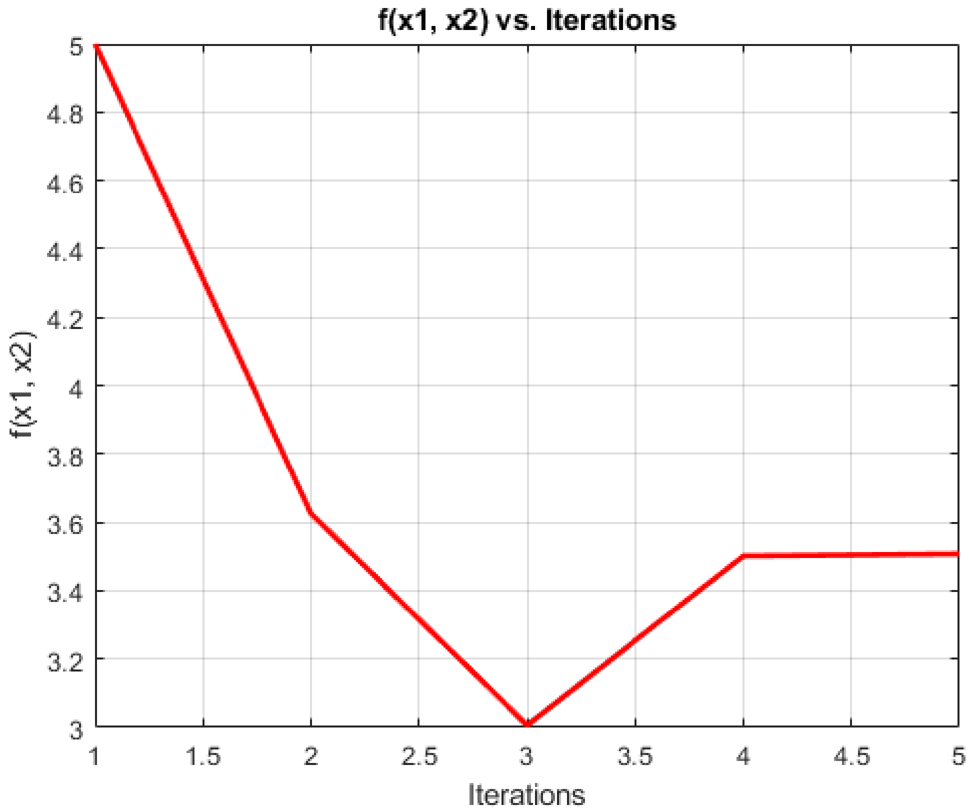
```
clc

f = @(x) x(1)^2+ (x(2)-3)^2;
df =@(x) [2*x(1), 2*x(2)-6];
g =@(x) [x(2)^2-2*x(1); (x(2)-1)^2+5*x(1)-15];
dg =@(x) [-2 2*x(2); 5 2*x(2)-2 ];
opt.alg = 'xe_1';
opt.linesearch = true;
opt.eps = 1e-3;
x0 = [1;1];

if max(g(x0)>0)
    errordlg('Infeasible intial point! You need to start from a feasible one!');
    return
end
solution = mysqp(f, df, g, dg, x0, opt);

for i = 1:length(solution.x)
    sol(i) = f(solution.x(:, i));
    con = g(solution.x(:, i));
    C_1(i) = con(1);
    C_2(i) = con(2);
end

count = 1:length(solution.x);
figure(1)
plot(count, sol,'r','LineWidth',2)
grid on
title('f(x1, x2) vs. Iterations')
xlabel('Iterations')
ylabel('f(x1, x2)')

figure(2)
hold on
plot(count, sol,'r','LineWidth',2)
plot(count, C_1,'LineWidth',1.5)
plot(count, C_2,'LineWidth',1.5)
grid on
legend('f(x) value', 'C1(x)', 'C2(x)')
title('Objective Function & Constraint Function vs Iterations')
xlabel('Iteration')
ylabel('Objective Function & Constraint Functions')
hold off

figure(3)
plot(solution.x(1, :), solution.x(2, :),'r','LineWidth',2)
grid on
title('Values of x2 vs. x1')
xlabel('x1')
ylabel('x2')

disp("x1 & x2 ");
disp(solution.x(:, end));
disp("F(x1, x2) = ");
disp(sol(end));
disp("C_1(x1, x2) = ");
disp(C_1(end));
disp("C_0(x1, x2) = ");
disp(C_2(end));
```

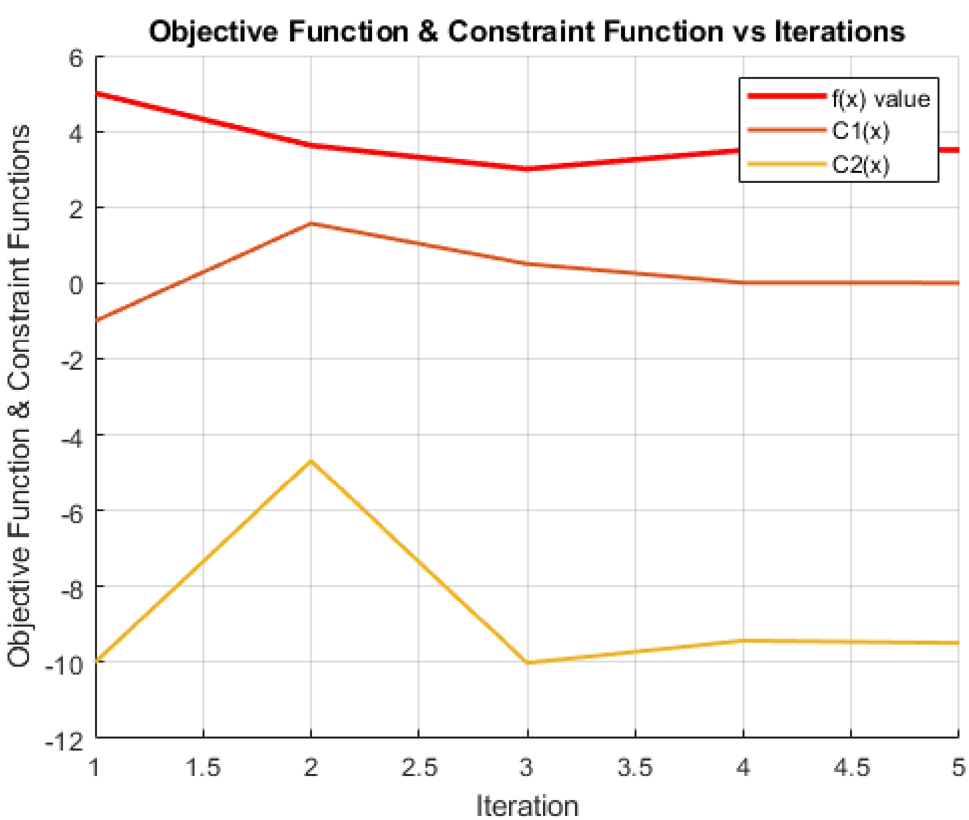
```matlab
function solution = mysqp(f, df, g, dg, x0, opt)
    x = x0;
    solution = struct('x',[]);
    solution.x = [solution.x, x];
    W = eye(numel(x));
    mu_old = zeros(size(g(x)));
    w = zeros(size(g(x)));
    gnorm = norm(df(x) + mu_old'*dg(x));
    while gnorm>opt.eps
        if strcmp(opt.alg, 'xe_1')
            [s, mu_new] = solveqp(x, W, df, g, dg);
        else
            qpalg = optimset('Algorithm', 'active-set', 'Display', 'off');
            [s,~,~,~,lambda] = quadprog(W,[df(x)]', dg(x), -g(x), [], [], [], [], x, qpalg);
            mu_new = lambda.ineqlin;
        end
        if opt.linesearch
            [a, w] = lineSearch(f, df, g, dg, x, s, mu_old, w);
        else
            a = 0.1;
        end
        dx = a*s;
        x = x + dx;
        y_k = [df(x) + mu_new'*dg(x) - df(x-dx) - mu_new'*dg(x-dx)]';
        if dx'*y_k >= 0.2*dx'*W*dx
            theta = 1;
        else
            theta = (0.8*dx'*W*dx)/(dx'*W*dx-dx'*y_k);
        end
        dg_k = theta*y_k + (1-theta)*W*dx;

        W = W + (dg_k*dg_k')/(dg_k'*dx) - ((W*dx)*(W*dx)')/(dx'*W*dx);
      gnorm = norm(df(x) + mu_new'*dg(x));
      mu_old = mu_new;
      solution.x = [solution.x, x];
    end
end

function [a, w] = lineSearch(f, df, g, dg, x, s, mu_old, w_old)
    t = 0.1;
    b = 0.8;
    a = 1;
    D = s;
    w = max(abs(mu_old), 0.5*(w_old+abs(mu_old)));
    count = 0;
    while count<100
        phi_a = f(x + a*D) + w'*abs(min(0, -g(x+a*D)));
        phi0 = f(x) + w'*abs(min(0, -g(x)));
        dphi0 = df(x)*D + w'*((dg(x)*D).*(g(x)>0));
        psi_a = phi0 +  t*a*dphi0;
        if phi_a<psi_a
            break;
        else
            a = a*b;
            count = count + 1;
        end
    end
end
```

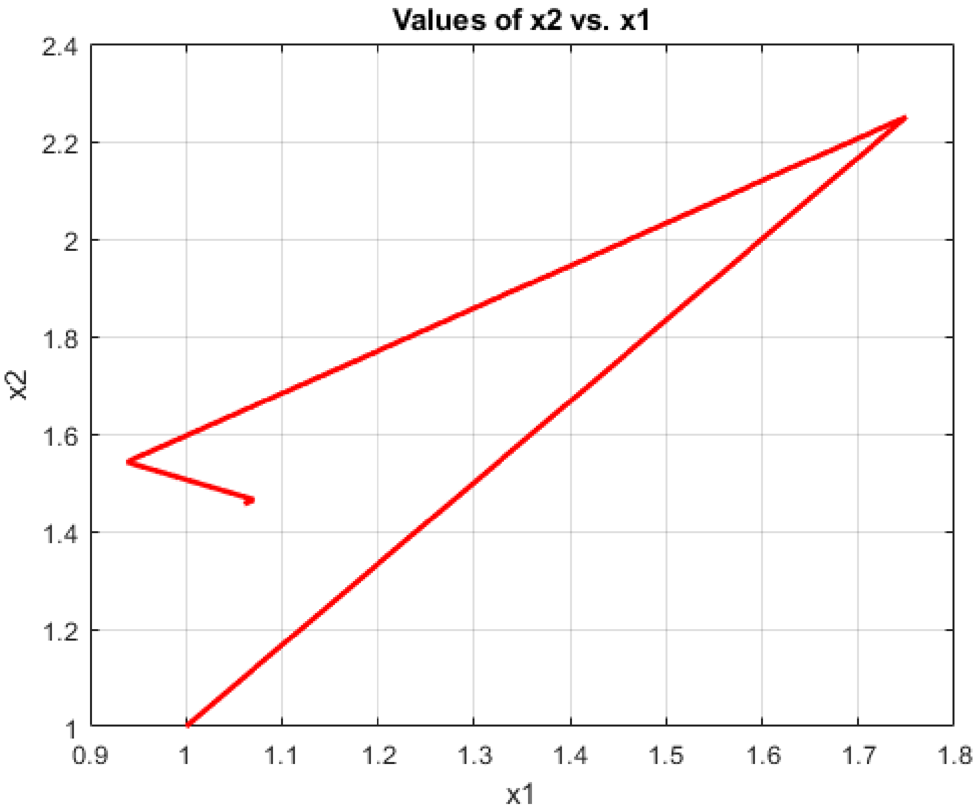
```
function [s, mu0] = solveqp(x, W, df, g, dg)
    c = [df(x)]';
    A0 = dg(x);
    b0 = -g(x);
    stop = 0;
    A = [];
    b = [];
    active = [];
    while ~stop
        mu0 = zeros(size(g(x)));
        A = A0(active,:);
        b = b0(active);
        [s, mu] = solve_activeset(x, W, c, A, b);
        mu = round(mu*1e12)/1e12;
        mu0(active) = mu;
        gcheck = A0*s-b0;
        gcheck = round(gcheck*1e12)/1e12;
        mucheck = 0;
        Iadd = [];
        Iremove = [];
        if (numel(mu) == 0)
            mucheck = 1;
        elseif min(mu) > 0
            mucheck = 1;
        else
            [~,Iremove] = min(mu);
        end
        if max(gcheck) <= 0

            if mucheck == 1
                stop = 1;
            end
        else
            [~,Iadd] = max(gcheck);
        end
        active = setdiff(active, active(Iremove));
        active = [active, Iadd];
        active = unique(active);
    end
end

function [s, mu] = solve_activeset(x, W, c, A, b)
    M = [W, A'; A, zeros(size(A,1))];
    U = [-c; b];
    sol = M\U;
    s = sol(1:numel(x));
    mu = sol(numel(x)+1:numel(sol));
end
```

```
x1 & x2
    1.0604
    1.4563

F(x1, x2) =
    3.5074

C_1(x1, x2) =
   7.9687e-05
```

```
C_0(x1, x2) =
   -9.4897
```







Published with MATLAB® R2020b