EE5121 Convex Optimization **Programming Assignment** April 27, 2020

Sourav Sahoo, EE17B040

1 Problem 1

(a) We need to minimize the following:

$$
\begin{aligned}
\underset{x_c, y_c, r}{\text{minimize}} \quad & \sum_{i=1}^{m} ((x_c - x_i)^2 + (y_c - y_i)^2 - r^2)^2 \\
& \implies \underset{x_c, y_c, r}{\text{minimize}} \sum_{i=1}^{m} (x_c^2 - 2x_c x_i + x_i^2 + y_c^2 - 2y_c y_i + y_i^2 - r^2)^2 \\
& \implies \underset{x_c, y_c, r}{\text{minimize}} \sum_{i=1}^{m} (-2x_i x_c - 2y_i y_c + (y_c^2 + x_c^2 - r^2) - (-y_i^2 - x_i^2))^2 \\
& \implies \underset{x_c, y_c, r}{\text{minimize}} \sum_{i=1}^{m} (\mathbf{a}_i^\top \mathbf{z} - \mathbf{b}_i)^2
\end{aligned}
$$

where $\mathbf{a}_i = (-2x_i, -2y_i, 1)^\top$, $\mathbf{z} = (x_c, y_c, y_c^2 + x_c^2 - r^2)^\top$ and $\mathbf{b}_i = -y_i^2 - x_i^2$. Stacking all the $\mathbf{a}_i$'s and $\mathbf{b}_i$'s for $i = 1, \ldots, m$, we get A and $\mathbf{b}$. So, the final objective is:

$$\underset{\mathbf{z} \in \mathbb{R}^3}{\text{minimize}} \quad \|A\mathbf{z} - \mathbf{b}\|_2$$

The details of the solution of the objective function is presented in Figure 1.

```
------------------------------------------------------------------
 number of iterations   =  7
 primal objective value = -2.08222236e+00
 dual   objective value = -2.08222238e+00
 gap := trace(XZ)       = 2.16e-08
 relative gap           = 4.18e-09
 actual relative gap    = 3.81e-09
 rel. primal infeas (scaled problem)   = 7.96e-12
 rel. dual     "        "       "      = 4.76e-12
 rel. primal infeas (unscaled problem) = 0.00e+00
 rel. dual     "        "       "      = 0.00e+00
 norm(X), norm(y), norm(Z) = 1.4e+00, 4.7e+01, 2.9e+00
 norm(A), norm(b), norm(C) = 1.0e+02, 2.0e+00, 3.7e+02
 Total CPU time (secs)  = 0.07
 CPU time per iteration = 0.01
 termination code       =  0
 DIMACS: 8.0e-12  0.0e+00  2.5e-11  0.0e+00  3.8e-09  4.2e-09
------------------------------------------------------------------

------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): +2.08222
```

Figure 1: Details of solution of Question 1

(b) It is to be noted that the optimal value of $\mathbf{z} = (z_1, z_2, z_3)^\top$ that is obtained needs to manipulated to obtain the required values. The optimal required values are $x_c = z_1 = -2.56$, $y_c = z_2 = 6.47$

and $r = \sqrt{z_1^2 + z_2^2 - z_3} = 1.32$. The plot of the best fit circle along with the data points is given in Figure 2.

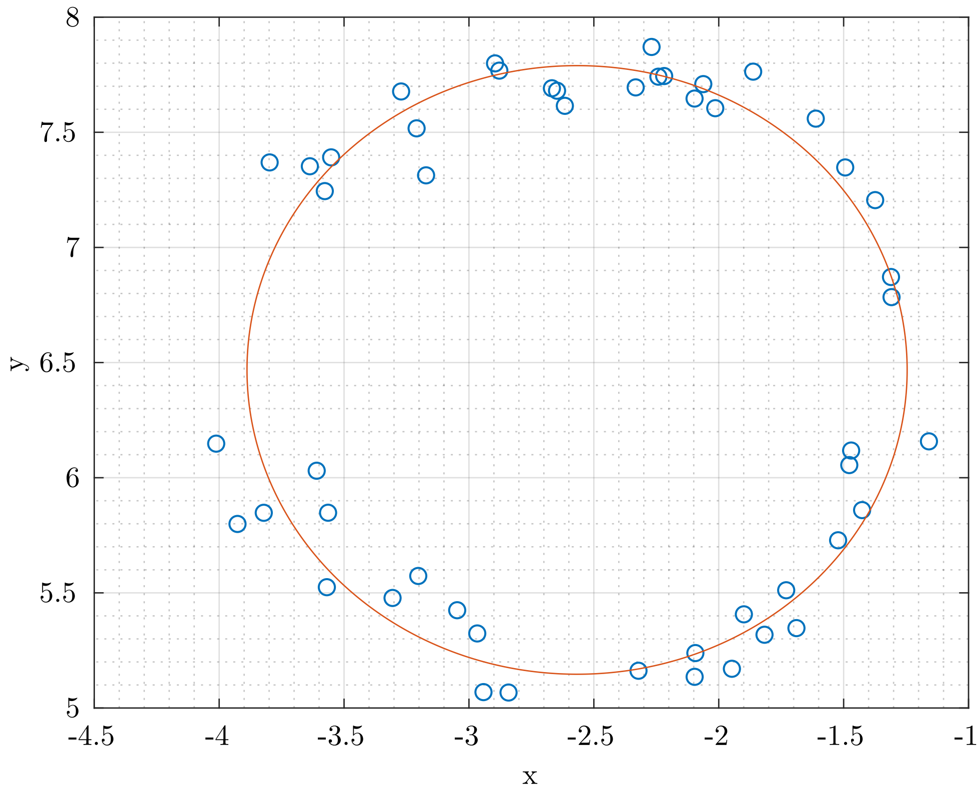

Figure 2: Plot of the data points and the best fit circle

2 Problem 2

The given problem is:

$$\underset{x_1,x_2}{\text{minimize}} \quad f(x) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$

The initial point $x^{(0)} = (-1, 0.7)^\top$ and the stopping criteria is taken to be $\|\nabla f(x)\|_2 \leqslant 0.01$. The gradient of $f(x)$ at $x = (x_1, x_2)$ is computed analytically and the expression is given in (1).

$$\nabla f(x) = \begin{bmatrix} \nabla_{x_1} f(x) \\ \nabla_{x_2} f(x) \end{bmatrix} = \begin{bmatrix} e^{x_1+3x_2-0.1} + e^{x_1-3x_1-0.1} - e^{-x_1-0.1} \\ 3e^{x_1+3x_2-0.1} - 3e^{x_1-3x_1-0.1} \end{bmatrix} \tag{1}$$

Three different methods of descent methods are tried and their performance and trajectory of descent are compared.

(a) **Gradient Descent**
The update step for gradient descent algorithm is given as:

$$x^{(k+1)} = x^{(k)} - t_k \nabla f(x^{(k)})$$

where t_k is the step size for kth iteration. In this question, $t_k = 0.1$ for all values of k till convergence. The trajectory of the descent algorithm is plotted in Figure 3. The optimal value of $x^* = (-0.3468, 0)^\top$.

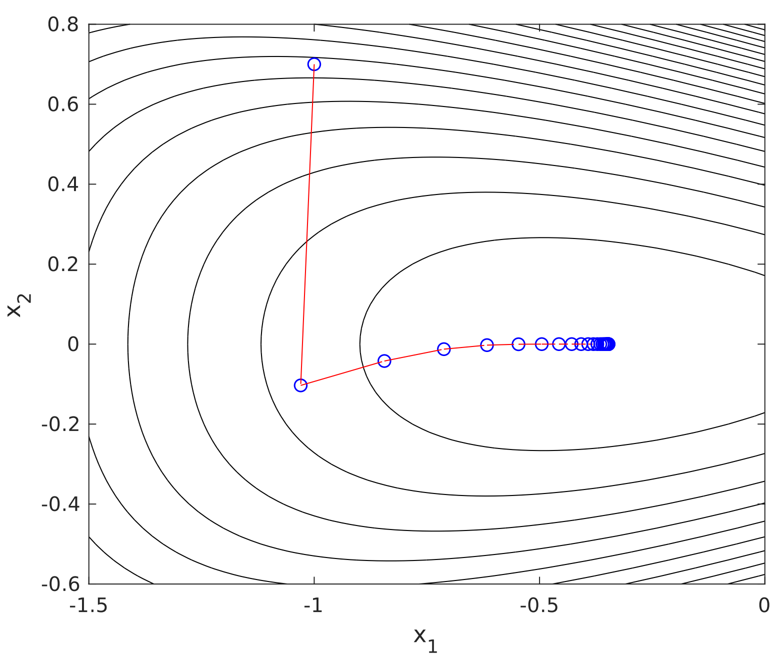

Figure 3: Plot of the trajectory for gradient descent along with equi-contour levels of $f(x)$

(b) **Gradient Descent with Backtracking Line Search**
The update step for a general backtracking line search algorithm is given as:

$$\begin{aligned} &\textbf{while} \quad f(x+tv) > f(x) + \alpha t \nabla f(x)^\top v \textbf{ do } t \leftarrow \beta t \\ &\textbf{else} \quad x^+ = x - t\nabla f(x) \end{aligned}$$

where α, β are constants. In this question, $\alpha = 0.1$ and $\beta = 0.5$. As gradient descent is performed, the value of $v = -\nabla f(x)$. The trajectory of the gradient descent with backtracking line search is plotted in Figure 4. The optimal value of $x^* = (-0.3487, 0)^\top$.

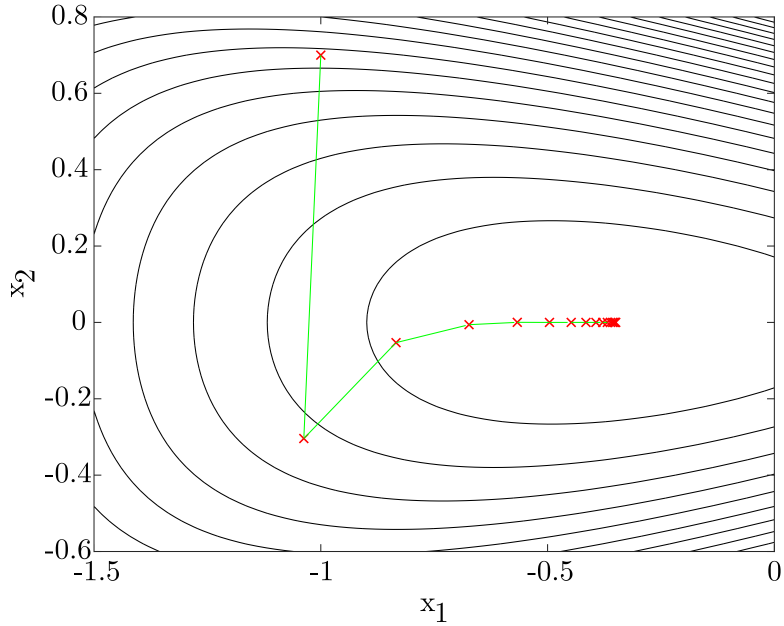

Figure 4: Plot of the trajectory for gradient descent with backtracking line search along with equi-contour levels of $f(x)$

(c) **Newton's Method with Backtracking Line Search**
The update step for a general backtracking line search algorithm is given in (b). Like (b), in this question also, $\alpha = 0.1$ and $\beta = 0.5$. Newton's method is a second order method unlike the gradient descent method. As Newton's method is used, the value of $v = -\nabla^2 f(x)^{-1}\nabla f(x)$. The trajectory of the Newton's method with backtracking line search is plotted in Figure 5. The optimal value of $x^* = (-0.3466, 0)^\top$.

To compare the convergence rates of three methods, we plot the value of $f(x)$ vs the number of iterations in Figure 6. From the four plots, we derive the following inferences:

- The first order methods (gradient descent) updates are orthogonal to the contour lines whereas the it is not necessarily true in second order methods (Newton's method). This is because in

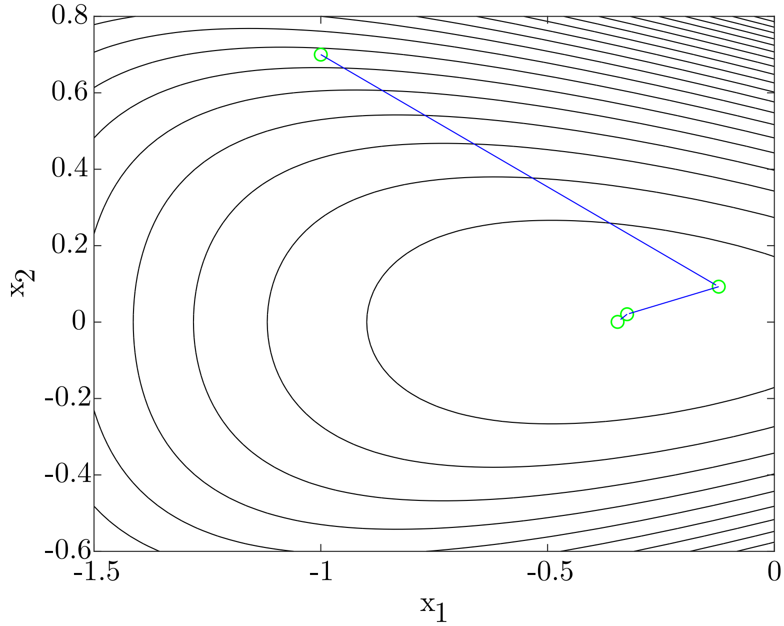

Figure 5: Plot of the trajectory for Newton's method with backtracking line search along with equi-contour levels of $f(x)$

the first-order methods the updates are along the gradients which are orthogonal to equicontour lines whereas in the second order methods the updates are an affine transformation of the gradient.

- The number of iterations required for convergence are in the order: gradient descent (28) > gradient descent with backtracking line search (16) > Newton's method with backtracking line search (4). The number in brackets is the number of iterations required for reaching stopping criteria.
- The first inequality can be explained as in the case of backtracking line search we always take the *largest* step that allows us to descent whereas it is not so in the case of gradient descent with fixed step size.
- In case of Newton's method, the drop is so significant because asymptotically, first order methods converge to ϵ-suboptimal solution in $O(log(1/\epsilon))$ steps whereas the second order method requires $O(log(log(1/\epsilon)))$ iterations.
- It is to be noted that the update costs of second order methods is $O(n^3)$ (matrix inversion) whereas it is only $O(n)$ (gradient calculation) in case of first-order methods, where n is the dimension of $\mathbf{x}$. So, it is infeasible to use vanilla second order methods when $dim(\mathbf{x}) \gg 1$.

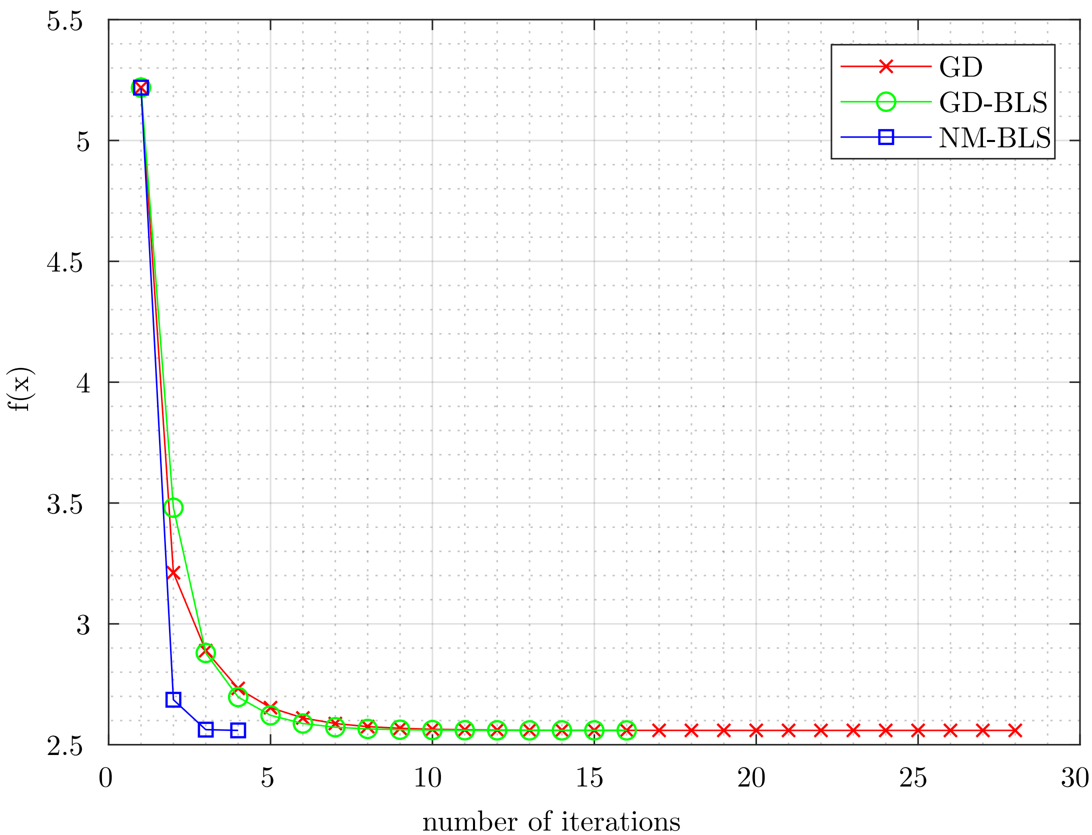

Figure 6: Plot of $f(x)$ with number of iterations. GD = Gradient Descent, GD-BLS = Gradient Descent with Backtracking Line Search and NM-BLS = Newton's Method with Backtracking Line Search.

3 Problem 3

(a) As $\mathbf{x}_i$ can be either 0 or 1 for $\forall i \in [n]$, the primal objective can be written as follows:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \leqslant \mathbf{b} \\ & \mathbf{x}_i(1-\mathbf{x}_i) = 0, \forall i \in [n] \end{aligned}$$

(b) The optimization problem mentioned in (a) is NOT a convex optimization problem as the equality constraints of the problem are neither affine nor linear.

(c) As the problem is not convex, we need to do a convex relaxation for the same. As $g(x) = x(1-x)$ is a concave function in x, we can consider $g(x) \geqslant 0 \implies -g(x) \leqslant 0$ to be a valid convex relaxation. So, the primal problem can be written as:

$$\begin{aligned} \underset{\mathbf{x}\in\mathbb{R}^n}{\text{minimize}} \quad & \mathbf{c}^\top \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \leqslant \mathbf{b} \\ & -\mathbf{x}_i(1-\mathbf{x}_i) \leqslant 0, \forall i \in [n] \end{aligned}$$

Clearly, the problem is now convex. Let $\lambda = (\lambda_1, \ldots, \lambda_m)^\top$ and $\mu = (\mu_1, \ldots, \mu_n)^\top$ be the Langrange multipliers. Let $\Lambda \in \mathbb{R}^{n\times n}$ be a diagonal matrix with its diagonal being μ. It is to be noted that Λ is not a Langrange multiplier, but only a notation to help in computation. So, the Langrangian is computed as follows:

$$\begin{aligned} L(\mathbf{x}, \lambda, \mu) &= \mathbf{c}^\top \mathbf{x} + \lambda^\top (A\mathbf{x} - \mathbf{b}) - \sum_{i=1}^{n} \mu_i \mathbf{x}_i (1-\mathbf{x}_i) \\ &= \mathbf{c}^\top \mathbf{x} + \lambda^\top (A\mathbf{x} - \mathbf{b}) - \sum_{i=1}^{n} \mu_i \mathbf{x}_i + \sum_{i=1}^{n} \mu_i \mathbf{x}_i^2 \\ \implies L(\mathbf{x}, \lambda, \mu) &= \mathbf{c}^\top \mathbf{x} + \lambda^\top (A\mathbf{x} - \mathbf{b}) - \mu^\top \mathbf{x} + \mathbf{x}^\top \Lambda \mathbf{x} \end{aligned}$$

As strong duality holds, we can apply the KKT conditions. So, $\nabla_{\mathbf{x}} L(\mathbf{x}, \lambda, \mu) = 0$ at optimal values of $\mathbf{x}, \lambda, \mu$. So,

$$\begin{aligned} \nabla_{\mathbf{x}} L(\mathbf{x}, \lambda, \mu) = 0 &= \mathbf{c} + A^\top \lambda - \mu + 2\Lambda \mathbf{x}^* \\ \implies \quad \mathbf{x}^* &= -\frac{1}{2}\Lambda^{-1}(\mathbf{c} + A^\top \lambda - \mu) \end{aligned}$$

Substituting the value of $\mathbf{x}^*$, we get:

$$\begin{aligned} & g(\lambda, \mu) = \inf_{\mathbf{x}} L(\mathbf{x}, \lambda, \mu) = L(\mathbf{x}^*, \lambda, \mu) \\ &= -\frac{1}{2}\mathbf{c}^\top \Lambda^{-1}(\mathbf{c} + A^\top \lambda - \mu) - \frac{1}{2}\lambda^\top A \Lambda^{-1}(\mathbf{c} + A^\top \lambda - \mu) - \lambda^\top \mathbf{b} + \frac{1}{2}\mu^\top \Lambda^{-1}(\mathbf{c} + A^\top \lambda - \mu) \\ &+ \frac{1}{4}(\mathbf{c} + A^\top \lambda - \mu)^\top \Lambda^{-\top} \Lambda \Lambda^{-1}(\mathbf{c} + A^\top \lambda - \mu) \\ &= -\lambda^\top \mathbf{b} - \frac{1}{2}(\mathbf{c} + A^\top \lambda - \mu)^\top \Lambda^{-1}(\mathbf{c} + A^\top \lambda - \mu) + \frac{1}{4}(\mathbf{c} + A^\top \lambda - \mu)^\top \Lambda^{-\top}(\mathbf{c} + A^\top \lambda - \mu) \end{aligned}$$

As $\Lambda = \Lambda^\top$, the final expression is:

$$g(\lambda, \mu) = -\lambda^\top \mathbf{b} - \frac{1}{4}(\mathbf{c} + A^\top \lambda - \mu)^\top \Lambda^{-1}(\mathbf{c} + A^\top \lambda - \mu)$$

Expanding the vectorized form, we get:

$$g(\lambda, \mu) = -\lambda^\top \mathbf{b} - \frac{1}{4}\sum_{i=1}^{n} \frac{(\mathbf{c}_i + \mathbf{a}_i^\top \lambda - \mu_i)^2}{\mu_i}$$

where $\mathbf{a}_i$ s are the columns of A. So, the dual problem is:

$$\begin{aligned} \underset{\lambda \in \mathbb{R}^m, \mu \in \mathbb{R}^n}{\text{maximize}} \quad & g(\lambda, \mu) = -\lambda^\top \mathbf{b} - \frac{1}{4}\sum_{i=1}^{n} \frac{(\mathbf{c}_i + \mathbf{a}_i^\top \lambda - \mu_i)^2}{\mu_i} \\ \text{subject to} \quad & \lambda \geqslant 0, \ \mu \geqslant 0 \end{aligned}$$

Now, we aim to remove μ from the expression. It is to be noted that μ appears only in second term of $g(\lambda, \mu)$. Expanding the term within the summation, we get:

$$\begin{aligned} h(\mu_i) &= \frac{(\mathbf{c}_i + \mathbf{a}_i^\top \lambda - \mu_i)^2}{\mu_i} = \frac{(\mathbf{c}_i + \mathbf{a}_i^\top \lambda)^2 + \mu_i^2 - 2\mu_i(\mathbf{c}_i + \mathbf{a}_i^\top \lambda)}{\mu_i} \\ &= \frac{(\mathbf{c}_i + \mathbf{a}_i^\top \lambda)^2}{\mu_i} + \mu_i - 2(\mathbf{c}_i + \mathbf{a}_i^\top \lambda) \end{aligned}$$

To maximize $g(\lambda, \mu)$, we need to minimize $h(\mu_i)$. Differentiating $h(\mu_i)$ w.r.t. μ_i we get:

$$\begin{aligned} h'(\mu_i) &= -\frac{(\mathbf{c}_i + \mathbf{a}_i^\top \lambda)^2}{\mu_i^2} + 1 \\ h''(\mu_i) &= \frac{2(\mathbf{c}_i + \mathbf{a}_i^\top \lambda)^2}{\mu_i^3} \geqslant 0, \ \text{for } \mu_i \geqslant 0 \end{aligned}$$

Hence, setting $h'(\mu_i^*) = 0$, we get $\mu_i^* = |\mathbf{c}_i + \mathbf{a}_i^\top \lambda|$. It is to be noted that as $h''(\mu_i) \geqslant 0$ for $\mu_i \geqslant 0$, μ_i^* indeed minimizes $h(\mu_i)$. Substituting the value of μ_i^* in $h(\mu_i)$, we get:

$$h(\mu_i) = \begin{cases} 0, & \mathbf{c}_i + \mathbf{a}_i^\top \lambda \geqslant 0 \\ -4(\mathbf{c}_i + \mathbf{a}_i^\top \lambda), & \mathbf{c}_i + \mathbf{a}_i^\top \lambda \leqslant 0 \end{cases}$$

Simplifying the expression, we get $h(\mu_i) = -4\min\{0, \mathbf{c}_i + \mathbf{a}_i^\top \lambda\}$. So, the dual problem is:

$$\begin{aligned} \underset{\lambda \in \mathbb{R}^m}{\text{maximize}} \quad & -\lambda^\top \mathbf{b} + \sum_{i=1}^{n} \min\{0, \mathbf{c}_i + \mathbf{a}_i^\top \lambda\} \\ \text{subject to} \quad & \lambda \geqslant 0 \end{aligned}$$

(d) The objective function is a positive combination of an affine function along with point-wise infimum of affine functions. So, it is a concave function. The inequality constraint is also affine. As the objective is to maximize the concave function, the problem is convex.

(e) The details of the solution of the dual problem is shown in Figure 7. The optimal value of the objective is -3 which is attained at $\mathbf{x}^* = (1, 1)^\top$.

(f) As strong duality holds, we have $p^* = d^*$.

```
-------------------------------------------------------------------
 number of iterations   =  7
 primal objective value =  3.00000005e+00
 dual   objective value =  2.99999999e+00
 gap := trace(XZ)       = 5.94e-08
 relative gap           = 8.48e-09
 actual relative gap    = 8.47e-09
 rel. primal infeas (scaled problem)   = 3.86e-13
 rel. dual     "        "       "      = 7.72e-12
 rel. primal infeas (unscaled problem) = 0.00e+00
 rel. dual     "        "       "      = 0.00e+00
 norm(X), norm(y), norm(Z) = 2.2e+00, 1.4e+00, 4.2e+00
 norm(A), norm(b), norm(C) = 4.9e+00, 3.2e+00, 6.1e+00
 Total CPU time (secs)  = 0.11
 CPU time per iteration = 0.02
 termination code       =  0
 DIMACS: 4.2e-13  0.0e+00  9.5e-12  0.0e+00  8.5e-09  8.5e-09
-------------------------------------------------------------------

------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): -3
```

Figure 7: Details of solution of Question 3

4 Problem 4

A linear dynamical system is given in (2)

$$\mathbf{x}(t+1) = A\mathbf{x}(t) + B\mathbf{u}(t) + \mathbf{w}(t),\ t = 1, \dots, T-1 \tag{2}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state, $\mathbf{u}(t) \in \mathbb{R}^m$ is the input, $\mathbf{w}(t) \in \mathbb{R}^n$ is the process noise, $A \in \mathbb{R}^{n \times n}$ is the state transition matrix and $B \in \mathbb{R}^{n \times m}$ is the input matrix. We assume that $\mathbf{w}(t) \sim \mathcal{N}(\mathbf{0}, W), W \succ 0$.

(a) In this part, we optimize an unconstrained optimization problem as given in (3).

$$\underset{A,B}{\text{minimize}} \quad \sum_{t=1}^{T-1} \|\mathbf{x}(t+1) - A\mathbf{x}(t) - B\mathbf{u}(t)\|_2^2 \tag{3}$$

In this question $n = 10$ and $m = 4$. For fast computation, the state vectors and input vectors are stacked into matrices and the Frobenius norm of the resultant matrix is minimized. So, the optimization problem is equivalent to (4).

$$\underset{A,B}{\text{minimize}} \quad \|Y - AX - BU\|_F \tag{4}$$

where $\|\cdot\|_F$ is the Frobenius norm. The ith column of $Y \in \mathbb{R}^{10\times 99}$ corresponds to $\mathbf{x}(i)$ for $i = 2, \dots, 100$, jth column of $X \in \mathbb{R}^{10\times 99}$ corresponds to $\mathbf{x}(j)$ for $j = 1, \dots, 99$ and kth column of $U \in \mathbb{R}^{4\times 99}$ corresponds to $\mathbf{u}(k)$ for $k = 1, \dots, 99$. The details of the optimization is presented in Figure 8. The optimal values $\hat{A}$ and $\hat{B}$ are given in Figure 9 and Figure 10.

(b) In this part of the problem, we introduce a parameter $\lambda > 0$ to trade-off between estimation error and sparsity of the matrices A and B. So, the optimization objective is as given below:

$$\underset{A,B}{\text{minimize}} \quad \|Y - AX - BU\|_F + \lambda(\|A\|_0 + \|B\|_0)$$

where $\|\cdot\|_0$ is the ℓ_0-norm of the matrix or cardinality of the matrix i.e., the number of non-zero entries of the matrix. However, ℓ_0-norm is a pseudo-norm as $\|\lambda x\|_0 \neq |\lambda| \|x\|_0$ for all x and $\lambda \in \mathbb{R}$. So, the objective is NOT convex.

A natural convex relaxation of the problem is to consider the ℓ_1-norm instead of the ℓ_0-norm. So, the modified optimization problem is given as:

$$\underset{A,B}{\text{minimize}} \quad \|Y - AX - BU\|_F + \lambda(\|A\|_1 + \|B\|_1)$$

Clearly, the objective function thus obtained is convex. The value of λ is varied from 0 to 70 in steps of 2. The variation of estimation error f^* and the total number of non-zero entries i.e. ($|A_{ij}| > 0.01$ and $|B_{ij}| > 0.01$) is plotted in Figure 11.

In the plot, we observe an *elbow-shaped* curve. In such a graph, the corner of the elbow is often an ideal location to trade off between estimation error and sparsity because considerable sparsity is obtained without a significant increase in error. So, at the corner of the elbow where $\lambda = 10$, we get $k = 60$ with error 40.93 as compared to the non-regularised case where $k = 95$ and error $= 29.06$. The parameter k is the total number of non-zero entries in $\hat{A}$ and $\hat{B}$.

```
------------------------------------------------------------------
 number of iterations   =  7
 primal objective value = -2.90664650e+01
 dual   objective value = -2.90664651e+01
 gap := trace(XZ)       = 3.98e-08
 relative gap           = 6.73e-10
 actual relative gap    = 6.30e-10
 rel. primal infeas (scaled problem)   = 1.70e-14
 rel. dual     "        "       "      = 2.77e-12
 rel. primal infeas (unscaled problem) = 0.00e+00
 rel. dual     "        "       "      = 0.00e+00
 norm(X), norm(y), norm(Z) = 1.4e+00, 2.9e+01, 4.1e+01
 norm(A), norm(b), norm(C) = 3.0e+03, 2.0e+00, 9.1e+02
 Total CPU time (secs)  = 1.79
 CPU time per iteration = 0.26
 termination code       =  0
 DIMACS: 1.7e-14  0.0e+00  1.5e-11  0.0e+00  6.3e-10  6.7e-10
------------------------------------------------------------------

------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): +29.0665
```

Figure 8: Details of solution of Question 4 (a)

A =

-0.0057	0.0034	0.0095	-0.0222	0.9829	0.0093	-0.0111	0.0048	-0.0066	-0.1041
0.1405	-0.0398	-0.0190	0.0427	0.7742	0.2570	0.0529	0.1793	0.0471	0.0011
-0.0099	0.8227	0.8090	-0.0197	0.0139	0.1830	-0.6037	0.0037	-0.0057	-0.0040
-0.1247	0.0343	0.0288	0.6073	0.0017	0.0425	-0.0493	0.5781	-0.0146	-0.0059
0.0104	-0.0580	-0.0121	0.0247	-0.0040	-0.0165	0.0215	0.0109	0.0053	0.0071
-0.0086	-0.3296	-0.0116	0.0116	-0.2526	-0.0069	0.0092	0.5793	0.0094	0.2927
0.4145	-0.0022	-0.0104	-0.1783	0.0027	-0.0212	-0.0017	0.0101	-0.0046	0.0073
0.0046	0.0604	0.1773	-0.1161	0.0244	-0.2632	0.4033	-0.0253	0.2687	0.6430
0.3041	0.0228	0.7629	-0.0303	0.3210	-0.1254	-0.0251	-0.3748	-0.0094	0.4959
0.0185	-0.3312	-0.0106	0.0291	-0.0101	-0.0229	0.0977	0.0149	-0.0030	0.0239

Figure 9: Optimal value of $\hat{A}$ in Question 4 (a)

B =

0.2754	0.0071	-0.0053	-0.7775
0.6680	-0.0149	-0.0156	0.6864
-2.6170	-0.5320	-0.0016	-1.6210
-0.3423	-0.0087	-0.0113	0.0025
-0.0050	-0.0117	0.0006	0.0078
0.5901	0.1532	-0.0163	0.5168
-0.0038	-0.4113	-0.0009	0.0083
-0.1140	1.2560	0.0044	0.0082
-0.0081	0.4697	-0.2290	-0.0062
0.0025	0.0394	-0.0076	1.3198

Figure 10: Optimal value of $\hat{B}$ in Question 4 (a)

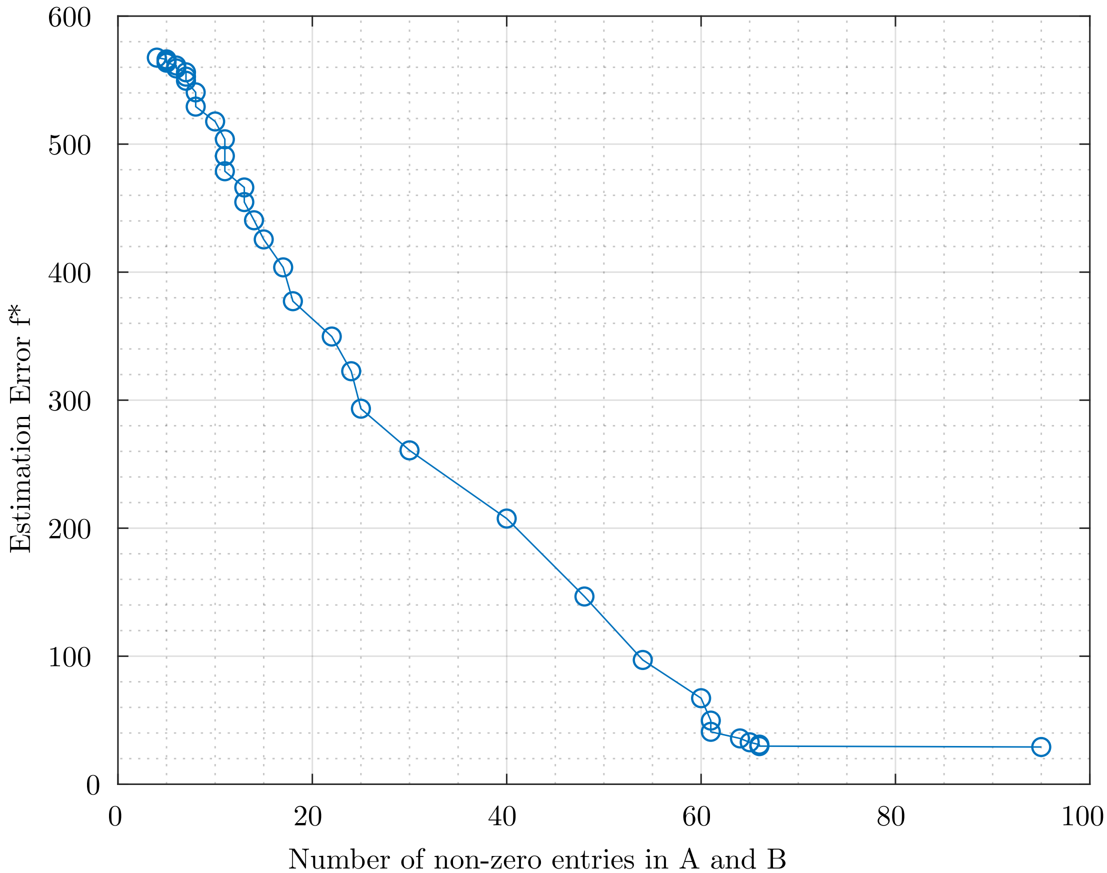

Figure 11: Trade off between estimation error f^* and number of non-zero values in $\hat{A}$ and $\hat{B}$

5 Problem 5

In this question, we do a k-degree polynomial approximation of a non-polynomial function $f(x)$ using N points uniformly spaced in a given interval $I \subset \mathbb{R}$. In the given problem, $k = 3, N = 20$, $f(x) = sin\ x$ and $I = [-\pi, \pi]$. Let the cubic polynomial be as given below:

$$g(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

As the question mandates to reduce the ℓ_1-norm of the approximation error, error for a particular data point x is given as:

$$E(x) = |a_0 + a_1 x + a_2 x^2 + a_3 x^3 - sin\ x|$$

So, the total approximation error is:

$$E(\mathbf{x}) = \sum_{i=1}^{N} E(x_i) = \sum_{i=1}^{N} |a_0 + a_1 x_i + a_2 x_i^2 + a_3 x_i^3 - sin\ x_i| = \sum_{i=1}^{N} |\mathbf{a}^\top \mathbf{x}_i - \mathbf{b}_i|$$

where $\mathbf{a} = (a_0, a_1, a_2, a_3)^\top$, $\mathbf{x}_i = (1, x_i, x_i^2, x_i^3)^\top$ and $\mathbf{b}_i = sin\ x_i$. Stacking the individual $\mathbf{x}_i^\top$s as rows of X and b_is as entries of $\mathbf{b}$, we get the following objective:

$$\underset{\mathbf{a} \in \mathbb{R}^{k+1}}{\text{minimize}} \quad \|X\mathbf{a} - \mathbf{b}\|_1$$

This is a convex optimization problem, but not a linear programming problem (LP) because the objective function is not linear in $\mathbf{a}$. The problem is equivalent to:

$$\underset{\mathbf{a} \in \mathbb{R}^{k+1}}{\text{minimize}} \quad \sum_{i=1}^{N} |X\mathbf{a} - \mathbf{b}|_i$$

Setting, $|X\mathbf{a} - \mathbf{b}|_i \leqslant \mathbf{z}_i$, objective thus obtained is:

$$\begin{aligned} \underset{\mathbf{a} \in \mathbb{R}^{k+1}, \mathbf{z} \in \mathbb{R}^N}{\text{minimize}} \quad & \sum_{i=1}^{N} \mathbf{z}_i \\ \text{subject to} \quad & |X\mathbf{a} - \mathbf{b}|_i \leqslant \mathbf{z}_i, \forall i \in [N] \end{aligned}$$

Simplifying it further, we get:

$$\begin{aligned} \underset{\mathbf{a} \in \mathbb{R}^{k+1}, \mathbf{z} \in \mathbb{R}^N}{\text{minimize}} \quad & \mathbf{1}^\top \mathbf{z} \\ \text{subject to} \quad & [X\mathbf{a} - \mathbf{b}]_i \leqslant \mathbf{z}_i \\ & -[X\mathbf{a} - \mathbf{b}]_i \leqslant \mathbf{z}_i, \forall i \in [N] \end{aligned}$$

So, the final optimization problem is:

$$\begin{aligned} \underset{\mathbf{a} \in \mathbb{R}^{k+1}, \mathbf{z} \in \mathbb{R}^N}{\text{minimize}} \quad & \mathbf{1}^\top \mathbf{z} \\ \text{subject to} \quad & X\mathbf{a} - \mathbf{b} \leqslant \mathbf{z} \\ & -X\mathbf{a} + \mathbf{b} \leqslant \mathbf{z} \end{aligned}$$

```
------------------------------------------------------------------
 number of iterations   = 10
 primal objective value = -1.31208543e+00
 dual   objective value = -1.31208543e+00
 gap := trace(XZ)       = 1.06e-09
 relative gap           = 2.94e-10
 actual relative gap    = 2.88e-10
 rel. primal infeas (scaled problem)   = 2.37e-14
 rel. dual     "        "       "      = 1.00e-12
 rel. primal infeas (unscaled problem) = 0.00e+00
 rel. dual     "        "       "      = 0.00e+00
 norm(X), norm(y), norm(Z) = 4.3e+00, 1.1e+00, 8.3e-01
 norm(A), norm(b), norm(C) = 1.3e+02, 7.7e+01, 7.2e+00
 Total CPU time (secs)  = 0.11
 CPU time per iteration = 0.01
 termination code       =  0
 DIMACS: 2.5e-14  0.0e+00  2.4e-12  0.0e+00  2.9e-10  2.9e-10
------------------------------------------------------------------

------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): +1.31209
```

Figure 12: Details of solution of Question 5

Clearly, this is a general form LP. The details of the solution is given in Figure 12. Solving this LP problem, we obtain $\mathbf{a} = (0, 0.8855, 0, -0.0975)^\top$. So, the closest cubic approximation of $f(x) = sin\ x$ in $I = [-\pi, \pi]$ is given as:

$$g(x) = 0.8855x - 0.0975x^3$$

The plot of $f(x) = sin\ x$ and $g(x)$ is presented in Figure 13.

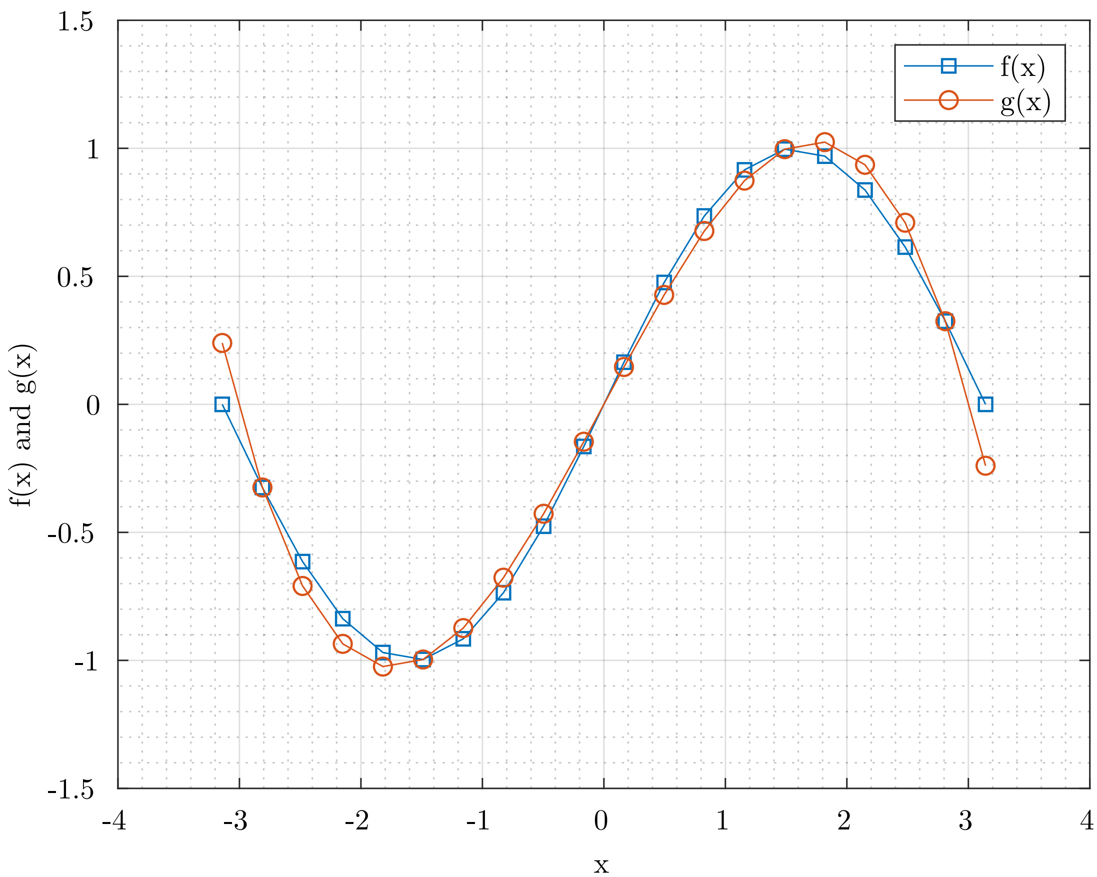

Figure 13: Plot of $f(x) = sin\ x$ and its cubic approximation $g(x)$

6 Problem 6

In this problem, we are given a correlation matrix $\hat{C}$ that has noise added to it. We need to find a matrix C such that C is symmetric, positive semi-definite and the diagonal entries of C are 1. So, the problem can be formalized as follows:

$$
\begin{aligned}
\underset{C}{\text{minimize}} \quad & \left\| C - \hat{C} \right\|_F \\
\text{subject to} \quad & C \succeq 0 \\
& C = C^\top \\
& C_{ii} = 1, \forall i \in \{1, 2, 3, 4\}
\end{aligned}
$$

where $\|\cdot\|_F$ is the Frobenius norm. The details of the solving the optimization problem is given in Figure 14. The optimal value of C is given in (5). We can verify that the obtained $C \succeq 0$ as the eigen values of C are 0, 0.78, 1.05 and 2.16.

```
-------------------------------------------------------------------
 number of iterations   =  8
 primal objective value = -3.58138375e-01
 dual   objective value = -3.58138390e-01
 gap := trace(XZ)       = 1.64e-08
 relative gap           = 9.54e-09
 actual relative gap    = 8.88e-09
 rel. primal infeas (scaled problem)   = 1.71e-11
 rel. dual     "        "       "      = 3.00e-10
 rel. primal infeas (unscaled problem) = 0.00e+00
 rel. dual     "        "       "      = 0.00e+00
 norm(X), norm(y), norm(Z) = 1.9e+00, 1.2e+00, 2.6e+00
 norm(A), norm(b), norm(C) = 6.0e+00, 2.0e+00, 3.7e+00
 Total CPU time (secs)  = 0.58
 CPU time per iteration = 0.07
 termination code       =  0
 DIMACS: 1.7e-11  0.0e+00  5.6e-10  0.0e+00  8.9e-09  9.5e-09
-------------------------------------------------------------------

------------------------------------------------------------
Status: Solved
Optimal value (cvx_optval): +0.358138
```

Figure 14: Details of solution of Question 6

$$
C = \begin{bmatrix} 1 & -0.6427 & -0.0053 & -0.7871 \\ -0.6427 & 1 & 0.1381 & 0.1661 \\ -0.0053 & 0.1381 & 1 & 0.3483 \\ -0.7871 & 0.1661 & 0.3483 & 1 \end{bmatrix} \tag{5}
$$