Primal-Dual Interior Point Methods

MATH 404

Dr. Ahmed Abdelsamea

Name: **Salma Yassin 201-801-012**

I. Introduction:

Interior-point methods are a class of techniques used to solve both linear and nonlinear convex optimization problems. The interior point method was devised by Soviet mathematician I. I. Dikin in 1967, and it was rediscovered in the United States in the middle of the 1980s. In 1984, Narendra Karmarkar created Karmarkar's algorithm, a linear programming technique that is both extremely effective in practice and runs in provably polynomial time. Problems with linear programming that the simplex method was unable to solve could now be solved. Unlike the simplex technique, IP methods traverse the interior of the feasible region to arrive at the optimal solution.

The IP methods had undergone a series of developments through the years. First, Yurii Nesterov, and Arkadi Nemirovski came up with a method to solve convex problems such that the number of iterations is guaranteed to be polynomially bounded. Later, Khachiyan developed the ellipsoid approach which was a polynomial-time algorithm but was too slow to be useful in real-world applications. Then, Karmarkar's discovery revived the study of interior-point techniques and barrier problems by demonstrating that it was feasible to develop a linear programming algorithm with polynomial complexity that was also competitive with the simplex approach. Many others have been developed then after and the most successful of these methods is the primal-dual path-following interior-point method specifically Mehrotra's predictor-corrector algorithm

---

II. Motivation:

One of the challenges with several real-world LP situations is that the number of constraints or variables is too huge, exceeding the computer's available storage space. An important class of these problems is known as transportation problems. These problems can not be solved efficiently using the simplex method. Instead, applying formulations and algorithms from nonlinear programming and nonlinear equations to those and other LP problems proved to be efficient. One of those important techniques is the interior-point method, especially the primal-dual subclass as they can solve large problems quite efficiently exceeding the simplex method.

III. Theory:

Consider a primal LPP in the standard form as

$$
\begin{gathered}
min\, c^T x \\
s.t.\ Ax = b \\
x \geqslant 0
\end{gathered}
$$

The dual problem can be formulated as

$$
\begin{gathered}
max\, b^T y \\
s.t.\ Ay + s = c \\
s \geqslant 0
\end{gathered}
$$

Then we can derive the KKT conditions for both problems as

$$
\begin{gathered}
A^T y + s = c \\
Ax = b \\
x^T s = 0 \\
(x, s) \geq 0
\end{gathered}
$$

The primal-dual method then does generate iterates that strictly satisfy the KKT conditions by following a two-step procedure. The first step is to determine a search direction and the second is to specify a step size in that search direction to ensure that the point resulting is a feasible point with $(x, s) \geq 0$.

The searching direction can be found using the newton method and with adding some parameters to loosen the aggressiveness of the method, then the desired system become

$$
\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -XSe + \mu\sigma e \end{bmatrix}
$$

Where σ is the centering parameter and μ represents the duality measure and can be found using

$$\mu = \frac{x^T s}{n}$$

The system of equations given can then be solved for $(\Delta x, \Delta y, \Delta s)$ and then be used to find the new step as

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (x^k, y^k, s^k) + \alpha(\Delta x^k, \Delta y^k, \Delta s^k)$$

In case of the central path method, we just ensure that all pairwise products $x_i s_i$ are all equal to.

In the case of Mehrotra however, two steps are performed in each iteration. One set the searching direction using affine scaling and the second corrects for the error resulting from the first system by incorporating the error in the system of equations and then solving it again.

For the predictor step, the system of equations that should be solved can be represented by:

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} -r_c \\ -r_b \\ -XSe \end{bmatrix}$$

Then to calculate the error that would be used in the next step, we multiply $(x_i + \Delta x_i^{aff})$ by $(s_i + \Delta s_i^{aff})$ which should be equal to zero at the optimal solution. This value can be simplified as $\Delta x_i^{aff} \Delta s_i^{aff}$. In the corrector step, that error would be then added to the above system of equations to give the final direction that should be used.

For the corrector step, the system of equations would then be

$$\begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} -r_c^k \\ -r_b^k \\ -X^k S^k + \Delta x_{aff}^k \Delta s_{aff}^k + \sigma_k \mu_k \end{bmatrix}$$

As before, the system of equations given can then be solved for $(\Delta x, \Delta y, \Delta s)$ and then be used to find the new step as

$$x^{k+1} = x^k + \alpha_k^{primal} \Delta x^k$$
$$y^{k+1} = y^k + \alpha_k^{dual} \Delta y^k$$
$$s^{k+1} = s^k + \alpha_k^{dual} \Delta s^k$$

IV. Algorithm:

The report provides an implementation for three variations of the Interior-point methods:

Central path with fixed step:

- Start by choosing an initial point (x^0, y^0, s^0) where $s^0 > 0$ and $x^0 > 0$
- Start a series of iterations until the condition ($X^T S$ > Tolerance) is not satisfied anymore. In each of those iterations:

    - Start by computing the value of μ_k using

    $$\mu_k = \frac{X^T S}{n}$$

    - Then Comput r_c, r_b, and r_{xs} using

    $$r_{xs} = X^T S - \sigma_k \mu_k$$
    $$r_c = A^T Y + s - c$$
    $$r_b = AX - b$$

- Cholesky factorization of AD^2A^T is computed as

$$LL^T = AD^2A^T$$

- If we compute Z as

$$Z = (L^T)^{-1}(-r_b - AD^2r_c + AS^{-1}r_{xs})$$

- The vector $(\Delta x, \Delta y, \Delta s)$ is then calculated using

$$\Delta y = L^{-1}Z$$

$$\Delta s = -r_c - A^T\Delta y$$

$$\Delta x = -S^{-1}r_{xs} - D^2\Delta s$$

- Set the new step using

$$x_{k+1} = x_k + \alpha_k \Delta x$$

$$y_{k+1} = y_k + \alpha_k \Delta y$$

$$s_{k+1} = s_k + \alpha_k \Delta s$$

- Start the next iteration

<u>Central path with adaptive step:</u>

- For the central path method with adaptive step size, it would have exactly the same implementation except that an additional step would be added in each iteration to compute the step size as

$$\alpha_k = min(1,\ min(\frac{x_i}{\Delta x_i}))$$

<u>Mehrotra predictor-corrector:</u>

- Start by choosing an initial point (x^0, y^0, s^0) where $s^0 > 0$ and $x^0 > 0$
- Start a series of iterations until the condition ($X^T S$ > Tolerance) is not satisfied anymore. In each of those iterations, we perform two major steps: a prediction step using newton's affine scaling directions and a correction step to correct the error resulting from using affine scaling. Each iteration goes as follows:

  - Start by computing the value of μ_k using

$$\mu_k = \frac{X^T S}{n}$$

- For the predictor step Comput $r_c, r_b, and\ r_{xs}$ using

$$r_{xs} = X^T S$$

$$r_c = A^T Y + s - c$$

$$r_b = AX - b$$

- Then compute Cholesky factorization of $AD^2 A^T$ as

$$LL^T = AD^2 A^T$$

Where $D = S^{-1} X$

- If we compute Z as

$$Z = (L^T)^{-1} (- r_b - AD^2 r_c + A S^{-1} r_{xs})$$

- The vector $(\Delta y_{aff}, \Delta s_{aff}, \Delta x_{aff})$ is then calculated using

$$\Delta y_{aff} = L^{-1} Z$$

$$\Delta s_{aff} =- r_c - A^T \Delta y$$

$$\Delta x_{aff} =- S^{-1} r_{xs} - D^2 \Delta s$$

- Then calculate the adaptive step size and adaptive centering parameter using

$$\alpha_{aff}^{pri} = min(1,\ min \frac{-x_i}{\Delta x_i^{aff}})$$

$$\alpha_{aff}^{dual} = min(1,\ min \frac{-s_i}{\Delta s_i^{aff}})$$

$$\mu_{aff} = (x + \alpha_{aff}^{pri} \Delta x^{aff})^T (s + \alpha_{aff}^{dual} \Delta s^{aff})/n$$

$$\sigma = (\frac{\mu_{aff}}{\mu_k})^3$$

- For the corrector step Comput r_{xs} using

$$r_{xs} = X^T S$$

- Then use precomputed Z value and Cholesky factorization of $LL^T = AD^2A^T$ to compute the vector $(\Delta x, \Delta y, \Delta s)$ using

$$\Delta y = L^{-1}Z$$
$$\Delta s = -r_c - A^T\Delta y$$
$$\Delta x = -S^{-1}r_{xs} - D^2\Delta s$$

- Set the new step to be

$$x_{k+1} = x_k + \alpha_k^{pri}\Delta x$$
$$y_{k+1} = y_k + \alpha_k^{dual}\Delta y$$
$$s_{k+1} = s_k + \alpha_k^{dual}\Delta s$$

Where

$$\alpha_k^{pri} = min(1,\ \eta min\frac{-x_i^k}{\Delta x_i})$$
$$\alpha_k^{dual} = min(1, \eta\ min\frac{-s_i^k}{\Delta s_i})$$

- Start new iteration

<u>For choosing the initial point used in all three algorithms:</u>

- First compute

$$x = A^T(AA^T)^{-1}b$$
$$y = (AA^T)^{-1}Ac$$
$$s = c - A^Ty$$

- To ensure that all components of x and s have nonnegative components to be a valid starting point, we define:

$$\delta_x = max(-\ (3/2)min\,x_i,\ 0)$$
$$\delta_s = max(-\ (3/2)min\,s_i,\ 0)$$

- Then we can modify both x and s as:

$$\bar{x} = x + \delta_x e$$
$$\bar{s} = s + \delta_s e$$

- Adding an additional step to make sure that the components of x and s aren't close to zero.

$$\widehat{\delta}_x = \frac{1\,\bar{x}^T \bar{S}}{2e^T \bar{s}}$$
$$\widehat{\delta}_s = \frac{1\,\bar{x}^T \bar{S}}{2e^T \bar{s}}$$

$$x^0 = \bar{x} + \widehat{\delta}_x e$$
$$s^0 = \bar{s} + \widehat{\delta}_s e$$
$$\lambda^0 = \lambda$$

V. Implementation:

I used MATLAB to implement the three different variations of the Interior point method that were discussed above. The first implementation was for Central Path with fixed step size (α) and centering parameter (σ) using the algorithm that was presented in the above section. Function centeral_IP takes the LP system in standard form as A, b, and c representing RHS, LHS of equality conditions, and cost factors of the primal problem respectively. Additionally, it takes a value of the step size, centering parameter, and tolerance for stopping criteria.
The second implementation was for Central Path with adaptive step size (α) and centering parameter (σ). Function central_IP_adaptive takes the same parameters as central_IP except for the step size as it implements an additional step to compute its value internally in each iteration.
The last function, mehrotra, was implementation for the Mehrotra predictor-corrector. The function also takes the LP system in standard form as A, b, and c representing RHS, LHS of equality conditions, and cost factors of the primal problem respectively, and calculate both adaptive steps and centering parameter for each iteration.

VI. Case studies:

The following set of problems were solved by each of the three implemented and compared by MALAB implementation:

Problem 1:

$$Maximize\ f = 1.1x_1 + x_2$$
$$x_1 + x_2 \leq 6$$
$$x_1, x_2 \geq 0$$

Problem 2:

$$\text{Minimize } f = 5x + 2y$$

subject to

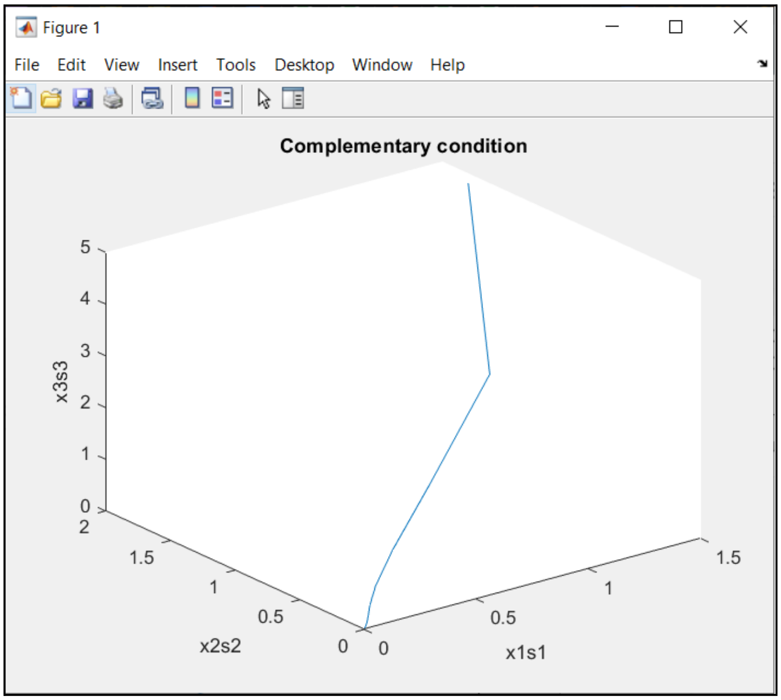
$$3x + 4y \leq 24$$
$$x - y \leq 3$$
$$x + 4y \geq 4$$
$$3x + y \geq 3$$
$$x \geq 0, \quad y \geq 0$$

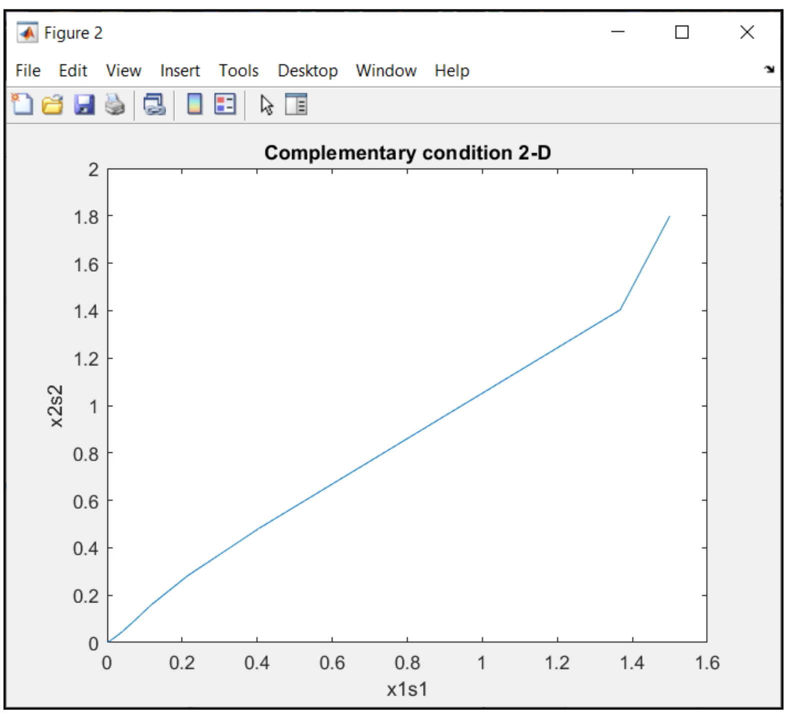
Problem 3:

$$\begin{array}{llrrrrl}
\min z = & x_1 & & - x_3 & - 3x_4 & & \\
\text{s.t.} & 2x_1 & & + 2x_3 & + 3x_4 & = & 10 \\
 & & - 2x_2 & - 2x_3 & - 6x_4 & = & -6 \\
\end{array}$$
$$x_j \geq 0, \quad (j = 1, 2, 3, 4)$$

- **Central Path with fixed step size (α) and centering parameter (σ):**

<u>Problem 1:</u>





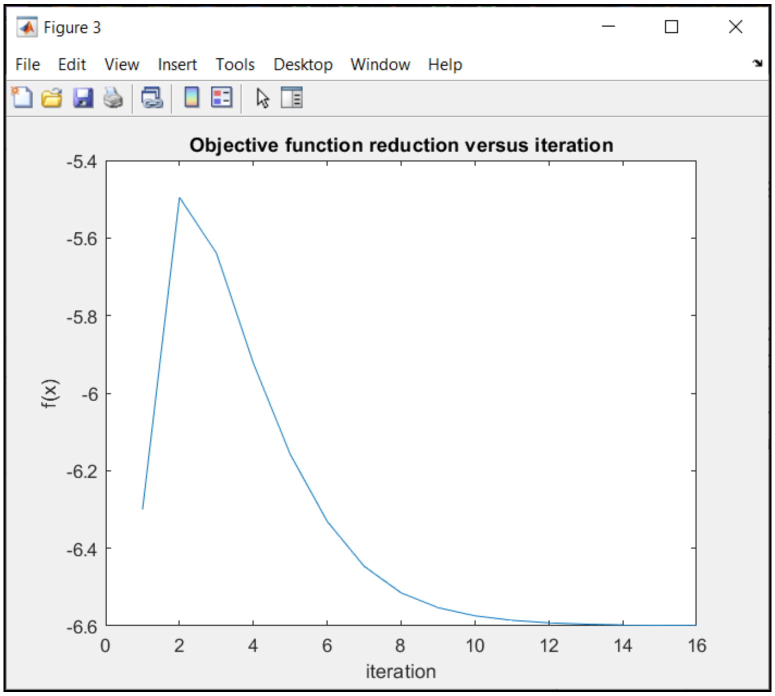


Figure 3
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
-5.4
-5.6
-5.8
-6
-6.2
-6.4
-6.6
0 2 4 6 8 10 12 14 16
iteration

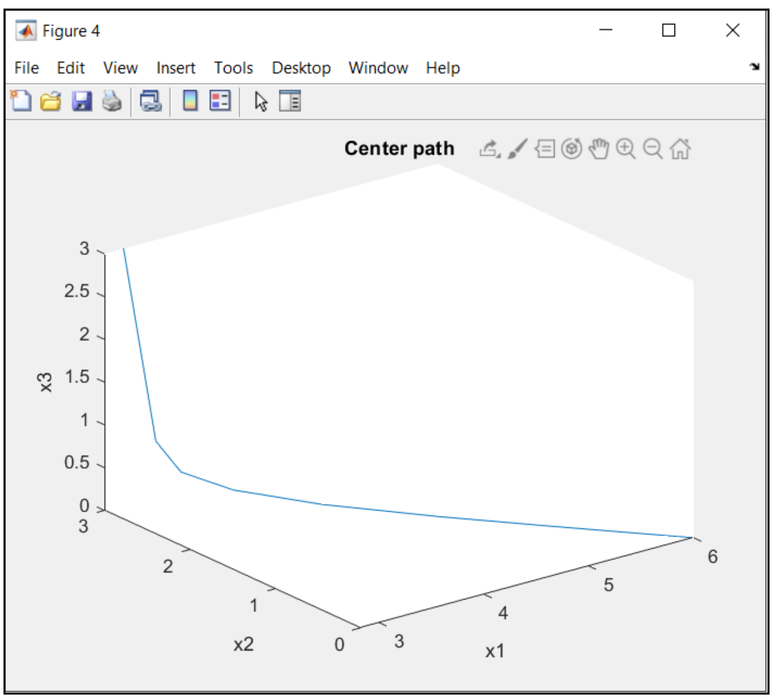
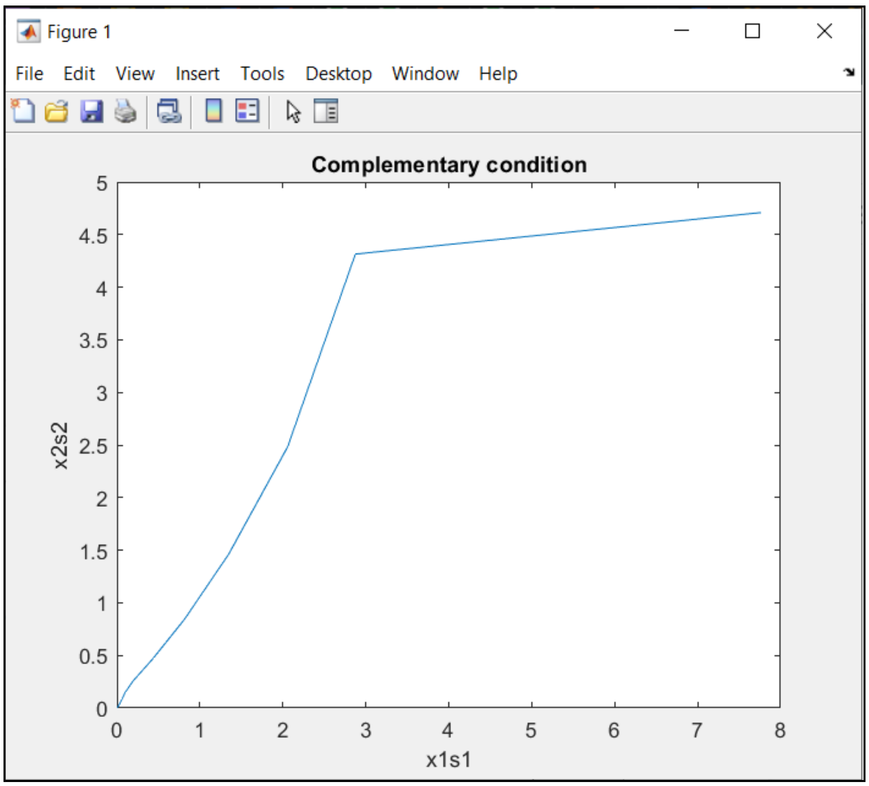


Figure 4
File Edit View Insert Tools Desktop Window Help
Center path
x3
3
2.5
2
1.5
1
0.5
0
x2
3 2 1 0
x1
3 4 5 6

```
The optimal Value is
   -6.5993

The number of iterations is    16

The solution is    5.9979
    0.0020
    0.0002
```

Problem 2:



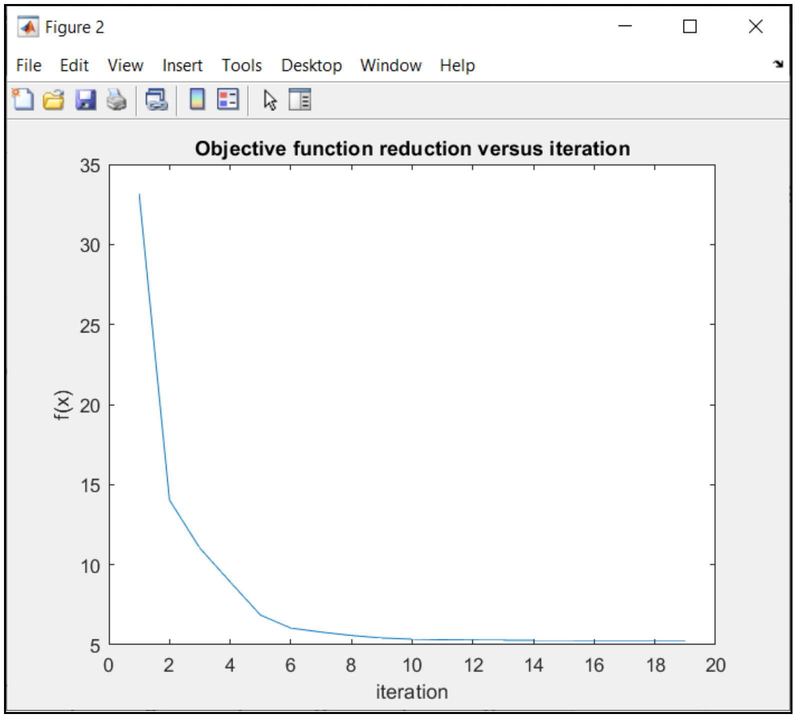


Figure 2
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
iteration
35 30 25 20 15 10 5
0 2 4 6 8 10 12 14 16 18 20

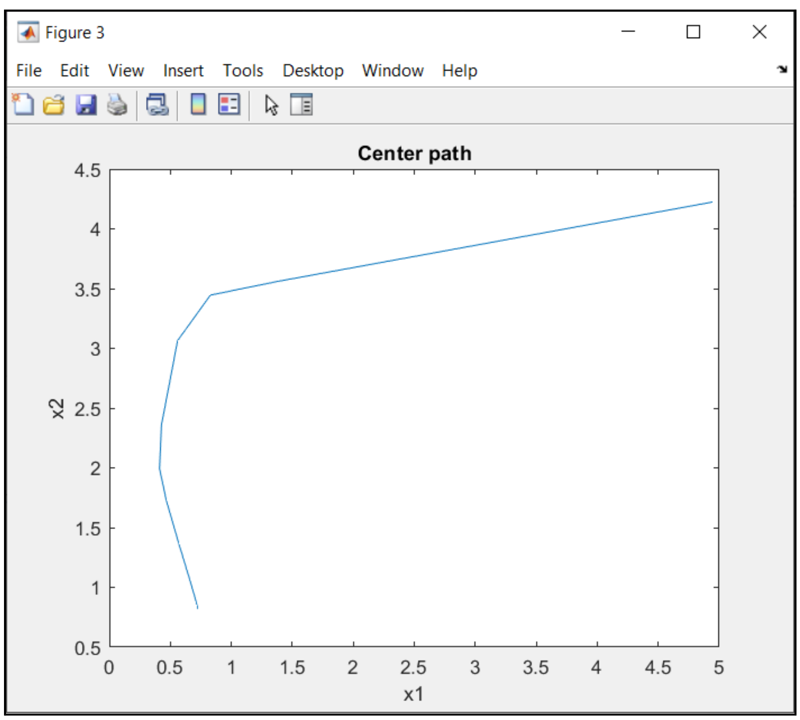
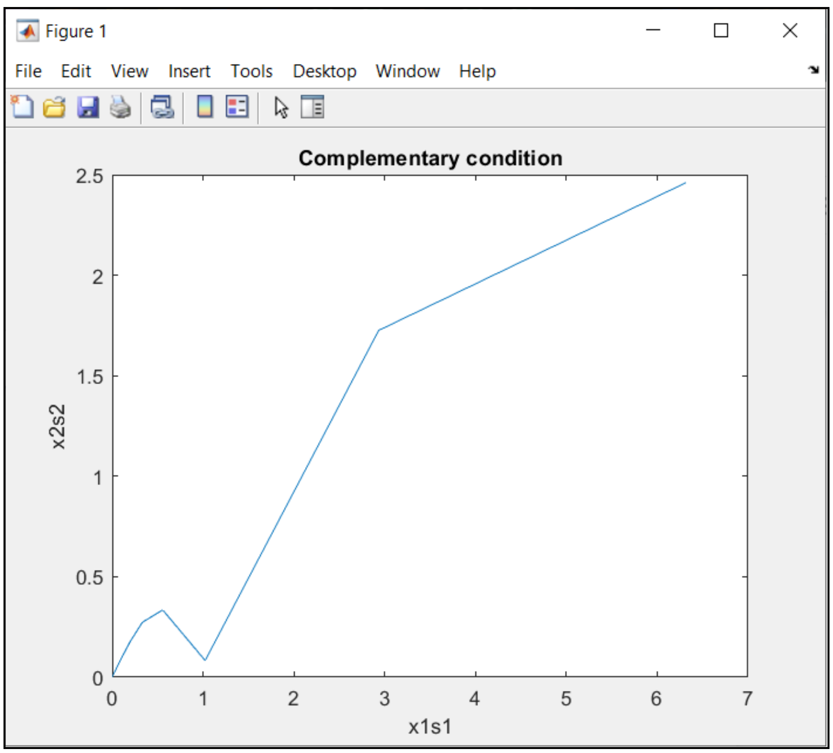


Figure 3
File Edit View Insert Tools Desktop Window Help
Center path
x2
x1
4.5 4 3.5 3 2.5 2 1.5 1 0.5
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5

```
The optimal Value is
    5.2731

The number of iterations is    19

The solution is    0.7272
    0.8185
   18.5445
    3.0913
    0.0011
    0.0001
```

Problem 3:



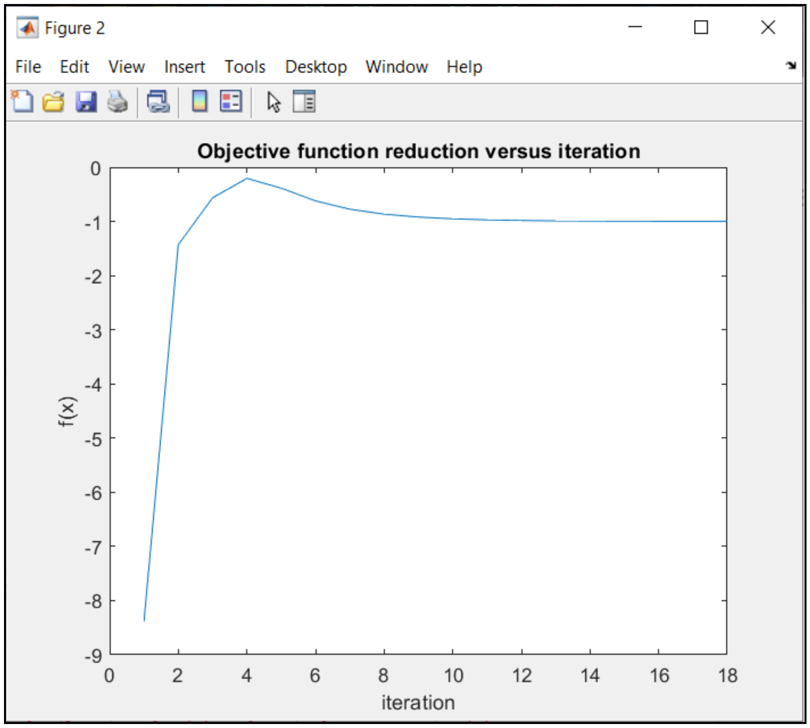


Figure 2
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
iteration
0
-1
-2
-3
-4
-5
-6
-7
-8
-9
0 2 4 6 8 10 12 14 16 18

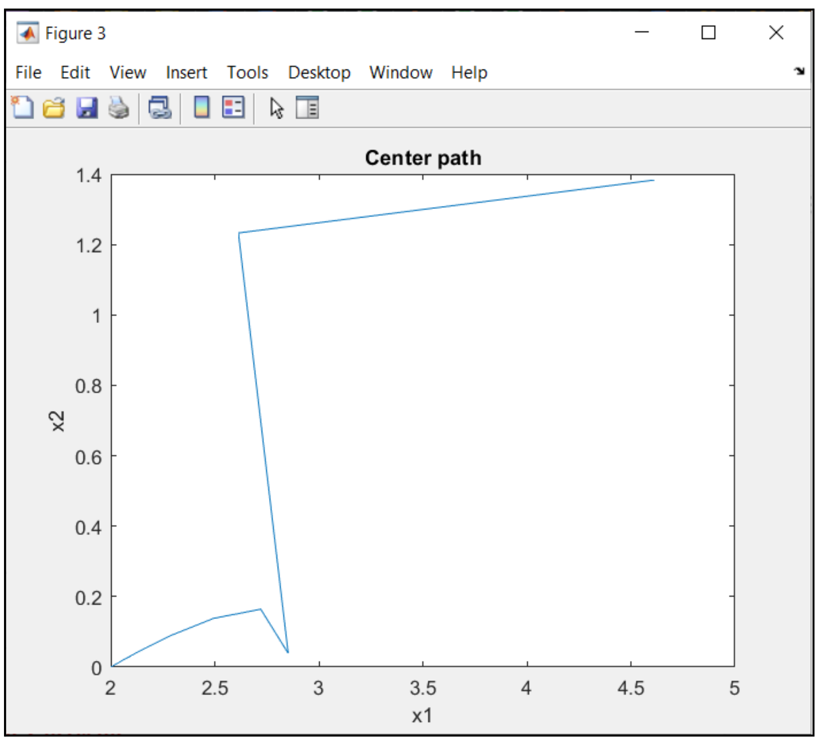
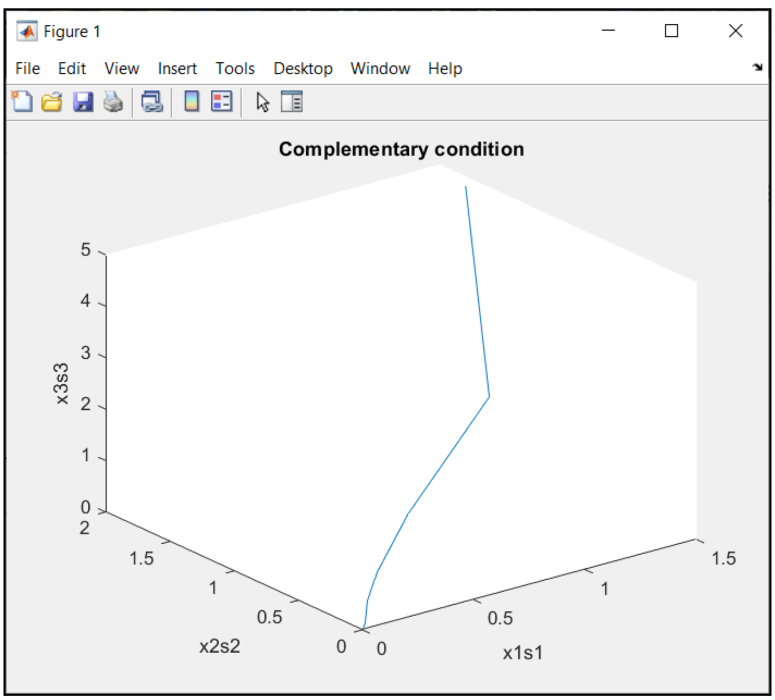


Figure 3
File Edit View Insert Tools Desktop Window Help
Center path
x2
x1
1.4
1.2
1
0.8
0.6
0.4
0.2
0
2 2.5 3 3.5 4 4.5 5

```
The optimal Value is
   -0.9992

The number of iterations is    18

The solution is    2.0004
    0.0001
    2.9994
    0.0002
```

- **Central Path with adaptive step size (α) and centering parameter (σ):**

<u>Problem 1:</u>



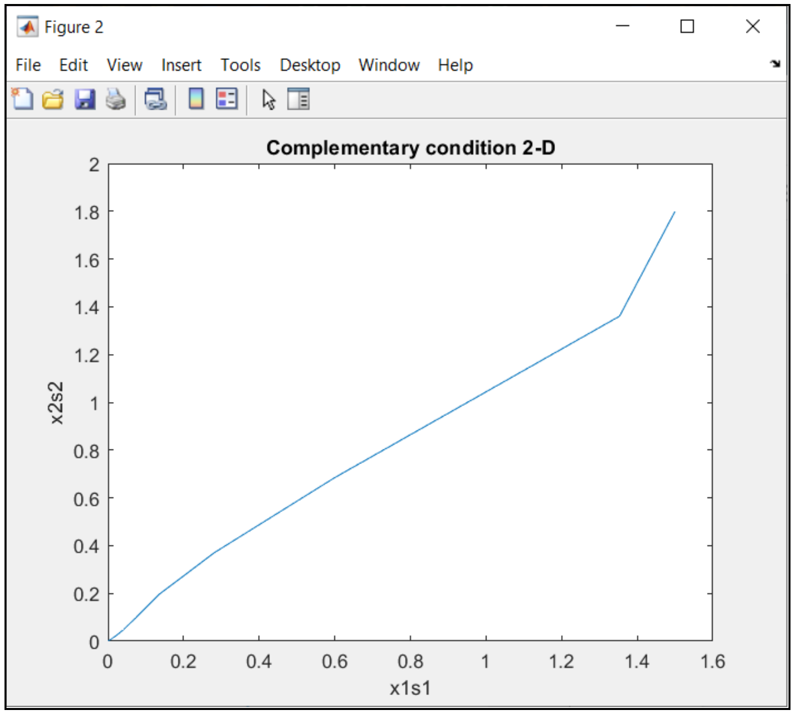


Figure 2
File Edit View Insert Tools Desktop Window Help
Complementary condition 2-D
x2s2
x1s1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6

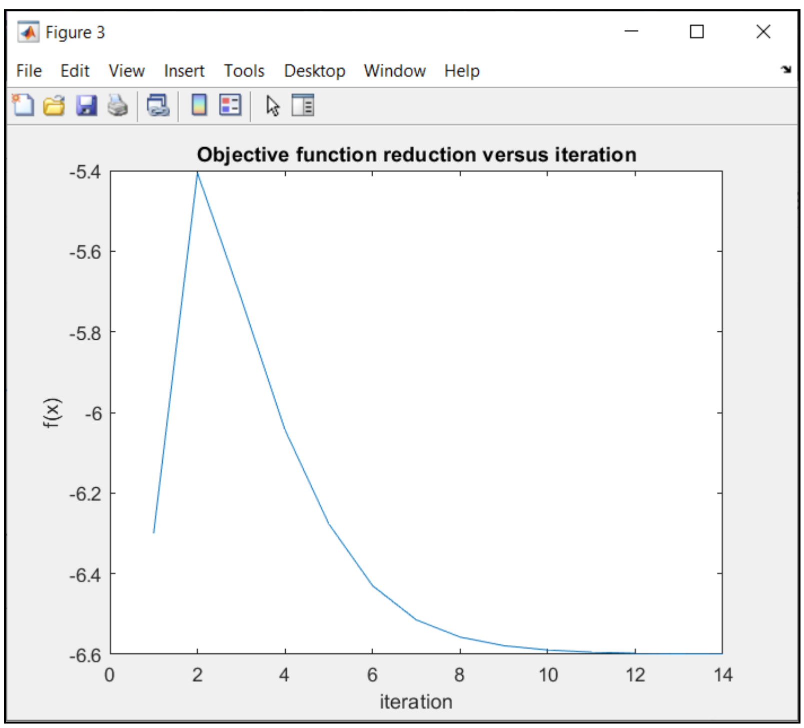
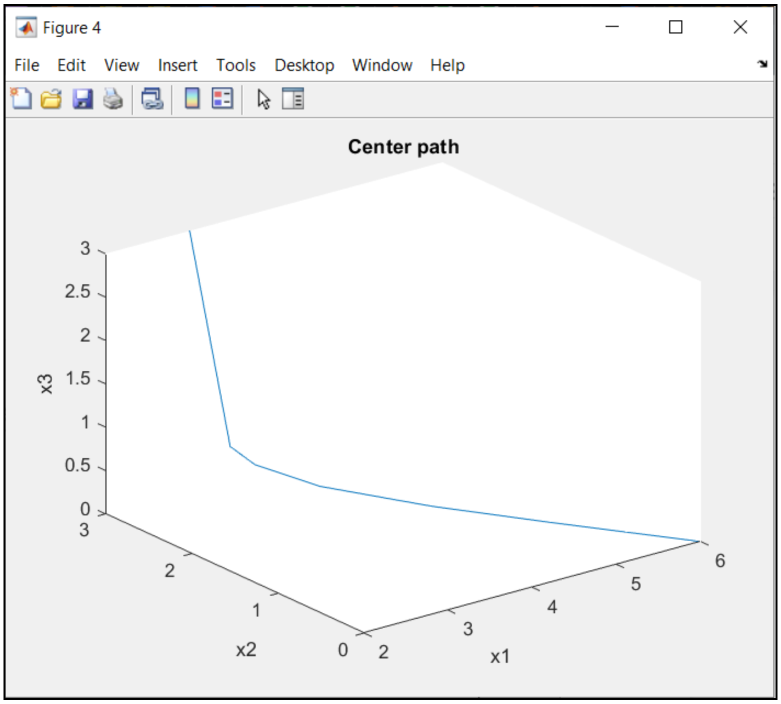


Figure 3
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
iteration
-5.4
-5.6
-5.8
-6
-6.2
-6.4
-6.6
0
2
4
6
8
10
12
14




```
The optimal Value is
   -6.5993

The number of iterations is    14

The solution is    5.9982
    0.0017
    0.0002
```

Problem 2:

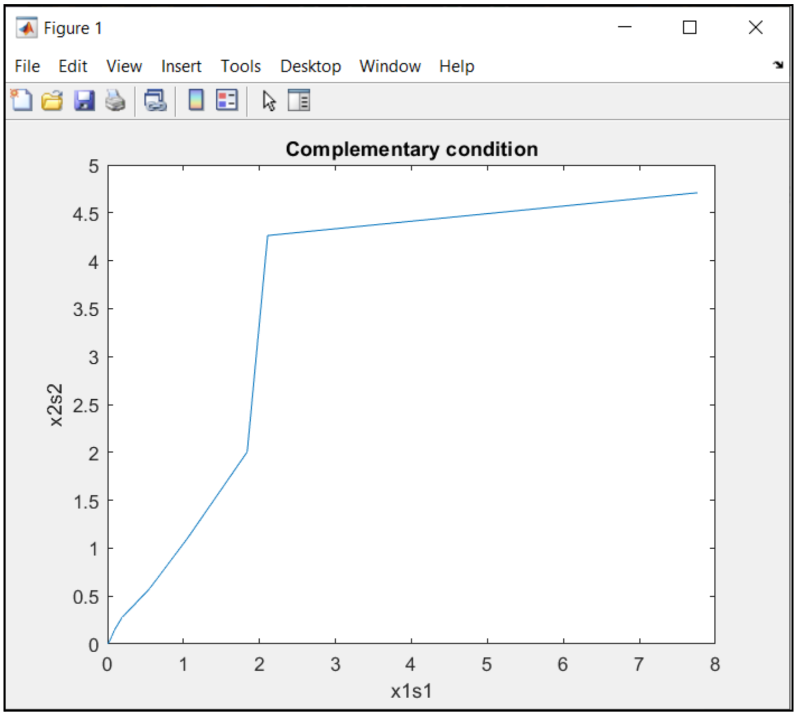


Figure 1
File Edit View Insert Tools Desktop Window Help
Complementary condition
x2s2
x1s1
0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5
0 1 2 3 4 5 6 7 8

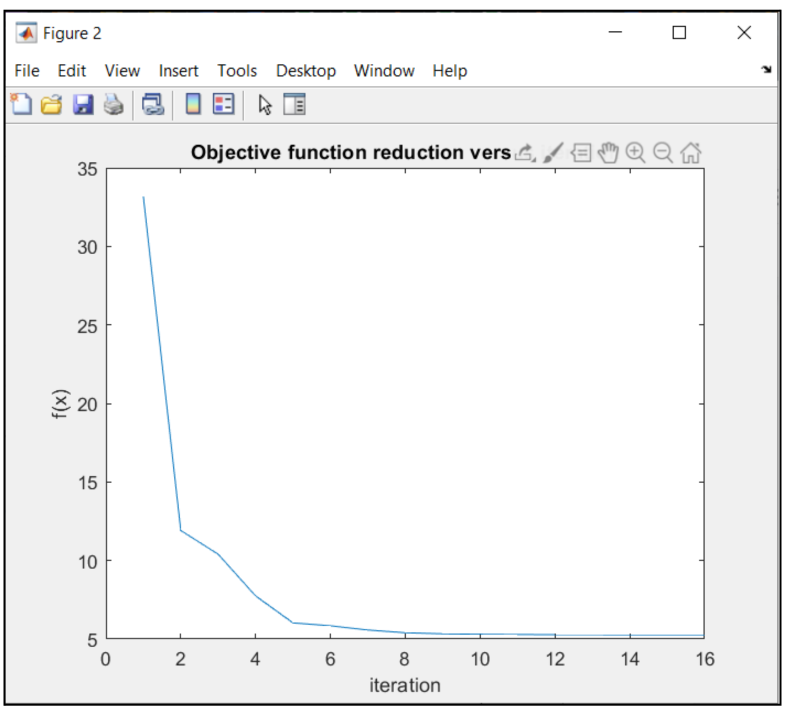
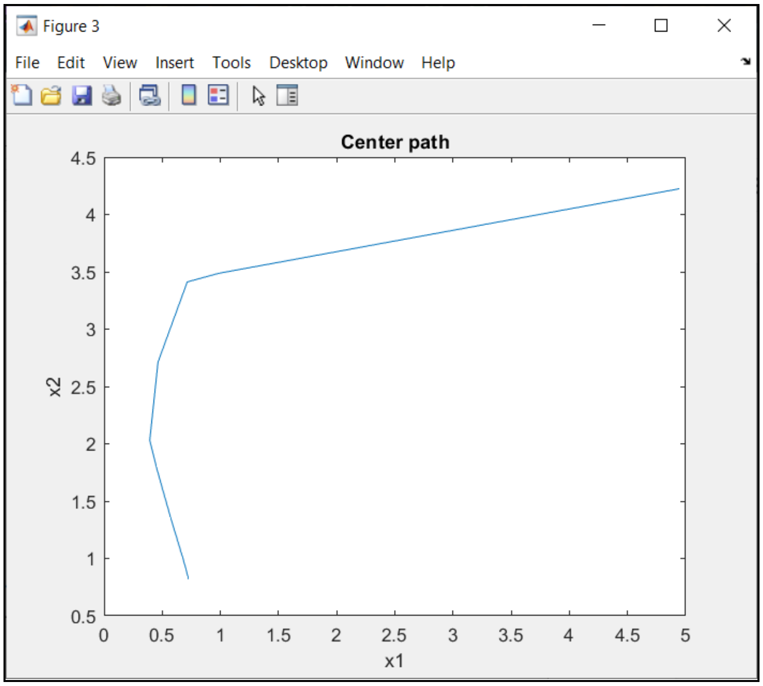


Figure 2
File Edit View Insert Tools Desktop Window Help
Objective function reduction vers
f(x)
iteration
5 10 15 20 25 30 35
0 2 4 6 8 10 12 14 16




```
The optimal Value is
    5.2732

The number of iterations is    16

The solution is    0.7272
    0.8185
   18.5443
    3.0914
    0.0013
    0.0001
```

Problem 3:

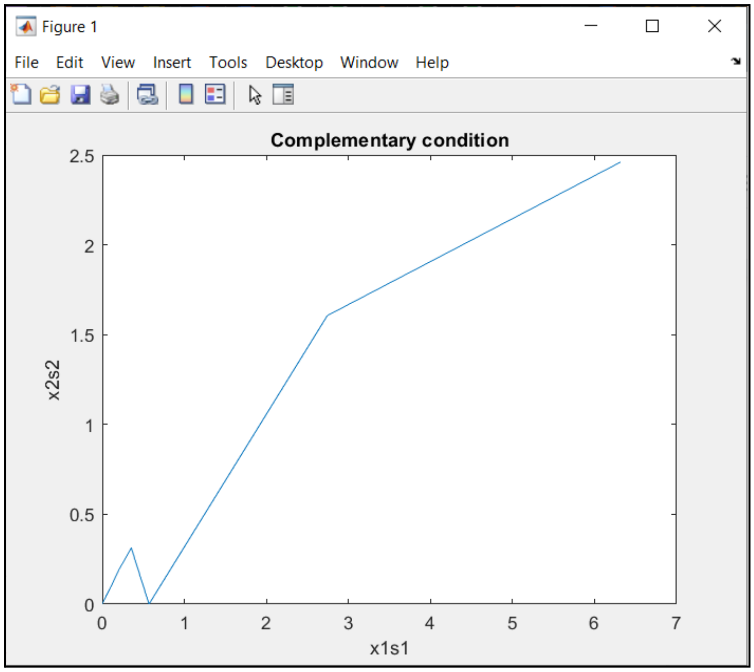


Figure 1
File Edit View Insert Tools Desktop Window Help
Complementary condition
x2s2
x1s1
0
0.5
1
1.5
2
2.5
0
1
2
3
4
5
6
7

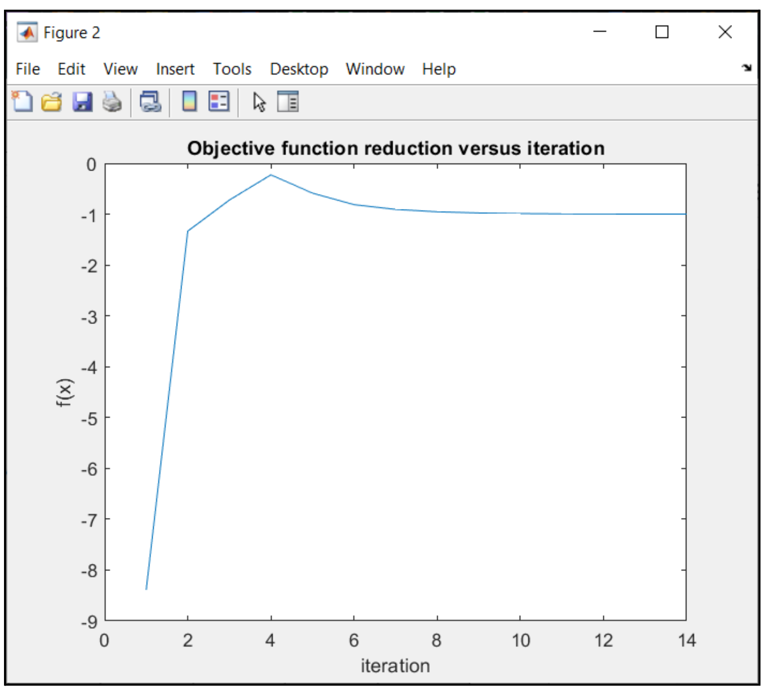
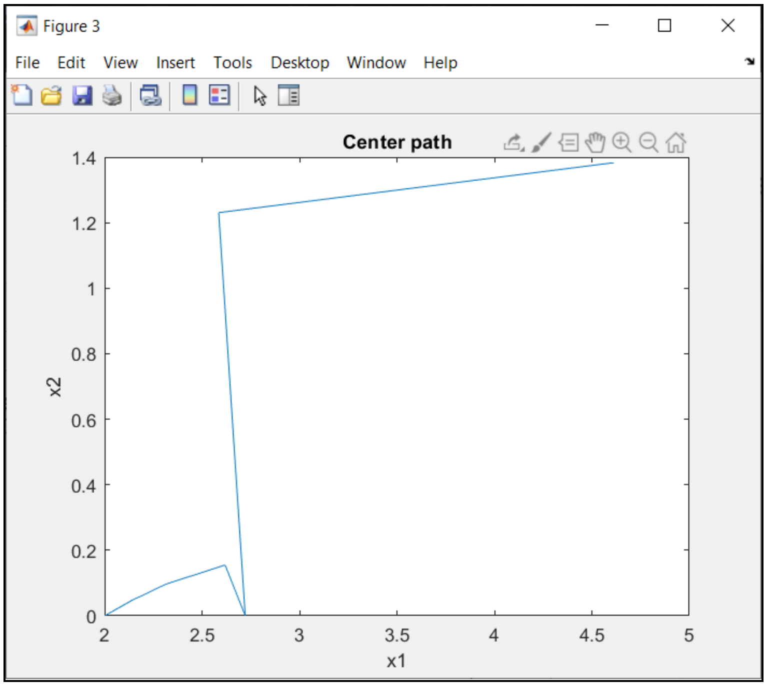


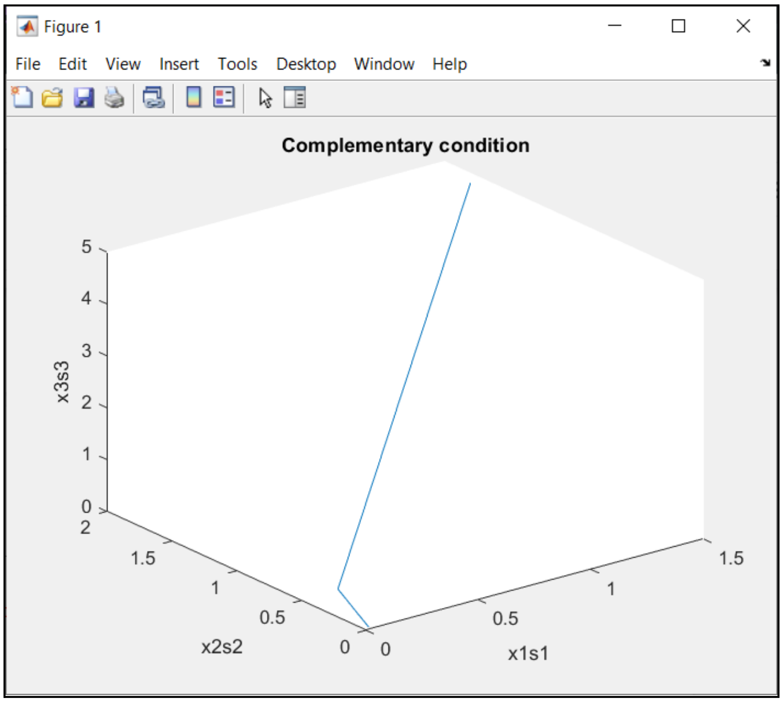
Figure 2
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
iteration
0
-1
-2
-3
-4
-5
-6
-7
-8
-9
0
2
4
6
8
10
12
14

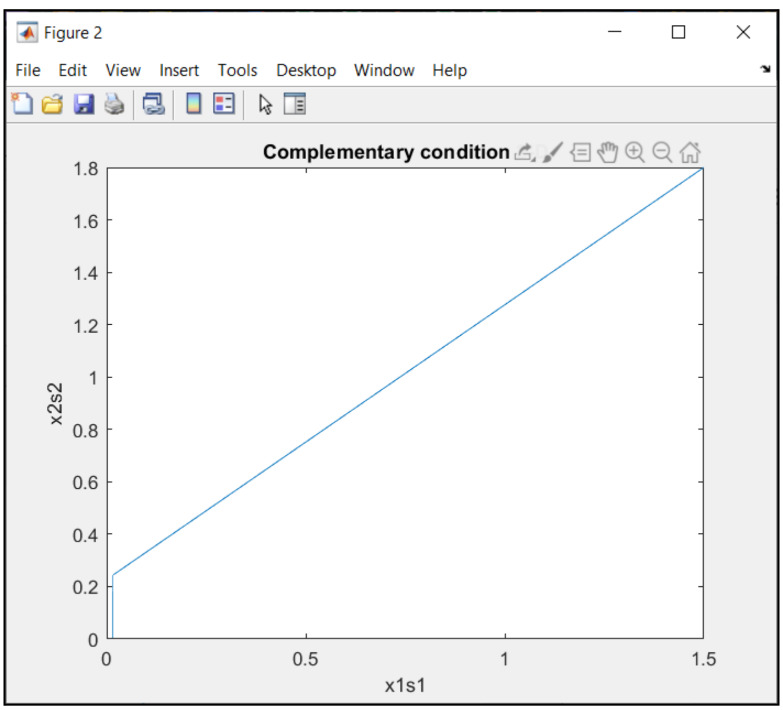



```
The optimal Value is
   -0.9993

The number of iterations is    14

The solution is    2.0003
    0.0001
    2.9995
    0.0001
```

- **Mehrotra Predictor-Corrector:**

<u>Problem 1:</u>





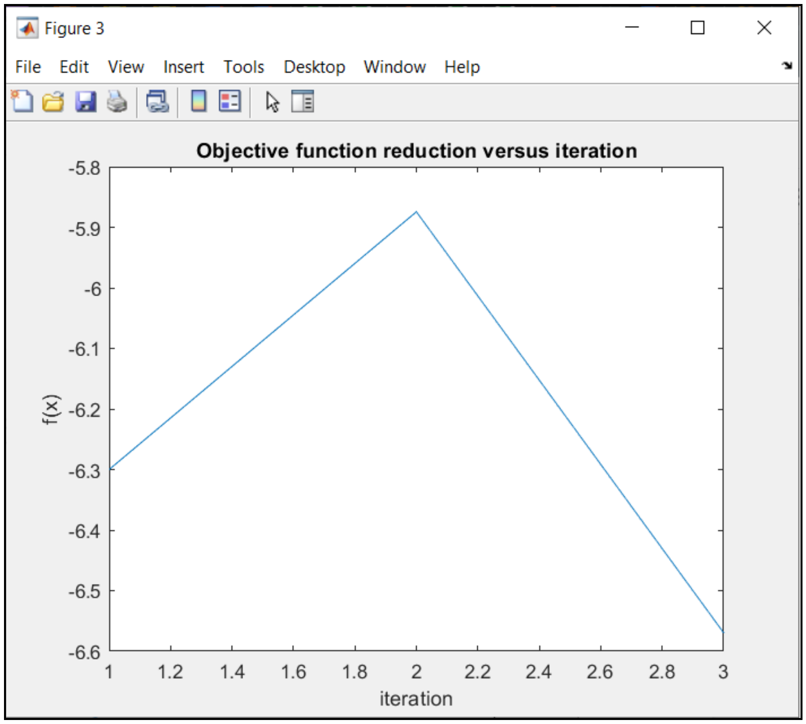


Figure 3
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
-5.8
-5.9
-6
-6.1
-6.2
-6.3
-6.4
-6.5
-6.6
1 1.2 1.4 1.6 1.8 2 2.2 2.4 2.6 2.8 3
iteration

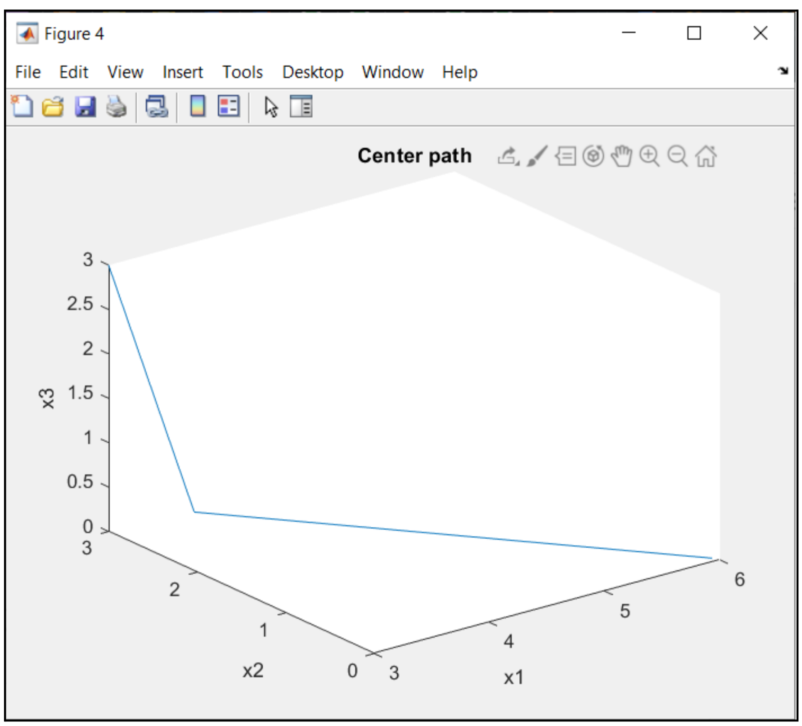
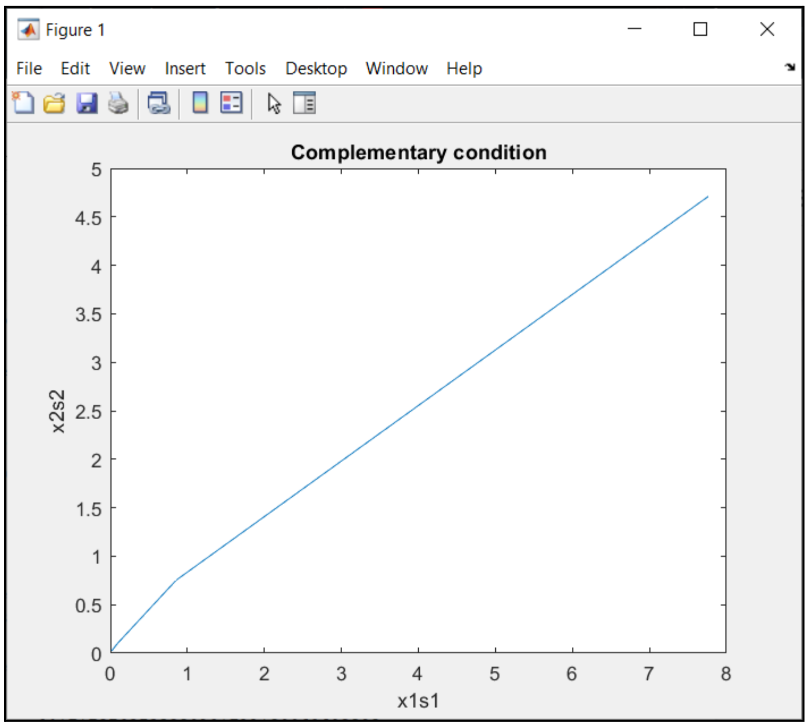


Figure 4
File Edit View Insert Tools Desktop Window Help
Center path
x3
3
2.5
2
1.5
1
0.5
0
x2
3 2 1 0
x1
3 4 5 6

```
The optimal Value is
-6.5696326286044590765540360042529

The number of iterations is     3

The solution is     5.9995012336197194446443886138379
 0.00024384612827936727998759108808587
 0.00025492025200118807562379507399788
```

Problem 2:



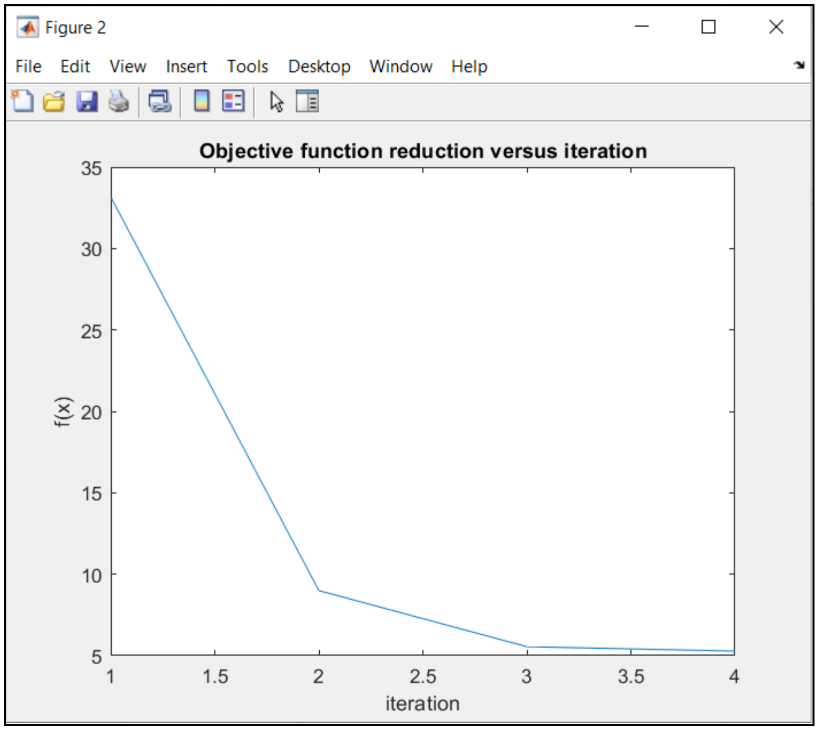


Figure 2
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
iteration
35
30
25
20
15
10
5
1
1.5
2
2.5
3
3.5
4

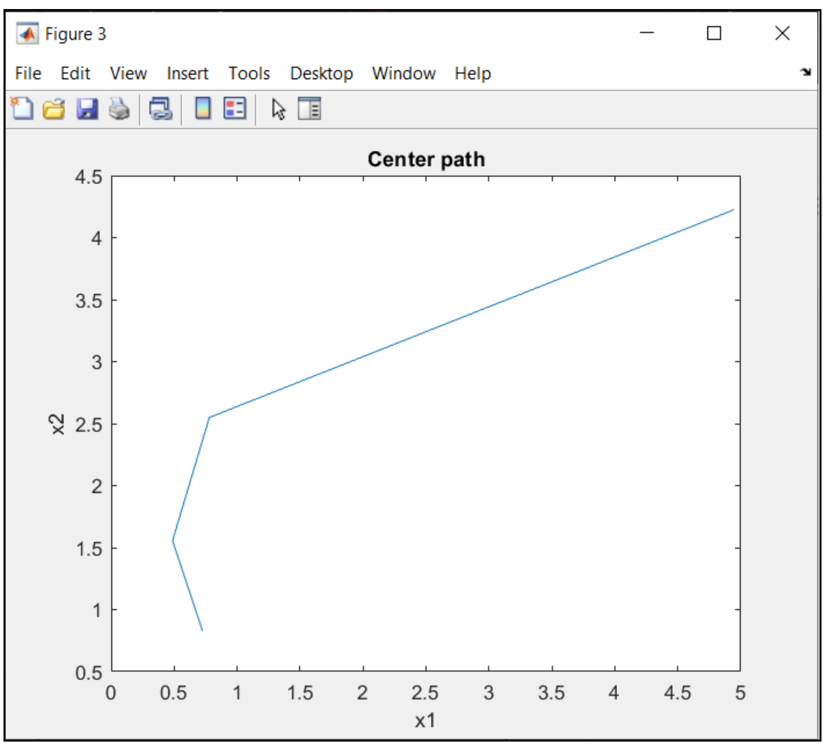
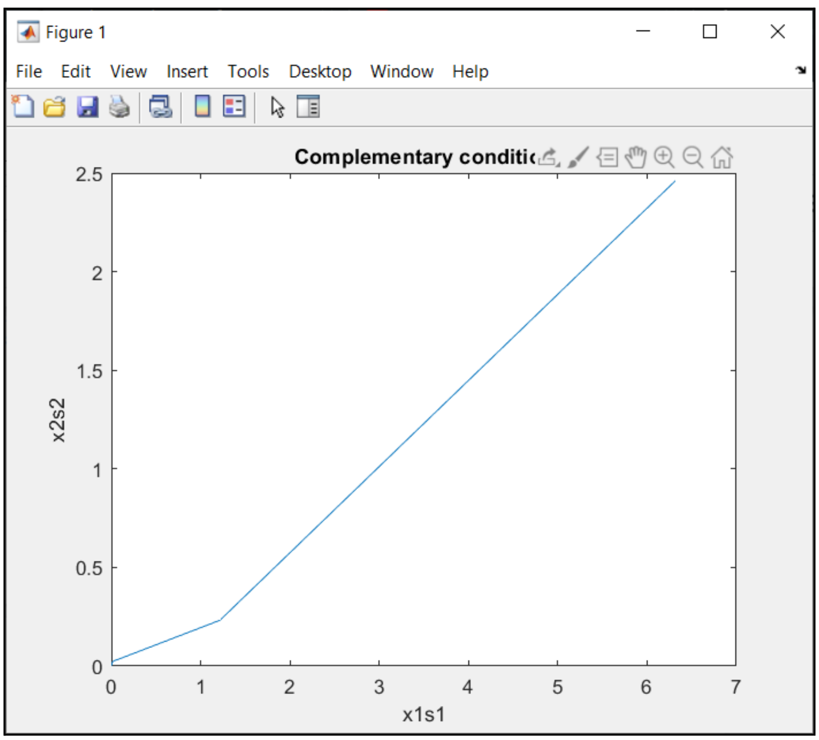


Figure 3
File Edit View Insert Tools Desktop Window Help
Center path
x2
x1
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

```
The optimal Value is
5.2791429531372833225627335802523

The number of iterations is      4

The solution is      0.72725265153530987299730048405383
     0.81826617774670272510980286851395
      18.545177334407259480646640837048
      3.0910135262113928521125023844601
  0.00031736252212077351426572137514172
 0.000024132352632344101704320675443264
```

Problem 3:



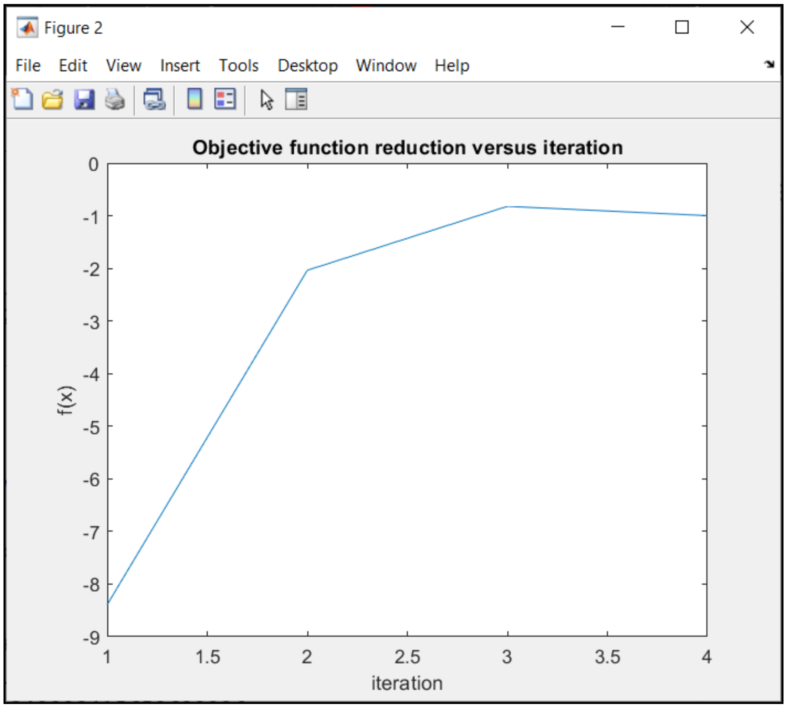


Figure 2
File Edit View Insert Tools Desktop Window Help
Objective function reduction versus iteration
f(x)
iteration
0
-1
-2
-3
-4
-5
-6
-7
-8
-9
1
1.5
2
2.5
3
3.5
4

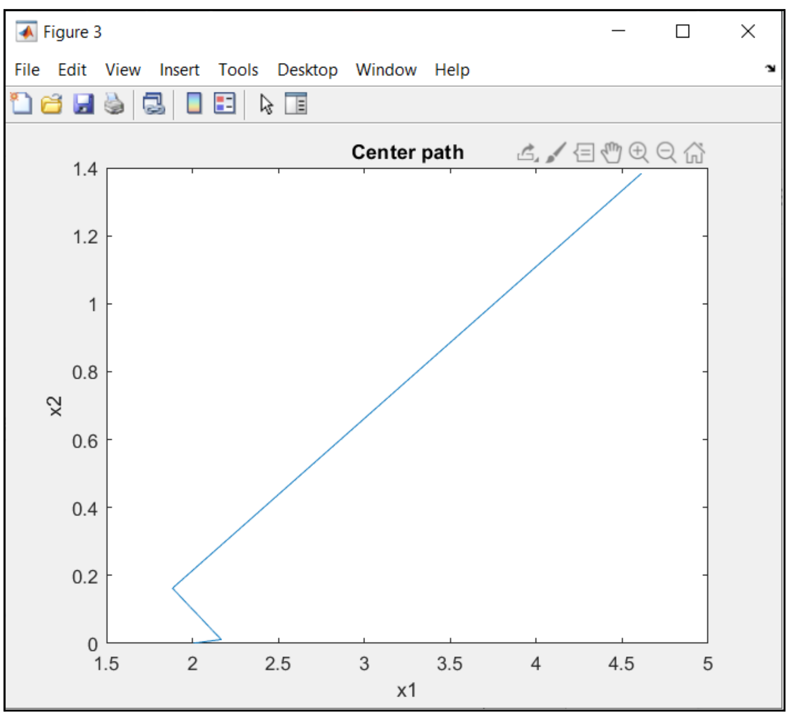


Figure 3
File Edit View Insert Tools Desktop Window Help
Center path
x2
x1
1.4
1.2
1
0.8
0.6
0.4
0.2
0
1.5
2
2.5
3
3.5
4
4.5
5

```
The optimal Value is
-0.99820049734099906971460502922802

The number of iterations is     4

The solution is       2.0000167801121547500087591392705
 0.0000012149324255109851190157824274341
       2.9999676547081160109676007372415
  0.000010376786486159349093415658699886
```

- **MATLAB Implementation:**

Problem 1:

```
PROBLEM 1
Optimal solution found.
x = 3x1
       6
       0
       0

f_val = -6.6000
```

Problem 2:

```
PROBLEM 2
Optimal solution found.
x = 6×1
      0.7273
      0.8182
     18.5455
      3.0909
           0
           0

f_val = 5.2727
```

Problem 3:

```
PROBLEM 3
Optimal solution found.
x = 4×1
       2
       0
       3
       0

f_val = -1
```

Comments:

- Between the three interior methods implmemented, Mehroter always reached the solution in a much smaller number of iterations compared to the other two methods.
- Using adaptive step size with the central path method did actually decrease the total number of iterations required to raech the solution.
- Solutions obtained by the implemented methods are very close to those obtained by MATLAB built-in functions, especially by Mehrotra.

VII. References:

[1] S. S. Rao, “Linear Programming II: Additional Topics and Extensions,” in *Engineering optimization: Theory and practice*, Hoboken, NJ, USA: John Wiley & Sons, Ltd, 2020.

[2] N. Ploskas and N. Samaras, “Interior Point Methods,” in *Linear programming using MATLAB®*, Cham: Springer, 2018.