Exercise 1, Process Control, advanced course

Henrik Manum

September 22, 2005

Contents

1 Problem 1. Optimization

1.1 Case I

Jeg implementerer problemet i MATLAB som vist under.

```
% Oppgave 1a)
clear all;clc
octane = [99 105 95 99]';        % oktantall
benzene = [0 0 0 2]';            % %-innhold
```

```
pris = [0.3 0.2 0.12 0.2]';    % $/kg, altså pris/masse
H = zeros(4,4);
f = pris;
A = [-octane';
    benzene';
    1 0 0 0; % max weigth fraction str 1 0.4
    -eye(4)];
b = [-98 1 0.4 zeros(1,4)]';
Aeq = ones(1,4); % summen av massefraksjonene er 1
beq = 1;
[w,fval,exitflag,output,lambda] = quadprog(H,f,A,b,Aeq,beq);
w,fval,A*w-b
```

1.1.1 Deloppgave a

```
>> fval, A*w - b
fval =
    0.1232
ans =
   -0.0000
   -1.0000
    0.0000
   -0.4000
   -0.1400
   -0.4600
   -0.0000
>>
```

Ved å betrakte output'en fra MATLAB og skriptet (som definerer A og b) ser vi at det er tre akttive ulikheter. I tillegg er selvfølgelig likheten $e^T w = 1$ aktiv, hvor $e = (1, 1, 1, 1)$. De tre aktive ulikhetene er {'oktanspec','maksimal vektfraksjon av strøm 1','$w_4 \geq 0$'}. Prosessen har fire pådrag $u = (w_1, w_2, w_3, w_4)$. Siden vi har 3 aktive ulikheter og en likhetsbetingelse er det ingen flere frihetsgrader igjen.

Parring: En åpenbar parring er kanskje å regulere $w_4 = 0$ med strøm 4. Siden makimal w_1 er aktiv kan vi kanskje bruke strøm 1 for å regulere dette. Videre foreslår jeg å bruke strøm 2 til å holde oktanspesifikasjonen på sin grense og den siste ventilen (strøm 3) til å regulere $e^T w = 1$.

1.1.2 Deloppgave b

```
>> fval, A*w - b
fval =
    0.1432
ans =
   -0.0000
   -1.0000
   -0.0000
```

```
   -0.4000
   -0.1400
   -0.4600
   -0.0000
>>
```

Vi ser at løsningsvektoren er den samme som i deloppgave a, men siden prisen på råvarer har økt har `fval` blitt litt større. (Altså objektfunksjonen). De aktive skranker og antall frihetsgrader den de samme som i a, og regulerings-diskusjonen blir også den samme som over.

1.1.3 Deloppgave c

Det ser ut som om settet av aktive beskrankninger endres når prisen for strøm 1 går over 0.15 \$/kg.

1.1.4 Deloppgave d

```
>> fval, A*w - b
fval =
    0.1175
ans =
   -0.0000
         0
   -0.1500
   -0.2500
         0
   -0.2500
   -0.5000
>>
```

Settet av aktive skranker er nå $\mathcal{A} = \{$'oktan spec','benzene spec','må ha positiv vektfraksjon fra strøm 2'$\}$. I tillegg er selvfølgelig $e^T w = 1$ aktiv. Vi har like mange aktive skranker som vi har pådrag, følgelig har vi ingen flere frihetsgrader igjen hvis vi velger å regulere de aktive skrankene.

Parring: Vi trenger å regulere alle de aktive beskrankningene. Siden $w_2 \geq 0$ er aktiv, bør vi kanskje parre denne med ventilen på strøm 2. Siden den eneste strømmen med benzene er strøm 4, kan vi kanskje parre benzene-spec'en med ventilen på strøm 4. Videre kan vi parre strøm 2 med oktan-spec'en (størst strømning) og tilslutt $e^T w = 1$ med det siste pådraget (ventil på strøm 1), men de siste to parringene kan sikkert byttes hvis de skulle være ønskelig.

1.1.5 Deloppgave e

`pris = [0.3 0.2 0.12 0.2]';` gir aktivt sett $\mathcal{A} = \{$'oktan spec', 'vektfrak. str 1 = 0', 'vektfrak. str 4 = 0' $\}$

2 Problem 2. Optimal operation

i). `fval` med $d = 95$ er 0.1372, mens `fval` med $d = 97$ er 0.1260.

ii).

2.1 Maximum gain

Implementerer problemet i MATLAB på følgende måte:

```
% Oppgave 2)
clc;clear all;
d = 95;
c = 0.1;
octane = [99 105 d 99]';        % oktantall
benzene = [0 0 0 2]';           % %-innhold
pris = [c 0.2 0.12 0.185]';     % $/kg, altså pris/masse
% x := weigth fractions
H = zeros(4,4);
H(1,1) = 2*c; % the price of str 1 is a function of how much we use
f = pris;
A = [-octane';
    benzene';
    1 0 0 0; % max weigth fraction str 1: 0.4
    -eye(4)];
b = [-98 1 0.4 zeros(1,4)]';
Aeq = ones(1,4); % summen av massefraksjonene er 1
beq = 1;
[w0,fval,exitflag,output,lambda] = quadprog(H,f,A,b,Aeq,beq);
w0,fval,A*w0-b
% tester forskjellige mulige kontrollerte variable
% m1 := konst
% må finne span(c=m1)
% Kjører en simulering ved d = 97
d = 97;octane = [99 105 d 99]';        % oktantall
A = [-octane';
    benzene';
    1 0 0 0; % max weigth fraction str 1: 0.4
    -eye(4)];
[w,fval,exitflag,output,lambda] = quadprog(H,f,A,b,Aeq,beq);
w,fval,A*w-b        % w := løsning ved d = 97
% ...søker deretter gain...
% must find how c(==m1) changes with u(==m2), given that the constraints
% must be fulfilled. This gives G. To find Gs you also need to consider the
% optimal variation in each variable
% Active constraints at this point = {'m_tot = 1','m_4 = 0','Octane = 98'}
d = 95;octane = [99 105 d 99]';        % oktantall
```

```
Ac = [1 1 1 1;% sum vektfrak = 1
    octane';  % oktantall = 98
    0 0 0 1;  % m4 := 0
    0 1 0 0]; % m2 := independent variable
delta = 0.05; % m2 er normalisert , 0 <= m2 <= 1
m2 = w0(2) + delta; % pertuberer m2 med delta w0 (løsning ved d=95)
b_ac = [1 98 0 m2]';
w2 = Ac\b_ac;   % holder resten av de aktive beg. konst + gir m2
impl_error = 0.01;
span_m1 = abs(w(1) - w0(1)) + abs(impl_error); % span i m1
gain(1) = abs((w2(1) - w0(1))/delta/span_m1) % gain c (m1) -> m2
Gain{1} = 'm1'
% m2 := konstant
% uskalert gain fra m2 -> m2 må være 1.
g_m2_m2 = (w2(2) - w0(2))/delta; % == 1
span_m2 = abs(w(2) - w0(2)) + abs(impl_error)
gain(2) = abs(g_m2_m2)/span_m2
Gain{2} = 'm2'
% m3 := konstant
g_m3_m2 = (w2(3) - w0(3))/delta;
span_m3 = abs(w(3) - w0(3)) + abs(impl_error)
gain(3) = abs(g_m3_m2)/span_m3
Gain{3} = 'm3'
% m1/m2 := konstant
g_m1m2_m2 = (w2(1)/w2(2) - w0(1)/w0(2))/delta
span_m1m2 = abs(w(1)/w(2) - w0(1)/w0(2)) + ...
    max([abs((w(1)+impl_error)/w(2)) abs((w(1)-impl_error)/w(2)) ...
    abs((w(1))/(w(2)+impl_error)) abs((w(1))/(w(2)-impl_error))])
gain(4) = abs(g_m1m2_m2)/span_m1m2
Gain{4} = 'm1/m2'
% linear combination of m1 and m2
% må først finne optimal kombinasjon
% bør prøve være i nærheten av optimalt område, og bare pertubere d litt
% for å finne sensitivitetsmatrisa F
% "maksimalt" utslag i d = 95 + 2. Ønsker å sjekke et mindre utslag
dd = 95 + 2*0.05;
octane = [99 105 dd 99]';        % oktantall
A = [-octane';
    benzene';
    1 0 0 0; % max weigth fraction str 1: 0.4
    -eye(4)];
[wd,fval,exitflag,output,lambda] = quadprog(H,f,A,b,Aeq,beq);
F(1,1) = (w0(1) - wd(1))/2/0.05;
F(2,1) = (w0(2) - wd(2))/2/0.05;
F(3,1) = (w0(3) - wd(3))/2/0.05;
F(4,1) = (w0(4) - wd(4))/2/0.05;
% m1 og m2
h1{1} = 1;
h2{1} = -F(1,1)/F(2,1);
```

```
% m2 og m3
h1{2} = 1;
h2{2} = -F(2,1)/F(3,1);
% m3 og m4  Egentlig menigsløst, da m4 er brukt fra før (m4=0 er aktiv)
h1{3} = 1;
h2{3} = -F(3,1)/F(4,1);  % blir veldig stor. altså bruk bare m4, noe vi visste
                         % bektakter derfor bare m1 - m2 og m2 - m3
% m1 + h2{1}*m2
g_l1_m2 = ((w2(1)+h2{1}*w2(2))- (w0(1)+h2{1}*w0(2)))/delta;
span_l1 = abs((w(1)+h2{1}*w(2)) - (w0(1)+h2{1}*w0(2))) + abs((1+abs(h2{1}))...
    *impl_error)
gain(5) = abs(g_l1_m2)/span_l1
Gain{5} = 'm1-0.53m2'

% m2 + h2{2}*m3
g_l2_m2 = ((w2(2)+h2{2}*w2(3))- (w0(2)+h2{2}*w0(3)))/delta;
span_l2 = abs((w(2)+h2{2}*w(3)) - (w0(2)+h2{2}*w0(3))) + abs((1+abs(h2{2}))...
    *impl_error)
gain(6) = abs(g_l2_m2)/span_l2
Gain{6} = 'm2+0.65m3'
```

"Resultatet" av analysen er vist i tabell 1. Jeg droppet å ta med lineærkombinasjonen av m_3 og m_4, da skranken $m_4 \geq 0$ er aktiv.

c	G	span(c)	$\lvert G_s\rvert = \frac{\lvert G\rvert}{\text{span}(c)}$	Rank
m_1	-2.5	0.07	35.7	3
m_2	1.0	0.1310	7.6	5
m_3	1.5	0.1910	7.9	4
m_1/m_2	15.6	4.42	3.5	6
$m_1 - 0.53m_2$	3.03	0.02	152.1	1
$m_2 + 0.65m_3$	1.978	0.0195	101.3	2

Table 1: Maximum gain

2.2 Bruteforce

Impelmentering i MATLAB:

```
% brute force evaluering av de valgte kontrollerte variable. Kjør case2.m
% først for å genrere variable etc...
cost = inline('0.5*x''*H*x + f''*x','x','H','f');
d = 97;octane = [99 105 d 99]';        % oktantall
A = [-octane';
    benzene';
    1 0 0 0; % max weigth fraction str 1: 0.4
    -eye(4)];
```

```
b = [-98 1 0.4 zeros(1,4)]';
% m1 := konstant
d = 97;octane = [99 105 d 99]';        % oktantall
D = [octane';
    1 1 1 1;
    0 0 0 1;
    1 0 0 0;]
c = [98 1 0 0.26]'; % m1 = 0.26 fra optimalisering
n = 1;
c(4) = 0.26 + impl_error;
if (A*(D\c)-b) <= 0
    j(n).pluss = cost(D\c,H,f)
else
    j(n).pluss = 'infeasible'
end
c(4) = 0.26 - impl_error;
if (A*(D\c)-b) <= 0
    j(n).minus = cost(D\c,H,f)
else
    j(n).minus = 'infeasible'
end
j(n).worst = max([j(n).pluss j(n).minus])
% m2 := konstant
d = 97;octane = [99 105 d 99]';        % oktantall
D = [octane';
    1 1 1 1;
    0 0 0 1;
    0 1 0 0;]
c = [98 1 0 w0(2)]';
n = 2;
c(4) = w0(2) + impl_error;
if (A*(D\c)-b) <= 0
    j(n).pluss = cost(D\c,H,f)
else
    j(n).pluss = 'infeasible'
end
c(4) = w0(2) - impl_error;
if (A*(D\c)-b) <= 0
    j(n).minus = cost(D\c,H,f)
else
    j(n).minus = 'infeasible'
end

j(n).worst = max([j(n).pluss j(n).minus])
% m3 := konstant
d = 97;octane = [99 105 d 99]';        % oktantall
D = [octane';
    1 1 1 1;
    0 0 0 1;
```

```
    0 0 1 0;]
c = [98 1 0 w0(3)]';
n = 3;
c(4) = w0(3) + impl_error;
if (A*(D\c)-b) <= 0
    j(n).pluss = cost(D\c,H,f)
else
    j(n).pluss = 'infeasible'
end
c(4) = w0(3) - impl_error;
if (A*(D\c)-b) <= 0
    j(n).minus = cost(D\c,H,f)
else
    j(n).minus = 'infeasible'
end
j(n).worst = max([j(n).pluss j(n).minus])
% m1/m2 := konstant
d = 97;octane = [99 105 d 99]';        % oktantall
D = [octane';
    1 1 1 1;
    0 0 0 1;
    1 -w0(1)/w0(2) 0 0;]
c = [98 1 0 0]';
n = 4;
c = [98 1 0 impl_error*(1+(w0(1)/w0(2)))]'; % impl_error i m1/m2
if (A*(D\c)-b) <= 0
    j(n).pluss =  cost(D\c,H,f)
else
    j(n).pluss = 'infeasible'
end
c = [98 1 0 -impl_error*(1+(w0(1)/w0(2)))]';
if (A*(D\c)-b) <= 0
    j(n).minus =  cost(D\c,H,f)
else
    j(n).minus = 'infeasible'
end
j(n).worst = max([j(n).pluss j(n).minus])
% lineærkombinasjonen l1 = m1 + h2{1}*m2
d = 97;octane = [99 105 d 99]';        % oktantall
D = [octane';
    1 1 1 1;
    0 0 0 1;
    1 h2{1} 0 0;]
konst = w0(1) + h2{1}*w0(2);
c = [98 1 0 konst]';
n = 5;
c(4) = konst + (abs(1)+abs(h2{1}))*impl_error
if (A*(D\c)-b) <= 0
    j(n).pluss = cost(D\c,H,f)
```

```
else
    j(n).pluss = 'infeasible'
end
c(4) = konst - (abs(1)+abs(h2{1}))*impl_error
if (A*(D\c)-b) <= 0
    j(n).minus = cost(D\c,H,f)
else
    j(n).minus = 'infeasible'
end
j(n).worst = max([j(n).pluss j(n).minus])
% lineærkombinasjonen l2 = m2 + h2{2}*m3
d = 97;octane = [99 105 d 99]';        % oktantall
D = [octane';
    1 1 1 1;
    0 0 0 1;
    0 1 h2{2}  0;]
konst = w0(2) + h2{2}*w0(3);
c = [98 1 0 konst]';
n = 6;
c(4) = konst + (abs(1)+abs(h2{2}))*impl_error
if (A*(D\c)-b) <= 0
    j(n).pluss = cost(D\c,H,f)
else
    j(n).pluss = 'infeasible'
end
c(4) = konst - (abs(1)+abs(h2{2}))*impl_error
if (A*(D\c)-b) <= 0
    j(n).minus = cost(D\c,H,f)
else
    j(n).minus = 'infeasible'
end
j(n).worst = max([j(n).pluss j(n).minus])

j_opt_d = cost(w,H,f);  % optimal cost at d = 0.97 (see case2.m)
for i = 1:6 j(i).loss = j(i).worst - j_opt_d; end
```

Resultatet av analysen er samlet i tabell 2. Man ser at denne analysen gir samme resultat som maximum gain analysen, nemlig at lineærkombinasjonen av m_1 og m_2 er det beste valget vi kan gjøre.

Kandidat variabel	$L(d) = J(c_s + n, d) - J_{\text{opt}}(d)$
m_1	$4.9 \cdot 10^{-4}$
m_2	infeasible
m_3	infeasible
m_1/m_2	$8.6 \cdot 10^{-4}$
$m_1 - 0.53m_2$	$3.1 \cdot 10^{-5}$
$m_2 + 0.65m_3$	$7.0 \cdot 10^{-5}$

Table 2: Brute force

2.3 Would your results change if we included the implementation error on the active constraints?

Det er opplagt at de aktive skrankene skal kontrolleres, så dette ville ikke endres. Når disse regulerte variable pga implementasjonsfeil ikke er ved sitt settpunkt regner jeg med at avvikene kan betraktes som forstyrrelser, slik at dimensjonen på d øker. Analysen blir den samme, men vi må undersøke hvordan de regulerte variable "takler" de nye forstyrrelsene, og dette kan naturligvis påvirke valget av kontrollerte variable.