VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

FUNDAMENTINIŲ MOKSLŲ FAKULTETAS

INFORMACINIŲ TECHNOLOGIJŲ KATEDRA

PREKYBOS BAZĖS UŽDAVINYS

Laboratorinis darbas 1

Atliko: DGTfm-18 gr. st. Laura Cvilikaitė

Priėmė: Lekt. dr. Aleksandr Igumenov

Vilnius, 2019

I užduotis (Prekybos bazės uždavinys) (patikslinta)

Prekybos bazė įsipareigojo saugoti prekes ir išduoti jas vartotojui kiekvieną dieną po b^* tonų. Ji gali gauti prekes periodiškai kas t parų vienodo dydžio partijomis po q tonų. Vienos prekių partijos priėmimas ir pakrovimas į sandėlį kainuoja c_2 Lt. Vienos tonos vidutinės išdavimo išlaidos $c_1 = c^* \cdot t$ Lt. Naujos prekės vežamos paskutinę anksčiau atvežtų prekių išdavimo dieną. Reikia nustatyti optimalią vienos prekių partijos apimtį q ir jos vežimo periodą t, kad prekybos bazės išlaidos per parą būtų mažiausios. Taip pat (mod - liekana):

$$b^* = \frac{stud_{nr}\ mod\ 7}{2} + 2.5 = \frac{20182746\ mod\ 7}{2} + 2.5 = \frac{3}{2} + 2.5 = 4$$

$$c_2 = (stud_{nr}\ mod\ 30) + 150 = (20182746\ mod\ 30) + 150 = 6 + 150 = 156$$

$$c^* = (stud_{nr}\ mod\ 5) + 3 = 4$$

1. **Sudarykite uždavinio matematinį modelį.**

b^* – tonų skaičius, išduodamas vartotojui kiekvieną dieną;
t – parų skaičius;
q – tonų skaičius;
c_2 – vienos prekių partijos pakrovimo ir priemimo į sandėlį kaina;
$c_1 = c^* \cdot t$ – vienos tonos vidutinės išdavimo išlaidos;
Vienkartinės (vienos dienos) prekių išdavimo išlaidos – $c_1 \cdot b^*$ Lt
Vienos dienos prekių priemimo ir pakrovimo į sandėlį išlaidos - $\frac{c_2}{t}$ Lt
Prekybos bazės išlaidos per parą aprašomos kintamojo t funkcija:

$$c(t) = \frac{c_2}{t} + c_1 \cdot b^* = \frac{c_2}{t} + c^* \cdot t \cdot b^*$$

Be to, žinome, kad $t = \frac{q}{b^*}$, todėl $q = t \cdot b^*$
Taigi, uždavinio matematinis modelis:

$$c(t) = \frac{c_2}{t} + c^* \cdot t \cdot b^*$$

2. **Užrašykite tikslo funkciją.**

Kadangi reikia ieškoti mažiausių išlaidų per parą, naudojame minimumo funkciją.

$$\min(c(t)) = min_{t \in N}\left(\frac{c_2}{t} + c^* \cdot t \cdot b^*\right)$$

3. **Matlab aplinkoje suprogramuokite intervalo dalijimo pusiau metodą (tegu a= 1, b= 30**).

1) Tikslo funkcija

```
function [ct] =  tikslo_funkcija(t)       %tikslo funkcija, priklausanti nuo
                                          %laiko t
    b = mod(20182746, 7)/2 + 2.5;         %parametras b duotas sąlygoje,
                                          %įsistatome savo stud_nr
    c2 = mod(20182746, 30) + 150;         %parametras c2 duotas sąlygoje,
                                          %įsistatome savo stud_nr
    c = mod(20182746, 5) + 3;             %parametras c duotas sąlygoje,
                                          %įsistatome savo stud_nr
    ct = b*c*t + c2./t;                   %tikslo funkcijos išraiška
end
```

2) Intervalo dalijimo pusiau metodas

```
                                      %intervalo dalijimo pusiau metodas
function [x_min_m, f_min, nauja_a, nauja_b, iteration, L1, xpoint, xvalue] =
int_dal_metodas(a, b)
    iteration = 1;                 %pirma iteracija
    x_m = (a + b)/2;               %x_m - intervalo vidurio taškas, a -
                                   %intervalo pradžia, b - intervalo pabaiga
    fx_m = tikslo_funkcija(x_m);   %tikslo funkcijos reikšmė intervalo
                                   %vidurio taške
    x_min_m = x_m;                    %turima minimali intervalo vidurio
                                      %koordinatė
    fx_m1 = fx_m;                     %nauja funkcijos reikšmė
    fx_m2 = fx_m;                     %nauja funkcijos reikšmė
    xpoint(iteration) = x_m;          %išsaugomas taškas x_m
    xvalue(iteration) = fx_m1;        %išsaugoma funkcijos reikšmė x_m taške
    nauja_a = a;                      %nauja intervalo pradžia
    nauja_b = b;                      %nauja intervalo pabaiga
    L = b - a;                        %randame intervalo ilgį
    L1 = L;                           %išsaugome intervalo ilgį

    while fx_m2 >= fx_m1              %ciklas taškams palyginti
        iteration = iteration + 1;    %didiname iteracijų skaičių
        x1 = a + L/4;                 %didiname pradžios tašką ketvirtadaliu
        x2 = b - L/4;                 %mažiname pabaigos tašką ketvirtadaliu
        fx1 = tikslo_funkcija(x1);    %funkcijos reikšmė naujoje intervalo
                                      %pradžioje taške x1
        fx2 = tikslo_funkcija(x2);    %funkcijos reikšmė naujoje intervalo
                                      %pabaigoje taške x2
                                      %jeigu funkcijos reikšmė naujoje
                                      %intervalo pradžioje yra mažesne už
                                      %tikslo funkcijos reikšmę buvusiame
                                      %vidurio taške, tuomet
        if(fx1 < fx_m)
            b = x_m;                  %intervalo pabaiga yra lygi buvusiam
                                      %vidurio taškui
            x_m = x1;                 %buvęs vidurio taškas tampa nauja
                                      %intervalo pradžia
        else                          %jei funkcijos reikšmė naujoje
                                      %intervalo pabaigoje yra mažesnė
                                      %už matematinio modelio reikšmę
                                      %buvusiame vidurio taške, tuomet
            if(fx2 < fx_m)
                a = x_m;  %intervalo pradžia yra lygi buvusiam vidurio taškui
```

```
                x_m = x2; %buvęs vidurio taškas tampa nauja intervalo pabaiga
              else
               a = x1; %intervalo pradžia pasistumia ketvirtadaliu
               b = x2; %intervalo b sumažėja ketvirtadaliu
              end
          end

          fx_m1 = tikslo_funkcija(x_m); %tikslo funkcijos reikšmė naujame
                                        %vidurio taške
          xpoint(iteration) = x_m;      %išsaugomas taškas
          xvalue(iteration) = fx_m1;    %išsaugoma reikšmė
          L = b - a;                    %išsaugomas naujas intervalo ilgis
          if(fx_m2 > fx_m1)             %jeigu tenkinama sąlygą
              L1 = L;                   %perkeliamas taškas
              nauja_a = a;              %perkeliamas taškas
              nauja_b = b;              %perkeliamas taškas
              x_min_m = x_m;            %perkeliamas taškas
              fx_m2 = fx_m1;            %perkeliamas taškas
          end
      end

      f_min = fx_m2;                %rasta tikslo funkcijos minimali reikšmė
  end
```

3) Išlaidų apskaičiavimas ir grafiko braižymas

```
                                  %funkcija minimalioms išlaidoms rasti
function [q, t, x_min_m, f_min, islaidos, iteration, L] = min_islaidos(a, b)
  [x_min_m, f_min, nauja_a, nauja_b, iteration, L, xpoint, xvalue] =
int_dal_metodas(a, b);
  mid_point = (nauja_a+nauja_b)/2; %mid_point - intervalo vidurio taškas,
  %nauja_a - intervalo pradžia, nauja_b - intervalo pabaiga
  mid_point_floor = floor(mid_point);         %vidurio taško apatinė riba
  f_floor = tikslo_funkcija(mid_point_floor); %tikslo funkcijos reikšmė
%taške mid_point_floor
  mid_point_ceil = ceil(mid_point);          %vidurio taško viršutinė riba
  f_ceil = tikslo_funkcija(mid_point_ceil);  %tikslo funkcijos reikšmė taške
%mid_point_ceil

  if(f_floor > f_ceil)                       %lyginame tikslo funkcijos
                                             %reikšmę su apatine reikšme
    islaidos = f_ceil;                       %išsaugoma viršutinė reikšmė
    t = mid_point_ceil                       %išsaugoma viršutinė reikšmė
  else
    islaidos = f_floor;                      %išsaugoma apatinė reikšmė
    t = mid_point_floor;                     %išsaugoma reikšmė
  end
 x = a:1:b;                                  %vektorius nuo intervalo
                                             %pradžios iki pabaigos
  y = tikslo_funkcija(x);                    %tikslo funkcija

                                             %aukso pjūvis ir interpoliacija
  optimization = optimset('Display', 'iter', 'MaxFunEvals', 1000, 'PlotFcns',
@optimplotfval , 'TolX', L);
```

```
  [xfminbnd, yfminbnd] = fminbnd(@tikslo_funkcija, a, b, optimization)

  figure                                %brėžiame grafiką
  plot(x, y, '.-', xpoint, xvalue, '.-'), legend('Kiekviename taške',
'Dalijame intervalą pusiau');
  title('Išlaidų kitimas pagal vežimo periodą t');
  xlabel('Prekių vežimo periodas (t)');
  ylabel('Išlaidos per dieną');

  b = mod(20182746, 7)/2 + 2.5;         %parametras b duotas sąlygoje, įsistatome
                                        %savo stud_nr
  q = t * b;                            %vienos partijos dydis
  f_min                                 %tikslo f-jos minimumas
  x_min_m                               %tikslo f-jos minimumo taškas
  t                                    %optimalus prekių vežimo periodas dienomis
  q                                    %optimali prekių partijos apimtis
  islaidos                             %optimalus sprendinys
  iteration                            %iteracijų skaičius
  L                                    %intervalo ilgis
   xfminbnd                             %funkcijos minimalaus taško koordinatė
                                       %auksinio pjūvio ir kvadratinės
                                       %interpoliacijos metodu
  yfminbnd                             %funkcijos minimumo reikšmė auksinio
                                       %pjūvio ir kvadratinės interpoliacijos
                                       %metodu
end
```

4. **Raskite tikslo funkcijos minimumą (mažiausias prekybos bazės išlaidas) ir minimumo tašką (prekių vežimo intervalą). Rastas optimalus prekių vežimo periodas bus intervalas. Paimkite jo vidurinę reikšmę ir suapvalinkite ją iki sveikojo skaičiaus į vieną ir kitą pusę. Iš gautų dviejų reikšmių išrinkite optimalų sprendinį (kad tikslo funkcijos reikšmė būtų mažesnė). Taip pat apskaičiuokite optimalią prekių partijos apimtį.**

- Tikslo funkcijos minimumas
  f_min = 100.4667
- Tikslo funkcijos minimumo taškas
  x_min_m = 2.8125
- Rastas optimalus prekių vežimo periodas dienomis
  t = 3
- Optimalus sprendinys (kad tikslo funkcijos reikšmė būtų mažesnė).
  islaidos = 100
- Optimali prekių partijos apimtis
  q = 12

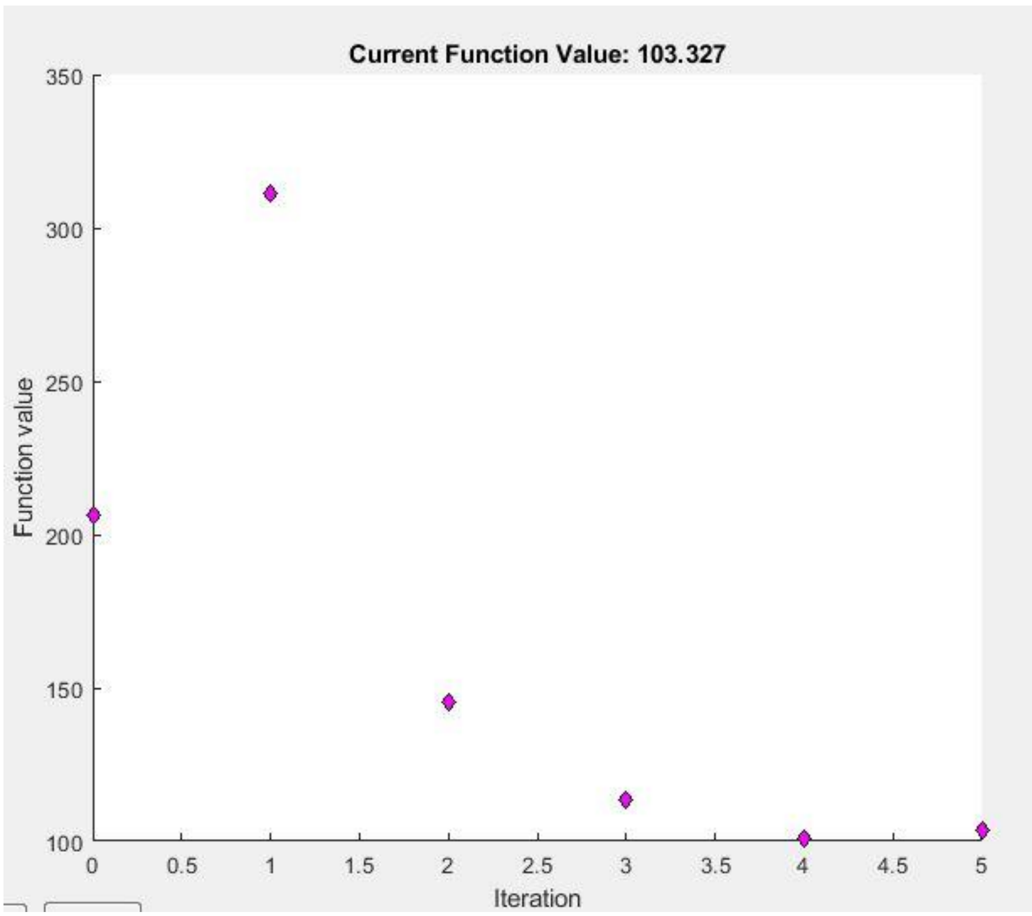
5. **Nustatykite, kiek iteracijų reikia atlikti, norint gauti rezultatą norimu tikslumu (tikslumas nurodomas intervalo ilgiu). Čia viena iteracija –funkcijos vienos reikšmės apskaičiavimas.**

iteration = 5

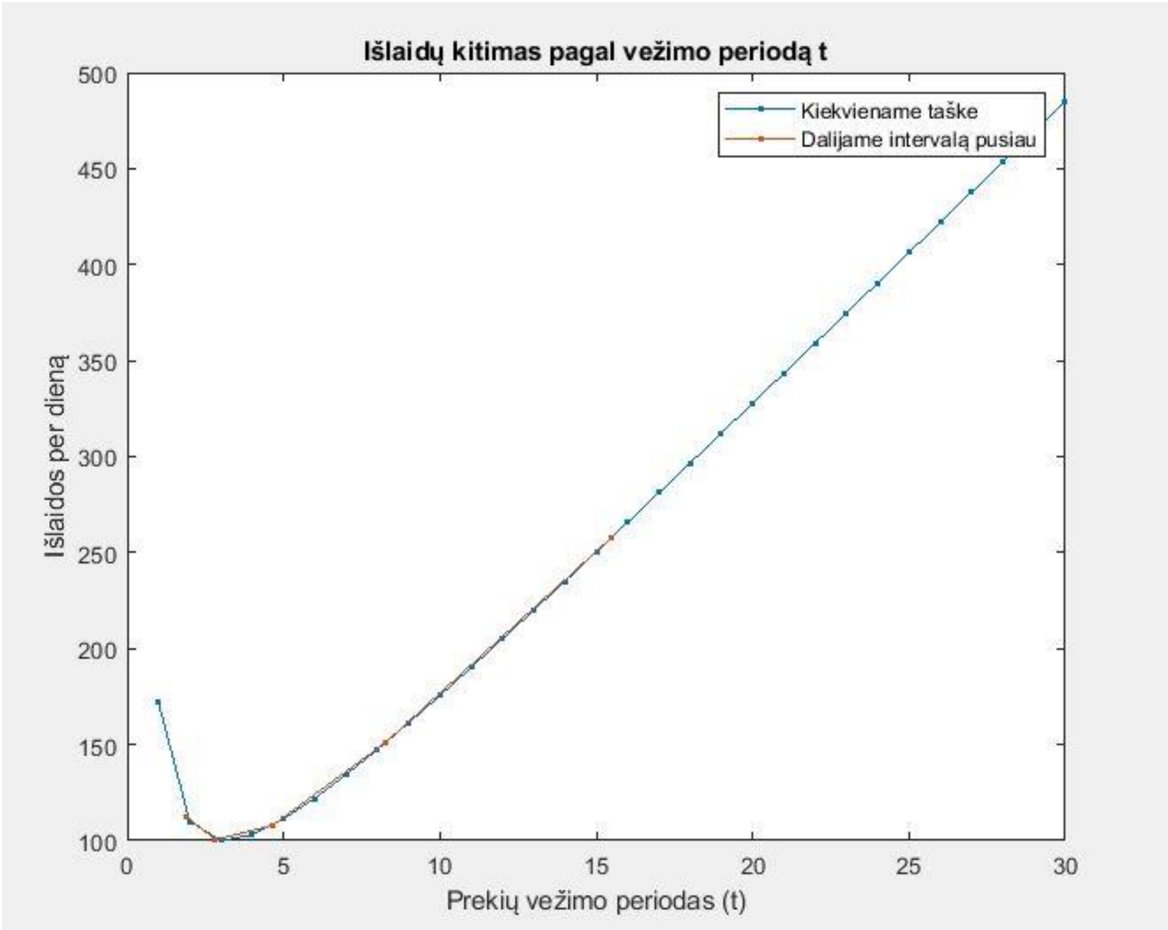
L = 3.6250

6. Nubraižykite tikslo funkcijos grafiką, nurodykite jo pavadinimą, x ir y ašių pavadinimus.

P. S.Turi būti suprogramuotos trys funkcijos:
•pirmoje apskaičiuojama tikslo funkcijos reikšmė;
•antroje įgyvendintas intervalo dalijimo pusiau metodas;
•trečioje kreipiamasi į antrąją funkciją esant įvairioms tikslumo (intervalo ilgio) reikšmėmis; taip pat papildomi veiksmai, nurodyti 4 žingsnyje; grafiko braižymas.



Pav. 1 Auksinio pjūvio ir kvadratinės interpoliacijos metodas



Pav. 2 Intervalo dalijimo pusiau metodas

7. **Raskite funkcijos minimumą auksinio pjūvio metodu ir kvadratine interpoliacija naudojant Matlab funkciją fminbnd. Nustatykite, kiek kartų reikia skaičiuoti funkcijos reikšmes, kad gautume tą patį tikslumą, kaip ir intervalo dalijimo pusiau metodu. P. S.Turi būti papildyta trečia funkcija kreipiniu į funkciją fminbnd.**

Funkcijos minimalaus taško koordinatė auksinio pjūvio ir kvadratinės interpoliacijos metodu:

xfminbnd = 3.6149

Funkcijos minimumo reikšmė auksinio pjūvio ir kvadratinės interpoliacijos metodu:

yfminbnd = 100.9932

Func-count	x	f(x)	Procedure
1	12.077	206.149	initial
2	18.923	311.012	golden

3	7.84597	145.418	golden
4	5.23104	113.519	golden
5	3.61493	100.993	golden
6	2.40659	103.327	golden

Optimization terminated:

the current x satisfies the termination criteria using OPTIONS.TolX of 3.625000e+00