עבודה 1 – שיטות אופטימיזציה ויישומיהן

**שאלה 1:**

1. עבור : הוקטור ימקסם את , זאת מכיוון שהוכחנו בהרצאה שזה שווה ל שזה הסכום של העמודה בעלת הסכום המקסימלי. לכן אם ניקח את העמודה בעלת הסכום המקסימלי, נמצא את ערך זה, והוקטור x הנ"ל מקיים את המשוואה . נשים לב כי , וכן ואכן מתקיים .

עבור : הוקטור ימקסם את . זאת מכיוון שהוכחנו בהרצאה שזה שווה ל שזה הסכום של השורה בעלת הסכום המקסימלי. ניתן לראות כי השורה השנייה בעלת הסכום המקסימלי בערך מוחלט, ועל מנת לקבל סכום זה, 18, יש לשים מינוס אחד בעמודה השלישית. באמצעות הוקטור x, נקבל כי ואכן מתקיים כי וכן ואז .

1. *נרצה למצוא את x שימקסם את הביטוי:*

*בהוכחה מהכיתה עבור ביטוי זה, הראנו שעבור נקבל את הביטוי המקסימלי (כאשר הוא וקטור עצמי המתאים לערך עצמי . נחשב בקוד את הע"ע של , נמצא את הוקטור עצמי המתאים לערך עצמי המקסימלי על מנת למצוא את x. הקוד:*

*import numpy as np*

*from numpy import linalg as LA*

*A = np.asarray([[1,2,3,4], [2,4,-4,8], [-5,4,1,5], [5,0,-3,-7]])*

*ATA = (A.transpose()@A)*

*eig = LA.eig(ATA)*

*indexOfMaxEig = np.argmax(eig[0])*

*v=np.asarray([[eig[1][0][indexOfMaxEig]],[eig[1][1][indexOfMaxEig]],[eig[1][2][indexOfMaxEig]],[eig[1][3][indexOfMaxEig]]])*

*print("v",v)*

*v [[-0.29618621]*

*[ 0.35616716]*

*[ 0.06730298]*

*[ 0.88367923]]*

*# calculate L2 norm of A*

*AVl2 = LA.norm(A@v)*

*vl2 = LA.norm(v)*

*Al2 = AVl2/vl2*

*print("AV", Al2*

*AV 13.858100376465327)*

*קיבלנו כי הוקטור x שממקסם את הביטוי הינו*

***שאלה 2:***

1. *נוכיח כי באמצעות הכלה דו כיוונית.*

*⊆: יהי אזי ומכאן ולכן*

*⊇: יהי אזי . מהרמז בשאלה מתקיים כי .*

*נסמן וכן מחוקי transpose מתקיים ולכן נקבל כי*

1. *נוכיח כי :*

*נסמן . מהרמז מתקיים כי*

*נשים לב כי ולכן*

*מסעיף א, ולכן נקבל כי כנדרש.*

1. המשוואה הנורמלית -

*נסמן . מכאן ש- . מסעיף קודם,*

*ולכן כלומר ולכן קיים בהכרח כך ש-*

*כנדרש.*

***שאלה 4:***

*המשוואות הנורמליות הן : :*

*נותר לנו למצוא את :*

*הקוד:*

*A = np.asarray([[2,1,2], [1,-2,1], [1,2,3], [1,1,1]])*

*AT=A.transpose()*

*ATA = (AT@A)*

*print("ATA",ATA)*

*ATA [[ 7 3 9]*

*[ 3 10 7]*

*[ 9 7 15]]*

*b = np.asarray([[6],[1],[5],[2]])*

*L = LA.cholesky(ATA)*

*print("L", L)*

*L [[2.64575131 0. 0. ]*

*[1.13389342 2.9519969 0. ]*

*[3.40168026 1.06465462 1.51495279]]*

*LLT = (L@L.transpose())*

*ATb = (AT@b)*

*print("ATb",ATb)*

*ATb [[20]*

*[16]*

*[30]]*

*LLTinv = LA.inv(LLT)*

*x = LLTinv@ATb*

*print("cholesky x" ,x)*

*cholesky x [[1.7]*

*[0.6]*

*[0.7]]*

1. *באמצעות פירוק QR אנו יודעים כי :*

*הקוד:*

*q, r = np.linalg.qr(A)*

*print("Q",q)*

*Q [[-0.75592895 0.04839339 0.41120147]*

*[-0.37796447 -0.82268766 -0.38955929]*

*[-0.37796447 0.53232731 -0.7574764 ]*

*[-0.37796447 0.19357357 0.32463274]]*

*print("R",r)*

*R [[-2.64575131 -1.13389342 -3.40168026]*

*[ 0. 2.9519969 1.06465462]*

*[ 0. 0. -1.51495279]]*

*x2 = LA.inv(r)@q.transpose()@b*

*print("QR x", x2)*

*QR x [[1.7]*

*[0.6]*

*[0.7]]*

*באמצעות פירוק SVD אנו יודעים כי כאשר .*

*הקוד:*

*(U,S,V) = LA.svd(A, full\_matrices=False)*

*print("U",U)*

*U [[-0.58403081 0.19965059 -0.60158523]*

*[-0.06246735 0.95511019 0.23514023]*

*[-0.73539957 -0.20695462 0.64525752]*

*[-0.33792502 -0.07123168 -0.40797919]]*

*print("S", S)*

*S [4.96978399 2.53339065 0.93977598]*

*V=V.transpose()*

*print("V",V)*

*V [[-0.46357217 0.42481607 -0.77758096]*

*[-0.45632169 -0.8667083 -0.20146273]*

*[-0.75952047 0.26143455 0.59563464]]*

*Utb = (U.transpose())@b*

*y = np.asarray([Utb[0]/S[0], Utb[1]/S[1], Utb[2]/S[2]])*

*x3 = V@y*

*print("SVD x",x3)*

*SVD x [[1.7]*

*[0.6]*

*[0.7]]*

*קיבלנו בשלושת הדרכים כי הוקטור*

1. *קיבלנו כי וזאת מכיוון שמתקיים : ואז :*

*וזה בדיוק ה-x שמצאנו שמקיים את המשוואה.*

*הקוד:*

*r = (A@x) - b*

*print("r", r)*

*r [[-6.00000000e-01]*

*[ 2.00000000e-01]*

*[ 3.55271368e-15]*

*[ 1.00000000e+00]]*

*Atr = AT@r*

*print("ATr", Atr)*

*ATr [[2.75335310e-14]*

*[1.42108547e-14]*

*[3.46389584e-14]]*

1. *על מנת למזער את הערך של כלומר להביא לכך ש- נשתמש בweight least square:*

*מצאנו כי עבור נקבל r כזה. הקוד:*

*W = np.asarray([[808,0,0,0], [0,1,0,0], [0,0,1,0], [0,0,0,1]])*

*ATWA = AT@W@A*

*ATWAinv = LA.inv(ATWA)*

*xw = ATWAinv@AT@W@b*

*rw = (A@xw)-b*

*print("r w",rw)r w [[-9.99191130e-04]*

*[ 2.69115478e-01]*

*[-4.79616347e-14]*

*[ 1.34557739e+00]]*

***שאלה 5:***

1. *GS:*

*def GramShmidthQR(A):*

*n=len(A[0])*

*R = np.zeros((n,n))*

*R[0][0] = LA.norm(A[:,0])*

*q1 = A[:,0]/R[0][0]*

*Q = [q1]*

*for i in range(1,n):*

*qi = A[:,i]*

*for j in range(i):*

*R[j][i] = reduce(lambda x, y: x + y, Q[j]\*A[:,i])*

*qi = qi- R[j][i]\*Q[j]*

*R[i][i] = LA.norm(qi)*

*qi = qi/R[i][i]*

*Q.append(qi)*

*return R,np.asarray(Q).transpose()*

*MGS:*

*def ModifiedGramShmidthQR(A):*

*n=len(A[0])*

*R = np.zeros((n,n))*

*R[0][0] = LA.norm(A[:,0])*

*q1 = A[:,0]/R[0][0]*

*Q = [q1]*

*for i in range(1,n):*

*qi = A[:,i]*

*for j in range(i):*

*R[j][i] = reduce(lambda x, y: x + y, Q[j]\*qi)*

*qi = qi- R[j][i]\*Q[j]*

*R[i][i] = LA.norm(qi)*

*qi = qi/R[i][i]*

*Q.append(qi)*

*return R, np.asarray(Q).transpose()*

*תוצאות ארבעת החישובים:*

*עבור GS :*

*עבור :*

*עבור MGS :*

*עבור :*

1. *עבור GS : 5.319287782567757e-16*

*עבור : 0.7071067811865477*

*עבור MGS : 4.987305196443834e-16*

*עבור : 1.1547005383855976e-10*

*קיבלנו כי כאשר אפסילון הוא 1, ההבדל בין האלגוריתמים הוא מינורי, אך כאשר אפסילון קטן מאוד, MGS טוב יותר. זאת מכיוון שבאלגוריתם MGS אנחנו מחשבים את הוקטורים בR לפי הוקטורים ב-Q, ואילו באלגורתים GS אנחנו מחשבים את הוקטורים ב-R לפי הוקטורים בA. יש שגיאה נומרית קטנה בחישוב לפי A, ואז נקבל שלכל וקטור ב-R שנחשב תהיה שגיאה וסך השגיאה תהיה הסכום שלהם. לעומת זאת, כאשר אנו מחשבים את הוקטורים ב-R לפי הוקטורים ב-Q, כל הוקטורים ב-Q אורותוגנלים זה לזה, וגם אם היו שגיאות, הן יהיו יחסית קטנות יותר.*

***שאלה 6:***

1. *לא להגשה*
2. *צ"ל:*

*מסעיף א' אנו יודעים כי*

*נציב במשוואה ונקבל :*

1. *צ"ל :*

*מהסעיף הקודם אנו יודעים כי וכן נתון כי .*

*נוכיח כי לכל מתקיים :*

*כאשר המעבר האחרון נכון מאחר וU מטריצה אורתוגונלית ולכן עמודות המטריצה מהווים בסיס אורתונורמלי, לכן לכל מתקיים וכן עבור*

*ולכן כנדרש.*

1. *נוכיח כי הפתרון ל- ניתן ע"י המשוואה :*

*נגדיר*

*נזכור כי כמו שלמדנו בהרצאה, ולכן*

*נגזור בדומה לאיך שגזרנו בהרצאה ונקבל:*

*נשווה את ונקבל:*

*כנדרש.*

*כעת נוכיח כי המטריצה חיובית לחלוטין:*

*נשים לב כי המטריצה מטריצה סימטרית, שכן תמיד סימטרית, וכן מטריצה אלכסונית ובפרט סימטרית, וכן סכום מטריצות סימטריות הוא סימטרי. לכן נותר להוכיח שכל הערכים העצמיים של הם אי-שליליים, ומכאן ינבע שהמטריצה חיובית לחלוטין.*

*יהי ע"ע של המטריצה . אזי מתקיים . נוכיח כי :*

*נעביר אגף ונקבל כי :*

*קיבלנו כי ע"ע של המטריצה ומאחר שהמטריצה חיובית לחלוטין מתקיים כי כל הערכים העצמיים שלה אי שליליים, ולכן . נזכור כי ונקבל כי כנדרש.*

1. *אנו יודעים כי . נציב במשוואה ונקבל:*

*כעת נכפול משמאל ב- ונקבל:*

*נשים לב כי המטריצה היא סכום של שתי מטריצות אלכסוניות, ולכן אלכסונית בעצמה, ומכאן שהיא הפיכה ונוכל לכפול משמאל בהופכית שלה ונקבל:*

*נכפול משמאל ב-V ונקבל:*

*נתבונן במטריצה :*

*מסעיף ב' נקבל כי הערכים על האלכסון של מקיימים*

*ומסעיף ג' נקבל כי כנדרש.*

1. *בדוגמא הנתונה, אנו יודעים כי כאשר A היא מדרגה מלאה. הניסיון הראשון היה למצוא את x באמצעות פתרון של ולפי סעיף c נקבל כי*

*הניסיון השני היה למצוא את x באמצעות פתרון של*

*לפי סעיף e נקבל כי .*

*נתון בשאלה כי הרעש תואם לוקטור סינגולרי עם ערכים סינגולריים .*

*בניסיון הראשון קיבלנו כי התוצאה הייתה רועשת מאוד, וזאת מכיוון שהע"ע של A מאוד קטנים, ניתן לראות זאת גם מהמשואה, שכאשר הערכים העצמיים קטנים מאוד, בהתאם גם הערכים הסינגולריים קטנים מאוד, וככל שהם קטנים יותר הפתרון x עם המשוואה גדול יותר ובכך פחות טוב.*

*בניסיון השני קיבלנו כי התוצאה הייתה די טובה, וזאת מכיוון שעל אף שהערכים הסינגולריים קטנים מאוד, במשוואה יש לנו את הפרמטר שמאזן את התוצאה (מונע מהמכנה להיות קטן מאוד ביחס למונה).*

***שאלה 7:***

1. *)על מנת למצוא את K נרצה למצוא השמה לנעלמים . עבור זוג אחד מתקיים כי :*

*נקבל את מערכת המשוואות הבאה:*

*קיבלנו מערכת של שתי משוואות עם 4 נעלמים.*

*נשים לב שעם נקודה אחת נקבל בכל משוואה אינסוף פתרונות, ועם שתי נקודות נקבל פתרון יחיד (עם שתי נקודות נקבל 4 משוואות עם 4 נעלמים, מה שייתן לנו פתרון יחיד), ולכן המספר המינימלי של זוגות הוא 2.*

1. *אנו רוצים למצוא את K ולכן נרצה לפתור את מערכת המשוואות מהסעיף הקודם :*

*נכתוב את מערכת המשוואות בצורה מטריציאלית ונקבל:*

*כעת עבור n זוגות כאלה, (בדומה לאיך שפתרנו את רגרסיה לינארית) נוכל להציג את מערכת המשוואות ככפל מטריצות:*

*זוהי בעיה מהצורה ולכן נוכל לפתור זאת באמצעות least square כאשר*