Министерство науки и высшего образования
Федеральное государтсвенное бюджетное образовательное учреждение высшего образования
Югорский государственный университет

Отчет о лабораторной работе №6
по дисциплине «Методы оптимизации»

Выполнил

Студент группы 1162б
_____ Панчишин И. Р.
«___» _____ 2019 г.

Принял

Доцент ИЦЭ
_____ Самарин В. А.
«___» _____ 2019 г.

Ханты-Мансийск, 2019

Цель

Изучить метод градиентного спуска для задач минимизации.

Задачи

1. Написать программную реализацию рассматриваемого метода.
2. Найти минимум функции, используя метод градиентного спуска.
3. Сравнить сходимость при различных значениях шага.

Ход работы

В листинге ниже представлена программная реализация нескольких типов метода градиентного спуска. Она также будет использована в следующих лабораторных работах на тему градиентного спуска.

```
addpath(../code)

set(0, defaultaxesfontsize, 14);
set(0, defaulttextfontsize, 14);


% исходные данные
F = @(X) 3*X(1)^2 - 3*X(1)*X(2) + X(2)^2 + 7*X(1) - 7*X(2);
X1 = X2 = linspace(-10, 10, 50);
[XX1, XX2] = meshgrid(X1, X2);

YY = [];
for i = 1:length(X1)
    Y = [];
    for j = 1:length(X2)
        Y = [Y, F([X1(i) X2(j)])];
    end
    YY = [YY; Y];
end

% вывод графика
surf(XX1, XX2, YY, edgecolor, none);
hold on
xlabel("x1");
ylabel("x2");
zlabel("y");


% поиск минимума
X0 = [1 -2];
[Xm ym] = fminunc(F, X0)
plot3(Xm(2), Xm(1), ym, b., MarkerSize, 40);

gradparams = struct(h, 1);
[Xm, ym, info] = graddesc(F, X0, 0.5, primal, gradparams)
plot3(Xm(2), Xm(1), ym, r., MarkerSize, 40);
plot3(info.Approx(:, 2), info.Approx(:, 1), info.Approx(:, 3), r, LineWidth, 3);


% количество вычислений в зависимости от масштаба (шага)
```

```
figure;

H = linspace(0.05, 0.8, 6);
for i = 1:length(H)
    subplot(2, 3, i);
    contour(XX1, XX2, YY, 20);
    hold on;

    gradparams = struct(h, H(i));
    [Xm, ym, info] = graddesc(F, X0, 0.5, primal, gradparams);
    plot(Xm(2), Xm(1), r., MarkerSize, 20);
    plot(info.Approx(:, 2), info.Approx(:, 1), r, LineWidth, 3);
    title([Масштаб  num2str(H(i)) , Шагов  num2str(info.nstep) , Вычислений  num2str(info.ncalc)])
end


pause
```

```
% Copyright © 2019 Panchishin Ivan

% градиентный спуск
% gradient descent

% type - тип спуска
% params - дополнительные параметры, которые зависят от типа спуска
% Approx - позволяет восстановить путь перемещения точки

function [Xm, ym, info] = graddesc(F, X0, e, type, params)
    y0 = F(X0);

    switch type
        case primal
            [Xm, ym, info] = graddesc_h(F, X0, y0, e, params.h);
        case hsearch
            [Xm, ym, info] = graddesc_h(F, X0, y0, e, NaN);
        case coord
            [Xm, ym, info] = graddesc_coord(F, X0, y0, e, params.a, params.b);
    end

    info.Approx = [X0, y0; info.Approx];
    ++info.ncalc;
end

function [X1, y1, info] = graddesc_h(F, X0, y0, e, h)
    [Gx0, ncalc_Gx0] = grad(F, X0);

    info.nstep = 1;
    info.ncalc = ncalc_Gx0;

    % поиск оптимального шага
    if isnan(h)
        fh = @(h) F(X0 - Gx0 * h);
        [hm, ym, info_hm] = gold(fh, 0.01, 2, e);
        info.ncalc += info_hm.ncalc;

        X1 = X0 - Gx0 * hm;
        y1 = ym;
    else
        while true
            % перемещение точки вдоль антиградиента
```

```
            X1 = X0 - Gx0 * h;

            y1 = F(X1);
            ++info.ncalc;
            if (y1 < y0)
                break;
            end

            h = h / 2;
        end
    end

    info.Approx = [X1, y1];

    % проверяем критерий останова (длина вектора)
    if (sqrt(sum(Gx0.^2)) <= e)
        return
    end

    [X1, y1, info1] = graddesc_h(F, X1, y1, e, h);

    info.nstep += info1.nstep;
    info.ncalc += info1.ncalc;
    info.Approx = [info.Approx; info1.Approx];
end

function [X1, ym, info] = graddesc_coord(F, X0, y0, e, a, b)
    info.nstep = 1;
    info.ncalc = 0;
    info.Approx = [];

    X1 = [];
    ym = NaN;
    for i = 1:length(X0)
        fx = @(x) F([X1, x, X0(i+1:end)]);
        [xm, ym, info_xm] = gold(fx, a, b, e);

        X1 = [X1, xm];

        info.ncalc += info_xm.ncalc;
        info.Approx = [info.Approx; [[X1, X0(i+1:end)], ym]];
    end

    if all(abs(X1 - X0) <= e)
        return
    end

    [X1, ym, info1] = graddesc_coord(F, X1, ym, e, a, b);

    info.nstep += info1.nstep;
    info.ncalc += info1.ncalc;
    info.Approx = [info.Approx; info1.Approx];
end
```

Результат поиска минимума функции из предыдущей лабораторной работы представлен на Рис. 1.

Количество вычислений функции при различном шаге отражено на Рис. 2. Каждый график представлен в виде набора уровней, где указана точка минимума, а также путь передвижения начального приближения.

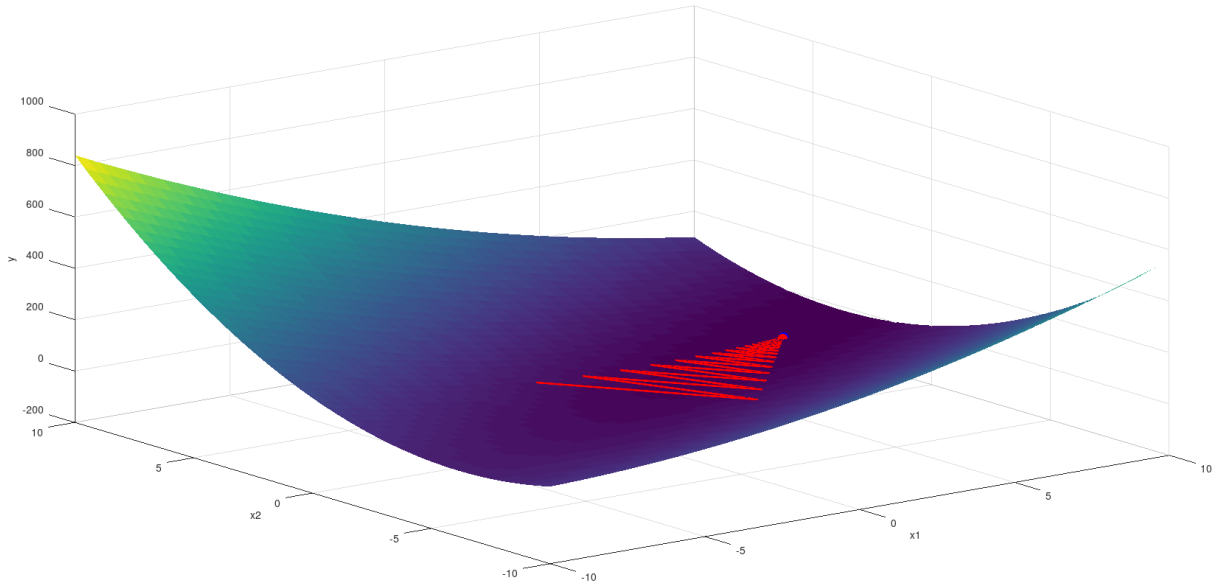

Рис. 1: Минимум функции

Вывод

Выполнил все поставленные задачи. Наилучшую сходимость обеспечивает наименьший шаг, но наилучшую скорость работы алгоритма обеспечило значение 0.2.

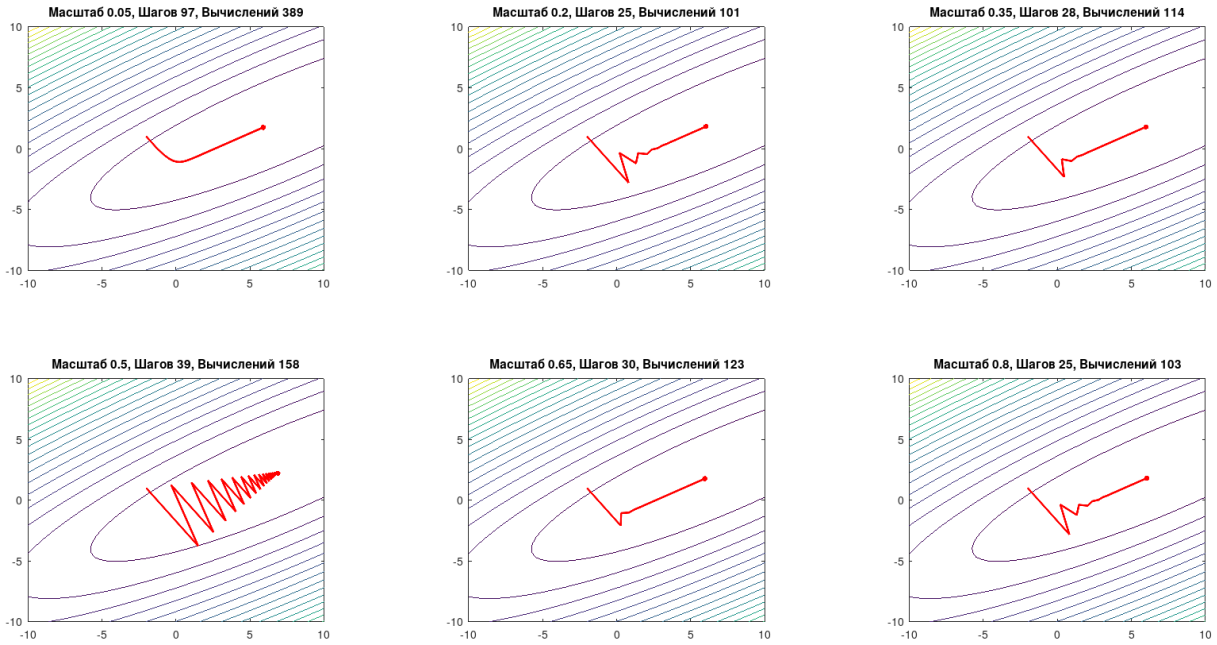

Рис. 2: Скорость работы при заданном шаге