Laboration 2: Den geometriska simplex-metoden

Niclas Börlin

26 september 2000

1 Inledning

Målet med denna laboration är att ni ska få implementera simplex-algoritmen i Matlab och se att det ibland räcker med en enkel algoritm för att lösa många svåra problem.

2 Gemensam uppgift

Implementera simplex-algoritmen nedan i Matlab för lösning av problemet

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b. \end{aligned}$$

Funktionen ska ta följande inparametrar: matrisen A, vektorn b, gradientvektorn c som beskriver problemet, ett starthörn x_0, en tolerans `eps`, och maximalt antal iterationer `nMax`. Toleransen behövs eftersom det p g a avrundningsfel, etc., inte går att utföra jämförelsen `A*x==b` för likhet, utan man måste i stället skriva `(A*x-b)<eps`. På samma sätt måste `A*p<0` skrivas om till `A*p<-eps`. Varvtalsbegränsningen `nMax` behövs för att hindra att funktionen går in i en oändlig loop.

Funktionens utparametrar skall minst vara lösningen x, antal iterationer n och en vektor λ med Lagrangemultiplikatorer i lösningspunkten. För att underlätta felsökning rekommenderas att ni "spårar" $x^{[n]}$ och $\lambda^{[n]}$ och returnerar två matriser `xx` och `ll` med $x^{[n]}$ resp $\lambda^{[n]}$ lagrade kolumnvis.

Funktionen ska kontrollera att given startpunkt är ett tillåtet hörn.

Testa algoritmen på några problem, t.ex. A Diet Problem i kapitel 3.4. För att ta fram ett tillåtet starthörn, plocka ut n st rader ur A och b och lös för x. Kontrollera om punkten är tillåten. Annars välj ut n andra rader och upprepa.

Eller så kan ni använda den utdelade funktionen `allcorners`, som ger er alla hörn till ett problem $Ax \geq b$.

Algoritm för Simplex-metoden

Antag att problemet saknar degenererade hörn och låt $x^{[n]}$ beteckna den aktuella punkten efter n iterationer. Givet ett hörn $x^{[n]}$ blir då algoritmen för simplex-metoden:

1. Bestäm en tillåten nedförsriktning p i $x^{[n]}$:
   (a) Identifiera den aktiva mängden $A_{\mathcal{A}}$ i $x^{[n]}$.
   (b) Lös $A_{\mathcal{A}}^T \lambda = c$.
   (c) Välj i sådan att $\lambda_i < 0$. Om alla $\lambda_i \geq 0$, avsluta.
   (d) Bestäm sökriktningen p som lösningen till $A_{\mathcal{A}} p = e_i$.
2. Bestäm steglängden $\alpha = \min_j \sigma_j$, där
$$\sigma_j = \begin{cases} \frac{-r_j(x^{[n]})}{a_j^T p} & \text{om } j \in \mathcal{D} \\ +\infty & \text{annars} \end{cases}.$$
3. $x^{[n+1]} = x^{[n]} + \alpha p$.

3 Gruppuppgifter

Varje grupp ska förutom den gemensamma uppgiften genomföra en egen uppgift. Max två grupper får lösa samma gruppuppgift.

3.1 Ta fram ett starthörn

Givet ett problem $Ax \geq b$ och en tillåten punkt x, ta fram ett tillåtet starthörn enligt algoritmen i Gill, Murray, Wright, kapitel 7.5.5 (utdelas).

3.2 Jämför olika Lagrange-strategier

Implementera två olika strategier för val av i i steg 1c i algoritmen. Lämpliga strategier är t.ex. välj den första negativa Lagrangemultiplikatorn resp. välj den mest negativa Lagrangemultiplikatorn (en "girig" strategi). Undersök på lämpligt problem om det blir någon skillnad i antalet iterationer som krävs.

3.3 Hantera degenererade hörn, version I

Modifiera er simplex-algoritm så att den klarar av degenererade hörn. Om ni kommer fram till ett degenererat hörn med $m \geq n$ aktiva bivillkor, så plocka ut n bivillkor slumpmässigt. Fel delmängd kommer att ge $\alpha = 0$, men algoritmen fungerar fortfarande. Testkör med några degenererade problem, t.ex. med likhetsbivillkor.

3.4 Hantera degenererade hörn, version II

Modifiera er simplex-algoritm så att den klarar av degenererade hörn. Om ni kommer fram till ett degenererat hörn med $m \geq n$ aktiva bivillkor, så testa alla $\begin{pmatrix} m \\ n \end{pmatrix}$ kombinationer av bivillkor tills ni hittar ett med enbart icke-negativa Lagrangemultiplikatorer (vi är vid optimum) eller ett med negativ Lagrangemultiplikator med tillåten steglängd $\alpha > 0$.

Ledning: Använd funktionen `nchoosek(1:m,n)` för att ta ut alla $\begin{pmatrix} m \\ n \end{pmatrix}$ alternativen.

4 Redovisning

Redovisningen skall vara dels skriftlig, dels muntlig. Beskriv det problem ni skulle lösa, hur ni löst det (inkl. algoritmer), ev. problem ni stött på. Redovisa också i lämpliga fall resultat av testkörningar på något exempelproblem plus ev. statistiska resultat (hur många iterationer tog det, hur lång exekveringstid, etc.).