Задание 1: **2.2. Метод штрафных функций.**  
c1 = - 1 ;

c2 = - 1 ;

a = 7 ;

b = 7 ;

R = 18 ;

d = 0.001; % дельта

% Начальная точка

x1 = 6;

x2 = 7 ;

k = 1 ; % Счетчик шагов

kmax = 100 ; % Предельное число шагов,

% задается для предотвращения зацикливания

% Массивы для хранения промежуточных координат

x1trace = [x1];

x2trace = [x2];

i = 2;

% Штрафная функция

g = @(x1 , x2 )(R - (x1 - a)^1 - (x2 - b)^2 );

% Производные

dfdx1 = @(x1, x2)(2\*c1\*x1) ;

dfdx2 = @(x1, x2)(2\*c2\*x2);

dgdx1 = @(x1, x2)(-2\*x1 + 2\*a);

dgdx2 = @(x1, x2)(-2\*x2 + 2\*b);

% Коэффициенты

gamma = 0.1;

alpha = 1.9 ;

while k < kmax;

% Вычисление шага по каждой координате

if g(x1, x2) > 0

step\_x1 = dfdx1(x1, x2);

step\_x2 = dfdx2(x1, x2);

else

step\_x1 = dfdx1(x1, x2) + alpha\*dgdx1(x1, x2);

step\_x2 = dfdx2(x1, x2) + alpha\*dgdx2(x1, x2);

end

% Вычисление новых координат

new\_x1 = max(0, x1 + gamma\*step\_x1);

new\_x2 = max(0, x2 + gamma\*step\_x2);

% Сохранение координат

x1trace( i) = new\_x1;

x2trace(i) = new\_x2;

i = i + 1;

x1 = new\_x1;

x2 = new\_x2;

% Проверка условия останова

if (dfdx1(x1, x2)/dgdx1(x1, x2)) - (dfdx2(x1, x2)/dgdx2(x1,x2)) <= d;

break; % Выход из цикла в случае выполнения условия

end;

k = k + 1;

end

% Построение графика

x = 0:0.1:10;

y = 0:0.1:10;

[X, Y] = meshgrid(x, y) ;

% Отображение допустимого множества

Zl = (X - a).^2 + (Y - b).^2 ;

[C, h] = contour(X, Y, Zl, [R R], 'LineWidth' , 2 );

hold on;

% Отображение линий уровня

Z = c1\*X.^2 + c2 \*Y.^2 ;

[C, h] = contour(X, Y, Z);

clabel (C, h); % Отображение меток на линиях уровня

% Отображение "траектории" алгоритма

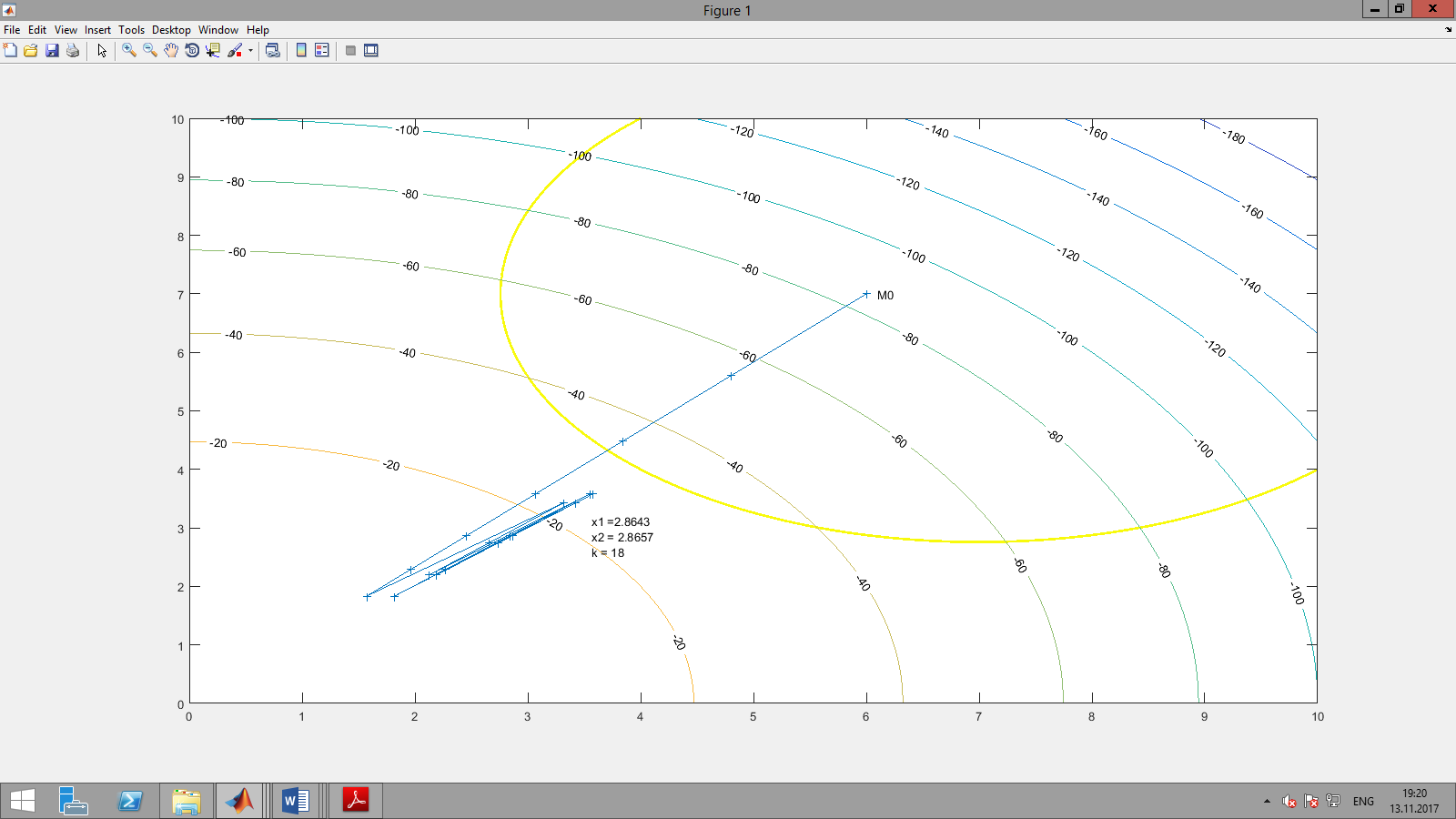
plot(x1trace, x2trace, '-+' );

% Вывод начальной точки на график

text (x1trace(1 ) + 0.1 , x2trace(1), 'M0');

% Вывод решения на график

text (x1 + 0.7 , x2 , strvcat( [ 'x1 =' num2str(x1)], [ 'x2 = ' num2str(x2)],[ 'k = ' num2str(k)]));



**2.3. Метод Эрроу-Гурвица**

c1 = - 1 ;

c2 = - 1 ;

a = 7 ;

b = 7 ;

R = 18 ;

d = 0.01; % дельта

% Начальная точка

x1 = 6;

x2 = 7 ;

k = 1 ; % Счетчик шагов

kmax = 100 ; % Предельное число шагов,

% задается для предотвращения зацикливания

% Массивы для хранения промежуточных координат

x1trace = [x1];

x2trace = [x2];

i = 2;

% Штрафная функция

g = @(x1 , x2 )(R - (x1 - a)^2 - (x2 - b )^2 );

% Производные

dfdx1 = @(x1, x2)(2\*c1\*x1) ;

dfdx2 = @(x1, x2)(2\*c2\*x2);

dgdx1 = @(x1, x2)(-2\*x1 + 2\*a);

dgdx2 = @(x1, x2)(-2\*x2 + 2\*b);

% Коэффициенты

gamma = 0.1;

alpha = 1.9 ;

while k < kmax;

% Вычисление шага по каждой координате

if g(x1, x2) > 0;

% Текущая точка принадлежит области допустимых решений

alpha = 0;

else

alpha = max(0, alpha - gamma\*g(x1, x2));

end

step\_x1 = dfdx1(x1, x2) + alpha\*dgdx1(x1, x2);

step\_x2 = dfdx2(x1, x2) + alpha\*dgdx2(x1, x2);

% Вычисление новых координат

new\_x1 = max(0, x1 + gamma\*step\_x1);

new\_x2 = max(0, x2 + gamma\*step\_x2);

% Сохранение координат

x1trace( i) = new\_x1;

x2trace(i) = new\_x2;

i = i + 1;

x1 = new\_x1;

x2 = new\_x2;

% Проверка условия останова

if (dfdx1(x1, x2)/dgdx1(x1, x2)) - (dfdx2(x1, x2)/dgdx2(x1,x2)) <= d;

break; % Выход из цикла в случае выполнения условия

end;

k = k + 1;

end

% Построение графика

x = 0:0.1:10;

y = 0:0.1:10;

[X, Y] = meshgrid(x, y) ;

% Отображение допустимого множества

Zl = (X - a).^2 + (Y - b).^2 ;

[C, h] = contour(X, Y, Zl, [R R], 'LineWidth' , 2 );

hold on;

% Отображение линий уровня

Z = c1\*X.^2 + c2 \*Y.^2 ;

[C, h] = contour(X, Y, Z);

clabel(C, h); % Отображение меток на линиях уровня

% Отображение "траектории" алгоритма

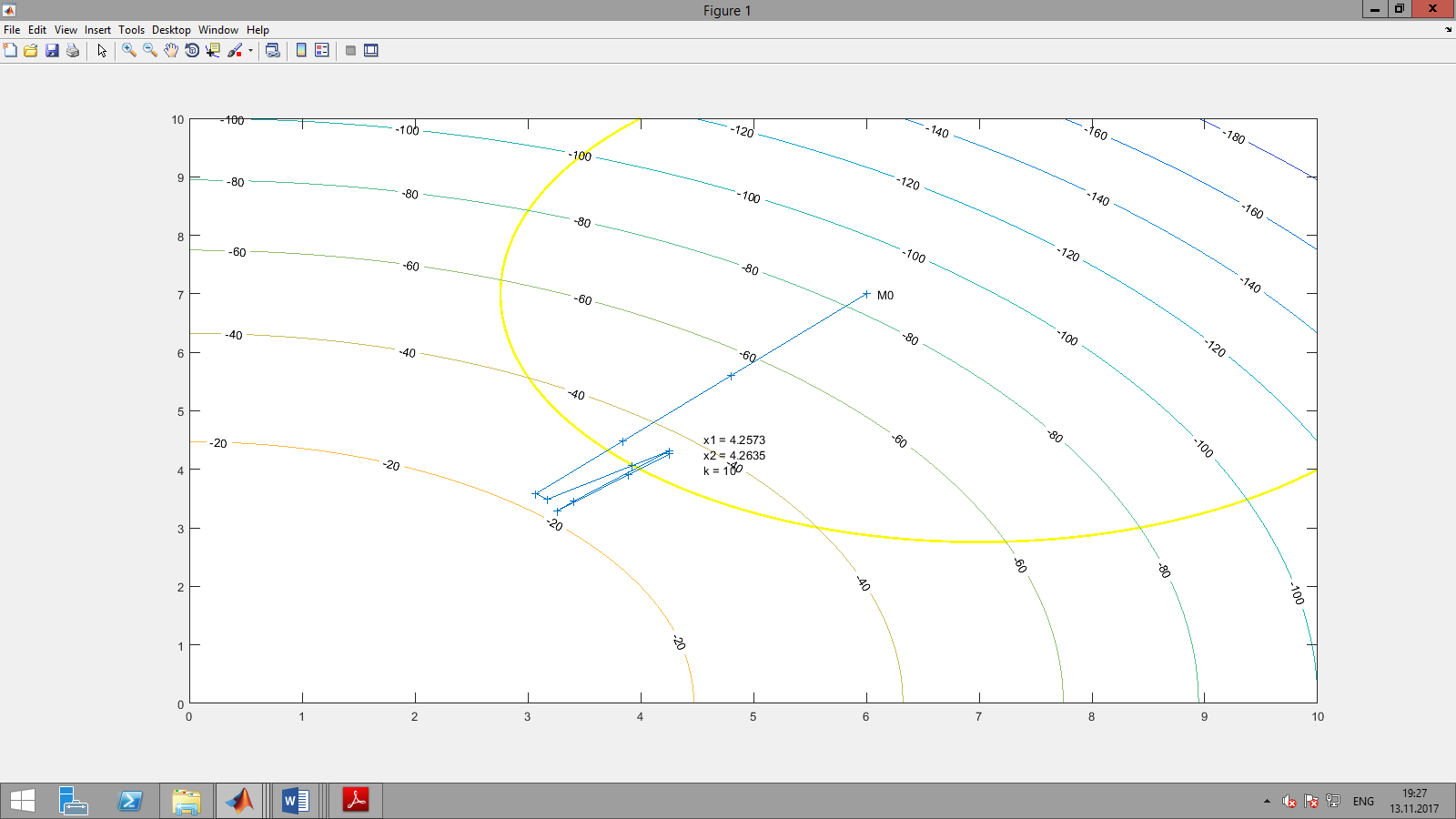
plot(x1trace, x2trace, ' - + ' );

% Вывод начальной точки на график

text (x1trace(1) + 0.1 , x2trace(1), 'M0 ');

% Вывод решения на график

text (x1 + 0.3 , x2 , strvcat ( [ 'x1 = ' num2str(x1)],[ 'x2 = ' num2str(x2)], [ 'k = ' num2str(k)]));



Задача 3  
c1 = -2;

c2 = - 1 ;

c12 = 1;

g = 0.2; % постоянная шага

d = 0.01; % дельта

% Начальная точка

x1 = 16 ;

x2 = 18;

k = 1 ; % Счетчик шагов

kmax = 100 ; % Предельное число шагов,

% задается для предотвращения зацикливания

% Массивы для хранения промежуточных координат

x1trace = [x1];

x2trace = [x2];

i = 2;

while k < kmax

% Спуск по первой координате

gr1 = 2\*c1\*x1 + c12\*x2;

x1 = x1 + g\*gr1;

% Сохранение координат

x1trace( i) = x1;

x2trace(i) = x2;

i = i + 1;

% Спуск по второй координате

gr2 = 2\*c2\*x2 + c12\*x1;

x2 = x2 + g\*gr2;

% Сохранение координат

x1trace( i) = x1;

x2trace(i) = x2;

i = i + 1;

% Проверка условия останова

if sqrt(gr1^2 + gr2^2) <= d;

break; % Выход из цикла в случае выполнения условия

end

k = k + 1;

end

% Построение графика

x = -20:0.1:20;

y = -20:0.1:20;

[X, Y] = meshgrid(x, y) ;

Z = c1\*X.^2 + c2 \*Y.^2 + c12\*X.\*Y;

[C, h] = contour(X, Y, Z);

clabel(C, h);

% Отображение меток на линиях уровня

hold on;

plot(x1trace, x2trace, ' - ' );

% Вывод начальной точки на график

text (x1trace(1 ) + 0.2 , x2trace(1) + 0.5 , 'M0 ' )

% Вывод решения на график

text(x1 + 2, x2,strvcat( [ 'x1 = ' num2str(x1)],[ 'x2 = ' num2str(x2)], [ 'k = ' num2str(k)]));

