Problem 2

b)

The contour plot is shown in Figure 1.

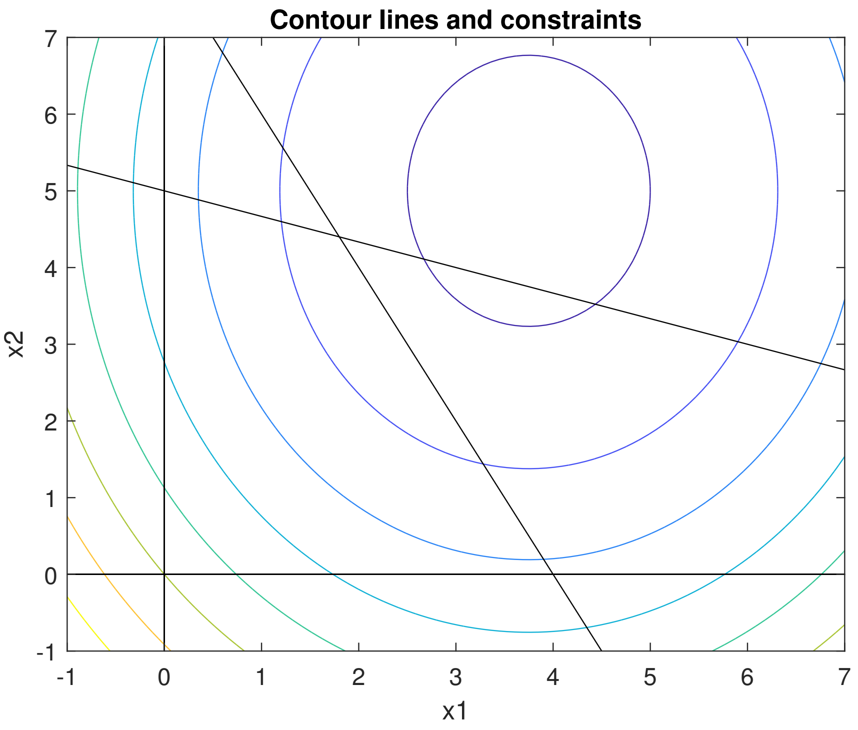


Figure 1: Contour plot of quadratic program

c)

The contour plot marked with points and iteration number is shown in Figure 2. This is provided by the code in Figure 3 and Figure 4.

d)

The first iteration we have the starting point $x_0 = [0, 0]^T$, with working set $W_0 = \emptyset$. Then, it will find the vector which points to the center of the contours. This is outside the constraints, and therefore a scaling factor will be applied to this vector, $x_1 = x_0 + \alpha p_1$, $\alpha > 1$. The next iteration the working set will containt the assumed active constraint $-2x_1 - x_2 + 8 \geq 0$, which means $W_1 = \{1\}$, now with starting point $x_1 = [2.4, 3.2]^T$. From here, the algorithm looks for

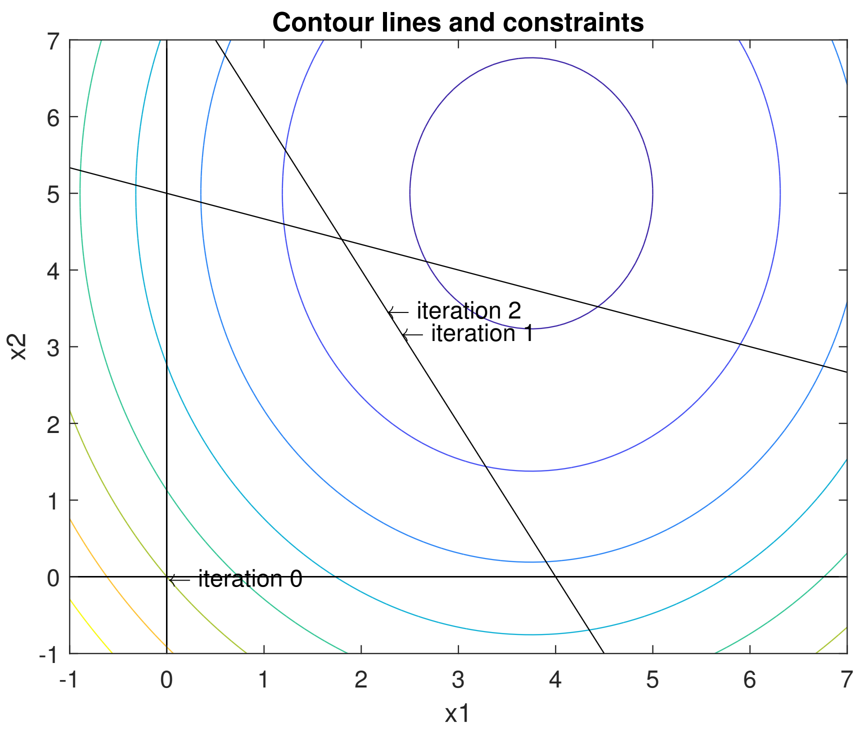


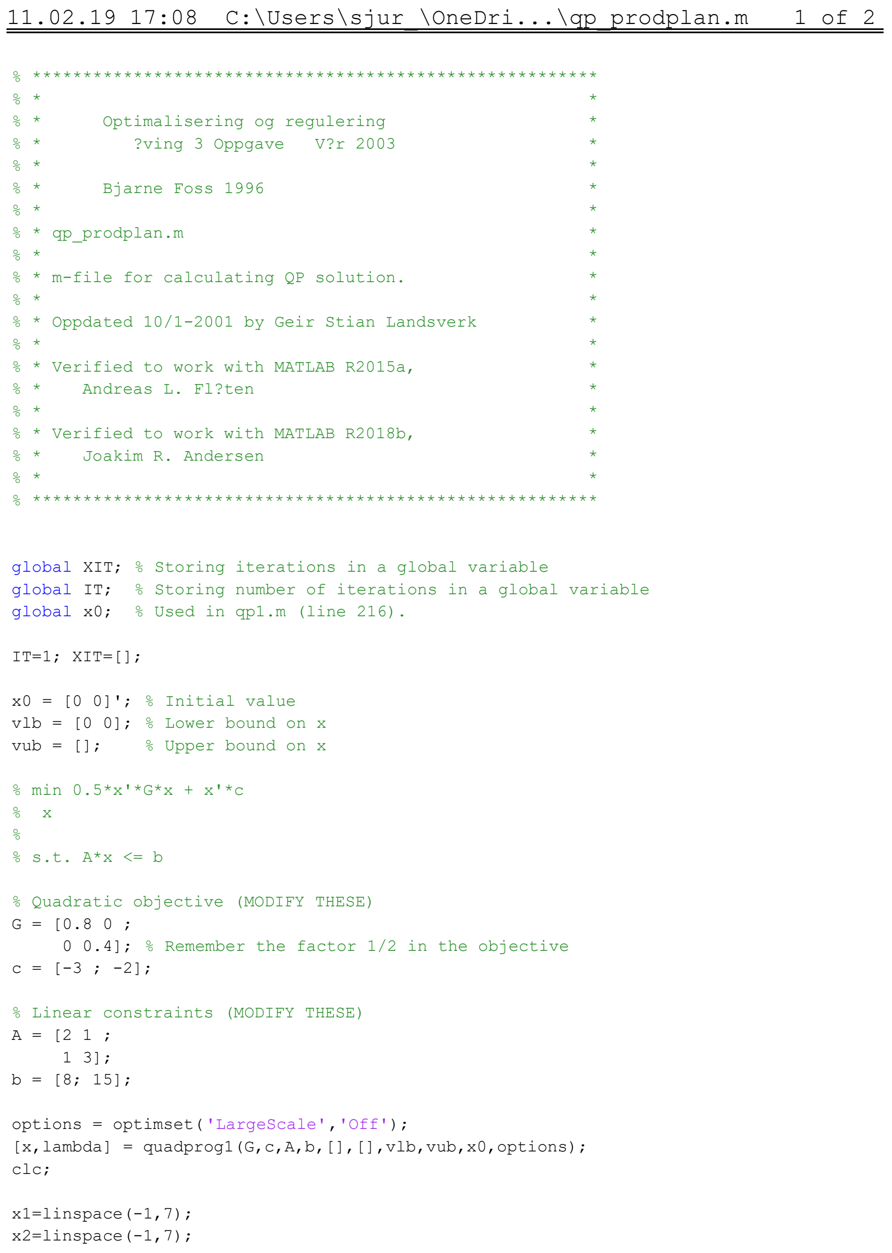
Figure 2: Contour plot with marked iterations

better solutions along the constraint, thus ending up in the point $x_2 = x_1 + p_2 = [2.25, 3.5]^T$. Nothing has changed with the working set. Next iteration returns the vector $p_3 = [0, 0]^T$, which means there is no better solution on this constraint, and since the lagrangian multiplier $\lambda > 0$, the KKT conditions are satisfied in our point, making it our global solution $x_2 = x^*$.

e)

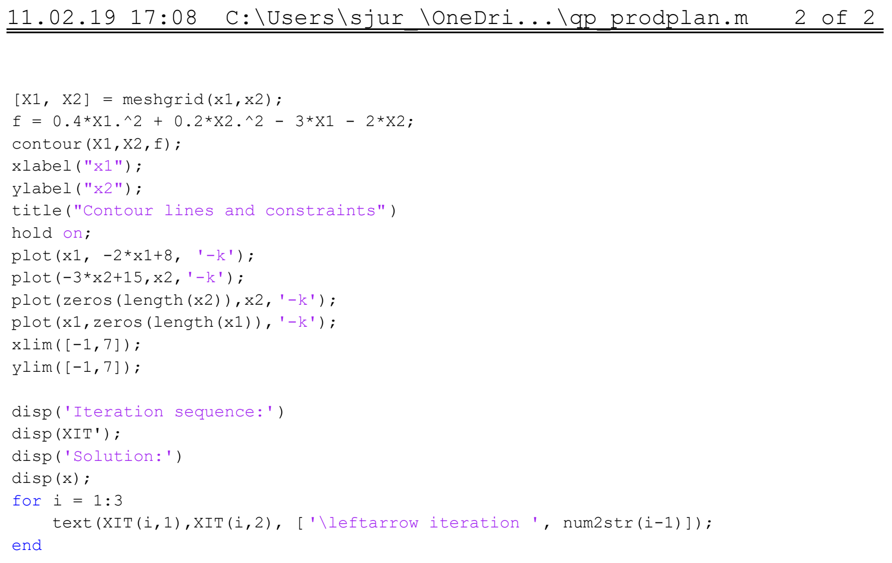
In problem 2c) in exercise 3 we executed the simplex method with a similar set of constraints. The largest difference is that the objective function is nonlinear in this case.
With the simplex method and a linear objective function, since the contour lines are linear, one is guaranteed to find the solutions in the corners of the polytope that spans the feasible set. This is because the function will always have it minimun value between two constraints (except if the contour is parallell with constraints, the function will here have its lowest value along the whole line, but this includes the corners:)).
In contrast, when the objective function is nonlinear, the curvature of the contour lines makes it harder to choose the set of active constraints. We here start with a feasible point and an active set, and systematically bring our guessed optimal point closer to the actual solution by seeing where the function is at its maximum along a constraint, then checking if we can minimize the function by removing one active constraint at a time without breaking the

constraints.



```
11.02.19 17:08  C:\Users\sjur_\OneDri...\qp_prodplan.m    1 of 2

% ********************************************************
% *                                                      *
% *      Optimalisering og regulering                    *
% *         ?ving 3 Oppgave   V?r 2003                   *
% *                                                      *
% *      Bjarne Foss 1996                                *
% *                                                      *
% * qp_prodplan.m                                        *
% *                                                      *
% * m-file for calculating QP solution.                  *
% *                                                      *
% * Oppdated 10/1-2001 by Geir Stian Landsverk           *
% *                                                      *
% * Verified to work with MATLAB R2015a,                 *
% *    Andreas L. Fl?ten                                 *
% *                                                      *
% * Verified to work with MATLAB R2018b,                 *
% *    Joakim R. Andersen                                *
% *                                                      *
% ********************************************************


global XIT; % Storing iterations in a global variable
global IT;  % Storing number of iterations in a global variable
global x0;  % Used in qp1.m (line 216).

IT=1; XIT=[];

x0 = [0 0]'; % Initial value
vlb = [0 0]; % Lower bound on x
vub = [];    % Upper bound on x

% min 0.5*x'*G*x + x'*c
%  x
%
% s.t. A*x <= b

% Quadratic objective (MODIFY THESE)
G = [0.8 0 ;
     0 0.4]; % Remember the factor 1/2 in the objective
c = [-3 ; -2];

% Linear constraints (MODIFY THESE)
A = [2 1 ;
     1 3];
b = [8; 15];

options = optimset('LargeScale','Off');
[x,lambda] = quadprog1(G,c,A,b,[],[],vlb,vub,x0,options);
clc;

x1=linspace(-1,7);
x2=linspace(-1,7);
```

Figure 3: Matlab code for 2c, part 1



11.02.19 17:08 C:\Users\sjur_\OneDri...\qp_prodplan.m   2 of 2

```matlab
[X1, X2] = meshgrid(x1,x2);
f = 0.4*X1.^2 + 0.2*X2.^2 - 3*X1 - 2*X2;
contour(X1,X2,f);
xlabel("x1");
ylabel("x2");
title("Contour lines and constraints")
hold on;
plot(x1, -2*x1+8, '-k');
plot(-3*x2+15,x2,'-k');
plot(zeros(length(x2)),x2,'-k');
plot(x1,zeros(length(x1)),'-k');
xlim([-1,7]);
ylim([-1,7]);

disp('Iteration sequence:')
disp(XIT');
disp('Solution:')
disp(x);
for i = 1:3
    text(XIT(i,1),XIT(i,2), ['\leftarrow iteration ', num2str(i-1)]);
end
```

Figure 4: Matlab code for 2c, part 2