KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS

Aurimas Gajauskas

MATLAB/SIMULINK PROGRAMŲ TIPINIŲ OPTIMIZAVIMO METODŲ TYRIMUI SUKŪRIMAS

Baigiamasis magistro projektas

Vadovas

Doc. dr. Tomas Tekorius

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS

AUTOMATIKOS KATEDRA

MATLAB/SIMULINK PROGRAMŲ TIPINIŲ OPTIMIZAVIMO METODŲ TYRIMUI SUKŪRIMAS

Baigiamasis magistro projektas

Valdymo technologijos (621H66001)

Vadovas
(parašas) Doc. dr. Tomas Tekorius
(data)

Recenzentas
(parašas) Prof. dr. Vytautas Galvanauskas
(data)

Projektą atliko
(parašas) Aurimas Gajauskas
(data)

KAUNAS, 2015

Gajauskas, A. MATLAB/Simulink programų tipinių optimizavimo metodų tyrimui sukūrimas. *Valdymo technologijos* magistro baigiamasis darbas / vadovas doc. dr. Tomas Tekorius; Kauno technologijos universitetas, Elektros ir elektronikos fakultetas, Automatikos katedra.

Kaunas, 2015. 58 psl.

SANTRAUKA

Darbe nagrinėjami tipiniai optimizavimo metodų algoritmai, naudojami valdymo sistemų technologinių parametrų optimalių verčių radimui. Pasirinkti algoritmai suskirstyti į dvi pagrindines grupes: vienmačiai ir daugiamačiai optimizavimo metodai. Algoritmai realizuoti MATLAB/Simulink aplinkoje. Naudojant algoritmus galima rasti matematinės funkcijos ekstremumus arba reguliatoriaus parametrus, su kuriais gaunama geriausia pasirinkto kriterijaus (pvz.: minimali integruota kvadratinė paklaida (ISE), minimali integruota absoliučioji paklaida su laiko svoriniu koeficientu (ITAE) ir kt.) vertė.

Atrinkti ir realizuoti algoritmai skirti supažindinti studentus su algoritmų veikimu. MATLAB/Simulink programoje gausu komentarų, palengvinančių algoritmų įsisavinimą. Atrinkti optimizavimo metodai yra šie: Nuoseklioji peržiūra (vieno ir dviejų kintamųjų tikslo funkcijai), Dichotomijos metodas, Auksinio pjūvio metodas, Chemotaxis metodas, Gauso-Zaidelio metodas, Evoliucinis programavimas, Imituojamo grūdinimo, Gradientinis greičiausio nusileidimo metodas.

Reikšminiai žodžiai: Optimizavimo metodai; MATLAB/Simulink; PID reguliatorius; Evoliucinis programavimas; Imituojamo grūdinimo metodas.

Gajauskas, Aurimas. Development of MATLAB/Simulink Programs for Investingation of Typical Optimization Methods. : Master's Work in *Control Technologies* / supervisor doc. dr. Tomas Tekorius; Kaunas University of Technology, Faculty of Electrical and Electronics Engineering, department of Automation.

Kaunas, 2015. 58 p.

SUMMARY

The typical optimization methods are analysed in work, which are used for optimization of technological parameters in the control systems. Selected algorithms are classified in two main groups: one-dimensional and multidimensional optimization methods. The algorithms are created in MATLAB/Simulink environment using main functions of the program. Created algorithms enable to find extreme values of mathematical functions and best parameters (for example: integral minimal quadratic error (ISE), integral time-weighted absolute error (ITAE)) of controller as well.

Optimization algorithms are designed in order to introduce students with the structure and working principle of the methods. There are many comments in MATLAB/Simulink programs that help to understand programs easily. Selected methods of optimization: Sequential search, Dichotomous method, Golden section search method, Chemotaxis method, Gaus-Zaidel method, Evolutionary programming, Simulated annealing method, Gradient steepest descend method.

Keywords: Optimization methods; MATLAB/Simulink; PID controller; Evolutionary programming; Simulated annealing optimization method.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

ĮVADAS

Nuo automatinio valdymo sistemų kūrimo pradžios pagrindinis automatinio valdymo uždavinys lieka tas pats – parinkti tokius reguliatoriaus parametrus, kurie tenkintų uždaros sistemos dinaminiams ir statiniams režimams keliamus reikalavimus. Siekiant užtikrinti geriausią valdymo sistemos kokybę, stengiamasi surasti tiriamos sistemos ekstremumo tašką t.y. optimizuoti sistemą. Optimizavimo algoritmai gali būti realizuojami skirtingose aplinkose, pvz.: C++, Java, specializuotose programose ir t.t. MATLAB yra programavimo kalba, turinti patogią vartotojo vizualinę sąsają. Programa MATLAB galima atlikti skaitmeninius skaičiavimus, analizuoti duomenis ir juos pavaizduoti, programuoti ir kurti algoritmus. Dėl plačių galimybių ši programa dažnai naudojama akademinėje bendruomenėje. MATLAB programos priedas Simulink sudarytas iš blokinių schemų, skirtų modeliuoti dinaminėms sistemoms. MATLAB programa turi priedą, skirtą spręsti optimizavimo uždaviniams: „*MATLAB optimization toolbox“*. Su optimizavimo metodais nesusipažinęs vartotojas nemato, kokiu būdu yra atliekamas optimizavimas šiame MATLAB priede.

Darbo metu norima sukurti pagrindinių optimizavimo metodų algoritmus MATLAB/ Simulink aplinkoje, kad vartotojas turėtų galimybę su jais susipažinti. Darbo tikslas: sukurti MATLAB/Simulink tipinių optimizavimo metodų paketą, skirtą mokymosi tikslams. Šiam tikslui pasiekti keliami uždaviniai: apžvelgti literatūrą apie optimizavimo metodus, taikomus technologinių procesų modeliavimui ir valdymui; atrinkti dažniausiai naudojamus optimizavimo metodus ir jų algoritmus; pasirinktus algoritmus realizuoti MATLAB/Simulink aplinkoje; sukurtus algoritmus pritaikyti matematinių funkcijų analizei ir reguliatorių parametrams nustatyti.

1. LITERATŪROS APŽVALGA

1.1 OPTIMIZAVIMO UŽDAVINYS

N-matėje Euklido erdvės poaibyje galime kalbėti apie parametrų vektoriaus su minimalia kriterijaus verte paiešką. Parametrų vektoriaus $X \in R^n$ komponentės x_i, i=1,...,n vadinamos *uždavinio kintamaisiais*, o kriterijų apibrėžianti funkcija *f(x) – tikslo funkcija* [1]. Erdvės R^n poaibis, kuriame tenkinamos uždavinyje apibrėžtos sąlygos, vadinamas *leistinąja sritimi* $A \in [a, b]$. Minimizavimo (analogiškai ir maksimizavimo) uždavinys užrašomas taip: min{ *f(x)*: x∈A} (arba max{*f(x)*: x∈A})

Šio optimizavimo uždavinio sprendimas reiškia minimalios (arba maksimalios) tikslo funkcijos vertės $f_* \le f(X),\ X \in A$ ir minimumo (ar maksimumo) taško $X_* \in A, f(X_*) = f_*$ suradimą [1]. Sprendimo metodus pakanka suformuluoti tik vienam iš dviejų uždavinių (maksimumo arba minimumo radimui), nes uždavinių max {*f(x)*: x∈A} ir min{- *f(x)*: x∈A} sprendinių aibės vienodos [2].

Optimizavimas gali būti atliekamas šiais metodais [3]:

1. *Analitinis metodas*. Šis metodas paremtas klasikiniais diferencialiniais skaičiavimais. Diferencialiniame skaičiavime minimumo (maksimumo) radimas atliekamas ieškant funkcijos išvestinės. Po to randama kintamojo reikšmė, su kuria funkcijos išvestinė lygi nuliui.
2. *Grafinis metodas.* Šiuo metodu grafiškai atvaizduojama tikslo funkcija ir nepriklausomi kintamieji. Iš grafiko randamos minimalios (maksimalios) vertės.
3. *Eksperimentinis metodas.* Šiuo metodu tiesiogiai matuojama funkcijos reikšmė keičiant kintojo reikšmes. Šiuo metodu rasti ekstremumai gali būti netikslūs dėl įvairių matavimo paklaidų atsiradimo.
4. *Skaitinis metodas.* Šiame metode naudojami iteraciniai skaičiavimai.

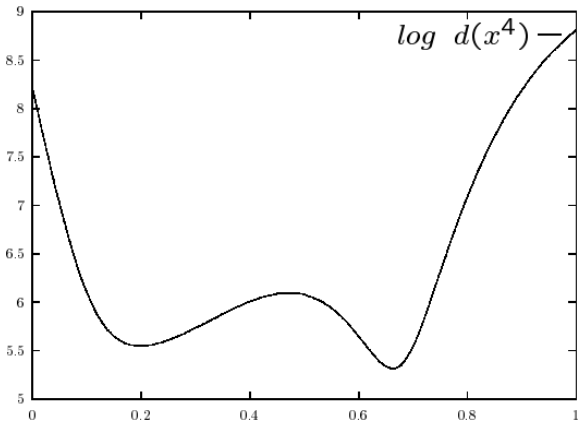
Pagrindiniai optimizavimo uždaviniai skirstomi pagal tikslo funkcijos ir apribojimų tipus [4]:

1. *Iškiliojo ir neiškiliojo matematinio programavimo uždaviniai.* Juose lokalinis ekstremumas tuo pat metu yra ir globalinis ekstremumas.
2. *Netiesinio matematinio programavimo uždaviniai.* Tikslo funkcija arba apribojimų funkcijos yra netiesinės.
3. *Tiesinio programavimo uždaviniai*. Tikslo funkcija tiesinė ir visi apribojimai aprašomi tiesinėmis funkcijomis.
4. *Kvadratinio programavimo uždaviniai.* Juose apribojimais aprašomi tiesinėmis funkcijomis, o tikslo funkcija turi kvadratinę formą.
5. *Trupmeninio programavimo uždaviniai.* Juose tikslo funkcija trupmeninė.

6. *Diskrečiojo programavimo uždaviniai.* Juose galimų sprendinių sritį sudaro diskrečios reikšmės.
7. *Sveikaskaičio programavimo uždaviniai.* Juose leistinuoju ir optimaliu sprendiniu gali būti tik sveikųjų skaičių rinkinys.
8. *Parametrinio programavimo uždaviniai.* Juose tikslo funkcija, apribojimai ir t.t. priklauso nuo tam tikrų parametrų.
9. *Geometrinio programavimo uždaviniai.* Juose tikslo funkcija ir apribojimai išreikšti polinomais.
10. *Dinaminio programavimo uždaviniai.*
11. *Stochastinio programavimo uždaviniai.* Juose visi ar dalis parametrų yra atsitiktinai arba neapibrėžti, arba iš dalies neapibrėžti.
12. *Daugiakriteriniai optimizavimo uždaviniai.*

1.2 LOKALIEJI IR GLOBALIEJI EKSTREMUMAI

Taškas x_* vadinamas funkcijos $f(x)$ lokaliojo minimumo tašku, jei egzistuoja tokia x_* aplinka $S(x_*)$, kad $f(x_*) \leq f(x),\ x \in S(x_*)$ [1]. Taškas x_*, vadinamas funkcijos $f(x)$ globaliojo minimumo tašku, jei $f(x_*) \leq f(x),\ x \in R^n$ [1].



1 pav. Lokalusis ir globalusis minimumai [5]

1 paveikslėlyje lokalus minimumas yra, kai x=0,2, o globalus minimumas, kai x=0,66. Jeigu funkcija yra griežtai iškila, jos lokalus minimumas sutampa su globaliu minimumu ir tokia funkcija vadinama unimodine funkcija. Jeigu funkcija turi ir lokaliųjų ir globaliųjų ekstremumų, ji vadinama multimodine.

1.3 VIENMAČIŲ OPTIMIZAVIMO METODŲ APŽVALGA

NUOSEKLIOJI PERŽIŪRA

Nuosekliosios peržiūros optimizavimo metodas yra pats paprasčiausias optimizavimo metodas. Šis metodas dažnai naudojamas, kai iš anksto nežinome funkcijos kitimo pobūdžio. Ekstremumo tašką (minimumą arba maksimumą) galime rasti, peržiūrint funkcijos vertes pasirinktu tikslumu *e* intervalo *[a, b]* taškuose.

DICHOTOMIJOS OPTIMIZAVIMO METODAS

Leistinosios srities sumažinimo metodas vadinamas dichotomijos metodu. Dichotomijos metodu ieškomas funkcijos ekstremumo taškas intervale [a, b]. Šis intervalas vadinamas leistinąją sritimi. Tikslo funkcijos f(x) ekstremumo taškas x^* ieškomas palaipsniui mažinant tikslo funkcijos leistinąją sritį, kol gaunamas pakankamai mažas x kintamojo kitimo diapazonas [3]. Leistinoji sritis gali būti N-matė. Klasikinis ir plačiausiai naudojamas yra vienmatis dichotomijos paieškos algoritmas. Literatūroje yra aprašyta ir daugiamačių (N^2 ir N^3) [6] paieškos algoritmų, tačiau jie dėl sudėtingumo nėra plačiai taikomi. Šiek tiek modifikuotas dichotomijos algoritmas (vienmatėje leistinojoje erdvėje) pritaikytas signalų periodinės funkcijos dažnio skaičiavimui [7] .

AUKSINIO PJŪVIO OPTIMIZAVIMO METODAS

Auksinio pjūvio optimizavimo metodas yra taikomas, kai yra žinomas sprendinio intervalas [a, b] ir tikslo funkcija f(x). Šis metodas yra pagrįstas ieškomo sprendinio intervalo mažinimu. Šiuo metodu yra sugeneruojami segmentai{I_1, I_2, I_3,...} naudojant formulę [3]:

$$I_k = I_{k+1} + I_{k+2} \tag{1.1}$$

Taisyklė, pagal kurią segmentų intervalų ilgis yra generuojamas, yra tokia, kad dviejų gretimų intervalų santykis K yra pastovus:

$$\frac{I_k}{I_{k+1}} = \frac{I_{k+1}}{I_{k+2}} = \frac{I_{k+2}}{I_{k+3}} = \ldots = K \tag{1.2}$$

Sudarius intervalų santykį, jis yra lygus pastoviam dydžiui K, pakeltam tam tikru laipsniu, pvz.:

$$\frac{\mathrm{I_k}}{\mathrm{I_{(k+3)}}} = \mathrm{K}^3 \tag{1.3}$$

Jei padaliname (1.1) formulę iš I_{k+2} gauname:

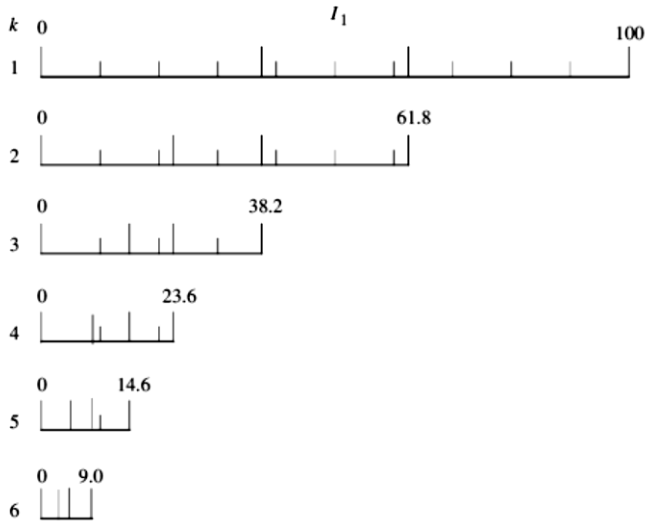
$$\frac{I_k}{I_{k+2}} = \frac{I_{k+1}}{I_{k+2}} + 1 \tag{1.4}$$

Pasinaudojus (1.3) formulę, iš (1.4) formulės gauname, kad K dydžiai yra susieti sąryšiu:

$$K^2 = K + 1 \tag{1.5}$$

Išsprendę šią lygtį gavome, kad

$$K = \frac{1 \pm \sqrt{5}}{2} \tag{1.6}$$



2 pav. Auksinio pjūvio paieška [3]

Neigiamas K ženklo nenaudosime, o didumas yra K=1,618034. Ši konstanta yra žinoma kaip auksinis santykis. Šio metodo terminai kilo iš klasikinės graikų kultūros, kurioje stačiakampis, kurio kraštinių ilgių santykis atitinkamai 1:K buvo laikomas tobuliausiu ir todėl jis pradėtas vadinti auksiniu stačiakampiu [3]. Vėliau, seka $\left\{I_1, \frac{I_1}{K}, \frac{I_1}{K^2}, ..., \frac{I_1}{K^{n-1}}\right\}$ buvo pavadinta auksinio pjūvio seka.

Auksinio pjūvio metodas pasižymi šiomis savybėmis:

1. Intervalų ilgis nepriklauso nuo intervalų skaičiaus n, todėl skaičiavimai gali būti atliekami iki tam tikro tikslumo ε.
2. Santykis tarp dviejų gretimų intervalų $\frac{F_{n-k-1}}{F_{n-k}}$ pakeičiamas santykiu $\frac{1}{K}$.

Ausinio pjūvio metodas pritaikytas apskaičiuoti celiuliozės pirolizės proceso aktyvacijos energijas [8], bei parametrų optimaliam nustatymui sprendžiant diferencialines lygtis [9]. Šis optimizacijos metodas laikomas vienu efektyviausiu metodu, neatliekančiu diferencialinių skaičiavimų.

1.4 DAUGIAMAČIŲ OPTIMIZAVIMO METODŲ APŽVALGA

GAUSO-ZAIDELIO OPTIMIZAVIMO METODAS

Gauso-Zaidelio metodas yra patobulintas Jacobi optimizacijos metodas. Šio metodo kiekvienoje iteracijoje ieškoma tikslo funkcijos reikšmės vieno iš kintamųjų atžvilgiu, skirtingai negu Jcobi metode, kitiems esant fiksuotiems [10]. Metodo veikimas paprastas [11]: iš pradinio taško judama pagal pirmąją koordinatę x_1 kol pasiekiamas vienmatis minimumas pagal x_1, kitus kintamuosius laikant pastoviais. Iš gautojo taško judama pagal antrąją koordinatę kol pasiekiamas minimumas pagal x_2 ir taip atliekama su visais kintamaisiais. Gauso-Zaidelio metodo trūkumas yra tai, kad kintamieji gali priklausyti vienas nuo kito ir iteracijų metu turi būti įvertinami vienu metu, o ne atskirai po vieną (t.y. vienų kintamųjų negalima laikyti konstantomis kitų atžvilgiu) [10]. Gauso-Zaidelio metodas yra lėtesnis negu Jacobi metodas, tačiau dažniau konverguoja esant dideliam duomenų kiekiui, todėl yra patikimesnis [10].

GRADIENTINIS GREIČIAUSIO NUSILEIDIMO OPTIMIZAVIMO METODAS

Gradiento metodai – tai pirmos eilės netiesinio programavimo uždavinių metodai, kuriuos taikant naudojamas tikslo funkcijos *f(x)* gradientas [1]. Šiuo metodu surandamas tikslo funkcijos lokalus ekstremumas. Greičiausio nusileidimo optimizavimo metodas N-dimensinės funkcijos minimumo ieško neigiamo gradiento kryptimi. Kiekvienos iteracijos metu žengiamas žingsnis iki funkcijos minimumo antigradiento kryptimi pagal formulę [1]:

$$x_{k+1} = x_k + a \cdot S_k$$

S_k – žingsnio krypties vektorius; a– žingsnio daugiklis (jeigu $\|S_k\| = 1$, tai a yra žingsnio ilgis).

Žingsnio daugiklis a yra kintamas, krypties vektorius yra tikslo funkcijos antigradientas $S_k = -\nabla f(x_k)$. Žingsnio dydis yra priderinamas taip, kad funkcijos reikšmė būtų minimizuojama vienos dimensijos linijos kryptimi.

IMITUOJAMO GRŪDINIMO OPTIMIZAVIMO METODAS

Imituojamo grūdinimo metodas (*angl. k. simulated annealing*) yra atsitiktinės paieškos optimizavimo metodas, skirtas spręsti optimizavimo uždaviniams, surandant globalų ekstremumo tašką. Alternatyva tokiai paieškai yra pagrįsta analogija su metalo atkaitinimo ir šaldymo procesu. Iš pradžių metalas yra įkaitinamas iki lydimosi temperatūros ir po to lėtai atšaldomas. Aukštoje temperatūroje atomai dažnai juda ir jų judėjimo trajektorijos atsitiktinės ir ilgos. Temperatūrai

mažėjant atomų judėjimas tampa mažiau chaotiškas, atomų kinetinė energija mažėja ir tam tikrame minimaliame energijos būvyje sudaroma termodinaminė pusiausvyra su aplinka. Pagal analogija su termodinaminiu procesu, šiame optimizavimo metode energija *E* yra laikoma tikslo funkcijos vertė *f(x)*, temperatūra *T* metodo parametru *t*, o atomų judėjimo trajektorijos – kintamuoju x [12]. Šio metodo technika leidžia išvengti lokalaus minimumo radimo [8]. Lokalaus minimumo galima išvengti, nes tikimybė p(T) pajudėti iš taško x_i į tašką x_j, kai tikslo funkcijos reikšmės yra $f(x_i)$ ir $f(x_j)$ $\left(f(x_j) > f(x_i)\right)$ nėra lygi nuliui. Dažniausiai ši tikimybė aprašoma Metropolio kriterijumi [13]:

$$p(T) = \begin{cases} \exp\left[-\dfrac{f(x_i) - f(x_j)}{T}\right]; f(x_j) > f(x_i) \\ 1; f(x_j) < f(x_i) \end{cases} \quad (1.7)$$

Čia T– temperatūra. Kaip matoma iš (1.7) formulės, tikimybė pabėgti iš lokalaus minimumo taško priklauso nuo temperatūros. Mažesnėje temperatūroje ši tikimybė mažėja.

Pagal analogija su atomų chaotiniu judėjimu metale, atkaitinimą imituojančiame metode nepriklausomo kintamojo reikšmės yra generuojamos atsitiktinai ir taip analizuojama pasirinkta tikslo funkcijos erdvė. Judėjimo žingsnis Δx, kuris yra pradinės kintamojo reikšmės x_i ir galinės reikšmės $x_j = x_i + \Delta x$ skirtumas, sugeneruojamas atsitiktinai, panaudojant tikimybinę tankio pasiskirstymo funkciją. Dažniausiai ši funkcija yra Gauso:

$$g(\Delta x, T) \propto \exp\left(-\frac{(\Delta x)^2}{T}\right) \quad (1.8)$$

Visos sistemos temperatūra kontroliuojama pagal sudarytą grafiką. Kai sistema pasiekia šiluminę pusiausvyrą temperatūroje T_k, sistema atšaldoma iki temperatūros T_{k+1} pagal formulę [14]:

$$T_{k+1} = \alpha T_k \quad (1.9)$$

čia α– atšaldymo parametras, kuris būna 0,5-0,99 [15]. Labiau priimtina sistemą šaldyti iš lėto (t.y. $\alpha \to 1$), nes greitas šaldymas gali surasti lokalųjį ekstremumą.

Sistemos pradinė temperatūra T_0 turi būti parinka nei per didelė nei per maža. Jeigu T_0 yra per maža, algoritmas gali surasti lokalųjį minimumą, o kai T_0 per didelė atliekami bereikalingi skaičiavimai [13].

Sistema laikoma pasiekusia šiluminę pusiausvyrą, kai palankių žingsnių skaičius pasiekia tam tikrą numatytą reikšmę arba kai visų žingsnių skaičius pasiekia tam tikrą skaičių tam tikroje temperatūroje Tk. Algoritmo veikimo metu palankių ir visų žingsnių santykis turėtų būti 0,5 [16]. Šis santykis gali būti pasiektas panaudojant kintamą pradinę temperatūrą: $T' = T \cdot \lambda$. Kai $N_{pal}/N<0,4$ λ didėja, o kai $N_{pal}/N>0,6$ λ mažėja [15].

CHEMOTAXIS OPTIMIZAVIMO METODAS

Chemotaxis optimizavimo algoritmas paremtas analogija su bakterijos reakcija į cheminį dirgiklį. Paprastos sandaros vienaląsčių ar daugialąsčių organizmų reakcija į aplinką yra pagrįsta informacijos rinkimu jų aplinkoje, taip įtakojant jų efektyvų atsaką į pasikeitusias aplinkos sąlygas. Optimizavimo algoritmas, taip pat gali būti vertinamas kaip subjektas, kuris renka informaciją, kad galėtų pasiekti optimalų tašką.

Mokslininkai modeliavo bakterijos judėjimą 2-D koordinatėse, naudodamiesi šiais teiginiais [17]:

1) Bakterijos trajektorija susideda iš tiesių, nuosekliai sujungtų linijų.
2) Visose trajektorijose bakterija juda vienodu ir pastoviu greičiu.
3) Kai bakterija pakeičia savo judėjimo kelią, naujos krypties pasirinkimas (tiek pasisukimo kampas, tiek judėjimo laikas) yra visiškai apibrėžiamas tikimybiniu pasiskirstymu.
4) Judėjimo laiko ir pasisukimo kampo tikimybinis pasiskirstymas nėra priklausomas nuo prieš tai buvusios trajektorijos parametrų.

Taigi, bakterijos judėjimą pilnai apibūdina jos greitis, judėjimo trukmė ir posūkio kampas. *Chemotaxis* algoritmas yra nesudėtingas, paieškos metu tikslo funkcijos reikšmės parenkamos tik mažėjimo tvarka, dėl to algoritmas greičiau konverguoja, bet jam sunkiau išeiti iš lokalaus minimumo taško [18].

EVOLIUCINIO PROGRAMAVIMO METODAS

Evoliucinis algoritmas priklauso stochastiniams optimizavimo metodams, kurie modeliuoja natūralios evoliucijos procesus. Bendrai evoliuciniai algoritmai (EA) yra charakterizuojami 3 faktais [19]:

1. Nekintančiu populiacijos dydžiu
2. iš kurio atrenkami individai ir
3. jie modifikuojami genetiniais operatoriais, dažniausiai kryžminimu ir mutacijos operatoriais.

Analogiškai kaip ir natūralioje evoliucijoje pasirinkimų nariai yra vadinami individais, o individų rinkinys yra vadinamas populiacija. Kiekvienas individas atitinka atsakymą, t.y. galimą sprendimo vektorių tam tikrai problemai spręsti. Vis dėlto, individas nėra sprendimo vektorius, bet jame yra užšifruota tam tikra struktūra. Šį vektorių gali sudaryti pavyzdžiui bitų vektorius, realių reikšmių vektorius ir kitos struktūros vektoriai. Visų galimų vektorių rinkinys sudaro individų visumą I. Populiacija yra rinkinys vektorių i ϵ I.

Individų atrankos procesas, tolesnei regeneracijai, gali būti stochastinis arba visiškai determinuotas. Atrankos metu prasčiausi individai yra pašalinami iš populiacijos, o geriausi individai yra perdirbami. Šio metodo tikslas yra kiekvienos iteracijos metu padidinti bendrą populiacijos kokybę.

2. METODOLOGINĖ DALIS

2.1 MATLAB PROGRAMINĖ ĮRANGA

Kompanijos MathWorks sukurta MATLAB programa daugiausia skirta inžineriniams skaičiavimams. MATLAB turi savo programavimo kalbą ir patogią vartotojo vizualinę sąsają. MATLAB programoje naudojami priedai (*angl. k. toolbox*), kurie suteikia galimybę apdoroti elektrinius signalus, vaizdus, atlikti modeliavimą ir kt. Vienas iš MATLAB programoje galimų priedų yra skirtas atlikti optimizavimo užduotis (*angl. k. optimization toolbox*). Pastarasis optimizavimo priedas suteikia galimybę atlikti netiesinį, daugiakriterinį optimizavimą, taip pat galima atlikti kvadratinį ir tiesinį programavimą ir kt [20]. Šiuo optimizavimo priedu patogu naudotis inžinieriniams skaičiavimams, kai vartotojas jau yra susipažinęs su optimizavimo metodų algoritmais. Tačiau mokymosi tikslais, šis *optimization toolbox* per sudėtingas ir negalima matyti pačių metodų algoritmų ir ekstremumo paieškos kelio.

2.2 VIENMAČIŲ OPTIMIZAVIMO METODŲ ALGORITMAI

2.2.1 NUOSEKLIOSIOS PERŽIŪROS ALGORITMAS

Nuosekliosios peržiūros algoritmo etapai:

1) Nustatomos nepriklausomo kintamojo paieškos intervalo ribos [a, b].
2) Užsiduodamas reikiamas minimumo (arba maksimumo) taško x* nustatymo tikslumas ε.
3) Nustatomi taškai *n*, kuriuose reikės apskaičiuoti tikslo funkcijos vertes f(x), pagal formulę:

$$n = \left[\frac{b-a}{\varepsilon} \right] + 1$$

4) Palyginamos apskaičiuotos funkcijos vertės ir randama mažiausia (arba didžiausia) tikslo funkcijos vertė f(x*).

2.2.2 DICHOTOMIJOS OPTIMIZAVIMO METODO ALGORITMAS

Dichotomijos optimizavimo metodo algoritmo etapai [3]:

1) Nustatomos nepriklausomo kintamojo paieškos intervalo ribos [a, b].
2) Užsiduodamas reikiamas minimumo (arba maksimumo) taško nustatymo tikslumas ε.
3) Imami 3 taškai dalijantys intervalą [a, b] į 4 lygias dalis:

   $x_1 = a + (b+a)*0,25;$

   $x_2 = a + (b+a)*0,5;$

   $x_3 = a+ (b+a)*0,75;$
4) Apskaičiuojamos tikslo funkcijos vertės taškuose f(a), f(x_1), f(x_2), f(x_3), f(b).

5) Iš apskaičiuotų tikslo funkcijos verčių išrenkama mažiausia tikslo funkcijos vertė. Tikrinama sąlyga ir sumažinama leistinoji minimumo sritis:
   - $f(x_1) = \min_i f(x_i)$, tai pereiname į intervalą [a, x_2];
   - $f(x_2) = \min_i f(x_i)$, tai pereiname į intervalą [x_1, x_3];
   - $f(x_3) = \min_i f(x_i)$, tai pereiname į intervalą [x_2, b];
6) Naujai gautame intervale pažymimos naujos intervalo ribos [a, b] ir kartojami skaičiavimai.
7) Procedūra kartojama tol kol intervalas b-a yra nedidesnis nei užduotas tikslumas ε.

2.2.3 AUKSINIO PJŪVIO OPTIMIZAVIMO METODO ALGORITMAS

Auksinio pjūvio optimizavimo metodo algoritmas [3]:

1) Nustatomos nepriklausomo kintamojo paieškos intervalo ribos [a, b].
2) Užsiduodamas reikiamas minimumo (arba maksimumo) taško nustatymo tikslumas ε.
3) Apskaičiuojamas intervalo ilgis I1=b-a, konstanta lygi K=1,618034. Apskaičiuojamas naujas intervalo ilgis: I2=I1/K.
4) Surandami du taškai:

   $x_2 = a + \frac{I_1}{K}$ ir $x_1 = b - x_2$.

5) Apskaičiuojamos tikslo funkcijos vertės f(x_1) ir f(x_2) taškuose x_1 ir x_2.
6) Palyginamos tikslo funkcijos vertės taškuose x_1 ir x_2 ir sumažinama leistinoji sritis:

   Jei f(x_1)<f(x_2), pereiname į intervalą [a, x_2];

   Jei f(x_1)>f(x_2), pereiname į intervalą [x_1, b];

   Jei f(x_1)=f(x_2), pereiname į intervalą [x_1,x_2].
7) Gautą naują intervalą pažymime [a, b] ir atliekame skaičiavimus nuo 3 žingsnio.
8) Procedūra kartojama tol kol intervalas b-a yra nedidesnis nei užduotas tikslumas ε.

2.3DAUGIAMAČIŲ OPTIMIZAVIMO METODŲ ALGORITMAI

2.3.1 GAUSO-ZAIDELIO OPTIMIZAVIMO METODO ALGORITMAS

Gauso-Zaidelio optimizavimo metodo algoritmas:

1) Nustatomos nepriklausomų kintamųjų paieškos intervalo ribos [a_i, b_i].
2) Užsiduodamas reikiamas minimumo taško nustatymo tikslumas ε.
3) Pasirenkamas vienas kintamasis, o kiti laikomi pastoviais. Nustatomi taškai *n*, kuriuose reikės apskaičiuoti tikslo funkcijos vertes, pagal formulę:

$$n = \left[\frac{b_i - a_i}{\varepsilon} \right] + 1$$

4) Apskaičiuojamos tikslo funkcijos vertės nustatytuose n taškuose ir randama minimali tikslo funkcijos vertė. Nepriklausomo kintamojo reikšmė šiame taške užfiksuojama ir paieška atliekama su kitu kintamuoju.
5) Procedūra kartojama, kol tikslo funkcijos pagerėjimas gaunamas didesnis už užduotą tikslumą ε.

2.3.2 CHEMOTAXIS OPTIMIZAVIMO METODO ALGORITMAS

Chemotaxis optimizavimo metodo algoritmas [21]:

1) Užduodamos pirmojo priartėjimo optimizuojamų parametrų vertės $\vec{x}_0$. Jei tokios vertės iš anksto nėra žinomos, užduodamos atsitiktinės vertės. Apibrėžiama leistonoji sritis $\left[\vec{a}\ \vec{b} \right]$.
2) Apskaičiuojama tikslo funkcijos vertė $f(\vec{x}_0)$ pradiniame taške $\vec{x}_0$.
3) Parenkama optimizuojamų parametrų „mutacija“. Tai atliekama, prie parametrų $\vec{x}_0$ pridėjus atsitiktinai parinktus pokyčius. Tam paprastai naudojami Gauso pasiskirstymo atsitiktiniai dydžiai su nuline matematine viltimi ir vienetine dispersija:

$$\vec{x}_1 = \vec{x}_0 + \Delta\vec{x} \cdot RAND$$

$$\text{čia } \vec{x}_1 = \begin{bmatrix} x_{01} + \Delta x_1 \cdot rand_1 \\ x_{02} + \Delta x_2 \cdot rand_2 \\ \vdots \\ x_{0n} + \Delta x_n \cdot rand_n \end{bmatrix},$$

čia *RAND* – atsitiktinis Gauso kintamasis su nuline viltimi. $M\{RAND\} = 0$ *ir* $D\{RAND\} = 1$. Δx_j, $j=1,...,n$ – parametrai, nustatantys „mutacijų“ laipsnį. Skaičiavimų algoritmuose tai yra derinimo parametrai, turintys įtakos algoritmų konvergavimo greičiui.

4) Apskaičiuojama nauja tikslo funkcijos vertė $f(\vec{x})$.
5) Tikrinama, ar naujos parametrų vertės pagerino tikslo funkcijos vertę.
   - jei $f(\vec{x}) < f(\vec{x}_0)$ ir naujų nepriklausomų kintamųjų vertės yra leistinojoje srityje $\left[\vec{a}\ \vec{b} \right]$, tai $\vec{x}_0 = \vec{x}$ kita mutacija atliekama tokiu pačiu būdu (procedūra kartojama nuo 3 žingsnio).
   - jei $f(\vec{x}) > f(\vec{x}_0)$ arba naujos nepriklausomų kintamųjų vertės nėra leistinojoje srityje $\left[\vec{a}\ \vec{b} \right]$, tai grįžtama prie ankstesnių parametrų verčių ($\vec{x}_0 = \vec{x} - \Delta\vec{x}$) ir procesas tęsiamas nuo 3 žingsnio.

6) Procedūra baigiama, kai pasiekiamas užduotas iteracijų skaičius.

2.3.3 IMITUOJAMO GRŪDINIMO OPTIMIZAVIMO METODO ALGORITMAS

Imituojamo grūdinimo optimizavimo metodo algoritmas:

1) Pasirenkamas paieškos pradžios taškas $\vec{x}_0$ ir jame apskaičiuojama tikslo funkcijos vertė $f(\vec{x}_0)$. Apibrėžiama leistonoji sritis $\left[\vec{a}\ \vec{b}\right]$.
2) Generuojama n nepriklausomų atsitiktinių dydžių $y_1,...,y_n$ ir apskaičiuojamos paieškos krypties vektoriaus $\vec{u}$ dedamosios:

$$\vec{u} = \begin{bmatrix} u_1 \\ \vdots \\ \vdots \\ u_n \end{bmatrix}, \text{ čia } \mathbf{u}_i = \frac{y_i}{\sqrt{y_1^2 + ... + y_n^2}}.$$

3) Apskaičiuojama nauja nepriklausomo kintamojo reikšmė: $\vec{x}^* = \vec{x}_0 + \lambda\vec{u}$.
4) Tikrinama ar nepriklausomas kintamasis x_i priklauso leistinajai sričiai $\left[a_i\ b_i\right]$. Jei $x_i > b_i$ arba $x_i < a_i$ tuomet atitinkamai $x_i = b_i$ arba $x_i = a_i$.
5) tikslo funkcijos reikšmė tame taške $f(\vec{x}^*)$.
6) Tikrinama sąlyga, jeigu $f(\vec{x}^*) \le f(\vec{x}_0)$ ir $\left|\frac{f(\vec{x}^*) - f(\vec{x}_0)}{f(\vec{x}_0)}\right| > \varepsilon$ tai nustatoma $\vec{x}_0 = \vec{x}^*$ ir grįžtama į 3 žingsnį. Jeigu iteracijų skaičius $N \ge N_{min}$ ir $\frac{f(\vec{x}^*) - f(\vec{x}_0)}{f(\vec{x}_0)} \le \varepsilon$ tai skaičiavimas baigiamas.
7) Tikrinama sąlyga, jeigu $f(\vec{x}^*) > f(\vec{x}_0)$, tai nustatoma p vertė $p = \frac{1}{e^{\left(\frac{f(x^*) - f(x_0)}{t}\right)}}$ čia t – algoritmo derinimo parametras. Generuojamas normalizuotas atsitiktinis dydis V.
   - jei $V \ge p$, paliekama $\vec{x}_0 = \vec{x}_0$;
   - jei $V < p$, nustatoma $\vec{x}_0 = \vec{x}^*$ ir grįžtama į 3 žingsnį.

2.3.4 EVOLIUCINIO PROGRAMAVIMO ALGORITMAS

Evoliucinio programavimo algoritmas:

1) Parenkama pradinė p vektorių-tėvų populiacija $\vec{x}_1, \vec{x}_2, ..., \vec{x}_p$ kur kiekvienas vektorius turi n nepriklausomų kintamųjų verčių.
2) Apskaičiuojamos tikslo funkcijos vertės: $f(\vec{x}_i),\ i = 1,...,p$.
3) Iš „tėvų“ vektorių $\vec{x}_{i,}\ i = 1,...,p$ sudaromi palikuonių vektoriai $\vec{x}_i$, $i=p+1,...,2p$, prie kiekvieno vektoriaus elementų pridedant Gauso kintamuosius su nuliniu vidurkiu ir normuota dispersija, padaugintus iš parametrų Δx_i, nustatančių mutacijos laipsnį. Tai atliekama, prie parametrų $\vec{x}_i$ pridėjus atsitiktinai parinktus pokyčius. Tam paprastai naudojami Gauso pasiskirstymo atsitiktiniai dydžiai su nuline matematine viltimi ir vienetine dispersija:

$$\vec{x}_i = \vec{x}_i + \Delta\vec{x} \cdot RAND$$

$$\text{čia } \vec{x}_i = \begin{bmatrix} x_{i1} + \Delta x_1 \cdot rand_1 \\ x_{i2} + \Delta x_2 \cdot rand_2 \\ \vdots \\ x_{in} + \Delta x_n \cdot rand_n \end{bmatrix},$$

čia *RAND* – atsitiktinis Gauso kintamasis su nuline viltimi. $M\{RAND\} = 0$ *ir* $D\{RAND\} = 1$. Δx_j, $j=1,...,n$ – parametrai, nustatantys „mutacijų“ laipsnį. Skaičiavimų algoritmuose tai yra derinimo parametrai, turintys įtakos algoritmų konvergavimo greičiui.

4) Tikrinama ar palikuonio vektoriaus x_j nepriklausomas kintamasis x_{ji} priklauso leistinajai sričiai $[a_i\ b_i]$. Jei $x_{ji} > b_i$ arba $x_{ji} < a_i$ tuomet atitinkamai $x_{ji} = b_i$ arba $x_{ji} = a_i$.
5) Apskaičiuojamos tikslo funkcijos vertės vektoriams-palikuoniams $f(\vec{x}_i),\ i = p+1,...,2p$.
6) Pravedamos „varžybos“ tarp kiekvieno pradinio vektoriaus ir parinktų „oponentų“. Pvz.: jei $\vec{x}_i$ yra vienas iš vektorių-tėvų ir $\vec{x}_j^*$ yra vienas iš atsitiktinai parinktų vektorių-oponentų, tai laimi:

$$\text{vektorius } \vec{x}_i, \text{ jei } f(\vec{x}_i) > f(\vec{x}_j^*),$$

$$\text{vektorius } \vec{x}_j^*, \text{ jei } f(\vec{x}_i) < f(\vec{x}_j^*).$$

Toliau atliekamas tokių vektorių ranžavimas: visi vektoriai surikiuojami pagal „laimėjimų“ skaičių.

7) Atrenkami pirmieji p vektoriai su didžiausiu laimėjimų skaičiumi, kaip vektorius tėvus naujai kintamųjų variantų generacijai.

8) Skaičiavimas baigiamas, kai mutacijos nebeduoda tikslo funkcijos pagerėjimo, arba pasiekiamas užduotas iteracijų skaičius. Kitu atveju grįžtama prie 3-io žingsnio.

2.3.5 GRADIENTINIO GREIČIAUSIO NUSILEIDIMO OPTIMIZAVIMO METODO ALGORITMAS

Greičiausio nusileidimo gradientinio optimizavimo metodo algoritmas [3]:

1) Pasirenkama pradinė nepriklausomo kintamojo reikšmė x_0 ir minimumo nustatymo tikslumas ε, pasirenkamas paieškos žingsnis $\lambda^{(k)}$.
2) Skaičiuojamas gradientas g_k k- tame iteraciniame žingsnyje. Žingsnio krypties vektoriui priskiriama antigradiento kryptis $S_k = -g_k$.
3) Apskaičiuojama sekanti kintamojo reikšmė panaudojant žingsnio krypties vektorių ir žingsnio daugiklį: $x_{k+1} = x_k + a_k \cdot S_k$.
4) Apskaičiuojama tikslo funkcijos vertė naujame taške: $f_{k+1}=f(x_{k+1})$.
5) Jeigu $\|a_k \cdot S_k\| < \varepsilon$, tuomet $x^*=x_{k+1}$ ir $f(x^*)=f(x_{k+1})$ (čia x^* minimumo taškas) ir nutraukiamas skaičiavimas. Priešingu atveju, tęsiame skaičiavimus su k=k+1 nuo 2 punkto.

3. TYRIMŲ REZULTATAI IR JŲ APTARIMAS

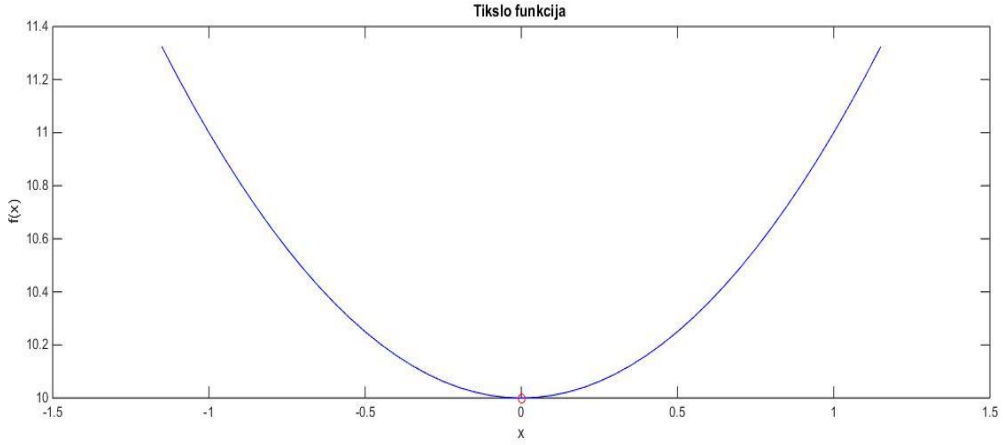
3.1 PAGRINDINĖ PROGRAMA VIENMAČIAMS OPTIMIZAVIMO METODAMS

Pagrindinėje programoje, skirtoje vienmačiams optimizavimo metodams paleisti (1 priedas), galima pasirinkti, kurį metodą norite naudoti ir įrašomas jo pavadinimas, tai atliekama naudojant Matlab „switch case“ funkciją (14 priedas). Galimas pasirinkimas šių metodų: vieno kintamojo tikslo funkcijos ekstremumo taško suradimui: Nuosekliosios paieškos vieno kintamojo, Auksinio pjūvio, Dichotomijos metodus; dviejų kintamųjų tikslo funkcijos ekstremumo taško suradimui: Nuoseklioji paieška dviejų kintamųjų tikslo funkcijai. Taip pat šioje programoje surašomi tikslo funkcijos kintamųjų parametrai: leistinoji sritis *[a, b]* ir kintamųjų reikšmių tikslumas *e*. Tikslo funkcija įrašoma į failą *TikslоFunkcija1Kint*.m (žr. 6 priedą) arba *TikslоFunkcija2Kint*.m, (žr. 7 priedą) priklausomai nuo to, ar tikslo funkcija priklauso nuo vieno ar nuo dviejų kintamųjų. Optimizavimo metodas pasirenkamas įrašius programoje metodo pavadinimą. Optimizavimo metodų pavadinimas programoje: *AP* – Auksinio pjūvio metodas, *DM* – Dichotomijos metodas, *NP1kint* ir *NP2kint* – atitinkamai vieno ir dviejų kintamųjų nuosekliosios peržiūros metodas.

3.2 VIENMAČIŲ OPTIMIZAVIMO METODŲ TAIKYMAS MATEMATINIŲ FUNKCIJŲ OPTIMIZAVIMUI

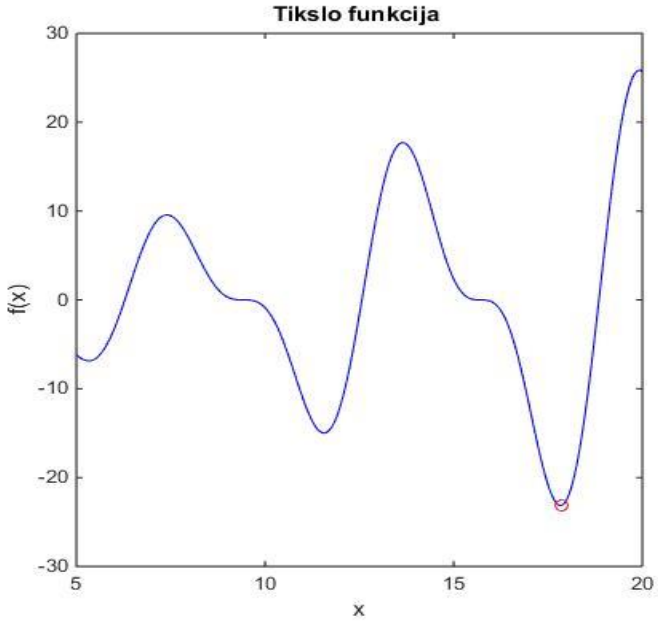
Nuoseklioji peržiūra (vieno kintamojo tikslo funkcijai)

Realizuotas Nuosekliosios peržiūros optimizavimo algoritmas MATLAB/ Simulink aplinkoje (žr. 6 priedą). Atliktas Nuosekliosios peržiūros vieno kintamojo tikslo funkcijos taikymas matematinės funkcijos optimizavimui. Naudotos unimodaliosios tikslo funkcijos išraiška: $f(x)=x^2+10$. Kintamojo x leistinoji sritis parinkta *[-1,2 1,2]*, užduotas tikslumas e=0,05. Vieno kintamojo tikslo funkcijos nuosekliosios peržiūros metodo rezultatai parodyti 3 pav. Ekstremumo (minimumo) taškas pažymėtas raudonu apskritimu. Nepriklausomo kintamojo vertė minimumo taške x*=0, o tikslo funkcijos vertė f(x*)=10.



3 pav. Unimodaliosios tikslo funkcijos $f(x)=x^2+10$ minimalios vertės radimas nuoseklios peržiūros metodu

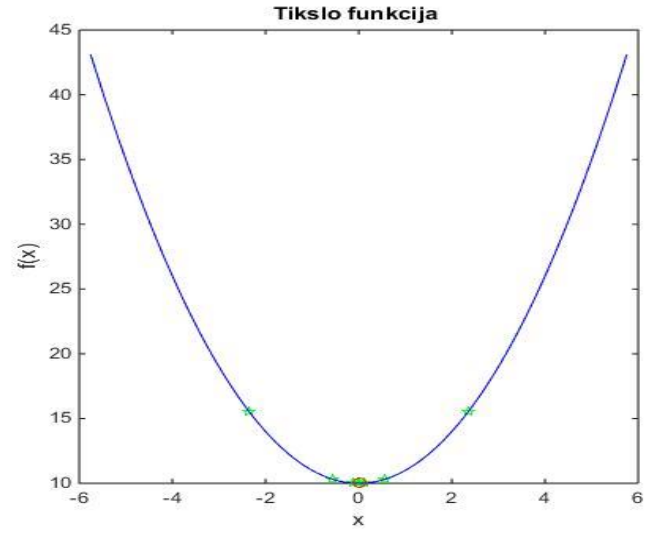
Taip pat Nuosekliosios peržiūros metodas taikomas multimodaliosios funkcijos minimumui rasti, kai tikslo funkcijos išraiška $f(x)=x\cdot\left(\sin x\cdot\left(1+\cos x\right)\right)$. Kintamojo x leistinoji sritis parinkta *[5 20]*, užduotas tikslumas e=0,05. Vieno kintamojo tikslo funkcijos nuosekliosios peržiūros metodo rezultatai parodyti 4 pav. Rastas globalus ekstremumo (minimumo) taškas leistinojoje srityje ir pažymėtas raudonu apskritimu (žr. 4 pav.). Nepriklausomo kintamojo vertė tame taške x*=17,85, o tikslo funkcijos vertė f(x*) = -23,13.



4 pav. Multimodaliosios tikslo funkcijos $f(x) = x \cdot \left(sin\, x \cdot \left(1 + cos\, x \right) \right)$ minimalios vertės radimas Nuoseklios peržiūros metodu

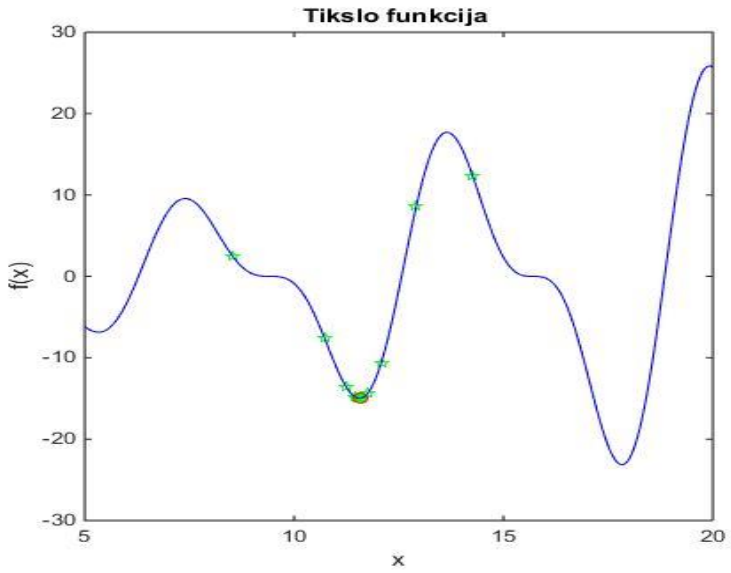
Auksinio pjūvio optimizavimo metodas

Realizuotas Auksinio pjūvio optimizavimo algoritmas MATLAB/ Simulink aplinkoje (žr. 4 priedą). Atliktas Auksinio pjūvio optimizavimo metodo taikymo matematinės funkcijos optimizavimui pavyzdys. Naudotos unimodaliosios tikslo funkcijos išraiška $f(x) = x^2 + 10$. Kintamojo x leistinoji sritis parinkta *[-6 6]*, užduotas tikslumas e=0,05. Auksinio pjūvio optimizavimo metodo rezultatai parodyti 6 pav. Minimumo taškas pažymėtas raudonu apskritimu 5 pav., o žaliais taškais parodytas algoritmo nueitas kelias iki minimumo suradimo. Ekstremumo taške kintamojo ir tikslo funkcijos reikšmės: x*=0, f(x*)=10.



5 pav. Unimodaliosios tikslo funkcijos $f(x) = x^2 + 10$ minimalios vertės radimas Auksinio pjūvio optimizavimo metodu

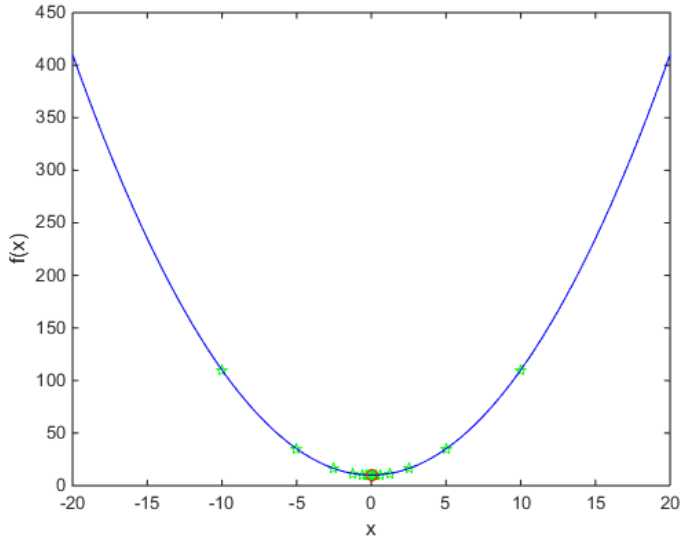
Taip pat Auksinio pjūvio optimizavimo metodas taikomas multimodaliosios funkcijos minimumui rasti, kai tikslo funkcijos išraiška $f(x) = x \cdot \left(sin\, x \cdot \left(1 + cos\, x \right) \right)$. Kintamojo x leistinoji sritis parinkta *[5 20]*, užduotas tikslumas e=0,05. Auksinio pjūvio optimizavimo metodo rezultatai parodyti 6 pav. Rastas lokalus minimumo taškas leistinojoje srityje ir pažymėtas raudonu apskritimu (žr. 6 pav.), o žaliais taškais parodytas algoritmo nueitas kelias iki minimumo suradimo. Nepriklausomo kintamojo vertė lokalaus minimumo taške x*=11,55, o tikslo funkcijos vertė f(x*)=-14,99.



6 pav. Multimodaliosios tikslo funkcijos $f(x) = x \cdot \left(sin\, x \cdot \left(1 + cos\, x \right) \right)$ minimalios vertės radimas Auksinio pjūvio optimizavimo metodu

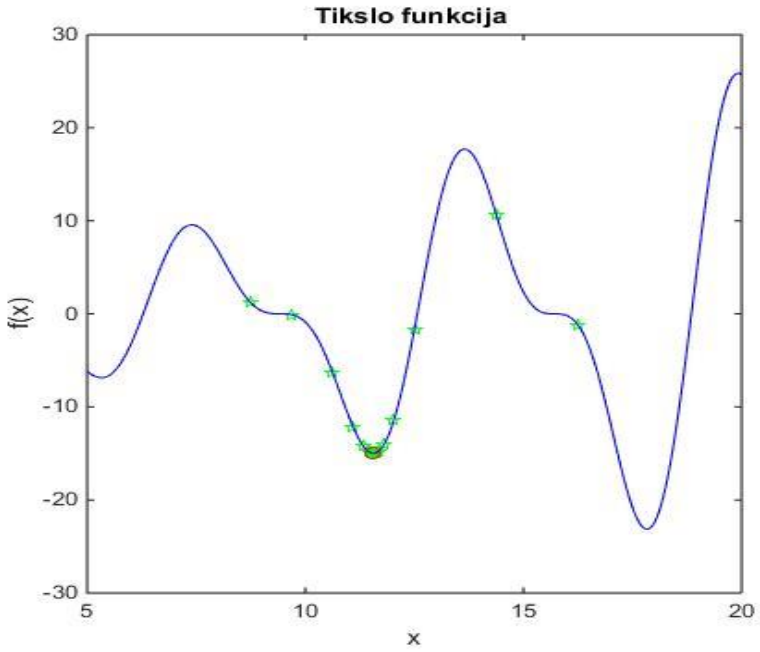
Dichotomijos optimizavimo metodas

Realizuotas Dichotomijos optimizavimo metodo algoritmas MATLAB/ Simulink aplinkoje (žr. 5 priedą). Atliktas Dichotomijos optimizavimo metodo taikymas matematinės funkcijos optimizavimui. Naudotos unimodaliosios tikslo funkcijos išraiška: $f(x) = x^2 + 10$. Kintamojo x leistinoji sritis parinkta *[-20 20]*, užduotas tikslumas e=0,05. Gauti rezultatai parodyti 8 pav. Minimumo taškas pažymėtas raudonu apskritimu (žr. 7 pav.), o žaliais taškais parodytas algoritmo nueitas kelias iki minimumo suradimo. Ekstremumo taške kintamojo ir tikslo funkcijos reikšmės: x*=-2,5, f(x*)=-15,0.



7 pav. Unimodaliosios tikslo funkcijos $f(x)=x^2+10$ minimalios vertės radimas Dichotomijos optimizavimo metodu

Taip pat Dichotomijos optimizavimo metodas taikomas multimodaliosios funkcijos minimumui rasti, kai tikslo funkcijos išraiška: $f(x)=x\cdot\left(sin\,x\cdot\left(1+cos\,x\right)\right)$. Kintamojo x leistinoji sritis parinkta *[5 20]*, užduotas tikslumas e=0,05. Dichotomijos optimizavimo metodo rezultatai parodyti 8 pav. Rastas lokalus minimumo taškas, kuris leistinojoje srityje ir pažymėtas raudonu apskritimu (žr. 8 pav.), o žaliais taškais parodytas algoritmo nueitas kelias iki minimumo suradimo. Nepriklausomo kintamojo vertė lokalaus minimumo taške x*=11,55, o tikslo funkcijos vertė f(x*)=-14,99.



8 pav. Multimodaliosios tikslo funkcijos $f(x)=x\cdot\left(sin\,x\cdot\left(1+cos\,x\right)\right)$ minimalios vertės radimas Dichotomijos metodu

Gautų rezultatų apibendrinimas

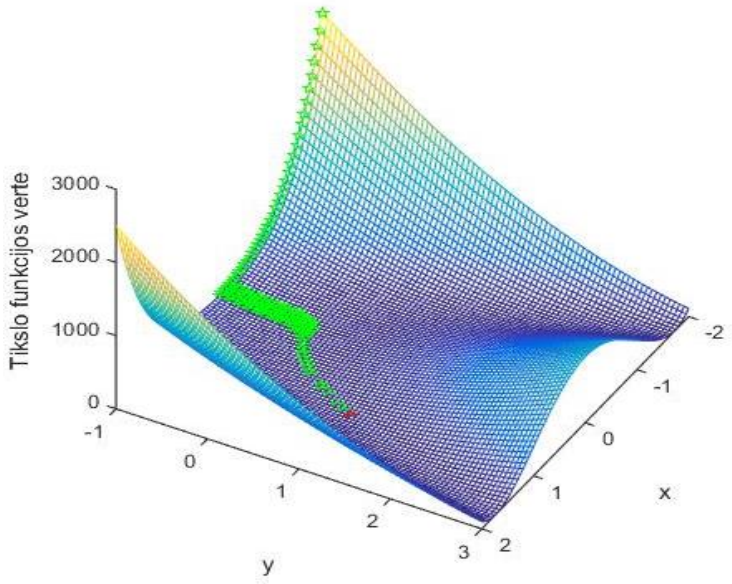
Visais trimis vienmačiais optimizavimo metodais surasti unimodaliosios ir multimodeliosios funkcijų ekstremumai ir gauti rezultatai surašyti 1 lentelėje. Unimodinės funkcijos ekstremumas: x*=0, o tikslo funkcija f(x*)=10. Unimodinės funkcijos ekstremumo taško suradimui daugiausia skaičiavimų reikia atlikti naudojant Nuosekliosios paieškos metodą (žr. 4 pav.), o mažiausia – taikant Auksinio pjūvio metodą (žr. 6 pav.). Ieškant multimodinės funkcijos ekstremumo, taikant Nuosekliosios peržiūros metodą randamas globalus minimumas, o kitais dviem metodais: Dichotomijos ir Auksinio pjūvio, randamas lokalus minimumas (žr. 1 lent.). Ieškant multimodinės funkcijos ekstremumų daugiausiai skaičiavimų taip pat atlikta Nuosekliosios peržiūros metodu (žr. 5 pav.), o mažiausiai Auksinio pjūvio (žr. 7 pav.).

1 lentelė. Vienmačiais optimizavimo metodais atlikto optimizavimo duomenys

Optimizavimo metodas	**Unimodinė funkcija** $f(x)=x^2+10$		**Multimodinė funkcija** $f(x)=x\cdot\left(\sin x\cdot\left(1+\cos x\right)\right)$		
	x*	f(x*)	Surastas ekstremumas	x*	f(x*)
Nuoseklioji peržiūra (vieno kintamojo)	0	10	globalusis	17,85	-23,13
Dichotomijos metodas	0	10	lokalusis	11,55	-14,99
Auksinio pjūvio	0	10	lokalusis	11,55	-14,99

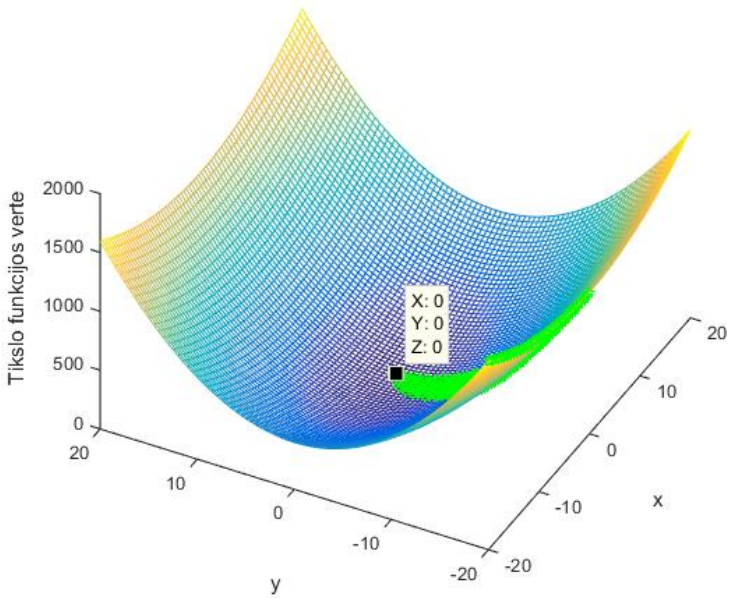
Nuoseklioji peržiūra dviejų kintamųjų

Realizuotas Nuosekliosios peržiūros (dviejų kintamųjų tikslo funcijai) optimizavimo algoritmas MATLAB/ Simulink aplinkoje (žr. 7 priedą). Atliktas šio metodo taikymas matematinės funkcijos optimizavimui. Metodui testuoti pasirinkta Rosenbroko „banana“ funkcija, kurios išraiška $f(x,y)=\left(1-x\right)^2+100\cdot\left(y-x^2\right)^2$. Kintamojo x leistinoji sritis parinkta *[-2 2]*, o y: *[-1 3]*, užduotas tikslumas e=0,05. Dviejų kintamųjų nuosekliosios peržiūros metodo rezultatai parodyti 9 pav. Minimumo taškas pažymėtas raudonu apskritimu, o žaliais apskritimais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=1, y*=1, o tikslo funkcijos vertė f(x*,y*)=0.



9 pav. Rosenbroko „banana“ funkcijos $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$ minimalios vertės radimas Nuoseklios peržiūros metodu

Antram dviejų kintamųjų nuosekliosios peržiūros metodo testavimui pasirinkta paraboloido funkcija, kurios išraiška $f(x,y)=x^2+y^2$. Kintamojo x leistinoji sritis parinkta *[-20 20]*, o y: *[-20 20]*, užduotas tikslumas e=0,05. Dviejų kintamųjų nuosekliosios peržiūros metodo rezultatai parodyti 10 pav. Minimumo taškas pažymėtas juodu apskritimu, o žaliais apskritimais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=0, y*=0, o tikslo funkcijos vertė f(x*,y*)=0.



10 pav. Paraboloido funkcijos $f(x,y)=x^2+y^2$ minimalios vertės radimas nuoseklios peržiūros metodu

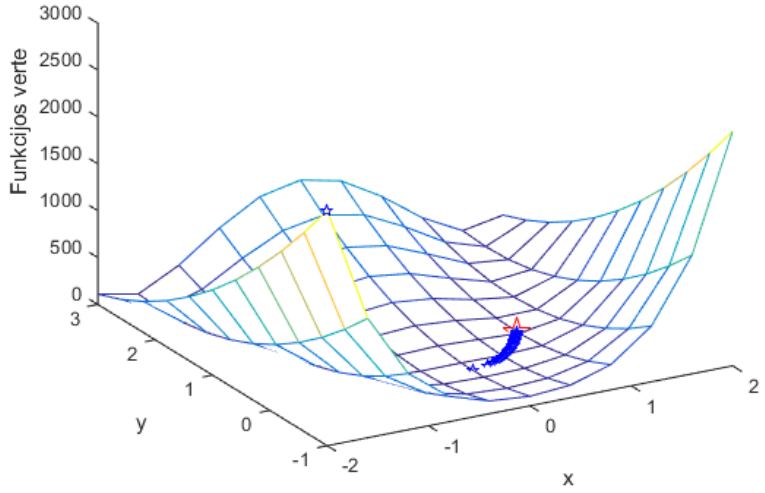
3.3 PAGRINDINĖ PROGRAMA DAUGIAMAČIAMS OPTIMIZAVIMO METODAMS

Pagrindinėje programoje, skirtoje daugiamačiams optimizavimo metodams paleisti (žr. 8 priedą), galima pasirinkti, kurį metodą norite naudoti ir įrašomas jo pavadinimas, tai atliekama naudojant Matlab „switch case“ funkciją (žr. 15 priedą). Galimas pasirinkimas iš šių metodų: Evoliucinio programavimo, *Chemotaxis* metodo, Imituojamo grūdinimo opt. metodo, Gauso-Zaidelio opt. metodo, Gradientinio greičiausio nusileidimo. Taip pat šioje programoje surašomi parametrai, kurie reikalingi visiems optimizavimo metodams: tikslo funkcijos kintamųjų leistinosios sritys *[a, b]*, pradinės kintamųjų vertės x_0, reikalingas tikslumas ir parametrai, kurie reikalingi tik tam tikram optimizavimo metodui. Tikslo funkcija įrašoma į programą (žr. 9 priedą). Optimizavimo metodų žymėjimas programoje: *GausoZeidelio* – Gauso-Zeidelio metodas, *Gradientinis* – Gradientinis opt. metodas, *Simulated-Annealing* – Imituojamo grūdinimo metodas, *Evoliucinis* – Evoliucinio programavimo metodas, *Chemotaxis* – Chemotaxis opt. metodas.

3.4 DAUGIAMAČIŲ OPTIMIZAVIMO METODŲ TAIKYMAS MATEMATINIŲ FUNKCIJŲ OPTIMIZAVIMUI

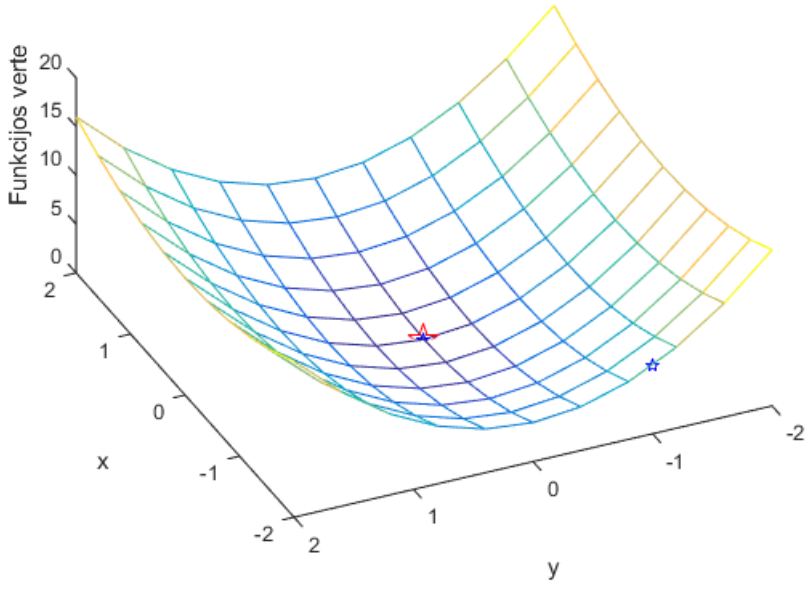
Gauso-Zaidelio optimizavimo metodas

Realizuotas Gauso-Zaidelio optimizavimo algoritmas MATLAB/ Simulink aplinkoje (žr. 12 priedą). Atliktas metodo taikymas matematinės funkcijos optimizavimui. Metodui testuoti pasirinkta Rosenbroko „banana“ funkcija, kurios išraiška: $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$. Kintamojo x leistinoji sritis parinkta *[-2 2]*, o y: *[-1 3]*, funkcijos parametrų kitimo žingsnis e=0,00005. Rezultatai parodyti 11 pav. Minimumo taškas pažymėtas raudona žvaigždute, o mėlynais simboliais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=0,8, y*=0,6, o tikslo funkcijos vertė f(x*,y*)=0, 0384.



11 pav. Rosenbroko „banana“ funkcijos $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$ minimalios vertės radimas Gauso-Zaidelio optimizavimo metodu

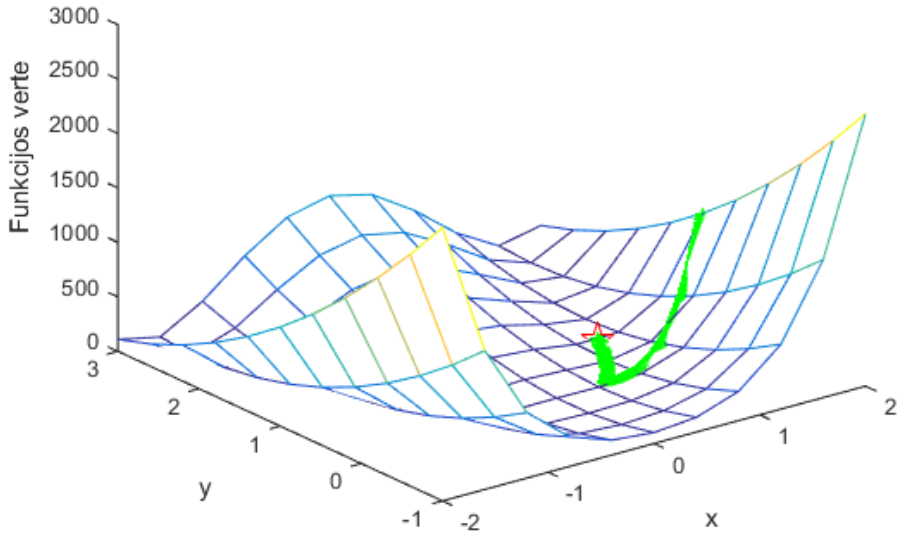
Antram Gauso-Zaidelio metodo testavimui pasirinkta paraboloido funkcija, kurios išraiška: $f(x,y)=x^2+y^2$. Kintamojo x leistinoji sritis parinkta *[-20 20]*, o y: *[-20 20]*, užduotas tikslumas e=0,00005. Dviejų kintamųjų nuosekliosios peržiūros metodo rezultatai parodyti 12 pav. Minimumo taškas pažymėtas raudona žvaigždute, o mėlynais simboliais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=0, y*=0, o tikslo funkcijos vertė f(x*,y*)=0.



12 pav. Paraboloido funkcijos $f(x,y)=x^2+y^2$ minimalios vertės radimas Gauso-Zaidelio optimizavimo metodu

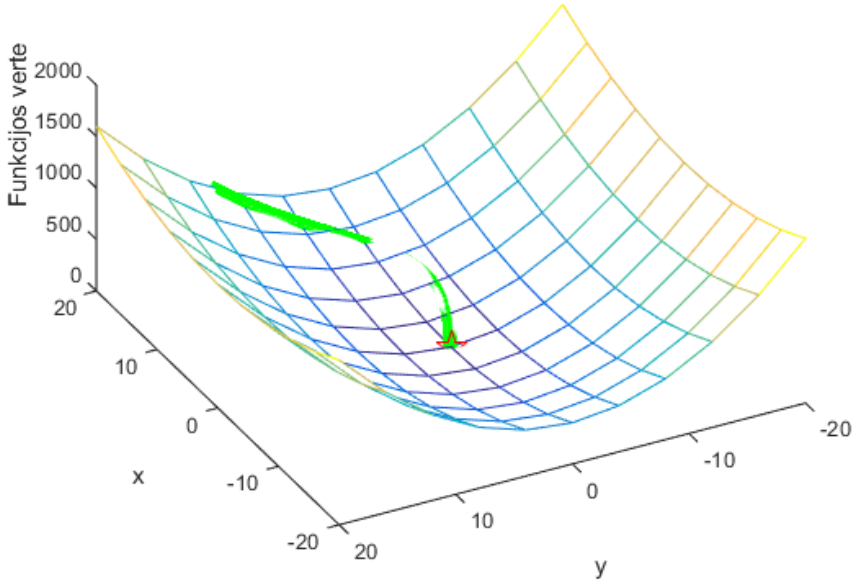
Gradientinis greičiausio nusileidimo optimizavimo metodas

Realizuotas Gradientinis greičiausio nusileidimo optimizavimo algoritmas MATLAB/ Simulink aplinkoje (žr. 13 priedą). Atliktas metodo taikymas matematinės funkcijos optimizavimui. Metodui testuoti naudojama Rosenbroko „banana“ funkcija, kurios išraiška: $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$. Kintamojo x leistinoji sritis parinkta *[-2 2]*, o y: *[-1 3]*, užduotas tikslumas eps=0,00005. Pradinės nepriklausomų kintamųjų vertės: x_0=2, y_0=-1. Parinktas fiksuotas paieškos žingsnis α=0,001. Naudojant šį metodą ekstremumo paieškai gauti rezultatai parodyti 13 pav. Minimumo taškas pažymėtas raudonu apskritimu, o žaliais apskritimais pažymėtas ekstremumo taško paieška minimumo link. Gautos nepriklausomų kintamųjų vertės minimumo taške x*=1,0005, y*=1,0005, o tikslo funkcijos vertė f(x*,y*)=2,5*10^{-5}.



13 pav. Rosenbroko „banana“ funkcijos $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$ minimalios vertės radimas Gradientiniu greičiausio nusileidimo optimizavimo metodu

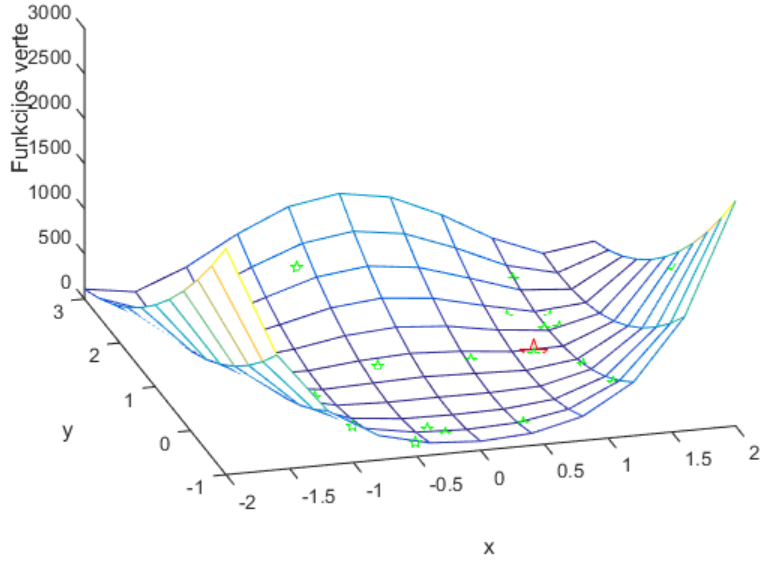
Kitam algoritmo testavimui naudojama paraboloido funkcija, kurios išraiška: $f(x,y)=x^2+y^2$. Kai kintamojo x leistinoji sritis parinkta *[-20 20]*, o y: *[-20 20]*. Pasirinktas fiksuotas paieškos žingsnis α=0,01. Gauti rezultatai pateikti 14 pav. Minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas minimumo paieškos kelias. Gautos nepriklausomų kintamųjų vertės minimumo taške x*=0, y*=-0,0011, o tikslo funkcijos vertė f(x*,y*)=4*10^{-6}.



14 pav. Paraboloido funkcijos $f(x,y)=x^2+y^2$ minimalios vertės radimas Gradientiniu greičiausio nusileidimo optimizavimo metodu

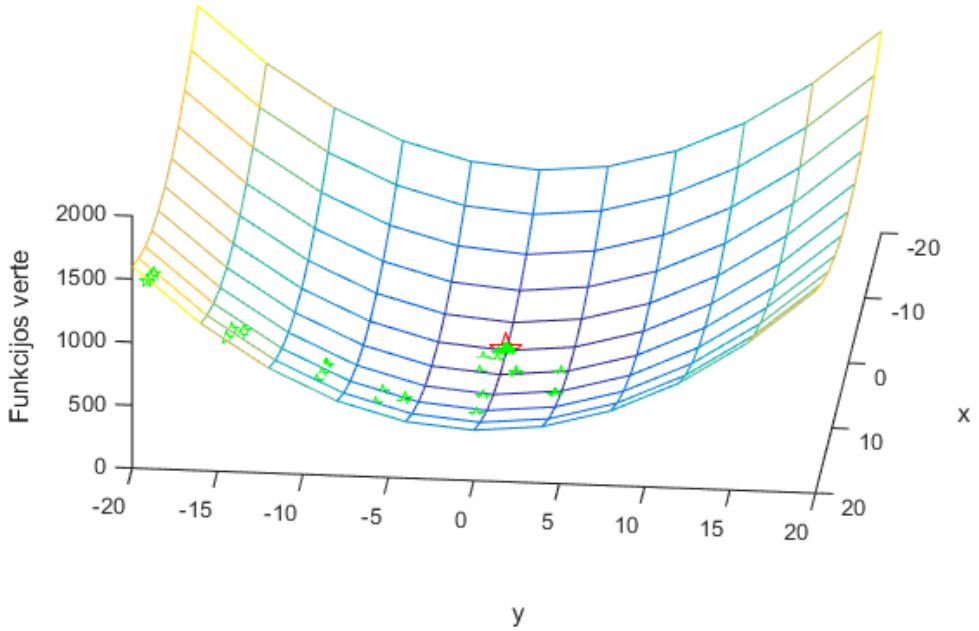
Chemotaxis optimizavimo metodas

Realizuotas Chemotaxis optimizavimo algoritmas MATLAB/ Simulink aplinkoje (žr. 10 priedą). Atliktas metodo taikymas matematinės funkcijos optimizavimui. Metodui testuoti pasirinkta Rosenbroko „banana“ funkcija, kurios išraiška $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$ Kintamojo x leistinoji sritis parinkta *[-2 2]*, o y: *[-1 3]*, užduotas tikslumas eps=0,00005. Pradinės nepriklausomų kintamųjų vertės: x_0=2, y_0=1. Gauti rezultatai parodyti 15 pav. Minimumo taškas pažymėtas raudonu apskritimu, o žaliais apskritimais pažymėtas funkcijos minimumo taško paieškos eiga. Gautos nepriklausomų kintamųjų vertės minimumo taške x*=0,89, y*=0,774, o tikslo funkcijos vertė f(x*,y*)=0,0517.



15 pav. Rosenbroko „banana“ funkcijos $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$ minimalios vertės radimas Chemotaxis optimizavimo metodu

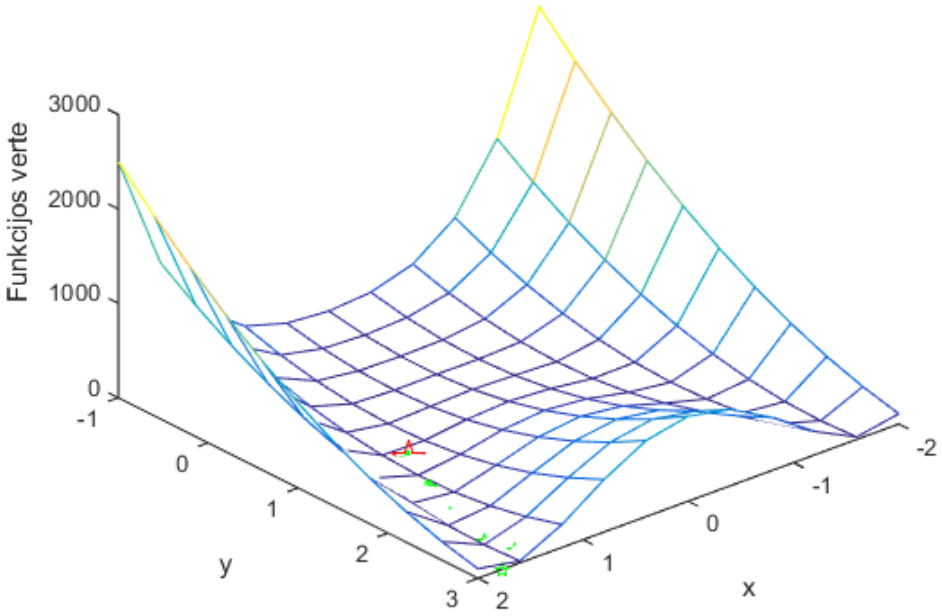
Kitam chemotaxis metodo testavimui naudojama paraboloido funkcija, kurios išraiška: $f(x,y)=x^2+y^2$. Kintamojo x leistinoji sritis parinkta *[-20 20]*, o y: *[-20 20]*. Pradinės nepriklausomų kintamųjų vertės: x_0=20, y_0=-20. Gauti rezultatai pateikti 16 pav. Minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės gautame minimumo taške x*=0,0261, y*=-0,0362, o tikslo funkcijos vertė f(x*,y*)=0,0046.



16 pav. Paraboloido funkcijos $f(x,y)=x^2+y^2$ minimalios vertės radimas Chemotaxis optimizavimo metodu

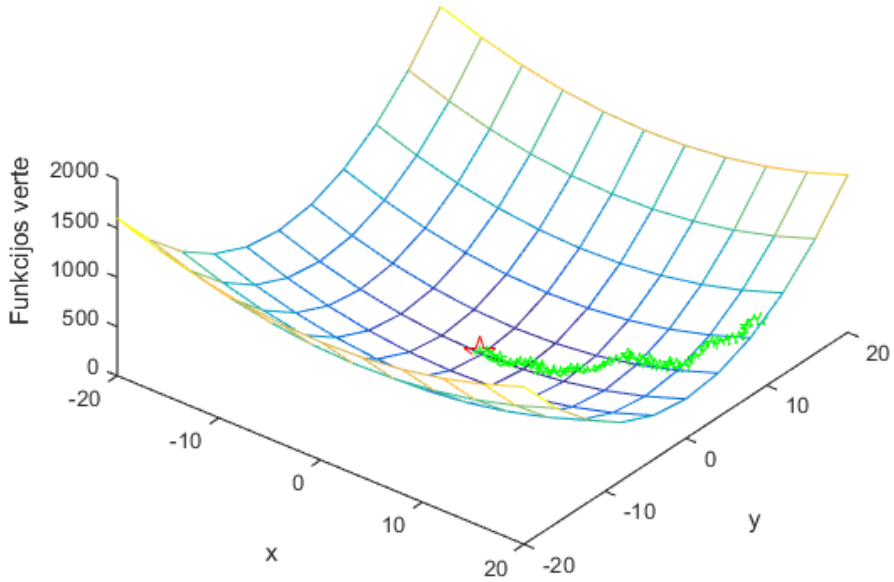
Imituojamo atkaitinimo optimizavimo metodas

Realizuotas Imituojamo atkaitinimo optimizavimo algoritmas MATLAB/Simulink aplinkoje (žr. 16 priedą). Atliktas metodo taikymas matematinės funkcijos optimizavimui. Metodui testuoti pasirinkta Rosenbroko „banana“ funkcija, kurios išraiška $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$. Kintamojo x leistinoji sritis parinkta *[-2 2]*, o y: *[-1 3]*, užduotas tikslumas eps=0,00005. Pradinės nepriklausomų kintamųjų vertės: x_0=2, y_0=3, paieškos žingsniai: λ_1=0,5 ir λ_2=0,5. Gauti rezultatai parodyti 17 pav. Minimumo taškas pažymėtas raudonu apskritimu, o žaliais apskritimais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=1,06, y*=1,12, o tikslo funkcijos vertė f(x*,y*)=0,3709.



17 pav. Rosenbroko „banana“ funkcijos $f(x,y)=(1-x)^2+100\cdot\left(y-x^2\right)^2$ minimalios vertės radimas Imituojamo atkaitinimo optimizavimo metodu

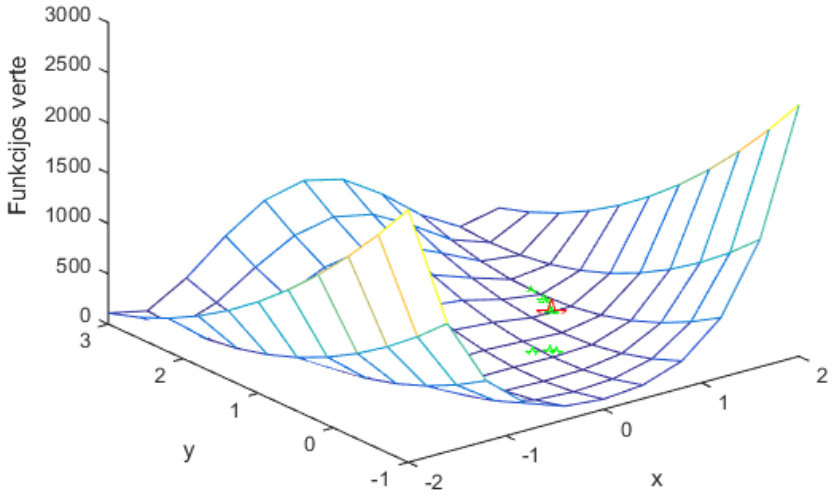
Kitam Imituojamo grūdinimo metodo testavimui naudojama paraboloido funkcija, kurios išraiška: $f(x,y)=x^2+y^2$. Kintamojo x leistinoji sritis parinkta *[-20 20]*, o y: *[-20 20]*. Pradinės nepriklausomų kintamųjų vertės: x_0=20, y_0=10, paieškos žingsniai λ_1=0,5 ir λ_2=0,5. Gauti rezultatai pateikti 18 pav. Minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=0,012, y*=-0,0917, o tikslo funkcijos vertė f(x*,y*)=0,02403.



18 pav. Paraboloido funkcijos $f(x,y)=x^2+y^2$ minimalios vertės radimas Imituojamo grūdinimo optimizavimo metodu

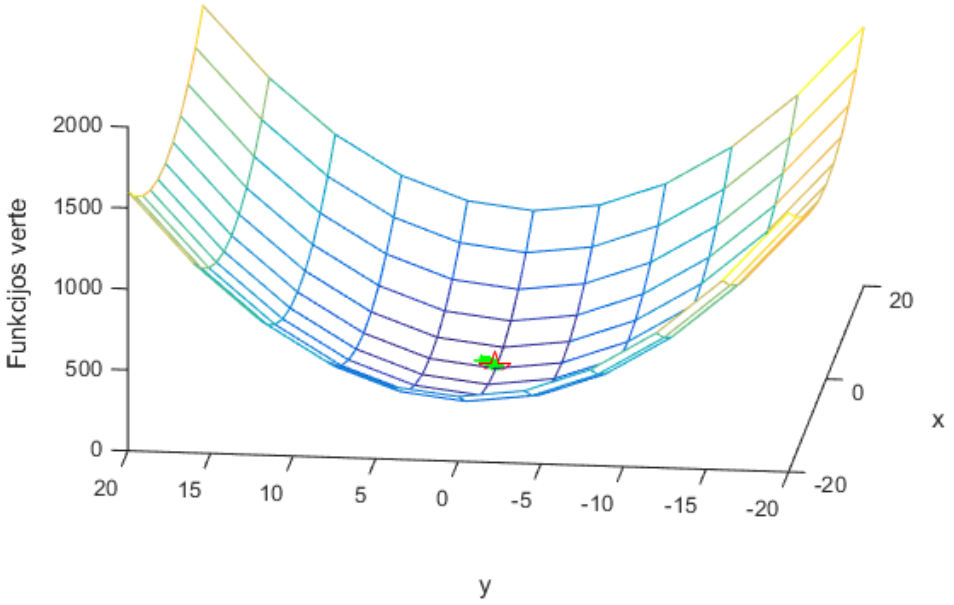
Evoliucinis programavimas

Realizuotas Evoliucinio programavimo algoritmas MATLAB/ Simulink aplinkoje (žr. 20 priedą). Atliktas Evoliucinio programavimo taikymas matematinės funkcijos optimizavimui. Metodui testuoti pasirinkta Rosenbroko „banana“ funkcija, kurios išraiška: $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$. Kintamojo x leistinoji sritis parinkta *[-2 2]*, o y: *[-1 3]*, užduotas tikslumas eps=0,00005. Pradinės nepriklausomų kintamųjų vertės: x_0=2, y_0=3, parametrai, nusakantys mutacijos laipsnį: Δx=0,025, Δy=0,025, tėvų skaičiu p=10. Rezultatai parodyti 19 pav. Minimumo taškas pažymėtas raudonu apskritimu, o žaliais apskritimais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=0,998, y*=0,9858, o tikslo funkcijos vertė f(x*,y*)=5,5399*10^{-5}.



19 pav. Rosenbroko „banana“ funkcijos $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$ minimalios vertės radimas Evoliucinio programavimo metodu

Taip pat Evoliucinio programavimo metodo testavimui naudojama paraboloido funkcija, kurios išraiška: $f(x,y)=x^2+y^2$. Kintamojo x leistinoji sritis parinkta *[-20 20]*, o y: *[-20 20]*. Pradinės nepriklausomų kintamųjų vertės: x_0=20, y_0=10, parametrai, nusakantys mutacijos laipsnį: Δx=0,025, Δy=0,025, tėvų skaičiu p=10. Gauti rezultatai pateikti 20 pav. Minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimumo link. Nepriklausomų kintamųjų vertės minimumo taške x*=-0,183*10^{-3}, y*=0,077*10^{3}, o tikslo funkcijos vertė f(x*,y*)=5,15*10^{-8}.



20 pav. Paraboloido funkcijos $f(x,y)=x^2+y^2$ minimalios vertės radimas Evoliucinio programavimo metodu

Gautų rezultatų apibendrinimas

Daugiamačių optimizavimo metodų taikymo dviejų funkcijų optimizavimui rezultatai surašyti 2 ir 3 lentelėse. Rosenroko „banana“ funkcijos ekstremumas tiksliausiai nustatomas Nuosekliosios paieškos metodu (žr. 2 lent.), tačiau šiuo metodu optimizavimas atliekamas lėčiausiai. Evoliucinio programavimo, Imituojamo grūdinimo ir Chemotaxis metodai yra atsitiktinės paieškos, todėl jais didesnė tikimybė rasti globalųjį ekstremumą.

2 lentelė. Daugiamačių optimizavimo metodų taikymo Rosenbroko „banana“ funkcijos optimizavimui rezultatai

Optimizavimo metodas	**Rosenbroko „banana“** $f(x,y)=(1-x)^2+100\cdot(y-x^2)^2$		
	x*	y*	f(x*,y*)
Nuoseklioji peržiūra (dviejų kintamųjų)	1	1	0
Gauso-Zaidelio	0,8	0,6	0,0384
Gradientinis greičiausio nusileidimo	1,0005	1,0005	$2,5*10^{-5}$
Chemotaxis	0,89	0,774	0,0517
Imituojamo grūdinimo	1,06	1,12	0,3709
Evoliucinio programavimo	0,998	0,9858	$5,539*10^{-5}$

Paraboloido funkcijos ekstremumas tiksliausiai nustatomas Nuosekliosios peržiūros ir Gauso-Zaidelio metodais (žr. 3 lent.), tačiau su Gauso-Zaidelio metodu reikia atlikti mažiau skaičiavimų negu su Nuosekliosios peržiūros.

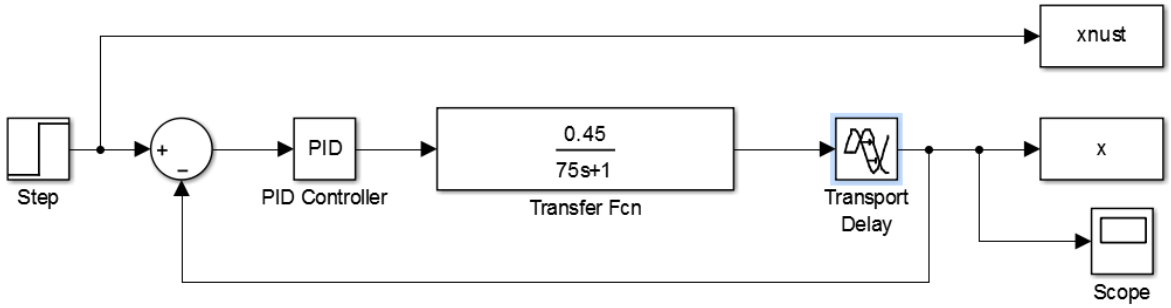
3 lentelė. Daugiamačių optimizavimo metodų taikymo paraboloido funkcijos optimizavimui rezultatai

Optimizavimo metodas	**Paraboloidas** $f(x,y)=x^2+y^2$		
	x*	y*	f(x*,y*)
Nuoseklioji peržiūra (dviejų kintamųjų)	0	0	0
Gauso-Zaidelio	0	0	0
Gradientinis greičiausio nusileidimo	0	-0,0011	$4*10^{-6}$
Chemotaxis	0,0261	-0,0362	0,0046
Imituojamo grūdinimo	0,012	-0,0917	0,02403
Evoliucinio programavimo	$-0,183*10^{-3}$	$0,077*10^{-3}$	$5,15*10^{-8}$

3.5 OPTIMIZAVIMO METODŲ TAIKYMAS REGULIATORIAUS PARAMETRAMS NUSTATYTI

3.5.1 MATLAB/SIMULINK MODELIS OPTIMIZAVIMO PARAMETRAMS TIRTI

Optimizavimo metodams testuoti buvo sudaryta vienkontūrė ARS (Automatinė reguliatoriaus sistema), kuri parodyta 21 pav.



21 pav. Simulink modelis optimizavimo metodų taikymui

Objekto perdavimo funkcija :

$$W_0(s)=\frac{k_0\cdot e^{-\tau_0 s}}{T_0 s+1} \qquad (3.1)$$

Čia:

τ_0 - perdavimo funkcijos vėlavimas;

T_0- perdavimo funkcijos vėlavimas;

k_0- perdavimo funkcijos koeficientas;

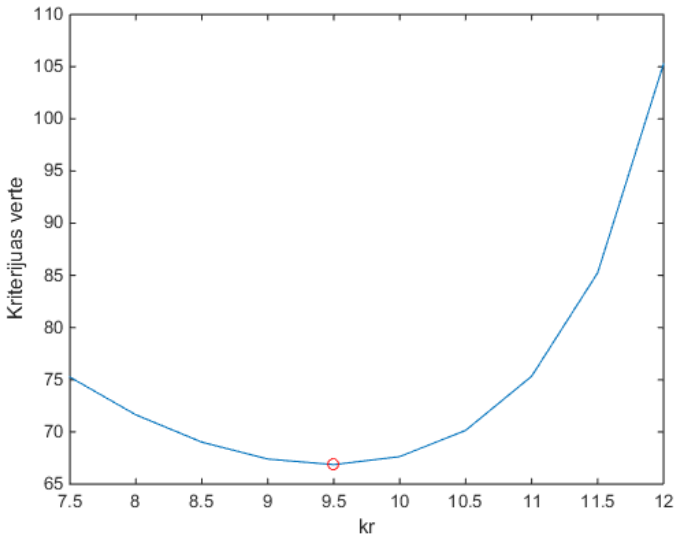
S– Laplaso transformacijos operatorius.

Optimizavimo metodams tirti parinkti tokie perdavimo funkcijos parametrai: stiprinimo koeficientas k_0= 0,45, vėlavimas τ_0=22, laiko pastovioji T_0= 75.

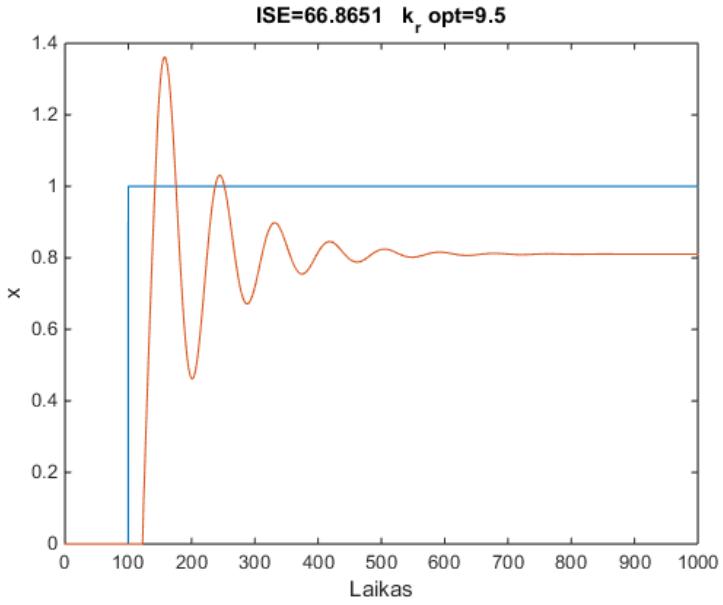
3.5.2 VIENMAČIŲ OPTIMIZAVIMO METODŲ TAIKYMAS REGULIATORIAUS PARAMETRAMS NUSTATYTI

Nuoseklioji peržiūra (vieno kintamojo tikslo funkcijai)

Atliktas Nuosekliosios peržiūros (vieno kintamojo) taikymas P reguliatoriaus parametro geriausiam parinkimui, pagal pasirinkto kriterijaus (ISE ar ITAE) vertę. Reguliatoriaus proporcinės dedamosios k_r leistinoji sritis parinkta *[7,5 12]*, užduotas tikslumas e=0,5. Vieno kintamojo nuosekliosios peržiūros metodo rezultatai parodyti 22 pav. Ekstremumo (minimumo) taškas pažymėtas raudonu apskritimu ir stiprinimo vertė tame taške k_r*=9,5, o kriterijaus vertė – 66,86. 23 grafike pavaizduota pereinamojo proceso reakcijos kreivė, kai reguliatoriaus stiprinimas k_r=9,5.



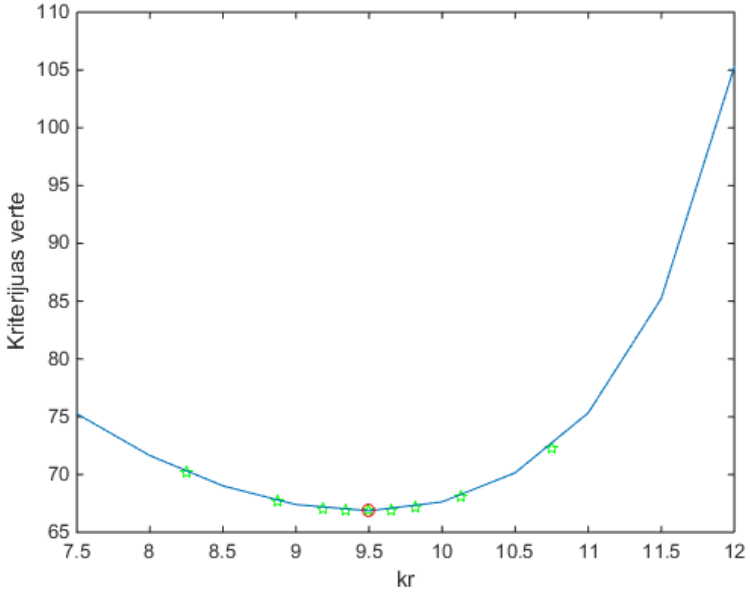
22 pav. Minimalios kriterijaus vertės (ISE) ir stiprinimo kr optimalios vertės radimas naudojant Nuosekliosios paieškos metodą



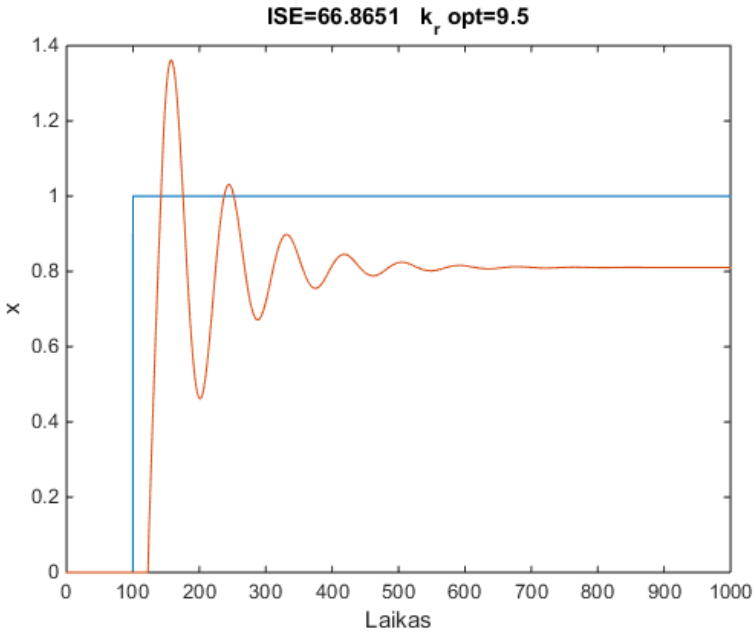
23 pav. Pereinamojo proceso reakcijos kreivė, su optimalia k_r verte, kuri buvo gauta taikant nuosekliosios paieškos metodą

Dichotomijos optimizavimo metodas

Atliktas Dichotomijos optimizavimo metodo taikymas P reguliatoriaus parametro geriausiam parinkimui pagal pasirinkto kriterijaus (ISE ar ITAE) vertę. Kintamojo k_r leistinoji sritis parinkta *[7,5 12]*, užduotas tikslumas e=0,5, diskretizavimo žingsnis T_s=0,1. Gauti rezultatai parodyti 24 paveiksle. Minimumo taškas pažymėtas raudonu apskritimu, kuriame reguliatoriaus vertė k_r*=9,5, o kriterijaus minimali verte – 66,86. 25 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė, su optimaliu stiprinimu k_r.



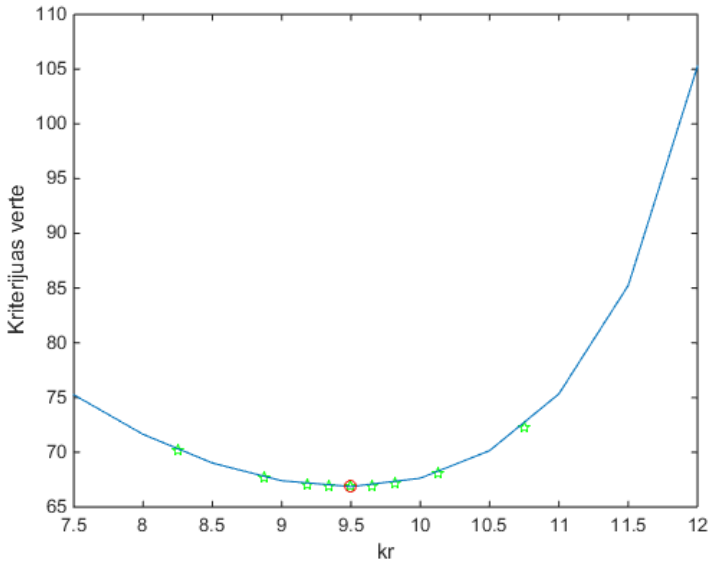
24 pav. Minimalios kriterijaus vertės (ISE) suradimas Dichotomijos metodu



25 pav. Pereinamojo proceso reakcijos kreivė, su optimalia k_r verte, kuri buvo gauta taikant Dichotomijos metodą

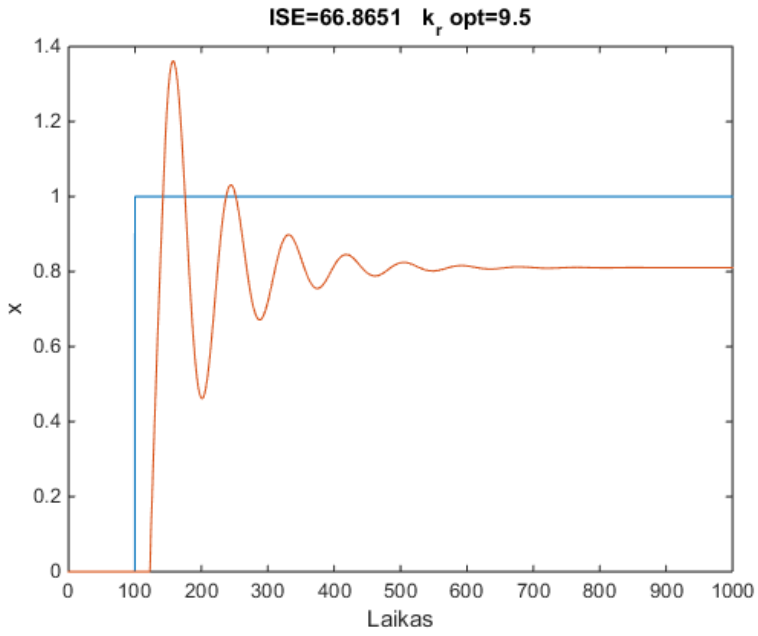
Auksinio pjūvio optimizavimo metodas

Atliktas Auksinio pjūvio optimizavimo taikymas P reguliatoriaus stiprinimui kr optimaliam parinkimui pagal pasirinkto kriterijaus (ISE ar ITAE) vertę. Kintamojo k_r leistinoji sritis parinkta *[7,5 12]*, užduotas tikslumas e=0,5, diskretizavimo žingsnis T_s=0,1. Gauti rezultatai parodyti 26 paveiksle. Minimumo taškas pažymėtas raudonu apskritimu, kuriame reguliatoriaus vertė k_r*=9,5, o kriterijaus minimali verte – 66,86.



26 pav. Minimalios kriterijaus vertės (ISE) suradimas Auksinio pjūvio metodu

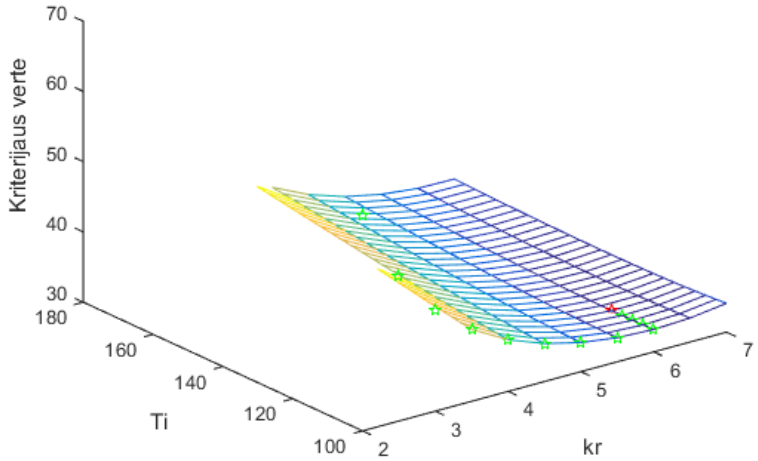
27 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė, su optimaliu stiprinimu k_r.



27 pav. Pereinamojo proceso reakcijos kreivė, su optimalia k_r verte, kuri buvo gauta taikant Dichotomijos metodą

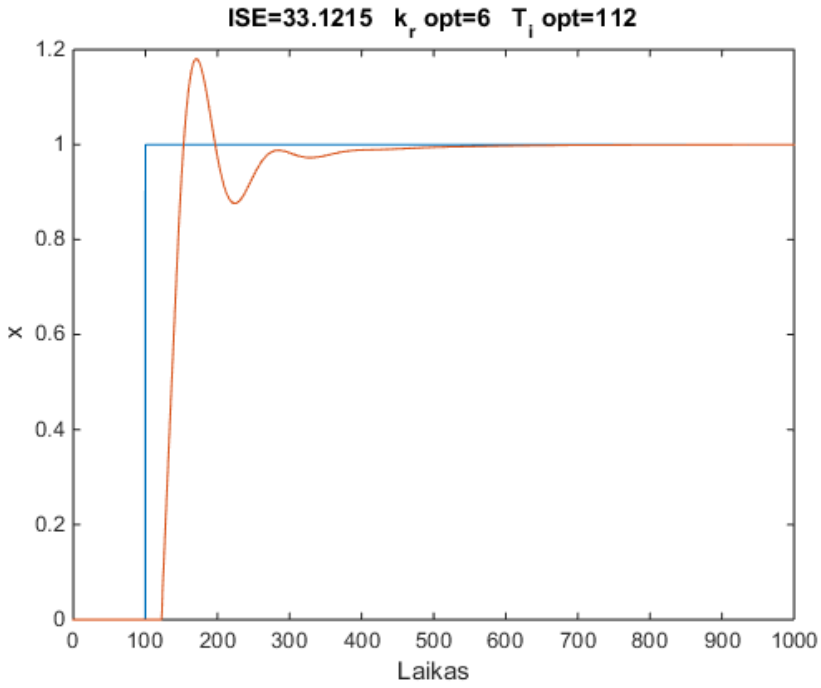
Nuoseklioji peržiūra (dviejų kintamųjų tikslo funkcijai)

Atliktas Nuosekliosios peržiūros (dviejų kintamųjųtikslo funkcijai) optimizavimo taikymas PI reguliatoriaus stiprinimui kr ir Ti integralinės dedamosios optimaliam parinkimui pagal pasirinkto kriterijaus (ISE ar ITAE) vertę. Kai stiprinimo (k_r) leistinoji sritis parinkta *[7 12]*, integralinės dedamosios T_i leistinoji sritis *[100 180]*, užduotas kintamųjų kitimo žingsnis e=*[0,5 3]*, diskretizavimo žingsnis T_s=0,1. Gauti rezultatai parodyti 28 paveiksle. Minimumo taškas pažymėtas raudonu apskritimu, kuriame reguliatoriaus stiprinimo vertė k_r*=6, integralinės dedamosios vertė T_i*=112. Žaliais simboliais pavaizduotas artėjimas prie kriterijaus (ISE) minimumo. Kriterijaus minimali vertė (ISE) – 33,12.



28 pav. Minimalios kriterijaus vertės (ISE) suradimas nuosekliosios peržiūros metodu

39 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė esant reguliatoriaus parametrams: kr*, Ti*.



29 pav. Pereinamojo proceso reakcijos kreivė, su optimalia k_r ir Ti verte, kuri buvo gauta taikant nuosekliosios peržiūros metodu

3.5.3 DAUDIAMAČIŲ OPTIMIZAVIMO METODŲ TAIKYMAS REGULIATORIAUS DERINIMO PARAMETRAMS NUSTATYTI

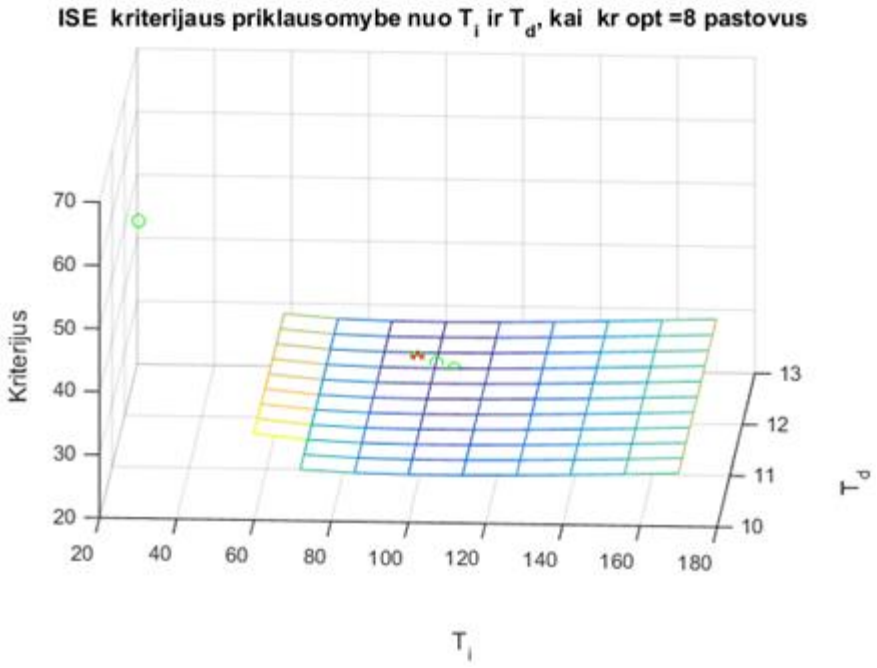
Realizuotiems daugiamačiams optimizavimo metodams testuoti buvo pasirinkti tokie bendri parametrai :

- pradinės reguliatoriaus parametrų vertės: $kr_0 = 1$, $T_i = 30$, $T_d = 10$;
- diskretizavimo žingsnis $T_s = 0,1$;
- reguliatoriaus parametrų ribos: kr lestinoji sritis *[1 10]*, T_i leistinoji sritis *[30 170]*, T_d leistinoji sritis *[10 16];*
- paieškos tikslumo parametras eps = 0,000001.

Gauso-Zaidelio optimizavimo metodas

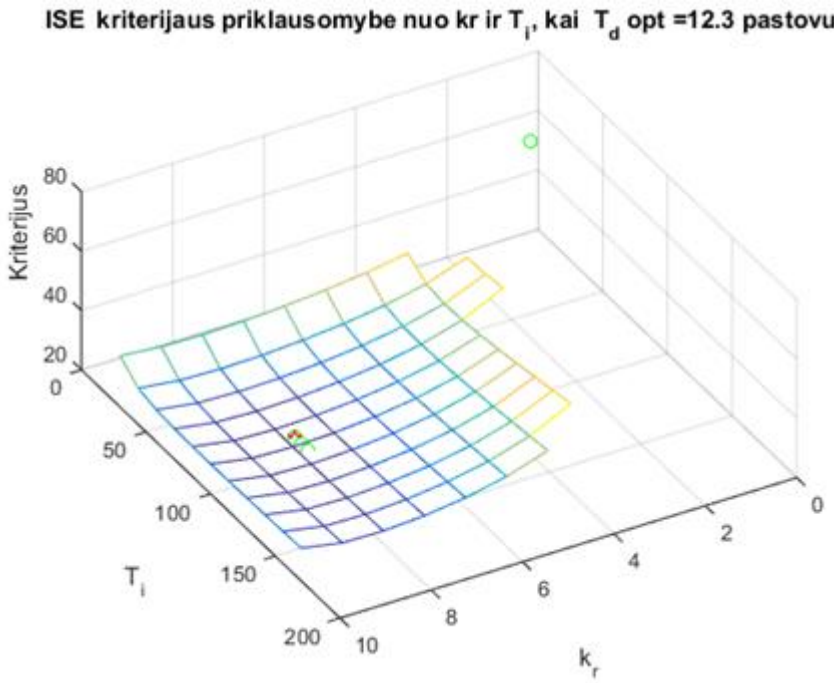
Atliktas Gauso-Zaidelio optimizavimo metodo taikymas PID reguliatoriaus parametrams rasti pagal pasirinktą kriterijų (ISE). Reguliatoriaus parametrų kitimo žingsnis e = [1 5 0,1]. Rezultatai parodyti 30, 31, 32 pav. Šiuose grafikuose minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimumo link. Reguliatoriaus parametrų vertės minimumo taške: $k_r^* = 8$, $T_i^* = 95$, $T_d^* = 12,3$, o kriterijaus vertė ISE= 27,375.

30 pav. parodyta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i ir diferencialinės T_d laiko dedamosios, kai reguliatoriaus stiprinimas k_r yra pastovus dydis, kurio vertė k_r=8.



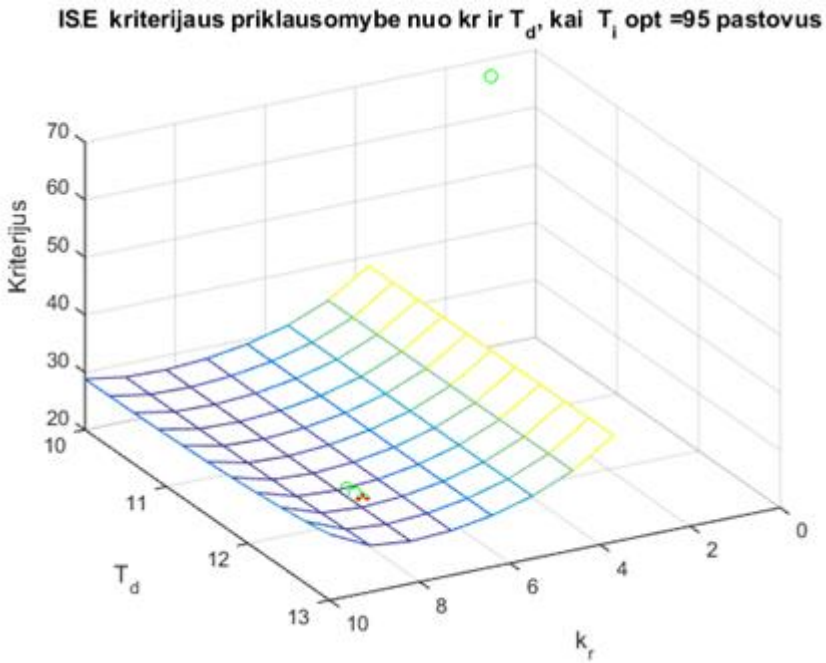
30 pav. Minimalaus taško radimas Gauso-Zaidelio optimizavimo metodu

31 pav. atvaizduota ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i laiko dedamosios ir stiprinimo k_r, kai diferencialinė laiko dedamoji yra pastovus dydis, kurios vertė T_d=12,3.



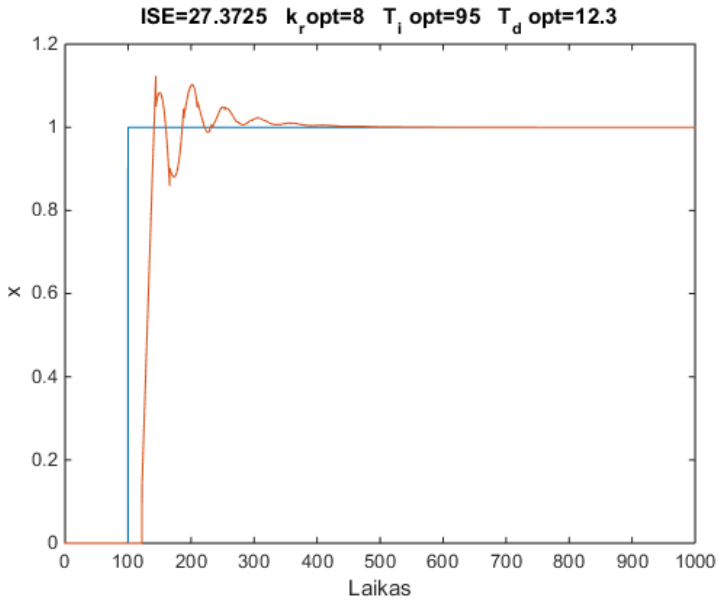
31 pav. Minimalaus taško radimas Gauso-Zaidelio opt. metodu

32 pav. atidėta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus diferencialinės laiko dedamosios T_d ir stiprinimo k_r, kai integralinė laiko dedamoji yra pastovus dydis, kurios vertė T_i=95.



32 pav. Minimalaus taško radimas Gauso-Zaidelio opt. metodu

33 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė esant reguliatoriaus parametrams: k_r*, T_i*, T_d*.



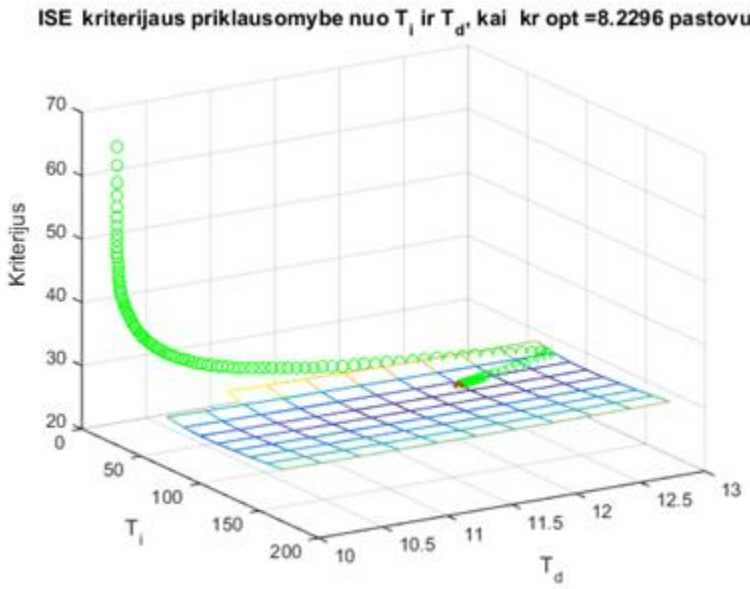
33 pav. Pereinamojo proceso kreivė, su optimaliais reguliatoriaus parametrais

Gadientinis greičiausio nusileidimo optimizavimo metodas

Atliktas Gradientinis greičiausio nusileidimo optimizavimo metodo taikymas PID reguliatoriaus parametrams rasti pagal pasirinktą kriterijų (ISE). Reguliatoriaus parametrų

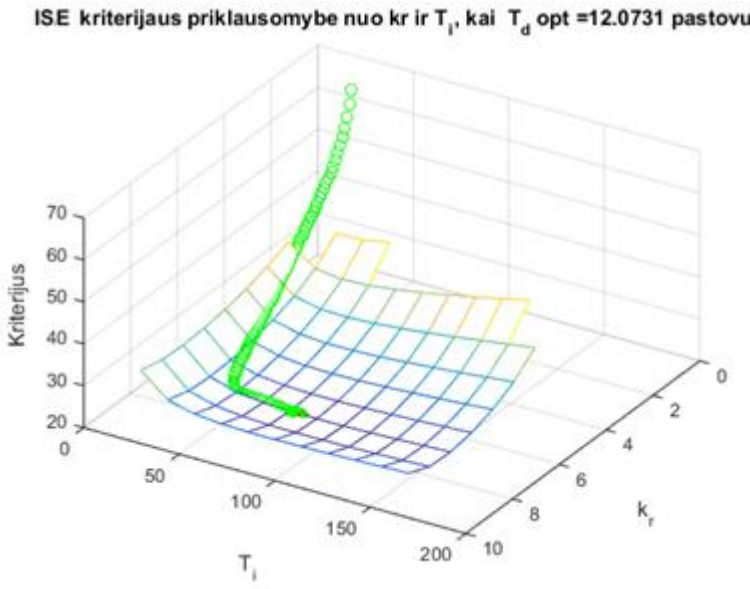
fiksuotas paieškos žingsnis λ = *[0,1 2 0,1]*. Rezultatai parodyti 34, 35, 36 pav. Grafikuose minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimumo link. Reguliatoriaus parametrų vertės minimumo taške: k_r*=8,23, T_i*=92,8, T_d*=12,07, o kriterijaus vertė ISE= 27,34.

34 pav. parodyta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i ir diferencialinės T_d laiko dedamosios, kai reguliatoriaus stiprinimas k_r yra pastovus dydis, kurio vertė k_r=8,23.



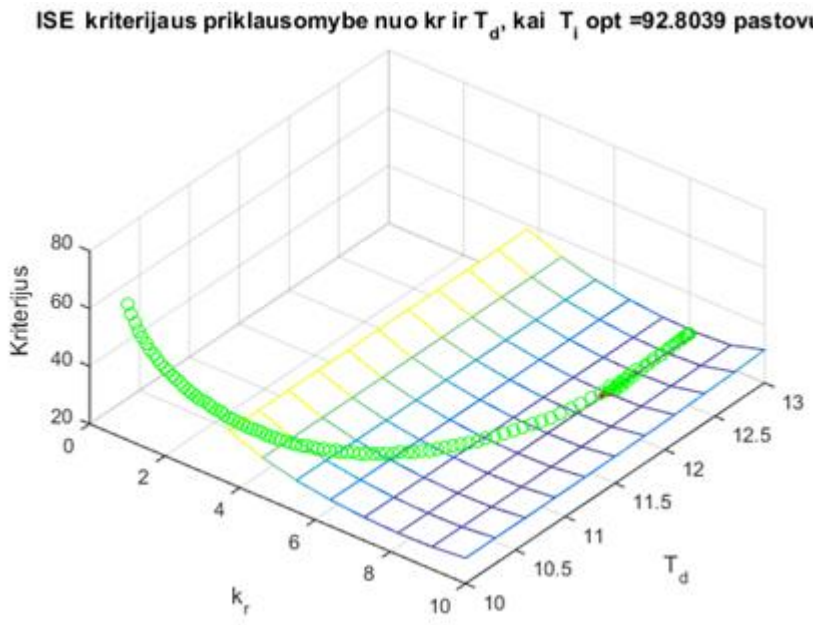
34 pav. Minimalaus taško radimas naudojant Gradientinį greičiausio nusileidimo opt. metodą

35 pav. atvaizduota ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i laiko dedamosios ir stiprinimo k_r, kai diferencialinė laiko dedamoji yra pastovus dydis, kurios vertė T_d=12,0731.



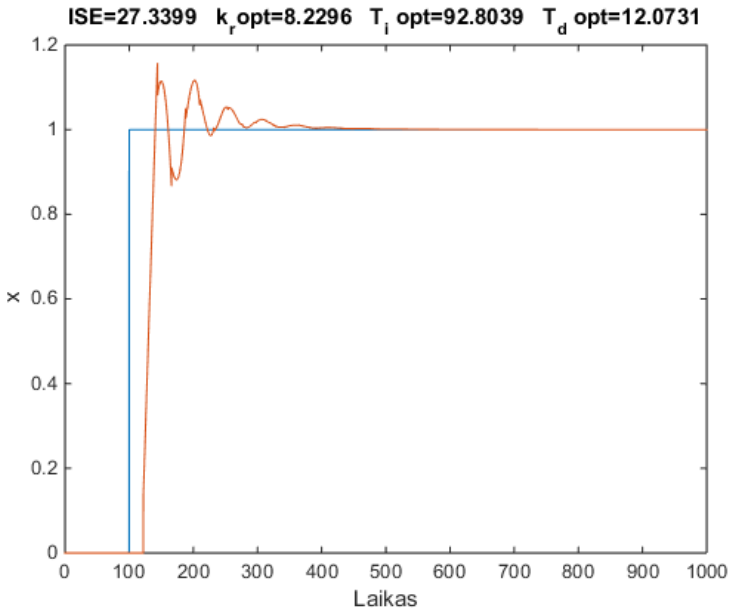
35 pav. Minimalaus taško radimas naudojant Gradientinį greičiausio nusileidimo opt. metodą

36 pav. atidėta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus diferencialinės laiko dedamosios T_d ir stiprinimo k_r, kai integralinė laiko dedamoji yra pastovus dydis, kurios vertė T_i=92,8039.



36 pav. Minimalaus taško radimas naudojant Gradientinį greičiausio nusileidimo opt. metodą

37 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė esant reguliatoriaus parametrams: k_r*, T_i*, T_d*.



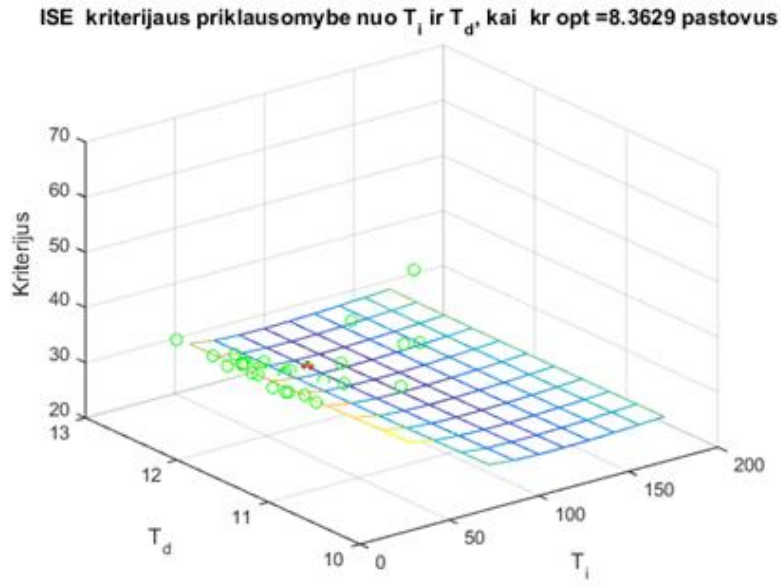
37 pav. Pereinamojo proceso reakcijos kreivė, su gautais geriausiais reguliatoriaus parametrais

Chemotaxis optimizavimo metodas

Atliktas Chemotaxis optimizavimo metodo taikymas PID reguliatoriaus parametrams rasti pagal pasirinktą kriterijų (ISE). Parinktas PID reguliatoriaus parametrų „mutacijos laipsnis“ Δ = *[1 5 1]*. Gauti rezultatai parodyti 38, 39, 40 paveiksluose. Minimumo taškas pažymėtas raudona

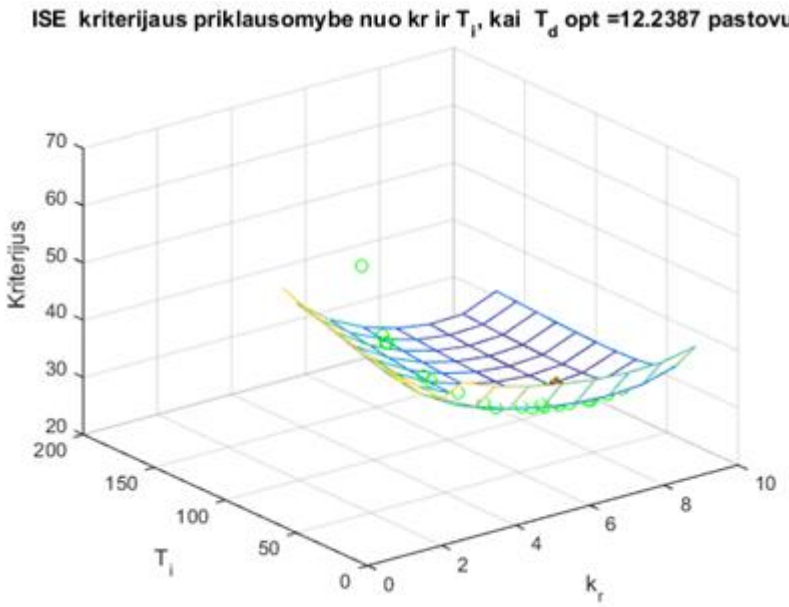
žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimalios kriterijaus vertės link. Reguliatoriaus parametrų vertės minimumo taške: k_r*=8,36, T_i*=84,99, T_d*=12,23, o kriterijaus vertė ISE= 27,4.

38 pav. parodyta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i ir diferencialinės T_d laiko dedamosios, kai reguliatoriaus stiprinimas k_r yra pastovus dydis, kurio vertė k_r=8,3629.



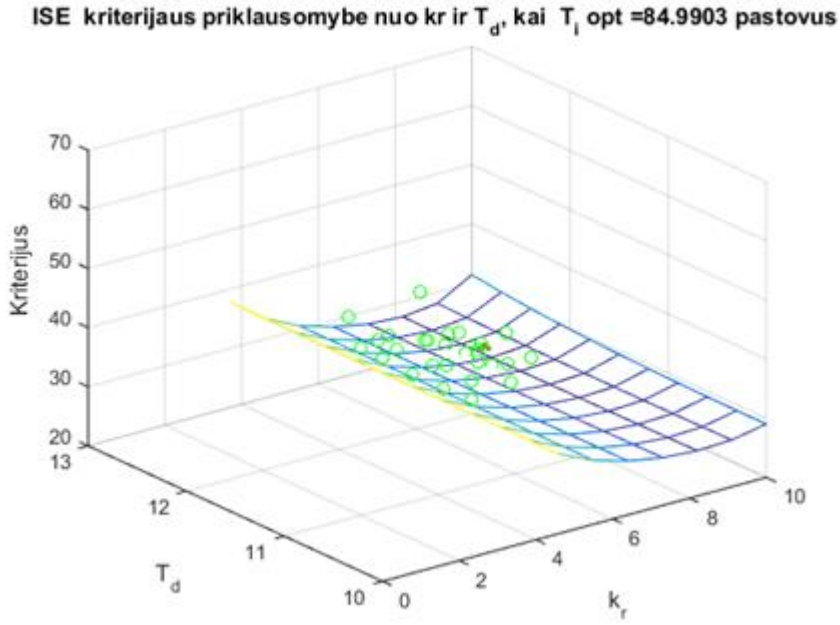
38 pav. Minimalaus taško radimas naudojant Chemotaxis opt. metodą

39 pav. atvaizduota ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i laiko dedamosios ir stiprinimo k_r, kai diferencialinė laiko dedamosioji yra pastovus dydis, kurios vertė T_d=12,2387.



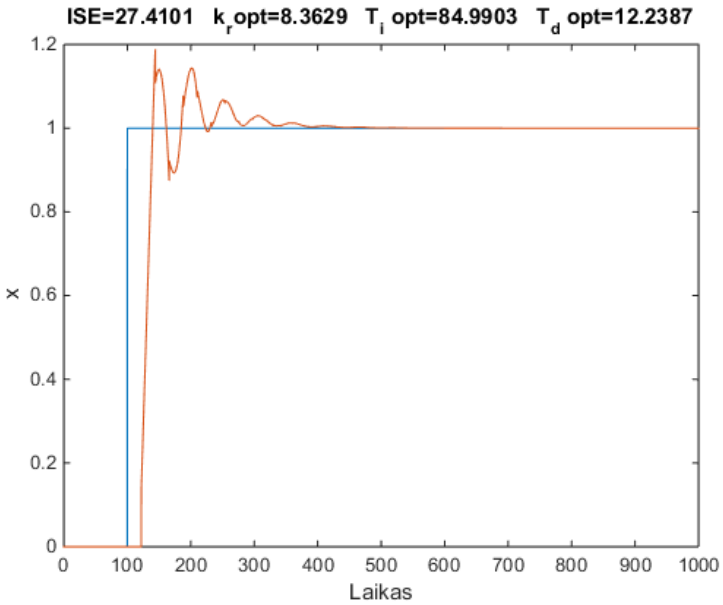
39 pav. Minimalaus taško radimas naudojant Chemotaxis opt. metodą

40 pav. atidėta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus diferencialinės laiko dedamosios T_d ir stiprinimo k_r, kai integralinė laiko dedamoji yra pastovus dydis, kurios vertė T_i=84,9903.



40 pav. Minimalaus taško radimas naudojant Chemotaxis opt. metodą

41 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė esant reguliatoriaus parametrams: k_r*, T_i*, T_d*.



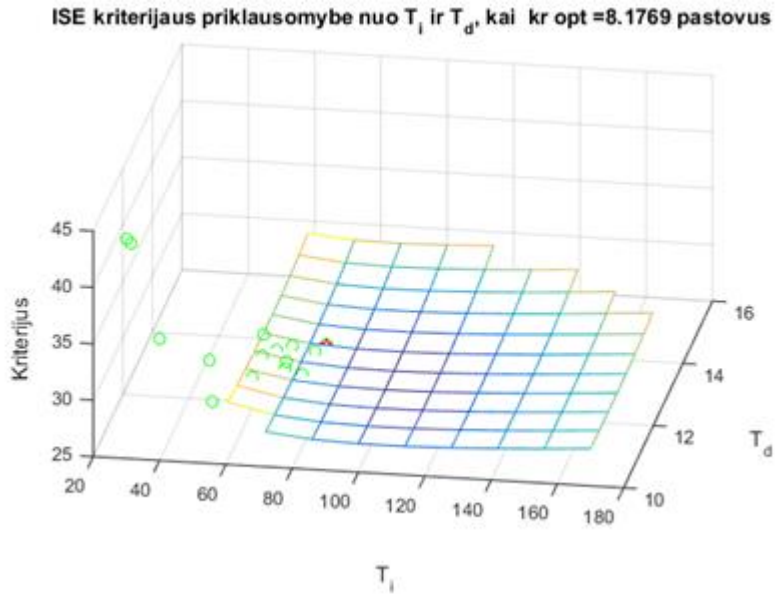
41 pav. Pereinamojo proceso reakcijos kreivė, su gautais geriausiais PID parametrais

Evoliucinio programavimo optimizavimo metodas

Atliktas Evoliucinio programavimo optimizavimo metodo taikymas PID reguliatoriaus parametrams rasti pagal pasirinktą kriterijų (ISE). Parinktas PID reguliatoriaus parametrų „mutacijos laipsnis“ $\Delta = [1\ \ 5\ 1]$, vektoriaus “tėvų” skaičius $p = 2$. Rezultatai parodyti 42, 43, 44

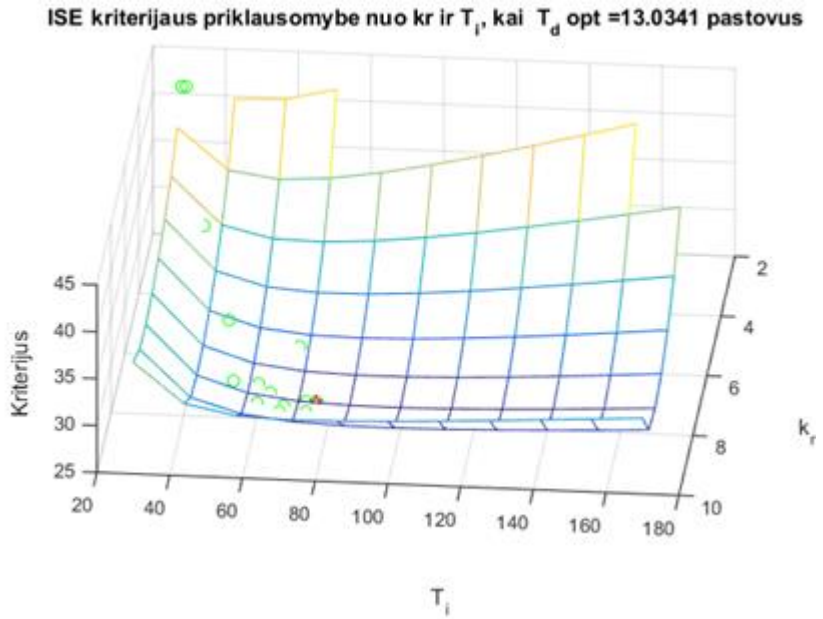
pav. Minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimalaus kriterijaus vertės link. Reguliatoriaus parametrų vertės minimumo taške: k_r*=8,1769, T_i*=76,86, T_d*=13,03, o kriterijaus vertė ISE= 27,5.

42 pav. parodyta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i ir diferencialinės T_d laiko dedamosios, kai reguliatoriaus stiprinimas k_r yra pastovus dydis, kurio vertė k_r=8,1769.



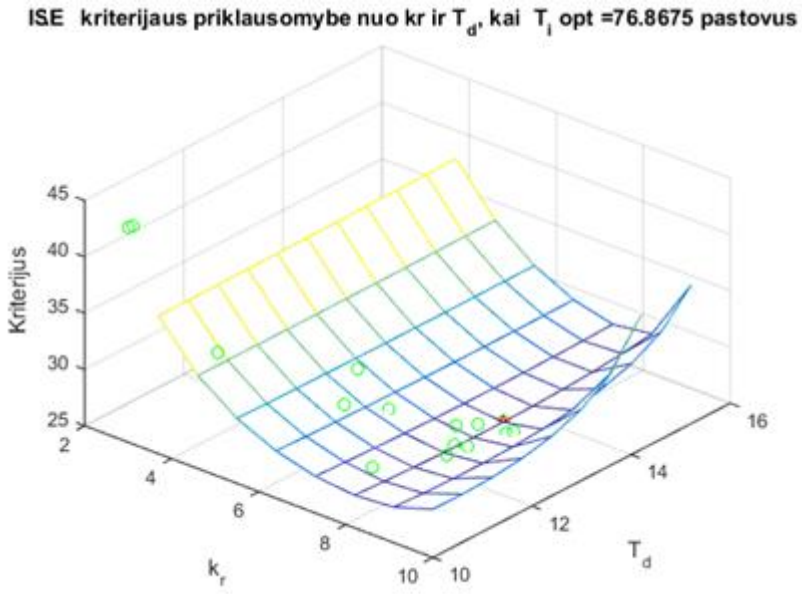
42 pav. Minimalaus taško radimas naudojant Evoliucinio programavimo opt. metodą

43 pav. atvaizduota ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i laiko dedamosios ir stiprinimo k_r, kai diferencialinė laiko dedamoji yra pastovus dydis, kurios vertė T_d=13,0341.



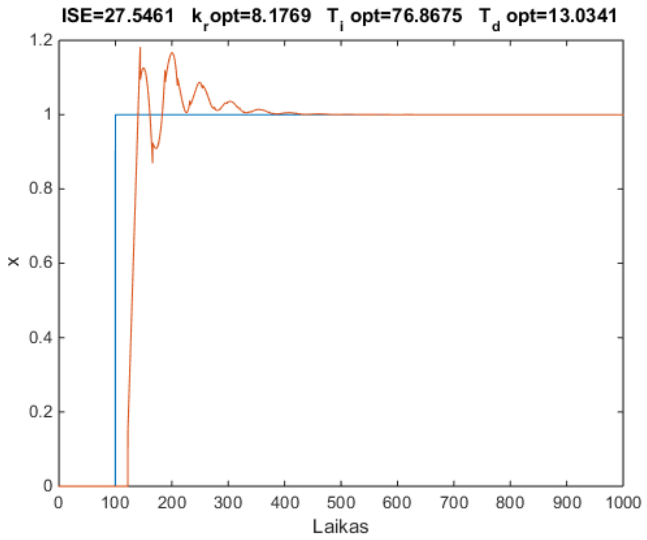
43 pav. Minimalaus taško radimas naudojant Evoliucinio programavimo opt. metodą

44 pav. atidėta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus diferencialinės laiko dedamosios T_d ir stiprinimo k_r, kai integralinė laiko dedamoji yra pastovus dydis, kurios vertė T_i=76,8675.



44 pav. Minimalaus taško radimas naudojant Evoliucinio programavimo opt. metodą

45 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė esant reguliatoriaus parametrams: k_r*, T_i*, T_d*.



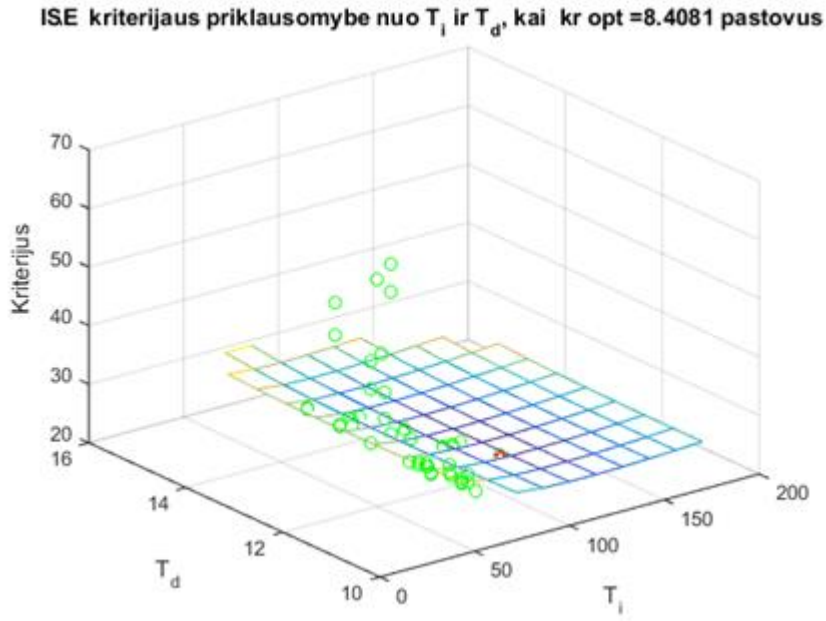
45 pav. Pereinamojo proceso reakcijos kreivė, su gautais geriausiais PID parametrais

Imituojamo atkaitinimo optimizavimo metodas

Atliktas Imituojamo Grūdinimo optimizavimo metodo taikymas PID reguliatoriaus parametrams rasti pagal pasirinktą kriterijų (ISE). Parinktas PID reguliatoriaus parametrų paieškos žingsnis λ = *[1 5 1]*. Rezultatai parodyti 46, 47, 48 pav. Minimumo taškas pažymėtas raudona žvaigždute, o žaliais simboliais pažymėtas algoritmo judėjimo kelias minimalaus kriterijaus vertės

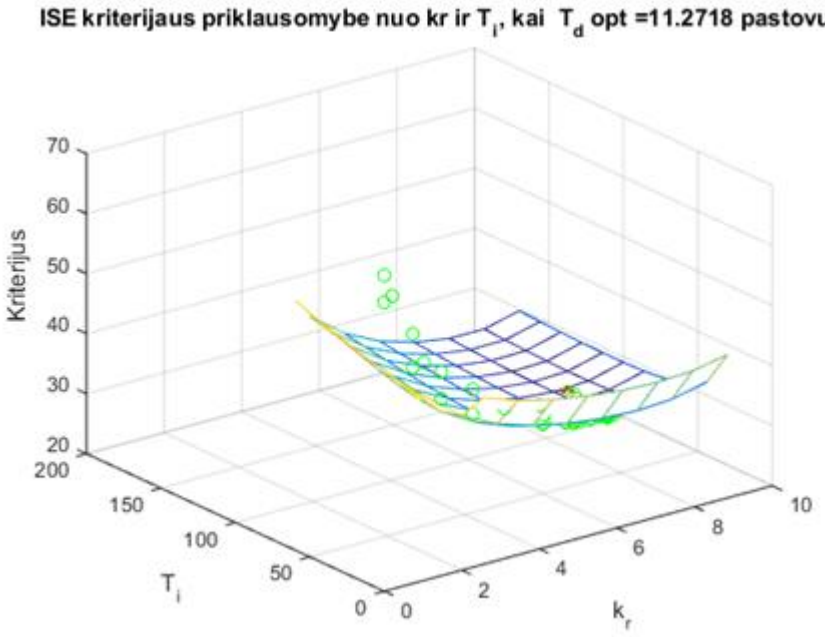
link. Reguliatoriaus parametrų vertės minimumo taške: kr*=8,408, Ti*=96,16, Td*=11,272, o kriterijaus vertė ISE= 27,347.

46 pav. parodyta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i ir diferencialinės T_d laiko dedamosios, kai reguliatoriaus stiprinimas k_r yra pastovus dydis, kurio vertė k_r=8,4081.



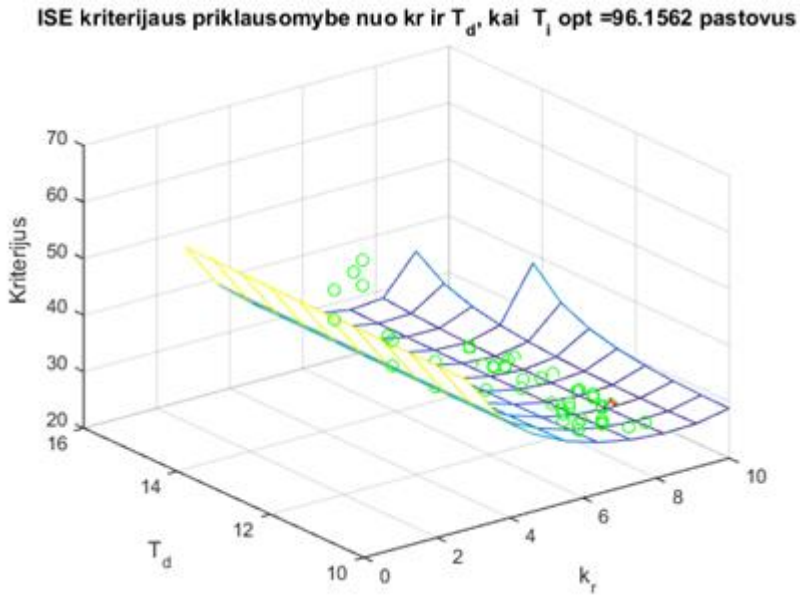
46 pav. Minimalaus taško radimas naudojant Imituojamo grūdinimo opt. metodą

47 pav. atvaizduota ISE kriterijaus vertės priklausomybė nuo reguliatoriaus integralinės T_i laiko dedamosios ir stiprinimo k_r, kai diferencialinė laiko dedamoji yra pastovus dydis, kurios vertė T_d=11,2718.



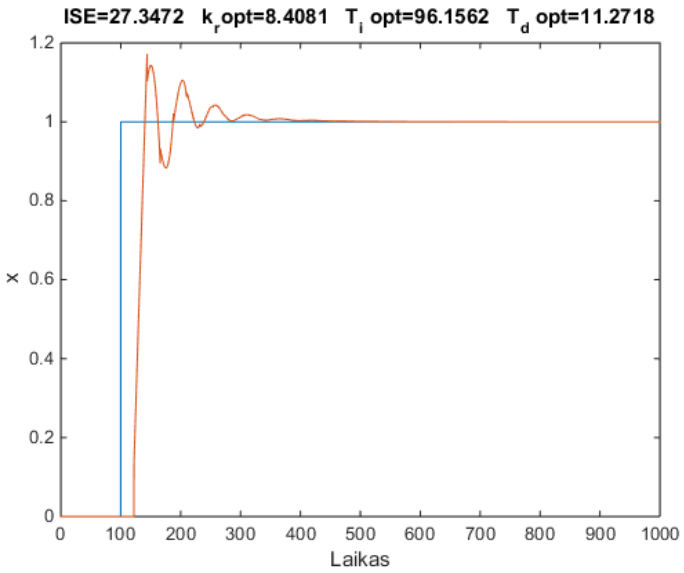
47 pav. Minimalaus taško radimas naudojant Imituojamo grūdinimo opt. metodą

48 pav. atidėta ISE kriterijaus vertės priklausomybė nuo reguliatoriaus diferencialinės laiko dedamosios T_d ir stiprinimo k_r, kai integralinė laiko dedamoji yra pastovus dydis, kurios vertė T_i=96,1562.



48 pav. Minimalaus taško radimas naudojant Imituojamo grūdinimo opt. metodą

49 paveikslėlyje pavaizduota pereinamojo proceso reakcijos kreivė esant reguliatoriaus parametrams: k_r*, T_i*, T_d*.



49 pav. Pereinamojo proceso reakcijos kreivė, su gautais geriausiais PID parametrais

PID reguliatoriaus gautų geriausių parametrų palyginimas naudojant daugiamačius optimizavimo metodus ir įvairias kriterijaus nustatymo funkcijas

Naudojant ITAE kriterijų paieška buvo pradėta skirtinguose pradiniuose taškuose ir analizuojama kaip ekstremumo radimas priklauso nuo pradinių verčių parinkimo. Rezultatai

surašyti 4 lentelėje. Beveik visais optimizavimo metodais surastas tas pats ekstremumo taškas, kai pradinės vertės iš [1 30 10] į [10 100 10]. Skyrėsi Gradientiniu greičiausio nusileidimo metodu surastas ekstremumas.

4 lentelė. ITAE kriterijaus vertė ir reguliatoriaus parametrai, gauti naudojant įvairius daugiamačius metodus

Pradinis taškas [kr Ti Td]	Naudojamas kriterijus	Optinizavimo metodas	Gauta minimali Kriterijaus vertė	Gauti PID parametrai
[1 30 10]	ITAE	Gauso-Zaidelio	ITAE = 737,5	kr = 7, Ti = 100, Td = 7,7
[10 100 10]	ITAE	Gauso-Zaidelio	ITAE = 737,5	kr = 7, Ti = 100, Td = 7,7
[1 30 10]	ITAE	„Chemotaxis“	ITAE = 761,8	kr = 6,9, Ti = 108,2, Td = 10,1
[10 100 10]	ITAE	„Chemotaxis“	ITAE = 763,8	kr = 7,1, Ti = 108,9, Td = 10,2
[1 30 10]	ITAE	Imituojamo atkaitinimo	ITAE = 714,94	kr = 6,53, Ti = 99,15 Td = 8,0
[10 100 10]	ITAE	Imituojamo atkaitinimo	ITAE = 705,5	kr = 6,69, Ti = 100,4, Td = 8,0
[1 30 10]	ITAE	Gradientinis	ITAE = 1629	kr = 9,1686, Ti = 52,748, Td = 7,92
[10 100 10]	ITAE	Gradientinis	ITAE = 702,5	kr = 6,85, Ti = 103,5, Td = 8,9
[1 30 10]	ITAE	Evoliucinio programavimo	ITAE = 715	kr = 6,75, Ti = 99,4, Td = 7,6
[10 100 10]	ITAE	Evoliucinio programavimo	ITAE = 706	kr = 6,78, Ti = 101, Td = 8,2

IŠVADOS

1. Apžvelgti ir atrinkti šie tipiniai optimizavimo metodai: Nuoseklioji peržiūra, Dichotomijos metodas, Auksinio pjūvio metodas, Gauso-Zaidelio, Gradientinis greičiausio nusileidimo, Chemotaxis, Evoliucinio programavimo, Imituojamo grūdinimo. Sukurtos MATLAB/Simulink programos atrinktiems tipiniams optimizavimo metodams.
2. Vienmačiai optimizavimo metodai taikyti matematinių funkcijų tyrimui, esant unimodinei ir multimodinei tikslo funkcijai. Šiais metodais surandama ta pati unimodinės funkcijos $f(x)=x^2+10$ ekstremumo vertė: *x*=0, f(x*)=10*. Analizuojant multimodinę funkciją $f(x)=x\cdot\left(\sin x\cdot\left(1+\cos x\right)\right)$, Nuosekliosios paieškos metodu rastas globalus minimumas: *x*=17,85, f(x*)=-23,13*, o Dichotomijos ir Auksinio pjūvio metodu – lokalus: *x*=11,55, f(x*)=-14,99*.
3. Daugiamačiai optimizavimo metodai taikyti matematinių funkcijų tyrimui. Analizuotos dvi funkcijos: Rosenbroko „banana“ $f(x,y)=\left(1-x\right)^2+100\cdot\left(y-x^2\right)^2$ ir paraboloidas $f(x,y)=x^2+y^2$. Rosenbroko „banana“ funkcijos ekstremumo vertė tiksliausiai surandama Nuosekliosios paieškos metodu (*x*=1, y*=1, f(x*,y*)=0*) o paraboloido funkcijos – Nuosekliosios paieškos ir Gauso-Zaidelio metodais (*x*=0, y*=0, f(x*,y*)=0*).
4. Optimizavimo metodai taikyti ARS reguliatoriaus parametrams rasti, naudojant ISE ir ITAE kriterijus. Ieškant ITAE kriterijaus buvo tirta, kaip pradinių verčių pasirinkimas lemia ekstremumo vertės suradimą – Gradientiniu greičiausio nusileidimo metodu rasti skirtingi ekstremumai paiešką pradedant skirtinguose taškuose, kitais metodais rastas tas pats ekstremumas.
5. Sukurtas MATLAB/Simulink programų paketas gali būti taikomas mokymosi tikslams nepatyrusiam vartotojui, nes optimizavimo algoritmai supaprastintai realizuoti MATLAB/Simulink aplinkoje ir programos kode parašyta daug komentarų, padedančių suprasti metodų veikimą.

LITERATŪROS SĄRAŠAS

[1] TEKORIUS, Tomas, KILDIŠAS, Valeras, Optimizavimo pagrindai: mokomoji knyga, Kaunas: Technologija, 2007. 70 p. ISBN: 9789955252856.

[2] APYNIS, A, Optimizavimo metodai: mokomoji knyga, Vilnius: Vilniaus universiteto leidykla, 2005.

[3] ANTONIOU, A., Wu-Sheng Lu, Practical Optimization, Algorithms and Engineering Applications, Canada: Springer, 2007.

[4] KALANTA, S., Taikomosios optimizacijos pagrindai; Tiesinių uždavinių formulavimas ir sprendimo metodai: mokomoji knyga, Vilnius: Technika, 2003.

[5] MOCKUS, J., Optimizavimas ir taikymas: paskaitų konspektas [interaktyvus]. [žiūrėta 2014 m. 04 21d.] Prieiga internete http://proin.ktu.lt/~mockus/docj/distgt.pdf.

[6] HASSIN, R., HENIG, M. „Monoticity and Efficient computation of optimal dichotomous search," *Discrete Applied Mathematics 46,* pp. 221-234, 1993.

[7] ABOUTANIOS, E., „Modified Dichotomous Search Frequency Estimator," *IEEE Signal processing letter,* t. Vol. 11, nr. NO. 2, 2004.

[8] CAI, J., HAN D., CHEN C., CHEN S., „Application of the golden section search algorithm in the nonlinear isoconversional calculations to the determination of the activation energy from nonisothermal konetic conversion data," *Solid State Sciences,* t. 12, pp. 829-833, 2010.

[9] TSAI C.H., KOLIBAL J., LI M., „The golden section search algortihm for finding a good shape parameter for meshless collocation methods," *Engineering Analysis with Boundary Elements ,* t. 34, pp. 738-746, 2010.

[10] TSITSIKLIS J. N., „A Comparison of Jacobi and Gauss-Seidel Parallel Iterations," *Appl. Math. Lett. Vol.2,* t. 2, pp. 167-170, 1989.

[11] ŽILINSKAS A., Matematinis programavimas, 2005 [interaktyvus]. Prieiga per internetą: https://docs.google.com/viewer?docex=1&url=http://www.mii.lt/antanas/uploads/MathematicalProgramming.pdf

[12] SUMAN B., „Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem," *Computers and Chemical Engineering,* t. 28, pp. 1849-1871, 2004.

[13] HANKE M., LI P., „Simulated annealing for the optimization of batch distilation processes,“ *Computers and Chemical Engineering*, t. 24, pp. 1-8, 2000.

[14] KIRKPATRICK S.; C. D. Gelatt; M. P. Vecchi, „Optimization by Simulated Annealing,“ *Science*, t. 220, pp. 671-681, 1983.

[15] FABER R., JOCKENHOVEL T., TSATSARONIS G., „Dynamic optimization with simulated annealing,“ *Computers and Chamical Engineering*, t. 29, pp. 273-290, 2005.

[16] FEEHERY W.F., Dynamic optimization with path constraints. Disertation., USA: Masachusetts Institute of Technology, 1998.

[17] MULLER S.D., MARCHETTO J.M., AIRAGHI S., KOUMOUTSAKOS P., „Optimization Based on Bacterial Chemotaxis,“ *IEEE TRANSACTIONS ON EVOLIUTIONARY COMPUTATION VOL.6*, pp. 16-29, 2002.

[18] LI B., JIANG W., „A Novel Stochastic Optimization Algorithm,“ *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, VOL. 30*,, pp. 193-198, 2000.

[19] ZITZLER E., Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications: Disertation., Zurich: Swiss Federal Insitute of Technology, 1999.

[20] Internetinis išteklius [žiūrėta 2015 04 28]: http://se.mathworks.com/products/matlab/v.

[21] GALVANAUSKAS V., LEVIŠAUSKAS D. Biotechnologinių procesų modeliavimas, optimizavimas ir valdymas. Praktikumas ir uždaviniai: mokomoji knyga. Vilniaus pedagoginio universiteto leidykla, 2008, 112 p. ISBN 978-609-02-0377-4.

PRIEDAI

1 priedas

MatlabSimulink pagrindinė programa vienmačių optimizavimo algoritmų paleidimui:

```
% % Paprastu optimizavimo algoritmu pagrindine programa
% %
% % (c) Aurimas Gajauskas
% % Si  programa taikoma paprastiems optimizavimo metodams analizuoti:
% % Nuoseklioji perziura(1 ir 2 kintamuju), Dichotomijos ir Auksinio pjuvio
metodai
% % Tiklso funkcija pasirenkama ar irasoma faile: TiksloFunkcija1Kint.m, ar
% % TiksloFunkcija2Kint.m atitinkamamai jei ji priklauso nuo 1 arba 2
kintamuju.

function main
clear all, clc, close all
% Optimizavimo metodas pasirenkamas  irasius metodo pavadinima
metodas = 'DM' % 'AP'- Auksinio pjuvio metodas, 'DM'- Dichotomijos metodas,
% 'NP1kint', 'NP2kint' atitinkamai nuoseklioji perziura 1 ir 2 kitnamuju.

% Pasirenkami  Tiklso funkcijos kintamuju x leistinoji sritis [a b]
a = [-20 ]; % kintamuju leistina minimali reiksme a= [x1min, x2min ... xnmin
]
b = [20]; % kintamuju leistina maksimali reiksme b= [x1max, x2max ... xnmax ]
e = [0.5 ];% kintamuju keitimo zingsnis e= [e1, e2 ... en ], priklauso nuo
% nepriklausomu kintamuju x skaiciaus.


[Xmin,Ymin] = AGoptim(metodas,a,b,e) % atkomentuojamas jei iskomas
% 1 kintamojo tikslo funkcijos ekstremumas

%[Xmin,Zmin] = AGoptim(metodas,a,b,e) % atkomentuojamas jei iskomas
%2 kintamuju tikslo funkcijos ekstremumas

% Braižomas gafikas ir pazymimas  gautas ekstremumo taškas
% Jei tikslo funkcija vieno kintamojo
if length(a) == 1
    x = a:e:b;
    y = TiksloFunkcija1Kint(x); % Apskaièiuojama tiklos funkcijos vertës

    ycut=y;
    % ycut(find(ycut > mean(mean(y)))) = NaN;
    plot(x(:),ycut(:),'-b',Xmin,Ymin,'ro'); % Braizomas grafikas ir pazymimas
    %minimumo taskas
    xlabel('x')
    ylabel('f(x)')


else
    % Jei tikslo funkcija priklauso nuo 2 kintamuju
    [X,Y] = meshgrid(a(1):e(1):b(1),a(2):e(2):b(2)); % suformuojamas
    %kintamuju tinklelis
    Z = TiksloFunkcija2Kint(X,Y);% randamos tikslo funkcijos reiksmes
    %kiekviename suformuoto tinklelio taske
    Zcut = Z;
    Zcut(find(Zcut > mean(mean(Z)))) = NaN;
    mesh(X,Y,Z) % braizomas grafikas
    hold on
```

```
    plot3(Xmin(1), Xmin(2), Zmin,'rp') % pazymimas raudonai gautas minimumo
taskas
    % grafike
    hold off
    xlabel('x')
    ylabel('y')
    zlabel('Tikslo funkcijos verte')

end
```

2 priedas

MatlabSimulink programa vartotojo tikslo funkcijai nuo vieno kintamojo įrašyti :

```
%%
%% (c) Aurimas Gajauskas
%% Vartotojo Tikslo funkcija nuo vieno kintamojo
% yra keli pavyzdziai taip pat galima isirasyti ir kitokia tikslo funkcija
function y = TiksloFunkcija1Kint(x)

% Unomodine tikslo funkcija
y = 10+4*x.^2+x*20;
%y = x.^2+10;

% Multimodine tikslo funkcija
%y = x.*(sin(x).*(1+cos(x)));
```

3 priedas

MatlabSimulink programa vartotojo tikslo funkcijai nuo dviejų kintamųjų įrašyti :

```
%%
%% (c) Aurimas Gajauskas
% Vartotojo Tikslo funkcija nuo 2 nepriklausomu kintamuju
% yra keli pavyzdziai taip pat galima isirasyti ir kitokia tikslo funkcija
function z = TiksloFunkcija2Kint(x,y)

%Rosenbrock's banana funkcija
z=(1-x).^2+100*(y-x.^2).^2;

% paraboloido funkcija
%z=x.^2+3*y.^2;
```

4 priedas

MatlabSimulink programa Auksinio pjūvio metodo realizavimui:

```
%%
%% (c) Aurimas Gajauskas
%  Auksinio pjuvio metodas
% Auksinio pjuvio metodu leistinojoje srityje apskaiciuojami 2 taskai ir
% juose aspkaiciuojama tikslo fukcijos reiksme, toliau leistinoji sritis
% yra trumpinama, kol tenkimama salyga b-a < e

function [Xmin,Ymin] = AGoptfun_AUKSINIO_PJUVIO(a,b,e);

K = 1.618034; % auksinio pjûvio konstanta, zinoma kaip auksinis santykis
```

```
while b-a >  e
    % randami 2 taskai: x(1) ir x(2), kurie leistinaja sriti dalina i 3 dalis
    x(2) = a + (b - a) /K ;
    x(1) = b -  (b - a) /K;
    % randamos tikslo funkcijos vertes tuose taskuose
    Y = TiksloFunkcija1Kint(x(:));
    % toliau pereinama i intervala, kuriame tiklso funkcijos verte yra
    % maziausia
    if Y(1) < Y(2)
        % Intervalas sumazinamas nuo a iki x(2)
        a = a;
        b = x(2);
    end

    if Y(1) > Y(2)
        % Intervalas sumazinamas nuo x(1) iki b
        a = x(1);
        b = b;
    end

    if Y(1) == Y(2)
        % Intervalas sumazinamas nuo x(1) iki x(2)
        a = x(1);
        b = x(2);
    end
    figure(11)
    plot(x(1),Y(1),'gp')
    hold on
    plot(x(2),Y(2),'gp')
    hold on
end
% Skaiciavimai baigiami, kai kintamojo sritis yra mazesne uz e
%Gaunamos tokios ekstemumo kordinates
Ymin = (Y(2)+Y(1))/2;
Xmin = (b+a)/2;
```

5 priedas

MatlabSimulink programa Dichotomijos optimizavimo metodo realizavimui:

```
%%
%% (c) Aurimas Gajauskas
%%  Dishotomijos metodo algoritmas
%% Siuo metodu kintamojo x leistinoji sritis yra padalinama i 3 lygias dalis
%% ir kiekviename is ju yra paskaiciuojama tiklso funkcijos verte,
%% toliau yra trumpinama leistinoji sritis [a, b] pereinant  i intervala
%% kuriame tiklso funkcijos verte yra maziausia

function [Xmin,Ymin] = AGoptfun_DICHOTOMIJOS(a,b,e)


while b-a >  e
    % Randami 3 taskai kurie leistinaja sriti dalina i 4 lygias dalis
    % ir apskaciuojamos tikso funkcijos reiksmes tuose taskuose
    x(1) = a + (b - a) * 0.25;
    x(2) = a + (b - a) * 0.5;
    x(3) = a + (b - a) * 0.75;
    Y = TiksloFunkcija1Kint(x(:));
    figure(11)
    plot(x(1),Y(1),'gp')
```

```
    hold on
    plot(x(2),Y(2),'gp')
    hold on
    plot(x(3),Y(3),'gp')
    hold on
    % randama minimali tiklso tunkicjos verte apskaiciuotuose taskuose
    Ymin = min(Y);

    % pereinama i intervala kuriame tiklso funkcijos verte maziausia
    if Y(1) == Ymin
        a = a;
        b = x(2);
    end

    if Y(2) == Ymin
        a = x(1);
        b = x(3);
    end

    if Y(3) == Ymin
        a = x(2);
        b = b;
    end
end
% algoritmo stabdomas kai sritis b-a yra mazesne uz e
% priskiriamos ekstremumo tasko vertes
Ymin = min(Y);
Xmin = (b+a)/2;
```

6 priedas

MatlabSimulink programa vieno kintamojo nuosekliosios paieškos realizavimui:

```
%% (c) Aurimas Gajauskas
%% Vieno kintamojo nuoseklioji paieska
% leistinoji sritis [a b] dalinama i lygias dalis tikslumu e ir
% apskaiciuojamos kiekviename taske tikslo funkcijos verte
%  ir is gautu verciu isrenkama maziausia tikslo funkcijos verte

function  [Xmin,Ymin]=AGoptfun_Nuoseklioji_1kint(a,b,e);
X = a:e:b; % leistinoji sritis padalinam i 5 vienodo ilgio intervalus
Y = TiksloFunkcija1Kint(X); % randama tikslo funkcjos verte tuose intevaluose
[Ymin,Imin] = min(Y); % randama minimali tikso funkcjos verte ir jos vieta
matricoje
Xmin = X(Imin); % radama  kintamojo x verte
```

7 priedas

MatlabSimulink programa dviejų kintamųjų nuosekliosios paieškos realizavimui:

```
%  Dvieju kintamuju nuoseklioji paieska
% (c) Aurimas Gajauskas
% Nuoseklioji paieska (2 kintamuju) atliekama padalinus kintamuju
% leistinaja sriti i lygias dalis ir apskaiciuojamos funkcijos vertes
% visuose x ir y vektoriu kombinacijose. Pvz jei x ir y  parametrai yra
% padalinami i 10 tasku, tai funkcijos verte yra paskaiciuojama 100
% tasku(10*10)
```

```
function [Xmin,Zmin]=AGoptfun_Nuoseklioji_2kint(a,b,e);

[X1,Y1] = meshgrid(a(1):e(1):b(1),a(2):e(2):b(2)); % Sudaromas kintamuju x1
% ir y1 tinklelis

z_min = +inf; x_min = NaN; y_min = NaN; % priskiriamos pradines reiksmes
for i = 1:size(X1,1)
    for j = 1:size(X1,2)
        x = X1(i,j); y = Y1(i,j);
        z = TiksloFunkcija2Kint(x,y);% apskaiciuojama tikslo funkcijos
reiksme
        %kiekviename sudaryto tinklelio taske
       % Toliau yra palyginama ar gauta funkcijos verte yra mazesne nei
       % seniau gauta maziausia funkcijos verte, jei salygos tenkinamos
       % tai priskiriamos naujosios reiksmes kaip ekstremumo tasko reiksmes
        if z < z_min
            x_min = x;
            y_min = y;
            z_min = z;
        end
```

8 priedas

MatlabSimulink pagrindinė programa daugiamačių optimizavimo algoritmų paleidimui :

```
%% Daugelio kintamuju tikslo funkcijos optimizavimo pagrindine programa
%
%% (c) Aurimas Gajauskas
% Si progama skirta susipazinimui su daugiamaciais optimizavimo
% metodais: Chemotaxis, Evoliucinio programavimo,
% Imituojamo atkaitinimo "Simulated annealing", Gradientinio geiciausio
% nusileidimo, Gauso- Zaidelio opt metodais.

function main
clear all, clc , close all

% Optimizavimo metodas pasirenkamas % irasius metodo pavadinima
metodas = 'Chemotaxis' % Pasirenkamas vienas is optimizavimo metodu:
%'GausoZaidelio'- Gauso- Zaidelio opt. metodas, 'Gradientinis' -Gradientinis
% opt. metodas, 'Simulated-Annealing' - Simulated Annealing (Imituojamo
atkaitinimo) optimizavimo
% metodas, 'Evoliucinis' - Evoliucinio programavimo metodas, 'Chemotaxis'-
Chemotaxis
%optimizavimo metodas.

x0 = [ -20, -20 ];        % pradines parametru x1 x2 ... xn vertes

% apribojimai tikslo funkcijos parametrams x:
a = [-50, -30];  % funkcijos parametru x apribojimai a(i) < x(i)
b = [60, 20];            % funkcijos parametru x apribojimai  x(i) < b(i)
% apribojimu skaicius turi buti lygus kintamuju skaiciui
eps = 0.000001;             %  Randamas ekstremumo taskas, kai  tikslo
%funkcijos pokytis tarp iteraciju yra mazesnis uz e  .

% parametrai reikalingi Gauso- Zeidelio paieskos metodui
e = [0.5, 0.5];  % parametru x kitimo zingsnis
% parametru skacius turi buti lygus nepriklausomu kintamuju skaiciui
%pasirinkus Gauso-Zaidelio opt. metoda reikia atkomentuoti apatine eilute
%[Xmin,Ymin] = AGoptim1(metodas, a, b, eps, x0, e )

% Parametrai reikalingi Gradientiniam paieskos metodui
```

```
beta= 10^-6; % mazas optimizuojamu kintamuju nuokrypos
alfa = 1; % fiksuotas paieskos zingsnis
%pasirinkus Gradientini greiciausio nusileidimo opt. metoda reikia
%atkomentuoti apatine eilute
%[Xmin,Ymin] = AGoptim2(metodas, a, b, eps, x0, beta, alfa)


% Parametrai reikalingi Chemotaxis paieskos metodui
deltax = [1, 1 ]; % parametrai nustatantys 'mutaciijos laipsni'
% parametru skacius turi buti lygus nepriklausomu kintamuju skaiciui
%pasirinkus Chemotaxis opt. metoda reikia atkomentuoti apatine eilute
[Xmin,Ymin] = AGoptim3(metodas, x0, a, b, deltax)


% Parametrai reikalingi Evoliucinio programavimo paieskos metodui
p = 5;  %  vektoriu tevu skaicius
deltaxe = [0.025, 0.025 ]; % parametrai nustatantys 'mutacijos laipsni'
% parametru skacius turi buti lygus nepriklausomu kintamuju skaiciui
%pasirinkus Evoliucinio opt. metoda reikia atkomentuoti apatine eilute
% [Xmin,Ymin] = AGoptim4(metodas, x0, deltaxe, a, b, p, eps)


% Parametrai reikalingi Simulated Annealing paieskos metodui
lemda = [0.5, 0.5 ];% paieskos zingsnis
% parametru skacius turi buti lygus nepriklausomu kintamuju skaiciui
%pasirinkus Imituojamo atkaitinimo opt. metoda reikia atkomentuoti apatine
eilute
%[Xmin,Ymin] = AGoptim5(metodas, x0, lemda, a, b, eps)

% Braizomas grafikas jei tikslo funkcija priklauso nuo 2 nepriklausomu
% kintamuju, kitu atveju gaunama tik tikslo funkcijos minimumo taskas ir
% kintamuju x vertes
if length(x0)==2
    [X,Y] = meshgrid(a(1):(b(1)-a(1))/10:b(1),  a(2):(b(2)-a(2))/10:b(2));
    for i = 1:size(X,1)
        for j = 1:size(X,2)
            x(1)=X(i,j);
            x(2)=Y(i,j);

            Z=TikslоFunkcijaDK(x);
            ZA(i,j)=Z;
        end
    end
    figure(11)
    mesh(X,Y,ZA);
    xlabel('x')
    ylabel('y')
    zlabel('Funkcijos verte')

    hold on;

    plot3(Xmin(1), Xmin(2), Ymin,'rp','MarkerSize', 15)
    hold off

end
```

9 priedas

MatlabSimulink programa vartotojo tikslo funkcijai nuo daugelio kintamųjų įrašyti :

```
%%
% (c) Aurimas Gajauskas
% Vartotojo Tikslo funkcija
function f = TiksloFunkcijaDK(x)
%f = 100- (10-x(1))^2+(5-x(2))^2;

%Rosenbrock function
%f=(1-x(1)).^2+100*(x(2)-x(1).^2).^2;

%paraboloidas
f=x(1).^2+3.*x(2).^2;

%f =(x(1)+10*x(2))+5*(x(3)-x(4))^2+(x(2)-2*x(3))^4+10*(x(1)-x(4))^4;
```

10 priedas

MatlabSimulink programa Chemotaxis optimizavimo algoritmo realizavimui:

```
%%
% % (c) Aurimas Gajauskas
%  Chemotaxis metodas
% Tai paprasciausias atsistiktines paieskos algoritmas.


function [Xmin,Ymin] = AGoptfun_CHEMOTAXIS(x0, a, b, deltax);



x = x0;% Paieskos pradzia
for i = 1:1:10000% maksimalus iteraciju skaicius


    f0 =  TiksloFunkcijaDK(x(:));% Apskaiciuojama tikslo funkcija pradineme
taske x0
    for k = 1:length(x) % Nustatomas tikslo funkcijos kintamuju skaicius
        xx(k) = x(k) + deltax(k)*((rand(1)*2)-1); % Atliekama optimizuojamu
parametru 'mutacija'
    end

    % paskaiciuojama tikslo funkcijos verte su naujais parametrais
    f =  TiksloFunkcijaDK(xx(:));
    if f < f0; % tikrinama salyga ar gauta nauja tikslo funkcijos verte yra
        %geresne nei pries tai buvusi

        % tikrinamos salygos ar gauti nauji kintamieji patenka i intervala
[a, b]
        for  k = 1:length(x)
            if a(k) < xx(k)
                if xx(k) < b(k)
                    % Jei salygos tenkinamos, senosios parametru vertes
pakeiciamos naujai
                    %gautomis parametro vertemis
                    x(k) = xx(k);
```

```
                    if length(a)==2
                        f =  TiksloFunkcijaDK(x(:));
                        figure(11)
                        plot3(x(1),x(2),f,'gp')
                        hold on
                    end
                end
            end
        end
    end
end
Xmin = x; % tikslo funkcijos parametro vertes ekstemumo taske
Ymin = f;% tikslo funkcijos ekstremumo reiksme
```

11 priedas

MatlabSimulink programa Evoliucinio programavimo algoritmo realizavimui:

```
%% (c) Aurimas Gajauskas
%  Evoliucinio programavimo metodas


function [Xmin,Ymin] = AGoptfun_EVOLIUCINIS(x0, deltaxe, a, b, p, eps);

x = x0; % paieskos prdzios taskas
f0 = -TiksloFunkcijaDK(x(:)); % tikslo funkcijos verte pradiniame taske

% sudaromi p tevu vektoriai
xp = x(:)* randn(1,p);

%apskaiciuojamos fikslo funkcijos vertes xp "tevu" vektoriams
for t = 1:p
    fp(t) = -TiksloFunkcijaDK(xp(:,t));
end
% Pradedamas algoritmo vykdimas
for j = 1:1000
    f0 = -TiksloFunkcijaDK(x(:));
    % atliekamos pradines sekos mutacijos(palikuoniu vektoriu sudarymas)
    for i = 1: p
        for k = 1:length(x)
            xc(k,i)=xp(k,i)+deltaxe(k)*randn(1);
            % tikrinama ar gautos kintamuju vertes patenka i intervala [a b]
            % jei nepaneka tai parenkamas arciausiai esanti ribine verte
            if xc(k,i)<a(k)
                xc(k,i)=a(k);
            end
            if xc(k,i)>b(k)
                xc(k,i)=b(k);
            end
        end
    end

    % apskaicuojamos tikslo funkcijos vertes  'vaiku' vektoriams
    for t = 1:p
        fc(t) = -TiksloFunkcijaDK(xc(:,t));
    end

    % pravedamos '' varzybos''  tarp tevu ir palikuoniu matricu geriausiu
    % parametu vektoriu suradimui
```

```
    % sukuriama tuscia vaiku laimejimu matrica gc
    gc = [];
    % sukuriama tuscia tevu laimejimu matrica gk
    gp = [];
    % pradiniai tevu vektoriu laimejimai 0
    gp(p)=0;

    % pradedama laimejimu iteracijos
    % kiekvienas tevu vektorius palyginamas su atsitiktinai atrinktais vaiku
vektoriais
    % parenkamas tevu vektorius k
    for k = 1:p
        % pradiniai vaiku vektoriu laimejimai 0
        gc(p)=0;
           for ka = 1:p
            ra = randi([1 p]) ;

            % atrinktas tevu vektorius palyginamas su atsitiktinai parinktu
vaiku vektoriumi
            % jei tikslo funkcijos verte didesne su pradiniais parametrais,
nei su
            %'vaiku' parametais tai laimi tevu vektorius ir yra jo laimejimu
skaicius padidinams 1
            if  fp(k) > fc(ra)
                gp(k)= gp(k)+1 ;

            end
            % jei tikslo funkcijos verte didesne su 'palikuonio' parametrais,
            %nei su pradiniais parametrais,  tai laimi 'vaiku' vektorius ir
            %yra jo laimejimu skaicius padidinams 1
            if  fp(k) <= fc(ra)
                gc(ra)= gc(ra)+1;
            end
        end
    end

    %  'tevu' ir 'vaiku' parametru matricos ya papildomos viena eilute
kurioje patalpinami ju laimejimu skaicius
    xc(end+1,:)=gc;
    xp(end+1,:)=gp;

    % 'tevu' ir 'vaiku' matricos yra patalpinamos i viena matrica
    g(:,1:p)=xp;
    g(:,p+1:2*p)=xc;
    % surikiuojami parametru vektoriai pagal laimejimu skaiciu nuo mažiausio
    % iki didžiausio
    [h, t]=sort(g(end,:))  ;
    sg=g(:,t);

    % atrenkami dazniausiai laimeje parametru vektoriai tolimesniems
    % skaiciavimams.
    xp=sg(1:length(x),p+1:p*2);
       % toliau 'vaiku' xc matrica yra isvaloma ir paliekama tuscia
    xc=[];
    % Is gautu geriausiu parametru vektoriu xp yra atrenkamas tas su kurio
tikslo
    % funkcijos verte didziausia.
    for t = 1:p
        fp(t) = -TiksloFunkcijaDK(xp(:,t));
    end

    [f1min xmin]= max(fp);
```

```
    xmin=xp(:,xmin)
    fmin = TiksloFunkcijaDK(xmin(:))

    % grafike pavaizduojami algoritmo judejimas minimumo link

    if length(a)==2
    figure(11)
    plot3(xmin(1),xmin(2),fmin,'gp')
    hold on
    end

    if abs((fmin-f0)/f0) < eps % algoritmas stabdomas ir gaunamas ekstremumo
        % taskas, jei tenkinama salyga
        break
    end
    f0 = fmin;

end
% Gauta geriausia tikslo funkcijos verte ir kintamuju vektorius yra
% priskiriami Xmin ir Ymin
Xmin = xmin;
Ymin = fmin;
```

12 priedas

MatlabSimulink programa Gauso-Zaidelio metodo realizavimui:

```
%% (c) Aurimas Gajauskas

%   Gauso- Zaidelio metodas
% Gauso -Zaidelio paieskos metodas yra vienas paprasciausias tiesiogines
% paieskos metodas.
% Sio metodo esme yra tai, kad vienu metu yra ieskomas ekstremumas tik
% pagal viena kintamaji kitus laikant konstantomis.
% Sioje programoje kiekvienas tiklso funkcijos ekstremumas ieskomas
% naudojant nuosiakliaja paieska, t. y. kiekvieno x1, x2, ..., xn parameto
% intervalas [a, b] yra sudalinamas
% i lygias dalis ir kiekviename is ju nustatoma tikslo funkcijos reiksme.
% Is gautu tiklso funkcijos reiksmiu randama minimali funkcijos
% reiksme ir randamas x parametru vertes su kuriomis gauta minimali
% funkcijos verte


function  [Xmin,Ymin]=AGoptfun_GAUSO_ZAIDELIO(a, b, eps, x0, e);

x = x0; % priskiriamas pradinis taskas

for i = 1:5000 % maksimalus algoritmo iteraciju skaicius

    f0 =  TiksloFunkcijaDK(x(:)); % Tiklso funkcijos vere su pradiniais
    % parametrais


       if length(a)==2
    figure(11)
    plot3(x(1),x(2),f0,'bp')
    hold on
       end
```

```
    for j = 1: length(x) % Nustatomas tikslo funkcijos kintamuju skaicius

        % Sukuriamos tuscios matricos laikimiems funkcijos ir kintamuju x
        % parametrams saugoti
        fa = [];          xa = [];

        for p = a(j) : e(j) : b(j)  % Suskirstomas funkcijos kintamasis
            %x(j) i intervalus nuo a iki b lygais intevalais e
            x(j) = p;
            xa = [xa, x(j)]; % parametru reiksmes saugomos xa matricoje
            f = TiksloFunkcijaDK(x(:)); % tikslo funkcijos verte kiekviename
intervalo taske
            fa = [fa, f];     % funkcijos reiksmes saugomos fa matricoje
        end

        [fmin,xmin] = min(fa); % randama minimali funkcijos verte ir jos
        % vieta matricoje fa
        % kadangi matricos fa ir xa yra tokio pat ilgio, todel minimalios
        % funkcijos indeksas xmin sutampa su xa parametro indeksu su kuria
        % gauta fmin verte
        x(j) = xa(xmin);
    end
    xmin=x;
    fmin = TiksloFunkcijaDK(x(:)); % tikslo funkcija su optimaliais
parametrais x

   if length(a)==2
    figure(11)
    plot3(x(1),x(2),fmin,'bp')
    hold on
   end

    if abs((fmin-f0)/f0) < eps % algoritmas stabdomas ir gaunamas ekstremumo
        % taskas, jei tenkinama salyga
        break
    end
    % Jei pries tai buvusi salyga netenkinama ir dar niera pasiektas
    % maksimalus iteraciju skiacius, tai skaiciavimai kartojami  nuo
    % pradziu
end

Xmin = xmin; % gautos minimalios funkcijos parametø (x1 x2 ... xn)vertes
Ymin = fmin;   % Gauta minimali tiklso funkcijos vere
```

13 priedas

MatlabSimulink programa Gradientinio greičiausio nusileidimo metodo realizavimui:

```
% (c) Aurimas Gajauskas
%  Gradiento greiciausio nusileidimo metodas


function [Xmin,Ymin] = AGoptfun_GRADIENTINIS(a, b, eps, x0, beta, alfa);
x = x0;
for i = 2:10000

    % tikslo funkcija
    f0 = TiksloFunkcijaDK(x(:)); % tikso funkcijos verte pradiniame taske x0
```

```
    for j = 1: length(x) % nustatomas kitnamuju kiekis
        % gradiento metodui reikalinkgos dalines isvestines yra gaunamos
        % naudojant skirtumines  israiskas
        x(j) = x(j) + beta; % prie kintamojo x pridedamas labai mazas
skaicius beta
        f(j) = TiksloFunkcijaDK(x(:)); % su pakeistu kintamuoju
apskaiciuojama tiklso funkcijos verte
        df_x(j) = (f(j) - f0)/beta; % aproksimuojant skirtuminemis lygtimis
gauta kintamojo x  daline isvestine
    end

    mod_grad = sqrt(sum(df_x(:).^2)); % gradiento vektoriaus modulis

    for p = 1: length(x)
        zingsn(p) =  -alfa* (df_x(p)./mod_grad); % apskaiciuojamas kintamojo
x paieskos zingsnis gradiento kryptimi
        x1(p) = x(p) + zingsn(p); % nauja kintamojo x verte. Minuso  zenklas
pries zingsn(p) reiskia,
        % kad ieskome funkcijos minimumo, + kai ieskomas tikslo funkcijos
maksimumas
                % tikrinama ar gauta verte patenka i lestinaja sriti
        if a(p) < x1(p)
            if x1(p)< b(p)
                x(p)=x1(p);
            end
        end
    end
    xmin=x;
    fmin =  TiksloFunkcijaDK(x(:)); % apskaiciuojamas minimali tiklso
funkcijos verte

    if length(a)==2
    figure(11)
    plot3(x(1),x(2),fmin,'gp')
    hold on
    end


    if  abs((fmin - f0)/f0) < eps % algoritmas stabdomas ir gaunamas
ekstremumo taskas, jei tenkinama salyga
        break
    end

end
Xmin = xmin  % gautos minimalios funkcijos parametru (x1 x2 ... xn)vertes
Ymin = fmin % Gauta minimali tiklso funkcijos vere
```

14 priedas

MatlabSimulink „switch-case“ funkcijos veinmačiams opt metodams programa:

```
%%
%% (c) Aurimas Gajauskas
function [Xmin,Ymin] = AGoptim(metodas,a,b,e)

switch metodas
    case 'NP1kint'
        [Xmin,Ymin]=AGoptfun_Nuoseklioji_1kint(a,b,e);
    case 'NP2kint'
        [Xmin,Ymin]=AGoptfun_Nuoseklioji_2kint(a,b,e);
```

```
    case 'AP'
        [Xmin,Zmin]=AGoptfun_AUKSINIO_PJUVIO(a,b,e);
    case 'DM'
        [Xmin,Ymin]=AGoptfun_DICHOTOMIJOS(a,b,e);

    otherwise
        error('case');
end
```

15 priedas

MatlabSimulink „switch-case“ funkcijos programa Gauso- Zaidelio metodui:

```
%%
%% (c) Aurimas Gajauskas

function [Xmin,Ymin] = AGoptim1(metodas, a, b, eps, x0, e )
switch metodas
    case 'GausoZaidelio'
        [Xmin,Ymin]=AGoptfun_GAUSO_ZAIDELIO(a, b, eps, x0, e);

    otherwise
        error('case');
end
```

MatlabSimulink „switch-case“ funkcijos programa Gradientiniam metodui:

```
%% (c) Aurimas Gajauskas

function [Xmin,Ymin] = AGoptim2(metodas,a,b,eps, x0, beta, alfa )
switch metodas
    case 'Gradientinis'
        [Xmin,Ymin]=AGoptfun_GRADIENTINIS(a, b, eps, x0, beta, alfa);

    otherwise
        error('case');
end
```

MatlabSimulink „switch-case“ funkcijos programa Chemotaxis opt. metodui:

```
%% (c) Aurimas Gajauskas

function [Xmin,Ymin] = AGoptim3(metodas, x0, a, b, deltax)
switch metodas
    case 'Chemotaxis'
        [Xmin,Ymin] = AGoptfun_CHEMOTAXIS(x0, a, b, deltax);

    otherwise
        error('case');
end
```

MatlabSimulink „switch-case“ funkcijos programa Evoliucinio programavimo metodui:

```
%%
%% (c) Aurimas Gajauskas
```

```
function [Xmin,Ymin] = AGoptim4(metodas, x0, deltaxe, a, b, p, eps)
switch metodas
    case 'Evoliucinis'
        [Xmin,Ymin] = AGoptfun_EVOLIUCINIS(x0, deltaxe, a, b, p, eps);

    otherwise
        error('case');
end
```

MatlabSimulink „switch-case“ funkcijos programa „Simulated annealing“ opt. metodui:

```
%%
%% (c) Aurimas Gajauskas

function [Xmin,Ymin] = AGoptim5(metodas, x0, lemda, a, b, eps)
switch metodas
    case 'Simulated-Annealing'
        [Xmin,Ymin] = AGoptfun_SIMULATED_ANNEALING(x0, lemda, a, b, eps);

    otherwise
        error('case');
end
```

16 priedas

MatlabSimulink programa Imituojamo grūdinimo metodo realizavimui:

```
%% (c) Aurimas Gajauskas
% Imituojamo gudinimo "Simulated Annealing" opt. metodas

function [Xmin,Ymin] = AGoptfun_SIMULATED_ANNEALING(x0, lemda, a, b, eps);

x=x0;% paieskos pradzios taskas

pa=[]
i_max=10000; % maksimalus iteraciju skaicius
for i=1:i_max
    % funkcijos verte pradiniame taske x
    f0=TiksloFunkcijaDK(x(:));


    % generuojama n nepriklausomu atsitiktiniu dydziu
    y=((rand(1,length(x))).*2)-1;

    % apskacioujamos paieskos krypties vektoriaus u dedamosios
    u=y(:)./(sqrt(sum(y(:).^2)));

    % apskaiciuojamas sekancio zingsnio nepriklausomu kintamuju vektorius
   % lemda(:) kintamuju x paieskos zingsnis
    x1=x(:)+lemda(:).*u(:);
    % tikrinama ar kintamuju vert4s nera uz ribiniu verciu jei yra
priskiriama
    % artimiausia ribine verte vietoje buvusios x vertes
    for as = 1:length(a)
        if x1(as)< a(as)
            x1(as)= a(as);
        end
        if x1(as)> b(as)
            x1(as)= b(as);
```

```
        end
    end

    f1=TiksloFunkcijaDK(x1(:));% apskaiciuojamas tikslo funkcijos verte
naujame taske x1
    % Jei tikslo funkcijos verte f1 yra geresne nei pries tai gauta verte
    %ir nepaisektas minimumas, tai naujai gautos reiksmes pakeicia senasias
gautas reiksmes
    if f1<= f0 & abs((f0-f1)/f0) > eps
        x=x1;

        if length(a)==2
        figure(11)
        plot3(x(1),x(2),f1,'gp')
        hold on
        end

    end

    if    f1<= f0 & abs((f0-f1)/f0) <= eps
        xopt = x;
        fopt=TiksloFunkcijaDK(x1(:));

        break
    end

    % Jei naujai gauta tikslo funkcijos verte  f1 yra blogesne nei f0
    if f1 > f0
        %nustatoma algoritmo derinimo parametro t funkcija
        t =0.0001/i;
        % Nustatoma parametro p verte (0 < p < 1), kuri yra mazejanti
        % funkcija kurios verte mazeja didejant iteraciju skaiciui
        p=1/(exp((f1-f0)/t));
        %generuojamas normalizuotas atsitiktinis dydis V [0, 1]
        V=rand(1);

        % jei skaicius V yra mazesnis uz parametra p
        % pradines kintamuju vertes yra pakeiciamos naujai gautomis vertemis
x1
        if V < p
            x=x1;
        end
    end
end
%Gauta geriausia tikslo funkcijos verte ir kintamuju vektorius yra
priskiriamos Xmin ir Ymin
Xmin = x
Ymin = f1
```