VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Optimizavimo metodai

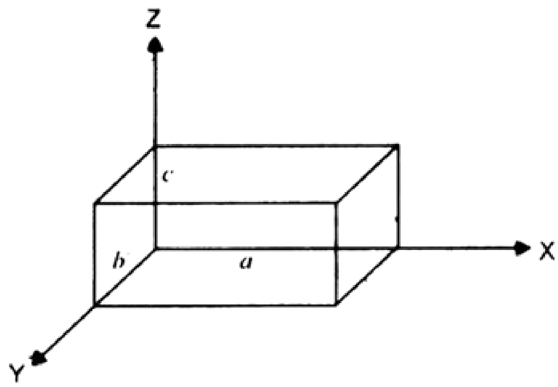
Laboratorinis darbas nr. 3

Netiesinis programavimas

Tomas Giedraitis
VU MIF Informatika
3 kursas 3 grupė

Vilnius
2019

Pagrindinis klausimas: Kokia turėtų būti stačiakampio gretasienio formos dėžė, kad vienetiniam paviršiaus plotui jos tūris būtų maksimalus?



Pažymėkime kraštinių ilgius x_1 , x_2 , x_3 $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0$.

Išreikškime dėžės paviršiaus plotą (priekinės ir galinės sienų plotų sumą, šoninių sienų plotų sumą, viršutinės ir apatinės sienų plotų sumą) per kraštinių ilgius:PenaltyMethod

$$S_{pav} = 2x_1x_2 + 2x_1x_3 + 2x_2x_3 = 1$$

Paverskime reikalavimą vienetinio dėžės paviršiaus plotui į lygybinį apribojimą:

$g(x_1, x_2, x_3) = 0$, kur $g(x_1, x_2, x_3) = 2x_1x_2 + 2x_1x_3 + 2x_2x_3 - 1$,

o apribojimus kraštinėms apirbrėžkime nelygybiniais apribojimais:

$h_1(x) \leq 0$, kur $h_1(x) = -x_1$,

$h_2(x) \leq 0$, kur $h_2(x) = -x_2$,

$h_3(x) \leq 0$, kur $h_3(x) = -x_3$,

arba $h_i(x) \leq 0$, kur $h_i(x) = -x_i$,

Formuojame optimizavimo su apribojimais uždavinį:

Dėžės tūris:
$V = x_1x_2x_3$

Ieškosime dežės parametrų esant maksimaliam tūriui: max(V), arba min(-V). Gauname tikslo funkciją, kurią minimizuosime:

$$f(x_1, x_2, x_3) = -x_1x_2x_3$$

Apsirašome vektorių $X=[x_1, x_2, x_3]$.

Aprašome kvadratinę baudos funkciją, apimančią tikslo funkciją ir apribojimus:

$$B(X,r)=f(X)+\frac{1}{r}b(X) \quad ,$$

kur *r (r > 0)* – baudos daugiklis (pasirenkamas), $b(X)$ - kvadratinė funkcija, susidedanti iš apribojimų funkcijų:

$$b(X)=(g(X))^2+max(0,h_1(X))^2+max(0,h_2(X))^2+max(0,h_3(X))^2 \quad ,$$

čia $f(x_1,x_2,x_3)$ pakeičiame į $f(X)=-X(1)X(2)X(3)$,

$g(x_1,x_2,x_3)$ pakeičiame į $g(X)=2X(1)X(2)+2X(1)X(3)+2X(2)X(3)-1$,

o $h_i(x)$ į $h_i(X)=-X(i)$.

<u>*r* (baudos daugiklio) įtaka baudos funkcijos reikšmėms</u>:

[1] Kai *r > 1, r* vis didėjant, *B(X,r)* funkcijos antrasis dėmuo ($\frac{1}{r}b(X)$) mažėja, ir *B(X,r)* reikšmė mažėja.

[2] Tuo tarpu, kai *r < 1,* ir *r* vis mažėja, $\frac{1}{r}b(X)$ didėja, ir *B(X,r)* reikšmė didėja.

Apskaičiuokime funkcijų $f(X)$, $g(X)$ ir $h_i(x)$ reikšmes taškuose $X_0=[0,0,0]$, $X_1=[1,1,1]$, $X_m=[0.1, 0.4, 0.7]$.

Taške $X=X_0$:

$$f(X)=-0*0*0=0 \quad ,$$

$$g(X)=2*0*0+2*0*0+2*0*0-1=-1 \quad ,$$

$$h_i(X)=0, i=1,2,3 \quad .$$

Taške $X=X_1$:

$$f(X)=-1*1*1=-1 \quad ,$$

$$g(X)=2*1*1+2*1*1+2*1*1-1=6-1=5 \quad ,$$

$h_i(X) = -1, i = 1, 2, 3$.

Taške $X = X_m$:

$f(X) = -0.1*0.4*0.7 = -0.028$,

$g(X) = 2*0.1*0.4 + 2*0.1*0.7 + 2*0.4*0.7 - 1 = 1.54$,

$h_1(X) = -0.1$,
$h_2(X) = -0.4$,

$h_3(X) = -0.7$.

Minimizuosime baudos funkciją optimizavimo be apribojimų algoritmu (deformuojamo simplekso) sprendžiant optimizavimo uždavinių seką su mažėjančia parametro *r* seka ($r \rightarrow 0$), kai pirmasis sekos uždavinys optimizuojamas pradedant iš taškų X_0 , X_1 ir X_m , o kiekvieno paskesnio uždavinio pradinis taškas yra ankstesnio uždavinio sprendinys.

Gauti sprendiniai priklausomai nuo pradinių taškų (artinių)

[1] Pradinis artinys: $X_0 = [0, 0, 0]$

Programos išvestis (iteracijų rezultatai):

x1	x2	x3	f(X)	B(X,r)	k	(funkc. kviet. sk)
0.418788	0.418810	0.418795	-0.0734536	-0.0707131	1	249
0.413486	0.413492	0.413491	-0.0706959	-0.0693602	2	544
0.410861	0.410861	0.410861	-0.069356	-0.0686966	3	1013
0.409559	0.409553	0.409546	-0.0686956	-0.0683679	4	1266
0.408900	0.408900	0.408900	-0.0683677	-0.0682044	5	1555
0.408574	0.408574	0.408574	-0.0682043	-0.0681228	6	1804
0.408416	0.408406	0.408411	-0.0681228	-0.0680821	7	1964
0.408329	0.408330	0.408329	-0.0680821	-0.0680617	8	2183
0.408290	0.408287	0.408289	-0.0680617	-0.0680516	9	2393

[2] Pradinis artinys: $X_1 = [1, 1, 1]$

Programos išvestis (iteracijų rezultatai):

```
  x1         x2        x3          f(X)          B(X,r)     k   (funkc. kviet. sk)
-----------------------------------------------
0.418797 0.418798 0.418798 -0.0734537 -0.0707131 1 896
0.413576 0.413758 0.413138 -0.0706964 -0.0693602 2 1277
0.410862 0.410876 0.410844 -0.0693559 -0.0686966 3 1638
0.409552 0.409552 0.409553 -0.0686955 -0.0683679 4 2179
0.408865 0.408889 0.408944 -0.0683673 -0.0682044 5 2391
0.408574 0.408574 0.408574 -0.0682043 -0.0681228 6 2622
0.408412 0.408412 0.408410 -0.0681228 -0.0680821 7 2793
0.408331 0.408326 0.408331 -0.0680821 -0.0680617 8 2996
0.408287 0.408300 0.408280 -0.0680617 -0.0680516 9 3341
```

[3] Pradinis artinys: $X_m = [0.1, 0.4, 0.7]$

Programos išvestis (iteracijų rezultatai):

```
x1         x2        x3          f(X)          B(X,r)     k   (funkc. kviet. sk)
-----------------------------------------------
0.418798 0.418798 0.418798 -0.0734536 -0.0707131 1 589
0.413493 0.413487 0.413489 -0.070696 -0.0693602 2 924
0.410861 0.410861 0.410861 -0.069356 -0.0686966 3 1485
0.409554 0.409558 0.409545 -0.0686956 -0.0683679 4 1736
0.408901 0.408900 0.408899 -0.0683677 -0.0682044 5 2157
0.408574 0.408574 0.408574 -0.0682043 -0.0681228 6 2360
0.408411 0.408411 0.408411 -0.0681228 -0.0680821 7 2534
0.408329 0.408329 0.408331 -0.0680821 -0.0680617 8 2749
0.408289 0.408288 0.408289 -0.0680617 -0.0680516 9 3124
```

Rezultatų palyginimas

Pradinis artinys	k (iteracijų skaičius)	Funkcijos iškvietimų skaičius	Tikslo funkcijos reikšmė $min\ f(X)$	Baudos funkcijos reikšmė, $min\ B(X, r)$	Argumentas x_1	Argumentas x_2	Argumentas x_3
X_0	9	2393	-0.0680617	*-0.0680516*	*0.408290*	*0.408287*	*0.408289*
X_1	9	3341	-0.0680617	-0.0680516	0.408287	0.408300	0.408280
X_m	9	3124	-0.0680617	-0.0680516	0.408289	0.408288	0.408289

Bandymų išvados:

[1] Naudojant baudos metodą, su kiekvienu naudotu artiniu X_0 , X_1 ir X_m funkcijos minimumas buvo rastas per 9 iteracijas. Iš rezultatų lentelės matome ir tikslo funkcijos reikšmę įsistačius rastus argumentus, kurios reikšmė yra labai arti baudos funkcijos reikšmės. Iš to galime spręsti, kad baudos funkcija pakankamai gerai aproksimuoja tikslo funkciją šiais atvejais.

[2] Tuo tarpu funkcijų kvietimų skaičius kiekvienam artiniui buvo skirtingas. Mažiausiai funkcijų kvietimų buvo panaudota X_0 atveju (2393), ir atotrūkis didesnis nei tarp f-jų kvietimų skaičių kitiems dviems artiniams (atitinkamai 3341 (X_1) ir 3124 (X_m)).

[3] Viena iš esminių įžvalgų – baudos parametro *r* mažėjimas kiekvieną iteraciją, ir to įtaka baudos funkcijos reikšmei. Mažindami *r,* taigi *r* artėjant į nulį, kvadratinės baudos funkcijos minimumas irgi artėja į tikslo funkcijos minimumą.

Šiame bandyme pradinis *r* prieš pradedant iteracijas buvo pasirinktas *r* = 1.
Kas iteraciją *r* buvo mažinamas taip: *r* = *r* / 2.

Eksperimento patobulinimui galima būtų pasirinkti skirtingus *r* mažinimo būdus, ir palyginti gautus rezultatus.

Baudos metodo kodas:

```
function PenaltyMethod

% tikslo funkcija
f = @(X) -X(1) .* X(2) .* X(3);

% apribojimu funkcijos
g = @(X) 2*X(1) .* X(2)+2*X(1) .* X(3) +2*X(2) .* X(3) - 1;

h1 = @(X) -X(1);
h2 = @(X) -X(2);
h3 = @(X) -X(3);

% kvadratine baudos funkcija su apribojimais
b = @(X) (g(X)).^2 + max(0,h1(X)).^2 + max(0,h2(X)).^2 + max(0,h3(X)).^2;

% kvadratine baudos funkcija, apimanti tikslo funkcija ir apribojimus
B = @(X,r) f(X) + (1/r) * b(X);

% pasirenkame, kaip maziname parametra r kiekviena iteracija
Fn_decrease_r = @(r) r/2;

% pradiniai artiniai
X_0 = [0, 0, 0];
X_1 = [1, 1, 1];
X_m = [1/10, 4/10, 7/10];

% pasirenkamas pradinis artinys
X0 = X_m;

% pasirenkamas pradinis baudos daugiklis
r = 1;

% tikslumas
epsilon = 10 ^ (-4);

k = 1; % iteraciju skaitliukas (pradinio simplekso sudarymas = 1 iteracija)
i = 0; % funkcijos kvietimu skaiciaus skaitliukas
kmax = 100; % maksimalus iteraciju skaicius
imax = 100; % maksimalus funkcijos kvietimu skaicius

format short;

% Metodo realizavimas
disp(['   x1  x2  x3    f(X) B(X,r)   k    (f kv. sk.)']);
disp('-----------------------------------------------');

norma = Inf;


while norma >= epsilon
```

```
    res=Simplex(B,X0,r);
    % naujas artinys
    X1 = res(1:3);
    norma = norm(X0-X1);
    % sekame funkciju kvietimu skaiciu
    i = i + res(4) + 1;

    fprintf('%f %f %f %d %d %d %d\n', X1, f(X1), B(X1,r), k, i);

    % maziname r
    r = Fn_decrease_r(r);
    X0 = X1;
    k = k+1;

endwhile

endfunction
```

Deformuojamo simplekso su 3 argumentais kodas:

```
function res=Simplex(B,X0,r)

f = @(X) B(X,r);

% pasirenkami parametrai
alpha = 0.5; % reguliuoja pradinio simplekso krastines ilgi
teta = 1.0; % reguliuoja tieses lygti, breziamos per vidurio taska, ieskant naujos virsunes
% simplekso deformavimo koeficientai
gamma = 2.0; % reguliuoja simplekso ispletima, gamma > 1
beta = 0.5; % reguliuoja simplekso suspaudima, 0 < beta < 1
eta = - 0.5; % reguliuoja simplekso suspaudima, -1 < eta < 0

% tikslumas
epsilon = 10 ^ (-6);

% Pradinio simplekso sudarymas
n = 3; % keliu kintamuju funkcija yra minimizuojama
delta1 = alpha * (sqrt(n + 1) + n - 1) / (n * sqrt(2));
delta2 = alpha * (sqrt(n + 1) - 1) / (n * sqrt(2));

% kitos simplekso virsunes (apskaiciuojame pagal teorine medziaga)
X1 = [X0(1, 1) + delta2, X0(1, 2) + delta1, X0(1, 3) + delta1];
X2 = [X0(1, 1) + delta1, X0(1, 2) + delta2, X0(1, 3) + delta1];
X3 = [X0(1, 1) + delta1, X0(1, 2) + delta1, X0(1, 3) + delta2];

% funkcijos reiksmes simplekso virsunese
y0 = f(X0);
y1 = f(X1);
y2 = f(X2);
y3 = f(X3);

% simplekso virsuniu masyvas
X = [X0; X1; X2; X3];

% funkcijos reiksmiu simplekso virsunese masyvas
y = [y0, y1, y2, y3];

k = 1; % iteraciju skaitliukas (pradinio simplekso sudarymas = 1 iteracija)
i = 4; % funkcijos kvietimu skaiciaus skaitliukas
kmax = 100; % maksimalus iteraciju skaicius
imax = 100; % maksimalus funkcijos kvietimu skaicius

format short;

% Metodo realizavimas

goal = false;
while ~ goal
   % Randami Xh, Xg, Xl ir funkcijos reiksmes siuose taskuose yh, yg, yl
   [~, nr] = sort(y); % y0, y1, y2, y3 reiksmes isdestomos didejimo tvarka; nr rodys ju numerius masyve y
```

```
    yl = y(nr(1)); % maziausia y reiksme
    Xl = X(nr(1), :);

    yh = y(nr(n + 1)); % didziausia y reiksme
    Xh = X(nr(n + 1), :);

    yg = y(nr(n)); % antra pagal dydi y reiksme
    Xg = X(nr(n), :);

    yi = y(nr(2)); % likusi vidurine reiksme
    Xi = X(nr(2), :);

    % Viduriu tasko Xc ir naujo artinio Xnew apskaiciavimas
    Xc = (Xg + Xl + Xi) / 3;

    Xnew = Xh + (1 + teta) * (Xc - Xh);
    ynew = f(Xnew);
    i = i + 1;

    % Naujo simplekso sudarymas
    if (yl < ynew) && (ynew < yg)
        teta = 1;
    elseif ynew < yl
        teta = gamma;
        Z = Xh + (1 + teta) * (Xc - Xh);
        yz = f(Z);
        i = i + 1;
        if yz < ynew
            ynew = yz;
            Xnew = Z;
        endif
    elseif ynew > yh
        teta = eta;
        Z = Xh + (1 + teta) * (Xc - Xh);
        Xnew = Z;
        ynew = f(Z);
        i = i + 1;
    elseif (yg < ynew) && (ynew < yh)
        teta = beta;
        Z = Xh + (1 + teta) * (Xc - Xh);
        Xnew = Z;
        ynew = f(Z);
        i = i + 1;
    endif

    count = 0;

    if max([norm(Xl - Xi), norm(Xl - Xg), norm(Xl - Xh), norm(Xi - Xg), norm(Xi - Xh), norm(Xg - Xh)]) <
epsilon
        count = count + 1;
    endif

    if max([abs(yl - yi), abs(yl - yg), abs(yl - yh), abs(yi - yg), abs(yi - yh), abs(yg - yh)]) < epsilon
```

```
        if ~count
        endif
        count = count + 1;
    endif

    if i >= imax
        count = count + 1;
        if count == 3
            goal = true;
        endif
    endif

    k = k + 1;
    % naujas artinys
    X = [Xl; Xi; Xg; Xnew];
    % funkcijos reiksmes naujame artinyje
    y = [yl, yi, yg, ynew];


endwhile
res=[Xnew,i];
endfunction
```