

ИТМО  
Кафедра ВТ  
Вычислительная Математика

Лабораторная работа №1  
«Численные методы решения нелинейных уравнений»

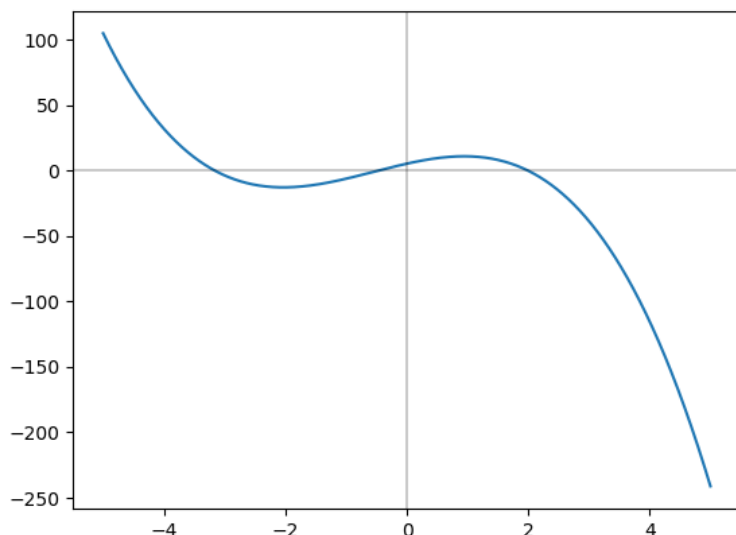
Нестеров Дали Константинович  
Группа Р3202  
Вариант 9

Санкт-Петербург  
2018 год

**Цель:** реализовать методы половинного деления и Ньютона для решения нелинейных уравнений.

**Порядок выполнения работы:**

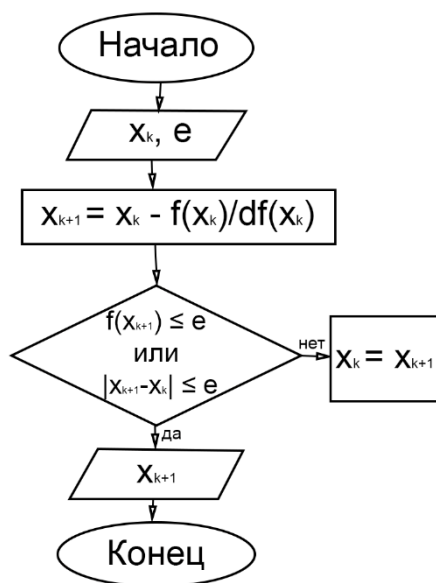
1. Построить график уравнения  $y = -1.8x^3 - 2.94x^2 + 10.37x + 5.38$  (вариант 9)



2. Определить корни уравнения графически
3. Определить интервалы изоляции корней  
(-4, -2), (-1, 0), (1.4, 2.4)
4. Заполнить Таблицы 1 и 2 (см. ниже)
5. Написать программную реализацию задачи

**Блок-схемы используемых методов и их рабочие формулы:**

Блок-схема метода Ньютона



## Блок-схема метода половинного деления



## Рабочие формулы

Метод половинного деления:

$$x_n = \frac{a_n + b_n}{2}$$

Метод Ньютона:

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

## Таблицы:

Таблица вычислений правого крайнего корня нелинейного уравнения методом половинного деления с точностью  $\epsilon = 10^{-2}$  с удержанием 3 знаков после запятой.

Таблица 1

| № шага | a     | b     | x            | f(a)  | f(b)    | f(x)   | a-b   |
|--------|-------|-------|--------------|-------|---------|--------|-------|
| 1      | 1.4   | 2.4   | 1.9          | 9.196 | -11.550 | 2.123  | 1.0   |
| 2      | 1.9   | 2.4   | 2.15         | 2.123 | -11.550 | -3.804 | 0.5   |
| 3      | 1.9   | 2.15  | 2.025        | 2.123 | -3.804  | -0.623 | 0.25  |
| 4      | 1.9   | 2.025 | 1.962        | 2.123 | -0.623  | 0.803  | 0.125 |
| 5      | 1.962 | 2.025 | 1.994        | 0.803 | -0.623  | 0.103  | 0.062 |
| 6      | 1.994 | 2.025 | 2.009        | 0.103 | -0.623  | -0.257 | 0.031 |
| 7      | 1.994 | 2.009 | <b>2.002</b> | 0.103 | -0.257  | -0.076 | 0.016 |

Таблица вычислений левого крайнего корня нелинейного уравнения методом Ньютона с точностью  $\epsilon = 10^{-2}$  с удержанием 3 знаков после запятой.

Таблица 2

| № итерации | $x_k$    | $f(x_k)$ | $f'(x_k)$ | $x_{k+1}$       | $ x_k - x_{k+1} $ |
|------------|----------|----------|-----------|-----------------|-------------------|
| 1          | -3.6     | 13.92640 | -38.44600 | -3.23777        | 0.36223           |
| 2          | -3.23777 | 2.07945  | -27.20087 | -3.16132        | 0.07645           |
| 3          | -3.16132 | 0.08420  | -25.00871 | <b>-3.15795</b> | 0.00337           |

## Листинг программы:

```
def y(x): # Заданная функция
    return -1.8 * x ** 3 - 2.94 * x ** 2 + 10.37 * x + 5.38

def dy(x): # Производная функции
    return -1.8 * 3 * x ** 2 - 2.94 * 2 * x + 10.37

def bisection_method(a, b, e): # Метод половинного деления
    n = 0
    c = "Ошибка"
    while abs(a - b) > e:
        c = (a + b) / 2
        n = n+1
        if y(c) == 0:
            return "Найденный корень = " + str(c) + ", число итераций = " + str(n) + ", значение функции в найденном корне = " + str(y(c))
        if y(a) * y(c) < 0:
            b = c
        else:
            a = c
    if c == "Ошибка":
        return c
    else:
        return "Найденный корень = " + \
```

```
        str(c) + ", число итераций = " + str(n) + ", значение функции в  
найденном корне = " + str(y(c))
```

```
def newton_method(x0, e): # Метод Ньютона  
    n = 1  
    xn = x0 - y(x0) / dy(x0)  
    while abs(x0 - xn) > e or y(xn) > e:  
        x0 = xn  
        xn = x0 - y(x0) / dy(x0)  
        n = n+1  
    return "Найденный корень = " + \  
        str(xn) + ", число итераций = " + str(n) + ", значение функции в  
найденном корне = " + str(y(xn))
```

```
x = np.arange(-5, 5, step = 0.001) # Вывод графика  
plt.plot(x, y(x))  
plt.axhline(0, color='black', linewidth=0.3)  
plt.axvline(0, color='black', linewidth=0.3)  
plt.savefig("Graph.png")
```

```
while True: # Обеспечение интерактивности  
    print("Введите 1 для метода половинного деления, 2 для метода ньютона или  
exit для выхода")  
    choice = input()  
    if choice == "exit":  
        break  
    print("Введите 1 для ввода из консоли или 2 для ввода из файла")  
    inp = input()  
    print("Введите 1 для вывода в консоль или 2 для вывода в файл")  
    outp = input()  
    if choice == str(1):  
        if inp == str("1"):  
            print("Введите начало интервала")  
            result = False  
            while not result:  
                try:  
                    a = float(input())  
                except ValueError:  
                    print("Вы ввели не число")  
            else:  
                result = True  
            print("Введите конец интервала")  
            result = False  
            while not result:  
                try:  
                    b = float(input())  
                except ValueError:  
                    print("Вы ввели не число")  
            else:  
                result = True  
            print("Введите точность вычислений")  
            result = False  
            while not result:  
                try:  
                    e = float(input())  
                except ValueError:  
                    print("Вы ввели не число")  
            else:
```

```

        result = True
if y(a)*y(b) < 0:
    if outp == str(1):
        print(bisection_method(a, b, e))
    if outp == str(2):
        print("Введите путь к файлу для вывода")
        try:
            with open(input(), 'w') as f:
                f.write(bisection_method(a, b, e))
        except OSError:
            print("Некорректный путь к файлу")
        else:
            result = True
else:
    print("Некорректный интервал!")

if inp == "2":
    result = False
    while not result:
        print("Введите путь к файлу для ввода")
        try:
            with open(input(), 'r') as f:
                string = [float(v) for v in f.read().split(' ')]
        except OSError:
            print("Некорректный путь к файлу")
        else:
            result = True
if y(string[0])*y(string[1]) < 0:
    if outp == str(1):
        print(bisection_method(string[0], string[1], string[2]))
    if outp == str(2):
        print("Введите путь к файлу для вывода")
        try:
            with open(input(), 'w') as f:
                f.write(bisection_method(string[0], string[1],
string[2]))
        except OSError:
            print("Некорректный путь к файлу")
        else:
            result = True
else:
    print("Некорректный интервал!")

if choice == str(2):
    if inp == str(1):
        print("Введите начальное приближение")
        result = False
        while not result:
            try:
                x0 = float(input())
            except ValueError:
                print("Вы ввели не число")
            else:
                result = True
    print("Введите точность вычислений")
    result = False
    while not result:
        try:
            e = float(input())
        except ValueError:
            print("Вы ввели не число")

```

```

        else:
            result = True
    if outp == str(1):
        print(newton_method(x0, e))
    if outp == str(2):
        print("Введите путь к файлу для вывода")
        try:
            with open(input(), 'w') as f:
                f.write(newton_method(x0, e))
        except OSError:
            print("Некорректный путь к файлу")
        else:
            result = True
    if inp == str(2):
        result = False
    while not result:
        print("Введите путь к файлу для ввода")
        try:
            with open(input(), 'r') as f:
                string = [float(v) for v in f.read().split(' ')]
        except OSError:
            print("Некорректный путь к файлу")
        else:
            result = True
    if outp == str(1):
        print(newton_method(string[0], string[1]))
    if outp == str(2):
        print("Введите путь к файлу для вывода")
        try:
            with open(input(), 'w') as f:
                f.write(newton_method(string[0], string[1]))
        except OSError:
            print("Некорректный путь к файлу")
        else:
            result = True

```

## Результаты выполнения программы:

Введите 1 для метода половинного деления, 2 для метода ньютона или exit для выхода

1

Введите 1 для ввода из консоли или 2 для ввода из файла

1

Введите 1 для вывода в консоль или 2 для вывода в файл

1

Введите начало интервала

-1

Введите конец интервала

0

Введите точность вычислений

0.001

Найденный корень = **-0.4736328125**, число итераций = 10, значение функции в найденном корне = 0.00015171144157655192

Введите 1 для метода половинного деления, 2 для метода ньютона или exit для выхода

2

Введите 1 для ввода из консоли или 2 для ввода из файла

1

Введите 1 для вывода в консоль или 2 для вывода в файл

1

55

[illegible]

Введите 1 для метода половинного деления, 2 для метода ньютона или exit для выхода

1

1

1

-4

-2

0.0000000000000001

Введите 1 для метода половинного деления, 2 для метода ньютона или exit для выхода

```
exit
```

График выводится в файл Graph.png (см. Порядок выполнения работы п.1)

Методом бисекции:

-0.4736452102661133

1.9982580184936523

-3.1579461190276503

1.9982583005213423

**Вывод:** В ходе лабораторной работы я научился программно реализовывать решение нелинейных уравнений методом половинного деления и методом касательных.