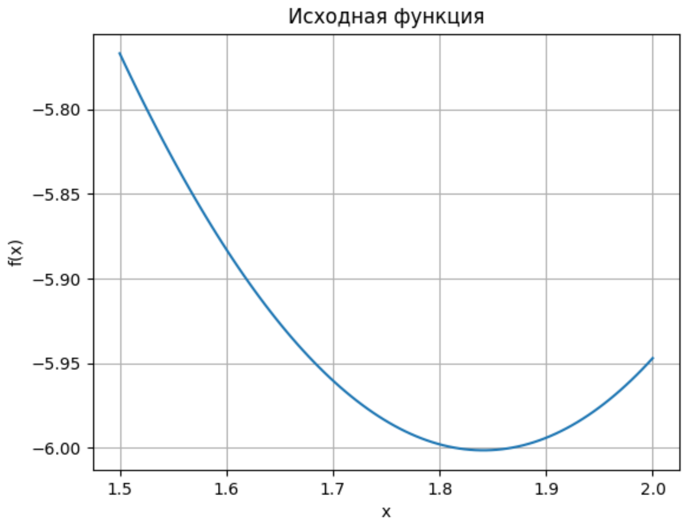
Практическое задание №1 по дисциплине «Методы оптимизации»

Выполнил: Маликов Глеб Игоревич, поток 3.5

Вариант № 9

Функция $f(x) = \frac{1}{3}x^3 - 5x + x\,ln\,x$, $[1{,}5; 2]$



Код

```
import math
import matplotlib.pyplot as plt
import numpy as np
import bisection, golden_section, newton


def f(x):
    return (1 / 3) * x ** 3 - 5 * x + x * math.log(x)


def f_prime(x):
    return x ** 2 - 4 + math.log(x)


def f_double_prime(x):
    return 2 * x + (1 / x)


def plot_results(f, a, b, results=None, color=None, title=None):
    x_values = np.linspace(a, b, 100)
    y_values = [f(x) for x in x_values]

    plt.plot(x_values, y_values)
    if results and color:
        plt.plot(results, [f(res) for res in results], color)
        plt.axvline(x=results[-1], color=color[0])

    plt.title(title if title else '')
    plt.xlabel('x')
    plt.ylabel('f(x)')
    plt.grid()
    plt.show()


if __name__ == '__main__':
    a = 1.5
    b = 2
    epsilon = 10 ** -10
```

```python
    delta = epsilon

    plot_results(f, a, b, None, None, "Исходная функция")

    # Метод половинного деления
    print("Метод половинного деления")
    result = bisection.bisection(f, a, b, delta, epsilon)
    print("Найденный минимум: ", result[-1], "\n")
    plot_results(f, a, b, result, 'bo', "Метод половинного деления")

    # Метод золотого сечения
    print("Метод золотого сечения")
    result = golden_section.golden_section(f, a, b, epsilon)
    print("Найденный минимум: ", result[-1], "\n")
    plot_results(f, a, b, result, 'go', "Метод золотого сечения")

    # Метод Ньютона
    print("Метод Ньютона")
    result = newton.newton(f_prime, f_double_prime, (a + b) / 2, epsilon)
    print("Найденный минимум: ", result[-1], "\n")
    plot_results(f, a, b, result, 'ro', "Метод Ньютона")
```

bisection.py

```python
# Поиск минимума функции методом деления отрезка пополам

def calculate_and_update(f, a, b, delta):
    x_1 = (b + a - delta) / 2
    x_2 = (b + a + delta) / 2
    f_1, f_2 = f(x_1), f(x_2)
    if f_1 <= f_2:
        b = x_2
    else:
        a = x_1
    return a, b


def bisection(f, a, b, delta, epsilon, max_steps=25):
    results = []
    i = 0
    print(f"i = {i}: a = {a}, b = {b}")

    for i in range(1, max_steps + 1):
        if abs(b - a) <= 2 * epsilon:
            break
        a, b = calculate_and_update(f, a, b, delta)
        print(f"i = {i}: a = {a}, b = {b}")
        results.append((a + b) / 2)
    return results
```

golden_section.py

```python
# Поиск минимума функции методом золотого сечения
import math


def golden_section(f, a, b, delta, max_steps=25):
    results = []
    tau = (math.sqrt(5) - 1) / 2
    i = 0
    print(f"i = {i}: a = {a}, b = {b}")
    x_1 = a + (1 - tau) * (b - a)
    x_2 = a + tau * (b - a)
    f_1, f_2 = f(x_1), f(x_2)

    for i in range(1, max_steps + 1):
```

```
        if abs(b - a) <= 2 * delta:
            break
        if f_1 <= f_2:
            b = x_2
            x_2, f_2 = x_1, f_1
            x_1 = b - (b - a) * tau
            f_1 = f(x_1)
        else:
            a = x_1
            x_1, f_1 = x_2, f_2
            x_2 = a + (b - a) * tau
            f_2 = f(x_2)

        print(f"i = {i}: a = {a}, b = {b}")
        results.append((a + b) / 2)

    return results
```

newton.py

```
# Поиск минимума функции методом Ньютона

def newton(f_prime, f_double_prime, x, epsilon, max_steps=25):
    results = []
    i = 0
    print(f"i = {i}: x = {x}")

    for i in range(1, max_steps + 1):
        if abs(f_prime(x)) <= epsilon:
            break
        x = x - f_prime(x) / f_double_prime(x)
        print(f"i = {i}: x = {x}")
        results.append(x)

    return results
```

Вывод программы

```
Метод половинного деления

i = 0: a = 1.5, b = 2

i = 1: a = 1.74999999995, b = 2

i = 2: a = 1.74999999995, b = 1.875000000025

i = 3: a = 1.8124999999375, b = 1.875000000025

i = 4: a = 1.8124999999375, b = 1.8437500000312501

i = 5: a = 1.8281249999343752, b = 1.8437500000312501

i = 6: a = 1.8359374999328126, b = 1.8437500000312501

i = 7: a = 1.8398437499320313, b = 1.8437500000312501

i = 8: a = 1.8398437499320313, b = 1.8417968750316407

i = 9: a = 1.8408203124318359, b = 1.8417968750316407

i = 10: a = 1.8408203124318359, b = 1.8413085937817382

i = 11: a = 1.841064453056787, b = 1.8413085937817382

i = 12: a = 1.841064453056787, b = 1.8411865234692626

i = 13: a = 1.841064453056787, b = 1.8411254883130248

i = 14: a = 1.8410949706349058, b = 1.8411254883130248
```

i = 15: a = 1.8410949706349058, b = 1.8411102295239652

i = 16: a = 1.8410949706349058, b = 1.8411026001294355

i = 17: a = 1.8410949706349058, b = 1.8410987854321705

i = 18: a = 1.8410949706349058, b = 1.8410968780835382

i = 19: a = 1.841095924309222, b = 1.8410968780835382

i = 20: a = 1.841095924309222, b = 1.84109640124638

i = 21: a = 1.8410961627278009, b = 1.84109640124638

i = 22: a = 1.8410961627278009, b = 1.8410962820370904

i = 23: a = 1.8410961627278009, b = 1.8410962224324456

i = 24: a = 1.8410961925301232, b = 1.8410962224324456

i = 25: a = 1.8410961925301232, b = 1.8410962075312844

Найденный минимум: 1.8410962000307038

Метод золотого сечения

i = 0: a = 1.5, b = 2

i = 1: a = 1.6909830056250525, b = 2

i = 2: a = 1.6909830056250525, b = 1.881966011250105

i = 3: a = 1.7639320225002102, b = 1.881966011250105

i = 4: a = 1.8090169943749475, b = 1.881966011250105

i = 5: a = 1.8090169943749475, b = 1.8541019662496845

i = 6: a = 1.8262379212492639, b = 1.8541019662496845

i = 7: a = 1.836881039375368, b = 1.8541019662496845

i = 8: a = 1.836881039375368, b = 1.847524157501472

i = 9: a = 1.836881039375368, b = 1.8434588481235803

i = 10: a = 1.8393935387456888, b = 1.8434588481235803

i = 11: a = 1.8393935387456888, b = 1.8419060381160095

i = 12: a = 1.8403532281084385, b = 1.8419060381160095

i = 13: a = 1.8403532281084385, b = 1.8413129174711884

i = 14: a = 1.8407197968263673, b = 1.8413129174711884

i = 15: a = 1.8409463487532598, b = 1.8413129174711884

i = 16: a = 1.8409463487532598, b = 1.841172900680152

i = 17: a = 1.841032883889116, b = 1.841172900680152

i = 18: a = 1.841032883889116, b = 1.841119419024972

i = 19: a = 1.8410659373697917, b = 1.841119419024972

i = 20: a = 1.841086365544296, b = 1.841119419024972

i = 21: a = 1.841086365544296, b = 1.8411067937188001

i = 22: a = 1.8410941684126285, b = 1.8411067937188001

i = 23: a = 1.8410941684126285, b = 1.8411019712809609

i = 24: a = 1.8410941684126285, b = 1.8410989908504678

i = 25: a = 1.8410960104199745, b = 1.8410989908504678

Найденный минимум: 1.8410975006352213

Метод Ньютона

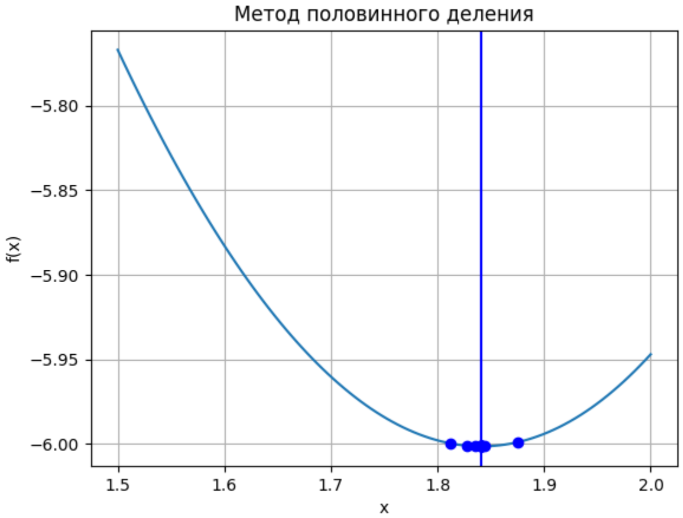
i = 0: x = 1.75

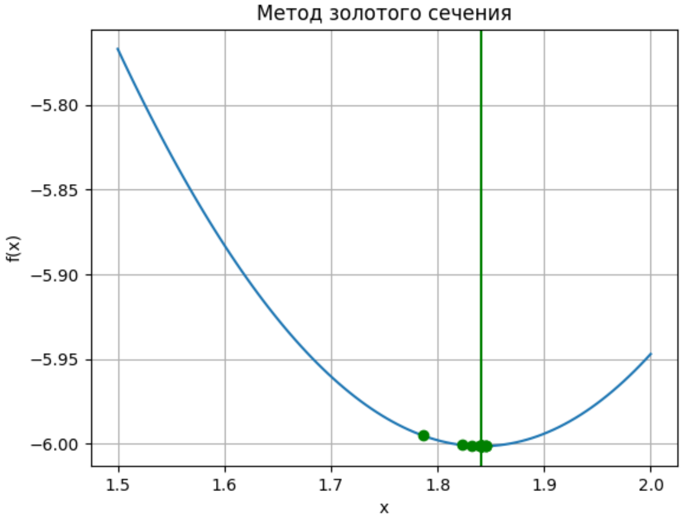
i = 1: x = 1.8428136661211243

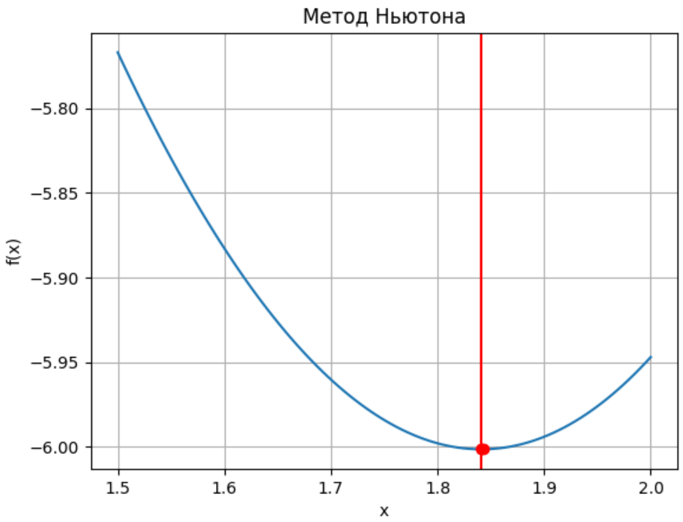
i = 2: x = 1.8410976526905583

i = 3: x = 1.84109705845015

Найденный минимум: 1.84109705845015






Метод Ньютона
−5.80
−5.85
−5.90
−5.95
−6.00
f(x)
1.5
1.6
1.7
1.8
1.9
2.0
x