ГУАП

КАФЕДРА № 44

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

доцент, к.т.н., доцент		А.А.Сенцов
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

ОДНОМЕРНАЯ ЛОКАЛЬНАЯ ОПТИМИЗАЦИЯ

по курсу: МЕТОДЫ ОПТИМИЗАЦИИ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	4242М		М.В.Климов
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2023

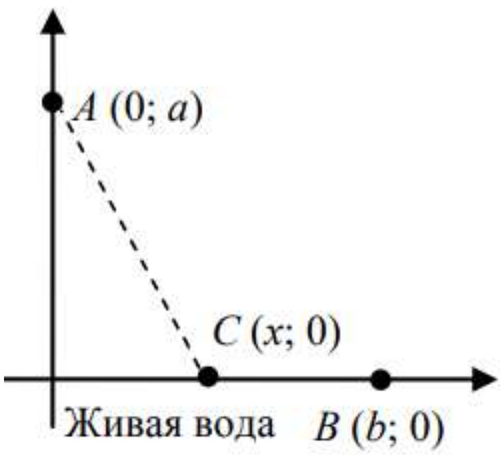
1. Цель работы и вариант

Ознакомиться с известными аналитическими и численными методами одномерной локальной оптимизации, получить опыт решения одномерной оптимизационной задачи, приобрести навыки использования математических программных средств для решения одномерных экстремальных задач.

Вариант 7.

Вариант	Задача	Метод	a	b
7	Б	g	7	15

Задача Б. Серый Волк находится на лугу в точке А и хочет оживить Ивана-Царевича, находящегося на берегу реки с живой водой в точке В. Скорость Волка при движении по лугу в 2 раза меньше, чем при движении по берегу реки. Как Волку быстрее всего добраться до Ивана?



g) метод дихотомии (вариант 2)

2. Вычисления:

Первым делом требуется определить целевую функцию.

В данном случае, расстояние от начальной точки до конечной можно вычислить по формуле:

S = S1 + S2 = $\sqrt{a^2 + x^2} + (b - x)$

Однако при вычислении времени, второе слагаемое уравнения будет влиять на всю формулу в 2 раза меньше за счет того, что скорость на втором промежутке в 2 раза выше. Следовательно, целевую функцию можно представить в виде:

F(x) = $\sqrt{a^2 + x^2} + \frac{1}{2}(b - x)$

Для нахождения экстремума аналитическим методом, требуется найти такое значение x, при котором производная от целевой функции будет нулевой.

F'(x) =($\sqrt{a^2 + x^2} + \frac{1}{2}(b - x)$)' = 0

$\frac{(a^2 + x^2)'}{2*\sqrt{a^2 + x^2}} - \frac{1}{2} = 0$

$\frac{2*x - \sqrt{a^2 + x^2}}{2*\sqrt{a^2 + x^2}} = 0$

$2 * x - \sqrt{a^2 + x^2} = 0$

$2 * x = \sqrt{a^2 + x^2}$

$4 * x^2 = \pm(a^2 + x^2)$

$x^2 = \frac{a^2}{3} => x = \pm \frac{a}{\sqrt{3}} => x \approx 4{,}04145188$

Значение целевой функции в данной точке будет равно:

F(4,04145)=$\sqrt{7^2 + (4{,}04145)^2} + \frac{1}{2}\big(15 - (4{,}04145)\big) \approx 13{,}5622$

Теперь требуется сравнить это значение со значениями на крайних точках заданного отрезка значений [0, 15]

F(0)=$\sqrt{7^2 + (0)^2} + \frac{1}{2}\big(15 - (0)\big) \approx 14.5$

F(15)=$\sqrt{7^2 + (15)^2} + \frac{1}{2}\big(15 - (15)\big) \approx 16.5529$

Учитывая, что значения на границах заданного отрезка больше найденного экстремума, следовательно, в точке 4,04145 находится минимум целевой функции.

Далее требуется найти минимум программным путём, используя метод дихотомии(2 вариант).

Для этого внесём в программу целевую функцию и разобьём отрезок значений на 4 равные части тремя точками. Посчитав значения функций в этих точках, определим минимальное значение и перейдём к такому же рассмотрению той половины, где центральное значение между двумя четвертями минимально. Данный цикл должен продолжаться до тех пор, пока величина рассматриваемого отрезка не будет меньше заданного числа. Код программы приведён в приложении 1, а результат выполнения представлен на рисунке 1.

Для удобства проверки правильности работы программы и вычислений, выводится так же подсчёт в начальных краевых точка функции и в центральных частях рассматриваемых отрезков на каждом прохождении цикла.

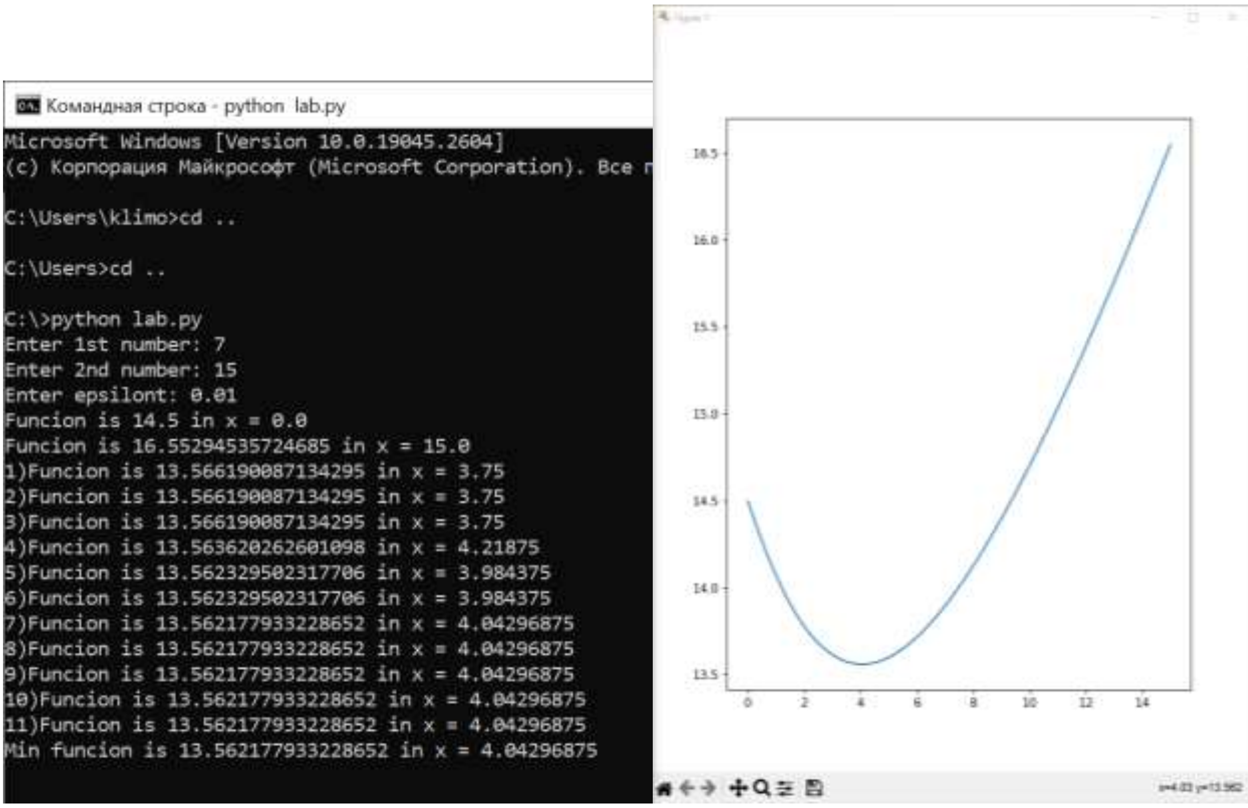


Рисунок 1. Результат выполнения программы.

3. Выводы.

В процессе выполнения лабораторной работы была решена поставленная задача по нахождению минимума двумя способами: аналитическим и программным (Методом дихотомии 2 варианта). В обоих случаях был получен одинаковый ответ, следовательно, при учёте правильности составления целевой функции, оба получившихся решений были правильными.

ПРИЛОЖЕНИЕ 1

Содержимое программы

```
import numpy as np
import matplotlib.pyplot as plt
import math;
def s(a, b, x):
    return (((a ** 2) + (x ** 2)) ** (1/2)) + (b - x)/2

a = float(input('Enter 1st number: '))
b1 = float(0)
b = float(input('Enter 2nd number: '))
e = float(input('Enter epsilont: '))
n = 0
b12 = b1
b2 = b
mini = s(a, b, b1)
print(f'Funcion is {mini} in x = {b1}')
mini = s(a, b, b)
print(f'Funcion is {mini} in x = {b}')
while (b-b1)>e:
    n = n + 1
    c = (b1+b)/2
    x1 = (b1 + c)/2
    x2 = (c + b)/2
    if s(a,b2,c)>s(a,b2,x1):
        b = c
    else:
        if s(a,b2,c)>s(a,b2,x2):
            b1 = c
```

```
        else:
            b1 = x1
            b = x2
    x = (b1 + b)/2
    mini = s(a, b2, (b1 + b)/2)
    print(f'{n})Funcion is {mini} in x = {x}')

print(f'Min funcion is {mini} in x = {x}')


xx=np.arange(b12,b2,e)
plt.plot(xx, s(a, b2, xx))
     plt.show()
```

ПРИЛОЖЕНИЕ 2

Вопросы:

1. Классическое вариационное исчисление, теория и примеры.

Вариационное исчисление – это аппарат для нахождения наилучших в определённом смысле функций. Математическая формулировка задачи вариационного исчисления имеет вид:

J(f(x)) –> min, при f(x) принадлежащем некоторому пространству функций (например – гладкие функции, или непрерывные функции и т.д.),

f(x) – представляет собой вектор функций, в простейшем случае f(x) – одна функция одного аргумента.

Функцию J(f(x)) называют функционалом. Функционал ставит каждому вектору функций f(x) из его области определения некоторое число.

Классическим примером задачи на вариационное исчисление(в данном случае, безусловное вариационное исчисление) является задача Бернулли о брахистохроне: «В вертикальной плоскости даны две точки А и В. Определить путь, спускаясь по которому под действием собственной тяжести тело М, начав двигаться из точки А, достигнет точки В за кратчайшее время.»

[Аналитические МО, стр 65]

2. Описать метод покоординатного спуска.

Метод покоординатного спуска является методом многомерной оптимизации, при котором за начальное значение берётся случайное значение одной из переменных и для этого значения переменной находится вторая переменная таким образом, чтобы значение было минимальным. Далее берётся найденное значение второй переменной и для неё находится значение первой переменной таким образом, чтобы значение функции было минимальным. Таким образом по кругу находятся значения для каждой переменной до тех пор, пока новые значения, найденные для переменных, не будут находится в заданных пределах от предыдущего значения.

Данный метод может применяться не только для функций, состоящих из двух переменны, но и для бОльших переменных. В этих случаях мы также будем фиксировать все переменные, кроме одной, при изменении которой мы и будем находить минимум, но также будут перебираться все переменные по кругу, пока изменение переменной не будет меньше, чем эпсилон. [Численные МО, стр 26]

3. Может ли пользователь ввести некорректные значения на вход программы? Привести примеры. Добавить в программу проверку на корректность ввода данных.

Возможные некорректности вводимых данных:

1) Отрицательное значение второй переменной.

   В данном случае цикл даже не активируется, а также на графике не задаётся координата x.

2) Отрицательное значение эпсилон.

   В этом случае цикл никогда не закончится. Но если его прервать, то некорректно может считаться координата x для графика (при разной отрицательности/положительности со второй переменной)

Для решения данных проблем, в программу была добавлена функция нахождения абсолютного значения для второй части функции, для условия цикла, а также для значения эпсилон. Для корректного получения координаты x для графика была добавлена проверка положительности второй переменной.

Полученная программа указана в Приложении 3

ПРИЛОЖЕНИЕ 3

Содержимое программы с изменением для учёта и корректировки некорректных вводимых данных (не подразумеваемых задачей).

```
import numpy as np
import matplotlib.pyplot as plt
import math;
def s(a, b, x):
    return (((a ** 2) + (x ** 2)) ** (1/2)) + abs(b - x)/2

a = float(input('Enter 1st number: '))
b1 = float(0)
b = float(input('Enter 2nd number: '))
e = float(input('Enter epsilont: '))
e = abs(e)
n = 0
b12 = b1
b2 = b
mini = s(a, b, b1)
print(f'Funcion is {mini} in x = {b1}')
mini = s(a, b, b)
print(f'Funcion is {mini} in x = {b}')
while abs(b-b1)>e:
    n = n + 1
    c = (b1+b)/2
    x1 = (b1 + c)/2
    x2 = (c + b)/2
    if s(a,b2,c)>s(a,b2,x1):
        b = c
    else:
```

```
        if s(a,b2,c)>s(a,b2,x2):
            b1 = c
        else:
            b1 = x1
            b = x2
    x = (b1 + b)/2
    mini = s(a, b2, (b1 + b)/2)
    print(f'{n})Funcion is {mini} in x = {x}')

print(f'Min funcion is {mini} in x = {x}')

if b2>0:
    xx=np.arange(b12,b2,e)
else:
    xx=np.arange(b12,b2,-e)
plt.plot(xx, s(a, b2, xx))
plt.show()
```