**Федеральное государственное автономное образовательное учреждение высшего образования**

**“Санкт-Петербургский политехнический университет Петра Великого”**

Кафедра “Прикладная математика”

Дисциплина “Численные методы”

Вариант 9

Отчет по учебной практике

Студента Хрипункова Д.В.

Группы 3630102/80001

Руководитель практики: Добрецова С.Б.

Санкт-Петербург 2019 г.

1. Формулировка задачи

Рассмотреть задачу приближенного нахождения нулей алгебраической или трансцендентной функции методом половинного деления и методом секущих. Провести математический и численный анализ. Рассматриваем:

Алгебраическую функцию вида

Трансцендентную функцию вида

2. Постановка задачи

Дано f(x) = 0, x , f: , f C[a, b]. На отрезке ξ: f(ξ) = 0.

Найти ζ [a, b] – приближённое значение корня с точность ε, т.е.

ζ [a, b] |ζ - ξ| < ε

3. Алгоритм метода

1) Метод половинного деления:

Дана f(x) = 0, монотонная, непрерывная функция, имеющая корень на промежутке [a; b], где f(a)\*f(b) < 0. Сначала находим середину отрезка [a; b] c = (a+b)/2. Считаем f(c), если f(a)\*f(c) < 0, то b = c, если f(a)\*f(c) > 0, то a = c и f(a) = f(c). Продолжаем итерации, пока |b – a| >= 2 \* ε, где ε - точность нахождения корня

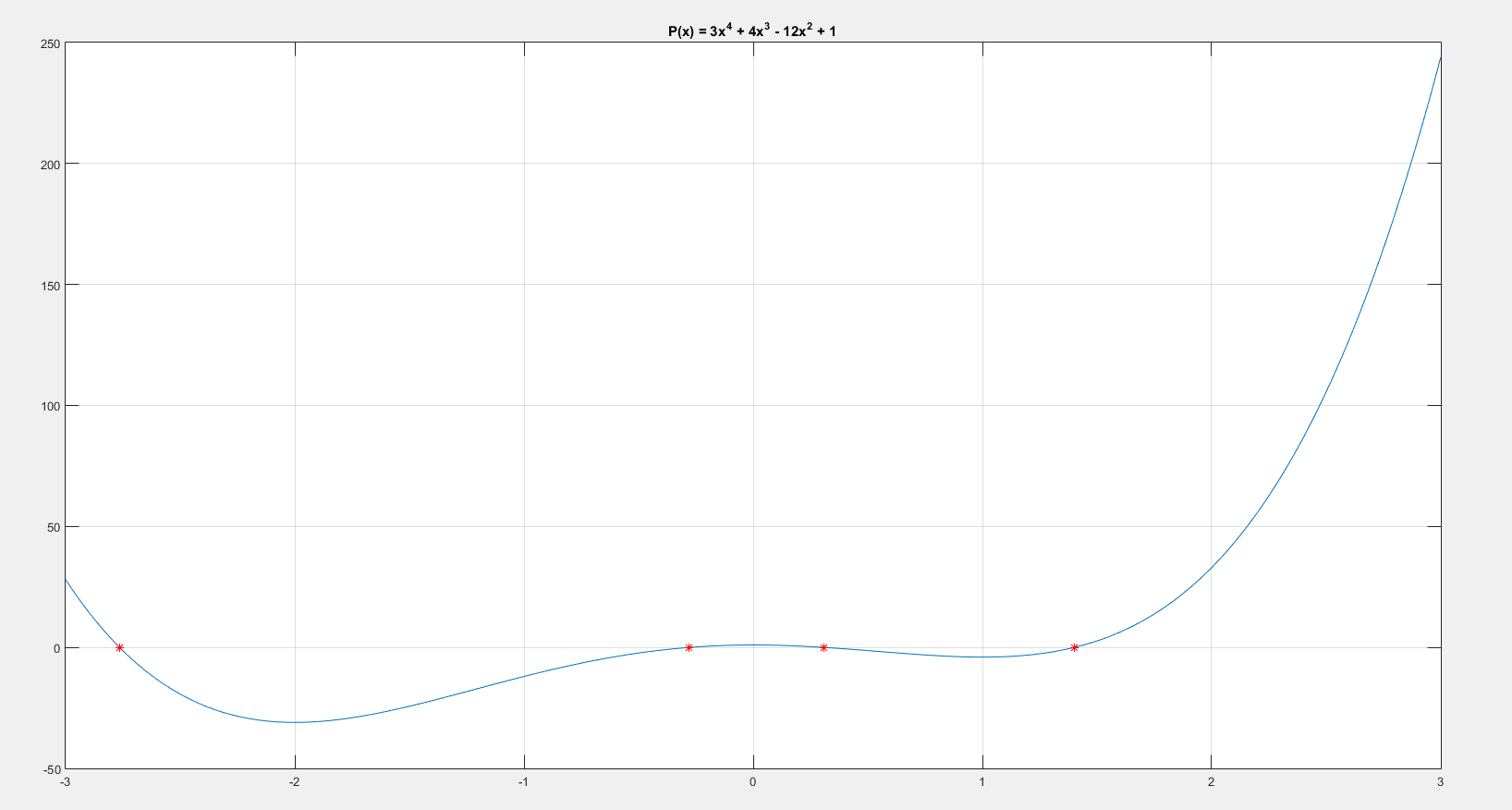
2) Метод секущих:

Метод секущих – модификация метода Ньютона. Дана f(x) = 0, монотонная, непрерывная функция, имеющая корень на промежутке [a; b]. Условия те же, что и для метода Ньютона. В формуле расчёта , (k = 2, 3, 4 …) можно заменить вычисление производной f’(x(k)) на приближённое значение производной через значение функции f. и тогда формула итераций принимает вид Таким образом на каждом шаге нужно знать две точки, а на первом шаге берутся две начальные точки, удовлетворяющие сходимости метода и принадлежащие промежутку [a; b]. Для выбора x(0) нужно использовать условие Фурье: f(x(0))\*f’’(x(0)) > 0, а точку x(1) можно взять в непосредственной близости от x(0), желательно между точкой x(0) и искомым корнем. После того, как были выбраны две начальные точки x(0) и x(1), можно рассчитать следующую x(2) по итерационной формуле для x(k+1). Итерации продолжаются, пока , α – минимальное значение первой производной на промежутке, β - максимальное значение второй производной на промежутке, ε – требуемое приближение.

4. Анализ задачи

В пакете Matlab был построен график полинома P(x) =

по которому видны 4 различные точки, в которых P(x) обращается в 0, две из которых являются положительными, а две отрицательными. Полином непрерывен на всём промежутке, что достаточно для применения метода половинного деления на интервалах с разными знаками на концах. Для применения метода секущих необходимо существование второй производной на отрезке.



По теореме о верхней границе корней полинома max = 1 + = 3, нижняя граница положительных корней min = 1/(1+4) = 0.2 , верхняя граница отрицательных корней max = -1/(1+4) = -0.2, а нижняя граница min = -1 - = -5. Таким образом, мы можем сказать, что все корни полинома P(x) лежат в промежутках [-5, -0.2] и [0.2, 3]

Пользуясь графиком, сузим область поиска корней до промежутков [-3, -2.5], [-0.5, 0], [0, 0.5], [1.1, 2]

Действительно,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| X | -3 | -2.5 | -0.5 | 0 | 0.5 | 1.1 | 2 |
| P(x) | 28 | -19.31 | -2.31 | 1 | -1.31 | -3.8 | 33 |

На концах каждого из 4 промежутков полином принимает значения разных знаков, а значит принимает значение 0 в каждом из них.

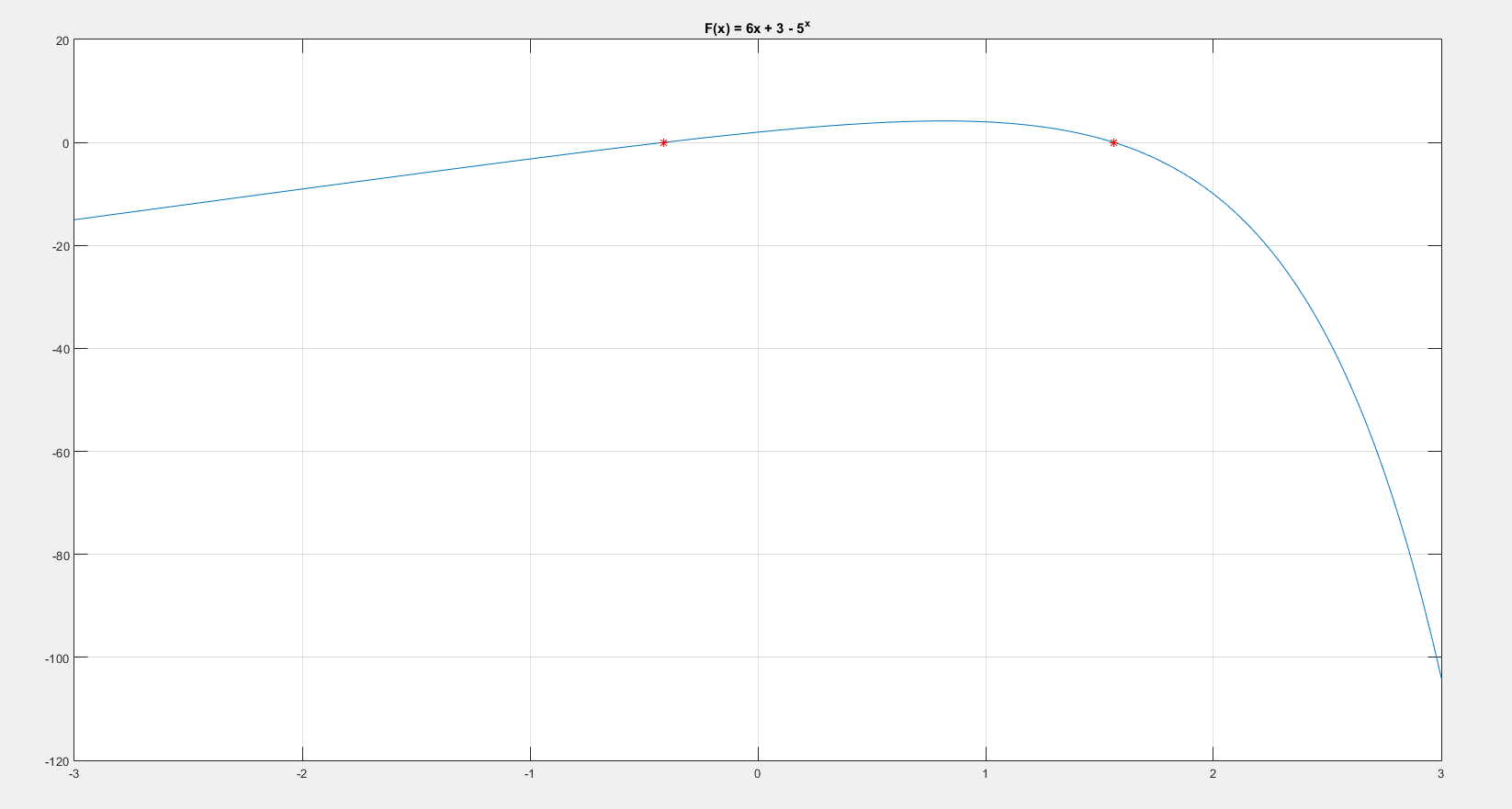
P’’(x) = 36x2 + 24x – 24

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| X | -3 | -2.5 | -0.5 | 0 | 0.5 | 1.1 | 2 |
| P’’(x) | 228 | 141 | -27 | -24 | -3 | 45.96 | 168 |

Стационарными точками, в которых знак функции и второй производной совпадают, являются -3, -0.5, 0.5 и 2.

Аналогично был построен график функции F(x) = 6x + 3 - 5x по которому видны 2 различные точки обращения функции в ноль, из которых одна положительна, а вторая отрицательна. Функция также непрерывна на всём промежутке, что позволяет применять метод половинного деления.

Из графика видно, что один из корней лежит в промежутке [-0.5, 0], а второй [1.5, 2]



Действительно,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X | -0.5 | 0 | 1.5 | 2 |
| F(x) | -0.45 | 2 | 0.82 | -10 |

На концах промежутков функция принимает значения разных знаков.

В качестве точки x0 для метода секущих на промежутке [1.5, 2] для F(x) возьмём точку x = 1.5, удовлетворяющую условию Фурье.

5. Тестовый пример

Возьмём f(x) = x2 – 4

f(x) – полином второй степени, т.е. имеет до 2 различных корней, в данном случае это 2 и -2

Составим промежутки, на которых будем искать корни. По теореме о верхней и нижней границе полинома корни лежат в промежутке [-3; 3]. Для того, чтобы функция принимала различные значения на концах промежутков, рассмотрим промежутки [-3; 0] и [0; 3]. f(-3) = f(3) = 5, f(0) = -4

Рассмотрим промежуток [-3; 0] по методу половинного деления и сделаем по нему 4 итерации вручную.

1. Середина промежутка [-3; 0] c = (a+b)/2 = -1.5, f(c) = f(-1.5) = -1.75, f(a) \* f(c) < 0 => b = c. Далее промежуток становится [-3; -1.5].
2. Середина c = (a+b)/2 = -2.25, f(c) = 1.0625‬, f(a) \* f(c) > 0 => a = c и f(a) = f(c) = 1.0625‬. Промежуток становится [-2.25; -1.5]
3. Середина c = (a+b)/2 = -1.875, f(c) = -0.484, f(a) \* f(c) < 0 => b = c. Промежуток становится [-2.25; -1.875]
4. Середина c = (a+b)/2 = -2.0625, f(c) = 0.2539, f(a) \* f(c) > 0 => a = c и f(a) = f(c) = 0.2539. Промежуток поиска становится [-2.0625; -1.875].

Теперь рассмотрим промежуток [0; 3] по методу секущих и сделаем по нему 4 итерации вручную.

1. В качестве точки x(0) возьмём 1, x(1) = 3. Рассчитаем x(2) по формуле . x(2) = 3 – 5 \*(3 - 1) / (5 - (-3)) = 1.75
2. Теперь мы имеем x(1) = 3, x(2) = 1.75. Рассчитаем x(3) = 1.75 – (-0.9375) \* (1.75 - 3) / (-0.9375 - 5) = 1.947
3. x(2) = 1.75, x(3) = 1.947. Рассчитаем x(4) = 1.947 – (-0.209191) \* (1.947 – 1.75) / (-0.209191 – (-0.9375)) = 2.00358
4. x(3) = 1.947, x(4) = 2.00358. Рассчитаем x(5) = 2.00358 – (0.0143328164) \* (2.00358 – 1.947) / (0.0143328164 – (-0.209191)) = 1.9999519

6. Контрольные тесты

Для полинома P(x) рассмотрим промежуток [-3, -2.5], для функции F(x) [1.5, 2]

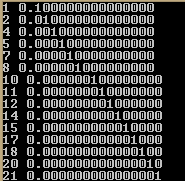
Будем рассматривать зависимость числа итераций от приближения ответа и от начального приближения в методе секущих и методе половинного деления.

При рассмотрении зависимости от приближения ответа будем считать точки начального приближения неизменными и заданными в зависимости от конкретной задачи, а точность решения менять от 10-1 до 10-15 с шагом в 0.1.

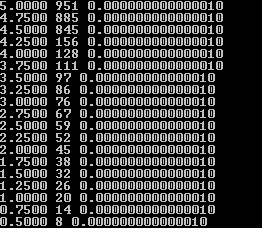
При рассмотрении зависимости числа итераций от начального приближения будем брать приближение решения фиксированным и равным 10-14, а начальное приближение выбирать в зависимости от конкретной задачи и с каждым шагом сдвигать границы на 0.125 с каждой стороны, сужая промежуток. То есть на каждом шаге границы промежутка [a, b] a += 0.125, b-= 0.125.

Проверка метода половинного деления и секущих на полиноме:

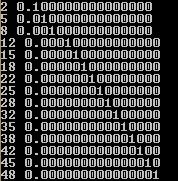
Зависимость числа итераций (слева) от приближения решения (справа)



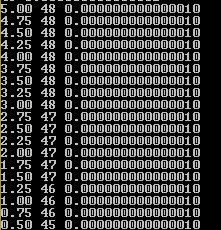
Зависимость числа итераций (2 колонка) от начального приближения (1 колонка). Справа выводится приближения решения



Зависимость числа итераций (слева) от приближения решения (справа)

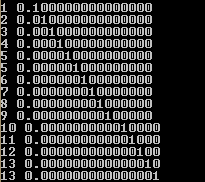


Зависимость числа итераций (2 колонка) от начального приближения (1 колонка). Справа выводится приближения решения



Проверка методов для трансцендентной функции:

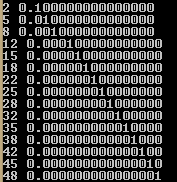
Зависимость числа итераций (слева) от приближения решения (справа)



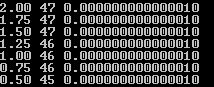
Зависимость числа итераций (2 колонка) от начального приближения (1 колонка). Справа выводится приближения решения



Зависимость числа итераций (слева) от приближения решения (справа)



Зависимость числа итераций (2 колонка) от начального приближения (1 колонка). Справа выводится приближения решения



7. Модульная структура программы

typedef struct {

double x, y;

} point\_t; - структура, содержащая координаты x и y точки

double SolveFunc(double x, int funcNum) – функция возвращает значение одной из заданных функций в обозначенной точке

point\_t SearchNextPoint(point\_t c1, point\_t c2, int funcNum) – функция возвращает координаты новой точки для метода секущих, получая координаты двух предыдущих и номер заданной функции

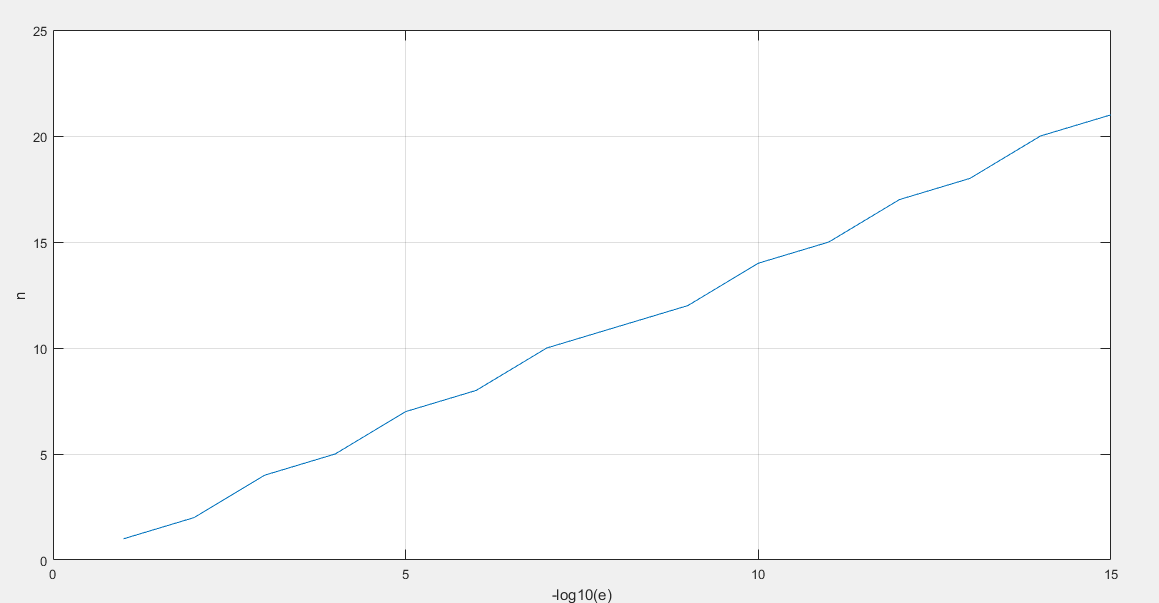
point\_t DivisionSearchNextPoint(point\_t c1, point\_t c2, int funcNum) - функция возвращает координаты новой точки для метода половинного деления, получая координаты двух предыдущих и номер заданной функции

void HordesSolution(point\_t p1, point\_t p2, double e, int funcNum, FILE \*F) – функция выводит в заданный файл координаты точки решения, полученного методом секущих, с количеством затраченных итераций и текущим приближением, получая на вход координаты концов отрезка приближения, приближение и номер функции

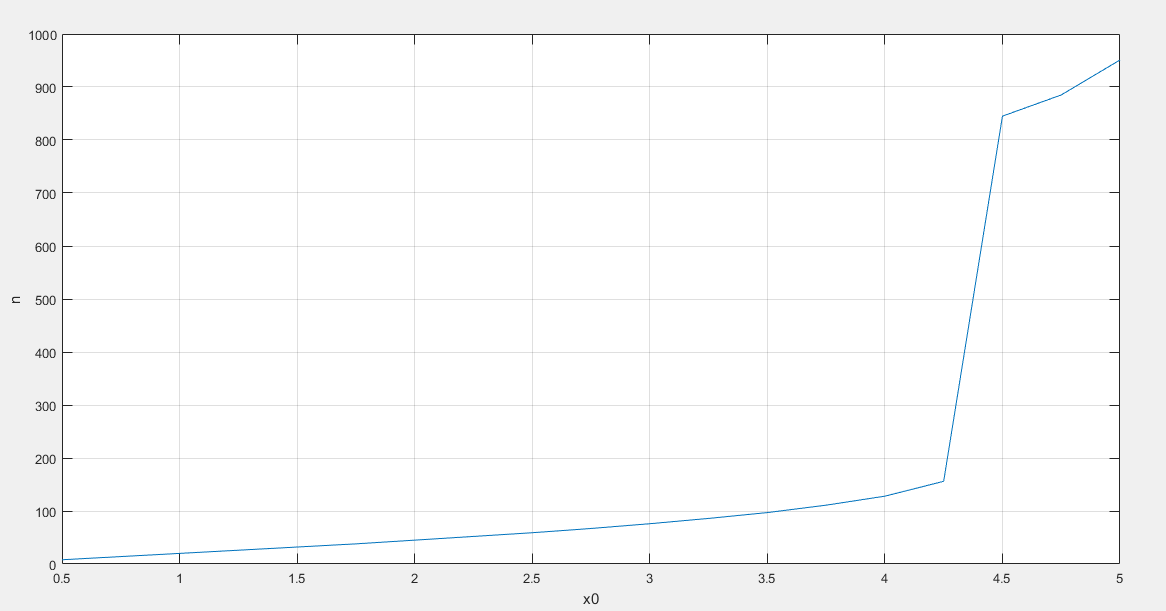
void DivisionSolution(point\_t p1, point\_t p2, double e, int funcNum, FILE \*F) – функция выводит в заданный файл координаты точки решения, полученного методом половинного деления, с количеством затраченных итераций и текущим приближением, получая на вход координаты концов отрезка приближения, приближение и номер функции

8. Численный анализ

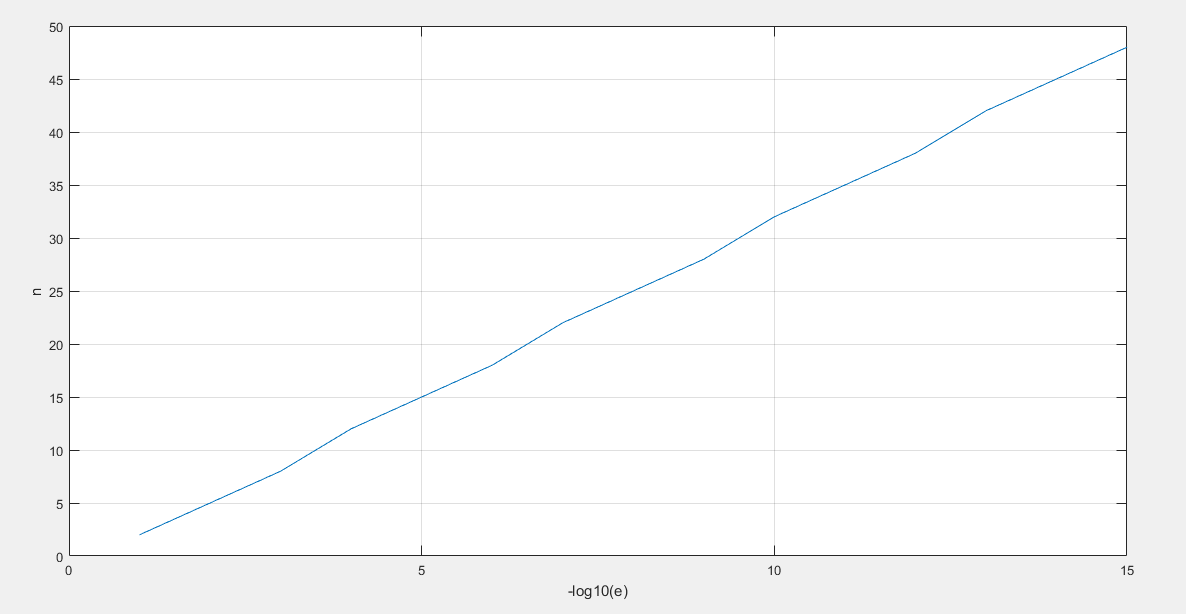
Графики зависимостей, построенные в пакете MATLAB после предварительной обработки данных в программе на Visual Studio:



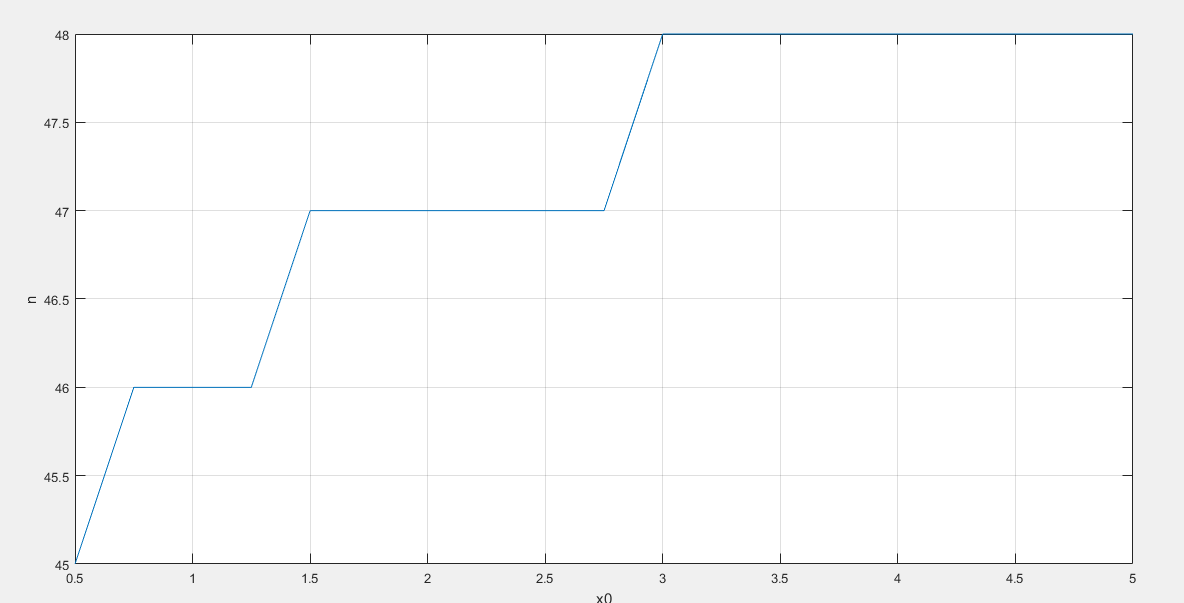
Полином P(x), метод секущих, зависимость от приближения решения с начальными точками -3 и -2.5



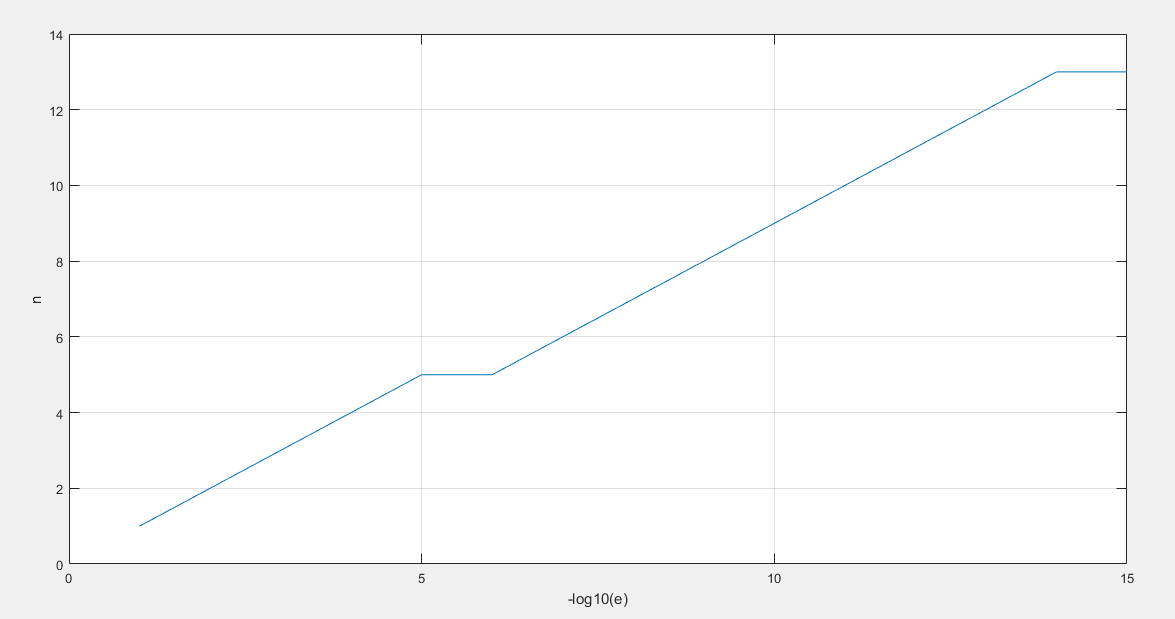
*Полином P(x), метод секущих, зависимость от начального приближения с фиксированной точностью 10-14*



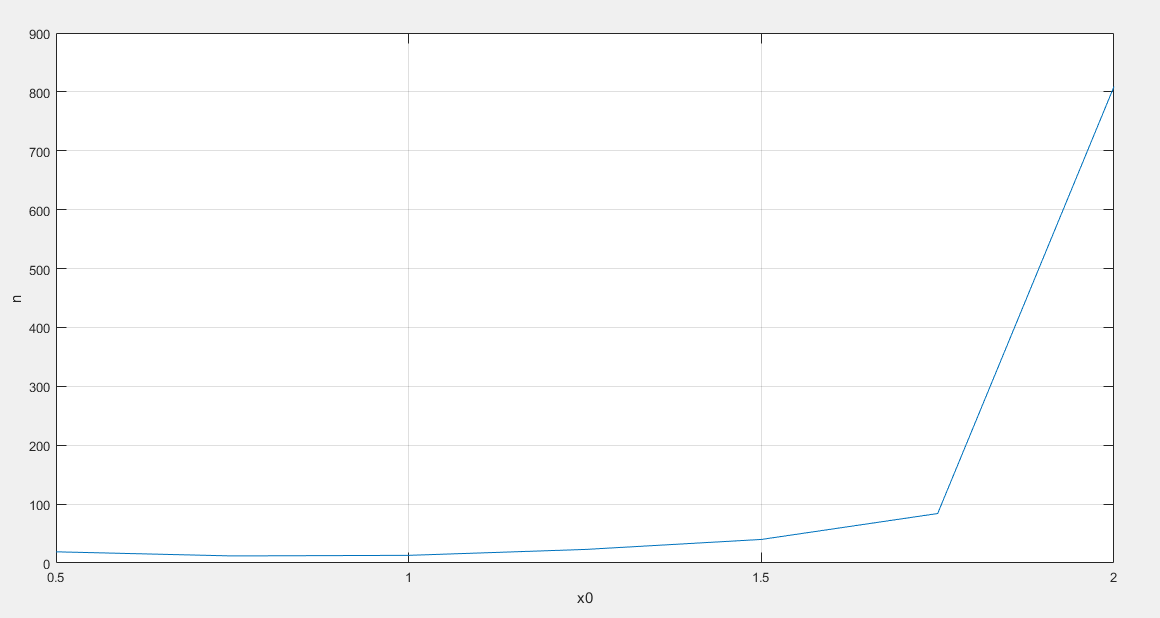
Полином P(x), метод половинного деления, зависимость от приближения решения с начальными точками -3 и -2.5



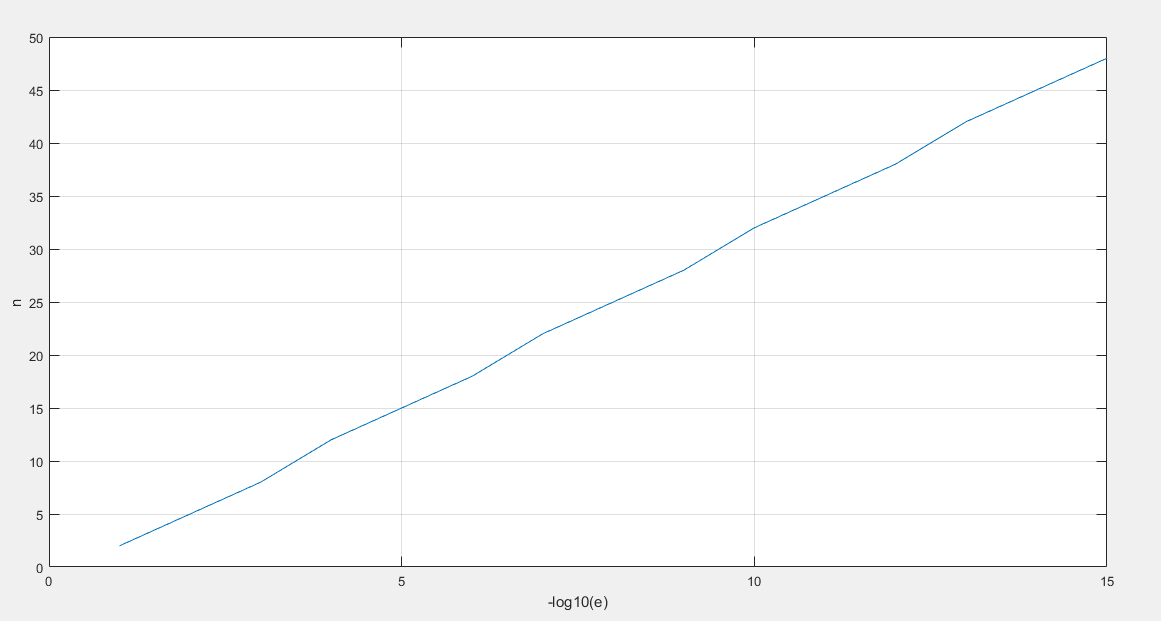
*Полином P(x), метод половинного деления, зависимость от начального приближения с фиксированной точностью 10-14*



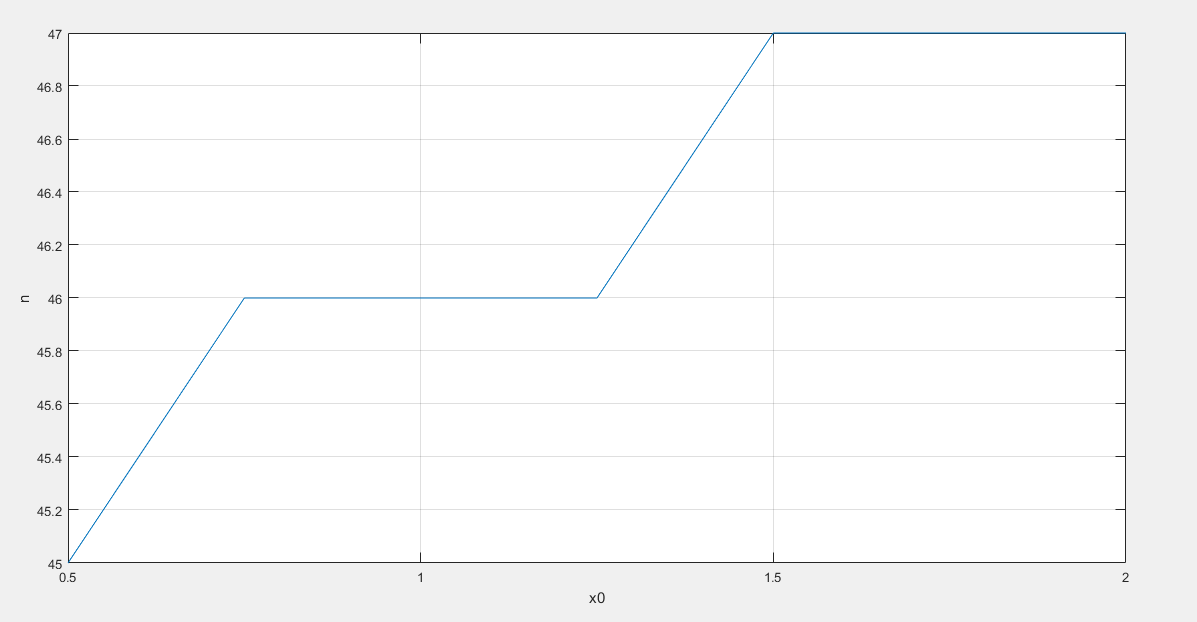
Функция F(x), метод секущих, зависимость от приближения решения с начальными точками 1.5 и 2



Функция F(x), метод секущих, зависимость от начального приближения с фиксированной точностью 10-14

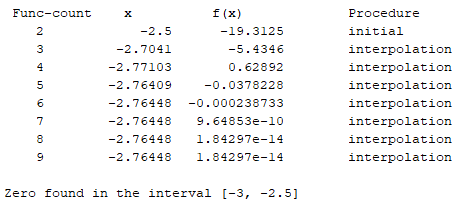


Функция F(x), метод половинного деления, зависимость от приближения решения с начальными точками 1.5 и 2

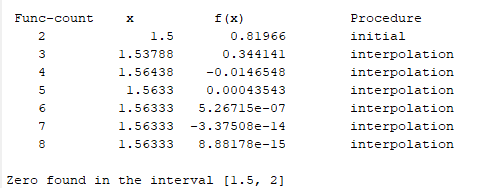


Функция F(x), метод половинного деления, зависимость от начального приближения с фиксированной точностью 10-14

Также было проведено исследование функции fzero из пакета MATLAB для полинома P(x) с начальными точками -3 и -2.5 и трансцендентной функции F(x) с начальными точками 1.5 и 2. Приближение решения в обоих случаях взято одинаковым и равно 10-15:



fzero для полинома P(x) с начальными точками -3 и -2.5 и приближением 10-15



fzero для функции F(x) с начальными точками 1.5 и 2 и приближением 10-15

На основе полученных выше графиков можно сделать следующие выводы:

1. И для полинома, и для трансцендентной функции в методе секущих зависимость числа итераций от приближения решения возрастает практически линейно.
2. Для трансцендентной функции метод секущих оказался гораздо эффективнее метода половинного деления при изменении приближения решения, и наоборот для изменяющегося начального приближения с фиксированным приближением решения 10-14.
3. Для полинома и для трансцендентной функции метод половинного деления даёт линейную зависимость для итераций от приближения решения и ступенчатую для итераций от начального приближения.
4. Для полинома так же, как и для трансцендентной функции, метод секущих заметно лучше при изменении приближения решения и гораздо хуже для зависимости от меняющегося начального приближения.

9. Вывод

Во всех тестах, рассматривающих зависимость количества итераций от приближения корня, метод секущих оказывается эффективнее метода половинного деления, находя приближённое значение корня за меньшее в 2-4 раза количество итераций.

В тестах, выявляющих зависимость количества итераций от начального приближения, метод половинного деления показывает более стабильные результаты, чем метод секущих, хотя второй при определённых приближениях может справляться и за гораздо большее число итераций, и за гораздо меньшее. В данном случае это обусловлено недостаточным количеством различных тестов, так как большее удаление от корня невозможно из-за близости корней друг к другу.

В целом, метод секущих можно считать более эффективным для использования, так как при одинаковых входных данных выдаёт результат в среднем за меньшее количество итераций.