), тогда производная не будет менять знак и равняться 0 при любых х, принадлежащих отрезкам. Проверим 3217.2,Все условия выполняются.

Исследуем функцию трансцендентного уравнения Ее график представлен на рисунке в приложении 4. Область определения D(y)ϵ (-∞;∞), область значений E(y)ϵ[-1; +∞). Функция, как мы можем видеть по графику и области значений, непрерывна, не имеет точек разрыва. Она не имеет асимптот и является четной функцией. Первая производная равна вторая производная равна , обе эти функции определены и непрерывны на всей области определения полинома. У трансцендентного уравнения есть два корня, в окрестностях точек -0.5 и 0.5.Для метода половинного деления: функция определена и непрерывна при всех x на отрезках [-1, -0.1] и [0.1, 1] (данные границы мы нашли с помощью графика). Также выполняется условие : ; . Для метода Ньютона наша функция также дважды дифференцируема. Знакопостоянство первой производной и неравенство нулю выполняется при всех х, принадлежащих отрезкам. Проверим 0.3,. Все условия выполняются.

## Проверка условий применимости метода

Произведем проверку условий метода половинного деления:

1. Нарушим условие непрерывности. Рассмотрим функцию на промежутке [-1; 1]. Метод дал результат -0.000008. Полученное решение соответствует точке разрыва функции.
2. Нарушим условие и наличие корней. Для этого возьмем функцию для на промежутке [0; 3]. Данная функция определена и непрерывна при всех х, принадлежащих отрезку [0, 3], а также не имеет корней не только на этом отрезке, но и на всей области определения. В этом случае метод даст ответ 2.999994, то есть происходит стремление к правой границе отрезка.
3. Нарушим единственность корня на отрезке. Рассмотрим функцию . Возьмем отрезок [-3, 4] (условие не выполняется). Результатом работы программы является -1.791287, то есть данный метод нашел лишь один правильный корень уравнения. Рассмотрим на отрезке [0, 9] (условие выполняется). Метод даст результат 1.570796, что является одним из трех корней уравнения на данном отрезке.

Произведем проверку условий метода Ньютона:

1. Нарушим условие и наличие корней. Рассмотрим на промежутке [0; 3]. В этом случае ответ будет равняться -11507.56173 с количеством итераций i = 2853304, хотя у функции и нет корней.
2. Нарушим условие существования производных. Рассмотрим на промежутке [-1; 1], программа зациклилась и не дала результата.
3. Нарушим знакопостоянство первой производной. Рассмотрим на промежутке [-0.2; 0.8]. Метод даст правильный ответ 0.000005.
4. Нарушим знакопостоянство второй производной. Рассмотрим функцию на отрезке [0, 5.5]. В результате получили 3.141592, абсолютно правильный ответ.
5. Нарушим единственность корня. Возьмем функцию на отрезке [-3, 4] (условие не выполняется). Программа даст ответ -1.791287, то есть мы нашли один правильный корень из двух существующих. Рассмотрим на отрезке [0, 9] (условие выполняется). Метод даст результат 4.712389, что является одним из трех корней уравнения на данном отрезке.

## Тестовый пример с детальными расчетами

Рассмотрим, как работает метод половинного деления на примере трансцендентной функции на отрезке [0, 2].

1. Считаем первое значение . , следовательно, полагаем, что
2. Считаем второе значение , следовательно, полагаем, что .
3. Считаем третье значение следовательно, полагаем, что
4. Считаем четвертое значение ,следовательно, полагаем, что .
5. Продолжаем алгоритм пока не дойдем до корня, удовлетворяющего заданной точности.

Рассмотрим, как работает метод Ньютона на примере трансцендентной функции на отрезке [0.1, 1].

1. За первое значение берем 0.4589, следовательно, следующее .
2. Считаем второе значение x: следовательно, следующее .
3. Считаем третье значение x: .
4. Продолжаем алгоритм пока не дойдем до корня, удовлетворяющего заданной точности.

## Перечень контрольных тестов для иллюстрации метода средствами пакета MATLAB

1. Используем метод половинного деления для функции полинома . Ищем корень на отрезке [0.5, 9.3], начиная с ϵ = 10^(-5). Также учитывается корень, найденный встроенный функцией fzero.Получаем следующее:
2. При погрешности 1.000000e-005 корень уравнения х = 3.002099, количество итераций i = 19. График зависимости погрешности решения от номера итерации для данного количества итераци показан в приложении 5.
3. При погрешности 1.000000e-007 корень уравнения х = 3.002106, количество итераций i = 26.
4. При погрешности 1.000000e-009 корень уравнения х = 3.002106, количество итераций i = 33.
5. Результат встроенной функции x = 3.002106 при любой погрешности.
6. Также используем метод половинного деления для полинома, но на отрезке [-3.3, -0.3]:
7. При погрешности 1.000000e-005 корень уравнения х = -0.738326, количество итераций i = 18.
8. При погрешности 1.000000e-007 корень уравнения х = -0.738327, количество итераций i = 24.
9. При погрешности 1.000000e-009 корень уравнения х = -0.738327, количество итераций i = 31.
10. Результат встроенной функции x = -0.738327 при любой погрешности.
11. Используем метод половинного деления для функции трансцендентного уравнения Ищем корень на отрезке [0.1, 1], начиная с ϵ = 10^(-5). Также учитывается корень, найденный встроенный функцией fzero.Получаем следующее:
12. При погрешности 1.000000e-005 корень уравнения х = 0.438427, количество итераций i = 16.
13. При погрешности 1.000000e-007 корень уравнения х = 0.438431, количество итераций i = 23.
14. При погрешности 1.000000e-009 корень уравнения х = 0.438431, количество итераций i = 29.
15. Результат встроенной функции x = 0.438431 при любой погрешности.
16. Также используем метод половинного деления, но на отрезке [-1, -0.1]:
17. При погрешности 1.000000e-005 корень уравнения х = -0.438427, количество итераций i = 16.
18. При погрешности 1.000000e-007 корень уравнения х = -0.438431, количество итераций i = 23.
19. При погрешности 1.000000e-009 корень уравнения х = -0.438431, количество итераций i = 29.
20. Результат встроенной функции x = -0.438431 при любой погрешности.
21. Используем метод Ньютона для функции полинома на отрезке [2.5, 9.3], начиная с ϵ = 10^(-5). Также учитывается корень, найденный встроенный функцией fzero. Получаем следующее:
22. При погрешности 1.000000e-005 корень уравнения х = 3.002106, количество итераций i = 5. График зависимости погрешности решения от номера итерации для данного количества итераци показан в приложении 6.
23. При погрешности 1.000000e-007 корень уравнения х = 3.002106, количество итераций i = 6.
24. При погрешности 1.000000e-009 корень уравнения х = 3.002106, количество итераций i = 6.
25. Апостериорная оценка выполнена для любой погрешности.
26. Результат встроенной функции x = 3.002106.
27. Используем тот же метод, но на отрезке [-3.3, -0.3]:
28. При погрешности 1.000000e-005 корень уравнения х = -0.738327, количество итераций i = 4.
29. При погрешности 1.000000e-007 корень уравнения х = -0.738327, количество итераций i = 4.
30. При погрешности 1.000000e-009 корень уравнения х = -0.738327, количество итераций i = 5.
31. Апостериорная оценка выполнена при любой погрешности.
32. Результат встроенной функции x = -0.738327 при любой погрешности.
33. Используем метод Ньютона для функции трансцендентного уравнения. Ищем корень на отрезке [0.1, 1], начиная с ϵ = 10^(-5). Также учитывается корень, найденный встроенный функцией fzero.Получаем следующее:
34. При погрешности 1.000000e-005 корень уравнения х = 0.438431, количество итераций i = 2.
35. При погрешности 1.000000e-007 корень уравнения х = 0.438431, количество итераций i = 3.
36. При погрешности 1.000000e-009 корень уравнения х = 0.438431, количество итераций i = 3.
37. Апостериорная оценка выполнена при любой погрешности.
38. Результат встроенной функции x = 0.438431 при любой погрешности.
39. Используем тот же метод, только на отрезке [-1, -0.1]:
40. При погрешности 1.000000e-005 корень уравнения х = -0.438431, количество итераций i = 2.
41. При погрешности 1.000000e-007 корень уравнения х = -0.438431, количество итераций i = 3.
42. При погрешности 1.000000e-009 корень уравнения х = -0.438431, количество итераций i = 3.
43. Апостериорная оценка выполнена при любой погрешности.
44. Результат встроенной функции x = -0.438431 при любой погрешности.
45. Результат встроенной функции roots для полинома: 3.002106 , -0.738327.

## Модульная структура программы

Программа состоит из 14 модулей:

1. «equation1» – алгебраическое уравнение;
2. «equation2» – трансцендентное уравнение;
3. «finding\_borders» – с помощью данного модуля осуществляется поиск границ корней полинома;
4. «half\_division\_method» – в данном модуле происходит поиск корня полинома с помощью метода половинного деления, на экран выводится результат;
5. «Newton» – это функция, в которой осуществлен метод Ньютона для алгебраического уравнения, входные параметры – это сама функция, отрезок, на котором нужно искать корень уравнения и вектор коэффициентов функции. На выходе – найденный корень, который также выводится на экран;
6. «Newton's\_method\_call» – модуль, который вызывает функцию «Newton», применяется только для алгебраического уравнения;
7. «graphic\_interpretation» –модуль, где реализуется графическая интерпретация метода Ньютона на примере алгебраического уравнения;
8. «Newton\_for\_tr» – это функция, в которой осуществлен метод Ньютона для трансцендентного уравнения, входные параметры – это сама функция, отрезок, на котором нужно искать корень уравнения. На выходе – найденный корень, выводится на экран;
9. «Newton's\_method\_call\_for\_tr» – модуль, который вызывает функцию «Newton\_for\_tr», применяется только для трансцендентного уравнения;
10. «graphic\_interpretation\_for\_tr» – модуль, где реализуется графическая интерпретация метода Ньютона на примере трансцендентного уравнения;
11. «Ass» - это функция, в которой осуществляется апостериорная оценка, входной параметр - заданная точность ϵ и отрезок, на котором мы ищем корень. На выходе результат оценки, выполнена она или нет выводится на экран.
12. «Check» - функция, которая рисует графики зависимости погрешности решения от номера итерации для метода половинного деления и Ньютона.
13. «filling» - функция помогает реализовать график зависимости для метода половинного деления, входные параметры: root – корень , f – функция , AB – отрезок, в котором лежит корень, e – заданная точность. На выходе массив, заполненный разницей по модулю между корнем и приближенным решением в зависимости от итерации.
14. «filling\_for\_Newton» - функция помогает реализовать график зависимости для метода Ньютона, входные параметры: root – корень , f – функция , df – вторая производная, ass – оценка погрешности, AB – отрезок, в котором лежит корень. На выходе массив, заполненный разницей по модулю между корнем и приближенным решением в зависимости от итерации.

## Численный анализ решения задач

У нас стояла 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. Но он достаточно медленный, корень с точностью 10^-9 достигается с помощью 33 итераций. Метод Ньютона обладает высокой скоростью сходимости, и с помощью него мы нашли корень с точностью до 10^-9 всего за 6 итераций. Но по сравнению с методом половинного деления недостатком этого метода является необходимость вычисления на каждой итерации не только левой части уравнения, но и ее производной. Таким образом, мы изучили алгоритмы этих методов для нахождения корней.

## Краткие выводы

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

# Решение СЛАУ прямыми методами.

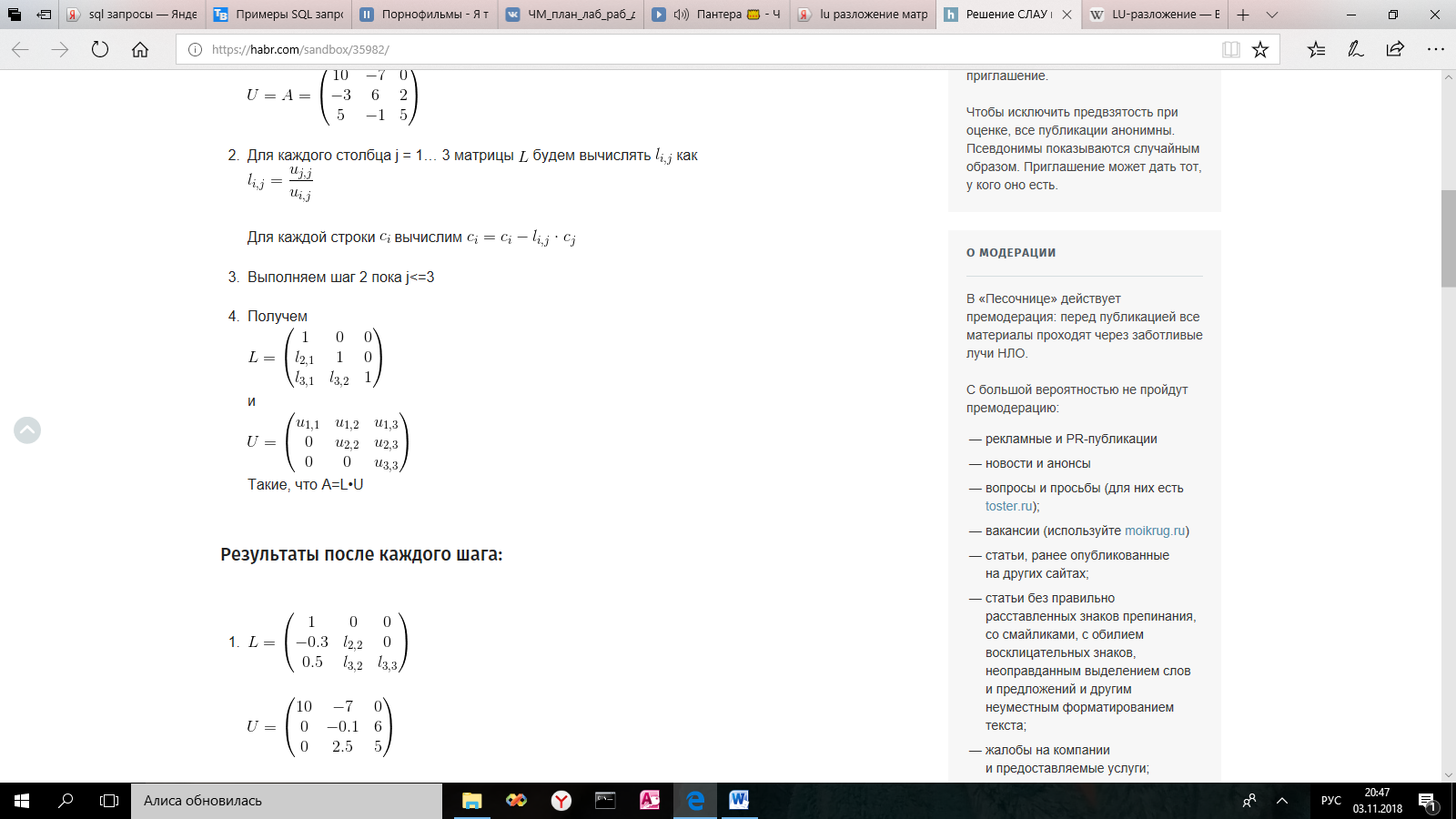
## Формулировка задачи и ее формализация.

В данной работе необходимо найти решение СЛАУ вида , где –вещественная матрица коэффициентов данной системы, - вектор неизвестных (вектор-решение) с вещественными координатами, - вектор свободных членов. Используется метод LU-разложения. Также нужно сравнить полученное решение с точным решением, проверив, таким образом, вычислительную ошибку для матриц с разными числами обусловленности. Размерность системы не менее 10.

## Алгоритм метода и условия его применимости

Мы используем метод LU-разложение. Пусть - данная матрица, и - соответственная нижняя (левая) и верхняя (правая) треугольные матрицы. Существует соответствующая теорема:

Если все главные миноры квадратной матрицы A отличны от нуля, то существуют такие нижняя L и верхняя U треугольные матрицы, что . Если элементы диагонали одной из матриц L или U фиксированы (ненулевые), то такое разложение единственно.



Алгоритм:

1. Находим матрицы L и U по следующим формулам:

Значит, выражение вида мы можем записать в виде эквивалентного уравнения .

1. Введем вектор вспомогательных переменных , эквивалентное уравнение можно переписать в виде системы .Следовательно, решение данной изначально системы свелось к последовательному решению двух систем с треугольными матрицами коэффициентами.
2. Рассмотрим первое уравнение . Все могут быть найдены последовательно по формуле

.

1. Рассмотрим второе уравнение . Все (то есть вектор-решение) могут быть найдены по формуле

.

## Предварительный анализ задачи

## Проверка условий применимости метода

По теореме о LU-разложении матрицы все главные миноры раскладываемой матрицы должны быть отличны от 0. Возьмем матрицу , у которой первый главный минор равен нулю. При разложении получили следующие две матрицы: , . Легко заметить, что получившаяся матрица L совершенно не соответствует нижней треугольной матрице, которая должна была бы получиться, в то время как U является верхней треугольной матрицей. Результатом произведения матриц L и U является исходная матрица A.

## Тестовый пример с детальными расчетами для задачи малой размерности (LU-разложение)

Проверим главные миноры квадратной матрицы A:

1. ;

,

\* y =

## Перечень контрольных тестов для иллюстрации метода средствами пакета MATLAB

Рассмотрим матрицу Гильберта, заданную формулой при i, j = 1, 2, 3…, n. Матрица Гильберта (в дальнейшем H) является [симметричной](http://wiki-org.ru/wiki/%D0%A1%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0) [положительно определённой](http://wiki-org.ru/wiki/%D0%9F%D0%BE%D0%BB%D0%BE%D0%B6%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE_%D0%BE%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D1%91%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0) матрицей. Также она плохо обусловлена. Код представлен в приложении 7. Возьмем 3 матрицы Гильберта размерности 5, 10, 15:

1. n = 5. Число обусловленности матрицы H при такой размерности равно 4.7661e+005. Раскладываем матрицу Гильберта с помощью LU-разложения. Нам необходимо решить уравнение , поэтому мы рандомно задаем вектор свободных членов b и, выполняя алгоритм метода, получаем вектор-решение x1. Найдем погрешность вычислений с помощью вектора невязок. Таким образом, получили следующее значение вектора r1: r1 = 1.0e-011 \* [0.0033, -0.1391, 0.2980, -0.2852, -0.0327].

Внесем возмущение в матрицу Гильберта на 3 процента. Выполнили алгоритм, получили новый вектор-решение x2 и следующий вектор невязок r2: r2 = 1.0e-011 \* [0.0033, -0.3210, 0.1161, 0.0786, 0.1492].

Внесем возмущение в вектор b на 3 процента. Получили вектор невязок r3: r3 =1.0e-011 \* [-0.0257, -0.3670, -0.1660, -0.0736, 0.2010].

1. n = 10. Число обусловленности матрицы H при такой размерности равно 1.6025e+013. Также рандомно создаем вектор b, получаем вектор-решение x1 и r1: r1 = 1.0e-003 \* [-0.0579, 0.0971, 0.0712, 0.2490, 0.0006, -0.0522, 0.0088, -0.0633, 0.0866, 0.0284].

Внесем возмущение в матрицу Гильберта на 3 процента. r2 = 1.0e-003 \* [ -0.0273, 0.0055, 0.1780, -0.2393, -0.0299, -0.1285, 0.0545, -0.1244, 0.0507, 0.0742].

Внесем возмущение в вектор b на 3 процента. r3 = 1.0e-003 \* [-0.0263, -0.1782, 0.0593, -0.0179, -0.1073, -0.0638, -0.0232, 0.0245, -0.0391, -0.0783].

1. n = 15. Число обусловленности матрицы H при такой размерности равно 2.4469e+017. r1 = [0.5901, 1.2991, 2.9382, -0.0561, -1.7560, 1.4962, -2.1282, -3.6822, 5.4872, -0.9647, 0.4065, 3.7698, 1.2547, -1.8342, 2.0171].

Внесем возмущение в матрицу Гильберта на 3 процента. r2= [0.3401, 30.2991, -4.3118, -24.3061, -5.8810, 12.3712, -33.3782, 35.9428, 60.1122, -16.7147, 4.6565, -13.4802, -5.8703, 1.5408, 26.8921].

Внесем возмущение в вектор b на 3 процента. r3 = [-0.6297, 2.8380, -0.8911, -0.4153, -0.6274, -0.0276, -2.3295, -3.8189, -0.4195, 0.4439, -3.7938, 1.8254, -2.7914, -2.7630, -2.2761].

Теперь рассмотрим созданную нами хорошо обусловленную матрицу A размером 10\*10. Также используем код, представленный в приложении 7:

1. Число обусловленности матрицы H при такой размерности равно 25.0625. Пользуясь алгоритмом метода LU-разложения и внося возмущение в матрицу A и в вектор b, получаем три вектора невязок:

r1 = 1.0e-013 \* [-0.0933, -0.0444, 0.0533, -0.1599, 0, 0, 0.0089, 0.0711, -0.0488, 0.1421];

r2 (вектор невязок при измененной матрице A) = 1.0e-013 \* [-0.0222, -0.1155, -0.0533, 0.0178, 0.0355, 0.1421, -0.0266, 0.0888, -0.1155, 0];

r3 (вектор-невязок при измененном векторе b) = 1.0e-013 \* [-0.0921, 0.0622, 0.0799, -0.1510, 0, -0.0977, 0.0178, -0.0178, -0.0555, 0.1421].

## Модульная структура программы

Программа состоит из 2 модулей:

1. «Hilbert\_matrix» - модуль, в котором происходит работа с матрицей Гильберта: выполняется алгоритм, находятся вектора-решения, вносятся возмущения в матрицу и вектор b, вычисляются вектора невязок.
2. «Well\_conditioned\_matrix» - модуль, в котором происходит работа с хорошо обусловленной матрицей. Аналогично модулем «Hilbert\_matrix».

## Численный анализ решения задачи

Для хорошо обусловленной матрицы:

Cond(A) = 15,3333;

Внесение возмущений в b.

.

Внесение возмущений в A.

.

Для плохо обусловленной матрицы:

Cond(A) = 27272727,2698;

Внесение возмущений в b.

.

Внесение возмущений в A.

.

## Краткие выводы

С помощью данной работы мы изучили метод LU-разложения для нахождения вектора-решения уравнения вида , а также проверили вычислительную ошибку для матриц с разными числами обусловленности.

# Приложение

Приложение 1: код метода половинного деления

while abs(a - b) > 2\*e

x = (a + b) / 2;

z = y(x);

if (z == 0)

break;

end

if (z \* y(a) < 0)

b = x;

else

a = x;

end

i = i + 1;

end

x = (a + b) / 2;

Приложение 2: код метода Ньютона

DF = polyder(p); %или DF = diff(sym('x').^2 - cos(pi\*(sym('x'))));

Xkm = (x0 + x1) / 2;

w = subs(DF, 'x', Xkm);

Xk = Xkm - (f(Xkm) / w);

i = 1;

ass = Ass(e, tx);

while abs(Xkm - Xk) > ass

Xkm = Xk;

w = polyval (DF, Xkm);

Xk = Xkm - f(Xkm) / w;

i = i + 1;

end

Код апостериорной оценки Ass:

function assesment = Ass(e, AB) %e - заданная точность , AB - отрезок

%первая производная

df = diff(3.2004 \* sym('x').^4 -0.7934 \* sym('x').^3 -16.3394 \* sym('x').^2 -26.4327 \* sym('x') -11.8793); %или df = diff(sym('x').^2 - cos(pi\*(sym('x'))));

df = inline(df);

m = fminbnd(df, AB(1), AB(2));

%вторая производная

df = diff(3.2004 \* sym('x').^4 -0.7934 \* sym('x').^3 -16.3394 \* sym('x').^2 -26.4327 \* sym('x') -11.8793); %или df = diff(sym('x').^2 - cos(pi\*(sym('x'))));

ddf = diff(df);

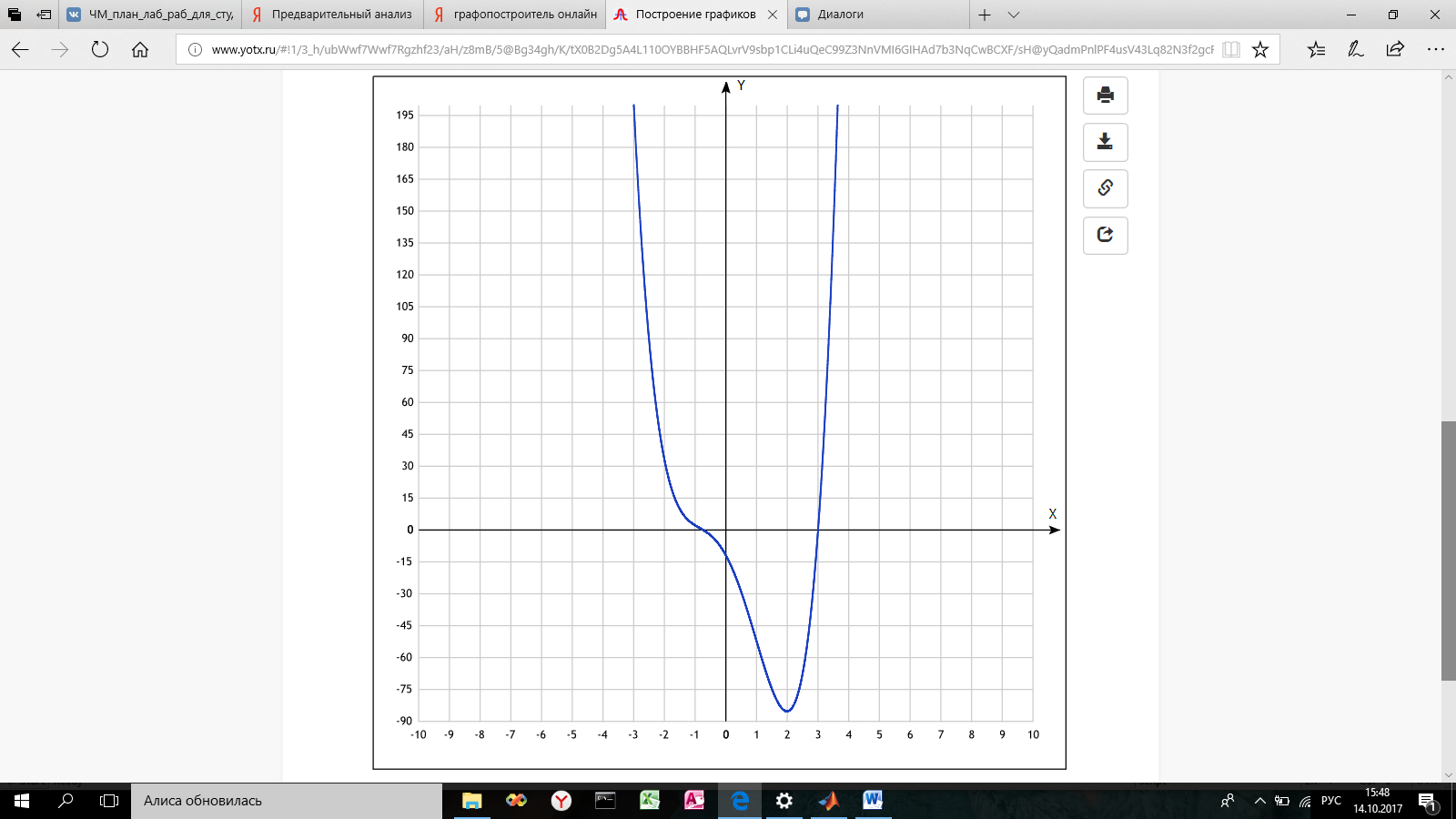
ddf = -1 \* ddf;

ddf = inline(ddf);

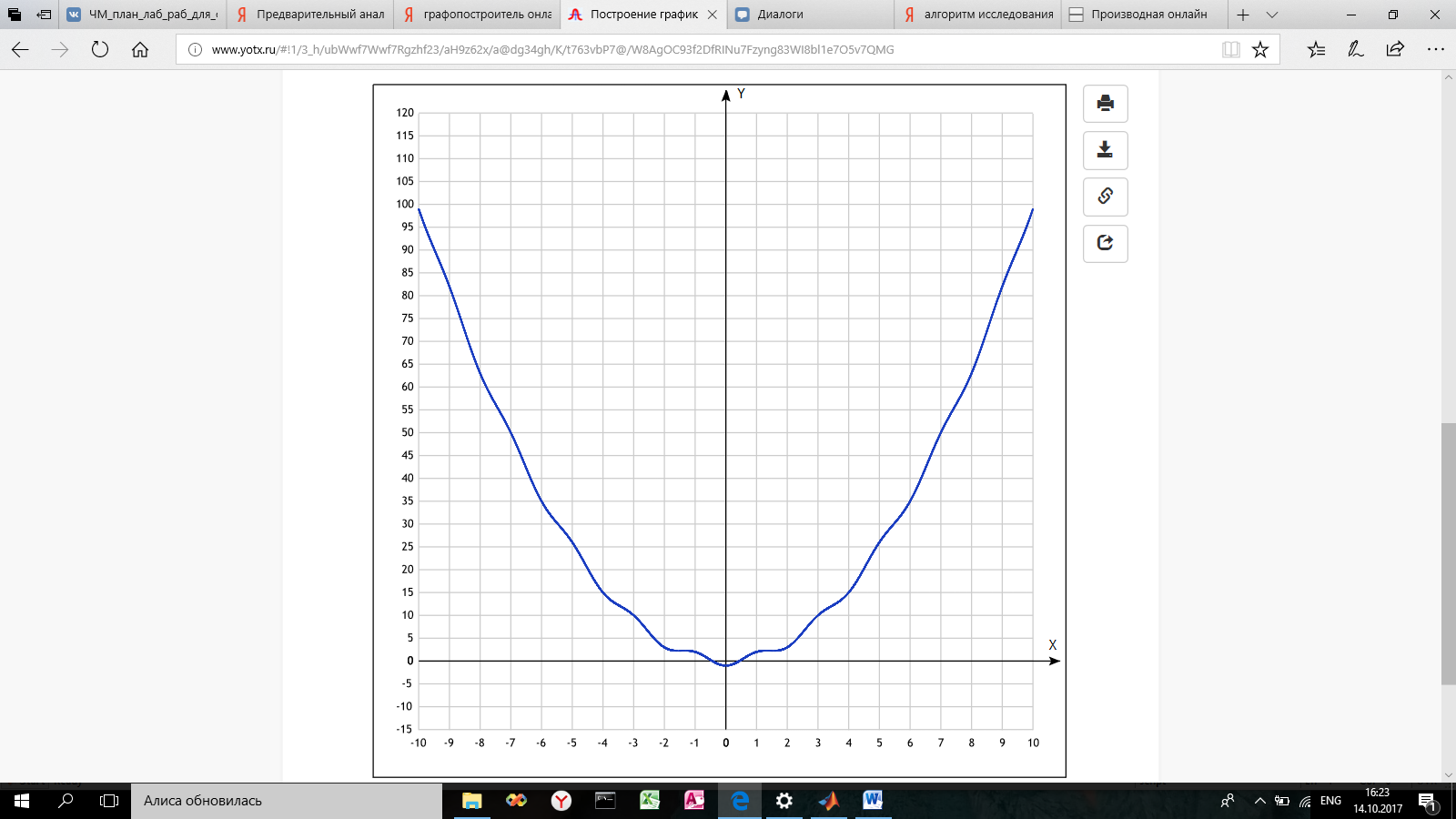
M = fminbnd(ddf, AB(1), AB(2));

assesment = ((2\*m\*e)/M)^(1/2);

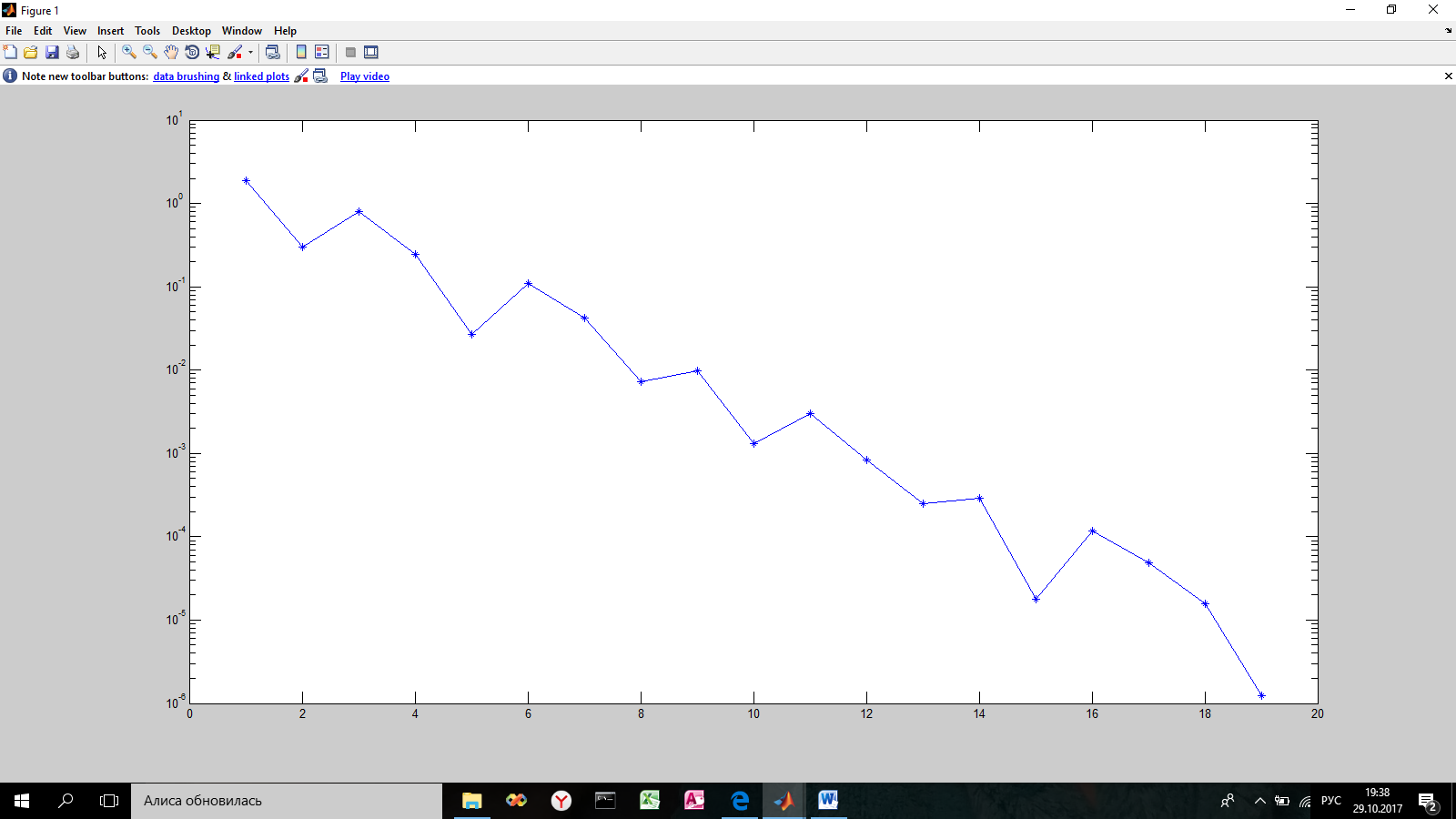
Приложение 3: график полинома



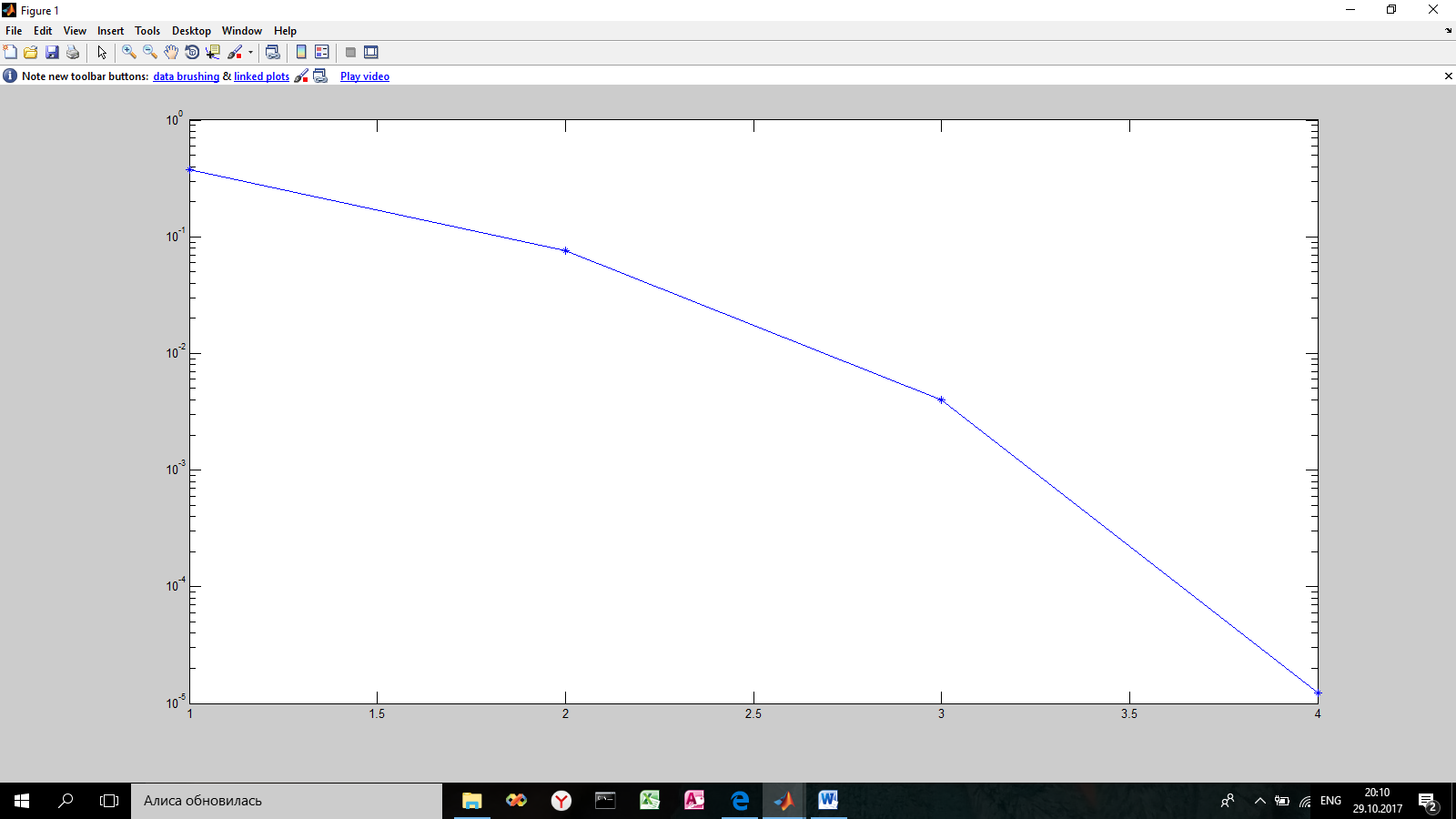
Приложение 4: график трансцендетной функции



Приложение 5: зависимость погрешности решения от номера итерации, метод половинного деления



Приложение 6: зависимость погрешности решения от номера итерации, метод Ньютона

\

Приложение 7: код метода LU-разложения при внесении возмущения в матрицу Гильберта (или в матрицу A) и вектор свободных коэффициентов b:

H = hilb(n); %задаем матрицу Гильберта, n = 5, 10, 15

или

d = rand(1, 10);

D = diag(d); %диагональная матрица размером 10\*10

x = rand(10) \* 10;

A = inv(x) \* D \* x;

%M – матрица A или матрица H, в зависимости от того, с какой мы работаем в данный момент

[L, U] = lu(M); %LU-разложение

cond(M); %число обусловленности

b = rand(1, n) \* n;%задаем рандомный вектор b

b = b';

x1 = U \ (L \ b); %находим вектор-решение

r1 = b – M \* x1; %вектор невязок

M1 = M \* 1.03; %внесение возмущения на 3% в матрицу

[L1, U1] = lu(M1);

x2 = U1 \ (L1 \ b);

r2 = b - M1 \* x2; %вектор-невязок при измененной матрице

b1 = b \* 1.03; %внесение возмущения на 3 % в вектор b

x3 = U \ (L \ b1);

r3 = b1 – M \* x3; %вектор невязок при измененном векторе b