Оглавление

[Задача 3](#_Toc51879782)

[Теоретическая часть 3](#_Toc51879783)

[Метод бисекции 3](#_Toc51879784)

[Метод секущих 3](#_Toc51879785)

[Метод ложной позиции 3](#_Toc51879786)

[Метод Ньютона 4](#_Toc51879787)

[Практическая часть 4](#_Toc51879788)

**Лабораторная работа №1.**

**«Решение уравнения с одним неизвестным»**

***Кущ Артем, БПМ-18-1***

# Задача

Решить уравнение с помощью метода бисекции, секущих, ложной позиции и Ньютона:

*(вариант 10)*

# Теоретическая часть

## Метод бисекции

Основан на следствии их теоремы Больцано-Коши (если непрерывная функция, определённая на вещественном промежутке, принимает два значения, то она принимает и любое значение между ними):

Если мы ищем ноль, то на концах отрезка функция должна быть противоположных знаков. Разделим отрезок пополам и возьмём ту из половинок, на концах которой функция по-прежнему принимает значения противоположных знаков. Если значение функции в серединной точке оказалось искомым нулём, то процесс завершается.

Повторяем алгоритм до тех пор, пока не достигнем хорошей оценки.

## Метод секущих

Данный метод получается из метода касательных (метод Ньютона) путем замены разностным приближением:

Подставляем:

Повторяем алгоритм до тех пор, пока не достигнем хорошей оценки.

На следующей итерации:

## Метод ложной позиции

Основан на предыдущем методе, меняется только выбор ***xk*** на следующей итерации:

## Метод Ньютона

Для данного метода необходимо и достаточно одной точки.

Для решения уравнения необходимо воспользоваться полезным эквивалентом:

Пропуская доказательство и вывод, получим необходимую нам формулу:

Повторяем алгоритм до тех пор, пока не найдем такую точку xn , которая даст хорошую оценку.

# Практическая часть

Алгоритмы реализованы на языке Python.

Для более удобного изучения лабораторной работы крайне рекомендую изучить Notebook: там возможно не только выполнить алгоритмы в реальном времени, но и изменить код.

<https://drive.google.com/file/d/1gcrUaYyzO5RlSjoH2JdXufMptco07f-H/view?usp=sharing>

## Метод бисекции

def bisection(eq, a, b, e=0.01):

df = pd.DataFrame(data={f'bisection\_{e}': []})

vals = []

Fa, Fb = eq(a), eq(b)

if Fa \* Fb > 0:

raise ValueError('Решение невозможно: f(a) \* f(b) <= 0')

# Пока точность не станет допустимой

while(b - a > e):

# середина

x = np.mean([a, b])

# значение функции в новой средней точке

f = eq(x)

# выбираем новую точку вместо a или b

if f \* Fa > 0:

a = x

else:

b = x

vals += [x]

df[f'bisection\_{e}'] = vals

return df

## Метод секущих

def secant(f,x0,x1,e,N=None):

df = pd.DataFrame(data={f'secant\_{e}': []})

vals = []

x2= x1

while abs(f(x2)) > e:

if f(x0) == f(x1):

print('Деление на ноль')

break

# формула итерационного процесса

x2 = x0 - (x1 - x0) \* f(x0)/(f(x1) - f(x0))

x0 = x1

x1 = x2

vals += [x2]

if N is not None:

if step > N:

print('Не сошлось')

break

df[f'secant\_{e}'] = vals

return df

## Метод ложной позиции

def falsePosition(f, x0, x1, e):

df = pd.DataFrame(data={f'falsePosition\_{e}': []})

step = 1

vals = []

condition = True

while condition:

# формула итерационного процесса

x2 = x0 - (x1-x0) \* f(x0)/( f(x1) - f(x0) )

if f(x0) \* f(x2) < 0:

x1 = x2

else:

x0 = x2

vals += [x2]

step = step + 1

condition = abs(f(x2)) > e

df[f'falsePosition\_{e}'] = vals

return df

## Метод Ньютона

def newton(f, Df, x0, epsilon=0.01):

df = pd.DataFrame(data={f'newton\_{e}': []})

vals = []

try:

xn = x0

while True:

# значение первообразной функции в точке

fxn = f(xn)

if abs(fxn) < epsilon:

df[f'newton\_{e}'] = vals

return df

# значение производной функции в точке

Dfxn = Df(xn)

if Dfxn == 0:

print('Деление на ноль.')

return None

vals += [xn]

# новая точка

xn = xn - fxn/Dfxn

except:

print("Нет решения.")

df[f'newton\_{e}'] = vals

return df

Результат работы каждого алгоритма:

|  |  |  |  |
| --- | --- | --- | --- |
| **Шаг** | **bisection\_0.01** | **bisection\_0.001** | **bisection\_0.0001** |
| **0** | 1,5 | 1,5 | 1,5 |
| **1** | 1 | 1 | 1 |
| **2** | 1,25 | 1,25 | 1,25 |
| **3** | 1,125 | 1,125 | 1,125 |
| **4** | 1,1875 | 1,1875 | 1,1875 |
| **5** | 1,21875 | 1,21875 | 1,21875 |
| **6** | 1,234375 | 1,234375 | 1,234375 |
| **7** | 1,2421875 | 1,2421875 | 1,2421875 |
| **8** |  | 1,23828125 | 1,23828125 |
| **9** |  | 1,240234375 | 1,240234375 |
| **10** |  | 1,239257813 | 1,239257813 |
| **11** |  |  | 1,239746094 |
| **12** |  |  | 1,239990234 |
| **13** |  |  | 1,239868164 |
| **14** |  |  | 1,239929199 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Шаг** | **secant\_0.01** | **secant\_0.001** | **secant\_0.0001** |
| **0** | 1,117444885 | 1,117444885 | 1,117444885 |
| **1** | 1,19151969 | 1,19151969 | 1,19151969 |
| **2** | 1,240106819 | 1,240106819 | 1,240106819 |
| **3** |  |  | 1,239932065 |
| **4** |  |  |  |
| **5** |  |  |  |
| **6** |  |  |  |
| **7** |  |  |  |
| **8** |  |  |  |
| **9** |  |  |  |
| **10** |  |  |  |
| **11** |  |  |  |
| **12** |  |  |  |
| **13** |  |  |  |
| **14** |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Шаг** | **falsePosition\_0.01** | **falsePosition\_0.001** | **falsePosition\_0.0001** |
| **0** | 1,117444885 | 1,117444885 | 1,117444885 |
| **1** | 1,19151969 | 1,19151969 | 1,19151969 |
| **2** | 1,22021764 | 1,22021764 | 1,22021764 |
| **3** | 1,231821436 | 1,231821436 | 1,231821436 |
| **4** | 1,23658228 | 1,23658228 | 1,23658228 |
| **5** | 1,238546578 | 1,238546578 | 1,238546578 |
| **6** |  | 1,239358869 | 1,239358869 |
| **7** |  | 1,239695085 | 1,239695085 |
| **8** |  | 1,239834301 | 1,239834301 |
| **9** |  |  | 1,239891956 |
| **10** |  |  | 1,239915834 |
| **11** |  |  |  |
| **12** |  |  |  |
| **13** |  |  |  |
| **14** |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Шаг** | **newton\_0.01** | **newton\_0.001** | **newton\_0.0001** |
| **0** | 1,5 | 1,5 | 1,5 |
| **1** | 1,246505221 | 1,246505221 | 1,246505221 |
| **2** |  | 1,239657641 | 1,239657641 |
| **3** |  |  |  |
| **4** |  |  |  |
| **5** |  |  |  |
| **6** |  |  |  |
| **7** |  |  |  |
| **8** |  |  |  |
| **9** |  |  |  |
| **10** |  |  |  |
| **11** |  |  |  |
| **12** |  |  |  |
| **13** |  |  |  |
| **14** |  |  |  |