Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

ОТЧЕТ

по лабораторной работе № 5

по курсу «Численные методы»

на тему: «Метод наискорейшего спуска поиска минимума функции многих переменных»

Вариант № 4

Студент	ИУ9-61Б (Группа)	(Подпись, дата)	Афанасьев И. (И. О. Фамилия)
Преподаватель		(Подпись, дата)	Домрачева А. Б. (И. О. Фамилия)

2024 г.

1 Постановка задачи

1. Найти минимум функции двух переменных с точностью $\varepsilon = 0.001$, начиная итерации с точки x^0.

2. Найтим минимум аналитичности.

3. Сравнить полученные результаты.

   Индивидуальный вариант: $f(x) = e^x + (x+y)^2$, $x^0 = (1, 1)$.

2 Основные теоретические сведения

Метод наискорейшего спуска является итерационным. Пусть для функции $f(x_1, \ldots, x_n)$ на k-м шаге имеем некоторое приближение к минимуму $x^k = (x_1^k, \ldots, x_n^k)$. Рассмотрим функцию одной переменной

$$\varphi_k(t) = f(x_1^k - t\frac{\partial f}{\partial x_1}(x^k), \ldots, x_n^k - t\frac{\partial f}{\partial x_n}(x^k) = f(x^k - t\mathrm{grad} f(x^k)).$$

Функция $\varphi_k(t)$ есть ограничение исходной функции $f(x)$ на прямую градиентного (наискорейшего) спуска, проходящую через точку k-го приближения x_k.

Минимум функции $\varphi_k(t)$ можно найти любым методом одномерной оптимизации. Обозначим эту точку минимума через t^*. Теперь для следующего приближения к точке экстремума полагаем

$$x^{k+1} = x^k - t^* \mathrm{grad} f(x^k) = (x_1^k - t^* \frac{\partial f}{\partial x_1}(x^k), \ldots, (x_n^k - t^* \frac{\partial f}{\partial x_n}(x^k).$$

Процесс поиска минимума продолжается до тех пор, пока $||\mathrm{grad} f(x^k)|| = \max_{1 \leq i \leq n} |\frac{\partial f}{\partial x_i}(x^k)|$ не станет меньше допустимой погрешности ε.

3 Реализация

В листинге 3.1 представлен исходный код программы на языке C++.

Листинг 3.1 – Исходный код программы на языке C++

```
#include <cmath>
#include <iostream>

namespace {

double F(const double x, const double y) {
  return std::exp(x) + std::pow(x + y, 2);
};

double DfDx(const double x, const double y) {
  return std::exp(x) + 2 * (x + y);
};

double D2fDx2(const double x, const double y) { return
    std::exp(x) + 2; };

double DfDy(const double x, const double y) { return 2 * (x +
    y); };

double D2fDy2(const double x, const double y) { return 2; };

double D2fDxDy(const double x, const double y) { return 2; };

}  // namespace

int main() {
  constexpr auto kEpsilon = 1e-3;

  auto x = 1.0;
  auto y = 1.0;

  do {
    const auto df_dx = DfDx(x, y);
    const auto df_dy = DfDy(x, y);

    if (std::max(df_dx, df_dy) <= kEpsilon) {
      break;
```

```
    }

    const auto sqr_df_dx = std::pow(df_dx, 2);
    const auto sqr_df_dy = std::pow(df_dy, 2);

    const auto dphi_dt = -sqr_df_dx - sqr_df_dy;
    const auto d2phi_dt2 = D2fDx2(x, y) * sqr_df_dx +
                           2 * D2fDxDy(x, y) * df_dx * df_dy +
                           D2fDy2(x, y) * sqr_df_dy;
    const auto t = -dphi_dt / d2phi_dt2;

    x -= t * df_dx;
    y -= t * df_dy;
  } while (true);

  const auto f_min = F(x, y);
  std::cout << "Function: exp(x) + (x + y)^2, (x0, y0) = (1.0, 1.0);\n"
            << "Calculated point of minimum: (" << x << ", " << y << ");\n"
            << "Calculated minimum: " << f_min << ";\n"
            << "Analytical minimum: tends to 0;\n"
            << "Error: " << f_min << ".\n";
}
```

4 Результаты

В листинге 4.1 представлены результаты работы программы.

Листинг 4.1 – Результаты работы программы

```
Function: exp(x) + (x + y)^2, (x0, y0) = (1.0, 1.0);
Calculated point of minimum: (-5.68689, 5.6857);
Calculated minimum: 0.00339154;
Analytical minimum: tends to 0;
Error: 0.00339154.
```