

Vilniaus universitetas

Matematikos ir informatikos fakultetas

Informatikos katedra

Programų sistemų studijų programa

Optimizavimo metodai

**Pirmojo laboratorinio darbo ataskaita**

Ataskaitą tikrino: Prof. Dr. Pranas Katauskis

Ataskaitą parengė: Mantas Jakaitis

Vilnius

**Įvadas**

**Laboratorinio darbo formulavimas:** Suprogramuoti vienmačio optimizavimo intervalo dalijimo pusiau, auksinio pjūvio ir Niutodo metodo algoritmus. Minimizuoti tikslo funkciją f(x) = (x2-4)2/9-1 intervalo dalijimo pusiau ir auksinio pjūvio metodais [0,10] intervale iki tikslumo 10-4 bei Niutono metodu nuo x0=5, kol žingsnsio ilgis didesnis už 10-4.

**Laboratorinio darbo tikslas:** Naudojantis intervalo dalijimo pusiau, auksinio pjūvio ir Niutono metodais surasti funkcijos minimumą, funkcijos reikšmę minimumo taške, surasti iteracijų skaičių bei skaičiuotų funkcijų kiekį, palyginti gautus rezultatus ir įvertinti, kuris metodas yra efektyviausias sprendžiant tokio tipo uždavinį.

**Darbo eiga**

Pirmąjame laboratoriniame darbe iš viso buvo suprogramuotos trys funkcijos, kurios skirtingais metodais minimizavo tikslo funkciją f(x) = (x2 - 4)2/9-1. Naudojantis intervalo dalijimo pusiau, auksinio pjūvio ir Niutono metodais visais metodais buvo surastas funkcijos minimumas, funkcijos reikšmė minimumo taške, iteracijų skaičius bei skaičiuotų funkcijų kiekis beieškant rezultato.

Pasirinkta programavimo kalba: Python

Chart, line chart

Description automatically generated

pav. 1 Tikslo funkcijos grafikas

1. **Intervalo dalijimo pusiau metodas**Intervalo dalijimo pusiau metodo metu pradiniame intervale pasirenkami trys tolygiai pasiskirstę bandymo taškai ir kiekvienos iteracijos metu yra atmetama pusė intervalo. Po kiekvienos iteracijos yra tikrinama ar pasiektas atitinkamas tikslumas, jei pasiekėme – baigiame skaičiavimus, jei ne – kartojame procesą, tol kol pasieksime.

Chart, line chart

Description automatically generated

pav. 2 Intervalo dalijimo pusiau metodo grafiko ir intervalų taškų vaizdavimas

Grafike galima matyti kaip kiekvienos iteracijos metu intervalas yra dalijamas pusiau ir intervalai siaurėja. Šiame grafike pavaizduoti taškai tik po 1, 2, 4 ir 17 iteracijų, kad grafike aiškiai matytųsi sprendinių artėjimas link minimumo.

Grafike vaizduojamas intervalo siaurėjimas kiekvienos iteracijos metu.

Visi taškai, po kiekvienos iteracijos[[1]](#footnote-1):

[[0, 5.0], [1.25, 3.75], [1.25, 2.5], [1.5625, 2.1875], [1.875, 2.1875], [1.953125, 2.109375], [1.953125, 2.03125], [1.97265625, 2.01171875], [1.9921875, 2.01171875], [1.9970703125, 2.0068359375], [1.9970703125, 2.001953125], [1.998291015625, 2.000732421875], [1.99951171875, 2.000732421875], [1.99981689453125, 2.00042724609375], [1.99981689453125, 2.0001220703125], [1.9998931884765625, 2.0000457763671875]].

1. **Auksinio pjūvio metodas**

Auksinio pjūvio metodo intervale yra du bandymo taškai, kurie yra vienodai nutolę nuo vidurio, kiekvienos iteracijos metu skaičiuojama viena tikslo funkcijos reikšmė ir atmetama intervalo dalis, kurioje minimumas neegzistuoja. Procesas kartojamas, kol pasiekiamas nurodytas tikslumas.

A picture containing sky, light

Description automatically generated

pav. 3 Auksinio pjūvio metodo grafiko ir intervalų taškų vaizdavimas

Grafike vaizduojama kaip po kiekvienos iteracijos taškai artėja sprendinio link. Taškai vaizduojami tik 1, 3, 5 ir 24 iteracijų metu, kad matytųsi aiškus intervalo trumpėjimas ir artėjimas link minimumo.

Grafike vaizduojamas intervalo siaurėjimas kiekvienos iteracijos metu.

Visi taškai, po kiekvienos iteracijos[[2]](#footnote-2):

[[0, 6.1803], [0, 3.8197], [1.459010809, 3.8197], [1.459010809, 2.9179875497137298], [1.459010809, 2.360689191], [1.8034249005725402, 2.360689191], [1.8034249005725402, 2.147830949985423], [1.9349776792667792, 2.147830949985423], [1.9349776792667792, 2.0665273861690228], [1.9349776792667792, 2.0162961546504237], [1.9660388973090699, 2.0162961546504237], [1.9852257208122293, 2.0162961546504237], [1.9852257208122293, 2.0044281810372486], [1.9925604845443798, 2.0044281810372486], [1.9970993900637468, 2.0044281810372486], [1.9970993900637468, 2.0016288027491003], [1.9988294898271712, 2.0016288027491003], [1.9988294898271712, 2.000559549192311], [1.9994903206028738, 2.000559549192311], [1.9994903206028738, 2.0001511359480038], [1.999742732240253, 2.0001511359480038], [1.9998950770078676, 2.0001511359480038], [1.9998950770078676, 2.00005332911464], [1.9999555245650915, 2.00005332911464], [1.9999555245650915, 2.000015970710849], [1.9999786131793864, 2.000015970710849], [1.9999951379837542, 2.000015970710849], [1.9999951379837542, 2.0000080132340807], [1.9999951379837542, 2.0000017012545563], [1.9999976449563024, 2.0000017012545563], [1.9999976449563024, 2.0000001518703123], [1.9999986025222467, 2.0000001518703123], [1.9999986025222467, 1.9999995600658318], [1.99999896827517, 1.9999995600658318], [1.99999896827517, 1.9999993340195528], [1.999999107978552, 1.9999993340195528], [1.999999107978552, 1.9999992476786717], [1.9999991613398067, 1.9999992476786717]]

1. **Niutono metodas**

Niutono metode kairiosiose lygčių sistemos pusėse esančios funkcijos ištiesinamos taške Xi ir, išsprendus tiesinę lygčių sistemą, turėtasis taškas pakeičiamas tiesinės lygčių sistemos sprendiniu Xi+1. Šiai idėjai realizuoti reikia pasinaudoti tikslo funkcijos Teiloro eilutės nariais iki antrosios eilės. Vėliau prilyginę nuliui, gausime iteracinę metodo formulę:

Xi+1 = Xi – f’(Xi)/f’’(Xi).

Chart, line chart

Description automatically generated

pav. 4 Niutono metodo grafiko ir jo sprendinių vaizdavimas

Grafike vaizduojami taškai po 1, 2, 3, 5 ir 7 iteracijų. Iš grafiko galima matyti kaip po kiekvienos iteracijos taškai artėjo sprendinio link.

Visi taškai, po kiekvienos iteracijos:

[3.521135202956932, 2.6302882752121075, 2.1721460724532657, 2.018511496347603, 2.0002468490522487, 1.9999950493854164, 1.9999949999940498]

**Rezultatų palyginimai**

|  |  |  |  |
| --- | --- | --- | --- |
| Rezultatai skaičiuojant funkciją  f(x) = (x2 - 4)2/9-1 | Intervalo dalijimo pusiau metodas | Auksinio pjūvio metodas | Niutono metodas |
| Iteracijų skaičius | 17 | 24 | 7 |
| Minimumo x reikšmė | 2.0000076293945312 | 1.9999951379837542 | 1.999995 |
| Funkcijos reikšmė f(xmin) | -0.9999999998965193 | -0.9999999999579748 | -1.000000 |
| Skaičiuotų funkcijų kiekis | 49 | 48 | 14 |

lentelė 1. tikslo funkcijos rezultatų palyginimai

Iš lentelės duomenų akivaizdžiai matosi, jog pagal iteracijų ir skaičiuotų funkcijų kiekį geriausiai pasirodė Niutono metodas, kuriam prireikė tik 7 iteracijų ir 14 apskaičiuotų funkcijų.

Lyginant Intervalo dalijimo pusiau ir auksinio pjūvio metodus, efektyvesnis metodas šiai funkcijai skaičiuoti yra intervalo dalijimo pusiau, jei teigtume, kad iteracijų skaičius nusako efektyvumą. Tačiau, nors auksinio pjūvio metodui reikėjo daugiau iteracijų, kad atrastų sprendinį, jis efektyvesnis iš skaičiuotų funkcijų kiekio – jam per didesnį kiekį iteracijų, prireikė suskaičiuoti mažiau funkcijų. Tai būtų naudingas rodiklis, jei iteracijų kiekis būtų kur kas didesnis nei dabartinis ir galėtume pamatyti kaip intervalo dalijimo pusiau metodui reikia apskaičiuoti daugiau funkcijų kiekvienos iteracijos metu, kad galėtų surasti teisingą atsakymą.

Visų metodų apskaičiavimai funkcijos minimumo taške yra panašūs, tačiau tiksliausiai jį sugebėjo apskaičiuoti intervalo dalijimo pusiau metodas, o Niutono metodas pateikė mažiausiai tikslią reikšmę.

**Išvados**

Apibendrinant galima būtų teigti, jog visi metodai tinkamai atliko savo darbą – surado funkcijos minimumą. Vis dėlto, iš 1 lentelės duomenų galima spręsti, jog tikslo funkcijos f(x) = (x2 - 4)2/9-1 minimumui apskaičiuoti efektyviausias būtų Niutono metodas, kuriam prireikė tik 7 iteracijų ir 14 apskaičiuotų funkcijų. Intervalo dalijimo pusiau ir auksinio pjūvio metodai pasirodė apylygiai, tačiau intervalo dalijimo pusiau metodui prireikė mažiau iteracijų, kad pasiektų tikslą, o auksinio pjūvio metodui prireikė mažiau funkcijų skaičiavimų.

1. Taškai vaizduojami formatu [XL, XR], kur L yra kairioji intervalo pusė, o R – dešinioji. [↑](#footnote-ref-1)
2. Taškai vaizduojami formatu [XL, XR], kur L yra kairioji intervalo pusė, o R – dešinioji. [↑](#footnote-ref-2)