LP. Leksjon 4

Kapittel 4: effektivitet av simpleksmetoden

- hvordan måle effektivitet?
- verste tilfelle analyse, Klee-Minty kuben
- gjennomsnittsanalyse og i praksis

Status

Hvor langt er vi kommet i vår LP-vandring?

- simpleksmetoden (Fase I og II)
- problemer: degenerasjon og sirkling
- løsning: antisirklingsregler, evt. perturbasjoner
- fundamentalteoremet for LP

Neste spørsmål: hvor god er simpleksmetoden ?

Men: først litt om ekvivalente optimeringsproblemer!

Ekvivalente optimeringsproblemer

Ofte er det nyttig å skrive om optimeringsproblemer til en mer hensiktsmessig form. Da er det viktig at man ender opp med et "ekvivalent problem". La oss presisere hva dette betyr.

La P og Q betegne to optimeringsproblemer, med variabelvektor $x \in \mathbb{R}^n$ i P og $y \in \mathbb{R}^k$ i Q. Her kan n og k være forskjellige. Vi sier at P og Q er ekvivalente dersom

1. P er tillatt hvis og bare hvis Q er tillatt.
2. P er ubegrenset hvis og bare hvis Q er ubegrenset.
3. P har optimal løsning hvis og bare hvis Q har optimal løsning. Videre finnes det da en funksjon $f : \mathbb{R}^n \to \mathbb{R}^k$ slik at x er optimal i P hvis og bare hvis $f(x)$ er optimal i Q.

Merk at man kan vise symmetri her: rekkefølgen på P og Q i definisjonen spiller ingen rolle.

Som et eksempel, og en oppgave, betrakte problemene

$$\begin{array}{ll} \mathrm{P:} & \max\{c^T x : Ax \leq b\} \\ \mathrm{Q:} & \max\{c^T(x' - x'') : A(x' - x'') \leq b,\ x', x'' \geq O\} \end{array}$$

Vis at P og Q er ekvivalente!

LP problemer kan f.eks. ha noen variable som er ikkenegative (evt. ikkepositive), noen som er frie, og det kan være uliheter ($\leq$ og $\geq$), og likninger. Lær deg hvordan man kan skrive om slike problemer til ekvivalente LP problemer!

Effektivitet

To typer mål på effektivitet av en algoritme:

- **verste tilfelle:** maksimalt antall regneoperasjoner for problemer av en gitt størrelse
- **gjennomsnitt:** gjennomsnitlig eller forventet antall regneoperasjoner for problemer av en gitt størrelse

LP: regnetid øker når antall variable eller begrensninger øker. Så regnetid blir en funksjon av "problemstørrelse".

Hvordan måle problemstørrelse?

- *enkelt:* antall tall, dvs $m \times (n+1)$. Svakhet: i "virkelige problemer" (der m og n er store) er mange av tallene null. Dette kan utnyttes i f.eks. simpleksalgoritmen til å "speede opp".
- *mer nøyaktig:* antall ikkenuller. Svakhet: regneoperasjoner med store tall tar lengre tid enn for små (heller $14 \cdot 7$ enn $832573928 \cdot 3722984$)

- *enda mer nøyaktig:* antall bits som kreves for å lagre alle dataene på en datamaskin

Hvordan måle totalarbeid (kompleksitet) for en algoritme?

- regnetid (CPU)
- antall iterasjoner
- antall elementære regneoperasjoner

Vårt valg: bruker m og n som mål på problemstørrelse og antall pivoteringer som kompleksitetsmål.

Skal se på verste tilfelle analyse av simpleksmetoden. Svaret blir dessverre at metoden, teoretisk sett, ikke er god. I praksis er situasjonen motsatt: simpleksalgoritme fungerer utmerket. "Forklaring": de "vanskelige LP eksemplene" dukker ikke opp i praktiske problemer.

I 1972 fant V.Klee og G.J.Minty en klasse av LP problemer som "tar knekken" på simpleksmetoden med den "vanlige" pivoteringsregelen (største koeffisient regel). LP problemet har n variable og det viser seg at antall pivoteringer blir $2^n - 1$. Så antall pivoteringer vokser eksponentielt i antall variable! Dette er håpløst mange iterasjoner, se f.eks. (ang. regnetid, anta en million pivoteringer pr. sekund): :

n :	10	20	50	100
n^2 :	100	400	2500	10000
2^n :	1024	1048576	1.125899e+15	1.267650e+30
tid	0.001 sek	1.0 sek	35 år	4.0e+16 år

Skal se på dette LP problemet. Idéen er å deformere kubel $[0, 1]^n$ i $\mathbb{R}^n$ på en slik måte at simpleksalgoritmen går gjennom alle de 2^n hjørnene! Dette gir $2^n - 1$ pivoteringer.

Her er Klee-Minty LP problemet:

$$\begin{array}{llrcl}
\max & \sum_{j=1}^{n} 10^{n-j} x_j & & & \\
\text{f.a.} & & & & \\
& 2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i & \leq & 100^{i-1} & \text{for } i = 1, \ldots, n \\
& x_j & \geq & 0 & \text{for } j = 1, \ldots, n.
\end{array}$$

Tolkning av begrensningene:

- $i = 1$: $x_1 \leq 1$
- $i = 2$: $20x_1 + x_2 \leq 100$, så $x_2 \leq 100 - 20x_1 \approx 100$ idet $0 \leq x_1 \leq 1$.
- $i = 3$: $200x_1 + 20x_2 + x_3 \leq 10000$, så $x_3 \leq 10000 - 200x_1 - 20x_2 \approx 10000$ idet idet $0 \leq x_1 \leq 1$ og $0 \leq x_2 \leq 100$.
- ...

Så tilnærmet har vi begrensningene

$$\begin{array}{ccccc} 0 & \leq & x_1 & \leq & 1 \\ 0 & \leq & x_2 & \leq & 100 \\ & & \vdots & & \\ 0 & \leq & x_n & \leq & 100^{n-1} \end{array}$$

slik at mengden P av tillatte løsninger er tilnærmet lik et n-dimensjonalt rektangel (kube). P kalles derfor *Klee-Minty kuben*.

For å lette analysen vil vi forandre litt på problemet.

- velg tall b_i slik at $1 = b_1 \ll b_2 \ll \ldots \ll b_n$. F. eks. velges b_1 så mye mindre enn b_2 at selv om vi multipliserer med visse tall i simpleksalgoritmen vil den nye verdien på b_1 stadig være langt mindre enn den nye verdien på b_2. Tenk på b_i'ene som uavhengige variable og at man kan finne passende verdier på dem senere.

- i Klee-Minty problemet erstatter vi først høyresiden 100^{i-1} med b_i. Merk at de gamle høyresidene økte med en faktor 100 for hver ny rad, og dette har vi tatt vare på ved valget av b_i-ene.
- erstatter så høyresiden b_i med

$$\sum_{j=1}^{i-1} 10^{i-j} b_j + b_i.$$

  Dette er en "mindre endring" idet det første leddet en mye mindre enn b_i (fordi $b_1, \ldots, b_{i-1}$ er tilstrekkelige små i forhold til b_i.)
- til slutt endrer vi objektivfunksjonen i LP problemet ved å trekke fra

$$(1/2) \sum_{j=1}^{i-1} 10^{n-j} b_j.$$

Resultatet er et modifisert Klee-Minty problem:

$$
\begin{array}{ll}
\max & \sum_{j=1}^{n} 10^{n-j} x_j - (1/2) \sum_{j=1}^{i-1} 10^{n-j} b_j \\
\text{f.a.} & \\
& \begin{array}{rcccll}
2\sum_{j=1}^{i-1} 10^{i-j} x_j & + & x_i & \leq & \sum_{j=1}^{i-1} 10^{i-j} b_j + b_i & \text{for } i \leq n \\
& & x_j & \geq & 0 & \text{for } j \leq n.
\end{array}
\end{array}
$$

Legg merke til at høyresidene er positive så vi trenger ikke Fase I av simpleksmetoden.

Resultat: simpleksalgoritmen med "største koeffisient pivoteringsregelen" bruker $2^n - 1$ pivoteringer for å løse det modifisert Klee-Minty problemet.

Vi lar beviset for dette være øvingsoppgave (Ex. 4.5 og 4.6 i Vanderbei).
Vi løser nå det modifiserte Klee-Minty problemet for $n = 3$.
Forhåpentligvis aner vi da et mønster som holder for alle n.

Basisliste 0:

$$
\begin{array}{rcrcrcrcrcrcr}
\eta & = & -\frac{100}{2}b_1 & - & \frac{10}{2}b_2 & - & \frac{1}{2}b_3 & + & 100x_1 & + & 10x_2 & + & x_3 \\
\hline
w_1 & = & b_1 & & & & & - & x_1 & & & & \\
w_2 & = & 10b_1 & + & b_2 & & & - & 20x_1 & - & x_2 & & \\
w_3 & = & 100b_1 & + & 10b_2 & + & b_3 & - & 200x_1 & - & 20x_2 & - & x_3
\end{array}
$$

Her skal x_1 inn i basis ved største koeffisient regelen. Siden b_1 er mye *mindre* enn b_2 og b_3 vil w_1 forlate basis. Gjennomfører pivoteringen (sjekk regningen ut fra $x_1 = b_1 - w_1$).

Basisliste 1:

$$
\begin{array}{rcrcrcrcrcrcr}
\eta & = & \frac{100}{2}b_1 & - & \frac{10}{2}b_2 & - & \frac{1}{2}b_3 & - & 100w_1 & + & 10x_2 & + & x_3 \\
\hline
x_1 & = & b_1 & & & & & - & w_1 & & & & \\
w_2 & = & -10b_1 & + & b_2 & & & + & 20w_1 & - & x_2 & & \\
w_3 & = & -100b_1 & + & 10b_2 & + & b_3 & + & 200w_1 & - & 20x_2 & - & x_3
\end{array}
$$

Fantastisk! Den nye basislisten inneholder de samme tallene bortsett fra noen fortegnsendringer! Endringene er:

- x_1 og w_1 byttet rolle
- bare endringer i de to kolonnene for x_1 og b_1
- og disse to kolonnene er multiplisert med -1, bortsett fra i pivoteringslikningen; der er det ingen fortegnsendring

Nå går x_2 inn i basis og w_2 ut.

Basisliste 2:

$$
\begin{array}{rcrcrcrcrcrcr}
\eta & = & -\frac{100}{2}b_1 & + & \frac{10}{2}b_2 & - & \frac{1}{2}b_3 & + & 100w_1 & - & 10w_2 & + & x_3 \\
\hline
x_1 & = & b_1 & & & & & - & w_1 & & & & \\
x_2 & = & -10b_1 & + & b_2 & & & + & 20w_1 & - & w_2 & & \\
w_3 & = & 100b_1 & - & 10b_2 & + & b_3 & - & 200w_1 & + & 20w_2 & - & x_3
\end{array}
$$

Basisliste 3:

$$
\begin{array}{rcrcrcrcrcrcr}
\eta & = & \frac{100}{2}b_1 & + & \frac{10}{2}b_2 & - & \frac{1}{2}b_3 & - & 100x_1 & - & 10w_2 & + & x_3 \\
\hline
w_1 & = & b_1 & & & & & - & x_1 & & & & \\
x_2 & = & 10b_1 & + & b_2 & & & - & 20x_1 & - & w_2 & & \\
w_3 & = & -100b_1 & - & 10b_2 & + & b_3 & + & 200x_1 & + & 20w_2 & - & x_3
\end{array}
$$

Basisliste 4:

$$\begin{array}{rcrcrcrcrcrcr}
\eta & = & -\frac{100}{2}b_1 & - & \frac{10}{2}b_2 & + & \frac{1}{2}b_3 & + & 100x_1 & + & 10w_2 & - & w_3 \\
\hline
w_1 & = & b_1 & & & & & - & x_1 & & & & \\
x_2 & = & 10b_1 & + & b_2 & & & - & 20x_1 & - & w_2 & & \\
x_3 & = & -100b_1 & - & 10b_2 & + & b_3 & + & 200x_1 & + & 20w_2 & - & w_3
\end{array}$$

Basisliste 5:

$$\begin{array}{rcrcrcrcrcrcr}
\eta & = & \frac{100}{2}b_1 & - & \frac{10}{2}b_2 & + & \frac{1}{2}b_3 & - & 100w_1 & + & 10w_2 & - & w_3 \\
\hline
x_1 & = & b_1 & & & & & - & w_1 & & & & \\
x_2 & = & -10b_1 & + & b_2 & & & + & 20w_1 & - & w_2 & & \\
x_3 & = & 100b_1 & - & 10b_2 & + & b_3 & - & 200w_1 & + & 20w_2 & - & w_3
\end{array}$$

Basisliste 6:

$$\begin{array}{rcrcrcrcrcrcr}
\eta & = & -\frac{100}{2}b_1 & + & \frac{10}{2}b_2 & + & \frac{1}{2}b_3 & + & 100w_1 & - & 10x_2 & - & w_3 \\
\hline
x_1 & = & b_1 & & & & & - & w_1 & & & & \\
w_2 & = & -10b_1 & + & b_2 & & & + & 20w_1 & - & x_2 & & \\
x_3 & = & -100b_1 & + & 10b_2 & + & b_3 & + & 200w_1 & - & 20x_2 & - & w_3
\end{array}$$

Basisliste 7:

$$
\begin{array}{rcrcrcrcrcrcr}
\eta & = & \frac{100}{2} b_1 & + & \frac{10}{2} b_2 & + & \frac{1}{2} b_3 & - & 100x_1 & - & 10x_2 & - & w_3 \\
\hline
w_1 & = & b_1 & & & & & - & x_1 & & & & \\
w_2 & = & 10b_1 & + & b_2 & & & - & 20x_1 & - & x_2 & & \\
x_3 & = & 100b_1 & + & 10b_2 & + & b_3 & - & 200x_1 & - & 20x_2 & - & w_3
\end{array}
$$

Optimal! Altså: $7 = 2^3 - 1$ pivoteringer.

Observasjoner:

- i hver pivotering byttes x_j og w_j for en viss j
- bortsett fra fortegnsendringer bevares alle tallene under pivoteringene
- men hvordan endres fortegnene? Oppgave 4.5 og 4.6!
- problemet kunne blitt løst med bare én pivotering, hvis vi istedet for x_1 hadde tatt x_3 inn i basis! Men valget ble jo styrt av pivoteringsregelen.

Kommentarer om effektivitet og LP

- finnes det andre pivoteringsregler for simpleksalgoritmen som ikke kan få 2^n pivoteringer ??? Helst ville vi hatt at antall pivoteringer ikke vokser fortere enn f.eks. n^2 eller n^3 (eller et polynom i antall variable $n =$)!
- svaret er ukjent! Men for alle forslåtte pivoteringsregler har noen funnet 2^n "moteksempler".
- K.-H. Borgwardt har gitt en statistisk analyse som viser at *forventet* antall pivoteringer (når LP problemene "trekkes tilfeldig") vokser som $n^3 m^{1/(n-1)}$
- i *praksis* regner man at antall pivoteringer typisk ligger mellom m og $2m$. Dette er meget bra! Og merk at antall variable n spiller mindre rolle for antall pivoteringer (selv om regnetiden øker med n).

Kommentarer om effektivitet og LP

- i 1979 kom ellipsoidemetoden utviklet av L.Khachian: første polynomtid algoritme for LP. Dette var et teoretisk gjennombrudd. Denne algoritmen er teoretisk "god", men håpløs i praksis!
- N. Karamarkar publiserte i 1984 en ny LP algoritme basert på helt andre idéer: i motsetning til simpleksalgoritmen lages en følge av punkter som ikke er basisløsninger; de ligger i det indre av mengden av tillatte løsninger.
- i de senere år har et svært aktivt forskningsfelt vært indrepunktsmetoder for LP. Del 3 av Vanderbei's bok omhandler slike metoder.