**תרגיל בית 8 – מודלים לא לינאריים בחקר ביצועים**

**מגישים :**

**אלעד בוכריס – 206202426**

**משה דידי – 311395834**

שאלה 1 :

1. נתונה בעיית האופטימיזציה :

*כעת, נבדוק האם פונקציית המטרה קמורה על ידי חישוב את מוגדרות ההסיאן:*

*ניתן לראות כי המינור הראשון שווה ל-1 ולכן חיובי, המינואר השני שווה ל-1 ולכן חיובי והמינור השלישי שווה ל-1 ולכן חיובי.*

*כלומר, ההסיאן מוגדר חיובית ולכן הפונקציה קמורה.*

*כעת, נראה כי האילוץ הראשון קמור על ידי כך שנראה כי מדובר על קבוצת רמה של פונקציה קמורה.*

*נקרא לפונקציה הקמורה המדוברת .*

*נסתכל על החלק הראשון :*

*היא פונקציה קמורה ולכן, גם היא פונקציה קמורה.*

*החלק השני:*

*בנוסף, מתקיים כי קמורה כריבועית מעל לינארית ולכן קמורה גם היא.*

*לכן, קמורה כסכום של קמורות.*

*ולכן, האילוץ הוא ולכן קמור.*

*בנוסף, האילוצים מייצגים חצי מרחב ולכן קמורים.*

*כלומר, בעיית האפוטימיזציה היא בעיית תכנות קמור.*

*מכיוון שמתקיים :*

*הקוד לשאלה:*

def q1\_a():  
 n = 3  
 A = np.array([[1, 1, 0],  
 [1, 2, 0],  
 [0, 0, 1]])  
 b = np.array([3, -4, 0])  
 st1 = np.array([[0.25, 2, 0],  
 [np.sqrt(31 / 16), 0, 0],  
 [0, 0, 0]])  
 b\_st1 = np.array([0, 0, 2])  
  
 x = cp.Variable(tuple([n]))  
 prob = cp.Problem(cp.Minimize(cp.quad\_form(x, A) + b.T @ x),  
 [cp.norm(st1 + b\_st1) + cp.quad\_over\_lin(np.array([1, -1, 1]) @ x + 1,  
 np.array([1, 1, 0]) @ x) <= 6,  
 x >= 1])  
 prob.solve()

*פלט:*

*The optimal value is 5.00000014872752*

*The optimal x is*

*[1. 1.0000002 0.99999986]*

1. *נתונה הבעיה :*

*נשים לב כי בבעיית מקסימום, על פונקצית המטרה להיות קעורה. נראה זאת על ידי כך שנראה כי מינוס פונקציית המטרה היא פונקציה קמורה.*

*נגדיר*

*כאשר היא פונקציית המטרה.*

*לכן:*

*ידוע כי היא פונקציה קמורה ותחת החלפת משתנים לינארית נשארת קמורה, לכן, קמורה.*

*פונקציה לינארית ולכן קמורה.*

*היא פונקציה קמורה, לכן, תחת החלפת משתנים לינארית נשארת קמורה, לכן קמורה.*

*קמורה כסכום של קמורות ולכן קעורה.*

*נסתכל על האילוצים :*

*קמורה פונקציה קמורה כריבועית מעל לינארית. ()*

*נשים לב כי קמורה כריבועית מעל לינארית ().*

*פונקציה קמורה ומונוטונים עולה בתחום לכן אם מרכיבים אותה על פונקציה קמורה הפונקציה נשארת קמורה. לכן קמורה.*

*כלומר, האילוץ הראשון הוא קבוצת רמה של פונקציה קמורה ולכן קמור.*

*לינארי ולכן קמור.*

*הוא חצי מרחב ולכן קמור.*

לכן, זוהי בעיית תכנות קמור.

קוד לשאלה:

def q1\_b():  
 n = 2  
 A = np.array([[1, 1],  
 [1, 1]])  
 x = cp.Variable(tuple([n]))  
 obj = cp.Maximize(-cp.quad\_form(x, A) + cp.sqrt(2\*x[1] + 5) - 2 \* x[0] - 3 \* x[1])  
 constraints = [cp.quad\_over\_lin(x[0], x[0] + x[1])  
 + cp.power(cp.quad\_over\_lin(x[0], x[1]) + 1, 8) <= 100,  
 cp.sum(x) >= 4,  
 x[1] >= 0]  
 prob = cp.Problem(obj, constraints)  
 try:  
 result = prob.solve()  
 except SolverError:  
 result = prob.solve(solver=SCS)

פלט:

*The optimal value is -23.39467499280918*

*The optimal x is*

*[1.41686665 2.58313335]*

שאלה 2 :

נתון כי

מיקום נקודת ביקוש הוא ומספר ההזמנות השבוע הממוצע בכל נקודה הוא .

תהי פונקציה המראה את זמן הטיסה הממוצע בדקות של הרחפן מנקודה ל-.

וכי קבוע. בנוסף, תהי עלות דקת טיסה. בנוסף, נתון כי כל רחפן יכול לשאת הזמנה אחת בלבד בכל רגע וכי יש מספר בלתי מוגבל של רחפנים.

1. *יש לנסח בעיית תכנות קמור, המוצאת את המיקום המיטבי למיקום המחסן במובן הממזער את עלות הטיסות השבועיות.*

*משתנה ההחלטה הוא המייצג את מיקום המחסן.*

*נראה כי פונקציית המטרה קמורה*

*כל הפרמטרים חיוביים וכי קמורה, החלפת משתנים לינארית לא משפיעה על הקמירות ולכן*

*קמורה. בנוסף, הכפלתה בסקלר חיובי משאירה את הפונקציה קמורה וכל פונקציית המטרה קמורה כסכום של קמורות.*

1. כעת, נתון כי עבור כל דקת איחור מעבר לזמן ניתנת הנחה של שקלים לדקה.

בנוסף, עבור כל דקת איחור מעבר לזמן ניתנת הנחה של .

כעת, פונקציית המטרה תהיה :

החלק הראשון קמור כמו בסעיף א', החלק השני הוא פונקציית מקסימום של פונקציות קמורות ולכן קמור וגם כך החלק השלישי. לכן כל הפונקציה קמורה כסכום של פונקציות קמורות.

1. קוד:

def q2\_c():  
 m, n, outliers\_num = 50, 2, 10  
 np.random.seed(314)  
 A = 3000 \* np.random.rand(n, m)  
 A[:, : outliers\_num] += 3000  
 p = (10 \* np.random.rand(m, 1) + 10).round()  
 alpha, gamma = 0.01, 1.2  
 eta1, eta2 = 20, 30  
 mu1, mu2 = 2, 5  
  
 x1 = cp.Variable(tuple([n]))  
 prob1 = cp.Problem(cp.Minimize(gamma \* alpha \* (cp.sum([p[i] \* cp.norm(A.T[i] - x1) for i in range(m)]))))  
 prob1.solve()  
 print("\nThe optimal value is", prob1.value)  
 print("The optimal x is")  
 print(x1.value)  
  
 x2 = cp.Variable(tuple([n]))  
 prob2 = cp.Problem(cp.Minimize(gamma \* alpha \*  
 (cp.sum([p[i] \* cp.norm(A.T[i] - x2) +  
 cp.max(cp.vstack([0, alpha \* cp.norm(A.T[i] - x2) - eta1])) \* mu1 + cp.max(cp.vstack([0, alpha \* cp.norm(A.T[i] - x2) - eta2])) \* (mu2 - mu1) for i in range(m)]))))  
 prob2.solve()  
  
 print('problem with delays')  
 print("\nThe optimal value is", prob2.value)  
 print("The optimal x is")  
 print(x2.value)  
  
 plt.scatter(x1.value[0], x1.value[1], label="Without penalty", color='red')  
 plt.scatter(x2.value[0], x2.value[1], label="With penalty", color='blue')  
 plt.scatter(A[0], A[1], label="Order positions", color='green')  
 plt.legend()  
 plt.show()

פלט:

*The optimal value is 16382.693743323161*

*The optimal x is*

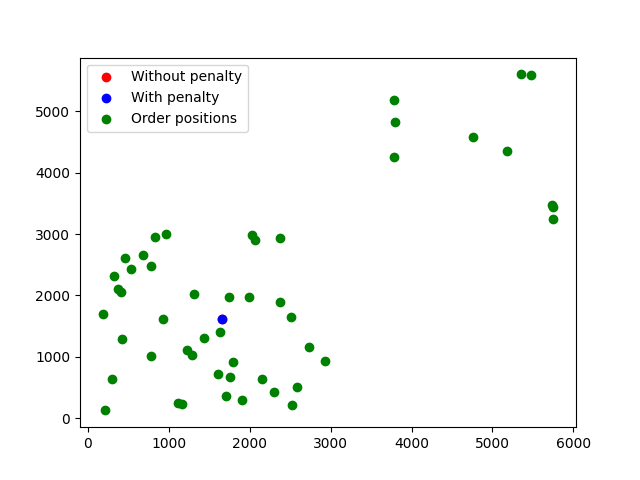
*[1662.00217786 1613.75204818]*

*problem with delays*

*The optimal value is 16393.782453669803*

*The optimal x is*

*[1662.88836343 1614.98496533]*



שאלה 3:

נתונות הנקודות כך ש- .

נרצה למצוא משטח ריבועי מפריד כך שאם אזי ערך פונקציית המשטח אי שלילי ואחרת שלילי. עבור סימטרית, ו- .

יש לנסח בעיית תכנות קמור המוצאת את המקדמים כך שהנורמה הספקטרלית היא

היא מזערית.

משתני ההחלטה הם .

ניסוח הבעיה :

ראשית, פונקציית המטרה היא נורמה ולכן קמורה.

כעת, נתייחס לאילוץ. נוכיח כי הוא קמור לפי הגדרה:

לכן פונקציה זו לינארית ובפרט לינארית, לכן האילוץ הנגזר ממנה הוא קבוצה קמורה.

כלומר, זו בעיית תכנות קמור.

שאלה 4:

1. קוד:

def grad\_proj(f, gf, proj, t, x0, eps):  
 fs = []  
 xk = x0  
 fs.append(f(x0))  
 xk\_1 = proj(xk - t \* gf(xk))  
 fs.append(f(xk\_1))  
 while np.linalg.norm(xk\_1 - xk) > eps:  
 xk = xk\_1  
 xk\_1 = proj(xk - t \* gf(xk))  
 fs.append(f(xk\_1))  
 return xk\_1, fs

1. יש למצוא את ההיטל האורתוגונלי של על הישר :

יש למצוא את

*כלומר, לפתור את בעיית האופטימיזציה :*

*ניתן לראות כי המרחק הכי קצר מנקודה p לישר הוא הקו האנך המחבר בין הנקודה p לישר.*

*לכן, נמצא את הקו האנך ל- הכולל את נקודה .*

*נניח שזהו ישר אזי*

*מכיוון שעבור ישרים אנכים צריך להתקיים :*

*בנוסף, אנו יודעים שנקודה צריכה להיות על האנך לכן :*

*לכן, משוואת הישר האנך ל- העובר דרך היא :*

*כעת, נמצא את נקודת החיתוך בין הישרים:*

1. *קוד:*

def proj\_section\_b():  
 return lambda x: np.array([(1 - (x[1] - x[0])) / 2, (1 + (x[1] - x[0])) / 2])

1. *נתונה הבעיה הבאה :*

*ניתן לראות כי פונקציית המטרה קמורה וגזירה ברציפות כ וכי הקבוצה קמורה כלינארית.*

*לכן, ניתן להפעיל את שיטת היטל הגראדינט. בנוסף, הגראדינט ליפשיציאני עם קבוע 2, לכן, סדרת הערכים הנוצרים משיטה זו היא מונוטונית לא עולה, כמו כן, אנו יודעים כי הפונקציה חסומה מלמטה ולכן מובטחת התכנסות.*

*קוד:*

def q4():  
 return grad\_proj(f=lambda x: np.square(np.linalg.norm(x)),  
 gf=lambda x: 2 \* x,  
 proj=proj\_section\_b(),  
 t=0.5,  
 x0=np.array([100, 100]),  
 eps=10 \*\* -8)

*פלט:*

*(array([0.5, 0.5]), [20000.0, 0.5000000000000001, 0.5000000000000001])*

*השיטה התכנסה תוך 2 איטרציות לנקודה (0.5,0.5).*

*שאלה 5 :*

1. *נתונים m מיקומים של חיישנים :*

*ומרחקים בין החיישנים לבין העצם .*

*כלומר, אנו יודעים כי*

*כאשר הוא מיקום העצם.*

*לכן, דרך להעריך את מיקום העצם היא לפתור את הבעיה :*

*בעיה זו לא גזירה ולא קמורה.*

*לכן, נחפש ניסוח גזיר ולא קמור של הבעיה.*

*נניח כי קיים*

*לכן*

*נראה כי ניתן למצוא עבורם (נשים לב כי לא תלוי ב-x ולכן לא משפיע על האופטימיזציה).*

*נבחר :*

*ונקבל :*

*בנוסף, אנו יודעים לפי אי שוויון קושי שוורץ כי*

*לכן, הערך המקסימלי של מתקבל עבור ה- שבחרנו, נשים לב שבפונקציה המטרה הביטוי הוא ב (-) ולכן ערך זה ממזער את הפונקציה ולא ניתן למצוא אחר המוצא פתרון טוב יותר.*

*בנוסף כלומר עומד באילוץ*

*לכן, בהינתן פתרון מיטבי לבעיית אזי פתרון מיטבי לבעיית .*

*כעת, נניח כי פתרון מיטבי עבור אזי*

1. מכיוון ש- הוא הממזער הוא ישאף להקטין את הביטוי כמה שניתן, לכן, צריך להקטין כמה שאפשר את וזה מתאפשר עבור מאותו שיקול שהראנו קודם.

לכן, ה שמצאנו ממזער את הביטוי ולכן זה שקול למציאת פתרון מיטבי לבעית .

1. *נראה כי לא קמורה עבור .*

*פונקציית המטרה היא :*

*נניח כי*

*נבחר שתי נקודות בהן*

*מתקיים כי :*

*לכן, הפונקציה לא קמורה.*

*נראה כי לא קמורה:*

*עבור ו- מתקיים:*

*נחשב את ההסיאן :*

*לכן מתקיים כי לכן, ההסיאן לא מוגדר אי שלילית והפונקציה אינה קמורה.*

*עבור נקבל פונקציה קמורה .*

1. *נגזור את הפונקציה לפי x ונקבל :*

*נגזור לפי ונקבל :*

*לכן*

*נמצא את קבוע ליפשיץ.*

*ולכן :*

1. *ניתן להפעיל את שיטת היטל הגראדינט מכיוון ש-f גזירה ברציפות וכי הקבוצה  
    היא קמורה ככדור.*

*מובטחת התכנסות עם צעד קבוע כי הפונקציה חסומה מלמטה ויש לה גראדינט ליפשיציאני.*

*נראה כי הפונקציה חסומה מלמטה :*

*בסעיף א' הראנו כי חסומה מלמטה על ידי ובנוסף חסומה מלמטה על ידי 0, לכן פונקציה חסומה מלמטה.*

*בנוסף, מכיוון שקבוצת האילוצים סגורה וחסומה וכי f רציפה (כסכום של רציפות)  
אזי לפי משפט ויירשטראס קיים פתרון מיטבי.*

*מכיוון שהפונקציה אינה קמורה, לא מובטח כי תהיה התכנסות למינימום גלובלי.*