בס"ד

פרויקט גמר – מבוא לאופטימיזציה

מתי עדיפה Ellipsoid Method על פני Simplex ולמה?

מגישים: לירון חיים 206234635 סתיו לידור 207299785

חלק א' – תיאור העבודה

בעבודה זו נדון על שני אלגוריתמים ידועים לפתירת בעיות ליניאריות וקבלת פתרון אופטימלי. בעיות אלו מורכבות משלושה גורמים: מטרה (objective), אילוצים (constraints) וכמות משאבים (resources). נשווה בין שתי השיטות לפי האלגוריתמים עליהן מבוססות.

שיטת Simplex

האלגוריתם לשיטה זו עובד כאשר יש שתי הנחות יסוד:

1. כל האילוצים הם מהצורה של קטן-שווה, כלומר הגבלה של משאבים על ידי חסם עליון.
2. כל המשתנים בצידו הימני של אי-השוויון הם מספרים חיוביים.

שלבי האלגוריתם מבוססים על טבלת tableau ופעולות שורה אלמנטריות עליה. זאת עד עצירה לפי תנאי וקבלת פתרון אופטימלי (או תשובה כי אין פתרון). לא נפרט מעבר לכך שהרי למדנו את האלגוריתם בפירוט במהלך הקורס.

שיטת Ellipsoid

תהי בעיה במימד n. האלגוריתם זה גם כן עובד עם הנחות יסוד:

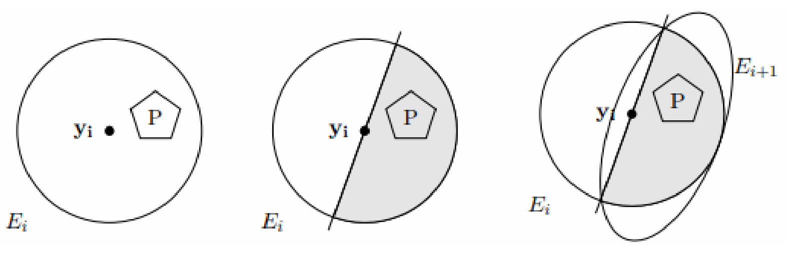
1. כל האילוצים הם מהצורה של קטן-שווה, כלומר הגבלה של משאבים על ידי חסם עליון.

אילוצים אלו מרכיבים פאון P במרחב, פאון זה מכיל את כלל הפתרונות האפשריים לבעיה.

2. קיים כדור B המורכב מנקודת מרכז 0 (במימד n) ורדיוס R, כש- הפאון P מוכל בתוך כדור B.
3. קיים כדור D המורכב מנקודת מרכז c (במימד n) ורדיוס r כך ש- הכדור שנוצר על ידי c ו-r מוכל כולו בתוך הפאון P.

כעת האלגוריתם פועל בצורה הבאה:

אם נקודת המרכז של הכדור/אליפסה נמצאת בתוך הפאון אז סיימנו ומחזירים את הנקודה בתור התשובה. אחרת, נעביר ישר דרך מרכז הכדור אשר יחתוך את הכדור בחצי. בהכרח האחד החצאים לא מכיל את הפאון P, נבחר בחצי השני ונקיף אותו באליפסה (נבחר את הישר החותך להיות מקביל לאחד מהישרים המרכיבים את P, כך נדע שלא נחתוך את P). כעת נחזור על השלבים ונבדוק האם מרכז האליפסה נמצא בתוך P. כמו כן, בכל איטרציה, נבדוק האם נפח האליפסה המתקבלת קטן מהנפח של כדור D. במידה וכן, נעצור ונחזיר תשובה כי אין פתרון.



זו שיטת האליפסה. על מנת למצוא פתרון אופטימלי ניאלץ להמשיך לעבוד ולהפעיל את אלגוריתם Interior-Point. זה מניח כי אכן קיים פתרון אפשרי לבעיה, לכן נשתמש בשיטת האליפסה כדי למצוא אותו (אם כי יש שיטות נוספות למצוא פתרון אפשרי כלשהו, נקודה בתוך הפאון).

מטרת ההשוואה

נשאלת השאלה: אם קיימות שתי שיטות למציאת פתרון, מדוע צריך את שתיהן? מתי עדיפה האחת על פני השנייה?

נרצה לבדוק ולענות על השאלות הללו.

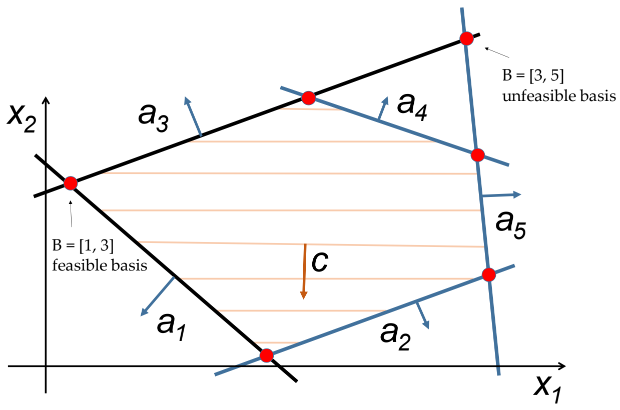
אופן ההשוואה

על מנת למדוד את הצורך בכל שיטה, נתבונן בפלט הסופי שכל שיטה מספקת לנו ונבדוק האם הפתרון פרקטי עבור הצורך שלנו. נחקור את סיבוכיות זמן הריצה של כל שיטה ונשווה ביניהן.

<u>חלק ב׳ – תהליך הבדיקה ותוצאות</u>

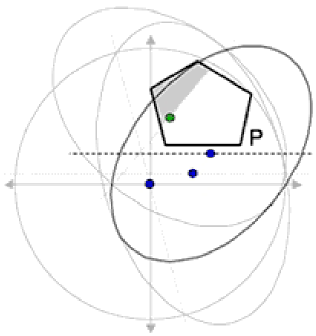
פלט האלגוריתמים

אלגוריתם Simplex מספק לנו פתרון אופטימלי לבעיה שלנו. ננסה להמחיש בעזרת איור כיצד האלגוריתם עובד:

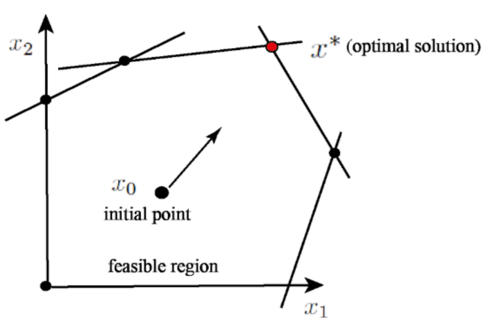


מה שניתן לראות באיור הנ״ל הוא מרחב הפתרונות לבעיה כלשהי בה נרצה למזער מטרה כלשהי c. לבעיה זו יש חמישה אילוצים (a1…5). במהלך האלגוריתם עוברים על נקודות החיתוך של האילוצים, כלומר על קודקודי הפאון של מרחב הפתרונות. כך עד שנגיע לנקודה המשיגה את המטרה שלנו תוך סיפוק כלל האילוצים.

האלגוריתם של שיטת האליפסה לעומת זאת, מספק לנו כתשובה האם קיים פתרון לבעיה ובמידה וכן – נקבל את פתרון **אפשרי** לבעיה ולאו דווקא פתרון אופטימלי. ננסה להמחיש גם כאן את הרעיון באופן ויזואלי:



נזכור כי תנאי העצירה של האלגוריתם הם קבלה של נקודת אמצע אליפסה הנמצאת בתוך הפאון P והחזרתה, או כאשר מגיעים לאיטרציה בה נפח האליפסה קטן מדי ולא יכול להכיל את כל פאון P, כלומר אין פתרון. על מנת לקבל פתרון אופטימלי לבעיה יש להפעיל אלגוריתם נוסף (Interior-point method) המקבל כקלט את פאון מרחב הפתרונות, את פלט הפתרון האפשרי ומגיע אל פתרון אופטימלי (אחד מקודקודי הפאון) על ידי התקדמות מהנקודה הפנימית אל הקודקוד.



סיבוכיות זמן ריצה

נטען כי לאלגוריתם Simplex יש סיבוכיות זמן אקספוננציאלית. קיימת בעיה ידועה על שמות קלי-מינטי (Klee-Minty) המראה כי זמן הריצה של האלגוריתם הוא $2^n - 1$ איטרציות. גרסה של הבעיה מתוארת באופן הבא:

המטרה:

$$maximize \sum_{j=1}^{d} 2^{d-j} x_j$$

האילוצים:

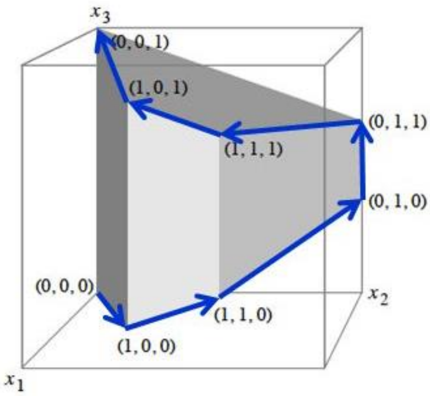
$$x_1 \leq 5$$

$$4x_1 + x_2 \leq 25$$

$$8x_1 + 4x_2 + x_3 \leq 125$$

$$\dots$$

$$\sum_{j=1}^{d-1} 2^{d-j+1} x_j + x_d \geq 5^d$$



דוגמה להיפר-קובייה הנוצרת מגרסה של בעיית Klee-Minty.

בהרצת האלגוריתם על הבעיה הנ״ל נקבל כי הוא ״מבקר״ בכל הקודקודים של פאון מרחב הפתרונות. במקרה הספציפי הזה, הפאון הוא היפר-קובייה ומורכב מ-2^n קודקודים וצלעות. לכן האלגוריתם נדרש לרוץ במשך $2^n - 1$ איטרציות עד אשר מתכנס.

על מנת לראות זאת בעצמנו, רשמנו תוכנית בפייתון אשר מריצה מספר דוגמאות לשאלות שאנו מעלים. השתמשנו בספריה הממשת את Simplex וכתבנו מספר פונקציות עזר ליצירת בעיות שונות ובדיקות קלט. אחת מהן, יוצרת את בעיית Klee-Minty מכל מימד נבחר, למען ההדגמה, הרצנו את התוכנית עם מימד 12 וקיבלנו שהאלגוריתם אכן רץ במשך 4095 איטרציות. ניתן לגשת אל קוד התוכנית דרך עמוד הgit המצוין בסוף העבודה.

מנגד, שיטת האליפסה מראה כי ניתן למצוא פתרון בסיבוכיות זמן של $\log\left(\frac{Volume(B)}{Volume(P)}\right)$, כאשר B הכדור הראשוני שלנו ו-P הוא הפאון. סיבוכיות זו פולינומית לגודל הקלט. ניתן למצוא הוכחה לכך כאן. אם נרצה לקבל אינטואיציה לכך, נזכור מה הפעולה המרכזית שלנו באלגוריתם, בכל איטרציה אנחנו חותכים את האליפסה לחצי ומקיפים אותו באליפסה קטנה יותר. אם נחלק את הנפח של אליפסה באליפסה שמגיעה אחריה נקבל: $\frac{Volume(E_{i+1})}{Volume(E_i)} < -\frac{1}{e^{2(n+1)}} < 1$. כיוון שזהו ביטוי קטן מ-1, אנחנו בעצם מקטינים את מרחב החיפוש שלנו באופן אקספוננציאלי בכל איטרציה. כעת, ניאלץ להפעיל את Interior-point method לקבלת הפתרון האופטימלי. הוכח כי לשיטה זו גם כן יש סיבוכיות זמן פולינומית, ובכך נקבל שניתן למצוא פתרון אופטימלי בסיבוכיות פולינומית לקלט.

אם כן, מדוע נרצה להשתמש ב-Simplex אם האלגוריתם כזה איטי?

אלגוריתם Simplex מתכנס מהר בהסתברות גבוהה

שאלה זו מהעמוד הקודם כמובן כבר נשאלה בעבר, מחקרים הראו כי למרות הסיבוכיות הגבוהה במקרה הכי גרוע עבור אלגוריתם Simplex, בפועל למרבית הפעמים הוא אפילו לא מתקרב לסיבוכיות הזו. על מנת לבדוק זו בעצמנו, רשמנו קוד אשר יוצר בעיות פתירות בצורה אקראית לפי מימד כרצוננו, הרצנו את האלגוריתם על כמות מסוימת של בעיות ומצאנו את כמות האיטרציות הממוצעת. אקראיות זו יכולה לדמות את הבעיות מהחיים האמיתיים.

להלן התוצאות עבור בדיקה של 100 בעיות מקסימום שונות ב12 מימדים :

Average iterations of 100 random problems (dim=12):

12.14

מספר האיטרציות הממוצע שהתקבל הוא 12.14. נזכור שעבור בעיית Klee-Minty נדרשו $2^{12}-1=\mathbf{4095}$ איטרציות כדי לפתור אותה ולהגיע לפתרון אופטימלי. זה בהחלט הבדל ניכר.

דרכים נוספות שניתן להריץ את Simplex באופן מהיר יותר

מצאנו כי עבור מספר בעיות אקראיות אנחנו נדרשים לכמות די קטנה של איטרציות, אך מה אם קיימת בעיה שאנחנו רוצים למצוא לה פתרון אופטימלי וכשמפעילים את Simplex הוא רץ בסיבוכיות גבוהה?

שהרי לא נרצה לשנות את האילוצים שלנו.

נטען כי אם נשנה מעט את אי-השוויונים של האילוצים על ידי Scaling של המקדמים, נוכל לקבל את אותו הפתרון בהרבה פחות איטרציות. ניקח לדוגמה את בעיית Klee-Minty שראינו מקודם ונפעיל על האילוצים והמטרה שלה את ה Scaling באופן הבא :

$$u_i = 2^{d-i}x_i$$

והבעיה ״החדשה״ תיראה כך :

המטרה :

$$maximize \sum_{j=1}^{d} 2^{d-1}x_j$$

האילוצים :

$$u_1 \le 5$$

$$4u_1 + 2u_2 \le 25$$

$$8u_1 + 8u_2 + 4u_3 \le 125$$

$$\dots$$

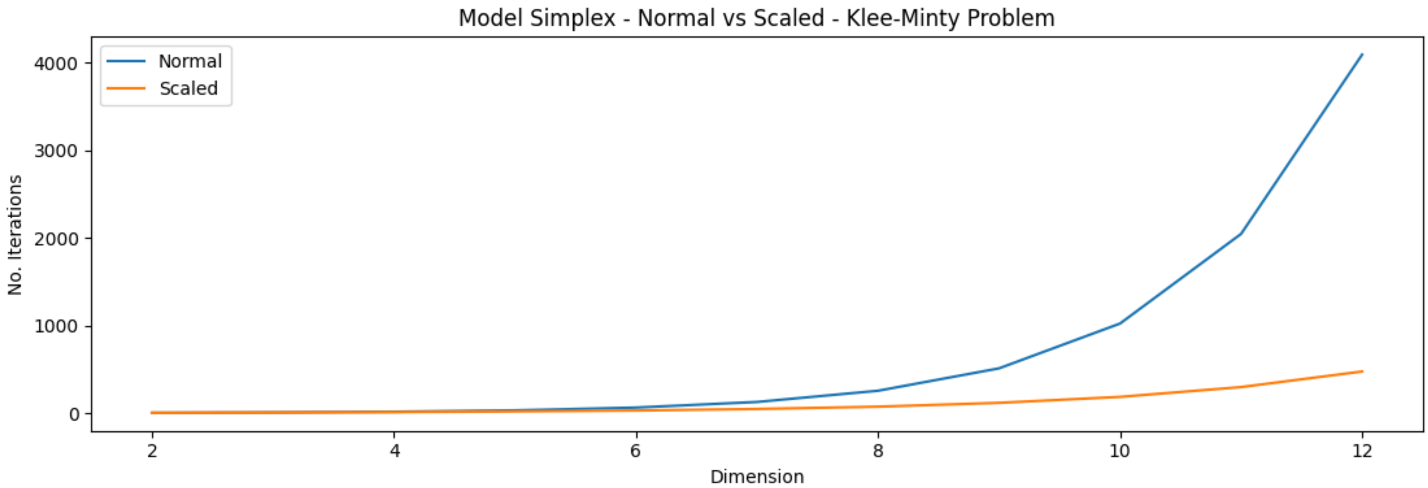
$$\sum_{j=1}^{d-1} 2^d u_j + 2^{d-1}u_d \ge 5^d$$

הערה – ניתן להשתמש בקבוע אחר במקום 2 ברדוקציה.

בכך אנחנו מבצעים רדוקציה לבעיה אחרת ופותרים אותה בזמן משמעותי יותר קטן. לאחר מכן ניתן להמיר חזרה את הפתרון האופטימלי ממשתני u למשתני x, שזה כמובן הכפלה של הפלט בקבועים.

להלן תוצאות ריצת התוכנית על הבעיה ״החדשה״:

- עבור מימד **12** קיבלנו **475** איטרציות במקום 4095.
- עבור מימד **10** קיבלנו **185** איטרציות במקום 1023.
- וריצת בונוס של מימד **21** קיבלנו **35441** איטרציות, במקום $2^{21} - 1 = 2097151$.

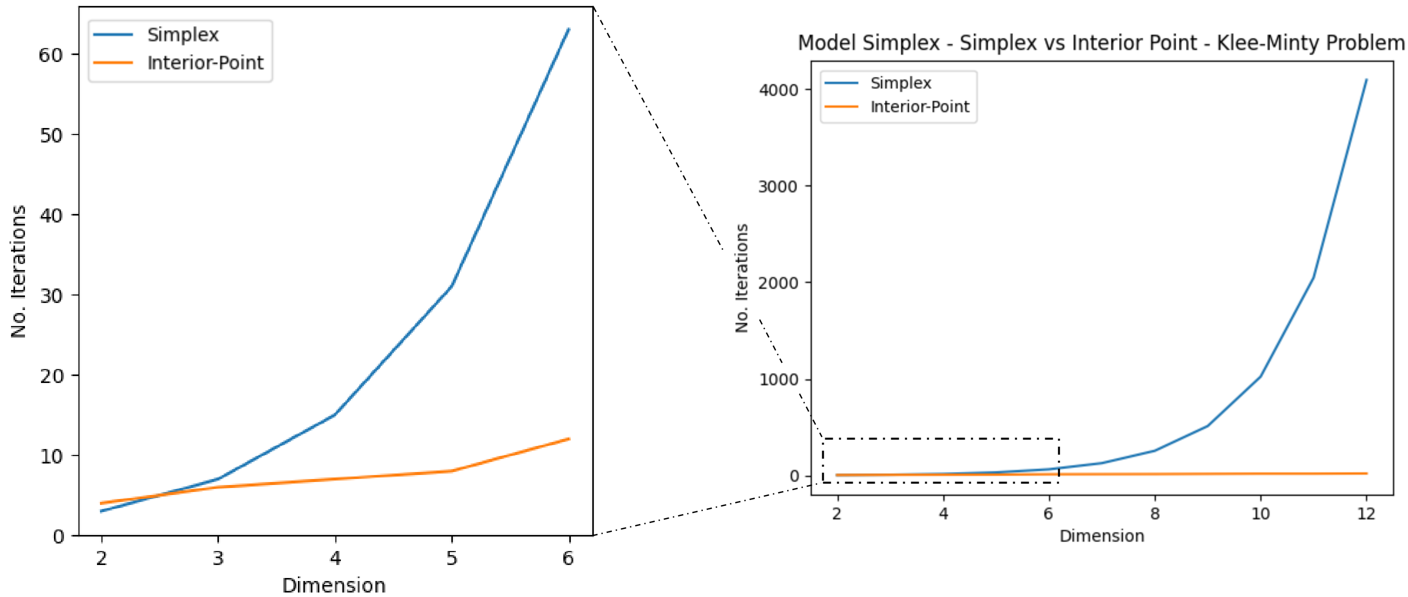


בדיקת זמן הריצה של Interior Point

בפרויקט זה לא נממש את שיטת האליפסות בשלמותה אלא נדגים את ריצת האלגוריתם Interior-Point שהוא בעצם השלב השני במציאת הפתרון האופטימלי.

נשתמש באותה ספריה שבה נעזרנו עבור ה-Simplex, ונריץ את אותן הבעיות שהרצנו מקודם:

נריץ את בעיית Klee-Minty במימד **12** ונקבל כי נמצא פתרון אופטימלי לאחר **14** איטרציות. נציין שמדובר באותו הפתרון ש- Simplex מצא ב-4095 (הפתרון זהה בערכו בשיעור גבוה, כלומר סטיה של כ- 10^{-7}).



שיטה נוספת להפחתת סיבוכיות זמן הריצה (לא צלחה בבדיקה שלנו)

במהלך החקר שלנו בנושא, נחשפנו במספר מאמרים אל דרך נוספת להפחתת כמות האיטרציות. בה מוסיפים רעש מזערי למקדמים באי-שוויוניות האילוצים. בעקבות טענות אלו, ניגשנו בעצמנו לשיטת הייעול זו וניסינו להשיג תוצאות המוכיחות את הטענה.

לקחנו לדוגמה את בעיית Klee-Minty שראינו מקודם והוספנו לאילוצים שלה מעט רעש.

התוצאות לא היו מאירות פנים. ככל שהקטנו את סדר הגודל של הרעש קיבלנו סטיה קטנה יותר בערך הפתרון האופטימלי אך כמות האיטרציות לא פחתה. כמו כן, ניסינו לשחק עם סדר הגודל של הרעש וככל שהגדלנו אותו כך התרחקנו מהפתרון לו אנו מצפים, אך כמות האיטרציות ירדה. תוצאות בדיקה זו לא עזרו לנו ולא ענו על הציפיות, שהרי אנחנו מעוניינים לשמור נאמנות לבעיה המקורית שלנו.

תהינו למה זה קורה. אנו מצליחים להבין את ההיגיון מאחורי רעיון הוספת הרעש, לנסות להפוך את הבעיה המורכבת של Klee-Minty ל״יותר אקראית״. ההשערה שלנו לתוצאות לא מספקות היא שאולי לא הבנו איך להפעיל את הרעש בצורה נכונה.

<u>חלק ג' – סיכום והצגת תובנות</u>

על מנת לבצע את ההשוואה הסופית בין שתי השיטות, נציין נקודות חשובות עבור כל אחת:

שיטת Simplex

- בהסתברות גבוהה מאוד, נקבל פתרון אופטימלי לאחר כמות איטרציות פולינומית לגודל הבעיה.
- דיוק גבוהה יותר ויציבות מבחינה מספרית.
- ניתן לבצע אנלוגיה לבעיה גאומטרית ידועה של מעבר על קודקודי פאון. יכול לתרום להבנת האלגוריתם.
- קלה יותר למימוש (לדעתנו).

שיטת האליפסה + Interior Point

- שיטה זו מוכחת כי רצה תמיד בסיבוכיות זמן פולינומית לקלט.
- דיוק מופחת יותר, ב-Interior Point אנחנו מתקדמים לפתרון האופטימלי על ידי תזוזה בכיוון **השואף** לאופטימליות.
- מחולקת לשני חלקים, ניתן לקבל תשובה של פתרון אפשרי לבעיה (או שלא קיים פתרון) וגם למצוא בעזרתו פתרון אופטימלי.

מתי עדיפה שיטה אחת על פני האחרת?

נרצה להבין למה שנבחר ב- Simplex עבור פתירת בעיה מהחיים האמיתיים או שמא נבחר בשיטת האליפסה.

<u>אין לכך תשובה חד-משמעית</u>, הדבר תלוי בבעיה שמוצגת בפנינו והעדיפות לכל שיטה יכולה להשתנות.

שיטת ה- Simplex עדיפה עבור בעיות קטנות, ואף אם קטנות מאוד – ניתן אפילו לפתור אותן באופן ידני על ידי מעבר על האלגוריתם ושימוש בטבלת Tableau ידנית. לעומת זאת, שיטת האליפסה יחד עם Interior Point תהיה עדיפה לבעיות גדולות יותר כיוון שהאלגברה הליניארית הנדרשת בכל איטרציה פשוטה יותר משל Simplex.

שיטת האליפסה היוותה פריצת דרך בנושא התכנות הליניארי, הראתה שניתן לפתור בעיות ולהשיג פתרון אופטימלי בסיבוכיות זמן פולינומית, לכן יש לה חשיבות רבה מבחינה תיאורטית.

שתי השיטות הן מאוד פרקטיות, אפילו Simplex עם הסיבוכיות המורכבת שלה באופן תיאורטי.

כדאי לבחון כל מקרה לסוגו.