Fast Solver of Closely Related Quadratic Programming Problems

Andreas Halle

June 10, 2013

PROMAPS

- Utviklet av Goodtech og MathConsult.
- Kalkurerer leveransepåliteligheten i et nettverk
- Dette er formulert som en rekke veldig like QP-problemer (Quadratic Programming).
- QP-løseren en flaskehals. Skjermbildet (PROMAPS) oppdaterer seg hvert femte minutt.
- `http://www.tu.no/energi/2011/10/07/her-beregnes-risikoen-for-svikt-i-kraftnettet`

Objektfunksjonen

$$f(x) = x^T \Phi D x + (g - c)^T x$$

- $f(x)$ representerer leveransekostnader (E/s)
- x representerer strømmen over grenene (W)
- Φ representerer strømtap $(1/W)$
- D representerer overføringskostnader (E/J)
- g representerer kostnader for å generere strøm (E/J)
- c representerer leveransepris (E/J)

Objektfunksjonen

$$f(x) = x^T H x + b^T x$$

- $H = \Phi D$ representerer kostnader $(E/(W^2 s))$
- $b = g - c$ representerer kostnader (E/J)
- H er en matrise på størrelsen $n \times n$
- b og x er vektorer i $\mathbb{R}^n$

Optimeringsproblemet

Vi definerer et konveks QP-problem

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to } Ax = 0,\ l \leq x \leq u$$

- A er en $m \times n$ insidensmatrise for et strømnettverk
- m noder
- n grener
- l og u er nedre og øvre grenkapasitet (W)

Utfall

Vi ønsker å modellere utfall.

- Setter $l_i = u_i = 0$
- Goodtech ønsker å løse QP-problemet for ulike utfall
- Definerer subinstanser for kombinasjoner av grener som faller ut
- Instans er et QP-problem uten utfall
- Subinstans er en instans med utfall

Utfall

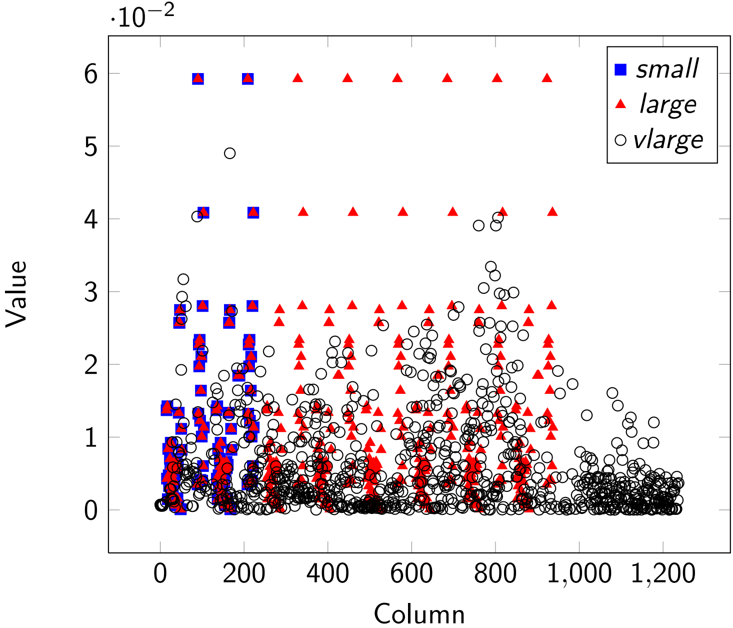
- Vil løse så mange subinstanser som mulig.
- Usannsynlig at det er mange utfall.
- Vi prøver å løse alle subinstanser som har mindre eller lik β utfall.

$$\sigma(\beta, n) = \sum_{j=0}^{\beta} \binom{n}{j}$$

Subinstanser

- En mengde av variabler som representerer utfall $\mathcal{M}_k$
- $k = \sum_{j \in \mathcal{M}_k} 2^{j-1}$
- Eks. $\mathcal{M}_k = \{1, 3, 5\}$. $k = 21$
- En subinstans $\mathcal{Q}_k$ defineres av $\mathcal{M}_k$
- Optimal løsning til subinstans $\mathcal{Q}_k$ noterer vi som x_k^*

Instanser fra Goodtech



Instanser fra Goodtech

Table : Størrelse for hver instans

Problemstørrelse	*small*	*large*	*vlarge*
Rader	82	328	1127
Kolonne	238	952	3437
Ikke-nuller A	348	1392	4840
Ikke-nuller H	108	432	894

Table : Verdier i objektfunksjonen

	small og *large*	*vlarge*
$\max(h_{ii})$	2.9614×10^{-2}	4.9011×10^{-2}
$\min(h_{ii})$	4.9290×10^{-5}	1.1026×10^{-5}
$\mathrm{mean}(h_{ii})$	5.2864×10^{-3}	5.8984×10^{-3}
$\max(b_i)$	20	20
$\min(b_i)$	-70	-50

Instanser fra Goodtech

- Vi ser at det lineære leddet har mye høyere verdier enn det kvadratiske
- Prøver å lineærisere objektfunksjonen
- Lineær Taylor-utvikling av objektfunksjonen i punkt a:
  $T_a(x) = -a^T Ha + 2a^T Hx + b^T x$
  $T_0(x) = b^T x$

Definerer et LP $\mathcal{L}$ for hvert QP $\mathcal{Q}$

$$\min_{x \in \mathbb{R}^n} g(x) \quad \text{subject to } Ax = 0, \; l \le x \le u$$

- $g(x) = T_0(x) = b^T x$

Hvor like er $\mathcal{L}$ og $\mathcal{Q}$?

- Optimal løsning til $\mathcal{Q}$ noteres x^*
- Optimal løsning til $\mathcal{L}$ noteres $\hat{x}$
- Vi noterer avvik mellom $f(\hat{x})$ og $f(x^*)$ for $\Delta = \left|\frac{f(\hat{x}) - f(x^*)}{f(x^*)}\right|$
- Vi genererer tilfeldige instanser og sjekker Δ

100Δ

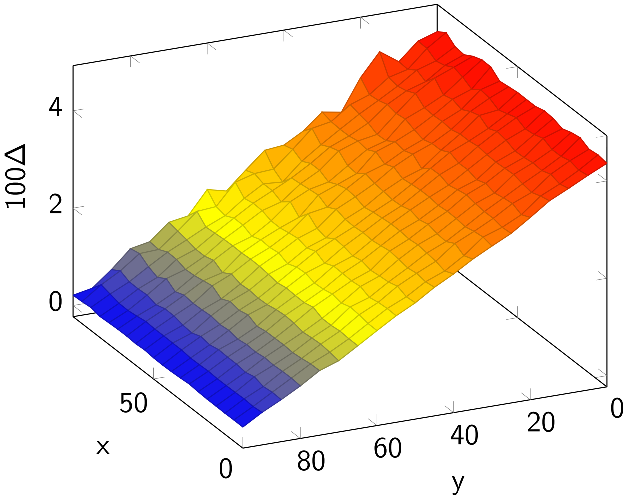


Figure : Avvik som en funksjon av tettheten i objektfunksjonen. x er prosent nuller på diagonalen til H. y er prosent nuller i b

Hvor like er $\mathcal{L}$ og $\mathcal{Q}$?

Oppnår 95% optimal verdi etter løst $\mathcal{L}$.

- Metode basert på successive linear programming (SLP)
- Lar vi startverdien $x_0 = 0$, når vi rundt 95% av optimal målfunksjonsverdi etter første iterasjon
- Taylor-utvikling i x_k noterer vi som T_k
- Definerer $\mathcal{L}_k$ som LP-problemet å minimere T_k underlagt sidekravene til $\mathcal{Q}$.

Slp

Algorithm 1: `slp`(x_0, $\epsilon \geq 0$)

Set $k \leftarrow 0$

repeat

$T_k \leftarrow -x_k^T H x_k + 2x_k^T Hx + b^T x$

$\hat{x}_k \leftarrow \text{optimal solution of } \mathcal{L}_k$ (Solve)

$\alpha_k \in \arg\min_{\alpha \leq 1} f((1-\alpha)x_k + \alpha \hat{x}_k)$

$x_{k+1} \leftarrow (1-\alpha_k)x_k + \alpha_k \hat{x}_k$

$k \leftarrow k+1$

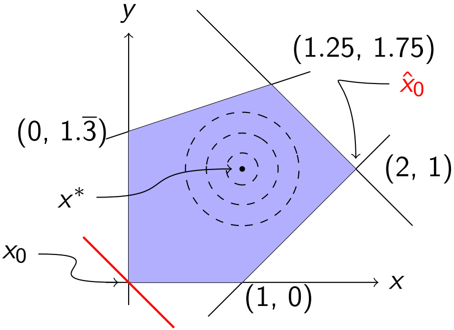
until $\dfrac{f(x_{k-1}) - f(x_k)}{|f(x_{k-1})|} \leq \epsilon$

Et eksempel

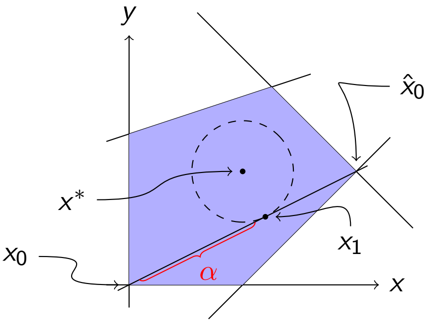
$$\begin{array}{lrcrcl}
\text{minimize} & (x-1)^2 & + & (y-1)^2 & - & 2 \\
\text{subject to} & x & + & y & \leq & 3 \\
 & x & - & y & \leq & 1 \\
 & x & + & 3y & \leq & 4 \\
 & & & x, y & \geq & 0
\end{array}$$

- $T_0 = -2x - 2y$ blir lineær objektfunksjon

$\mathcal{L}_0$

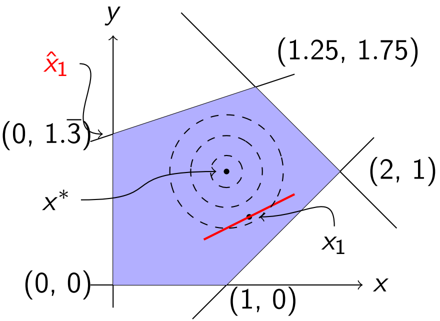


Linjesøk



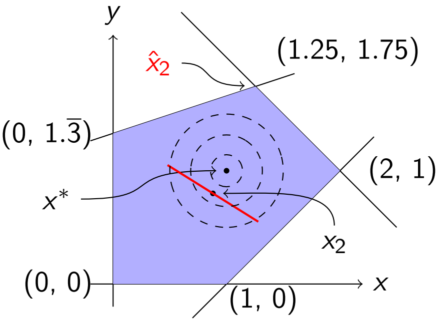
- $\alpha = 0.6$
- $x_1 = 0.4x_0 + 0.6\hat{x}_0 = (1.2, 0.6)$
- $T_1 = 0.4x - 0.8y - 1.8$

$\mathcal{L}_1$



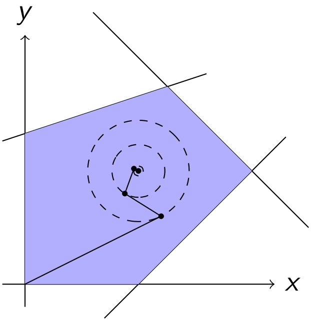
- $\alpha = 0.27$
- $x_2 = (0.88, 0.8)$
- $T_2 = -0.25x - 0.4y - 0.4$

$\mathcal{L}_2$

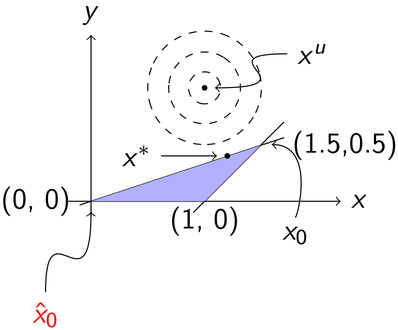


- $\alpha = 0.23$
- $x_3 = (0.96, 1.02)$

Sti



Endrer et sidekrav

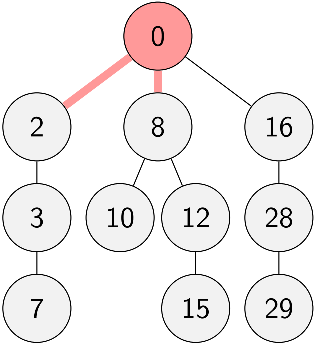


- $-x + 3y \leq 4$
- $-x + 3y \leq 0$
- $\alpha = 0.2$
- $x_1 = 0.8x_0 + 0.2\hat{x}_0 = (1.2, 0.4) = x^*$

Like optimale løsninger

- En mengde $\mathcal{M}_k$ med grener som faller ut.
- En subinstans $\mathcal{Q}_k$ definert av $\mathcal{Q}$ og $\mathcal{M}_k$
- Optimal løsning til $\mathcal{Q}_k$ noteres som x_k^*
- En mengde med variabler som er 0 i x_k^* noteres som $\mathcal{Z}_k$.
- $2^n - 1$ subinstanser. $|\mathcal{Z}_0| = 1749$ i *vlarge*.
- $2^{1749} \approx 3 \times 10^{526}$ subinstanser har løsning x_0^*

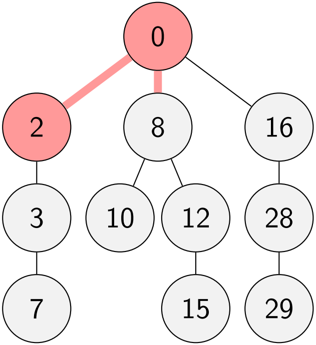
Søker etter $\mathcal{M}_{14} = \{2, 3, 4\}$



$$\mathcal{M}_0 = \{\} \qquad \mathcal{Z}_0 = \{1, 3\}$$
$$\mathcal{M}_2 = \{2\} \qquad \mathcal{Z}_2 = \{2, 3, 5\}$$
$$\mathcal{M}_3 = \{1, 2\} \qquad \mathcal{Z}_3 = \{1, 2, 4, 5\}$$
$$\mathcal{M}_7 = \{1, 2, 3\} \qquad \mathcal{Z}_7 = \{1, 2, 3, 5\}$$
$$\mathcal{M}_8 = \{4\} \qquad \mathcal{Z}_8 = \{1, 4, 5\}$$
$$\mathcal{M}_{10} = \{2, 4\} \qquad \mathcal{Z}_{10} = \{2, 3, 4, 5\}$$

$$\mathcal{M}_{12} = \{3, 4\} \qquad \mathcal{Z}_{12} = \{4, 3, 1\}$$
$$\mathcal{M}_{15} = \{1, 2, 3, 4\} \qquad \mathcal{Z}_{15} = \{1, 2, 3, 4\}$$
$$\mathcal{M}_{16} = \{5\} \qquad \mathcal{Z}_{16} = \{1, 3, 5\}$$
$$\mathcal{M}_{28} = \{3, 4, 5\} \qquad \mathcal{Z}_{28} = \{2, 3, 4, 5\}$$
$$\mathcal{M}_{29} = \{1, 3, 4, 5\} \qquad \mathcal{Z}_{29} = \{1, 2, 3, 4, 5\}$$

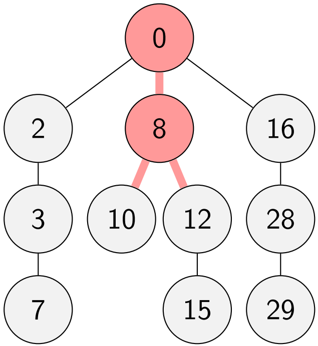
Søker etter $\mathcal{M}_{14} = \{2, 3, 4\}$



$$\mathcal{M}_0 = \{\} \qquad \mathcal{Z}_0 = \{1, 3\}$$
$$\mathcal{M}_2 = \{2\} \qquad \mathcal{Z}_2 = \{2, 3, 5\}$$
$$\mathcal{M}_3 = \{1, 2\} \qquad \mathcal{Z}_3 = \{1, 2, 4, 5\}$$
$$\mathcal{M}_7 = \{1, 2, 3\} \qquad \mathcal{Z}_7 = \{1, 2, 3, 5\}$$
$$\mathcal{M}_8 = \{4\} \qquad \mathcal{Z}_8 = \{1, 4, 5\}$$
$$\mathcal{M}_{10} = \{2, 4\} \qquad \mathcal{Z}_{10} = \{2, 3, 4, 5\}$$

$$\mathcal{M}_{12} = \{3, 4\} \qquad \mathcal{Z}_{12} = \{4, 3, 1\}$$
$$\mathcal{M}_{15} = \{1, 2, 3, 4\} \qquad \mathcal{Z}_{15} = \{1, 2, 3, 4\}$$
$$\mathcal{M}_{16} = \{5\} \qquad \mathcal{Z}_{16} = \{1, 3, 5\}$$
$$\mathcal{M}_{28} = \{3, 4, 5\} \qquad \mathcal{Z}_{28} = \{2, 3, 4, 5\}$$
$$\mathcal{M}_{29} = \{1, 3, 4, 5\} \qquad \mathcal{Z}_{29} = \{1, 2, 3, 4, 5\}$$

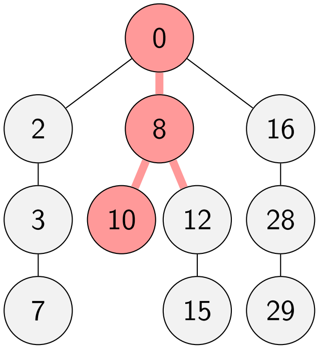
Søker etter $\mathcal{M}_{14} = \{2, 3, 4\}$



$$\begin{aligned}
\mathcal{M}_0 &= \{\} & \mathcal{Z}_0 &= \{1, 3\} \\
\mathcal{M}_2 &= \{2\} & \mathcal{Z}_2 &= \{2, 3, 5\} \\
\mathcal{M}_3 &= \{1, 2\} & \mathcal{Z}_3 &= \{1, 2, 4, 5\} \\
\mathcal{M}_7 &= \{1, 2, 3\} & \mathcal{Z}_7 &= \{1, 2, 3, 5\} \\
\mathcal{M}_8 &= \{4\} & \mathcal{Z}_8 &= \{1, 4, 5\} \\
\mathcal{M}_{10} &= \{2, 4\} & \mathcal{Z}_{10} &= \{2, 3, 4, 5\}
\end{aligned}$$

$$\begin{aligned}
\mathcal{M}_{12} &= \{3, 4\} & \mathcal{Z}_{12} &= \{4, 3, 1\} \\
\mathcal{M}_{15} &= \{1, 2, 3, 4\} & \mathcal{Z}_{15} &= \{1, 2, 3, 4\} \\
\mathcal{M}_{16} &= \{5\} & \mathcal{Z}_{16} &= \{1, 3, 5\} \\
\mathcal{M}_{28} &= \{3, 4, 5\} & \mathcal{Z}_{28} &= \{2, 3, 4, 5\} \\
\mathcal{M}_{29} &= \{1, 3, 4, 5\} & \mathcal{Z}_{29} &= \{1, 2, 3, 4, 5\}
\end{aligned}$$

Søker etter $\mathcal{M}_{14} = \{2, 3, 4\}$



$\mathcal{M}_0 = \{\}$
$\mathcal{M}_2 = \{2\}$
$\mathcal{M}_3 = \{1, 2\}$
$\mathcal{M}_7 = \{1, 2, 3\}$
$\mathcal{M}_8 = \{4\}$
$\mathcal{M}_{10} = \{2, 4\}$

$\mathcal{Z}_0 = \{1, 3\}$
$\mathcal{Z}_2 = \{2, 3, 5\}$
$\mathcal{Z}_3 = \{1, 2, 4, 5\}$
$\mathcal{Z}_7 = \{1, 2, 3, 5\}$
$\mathcal{Z}_8 = \{1, 4, 5\}$
$\mathcal{Z}_{10} = \{2, 3, 4, 5\}$

$\mathcal{M}_{12} = \{3, 4\}$
$\mathcal{M}_{15} = \{1, 2, 3, 4\}$
$\mathcal{M}_{16} = \{5\}$
$\mathcal{M}_{28} = \{3, 4, 5\}$
$\mathcal{M}_{29} = \{1, 3, 4, 5\}$

$\mathcal{Z}_{12} = \{4, 3, 1\}$
$\mathcal{Z}_{15} = \{1, 2, 3, 4\}$
$\mathcal{Z}_{16} = \{1, 3, 5\}$
$\mathcal{Z}_{28} = \{2, 3, 4, 5\}$
$\mathcal{Z}_{29} = \{1, 2, 3, 4, 5\}$

Algoritme: `find`

Algorithm 2: $\texttt{find}(\mathcal{M}_l,\ v_k)$

```
if 𝓜_l ⊆ 𝓩_k then
    return k
foreach child vertex v_i of v_k do
    if 𝓜_i ⊆ 𝓜_l then
        j ← find(𝓜_l, v_i)
        if j is not −1 then
            return j
return −1
```

if $\mathcal{M}_l \subseteq \mathcal{Z}_k$ **then**
  return k
foreach *child vertex* v_i *of* v_k **do**
  if $\mathcal{M}_i \subseteq \mathcal{M}_l$ **then**
    $j \leftarrow \texttt{find}(\mathcal{M}_l,\ v_i)$
    if j *is not* -1 **then**
      return j
return -1

Eksperiment 1

Vi løser *small*, med $\sigma(2, 238) = 28442$ subinstanser

Table : Resultater av de forskjellige implementasjonen med endrende toleranse.

ϵ	cClp	cSlp	nClp	nSlp
10^{-1}	45.51	55.61	72.32	85.51
10^{-2}	46.34	55.89	73.11	85.51
10^{-3}	51.12	59.04	75.60	85.28
10^{-4}	52.46	73.79	77.83	107.39
10^{-5}	54.48	232.53	81.16	355.47
10^{-6}	65.42	1363.46	93.29	2022.25
10^{-7}	70.78	6522.91	100.85	9395.92

Eksperiment 1

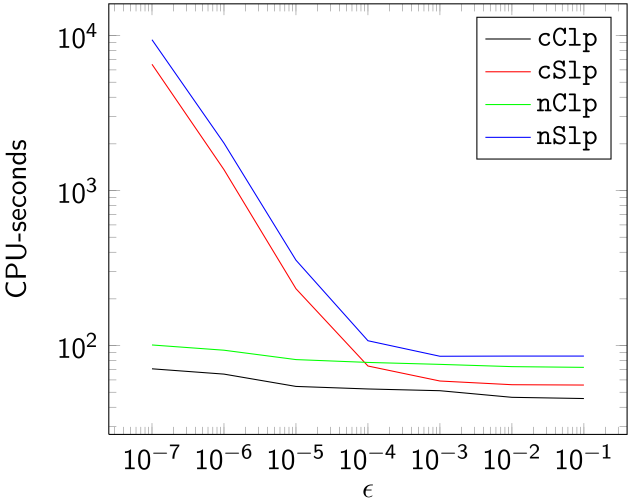


Figure : Kjøretid i CPU-sekunder for å løse *small* og dens subinstanser.

Eksperiment 2

Table : Kjøretid i CPU-sekunder for å løse de tre instansene.

Implementasjon	*small*	*large*	*vlarge*
`cClp`	0.52	9.55	76.18
`cSlp`	0.71	32.88	585.60
`nClp`	0.65	11.68	157.53
`nSlp`	0.89	39.87	1173.74

Tilfeldige instanser

- $m = \lfloor \frac{7}{20} n \rfloor$
- b_i har 50% sannsynlighet for å være null. Ellers $10 \leq |b_i| \leq 70$.
- h_{ii} har 50% sannsynlighet for å være null. Ellers $10^{-5} \leq h_{ii} \leq 10^{-1}$.

Eksperiment 3

Table : Kjøretid for å løse tilfeldige instanser med økende n. $\beta = 1$.

n	`cClp`	`nClp`	Relativ Speedup
500	4.9	5.9	16.9%
1000	42.1	53.0	20.6%
1500	181.5	234.5	22.6%
2000	547.1	710.2	23.0%

Eksperiment 4

Table : Kjøretid i CPU-sekunder for $n = 50$ og økende β.

β	`cClp`	`nClp`	Relativ Speedup	Distinkte løsninger
1	0.03	0.04	25.0%	37.4 (74.3%)
2	0.64	0.94	31.9%	744.3 (58.3%)
3	7.06	15.90	55.6%	9484.7 (45.4%)
4	77.59	188.83	58.9%	82262.5 (32.8%)
5	586.54	1758.23	66.6%	574685.0 (24.2%)

Eksperiment 4