Optimizavimo metodai

3 lab.

Netiesinis programavimas

Lukas Miežys

**Baudos funkcija**

**function kvadratinebauda**

**f=@(X) -X(1).\*X(2).\*X(3);**

**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);**

**b=@(X) (min(0,h1(X))).^2+(min(0,h2(X))).^2+(min(0,h3(X))).^2+(g(X)).^2;**

**B=@(X,r) f(X)+(1/r)\*b(X);**

**epsilon = 10^(-4);**

**kmax=100;**

**k=1;**

**i=1;**

**X0=[1,1,1];**

**r=1;**

**n=3;**

**norma=Inf;**

**format short**

**disp(['r X B(X,r) k (f kv. sk.), turis']);**

**disp('---------------------------------------------------------');**

**while norma>=epsilon**

**ats=Simplexo(B,X0,r);**

**X1=ats(1:n);**

**i=i+ats(n+1);**

**norma=norm(X0-X1);**

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

**if k==kmax**

**format short**

**disp(['Iteraciju skaicius k=',num2str(kmax)]);**

**break**

**end**

**k=k+1;**

**X0=X1;**

**r=r/5;**

**end**

**end**

**Simpleksas baudos funkcijai**

**function ats=Simplexo(B,X0,r)**

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

**%pasirenkam parametrus**

**alpha = 1/2; %pradinio simplekso krastinei**

**teta = 1; %naujos virsunes ieskojimui**

**%simplekso deformavimui**

**gamma = 2; %simplekso ispletimas, gamma >1**

**beta = 0.5;%simplekso suspaudimas, 0<beta<1**

**eta = - 0.5;%simplekso suspaudimas, -1<eta<0**

**epsilon = 10 ^ (- 6); %tikslumas**

**%pradinio simplekso sudarymas**

**n = 3; %dvieju kintamuju funkcija minimizuojam**

**delta1 = alpha \* (sqrt(n + 1) + n - 1) / (n \* sqrt(2));**

**delta2 = alpha \* (sqrt(n + 1) - 1) / (n \* sqrt(2));**

**%kitu virsuniu apskaiciavimas**

**X1 = [X0(1) + delta2, X0(2) + delta1, X0(3)+delta1];**

**X2 = [X0(1) + delta1, X0(2) + delta2, X0(3)+delta1];**

**X3 = [X0(1) + delta1, X0(2) + delta1, X0(3)+delta2];**

**%funkcijos reiksmes virsunese**

**y0 = f(X0);**

**y1 = f(X1);**

**y2 = f(X2);**

**y3 = f(X3);**

**%simplekso virsuniu masyvas**

**X = [X0; X1; X2; X3];**

**%funkcijos reiksmiu masyvas virsunese**

**y = [y0, y1, y2, y3];**

**k = 1; %iteracijos**

**i = 3; %funkcijos kvietimai**

**kmax = 100;% maksimalus iteraciju**

**imax = 100;%maksimalus funkcijos kvietimu**

**pasiekta = false;**

**%pradedame vykdyti iteracijas**

**% Randami Xh, Xg, Xl ir funkcijos reiksmes siuose taskuose yh, yg, yl**

**while ~ pasiekta**

**[~, nr] = sort(y);% y0, y1, y2 reiksmes isdestomos didejimo tvarka; nr rodys ju numerius masyve y**

**%didziausias y**

**yhigh = y(nr(4));**

**Xhigh = X(nr(4), :);**

**%maziausias y**

**ylow = y(nr(1));**

**Xlow = X(nr(1), :);**

**%antras y**

**yg = y(nr(2));**

**Xg = X(nr(2), :);**

**%trecias y**

**yi = y(nr(3));**

**Xi = X(nr(3), :);**

**% Viduriu tasko Xc ir naujo artinio Xnew apskaiciavimas**

**Xc = (Xg + Xlow+Xi) / 3;**

**X\_naujas = Xhigh + (1 + teta) \* (Xc - Xhigh);**

**y\_naujas= f(X\_naujas);**

**i = i + 1;**

**% Naujo simplekso sudarymas**

**if (ylow < y\_naujas) && (y\_naujas < yg)**

**teta = 1;**

**elseif y\_naujas < ylow %ispleciame**

**teta = gamma;**

**Z = Xhigh + (1 + teta) \* (Xc - Xhigh);**

**yz = f(Z);**

**i = i + 1;**

**if yz < y\_naujas**

**y\_naujas = yz;**

**X\_naujas = Z;**

**endif**

**elseif y\_naujas > yhigh %perverciame**

**teta = eta;**

**Z = Xhigh + (1 + teta) \* (Xc - Xhigh);**

**X\_naujas = Z;**

**y\_naujas = f(Z);**

**i = i + 1;**

**elseif (yg < y\_naujas) && (y\_naujas < yhigh) %suspaudziame**

**teta = beta;**

**Z = Xhigh + (1 + teta) \* (Xc - Xhigh);**

**X\_naujas = Z;**

**y\_naujas = f(Z);**

**i = i + 1;**

**endif**

**count = 0;**

**if max([norm(Xlow - Xg), norm(Xlow - Xhigh), norm(Xg - Xhigh),norm(Xlow - Xi),norm(Xg - Xi),norm(Xi - Xhigh)]) < epsilon**

**count = count + 1;**

**endif**

**if max([abs(ylow - yg), abs(ylow - yhigh), abs(yg - yhigh),abs(ylow - yi),abs(yg - yi),abs(yi - yhigh)]) < epsilon**

**count = count + 1;**

**endif**

**if i >= imax**

**count = count + 1;**

**if count == 3**

**pasiekta = true;**

**endif**

**endif**

**k = k + 1;**

**%naujas artinys**

**X = [Xlow; Xg; Xi; X\_naujas];**

**% funckijos reiksmes naujame artinyje**

**y = [ylow, yg, yi, y\_naujas];**

**endwhile**

**ats=[X\_naujas,i];**

**end**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Baudos metodas | | | | | | | | |
| Pradiniai artiniai | B(X,r) | Tūris | žingsniai | Funkcijos kvietimai | X | Y | Z | r |
| X0 | -0.068062 | 0.068368 | 4 | 1410 | 0.408331 | 0.408332 | 0.408332 | 0.008000 |
| X1 | -0.068062 | 0.068368 | 4 | 1781 | 0.408328 | 0.408342 | 0.408324 | 0.008000 |
| Xm | -0.068062 | 0.068368 | 4 | 2508 | 0.408332 | 0.408332 | 0.408331 | 0.008000 |

Išvados

Visi pradiniai taškai konverguoja, Baudos funkcijos galutinė reikšmė yra visų vienoda, iteracijos taip pat vienodas skaičius, tačiau daugiausiai funkcijos kvietimų buvo atlikta su Xm tašku (2508), o mažiausiai su X0 tašku(1410). Baudos indeksas (daugiklis) r galiniame rezultate visų vienodas, nes iteracijų visuose buvo įvykdyta vienodai. O gauti taškai X, Y,Z su visais pradiniais taškais rezultatai beveik vienodi.