Университет ИТМО

Факультет программной инженерии и компьютерной техники

**Лабораторная работа №2**

по «Методам оптимизации»

Выполнил:

Студент группы P3230

Гаврилин Олег Сергеевич

Преподаватели:

Селина Елена Георгиевна

Санкт-Петербург

2025

# Задание

Для заданной функции (f(x)=0.5\*x^2 – sinx, [a, b]=[0, 1], Ɛ=0.0001 вычислить минимум методам деления пополам, методом золотого сечения, методом хорд и методом Ньютона

# Метод деления пополам

## Итерации:

Первая итерация

1)

2)

3)

4)

Вторая итерация

1)

2)

3)

4)

Третья итерация

1)

2)

3)

4)

## Код программы

def bisection\_method(a, b, eps):

while b - a > 2 \* eps:

x1 = (a + b - eps) / 2

x2 = (a + b + eps) / 2

y1 = f(x1)

y2 = f(x2)

if y1 > y2:

a = x1

else:

b = x2

x\_min = (a + b) / 2

y\_min = f(x\_min)

return x\_min, y\_min

Вывод: (0.7390813140869141, -0.40048861210117354)

# Метод золотого сечения

## Итерации:

Первая итерация

Вторая итерация

Третья итерация

## Код программы

def golden\_section\_method(a, b, eps):

x1 = a + 0.382 \* (b - a)

x2 = a + 0.618 \* (b - a)

y1 = f(x1)

y2 = f(x2)

i = 1

while b - a > 2 \* eps:

print("ИТЕРАЦИЯ", i)

print("a=", a, "b=", b)

print("x1 =", x1)

print("x2 =", x2)

print("y1 =", x1)

print("y2 =", x2)

if y1 < y2:

b = x2

x2 = x1

x1 = a + 0.382 \* (b - a)

y2 = y1

y1 = f(x1)

else:

a = x1

x1 = x2

x2 = a + 0.618 \* (b - a)

y1 = y2

y2 = f(x2)

print("b-a=", b-a)

i+=1

x\_min = (a + b) / 2

y\_min = f(x\_min)

return x\_min, y\_min

Вывод: (0.7390492620405681, -0.4004886110366319)

# Метод хорд

## Итерации:

Первая итерация

Проверка

Поскольку f’(a) = 0, тогда минимум находится в точке 0, метод прекращается.

Ответ: (xm = 0, ym = 0)

## Код программы

def chord\_method(a, b, eps):

f\_pr\_a = f\_pr(a)

f\_pr\_b = f\_pr(b)

y\_s = 10 \*\* 9

i = 1

while abs(y\_s) > eps:

print("ИТЕРАЦИЯ", i)

print("a=", a, "b=", b)

x\_s = a - (f\_pr\_a / (f\_pr\_a - f\_pr\_b)) \* (a - b)

y\_s = f\_pr(x\_s)

print("x с волной:", x\_s)

print("f(x с волной):", y\_s)

if y\_s > 0:

b = x\_s

f\_pr\_b = y\_s

else:

a = x\_s

f\_pr\_a = y\_s

i+=1

x\_min = x\_s

y\_min = f(x\_s)

return x\_min, y\_min

Вывод: (1.0298555826500022, -0.3269233709177817)

# Метод Ньютона

## Итерации:

Первая итерация

Вторая итерация

Третья итерация

## Код программы

def newtons\_method(a, b, eps):

x\_k = a

i=1

while abs(f\_pr(x\_k)) > eps:

print("ИТЕРАЦИЯ", i)

print("x\_"+str(i-1)+"=",x\_k)

x\_k = x\_k - (f\_pr(x\_k))/(f\_pr\_2(x\_k))

print("f'(x\_k) =", abs(f\_pr(x\_k)))

i+=1

x\_min = x\_k

y\_min = f(x\_k)

return x\_min, y\_min

Вывод: 1.0298665403159568, -0.326917728562354

# Вывод

Все методы показывают приблизительно одинаковый результат по сравнению с друг другом, а также по сравнению с ручным решением. Это говорит о том, что все они работают правильно и их можно использовать для нахождения экстремума функции.