Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИУК «Информатика и управление»**

КАФЕДРА **ИУК4 «Программное обеспечение ЭВМ, информационные технологии»**

Лабораторная работа №1

«Минимизация функции»

ДИСЦИПЛИНА: «Моделирование»

Выполнил: студент гр. ИУК4-72Б ______________ (Калашников А.С.)
(подпись) (Ф.И.О.)

Проверил: ______________ (Никитенко У.В.)
(подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:

- Оценка:

Калуга, 2023

Цель работы: сформировать практические навыки анализа возможностей построения и выделения наиболее важных свойств объектов моделей для моделирования и использования специализированных программных пакетов и библиотек для решения задачи минимизации функции и визуализации результатов решения.

Задачи: найти минимум функции, указанной в варианте предложенным методом, сравнить результаты, выдвинуть и обосновать гипотезу целесообразности использования того или иного метода в зависимости от предложенной задачи и ее вариаций, точности результата, трудоемкости, сложности алгоритма, сложности обоснования применимости метода, вычислительной эффективности алгоритма. Визуализировать результаты.

Вариант №6

№	**Выполняемые задачи**
6	1.6, 4.2, 5.6, 6.6

Задание №1.6

Методом Ньютона найти минимум и максимум унимодальной на отрезке [a, b] функции f(x) с точностью $\varepsilon = 10^{-6}$. Предусмотреть подсчет числа итераций, потребовавшихся для достижения заданной точности.

F(x)	**a**	**b**
2^x-ln(x)	**0.1**	**3**

```
x = 0.818215; f(x) = 1.963853; n = 4
```

Рис. 1. Результат работы алгоритма

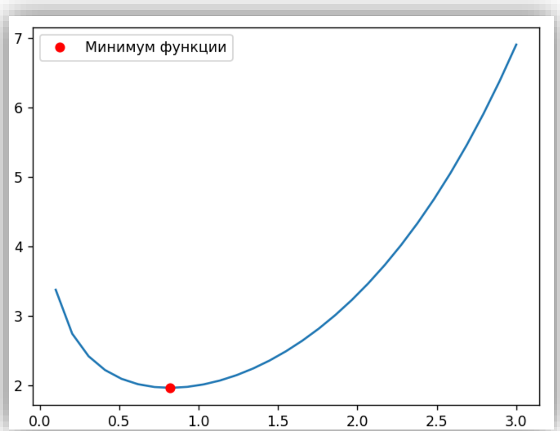

Рис. 2. График функции

Задание №4.2

Функция f(x) = (-1)^n*(cos(2nx)/n^3) представлена частичной суммой ряда. Построить график функции на заданном отрезке [0, 3] и найти ее минимумы и максимумы с указанной точностью 0,0001, n = 250, Методом минимизации деления отрезка пополам.

№	**u(x)**	**X1**	**X2**	**n**	$\mathscr{E}$	**Метод минимизации**
4.2	$-1^n \frac{\cos(2nx)}{n^3}$	0	3	250	0.0001	Деления отрезка пополам

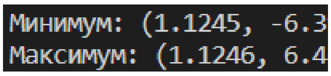

Рис. 3. Результат работы алгоритма

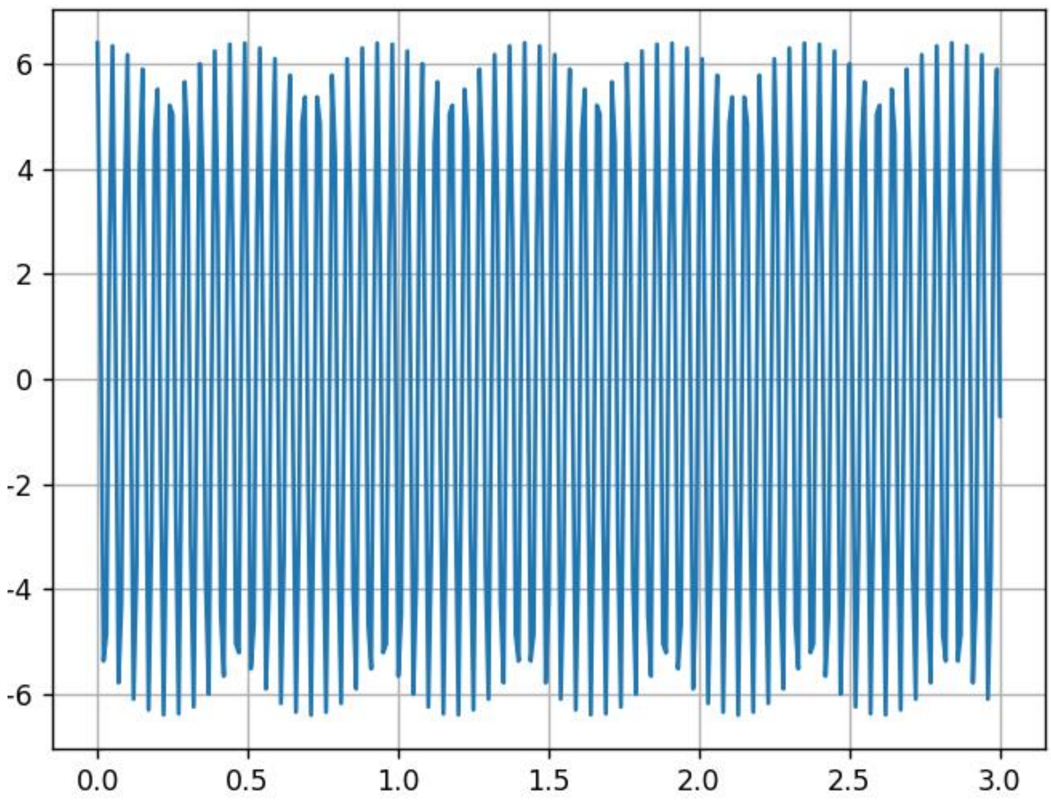

Рис. 4 – График функции

Задание №5.6

Найти минимум функции 2-х переменных f(x,y) с точностью $\varepsilon = 10^{-6}$ на прямоугольнике $[x_1, x_2] \times [y_1, y_2]$.

ПОРЯДОК РЕШЕНИЯ ЗАДАЧИ:

1. Задать указанную в варианте функцию f(x,y).

2. Построить графики функции и поверхностей уровня f(x,y).

3. По графикам найти точки начального приближения к точкам экстремума.

4. Найти экстремумы функции с заданной точностью.

№	**f(x)**	x_1	x_2	y_1	y_2
5.6	$4x^2 + y^2 + 3sin(x) - \cos(y+1)$	-2	0	-1	1

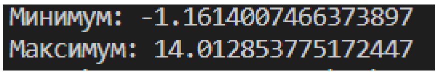

Рис. 6. Результат работы алгоритма

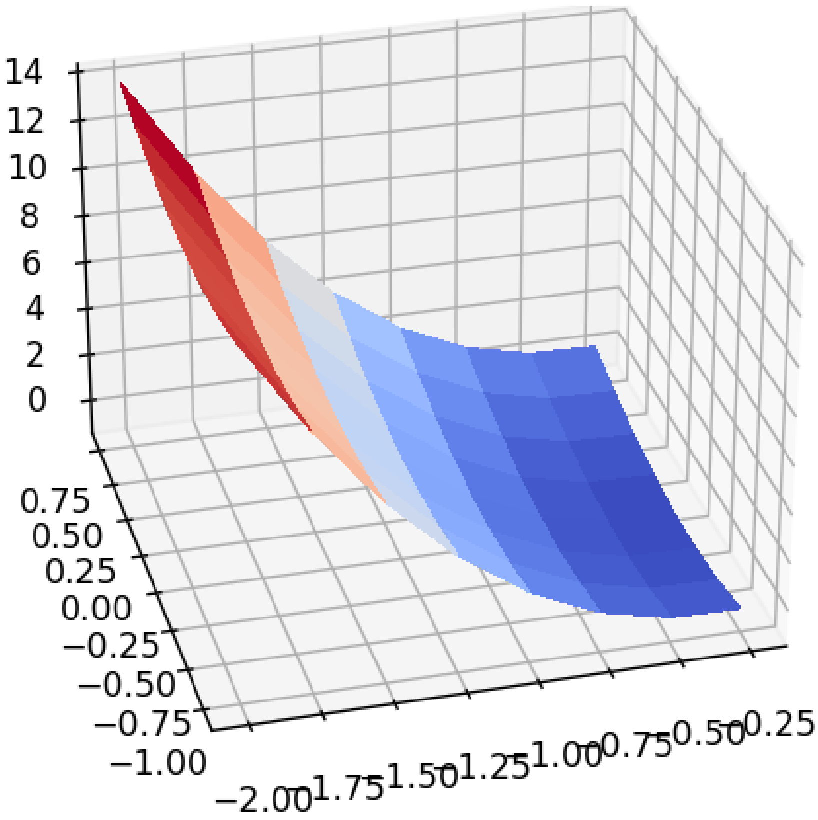

Рис. 6. – Вывод графика

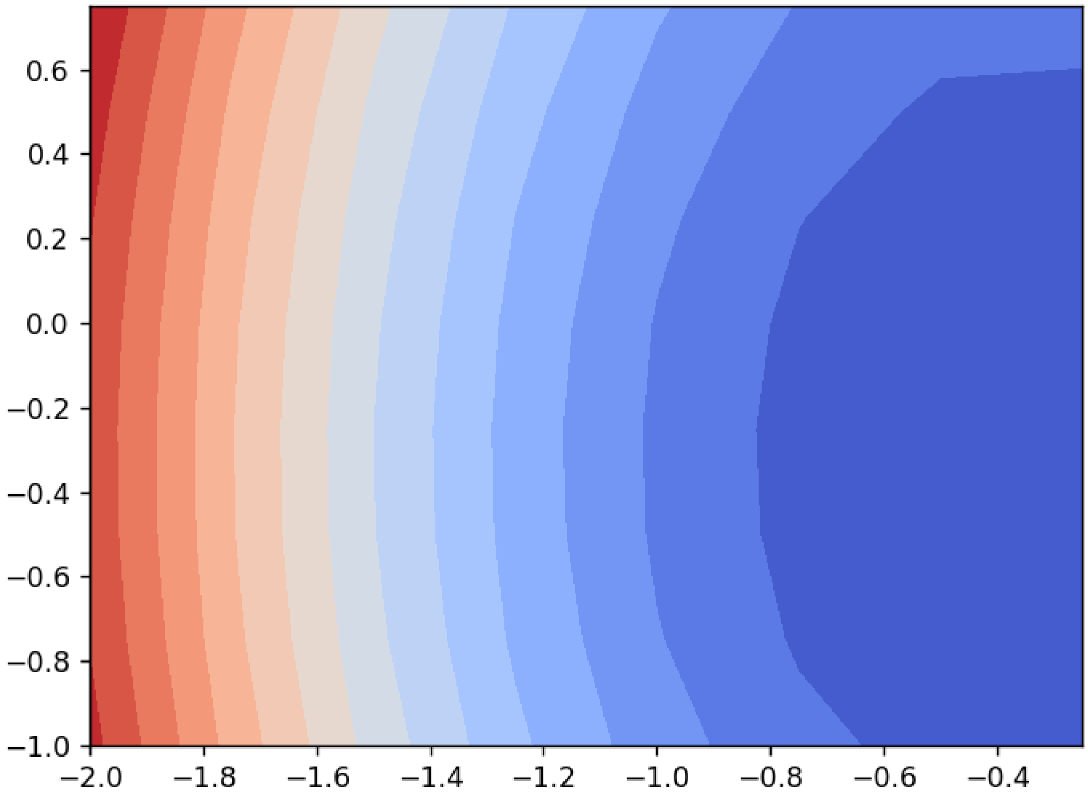

Рис. 6. – Вывод графика

Задание №6.6

Методом наискорейший спуск найти минимум квадратичной функции f(x,y)=a(11)x^2+2a(12)xy+a(22)y^2+2a(13)x+2a(23)y с точностью e=10^-6 Для решения задачи многомерной минимизации использовать метод Ньютона. Построить график функции f. Предусмотреть подсчет числа итераций, потребовавшихся для достижения заданной точности. a(11)=2.5, 2a(12)=1, a(22)=2, 2a(13)=-5, 2a(23)=-10.5

A11	2A12	A22	2A13	2A23	Метод
2.5	1	2	-5	-10.5	Наискорейший спуск

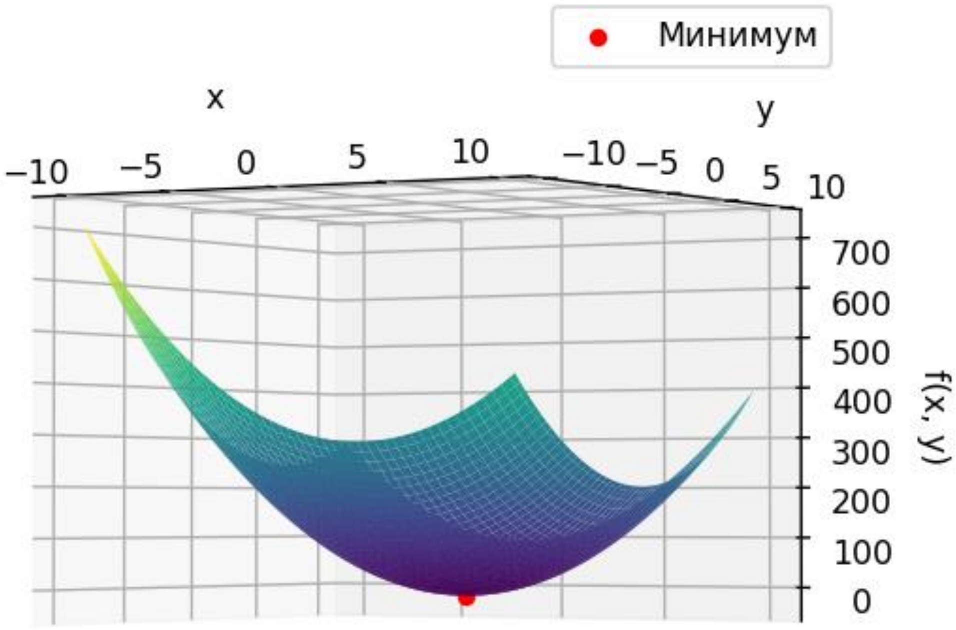

Рис. 7 – Вывод графика

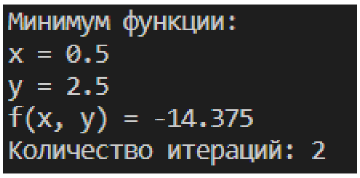
```
Минимум функции:
x = 0.5
y = 2.5
f(x, y) = -14.375
Количество итераций: 2
```

Рис. 8 – Результат работы алгоритма

Вывод: в ходе выполнения работы были сформированы практические навыки анализа возможностей построения и выделения наиболее важных свойств объектов моделей для моделирования и использования специализированных программных пакетов и библиотек для решения задачи минимизации функции и визуализации результатов решения

ПРИЛОЖЕНИЯ

Листинг программы

Ex. 1

```
import math, numpy
import matplotlib.pyplot as plt
f = lambda x: 2**x - numpy.log(x)
a = 0.1
b = 3
eps = 10**-6
df = lambda x: 2**x*numpy.log(2) - (1/x)
d2f = lambda x: 2**x*(numpy.log(2)**2) + (1/x**2)
x = (a + b) / 2
err = None
n = 0
while err == None or err > eps:
    if err == None:
        err = abs(df(x))
        continue
    err = abs(df(x))
    x = x - df(x)/d2f(x)
    n += 1
print(f"x = {x:.6f}; f(x) = {f(x):.6f}; n = {n}")
X = numpy.linspace(a, b, int((b-a)*10))
Y = [f(x) for x in X]
plt.plot(X, Y)
plt.plot(x, f(x), 'ro', label='Минимум функции')
plt.legend()
plt.show()
```

Ex. 2

```
import math
import matplotlib.pyplot as plt
# Определяем функцию
def f(x, n):
    return (-1)**n * (math.cos(2*n*x) / n**3)
# Определяем метод минимизации деления отрезка пополам
def bisection_method(f, a, b, eps):
    while (b - a) / 2 > eps:
        c = (a + b) / 2
        if f(c) == 0:
            return c
        elif f(c) * f(a) < 0:
            b = c
        else:
            a = c
    return (a + b) / 2
# Задаем параметры
n = 250
a = 0
b = 3
eps = 0.0001
# Строим график функции
x_values = [i / 100 for i in range(301)]
y_values = [f(x, n) for x in x_values]
plt.plot(x_values, y_values)
```

```
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Graph of f(x)')
plt.show()

# Ищем минимумы и максимумы функции
min_values = []
max_values = []
for i in range(1, 301):
    a = (i - 1) / 100
    b = i / 100
    if f(a, n) < f(b, n):
        min_value = bisection_method(lambda x: f(x, n), a, b, eps)
        min_values.append(min_value)
    elif f(a, n) > f(b, n):
        max_value = bisection_method(lambda x: -f(x, n), a, b, eps)
        max_values.append(max_value)

# Выводим результаты
print('Минимумы функции:', min_values)
print('Максимумы функции:', max_values)
```

Ex. 3

```
import matplotlib.pyplot as plt
from matplotlib import cm
from matplotlib.ticker import LinearLocator
import numpy as np
import math

x_1 = -2
x_2 = 0
y_1 = -1
y_2 = 1

fig, ax = plt.subplots(subplot_kw={"projection": "3d"})

X = np.arange(x_1, x_2, 0.25)
Y = np.arange(y_1, y_2, 0.25)
X, Y = np.meshgrid(X, Y)
Z = 4*X**2 + Y**2 + 3* np.sin(X)-np.cos(Y+1)

surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,
                       linewidth=0, antialiased=False)
plt.show()
levels = np.linspace(Z.min(), Z.max(), 15)
fig, ax = plt.subplots()
ax.contourf(X, Y, Z, levels=levels, cmap=cm.coolwarm, antialiased=False)
plt.show()
```

Ex. 4

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def f(x, y):
    a11 = 2.5
```

```
    a12 = 0.5
    a22 = 2
    a13 = -2.5
    a23 = -5.25

    return a11 * x**2 + 2 * a12 * x * y + a22 * y**2 + 2 * a13 * x + 2 * a23
* y

def grad(x, y):
    a11 = 2.5
    a12 = 0.5
    a22 = 2
    a13 = -2.5
    a23 = -5.25

    grad_x = 2 * a11 * x + 2 * a12 * y + 2 * a13
    grad_y = 2 * a12 * x + 2 * a22 * y + 2 * a23

    return np.array([grad_x, grad_y])

def hessian(x, y):
    a11 = 2.5
    a12 = 0.5
    a22 = 2

    hess_xx = 2 * a11
    hess_xy = 2 * a12
    hess_yy = 2 * a22
    return np.array([[hess_xx, hess_xy], [hess_xy, hess_yy]])

def line_search(x, y, grad, direction):
    t = 1.0
    alpha = 0.5
    beta = 0.8

    while f(x + t * direction[0], y + t * direction[1]) > f(x, y) + alpha * t
* np.dot(grad, direction):
        t *= beta

    return t

def newton_method(x0, y0, epsilon):
    x_current = np.array([x0, y0])
    iteration = 0

    while True:
        iteration += 1

        grad_current = grad(x_current[0], x_current[1])
        hessian_current = hessian(x_current[0], x_current[1])

        direction = -np.linalg.inv(hessian_current) @ grad_current

        t = line_search(x_current[0], x_current[1], grad_current, direction)

        x_next = x_current + t * direction

        if np.linalg.norm(x_next - x_current) < epsilon:
            break
```

```
        x_current = x_next

    return x_next, iteration

# Начальная точка
x0 = 0
y0 = 0

# Точность
epsilon = 1e-6

# Запуск метода Ньютона
solution, iteration = newton_method(x0, y0, epsilon)

# Результат
x_min, y_min = solution
min_value = f(x_min, y_min)

print("Минимум функции:")
print("x =", x_min)
print("y =", y_min)
print("f(x, y) =", min_value)
print("Количество итераций:", iteration)

# Построение графика функции
x = np.linspace(-10, 10, 100)
y = np.linspace(-10, 10, 100)
X, Y = np.meshgrid(x, y)
Z = f(X, Y)

fig = plt.figure()
ax = plt.axes(projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis')

ax.scatter(x_min, y_min, min_value, color='red', label='Минимум')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('f(x, y)')
ax.legend()

plt.show()
```