Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Santykinio augimo greičio valdymas periodinėje su pamaitinimu mikroorganizmų kultūroje naudojant neurovaldiklį

Baigiamasis magistro projektas

Paulius Stašinskas

Projekto autorius

Prof. Dr. Vytautas Galvanauskas

Vadovas

Kaunas, 2023

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Santykinio augimo greičio valdymas periodinėje su pamaitinimu mikroorganizmų kultūroje naudojant neurovaldiklį

Baigiamasis magistro projektas

Valdymo technologijos (6211EX014)

Paulius Stašinskas

Projekto autorius

Prof. Dr. Vytautas Galvanauskas

Vadovas

Prof. Dr. Renaldas Urniežius

Recenzentas

Kaunas, 2023

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Paulius Stašinskas

Santykinio augimo greičio valdymas periodinėje su pamaitinimu mikroorganizmų kultūroje naudojant neurovaldiklį

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdamas kitų asmenų autoriaus ar kitų teisių, laikydamasis Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs;

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalintas iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Paulius Stašinskas

Patvirtinta elektroniniu būdu

Stašinskas, Paulius. Santykinio augimo greičio valdymas periodinėje su pamaitinimu mikroorganizmų kultūroje naudojant neurovaldiklį. Magistro baigiamasis projektas vadovas prof. dr. Vytautas Galvanauskas; Kauno technologijos universitetas, Elektros ir elektronikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): elektronikos inžinerija, inžinerijos mokslai.

Reikšminiai žodžiai: santykinis augimo greitis, pamaitinimo srauto greitis, neuroninis valdiklis, PI reguliatorius, bioreaktorius.

Kaunas, 2023. 52 p.

Santrauka

Šiame darbe analizuojamas mikroorganizmų kultūrų augimo procesas, kuris yra svarbus tiek pramonėje, tiek moksle, ir yra susijęs su biologinių medžiagų, tokių kaip fermentai ir antibiotikai, gamyba. Pagrindinis dėmesys skiriamas biomasės santykinio augimo greičio reguliavimui periodinėse su pamaitinimu mikroorganizmų kultūrose, kai augimo medžiagos tiekiamos tam tikrais laiko intervalais. Darbe nagrinėjami du valdymo metodai – neuroninis valdiklis ir PI reguliatorius. Darbo tikslas yra sukurti ištirti neuroninio valdiklio valdymo algoritmą valdant pamaitinimo srauto greitį ir taip keičiant biomasės augimo greitį. Darbas suskirstytas į tris dalis, aprašomas biotechnologinis procesas, sukuriamas ir aprašomas PI reguliatorius ir neuroninis valdiklis. Atliekami bandymai su skirtingais pradiniais parametrais esant triukšmui. Rezultatai vertinami pagal gautus grafikus ir apskaičiuotas paklaidas.

Stašinskas Paulius. Relative Growth Rate Control in Periodic Feed Microorganism Cultures Using a Neural Controller. Master's Final Degree Project supervisor prof. dr. Vytautas Galvanauskas; Faculty of Electrical and Electronics Engineering , Kaunas University of Technology.

Study field and area (study field group): electronics engineering, engineering science.

Keywords: Relative growth rate, feed flow rate, neural controller, PI regulator, bioreactor.

Kaunas, 2023. 52 p.

Summary

This work examines the growth process of microbial cultures, which is crucial in both industry and science, and is associated with the production of biological materials such as enzymes and antibiotics. The primary focus is on the regulation of the relative growth rate of biomass in periodically fed microbial cultures, where growth materials are supplied at specific time intervals. The study investigates two control methods - a neural controller and a PI regulator. The aim of the work is to develop and investigate a neural controller algorithm for managing the feed flow rate, thereby altering the growth rate of the biomass. The work is divided into three parts, describing the biotechnological process, creating and explaining the PI regulator and the neural controller. Trials are carried out with various initial parameters in the presence of noise. Results are evaluated based on the obtained graphs and calculated errors.

Turinys

Paveikslų sąrašas

Lentelių sąrašas

Įvadas

Mikroorganizmų kultūrų augimas yra svarbus procesas tiek pramonėje, tiek moksle, nes jis leidžia gaminti ir analizuoti įvairias biologines medžiagas, tokias kaip fermentai ir antibiotikai, kurie yra gyvybiškai svarbūs kovojant su bakterinėmis infekcijomis ir užtikrinant visuomenės sveikatą. Vienas svarbiausių uždavinių mikroorganizmų kultūrų auginime, siekiant užtikrinti efektyvumą ir kokybę, yra biomasės santykinio augimo greičio reguliavimas.

Periodinėse su pamaitinimu mikroorganizmų kultūrose augimo medžiagos yra tiekiamos tam tikrais laiko intervalais. Toks maitinimo šaltinio tiekimas padeda geriau valdyti kultūros augimą, tačiau tokiu būdu atsiranda iššūkiai valdymo sistemai, kadangi ji turi greitai prisitaikyti prie kintančių sąlygų ir atitinkamai tiekti maitinimą. Darbe aptariamos valdymo strategijos, skirtos valdyti periodinę su pamaitinimu mikroorganizmų kultūrą, ir nagrinėjamas neurovaldiklio bei PI reguliatoriaus pritaikymas procesui.

Neurovaldikliai yra mokymosi algoritmai, kurie pagal sistemos pokyčius pritaiko savo elgesį. Neurovaldikliai dažniausiai naudojami sudėtingose sistemose, kurios reikalauja adaptyvaus valdymo. Darbe analizuojamas santykinio augimo greičio valdymas, naudojant neurovaldiklį, siekiant nustatyti, ar jis gali pasiekti užbrėžtus rezultatus ir ar jie yra geresni nei PI reguliatoriaus.

Neadaptyvusis PI reguliatorius yra paprastas, tačiau plačiai naudojamas reguliatorius. Jis veikia proporcingai mažindamas klaidą tarp norimos ir faktinės parametro reikšmės ir integruodamas klaidą per laiką, siekiant išvengti didelių svyravimų. Darbe tikrinama, ar šis paprastas reguliatorius gali valdyti pamaitinimo srauto greitį bioprocese, siekiant gauti nustatytą biomasės augimo greitį.

Darbo metu bus atliekami eksperimentai, skirti įvertinti neurovaldiklio ir neadaptyvaus PI reguliatoriaus veikimą santykinio augimo greičio valdyme periodinėse su pamaitinimu mikroorganizmų kultūrose. Įvertinant valdymo sistemas, bus sukuriama biomasės augimo greičio nuostata bei triukšmas.

Darbo tikslas yra naudojant imitacinį modeliavimą ištirti neuroninio valdiklio algoritmą, skirtą reguliuoti biomasės santykinį augimo greitį periodiniame su pamaitinimu *E.coli* procese.

Darbo tikslas. Naudojant imitacinį modeliavimą ištirti neuroninio valdiklio algoritmą, skirtą reguliuoti biomasės santykinį augimo greitį periodiniame su pamaitinimu *E.coli* procese.

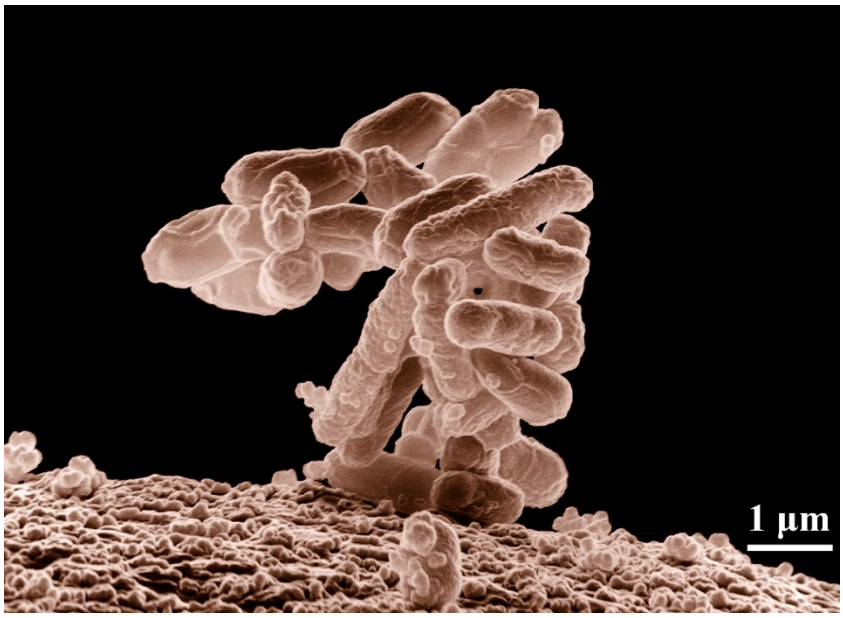
Darbo uždaviniai:
apžvelgti mokslinę literatūrą biotechnologinių procesų valdymo metodų tema;
pasirinkti ir aprašyti tiriamą biotechnologinį procesą ir sukurti jo matematinį modelį „Matlab“ aplinkoje;
sukurti neuroninio valdiklio algoritmą „Matlab“ aplinkoje;
atlikti imitacinį modeliavimą naudojant PI reguliatorių ir neuroninį valdiklį, naudojant įvairius santykinio augimo greičio laikinius profilius ir matavimo triukšmo vertes;
palyginti skirtingais valdymo algoritmais gautus rezultatus;
pateikti išvadas ir rekomendacijas.

Projektinė dalis suskirstyta į 3 dalis. Pirmoje dalyje yra aprašomas biotechnologinis procesas, pradiniai parametrai ir kt. Antroje dalyje yra sukuriamas PI reguliatorius, aprašomos jo formulės, veikimas. Atliekami bandymai su skirtingais pradiniais parametrais ir apskaičiuojamos paklaidos. Patikrinama, ar reguliatorius gerai veikia, sukuriant triukšmą. Trečioje dalyje yra aprašomas ir sukuriamas neuroninis valdiklis. Atliekami tokie pat bandymai kaip ir su PI reguliatoriumi. Galiausiai valdiklio ir reguliatoriaus veikimas yra palyginami, išsiaiškinama, kuris yra geresnis, įvertinant gautus grafikus bei apskaičiuotas paklaidas.

1. Teorinė dalis

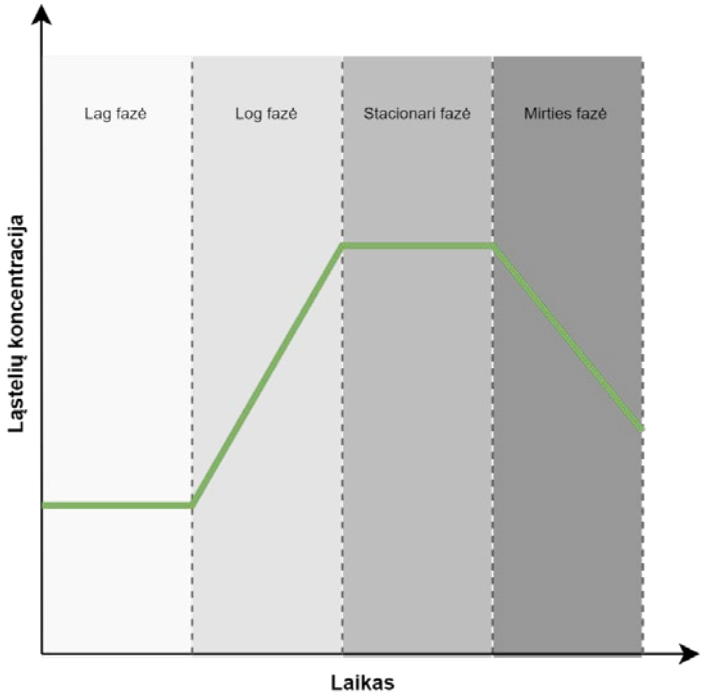
1.1. *E. Coli* bakterija

E. coli bakterija yra Enterobacteriaceae (enterobakterijų) šeimos rūšis [1]. Ji yra pailga, kelių mikronų ilgio ir 0,5 μm pločio. Ši bakterija yra fakultatyvinis anaerobinis organizmas, galintis augti anaerobiškai fermentacijos būdu ir aerobiškai oksidacijos būdu. Prokariotai yra vertinami dėl lengvo genetinio manipuliavimo, didelės išeigos, greito augimo ir gana paprastų augimo sąlygų. Pagrindinis šių bakterijų privalumas pramonėje – gebėjimas išskirti rekombinantinius baltymus ir naudingus fermentus į terpę, galimybė augti nebrangioje mitybos terpėje ir tolerancija dideliam ląstelių tankiui. Rekombinantiniai baltymai pasižymi stipriu vaistiniu poveikiu, yra saugūs vartoti ir neturi netikslinių, nepageidaujamų poveikių. Plačiausiai rekombinantiniai baltymai yra naudojami farmacijos srityje, juos lengva pritaikyti vaistų gamybai ir lengva modifikuoti [2].



1 pav. Escherichia bakterija

Sudarant optimalias sąlygas bakterijai augti, *E. coli* dvigubėjimo laikas yra apie 20 minučių. Stacionarioji ląstelių augimo fazė pasiekiama daug greičiau palyginti su kitomis bakterijomis. Ši fazė yra trečia iš keturių ląstelių augimo proceso fazėse (2 pav.), joje ląstelių koncentracija yra pati didžiausia [3]. "Lag" fazėje ląstelės adaptuojasi prie sudarytų augimo sąlygų bei augimo terpės. Šioje fazėje pagrindinė veikla yra adaptacija, ląstelių koncentracija beveik nekinta. Šio proceso trukmė priklauso nuo pradinės ląstelių koncentracijos, terpės ir ląstelių būklės. Kuomet pradeda didėti ląstelių koncentracija, pasiekiama "log" fazė. Šioje fazėje vyksta ypač greitas ląstelių dauginimasis. Dauginimosi greitis priklauso nuo specifinio augimo greičio, o pats specifinis augimo greitis priklauso nuo kitų faktorių, kaip substrato koncentracijos, ištirpusių mineralų koncentracijos ir kt. [4]. Stacionariojoje augimo fazėje ląstelių koncentracija nusistovi ir nekinta, kadangi ląstelės dauginasi tokia pačia sparta, kaip ir miršta. Pasibaigus maistinėms medžiagoms, ląstelių dauginimasis sulėtėja ir jų mirštamumas tampa didesnis, dėl to koncentracija sparčiai mažėja. Pramonėje siekiama pritaikyti įvairius būdus, medžiagų tiekimo profilius, valdymo būdus, kad būtų išgaunamas kuo didesnis biomasės kiekis, išlaikyti antrą ir trečią fazes kuo ilgiau [5].

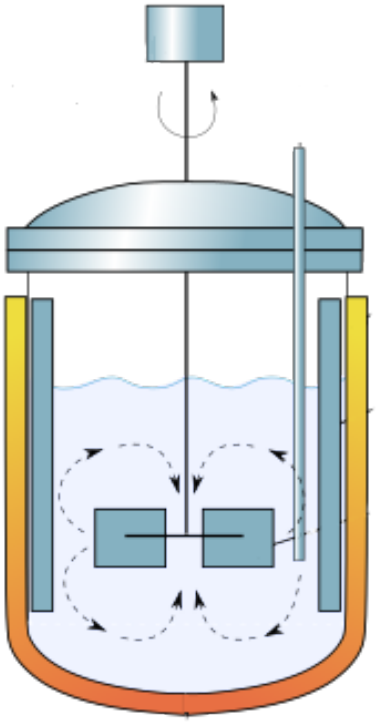


2 pav. Ląstelių augimo fazės

1.2. Bioreaktorius

Pagrindinis įrenginys biotechnologiniuose procesuose yra bioreaktorius. Yra kelių rūšių bioreaktoriai: anaerobiniai, aerobiniai, nenutrūkstamo ir periodinio veikimo, pusiau periodinio veikimo bioreaktoriai [6].

Aerobiniuose reaktoriuose deguonis gaunamas įpurškimo ar oro burbuliavimo sistemomis. Kadangi deguonies tirpumas bioreaktoriuje ganėtinai mažas, dažnai naudojamos maišyklės pagreitinti deguonies tirpumą. Deguonies tirpumas yra pagrindinis parametras, kurį reikia optimaliai palaikyti, kadangi kiti bioreaktoriuje esantys elementai kaip druska įtakoje deguonies tirpumą. Labiausiai paplitęs aerobinis reaktorius yra STR reaktorius [7] (angl. stirred tank reactor) (3 pav). Reaktoriuje įmontuota sparnuotė arba maišytuvas sukelia turbulenciją kas užtikrina, kad biomasė būtų gerai išmaišyta. Maišytuvas gali būti varomas elektriniu varikliu ar kitomis jėgos priemonėmis. Maišytuvo dydis ir forma gali skirtis priklausomai nuo konkrečios paskirties. Vienas pagrindinių STR reaktorių privalumų - universalumas. Jis gali būti naudojamas plataus spektro biologinėse ir cheminėse reakcijose, ypač tinka procesams kurie apima skystosios fazės reakcijas. Be to, galimybė valdyti reaktoriaus maišymo greitį ir temperatūrą [8] leidžia tiksliai kontroliuoti reakcijos sąlygas kurios ypač svarbios norint užtikrinti norimą produkto augimo spartą.



3 pav. CSTR bioreaktoriaus principinė schema

Anaerobiniai bioreaktoriai skirti valyti organines atliekas arba gaminti biokurą iš organinių šaltinių, pvz., atliekų, žemės ūkio atliekų arba biomases [9]. Anaerobinė fermentacija yra pagrindinis procesas, kuris vyksta šiame reaktoriuje[10]. Fermentacijos metu mikroorganizmai, kurie gyvena be deguonies, skaido organines medžiagas, o rezultate susidarančios medžiagos naudojamos kaip energijos šaltinis arba šalinamos kaip sausos ar skystos atliekos. Šie reaktoriai dažniausiai naudojami atliekų valymo procesuose, tačiau taip pat gali būti naudojami biomasės fermentacijos procesuose biokurui ar biometanolio gamybai.

Anaerobiniai bioreaktoriai [11] gali būti klasifikuojami kaip uždaro tipo reaktoriai arba atviro tipo reaktoriai. Uždaro tipo reaktoriai naudojami valyti didelių mastelių atliekas, tokias kaip nuotekos, pramoniniai šlakai ir kt. Atviro tipo reaktoriai dažniausiai naudojami mažesnių mastelių procesams, tokiems kaip biomasės fermentacija.

Anaerobiniai bioreaktoriai turi keletą privalumų palyginti su tradicinėmis nuotekų valymo sistemomis, įskaitant mažesnes energijos sąnaudas, mažesnį išmetamųjų teršalų kiekį ir mažesnę ekologišką poveikį.

Nenutrūkstamo veikimo bioreaktorius [12] yra bioreaktoriaus tipas, kuriame mikroorganizmai yra nuolat kultivuojami, nes jų augimo sąlygos yra nuolat palaikomos stabilios. Šis bioreaktorius turi nuolatinį medžiagų tiekimą ir nuolatinį produkto pašalinimą, kuris leidžia išlaikyti optimalias mikroorganizmų sąlygas ir sustabdyti produkto kaupimąsi.

Nenutrūkstamo veikimo bioreaktoriai gali būti naudojami įvairiems tikslams, tokiems kaip biokuro gamybai, fermentacijai ar vaistų gamybai. Šie bioreaktoriai dažnai naudojami dideliems gamybos apimtims, nes jie gali nuolat gaminti produkto srautą.

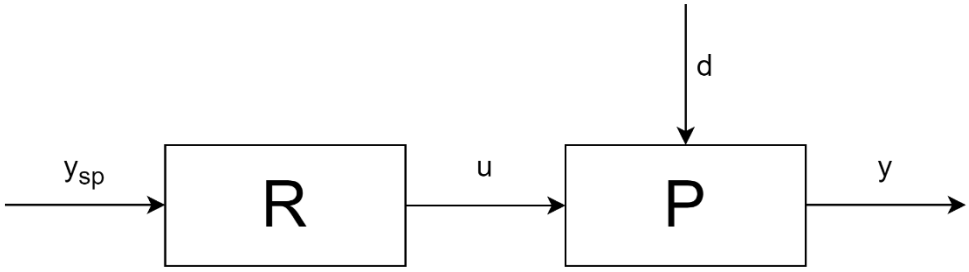
Nenutrūkstamo veikimo bioreaktoriai [13] yra įdomus technologinis sprendimas, nes jie leidžia pasiekti aukštesnį produktyvumo lygį, o tuo pačiu metu reikalauja mažesnių pastangų ir sąnaudų, palyginus su kitais bioreaktoriais. Tai reiškia, kad šie bioreaktoriai gali padėti sumažinti gamybos sąnaudas ir padidinti pelną.

Nenutrūkstamo bioreaktorius susideda iš kelių pagrindinių komponentų:

- Talpa : konteineris, kurioje vyksta mikroorganizmų kultivavimas. Konteinerio dydis gali skirtis priklausomai nuo konkretaus naudojimo atvejo ir mikroorganizmų apimties, kurią reikia kultivuoti.
- Cirkuliacijos sistema: Tai sistema, kuria naudojama nupilti naujų medžiagų į bioreaktorių ir išsiurbti produkto, kuris yra jau pagamintas. Dažniausiai cirkuliacijos sistema susideda iš pompos, filtro ir vamzdžių.
- Cirkuliacijos sistema: Tai sistema, kuria naudojama nupilti naujų medžiagų į bioreaktorių ir išsiurbti produkto, kuris yra jau pagamintas. Dažniausiai cirkuliacijos sistema susideda iš pompos, filtro ir vamzdžių.
- Maitinimo sistema: Tai sistema, kuria naudojama tiekti mikroorganizmams reikalingas maistines medžiagas, pvz., angliavandenius, baltymus ir vitaminus.
- Valdymo sistema: Tai sistema, kuri kontroliuoja ir palaiko tam tikrą reakcijos sąlygų režimą, pvz., temperatūrą, pH vertę, slėgį ir kt. Valdymo sistema dažnai naudojama automatizuoti bioreaktoriaus procesą ir užtikrinti konsistenciją bei produktyvumą.

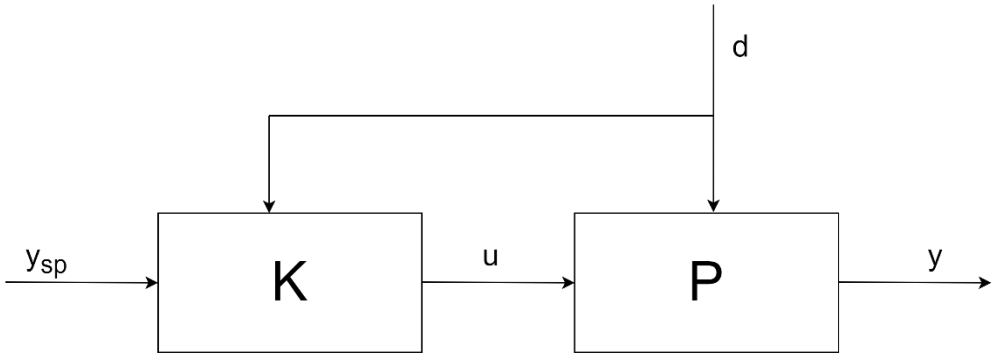
1.3. Valdymo sistema

Norint užtikrinti optimalų bioreaktoriaus darbą reikalinga gerai sureguliuota valdymo sistema. Jų pagalba procesas yra kontroliuojamas gaunant tam tikrus parametrus, signalus išėjime keičiant įėjimo parametrus. Periodiniuose su pamaitinimu biotechnologiniuose procesuose dažniausiai taikoma atviros sistemos režimas [14] (angl. open loop control)(4 pav.).Šio režimo metu realizuojami išankstiniai valdymo kintamųjų u profiliai, kurie turi užtikrinti kintamųjų y trajektorijas, kuomet sistemą veikia trikdžiai d. Reikia atkreipti dėmesį, kad būtina nusistatyti sistemos apribojimus, kadangi esant atsitiktiniams trikdžiams d sistema gali nukrypti nuo optimalių trajektorijų. Šis valdymo metodas paprastas ir plačiai naudojamas tiek pramonės srityje, tiek laboratoriniuose bandymuose.



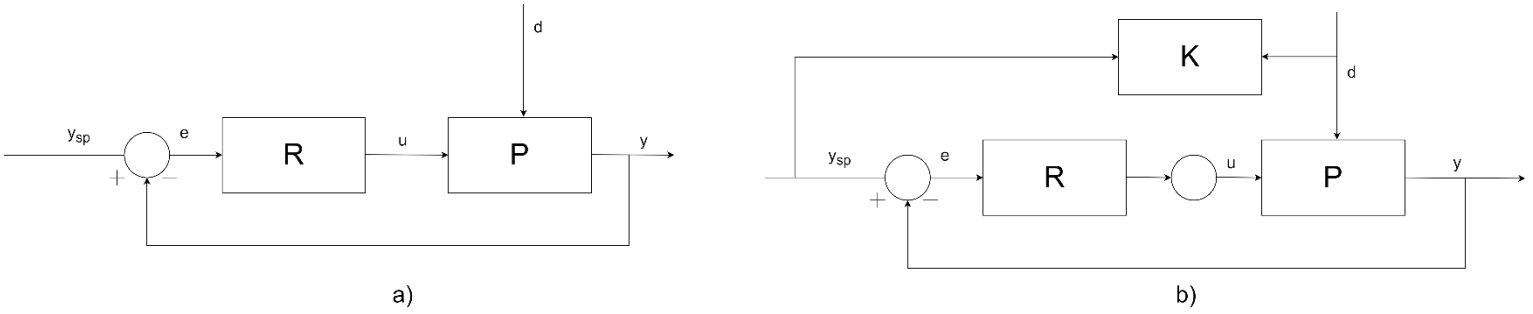
4 pav. Atvirosios sistemos režimo principinė schema

Norint išvengti trikdžio poveikio yra naudojamos atvirosios valdymo sistemos kompensuojančios trikdžius [15] (4 pav.). Esminis sistemos privalumas – atsižvelgiant į sistemą veikiantį trikdį d, realiu metu galimas valdymo poveikio u koregavimas. Nors šis režimas taip neturi grįžtamojo ryšio kaip ir atvirasis režimas, užtikrinama geresnė valdymo kokybė. Pagrindinė problema – funkcinių priklausomybių nustatymas tarp trikdžio d ir valdančiojo poveikio u korekcijos dydžio. Tai būtina norint pašalinti sistemoje trikdžio d sukeltus triukšmus. Taip yra nematuojami trikdžiai, kurie negali būti kompensuojami.



5 pav. Atvirosios sistemos režimo su trikdžio kompensavimo principinė schema

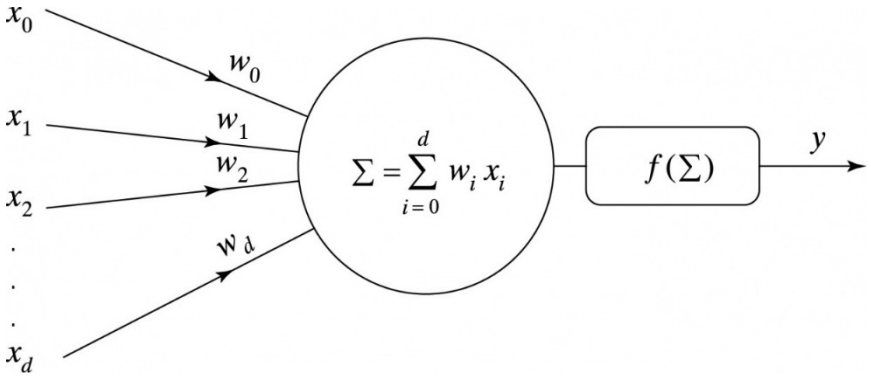
Valdymo sistemos su grįžtamuoju ryšiu [16] yra ypač plačiai naudojamos (6 pav.). Signalai gaunami per grįžtamojo ryšio kanalą trikdžio kompensavimui. Šios sistemos turi būti kokybiškai suderintos atkreipiant dėmesį į proceso netiesiškumus. Suderinus sistemą ji turi gerai veikti esant trikdžiams ar keičiant nuostatą. Taip pat yra sistemos su grįžtamuoju ryšiu kompensuojančios trikdžius, kaip ir atvirose sistemose. Šiuo atveju grįžtamojo ryšio reguliatorius ir kompensavimo grandis yra atskirai, kas užtikrina kokybiškesnę proceso valdymo kokybę.



6 pav. a)Valdymo sistema su grįžtamuoju ryšiu, b) valdymo sistema su grįžtamuoju ryšiu ir trikdžio kompensavimu

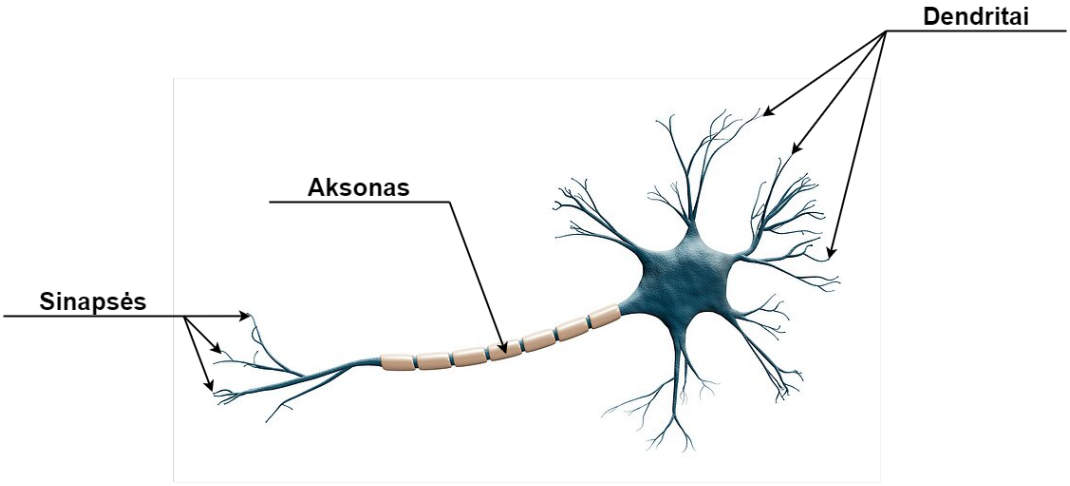
1.4. Dirbtiniai neuronai

Biologinis neuronas [17] (8 pav.) yra biologinis nervų ląstelių elementas, atsakingas už informacijos priėmimą, apdorojimą ir perdavimą kitiems neuronams, kurie sudaro nervų sistemą. Biologinis neuronas sudarytas iš kelių pagrindinių dalių: dendritai, kurie veikia kaip informacijos įėjimas ir priima informaciją, aksonai, kurie perduoda informaciją per neuroną į sinapsę, kuri turi daug išsišakojimų ir per tuos išsišakojimus perduoda informaciją kitiems neuronams elektros impulsais. Dirbtinis neuronas (7 pav.) yra matematinis modelis, kuriame išlaikoma panaši struktūra ir veikimas. Įvesties duomenys atitinka dendritus, kurie gauna duomenis iš išorės šaltinių arba kitų neuronų. Įvesties signalai turi skirtingą svarbą ir tam tikrus svorius, kurie gali būti teigiami arba neigiami, t. y. neuronas gali turėti slopinimo arba sužadinimo efektą. Šie svoriai atitinka biologinio neurono sinapsių efektyvumą. Jie yra sumuojami ir įvertinami perduodant juos per aktyvavimo funkciją.



7 pav. Dirbtinio neuroninio tinklo elementas

čia x1...xn – neurono įėjimai; w1...wn – atitinkami svoriai; w0 – slenksčio reikšmė; f() – perdavimo funkcija; y – neurono išėjimas.



8 pav. Neurono struktūra

1.5. Dirbtinis neuroninis tinklas

Sujungus du ar daugiau dirbtinių neuronų gaunamas dirbtinis neuroninis tinklas [18]. Galima teigti, kad neuroninis tinklas tai tam tikrų netiesinių matematinių priklausomybių visuma. Vienas dirbtinis neuronas negali būti naudojamas sistemoms optimizuoti, reikalinga dirbtinių neuronų sistema. Dažniausiai naudojami neuroninių tinklų ryšiai nėra aiškūs ryšiai tarp bioprocesų parametrų dėl to negalima tiesiogiai jų modeliuoti.

Dirbtinio neuroninio tinklo struktūrą galima išskaidyti į 3 pagrindines grupes: įėjimų sluoksnis, paslėptasis sluoksnis ir išėjimo sluoksnis. Šie sluoksniai sudaryti, kad būtų aiškesnis neuroninio tinklo struktūros vaizdas.

Neuroniniai tinklai yra mašininio mokymosi modeliai, kurie mėgdžioja biologinių neuroninių tinklų struktūrą ir veikimo principus. Jie yra plačiai naudojami įvairiose srityse, tokiose kaip nuotraukų klasifikavimas, kalbos atpažinimas, paieška ir rekomendacijų sistemose. Neuroniniai tinklai susideda iš neuronų, kurie yra sujungti į sluoksnius. Kiekvienas neuronas gauna įvesties signalus, apdoroja juos ir perduoda gautą rezultatą kitų sluoksnių neuronams. Šis procesas tęsiasi tol, kol pasiekiamas išvesties sluoksnis, kuriame gaunami galutiniai modelio rezultatai [19].

Yra įvairių neuroninių tinklų tipų, priklausomai nuo jų struktūros ir užduoties. Pagrindiniai neuroninių tinklų tipai yra:
Rekurentiniai neuroniniai tinklai (RNN) yra ypatinga neuroninių tinklų rūšis, kurios unikali savybė yra galimybė išsaugoti informaciją iš praeities įvesties duomenų [20]. Tai leidžia RNN efektyviai dirbti su sekos duomenimis, pavyzdžiui, tekstais, muzika ar bet kokia kita informacija, kurioje svarbus laiko aspektas. Štai kodėl RNN plačiai naudojami kalbos modeliavime ir mašininiame vertime.
Konvoliuciniai neuroniniai tinklai (CNN) yra inspiruoti biologinio regos proceso ir yra ypač tinkami darbui su vaizdo duomenimis [21]. Tai pasiekta per konvoliucijos, apjungimo ir pilnai sujungtų sluoksnių naudojimą, kurie kartu leidžia išmokti sudėtingas funkcijas iš duomenų. CNN gali būti naudojami vaizdų analizei, objektų aptikimui ir vaizdų generavimui.
Gilieji neuroniniai tinklai (DNN) yra neuroninių tinklų tipas, sudarytas iš daug sluoksnių, kurių kiekvienas atlieka skirtingą informacijos apdorojimo užduotį [22]. DNN naudojami įvairiose užduotyse, tokiuose kaip vaizdų klasifikavimas, kalbos atpažinimas ir paieška. Ypač svarbu tai, kad gilieji neuroniniai tinklai gali būti efektyviai mokomi naudojant specialius algoritmus, kurie leidžia perduoti gradientus per daug sluoksnių.
Ilgalaikio atminties neuroniniai tinklai (LSTM) yra specifinė RNN rūšis, sukurta siekiant spręsti ilgalaikės priklausomybės problemą, su kuria susiduria RNN [23]. Tai pasiekiama per specialią struktūrą, vadinamą "atminties ląstele", kuri leidžia modeliui išmokti ir išlaikyti informaciją per ilgesnį laiko periodą.
Daugiasluoksniai neuroniniai tinklai yra universali neuroninių tinklų klasė, kurioje kiekvienas sluoksnis yra sujungtas su kitais. Dėl savo lankstumo, jie gali būti naudojami įvairiose užduotyse, tokiuose kaip klasifikavimas, regresija ar rekomendacijų sistemos kūrimas.
Modelio neturintis adaptyvus valdymas MFA (angl. model free adaptive) yra realaus laiko valdymo metodas, kuris nepritaiko jokio proceso modelio. MFA valdymas leidžia automatiškai prisitaikyti prie proceso pokyčių ir kintamumo, nereikalaujant sudėtingų rankinių parametrų derinimų ar kiekybinės žinios apie procesą. MFA valdikliai gali veiksmingai reaguoti į proceso dinamikos, trikdžių ir nuostatų pokyčius, siekdami mažinti proceso kintamųjų paklaidas realiu laiku.

Skirtingai nuo identifikavimu paremtų valdymo metodų, MFA valdymo algoritmas neparemtas proceso identifikavimo mechanizmu, todėl jo veikimas yra paprastesnis ir greitesnis. Adaptyvus MFA valdymas taiko dirbtinio neuroninio tinklo principus, kurie automatiškai peržiūri ir atnaujina svorinius koeficientus, modifikuodami valdomo proceso dinamiką. Tokiu būdu siekiama minimizuoti nuokrypį tarp proceso rezultato ir iš anksto nustatytos vertės.
Vienu iš pagrindinių MFA valdymo metodo pranašumų laikomas jo gebėjimas natūraliai adaptuotis prie kintančių proceso sąlygų, apkrovos bei triukšmo, be poreikio atlikti rankinį reguliavimą. Papildomai, MFA valdiklis pritaiko įėjimo verčių vėlinimo principą per vieną diskretinio laiko intervalą, kas leidžia kaupti informaciją apie praeities duomenis ir valdyti procesą remiantis įgytomis žiniomis. Modelio neturintis adaptyvus valdymas (MFA) yra realaus laiko valdymo metodas, kuris nepritaiko jokio proceso modelio. MFA valdymas leidžia automatiškai prisitaikyti prie proceso pokyčių ir kintamumo, nereikalaujant sudėtingų rankinių parametrų derinimų ar kiekybinės žinios apie procesą. MFA valdikliai gali veiksmingai reaguoti į proceso dinamikos, trikdžių ir nuostatų pokyčius, siekdami mažinti proceso kintamųjų paklaidas realiu laiku [24].
Skirtingai nuo identifikavimu paremtų valdymo metodų, MFA valdymo algoritmas neparemtas proceso identifikavimo mechanizmu, todėl jo veikimas yra paprastesnis ir greitesnis. Adaptyvus MFA valdymo metodas yra grindžiamas dirbtinio neuroninio tinklo technologija, kuri automatiškai atnaujina svorinius koeficientus bei modifikuoja valdomo proceso dinamiką. Šio metodo tikslas - mažinti nuokrypį tarp realaus proceso rezultato ir iš anksto nustatytos vertės.

1.6. Apmokymas

Neuroninių tinklų apmokymas yra esminė sudėtingų modelių mokymo ir tobulinimo dalis, kurios pagrindinis tikslas yra išmokyti tinklą atpažinti tam tikras savybes ir modeliuoti funkcijas, kurios gali būti naudojamos sprendžiant įvairias užduotis, tokias kaip klasifikavimas, regresija, atpažinimas ir t.t. Neuroniniai tinklai yra sudaryti iš sluoksnių, kuriuose yra daug neuronų, sujungtų tarpusavyje. Kiekvienas neuronas atlieka tam tikrą paprastą funkciją, o visas tinklas sudaro sudėtingą funkcijų hierarchiją [25].

Apmokymo procesas prasideda nuo neuroninio tinklo inicializavimo su atsitiktiniais svoriais. Vėliau per mokymą, tinklas naudoja duomenų rinkinį, kuris yra suskirstytas į mokymo, validavimo ir testavimo porcijas, kad išmoktų tinkamus svorius ir būtų galima įvertinti tinklo veikimą. Duomenų rinkinys yra pateikiamas tinklui per daug iteracijų, vadinamų epochomis. Per kiekvieną epochą, tinklas pritaiko savo svorius pagal klaidą, kurią sukelia dabartiniai svoriai, naudodamasis tokiomis optimizavimo technikomis kaip, pavyzdžiui, gradientinis nusileidimas [26].

Vienas iš pagrindinių apmokymo aspektų yra tinkamo mokymosi tempimo (angl. learning rate) parinkimas, kuris kontroliuoja, kaip greitai tinklas pritaiko savo svorius. Per didelis mokymosi tempimas gali sukelti nestabilumą, o per mažas - lėtą konvergenciją [27].

Daugelyje šiuolaikinių neuroninių tinklų apmokymo atvejų naudojama aibių mokymas (angl. batch learning). Aibių mokymas reiškia, kad duomenys yra padalijami į mažesnes grupes, vadinamas aibėmis (angl. batches), ir per kiekvieną iteraciją tinklas yra apmokomas naudojant vieną aibę. Tai padeda sumažinti atminties naudojimą ir leidžia efektyviau atnaujinti svorius, ypač kai naudojami dideli duomenų rinkiniai [27].

Dar vienas svarbus apmokymo aspektas yra reguliarizacija. Reguliarizacija padeda kontroliuoti neuroninio tinklo sudėtingumą ir sumažinti persimokymą (angl. overfitting). Dažniausiai naudojamos reguliarizavimo technikos yra L1 ir L2, kurios prideda prie klaidos funkcijos baudos terminą, kuris yra proporcingas svorių dydžiui [28]. Kiti reguliarizavimo metodai apima išmetimą (angl. dropout), kai tam tikri neuronai yra atsitiktinai išjungiami apmokymo metu, ir kultūros normalizavimas (angl. batch normalization), kuri padeda sumažinti klaidų sklaidą tarp sluoksnių [29].

Neuroninių tinklų apmokymas yra itin svarbus žingsnis, kuris lemia tinklo veikimo efektyvumą. Apmokymo metu modelis naudoja duomenis ir optimizavimo algoritmus, kad išmoktų sudėtingus ryšius tarp įvesties ir išvesties. Nors egzistuoja įvairūs apmokymo metodai, pagrindinė idėja yra ta pati: tinkamas svorių parinkimas, kuris leidžia modeliui gerai prognozuoti naujus duomenis.

1.7. PID reguliatorius

Proporcinis integralinis išvestinis reguliatorius (toliau vadinama PID) yra plačiai naudojamas, nebrangus ir ganėtinai lengvai derinamas. Reguliatoriaus realizavimas priklauso nuo sistemos sudėtingumo, esamų trikdžių, esant įvairioms matavimo klaidoms. Atsižvelgiant į sistemos sudėtingumą galima naudoti P, PI, PID reguliatorius. Reguliatorius yra reguliavimo algoritmas, kurio tikslas yra išlaikyti valdomo objekto (pvz., temperatūros, greičio ar pozicijos) reikšmę norimoje vertėje.

Proporcingoji dalis reaguoja į dabartinį valdomo objekto nuokrypį nuo norimos vertės. Ji proporcingai keičia reguliatoriaus išvesties signalą, kad būtų sumažintas šis nuokrypis. Kai valdomo objekto nuokrypis yra didesnis, proporcinga dalis padidina išvesties signalą ir taip sumažina nuokrypį. Kai valdomo objekto nuokrypis yra mažesnis, proporcinga dalis sumažina išvesties signalą ir taip mažina jau mažą nuokrypį. Didelis proporcingojo koeficientas padidina proporcingos dalies reakcijos greitį, tačiau taip pat gali padidinti stabilumo problemų riziką, nes reguliatorius norimas vertes gali keisti per greitai. Mažas proporcingojo koeficientas sumažina reguliatoriaus jautrumą, todėl reikia ilgesnio laiko, kad valdomas objektas pasiektų norimą vertę. Proporcinga dalis yra labai svarbi reguliuojant valdomus objektus, kurie greitai keičia savo būseną, pvz., greičio reguliavimo sistemose. Tai leidžia valdomam objektui greitai reaguoti į kintančias sąlygas ir palaikyti norimą vertę. Toliau pateikiama matematinė proporcingosios dalies išraiška:

$$P = K_p * e(t); \quad (1)$$

čia: P – proporcingoji dalis; Kp – proporcingumo koeficientas; e(t) – dabartinis valdomo objekto nuokrypis nuo norimos vertės.

Integralinė dalis yra skirta reguliuoti nuokrypį tarp valdomo objekto būsenos ir norimos vertės per laiko tarpą, taigi ji yra naudinga stabilizuojant valdomą objektą nuo pastovios būsenos klaidos. Ji atlieka integravimo operaciją, kaupia valdomo objekto nuokrypį nuo norimos vertės per laiko tarpą ir reaguoja proporcingai į šį kumuliatyvų nuokrypį. Integralinė dalis dažniausiai taikoma tam, kad sumažintų pastovios būsenos klaidą, kuri gali išlikti netgi tada, kai proporcinga dalis reguliuoja valdomą objektą. Integralinė dalis gerai reguliuoja valdomus objektus kurių reakcijos laikas yra ilgas,

tačiau jei valdomas objektas pernelyg greitai keičia savo būseną integralinė dalis gali sukelti perreguliavimą ir nestabilumą. Toliau pateikiama matematinė integralinės dalies išraiška:

$$I = K_i \int_0^t e(t)dt\,;$$

čia: I – integralinė dalis; Ki – integravimo koeficientas; $\int_0^t e(t)dt$ – valdomo objekto nuokrypio nuo norimos vertės integralas per laiką. (2)

Diferencialinė dalis reaguoja į valdomo objekto reikšmės pokyčio greitį, bandydama iš anksto numatyti, kaip jis bus keičiamas ir taip sumažinti nuokrypį. Ši dalis padeda sumažinti perreguliavimą ir tikrina, kad valdomas objektas greitai reaguotų į pokyčius. Tačiau jei valdomas objektas pernelyg jautrus triukšmui diferencinė dalis gali sustabdyti reguliavimo procesą arba sukelti nestabilumą. Didelis diferencinės dalies koeficientas padidina diferencinės dalies jautrumą, tačiau taip pat padidina triukšmo įtakos riziką. Mažas koeficientas sumažina diferencinės dalies jautrumą, todėl reikia daugiau laiko, kad valdomas objektas pasiektų norimą vertę. Toliau pateikiama matematinė diferencinės dalies išraiška.

$$D = K_d \frac{de(t)}{dt}; \quad (3)$$

čia: D– diferencinė dalis; Kd – diferencinės dalies koeficientas; $\frac{de(t)}{dt}$ – valdomo objekto nuokrypio pokytis per laiką.

PID reguliatorius yra universalus ir gali būti pritaikytas prie daugelio sistemų. Jis yra naudojamas pramonėje, pvz., cheminių procesų reguliavimui, skysčių ir dujų srautų reguliavimui, robotikos ir automatikos srityse. Taip pat jis yra naudojamas medicinoje, automobilių valdymo sistemose, šildymo ir aušinimo sistemose, vandens tiekimo ir valymo sistemose, klimato valdymo sistemose ir kt.

PID reguliatorius yra labai svarbus automatinio valdymo sistemos elementas, nes jis užtikrina, jog valdomo objekto reikšmė būtų reguliuojama taip, kad ji išlaikytų norimą vertę. Reguliavimo algoritmo sudėtingumas priklauso nuo to, kokią sistemą jis valdo, tačiau pritaikant PID reguliatorių galima gauti gerų reguliavimo rezultatų.

1.8. Biotechnologinis procesas

Yra 3 tipiniai biotechnologinių procesų tipai – periodinis su pamaitinimu procesas, periodinis procesas, nenutrūkstantis procesas.

Periodinis su pamaitinimu procesas yra biotechnologinis procesas, kuriame fermentacinis reaktorius yra periodiškai papildomas naujomis medžiagomis ir iš jo pašalinamos senosios medžiagos, siekiant

išlaikyti reikiamą koncentraciją ir užtikrinti optimalią mikroorganizmų augimo ir produkto sintezės sąlygą. Šis procesas yra naudojamas įvairiose biotechnologijos srityse, įskaitant bioetanolio gamybą, antibiotikų sintezę, biodegradaciją ir kitus fermentacijos procesus [31].

Periodinio su pamaitinimu proceso pranašumas yra tas, kad jis leidžia kontroliuoti reaktoriaus sąlygas, pavyzdžiui, substrato koncentraciją, pH, temperatūrą ir kitus veiksnius, kurie gali turėti įtakos mikroorganizmų augimui ir produkto sintezei. Be to, šis procesas sumažina produktų ar šalutinių produktų kaupimąsi reaktoriuje, kuris gali slopinti mikroorganizmų augimą ar produkto sintezę [32].

Periodinis su pamaitinimu procesas susideda iš kelių etapų. Pirmasis etapas yra reaktoriaus užpildymas pirminiu substratu t.y. specifinis maisto šaltinis, pritaikytas, produkto auginimui. Antrasis etapas yra fermentacijos proceso vykdymas, per kurį mikroorganizmai vartoja substratą ir sintetina produktą. Fermentacijos metu reaktoriaus sąlygos yra kontroliuojamos, siekiant užtikrinti optimalų mikroorganizmų augimą ir produkto sintezę.

Trečiasis etapas yra periodinis pamaitinimas, kai į reaktorių yra įpilama naujo substrato ir iš jo pašalinamos senosios medžiagos. Šis etapas leidžia palaikyti reikiamą substrato koncentraciją ir sumažinti produktų ar šalutinių produktų kaupimąsi reaktoriuje. Pamaitinimas gali būti atliekamas įvairiais būdais, pavyzdžiui, naudojant skysto arba granuliuoto substrato įpylimą, substrato infuziją arba substrato recirkuliaciją [32].

Ketvirtasis etapas yra fermentacijos proceso baigimas, kai produktas yra išgautas iš reaktoriaus, o likusios medžiagos yra pašalinamos arba panaudojamos kituose procesuose. Pavyzdžiui, fermentacijos metu susidariusios biomasės atliekos gali būti naudojamos kompostavimui arba dujų gamybai anaerobinėje fermentacijoje. Galiausiai, gautas produktas yra valomas ir išvalytas, siekiant atitikti komercinius standartus ir reikalavimus.

Periodinio su pamaitinimu proceso optimizavimas yra svarbus, siekiant užtikrinti aukštą produkto gamybos našumą ir efektyvumą. Proceso optimizavimas apima substrato koncentracijos, pamaitinimo strategijos, pH, temperatūros ir kitų veiksnių reguliavimą. Dažnai naudojami matematiniai modeliai ir mašininio mokymosi algoritmai, kad būtų galima nustatyti optimalias proceso sąlygas [33].

Vienas iš pagrindinių periodinio su pamaitinimu proceso trūkumų yra tai, kad jis yra mažiau efektyvus nei nepaliauji fermentacijos procesai, kuriuose substratas ir produktas nuolat įvedami ir išleidžiami iš reaktoriaus. Tačiau periodinis su pamaitinimu procesas yra naudingas tam tikruose taikymuose, kai reikia kontroliuoti reaktoriaus sąlygas ir sumažinti produkto ar šalutinių produktų kaupimąsi [32].

Kitas biotechnologinio proceso tipas yra periodiniai procesai. Juose bioreaktorius užpildomas nauja biomase po kiekvieno ciklo. Tai reiškia, kad pamaitinimas atliekamas proceso pradžioje ir daugiau netiekiamas. Nors substratas netiekiamas, į bioreaktorių yra tiekiamos kitos medžiagos proceso valdymui ar specifiniams parametrams palaikyti kaip rūgšties tirpalų srautai, pH reguliavimo bazė ir kt. Atlikus procesą bioreaktoriaus turinys yra mažinamas ir galiausiai ištuštinamas. Vėliau bioreaktorius yra išvalomas ir vėl iš naujo užpildomas. Šis procesas paprastesnis ir pigesnis nei kiti procesai, tačiau mažiau efektyvūs ilguoju laikotarpiu, nes biomasės augimas gali būti ribojimas.

Trečias biotechnologinių procesų tipas yra nenutrūkstamieji procesai. Jie veikia kai substratas ir kitos maisto medžiagos nuolat tiekiamos į bioreaktorių o galutinis produktas yra ištraukiamas tokiu pat srautu. Tai leidžia išgauti didelį galutinio produkto kiekį ir procesas atliekamas efektyviai, tačiau tokie procesai yra sunkiau suvaldomi, reikalinga pastovi kontrolė ir priežiūra.

2. Metodinė dalis

2.1. *E. coli* periodinio su pamaitinimu kultivavimo proceso modeliavimas

Periodinio su pamaitinimu kultivavimo proceso modelis yra pagrįstas "Model based design of a biochemical cultivation process" straipsniu [34]. Straipsnyje autoriai išsamiai aprašo ir pateikia mikrobiologinės fermentacijos proceso matematinį modelį. Procese vyksta periodinis su pamaitinimu (angl. fed-batch) tipo procesas, kuriame kultūros terpėje palaipsniui pridedama gliukozė, kad būtų palaikomas optimalus bakterijų augimas. Gliukozės srautas yra valdomas naudojant PI reguliatorių arba ANN valdiklį. Pagrindiniai parametrai yra specifinis gliukozės suvartojimas, kuris priklauso nuo gliukozės ir acetatų koncentracijos, bei specifinis augimo greitis, kuris yra susijęs su specifiniu gliukozės suvartojimu ir palaikymo koeficientu. Acetatų gamyba vyksta, kai specifinis augimo greitis yra didesnis arba lygus kritiniam augimo greičiui. Acetatų suvartojimas vyksta, kai specifinis augimo greitis yra mažesnis už kritinį augimo greitį, o acetatų koncentracija yra didesnė už 0. Deguonies suvartojimas ir anglies dvideginio išsiskyrimas yra apskaičiuojami remiantis specifiniu augimo greičiu ir acetatų suvartojimu. Masės balanso lygtys apskaičiuojamos biomasei, gliukozei, acetatui ir tirpalo svoriui, jos yra apskaičiuojamos remiantis proceso parametrais. Toliau pateikiami pradiniai parametrai, jų įtaka procesui ir vertės:

- Maksimalus gliukozės specifinis suvartojimo greitis (qSmax) yra rodiklis, kuris parodo maksimalų gliukozės kiekį, kurį mikroorganizmai gali panaudoti per tam tikrą laikotarpį palyginus su savo biomase. Jis matuojamas gramais gliukozės per gramą biomasės per valandą [g/g/h]. Šis parametras yra svarbus, nustatant mikroorganizmų metabolizmo ir augimo savybes. Tai leidžia įvertinti mikroorganizmų gebėjimą efektyviai panaudoti gliukozę energijai ir augimui. Maksimalaus gliukozės specifinio suvartojimo greičio pradinė vertė yra 2,125;
- monod konstanta gliukozei (Ks) yra parametras, kuris parodo gliukozės koncentraciją, kurioje mikroorganizmo augimo greitis yra lygus pusei maksimalios vertės. Ji matuojama gramais gliukozės litre [g/l]. Monod konstanta yra svarbus rodiklis, nustatantis mikroorganizmų augimo greičio priklausomybę nuo maisto šaltinio, šiuo atveju gliukozės koncentracijos. Šis parametras plačiai naudojamas bakterijų augimo ir metabolizmo aprašyme fermentacijos procesuose. Monod konstanta padeda nustatyti optimalias gliukozės koncentracijas ir maitinimo strategijas, siekiant užtikrinti efektyvų mikroorganizmų augimą ir galutinio produkto gamybą fermentacijos procese. Pradinė Monod konstantos vertė yra 0,119;
- monod konstanta acetatams (Ki) yra parametras, kuris parodo acetatų koncentraciją. Ji matuojama gramais acetatų litre [g/l]. Acetatų koncentracija procese veikia kaip šalutinis produktas, susidaro dėl mikroorganizmų metabolizmo. Monod konstanta leidžia nustatyti, kaip mikroorganizmų augimo greitis priklauso nuo acetatų koncentracijos. Didelė acetatų koncentracija gali slopinti mikroorganizmų augimą ir gliukozės panaudojimą, o kai kurie mikroorganizmai gali vartoti acetatą kaip papildomą energijos šaltinį. Monod konstanta acetatams padeda nustatyti optimalią acetatų koncentraciją, efektyviausią mikroorganizmų augimui ir acetato vartojimui. Pradinė Monod konstantos acetatams vertė yra 17,64;
- konversijos santykis gliukozei/biomasei (Ysx) yra parametras, kuris parodo, kiek gramų biomasės gaminama iš kiekvieno gramo gliukozės. Tai svarbus parametras, nes jis apibūdina, kaip efektyviai mikroorganizmai panaudoja gliukozę augimui ir dauginimuisi. Didelis konversijos santykis rodo, kad mikroorganizmai efektyviai panaudoja gliukozę biomasės augimui. Tačiau per didelis santykis gali sukelti problemas, tokių kaip gliukozės trūkumas,

aukštas šilumos išsiskyrimas ar pašalinių produktų atsiradimas. Pradinė konversijos santykio vertė yra 1,9075;

- palaikymo koeficientas (m) yra parametras, kuris parodo mikroorganizmų energijos sąnaudas, reikalingas ne augimui, bet vidinių funkcijų palaikymui. Tai gali lemti mažesnę galutinę biomasės koncentraciją fermentacijos procese. Pradinė palaikymo koeficiento vertė yra 0,0089;
- kritinis specifinis augimo greitis (miu_crit) yra specifinis augimo greitis, kuriame mikroorganizmų metabolizmas persijungia nuo vieno režimo į kitą, pvz., nuo acetatų gamybos į acetatų vartojimą. Kritinis specifinis augimo greitis taip pat gali paveikti proceso efektyvumą. Mikroorganizmai, kurių kritinis specifinis augimo greitis yra žemesnis, gali būti našesni energijos gamyboje, nes jie gali vartoti acetatą mažesniu augimo greičiu. Tačiau tai gali lemti mažesnį biomasės augimą ir galimai mažesnį galutinio produkto kiekį. Pradinė kritinio specifinio augimo greičio vertė yra 0,52;
- alpha_ap yra koeficientas, kuris nurodo acetatų gamybos santykį, kai mikroorganizmų augimo greitis yra didesnis už kritinį specifinį augimo greitį (miu_crit). Jis įtakoja acetatų gamybą fermentacijos metu. Kuo didesnis koeficientas, tuo daugiau acetatų bus gaminama, kai mikroorganizmų augimo greitis viršija kritinį specifinį augimo greitį. Tai leidžia kontroliuoti acetatų gamybą ir kaupimąsi proceso metu. Pradinė Alpha_ap vertė yra 0,53;
- alpha_ac yra koeficientas, kuris nurodo acetatų suvartojimo santykį, kai mikroorganizmų augimo greitis yra mažesnis už kritinį specifinį augimo greitį (miu_crit). Jis įtakoja acetatų suvartojimą fermentacijos metu. Kuo didesnis koeficientas, tuo daugiau acetatų bus suvartojama, kai mikroorganizmų augimo greitis yra mažesnis už kritinį specifinį augimo greitį. Tai leidžia kontroliuoti acetatų suvartojimą ir sumažinti jų koncentraciją, priklausomai nuo Alpha_ac vertės. Pradinė Alpha_ac vertė yra 0,885;
- maksimalus specifinis augimo greitis (miu_max) yra didžiausias greitis, kuriuo mikroorganizmai gali daugintis ir augti, kai jiems suteikiama pakankamai maisto ir tinkamos sąlygos. Svarbu stebėti ir reguliuoti mikroorganizmų augimo greitį, kad būtų išvengta potencialių problemų. Pradinė miu_max vertė yra 0,394;
- konversijos koeficientas (Yax) yra koeficientas, rodantis acetatų sąnaudas ir biomasės santykį. Jis veikia kaip sąsaja tarp acetatų suvartojimo ir biomasės augimo bei deguonies ir anglies dvideginio balanso. Didelis konversijos koeficientas reiškia, kad reikia daugiau acetatų norint pagaminti tokį patį kiekį biomasės. Tai gali išbalansuoti anglies dvideginio ir deguonies balansą bei biomasės augimo greitį. Konversijos koeficiento pradinė vertė yra 3,868.
- konversijos koeficientas (Yas) rodo acetatų gamybos santykį su gliukozės sąnaudomis. Netinkamas koeficiento parinkimas gali sukelti gliukozės sąnaudų balanso ir acetatų gamybos greičio disbalansą. Pradinė Yas vertė yra 0,994;
- substrato koncentracija (sf) daro įtaką gliukozės koncentracijai, apskaičiuojant masės balanso lygtį. Ji nustato, kiek gliukozės reikia tiekti į bioreaktorių kartu su maitinimo srautu ir reaktoriaus svoriu. Pradinė substrato koncentracijos vertė yra 300.
- garavimo greitis (e1) atspindi reaktoriaus skysčio tūrio sumažėjimą dėl garavimo. Tai naudojama apskaičiuojant bendrą masės srautą. Kuo didesnis garavimo greitis, tuo greičiau garuoja reaktoriaus skysčio tūris. Pradinė greičio vertė yra 0,015;
- mėginio paėmimo srautas (fsmp) yra parametras, rodantis kiek skysčio yra pašalinama iš proceso imant mėginius analizei. Pradinė mėginio paėmimo srauto vertė yra 0,05.

Toliau yra pateikiama modelio matematinė reprezentacija simuliacijos ir optimizavimo procedūrose:

– Specifinis gliukozės suvartojimo greitis:

$$q_s = q_{smax} * \frac{s}{K_s + s} * \frac{K_i}{K_i + a}; \quad (4)$$

čia: qs– konkretus gliukozės suvartojimo greitis; s – gliukozės koncentracija; Ks – Monod konstanta gliukozei; Ki – Monod konstanta acetatams; a – acetatų koncentracija;

– Specifinis augimo greitis:

$$\mu_s = \frac{q_s}{Y_{sx}} - m; \quad (5)$$

čia: μ_s – specifinis augimo greitis; q_s – specifinis gliukozės suvartojimo greitis; Y_sx - konversijos santykis gliukozei/biomasei; m – palaikymo koeficientas;

– Specifiniai acetatų gamybos ir suvartojimo greičiai

$$r_{ap} = \alpha_{ap}(\mu_s - \mu_{cr}), jei\ \mu_s \geq \mu_{cr}; \quad (6)$$

$$r_{ap} = 0, jei\ \mu_s < \mu_{cr};$$

čia: r_ap – specifinis acetatų gamybos greitis; α_ap – acetatų gamybos koeficientas; μ_s – specifinis augimo greitis; μ_cr – kritinis specifinis augimo greitis; (7)

$$r_{ac} = \mu_{amax} - \alpha_{ac} * \mu_s, jei\ \mu_s < \mu_{cr}; \quad (8)$$

$$r_{ac} = 0, jei\ \mu_s \geq \mu_{cr}\ arba\ a \leq 0; \quad (9)$$

čia: r_ac- specifinis acetatų suvartojimo greitis; μ_amax – maksimalus specifinis augimo greitis; α_ac – acetatų suvartojimo santykis; μ_s – specifinis augimo greitis; μ_cr – kritinis specifinis augimo greitis;

– Bendras masės srautas:

$$F = F_s + F_b - F_{evp} - F_{cl}; \quad (10)$$

čia: F – bendras masės srautas; F_s – maitinimo srauto greitis; F_b – bazės priedo greitis pH valdymui; F_evp – išgaravimo greitis; F_cl – masės netekimas dėl CO2 išsiskyrimo.

– Anglies dvideginio išmetimas:

$$CER = c_1 * x * \left(\mu_s + \frac{r_{ac}}{Y_{ax}}\right) + c_2 * x; \quad (11)$$

čia: CER – anglies dvideginio išmetimas; c_1 – modelio parametras; x – biomasės koncentracija; μ_s – specifinis augimo greitis; r_ac- specifinis acetatų suvartojimo greitis; Y_ax - konversijos santykis gliukozei/biomasei; c_2 – modelio parametras;

– Deguonies suvartojimas:

$$OUR = o_1 * x * \left(\mu_s + \frac{r_{ac}}{Y_{ax}}\right) + o_2 * x; \quad (12)$$

čia: OUR – deguonies suvartojimas; o_1 – modelio parametras; x – biomasės koncentracija; μ_s – specifinis augimo greitis; r_ac- specifinis acetatų suvartojimo greitis; Y_ax - konversijos santykis gliukozei/biomasei; o_2 – modelio parametras;

– Masės netekimas dėl anglies dvideginio išsiskyrimo:

$$F_{cl} = (CER - OUR) * w; \quad (13)$$

čia: F_cl – masės netekimas dėl CO2 išsiskyrimo; CER – anglies dvideginio išmetimas; OUR – deguonies suvartojimas; w – bioreaktoriaus svoris;

– Basės priedo greitis pH valdymui:

$$F_b = b_1 * x * \left(\mu_s + \frac{r_{ac}}{Y_{ax}}\right) * w; \quad (14)$$

čia: F_b – basės priedo greitis pH valdymui; b_1 – modelio parametras; x- biomasės koncentracija; μ_s – specifinis augimo greitis; r_ac- specifinis acetatų suvartojimo greitis; Y_ax - konversijos santykis gliukozei/biomasei; w – bioreaktoriaus svoris;

– Maitinimo srauto greitis:

$$F_s = f_s(t); \quad (15)$$

čia: F_s – maitinimo srauto greitis; f_s(t) – maitinimo srauto greitis laiko momentu;

– Išgaravimo greitis:

$$F_{evp} = e_1; \quad (16)$$

čia: F_evp – išgaravimo greitis; e_1 – išgaravimo greitis;

Masės balanso lygtys:

– Biomasės koncentracija:

$$\frac{dx}{dt} = \left(\mu_s + \frac{r_{ac}}{Y_{ax}} - \frac{F}{w}\right) * x; \quad (17)$$

čia: μ_s – specifinis augimo greitis; r_ac- specifinis acetatų suvartojimo greitis; Y_ax - konversijos santykis gliukozei/biomasei; F – bendras masės srautas; w – bioreaktoriaus svoris; x – biomasės koncentracija;

– Gliukozės koncentracija:

$$\frac{ds}{dt} = -\left(q_s + \frac{r_{ap}}{Y_{as}}\right) * x - \frac{F}{w} * s + \frac{F_s}{w} * S_f; \quad (18)$$

čia: qs– konkretus gliukozės suvartojimo greitis; : r_ap – specifinis acetatų gamybos greitis; Y_as - acetatų ir gliukozės konversijos santykis; x – biomasės koncentracija; F – bendras masės srautas; w – bioreaktoriaus svoris; s – gliukozės koncentracija; F_s – maitinimo srauto greitis; S_f – gliukozės koncentracijos įeinančiame sraute konstanta;

– Acetatų koncentracija:

$$\frac{da}{dt} = -\left(r_{ap} - r_{ac}\right) * x - \frac{F}{w} * a; \quad (19)$$

čia: r_ap – specifinis acetatų gamybos greitis; ; r_ac- specifinis acetatų suvartojimo greitis; x – biomasės koncentracija; F – bendras masės srautas; w – bioreaktoriaus svoris; a – acetatų koncentracija;

– Bioreaktoriaus svoris:

$$\frac{dw}{dt} = -\left(F - F_{smp}\right); \quad (20)$$

čia: F – bendras masės srautas; F_smp – mėginio paėmimo konstanta;

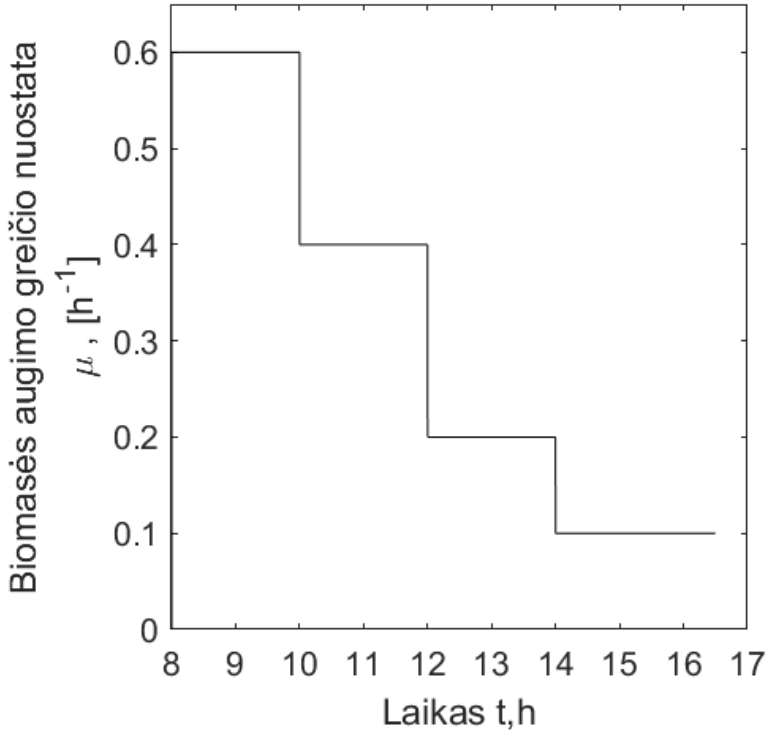
2.2. PI reguliatorius

Darbo tikslas - palaikyti nustatytą mikroorganizmų augimo greitį bioreaktoriuje. Šiam reguliavimui naudojamas pamaitinimo srautas F_s. Pamaitinimo srautas yra valdomas naudojant PI reguliatorių, kuris kontroliuoja fermentacijos procesą laikui bėgant. Taip pat sukuriami globalieji kintamieji: x, s, w, a, CER, OUR, miu, time, FS ir miu_setpoint, kurie yra naudojami visame kode. Pradiniai parametrai:

- Laiko diskretizavimo periodas c=0,001;
- pradinė laiko reikšmė tstart = 8;
- proceso laiko trukmė t = 16,5;
- proporcinis koeficientas P = 1;
- integracinis koeficientas I = 0,5;
- maksimalus gliukozės maitinimo srautas FSmax = 5,0;
- pradinis gliukozės maitinimo greitis F0s =0,1;

- μ vertės miu0 – miu4 ir skirtingi laiko momentai t2 – t5 μ nuostatai sukurti:
  - miu0=0,375;
  - miu1=0.6;
  - t2=10;
  - miu2=0.4;
  - t3=12;
  - miu3=0.2;
  - t4=14;
  - miu4=0.1;
  - t5=17;

Sukuriama μ nuostata, pagal kurią galima reguliuoti biomasės augimo greitį, valdant maitinimo srauto greitį Fs. Sukūrus μ nuostatą, gaunamas grafikas (žr. 9 pav.).



9 pav. μ nuostata laiko intervale

Iš grafiko matyti, kad sukurtos μ nuostatos vertės kinta nuo 0,6 iki 0,1 laikotarpiu nuo 8 iki 17 valandos. Pagal šią nuostatą PI reguliatorius turės reguliuoti maitinimo srauto greitį taip, kad biomasės augimo greitis atitiktų grafike pateiktą dinamiką.

PI reguliatorius pradeda veikti tik tada, kai gliukozės koncentracija yra ne didesnė nei 0,15. Jei sąlyga yra patenkinama, apskaičiuojama klaida – paklaida tarp nustatytos ir faktinės μ reikšmės.

[35] straipsnyje pateikiamos prognozavimo paklaidų formulės: MAE (angl. mean absolute error) - vidutinė absoliutinė, MAPE (angl. mean absolute percentage error) - vidutinė absoliutinė procentinė, RMSE (angl. root mean square error) - kvadratinė. Šios (21-23) formulės bus naudojamos reguliatoriaus ir valdiklio darbo kokybei palyginti.

$$MAE = \frac{\sum_{i=1}^{n} |\hat{y}_i - y_i|}{n}; \quad (21)$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{\hat{y}_i - y_i}{y_i}\right|; \quad (22)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}; \quad (23)$$

Sukuriamas triukšmas, siekiant patikrinti, ar sukurtas PI reguliatorius palaiko μ reikšmę pagal sukurtą nuostatą esant trikdžiams. Iš pradžių apskaičiuojama μ reikšmė, sumažinta ir padidinta 2 procentais, po to tarp šių ribų parenkamas atsitiktinis skaičius. Po atsitiktinės vertės sukūrimo, ji pridedama prie esamos μ reikšmės. Šis triukšmo modeliavimas leidžia efektyviau patikrinti, ar sukurtas PI reguliatorius veikia tinkamai ir ar gali reguliuoti μ reikšmę pagal nustatytą nuostatą (24-26 formulės).

$$T_a = \mu * (1 - 0{,}002); \quad (24)$$

$$T_B = \mu * (1 + 0{,}002); \quad (25)$$

$$T = (T_a + (T_B - T_a) * R) + \mu; \quad (26)$$

čia: T_a – -2% μ vertės; T_b – +2% μ vertės; μ – biomasės augimo greitis; R – atsitiktinis skaičius tarp 0 ir 1; T – triukšmas.

Aprašomas PI reguliatoriaus veikimas. Atnaujinama klaidų suma, vadinama error_sum, kuri atitinka klaidos integralą. Klaidų suma atnaujinama pridedant dabartinę klaidą, padaugintą iš žingsnio dydžio c, apskaičiuotą pagal (27) formulę. Apskaičiuojama nauja maitinimo srauto greičio reikšmė pagal PI reguliatoriaus formulę – dabartinė klaida padauginama iš proporcinio koeficiento P, o klaidos suma – iš integracinio koeficiento I. Šių reikšmių suma sudaro naują Fs vertę, apskaičiuotą pagal (28) formulę.

Reguliatoriaus darbui sudaryti ribojimai – jei maitinimo srauto greitis Fs yra mažesnis už 0, jis bus lygus 0, kadangi gamtoje negali būti neigiamų reikšmių. Taip pat tai padeda išvengti papildomų klaidų atliekant reguliatoriaus reguliavimą, kas apskaičiuota pagal (29) formulę. Yra nustatyta maksimali Fs reikšmė, kurią viršijus, Fs bus lygi Fsmax=5, apskaičiuota pagal (29) formulę. Šis reguliatoriaus darbas vyksta per ciklą, kuriame atliekamos iteracijos per visus laiko žingsnius nuo tstart=8 iki t=16.

$$E_s = E_s + (E * c); \quad (27)$$

$$Fs = E * P + E_s * I; \quad (28)$$

$$0 < Fs \leq 5; \quad (29)$$

čia: E_s – klaidų suma; E – klaida; c – žingsnių skaičius; F_s – pamaitinimo srauto greitis; P – proporcinė dedamoji; I – integralinė dedamoji.

Apskaičiavus Fs reikšmę iškviečiama „Balanceprocess“ .m failo praėjusio žingsnio kintamieji – x, s, a, w kartu su nauja Fs reikšme. Atnaujinami kintamieji pagal naują Fs reikšmę – atnaujinta reikšmė yra lygi praėjusio laiko žingsnio reikšmei pridėjus kintamojo pokytį per laiką padaugintą iš žingsnio dydžio c, apskaičiuota pagal (30-36) formules:

$$x_i = x_{(i-1)} + \left(\mu_{(i-1)} + \left(\frac{rac_{(i-1)}}{Yax}\right) - \left(\frac{F_{(i-1)}}{w_{(i-1)}}\right)\right) * x_{(i-1)} * c; \quad (30)$$

$$s_i = s_{(i-1)} + \left(-\left(q_{S(i-1)} + \left(\frac{rap_{(i-1)}}{Yas}\right)\right) * x_{(i-1)} - \left(\frac{F_{(i-1)}}{w_{(i-1)}}\right) * s_{(i-1)} + \left(\frac{Fs(i)}{w_{(i-1)}}\right) * sf\right) * c; \quad (31)$$

$$a_i = a_{(i-1)} + \left(\left(rap_{(i-1)} - rac_{(i-1)}\right) * x_{(i-1)} - \left(\frac{F_{(i-1)}}{w_{(i-1)}}\right) * a_{(i-1)}\right) * c; \quad (32)$$

$$w_i = w_{(i-1)} + \left(F_{(i-1)} - F_{smp}\right) * c; \quad (33)$$

$$miu_i = \frac{q_{s_i}}{Y_{sx}} - m; \quad (34)$$

$$OUR_i = o_1 * \left(miu_i + \left(\frac{rac_i}{Y_{ax}}\right)\right) * x_i + o_2 * x_i; \quad (35)$$

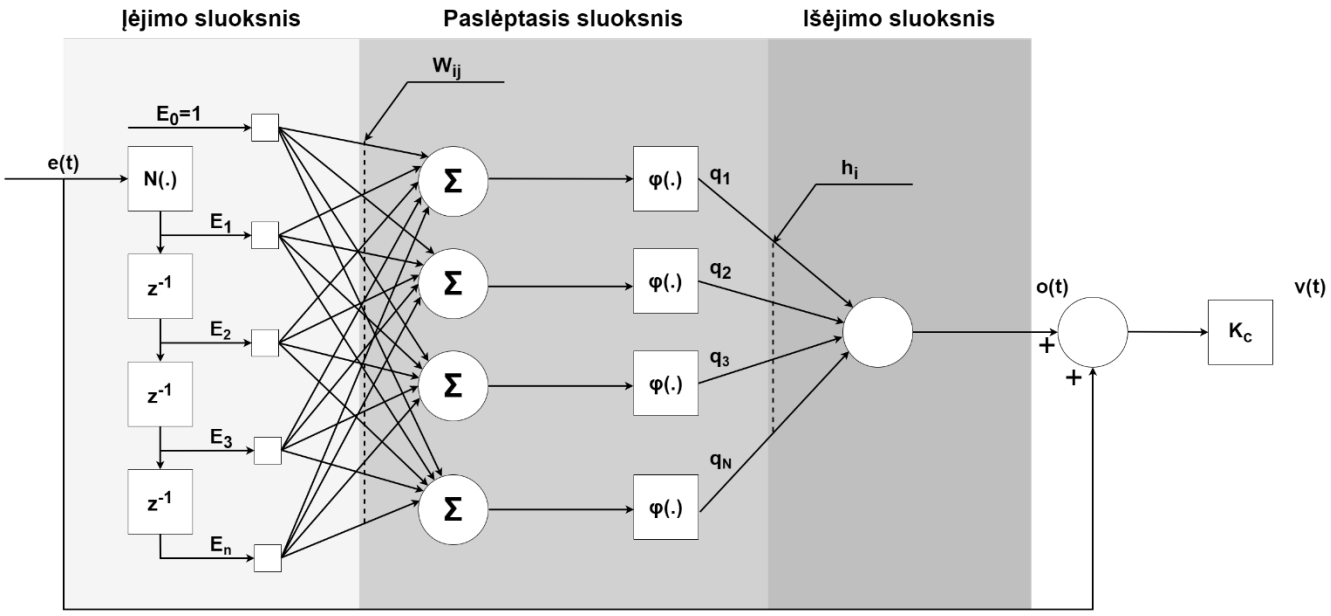
$$CER_i = c_1 * \left(miu_i + \left(\frac{rac_i}{Y_{ax}}\right)\right) * x_i + o_2 * x_i; \quad (36)$$

2.3. Neuroninis valdiklis

Specifiniam augimo greičiui valdyti, keičiant pamaitinimo greičio srautą, taip pat naudojamas neuroninis valdiklis ANN (angl. Artificial Neural Network), kuris paremtas [36] straipsniu. Straipsnyje autoriai pateikia išsamų MFA (angl. Model Free Adaptive) neuroninio valdiklio modelį, aprašydami matematines formules. ANN valdikliui sukuriamas atskiras failas, pavadinimu „ProcessANN“. Įdiegti globalūs kintamieji, pradinės vertės, laikas, matricos, paklaidų skaičiavimas - viskas vyksta identiškai kaip ir su PI reguliatoriumi. Įvedamos pradinės neuroninio valdiklio vertės:

- – Mokymosi greitis (Eta): 0,2;
- – Svorio koeficientas: 0,12;
- – Neuronų skaičius (neurons): 5.

Proceso kriterijai, laiko žingsniai ir nuostatos yra tokie patys, kaip ir PI reguliatoriaus procese, o triukšmo dydis yra identiškas. Taip pat, kaip ir PI reguliatorius, ANN pradeda veikti tik tuomet, kai gliukozės koncentracija yra mažesnė už 0,15.



10 pav. ANN struktūros schema [36]

ANN valdiklis apskaičiuoja klaidą (error) tarp nuostatos (miu_setpoint) ir realios miu vertės. Ši klaida naudojama ANN valdiklio apmokymui. Klaidos (e) ir svorių (weight) masyvai atnaujinami naudojant gautą klaidą (error). Taip pat atsižvelgiama į mokymosi greitį (Eta), valdymo lygties koeficientą (Kc) ir neuronų skaičių. Apskaičiuojamas kiekvieno neurono išvesties signalas (q) naudojant sigmoidinę funkciją. Sigmoidinė funkcija suteikia netiesinę funkciją neuronams, leidžiant jiems modeliuoti sudėtingesnes sąsajas tarp įvesties ir išvesties. Apskaičiuojama išvesties suma (o) naudojant kiekvieno neurono išvesties signalą (q) ir išvesties sluoksnio svorius (h). Atnaujinimo taisyklė grindžiama šaltinio klaidos gradiento mažinimu, siekiant optimizuoti ANN valdiklio veikimą. Apskaičiuojamas naujas pamaitinimo greičio srautas pagal atnaujintą ANN valdiklį, naudojant klaidą (error) ir išvesties signalą (o).

$$v(t) = K_c\big(o(t) + e(t)\big); \tag{37}$$

čia v(t) - nuolatinė funkcija valdymo išvesties (valdymo kintamųjų) atžvilgiu; K_c - reguliatoriaus stiprumo koeficientas; o(t) –išvesties vertė; e(t) – stebima klaida.

$$\varphi(x) = \frac{1}{1 + e^{-x}}; \tag{38}$$

čia φ – sigmoidinė funkcija; x - n-ojo aktyvacijos funkcijos įvestis paslėptame sluoksnyje;

Kode sigmoidinė funkcija yr naudojama apskaičiuoti q_n reikšmę. Toliau pateikiama formulės išraiška kode:

$$q(j) = 1 / (1 + exp(-p(j)));$$ (39)

$$p_j(n) = \sum_{i=0}^{N} w_{ij}(n) \frac{1}{2} e(t)^2; \quad (40)$$

čia p_j – įėjimas n-osios aktyvavimo funkcijos paslėptame sluoksnyje; w_i – svoris paslėptame sluoksnyje; e – stebima klaida;

Šia formule yra apskaičiuojamas ANN įėjimas aktyvavimo funkcijos paslėptame sluoksnyje. Jis gaunamas sumuojant užvėlintą paklaidą iš svorio paslėptame sluoksnyje. Toliau yra pateikiama formulė kodo išraiška:

$$p(j) = p(j) + weight(l, j) * (e(l)\text{^}2 * 0.5); \quad (41)$$

$$q_j(n) = \varphi\left(p_j(n)\right); \quad (42)$$

Čia q – aktyvavimo funkcijos išėjimas paslėptame sluoksnyje; p – aktyvavimo funkcijos įėjimas paslėptame sluoksnyje;

Šia formule įėjimo vertės paslėptame sluoksnyje yra konvertuojamos naudojant sigmoidinę funkciją – (38) formulė, ir gaunamas išėjimas q.

$$o(n) = \sum_{j=0}^{N} h_j(n) q_j(n); \quad (43)$$

čia o – išvesties vertė; h – išėjimo sluoksnio svoris; q - – aktyvavimo funkcijos išėjimas paslėptame sluoksnyje.

Šia formule yra apskaičiuojama išvesties vertė. Ji gaunama paslėpto sluoksnio išėjimą dauginant su paslėpto sluoksnio svoriu. Toliau pateikiama formulė kodo išraiška:

$$o = o + h(j).* q(j); \quad (44)$$

Apskaičiavus maitinimo srauto greitį atitinkamai pagal klaidą error yra atnaujinami svoriai tikslesniam miu keitimui pagal nuostatą. Svoriai yra atnaujinami pagal pateiktas (45-46) formules.

$$\Delta w_{ij}(n) = \eta K_c e(n) q_j(n)(1 - q_j(n)) E_i(n) \sum_{k=0}^{N} h_k(n); \quad (45)$$

$$\Delta h_j(n) = \eta K_c e(n) q_j(n); \quad (46)$$

Čia η – mokymosi greitis; K_c – reguliatoriaus stiprumo koeficientas; išvesties vertė; e – stebima klaida; q - aktyvavimo funkcijos išėjimas paslėptame sluoksnyje; E – klaidos signalas; h – išėjimo sluoksnio svoris.

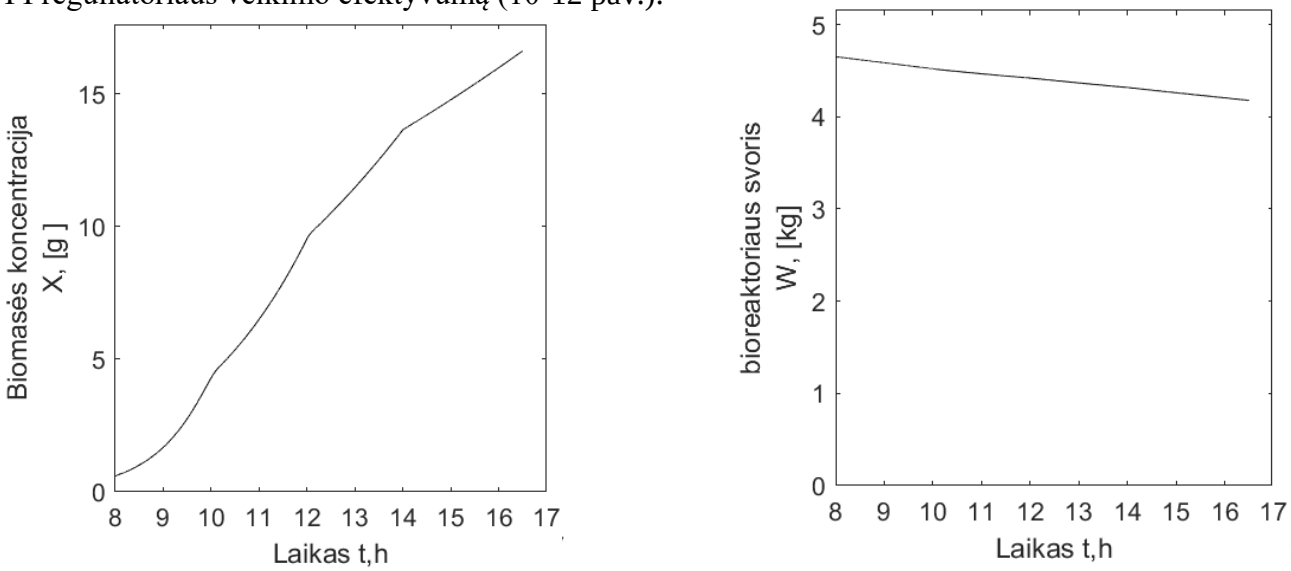
Svorio atnaujinimo formulė yra grindžiama gradiento nuostolių optimizavimo metodu – atnaujinant svorius pagal klaidos funkcijos gradientą, kad būtų sumažinta bendra klaida. Svorio atnaujinimui naudojama delta taisyklė, kuri sudaryta iš klaidos, sigmoidinės funkcijos išvesties mokymosi greičio.

Tikslas yra sumažinti klaidą tarp numatytųjų reikšmių ir tikrųjų, o paslėptojo sluoksnio neuronai atsakingi už modelio sudėtingumo valdymą.

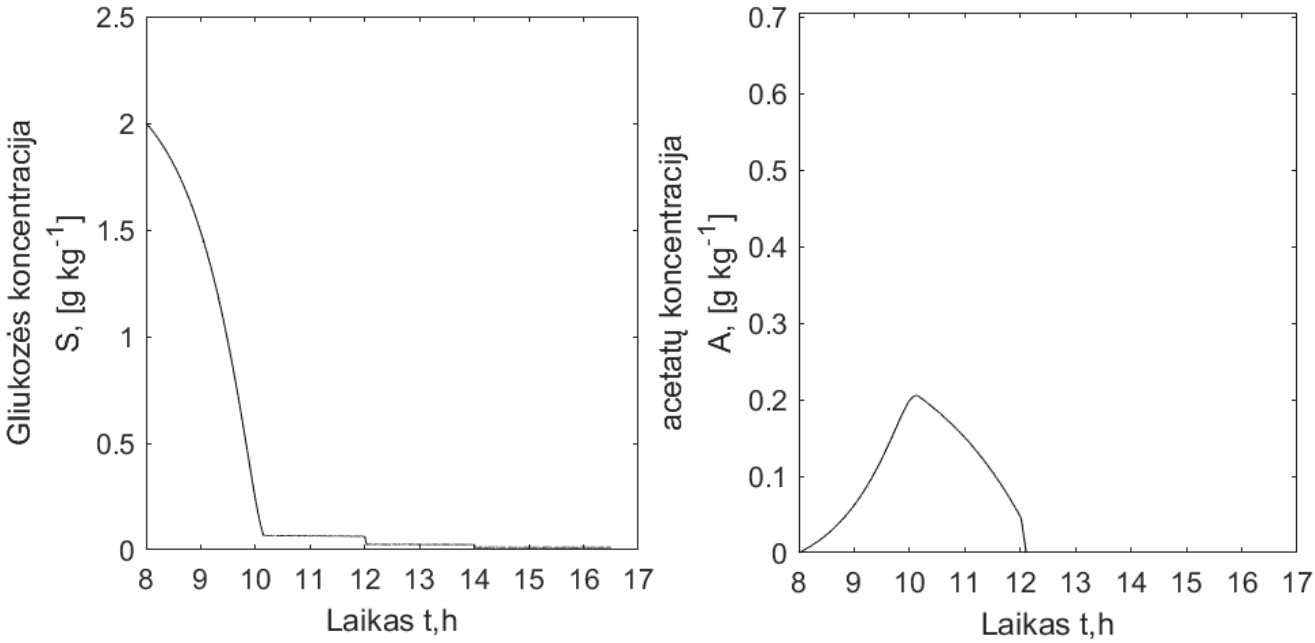
3. Eksperimentinė dalis

3.1. PI reguliatorius

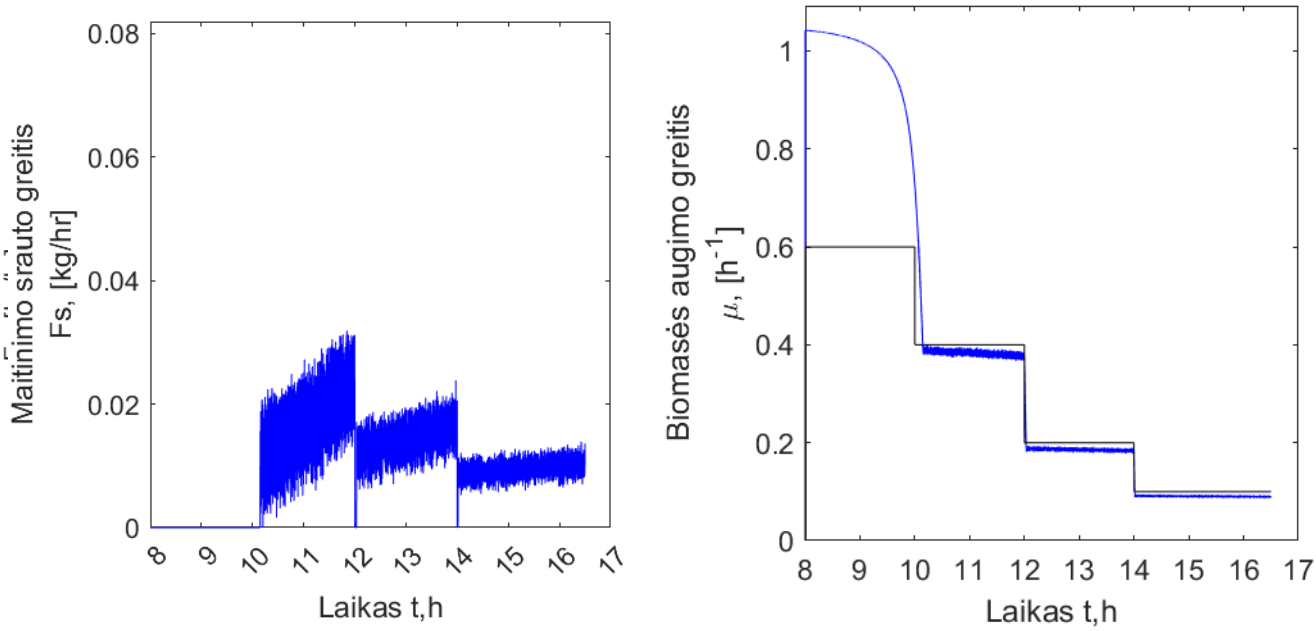
Pasitelkus sukurtą PI reguliatorių, inicijuojamas procesas, kurio veiklą galima stebėti per vizualizuotus grafikus. Šie grafikai atspindi proceso kintamųjų dinamiką ir leidžia detaliai analizuoti PI reguliatoriaus veikimo efektyvumą (10-12 pav.).



11 pav. biomasės koncentracijos ir bioreaktoriaus masės grafikai esant triukšmui naudojant PI reguliatorių



12 pav. Gliukozės ir acetatų koncentracijų grafikai esant triukšmui naudojant PI reguliatorių



13 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai esant triukšmui naudojant PI reguliatorių

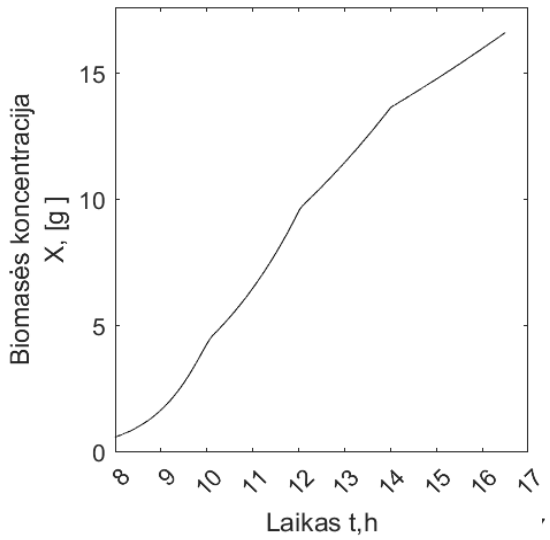
PI reguliatoriaus paklaidų vertės esant triukšmui:

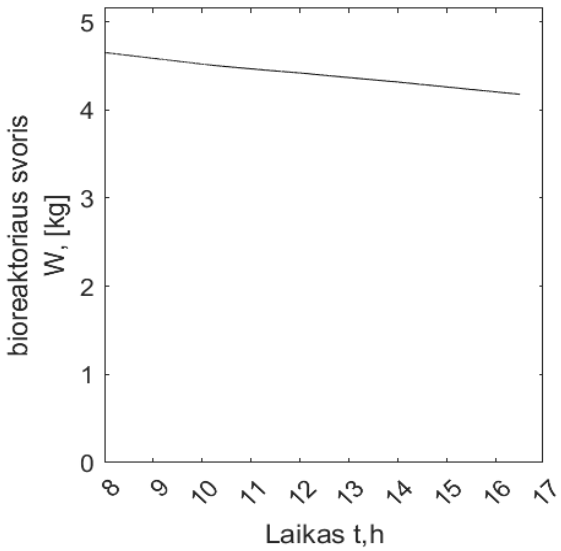
MAE: 0,0081;

MAPE: 2,7891;

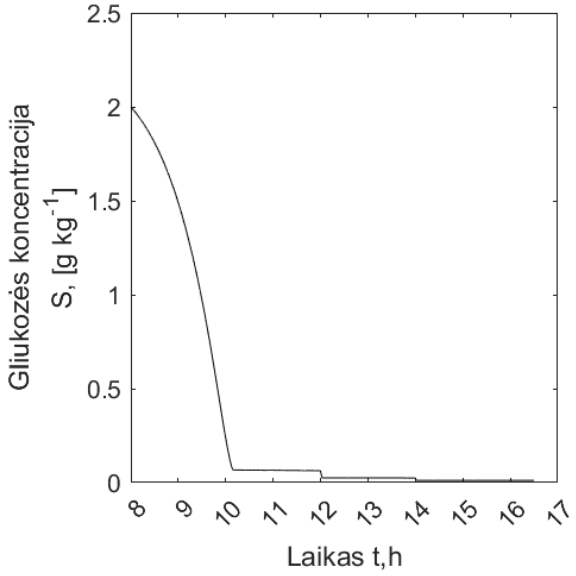
RMSE: 0,0183.

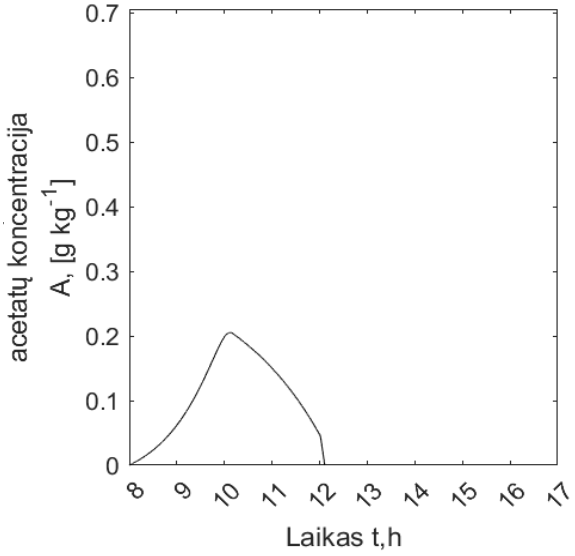
Bandymas su tomis pačiomis sąlygomis yra pakartojamas be jokių trikdžių, siekiant išsiaiškinti, kokia įtaka proceso veiklai yra triukšmo faktoriaus. Gaunami grafikai, suteikiantys galimybę detaliau analizuoti procesą ir jo kintamųjų dinamiką (13-15 pav.).



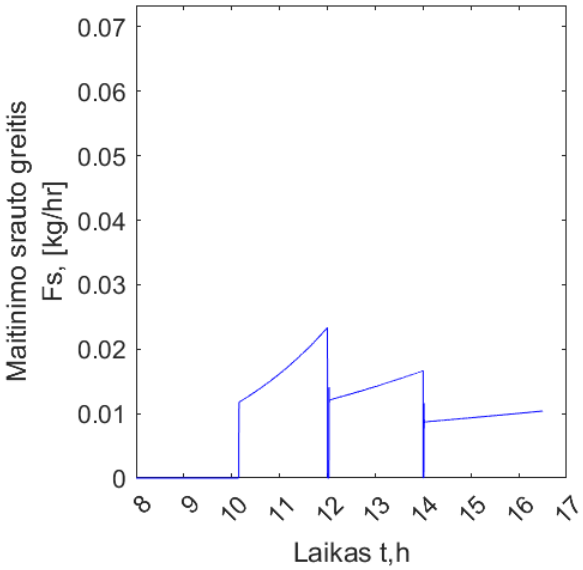


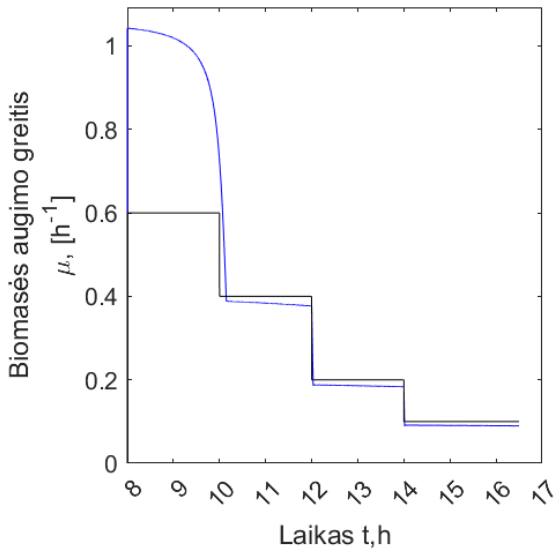
14 pav. biomasės koncentracijos ir bioreaktoriaus masės grafikai naudojant PI reguliatorių





15 pav. Gliukozės ir acetatų koncentracijų grafikai naudojant PI reguliatorių





16 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant PI reguliatorių

PI reguliatoriaus paklaidų vertės be triukšmo:

MAE: 0,0080;

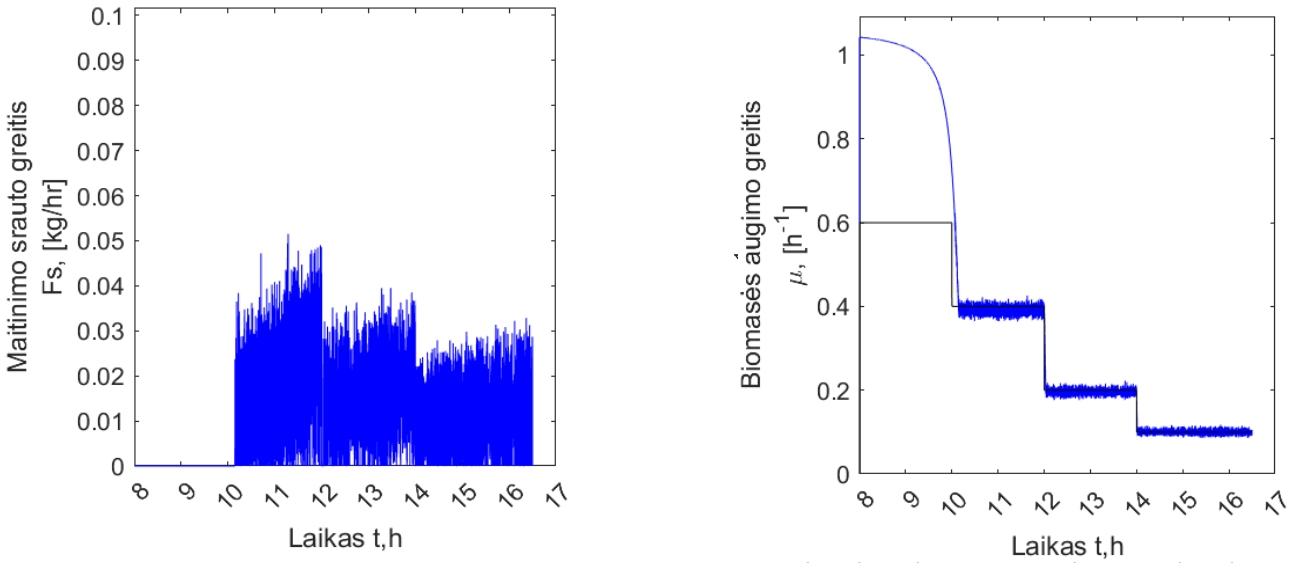
MAPE: 2,6729;

RMSE: 0,0182.

Bandymai yra pakartojami su kitais PI reguliatoriaus pradiniais parametrais, siekiant palyginti gautus rezultatus ir ištirti, kokią įtaką procesui daro šie parametrai. Gauti grafikai pateikti (16 pav.). Pakeisti pradiniai PI reguliatoriaus parametrai:

Proporcinis koeficientas P = 2;

integracinis koeficientas I = 1.



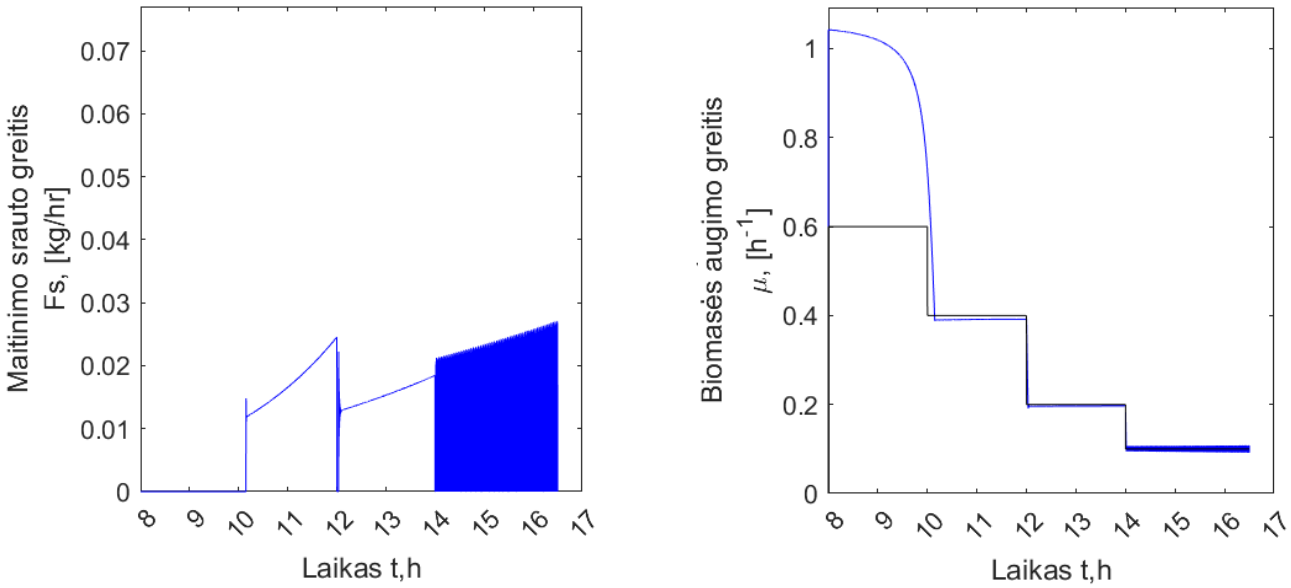
17 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant PI reguliatorių su pakeistais pradiniais parametrais ir esant triukšmui

Atlikus bandymą su modifikuotais pradiniais parametrais ir kuomet sistema veikia esant triukšmui, gaunamos šios paklaidos:

MAE: 0,0075;

MAPE: 3,4006;

RMSE: 0,0177.



18 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant PI reguliatorių su pakeistais pradiniais parametrais

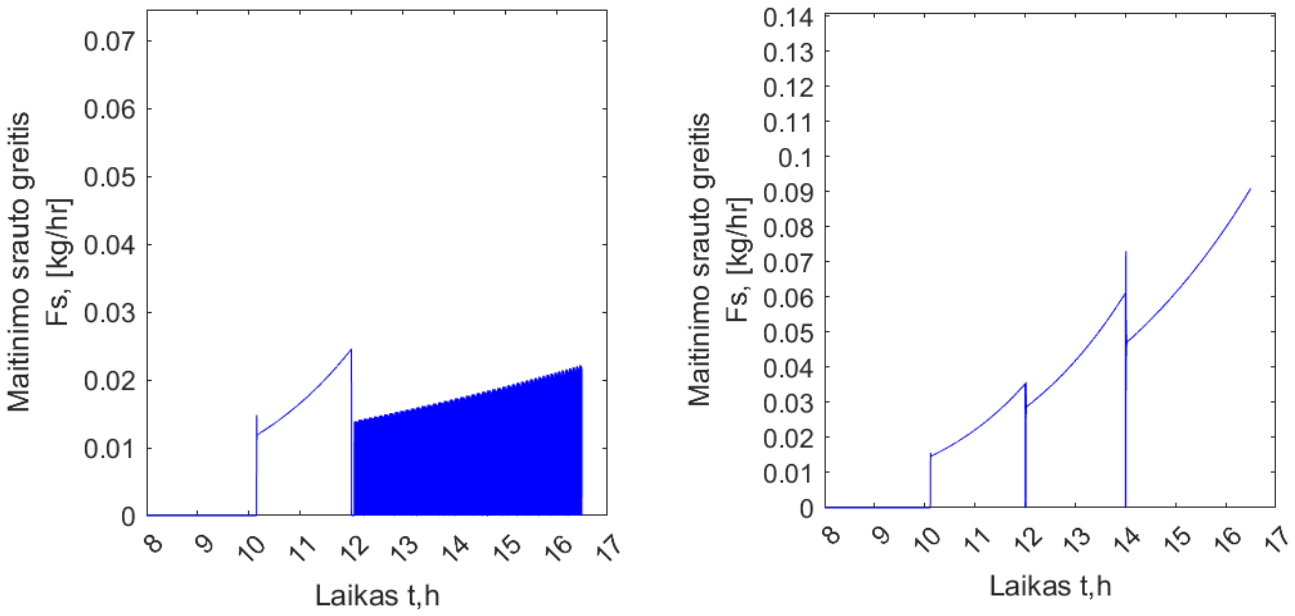
Atlikus bandymą su modifikuotais pradiniais parametrais ir kuomet sistema veikia be triukšmo, gaunamos šios paklaidos:

MAE: 0,0073;

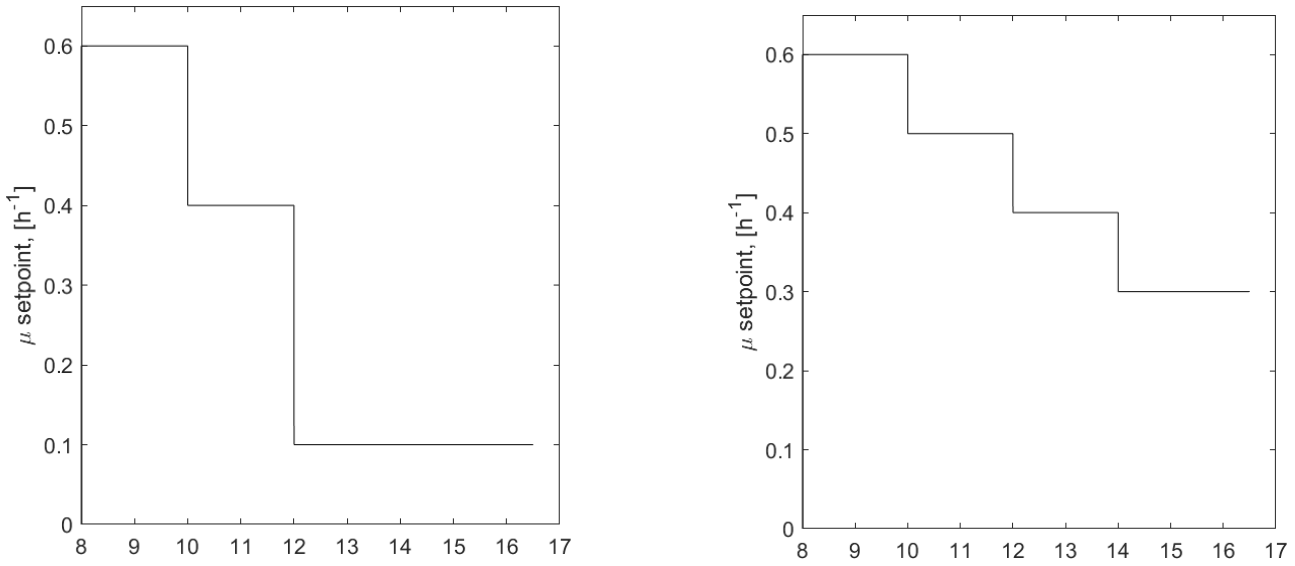
MAPE: 3,3877;

RMSE: 0,0174.

Iš (17 pav.) matoma, kad nuo 14 valandos, kai nuostata keičiama į 0,1, atsiranda didesni svyravimai nei esant kitiems nuostatos pakeitimams. Tai gali atsirasti dėl dviejų priežasčių. Pirmoji – nuostatos pokytis 14 valandą yra mažesnis nei kiti pokyčiai. PI reguliatorius paprastai geriau veikia, kai pokyčiai yra didesni, nes tai leidžia jam geriau prisitaikyti prie naujos situacijos. Per mažas pokytis gali sukelti didesnius svyravimus, nes reguliatorius gali neturėti pakankamai reguliavimo galios, kad efektyviai prisitaikytų prie proceso pokyčių šiame etape. Antra priežastis gali būti ta, kad anksčiau nustatytos nuostatos buvo gerai pritaikytos PI reguliatoriui, tačiau pakeitus nuostatą į 0,1, reguliatoriaus parametrai tampa netinkami. Siekiant išsiaiškinti, kas sukelia svyravimus, atliekami du bandymai. Pirmajame bandyme visi nuostatos pokyčiai pakeičiami po 0,1, taip patikrinant, ar per mažas nuostatos pokytis sukelia svyravimus. Antrajame bandyme miu4 ir miu3 pakeičiami į tą pačią nuostatą – 0,1, taip patikrinant, ar būtent specifinė nuostata sukelia svyravimus.



19 pav. Pamaitinimo srauto greičių grafikai esant specifiniai nuostatai ir vienodiems nuostatų pokyčiams

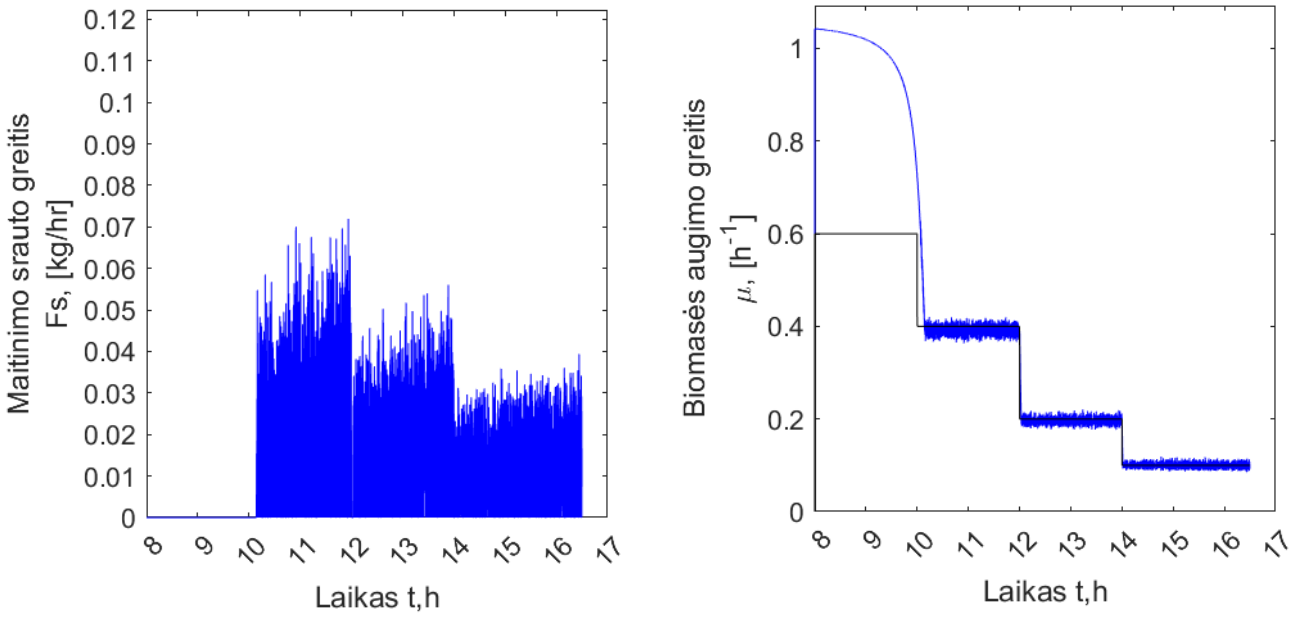


20 pav. Specifinės nuostatos ir vienodų nuostatų pokyčių grafikai

Pastebėta, kad būtent specifinė nuostata – 0,1 sukelia ypač didelius svyravimus. Norint išspręsti šią problemą, reikėtų keisti PI reguliatoriaus parametrus skirtingais laikotarpiais esant specifinėms nuostatoms. Tačiau šie papildomi pakeitimai apsunkina proceso valdymą.

Atliekamas dar vienas bandymas, kuriame sukeliamas didesnis triukšmas norint patikrinti reguliatoriaus proceso valdymo kokybę. Gaunami rezultatai pateikiami (20 pav.). Pradiniai duomenys imami iš pirmojo bandymo, o triukšmo lygis padidinamas dvigubai, iki 4%. Gautos paklaidos yra šios:

MAE: 0,0076;
MAPE: 4,0806;
RMSE: 0,0215.



21 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant PI reguliatorių esant didesniam triukšmui

Atlikus bandymus, pastebima, kad esant triukšmui, maitinimo srauto greitis kiek skirtasi (13, 16 pav.). Nepaisant to, specifinis augimo greitis lieka arti nustatytos nuostatos. Kai proporcinis koeficientas yra lygus 1, o integracinis - 0,5, triukšmo įtaka yra gana maža: MAE paklaidos skirtumas tarp situacijos su triukšmu ir be jo yra 0,0001, MAPE - 0,1162, o RMSE - 0,0001. Kai proporcinis koeficientas yra lygus 2, o integracinis - 1, paklaidų verčių skirtumai esant triukšmui ir be jo yra tokie: MAE - 0,0002, MAPE - 0,0129, RMSE - 0,0003. Tačiau reikėtų paminėti, kad pradinės parametrų vertės buvo parinktos geriau: lyginant su kitomis parametrų vertėmis, paklaidų skirtumai yra tokie:

su triukšmu:

- MAE – 0,006;
- MAPE – 0,6115;
- RMSE – 0,006;

Be triukšmo: MAE – 0,007;

- MAPE – 0,70879;
- RMSE – 0,0008.

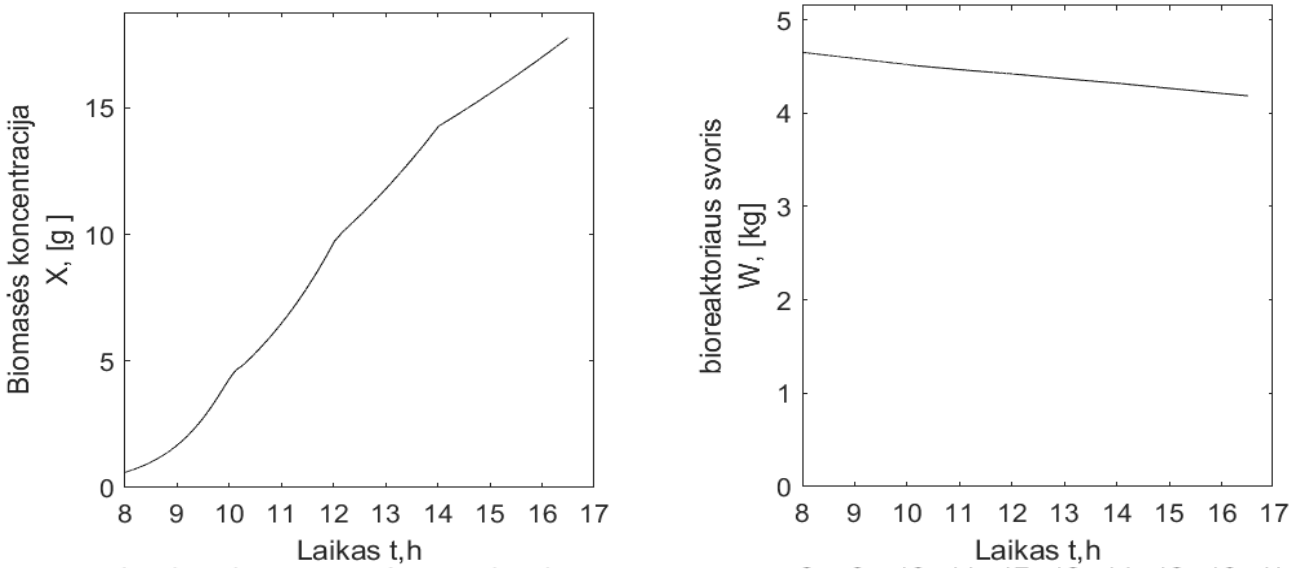
Galima teigti, kad nepaisant triukšmo poveikio, PI reguliatorius beveik pasiekia norimą specifinio augimo greitį, tačiau vis tiek kyla paklaidų. Siekiant geresnių rezultatų, būtina tobulinti PI reguliatoriaus derinimą. Visos bandymų metu gautos paklaidos yra pateiktos (1) lentelėje.

1 lentelė. Neuroninio valdiklio ir PI reguliatoriaus paklaidos

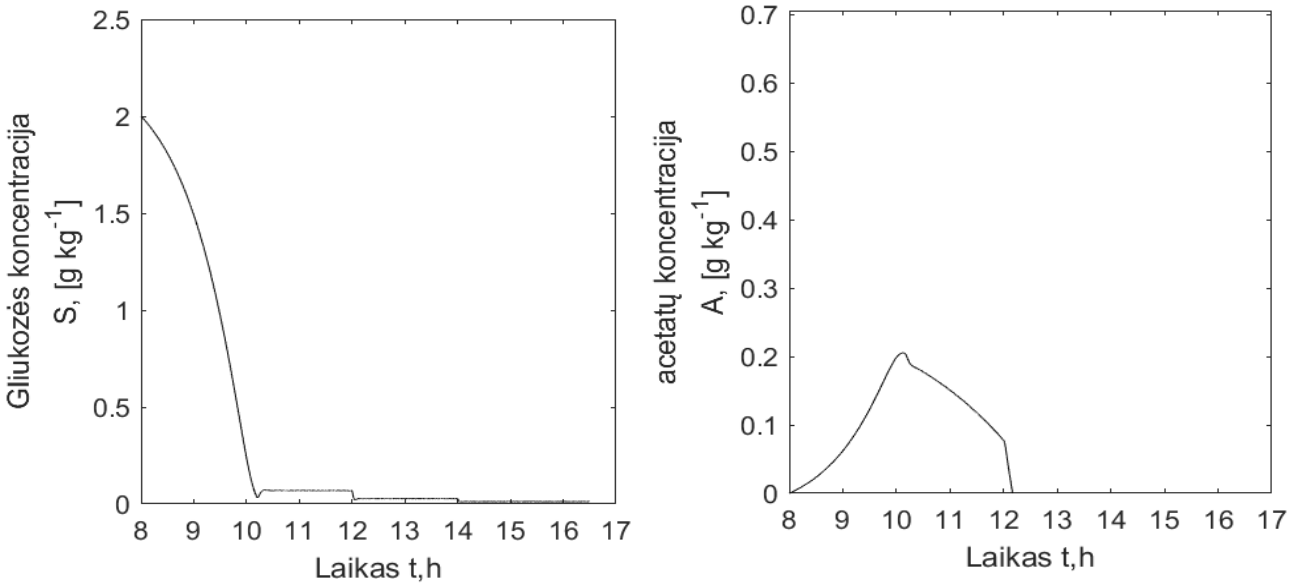
Paklaidos							
1 bandymas				2 Bandymas			
PI reguliatorius		Neuroninis valdiklis		PI reguliatorius		Neuroninis valdiklis	
Su triukšmu				Su triukšmu			
MAE	0,0081	MAE	0,0052	MAE	0,0075	MAE	0,0066
MAPE	2,7891	MAPE	1,8706	MAPE	3,4006	MAPE	2,3922
RMSE	0,0183	RMSE	0,0231	RMSE	0,0177	RMSE	0,025
Be triukšmo				Be triukšmo			
MAE	0,008	MAE	0,0055	MAE	0,0073	MAE	0,0067
MAPE	2,6729	MAPE	1,8932	MAPE	3,3877	MAPE	2,4006
RMSE	0,0182	RMSE	0,0231	RMSE	0,0174	RMSE	0,025
Su padidintu triukšmu							
MAE	0,0076	MAE	0,0053				
MAPE	4,0806	MAPE	1,8357				
RMSE	0,0215	RMSE	0,0232				

3.2. Neuroninis valdiklis

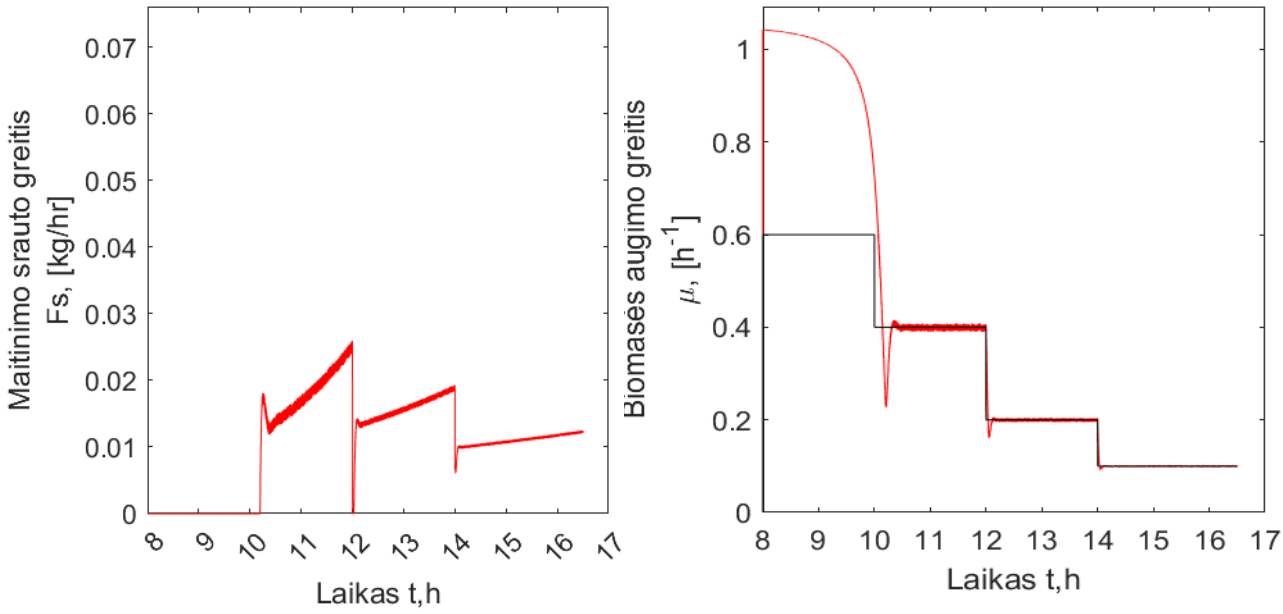
Po ANN valdiklio aprašymo yra paleidžiamas procesas, kurio metu gaunami grafikai ir paklaidos(22-27 pav.). Atliekami trys bandymai, kurių metu keičiamos pradinių parametrų vertės ir vertinama triukšmo įtaka. Pirmieji du bandymai yra atliekami su ir be triukšmo, siekiant įvertinti valdiklio veikimą skirtingomis pradinių parametrų sąlygomis. Trečiasis bandymas yra skirtas padidinti triukšmo lygį ir tikslingai stebėti, kaip šis triukšmas veikia procesą.



22 pav. biomasės koncentracijos ir bioreaktoriaus masės grafikai naudojant ANN valdiklį esant triukšmui



23 pav. Gliukozės ir acetatų koncentracijų grafikai naudojant ANN valdiklį esant triukšmui



24 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant ANN valdiklį esant triukšmui

Gaunamos MAE, MAPE ir RMSE paklaidos esant trikdžiams:

MAE: 0,0052;

MAPE: 1,8706;

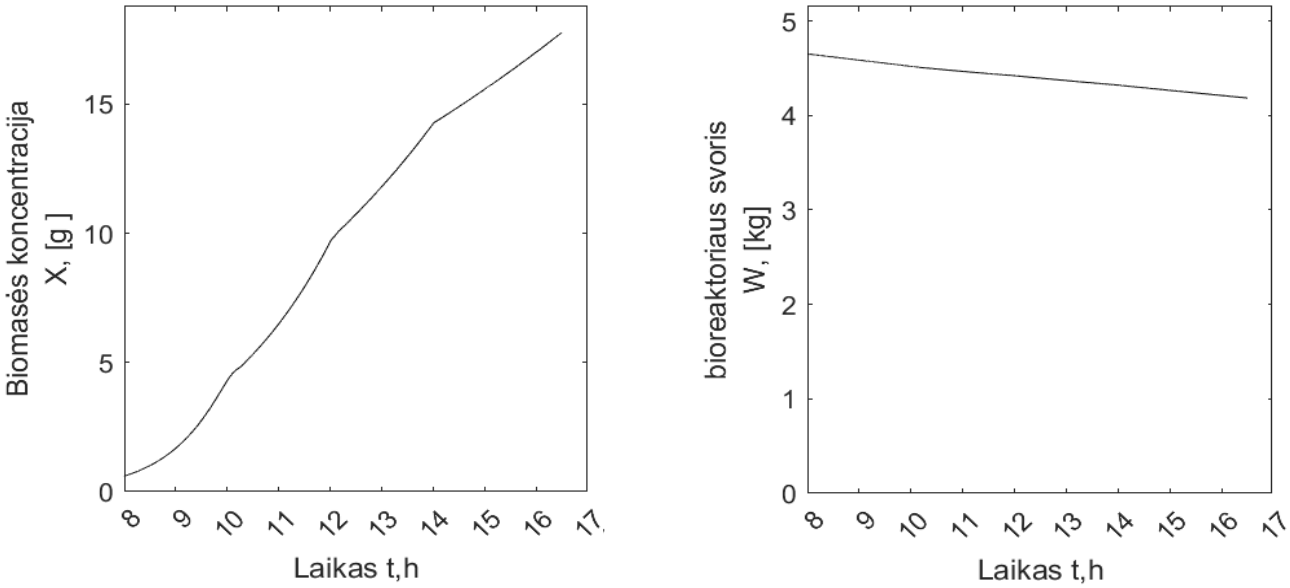
RMSE: 0,0231.

Atliktas bandymas su neuroniniu valdikliu neįtraukiant triukšmo. Tokiu atveju gaunamos šios paklaidos:

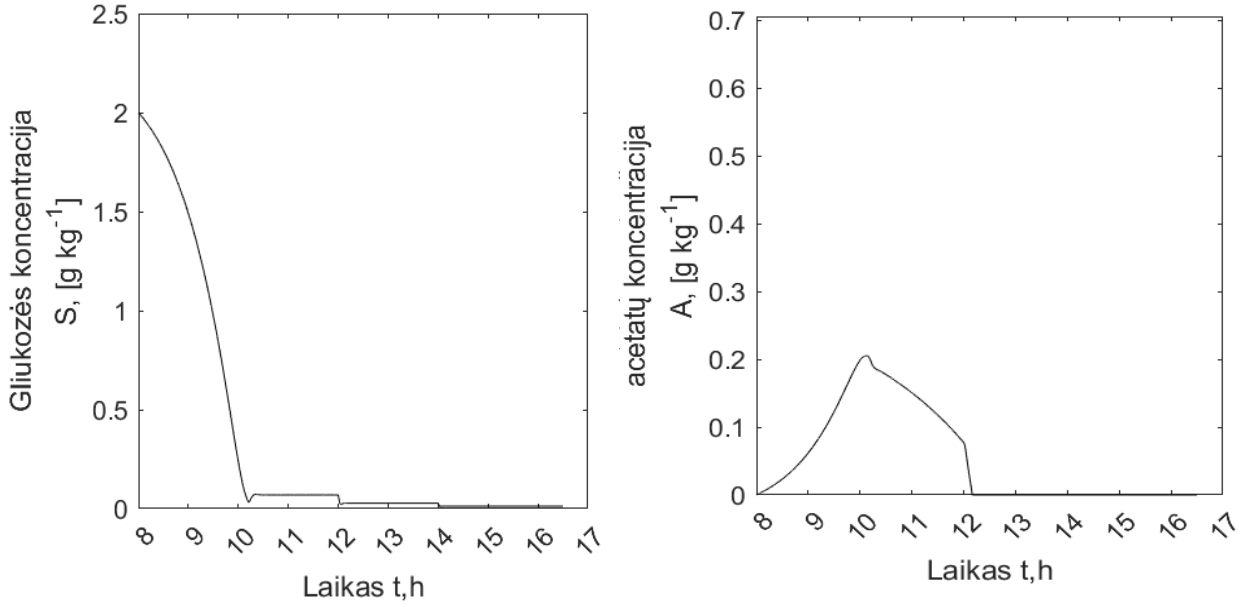
MAE: 0,0055;

MAPE: 1,8932;

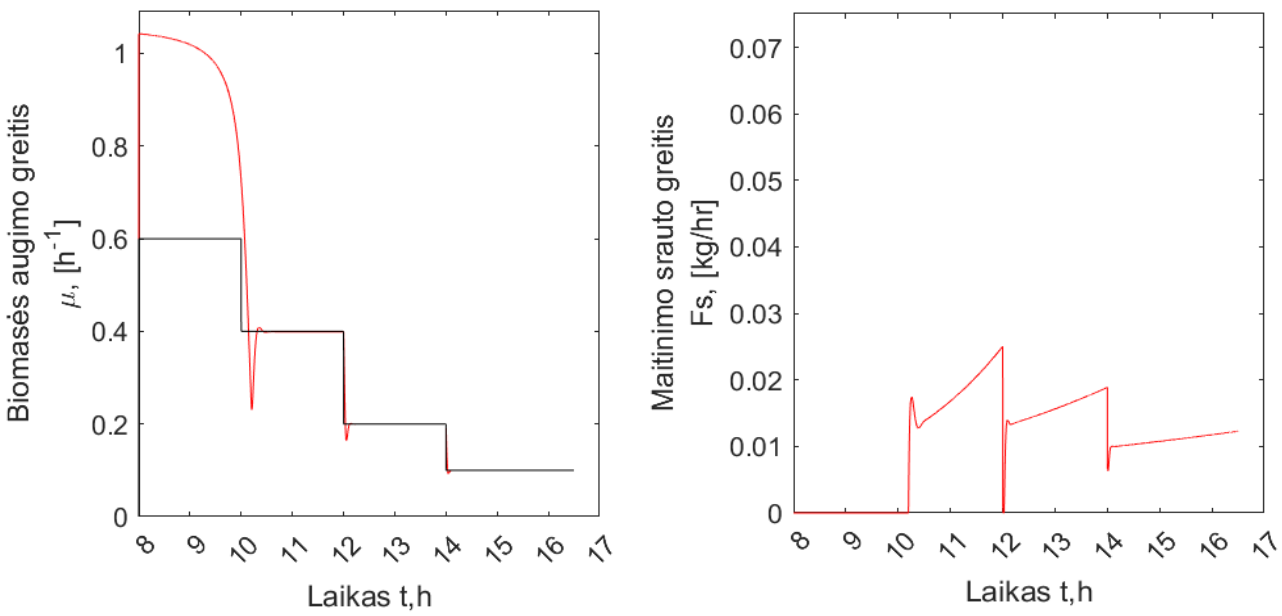
RMSE: 0,0231.



25 pav. biomasės koncentracijos ir bioreaktoriaus masės grafikai naudojant ANN valdiklį



26 pav. Gliukozės ir acetatų koncentracijų grafikai naudojant ANN valdiklį



27 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant ANN valdiklį

Bandymai pakartojami keičiant pradines neuroninio valdiklio parametrų vertes, siekiant gauti geresnius rezultatus, taip pat palyginti kokia įtaka daroma procesui keičiant parametrus.

Pakeisti pradiniai duomenys:

Eta = 0.05;

Kc = 0.1;

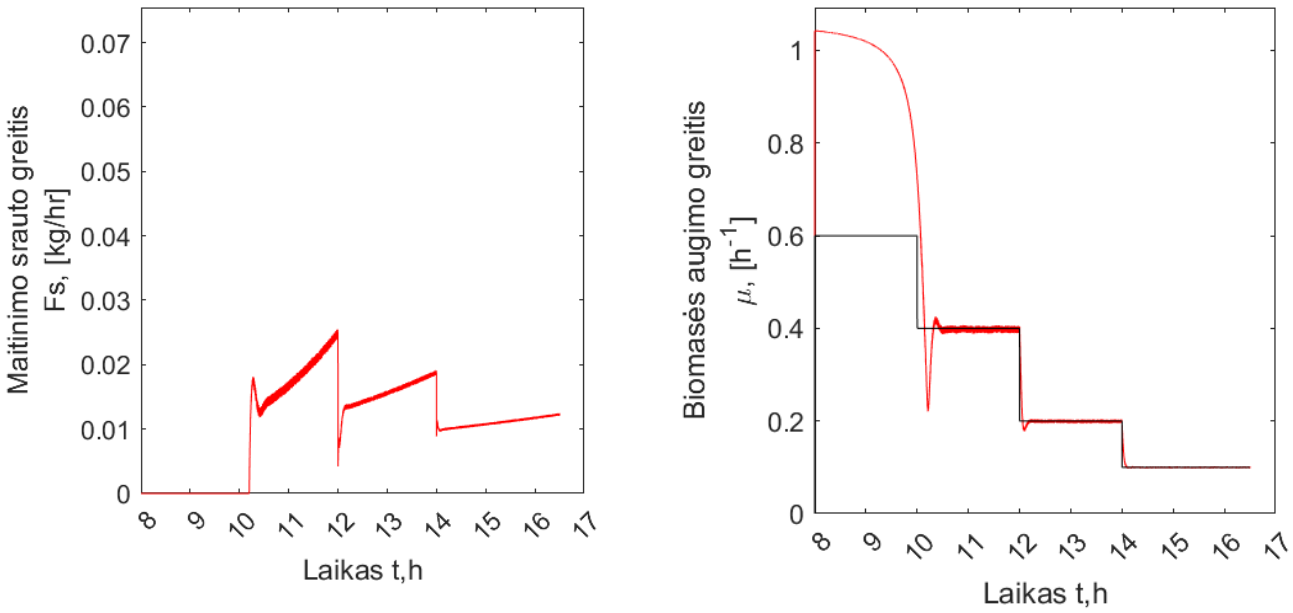
Neurons = 20.

Atliktas bandymas su neuroniniu valdikliu pakeitus pradinius parametrus ir įtraukus triukšmą. Tokiu atveju gaunamos šios paklaidos:

MAE: 0,0066;

MAPE: 2,3922;

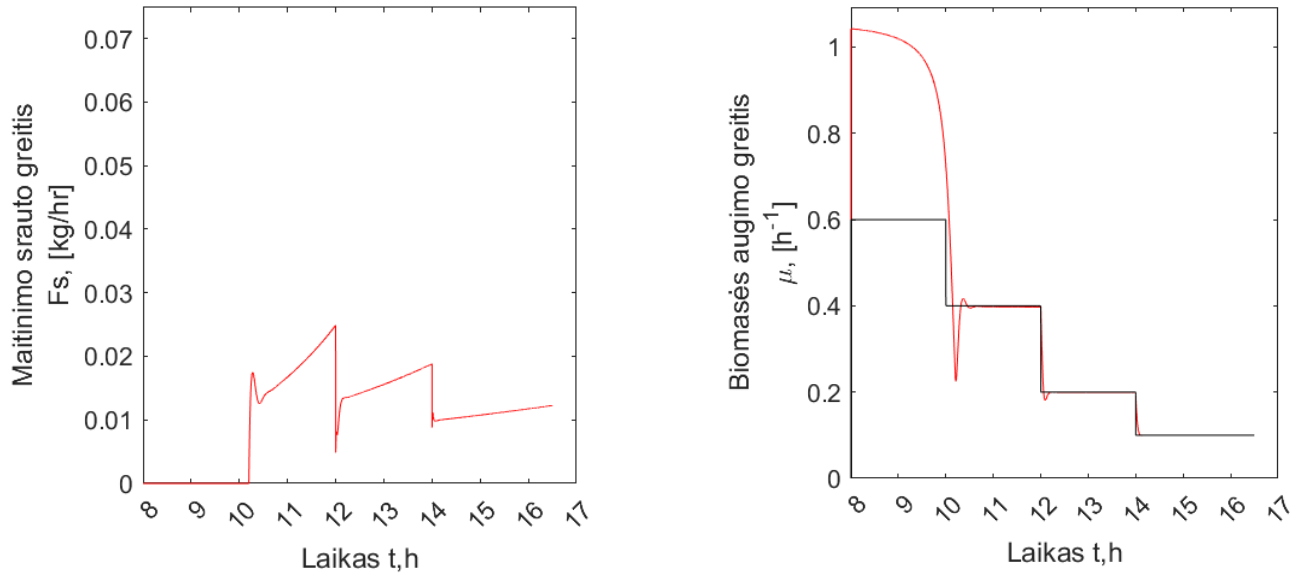
RMSE: 0,0250.



28 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant ANN valdiklį su triukšmu pakeitus pradinius parametrus

Atliktas bandymas su neuroniniu valdikliu pakeičiant pradinius parametrus ir neįtraukiant triukšmo. Tokiu atveju gaunamos šios paklaidos:

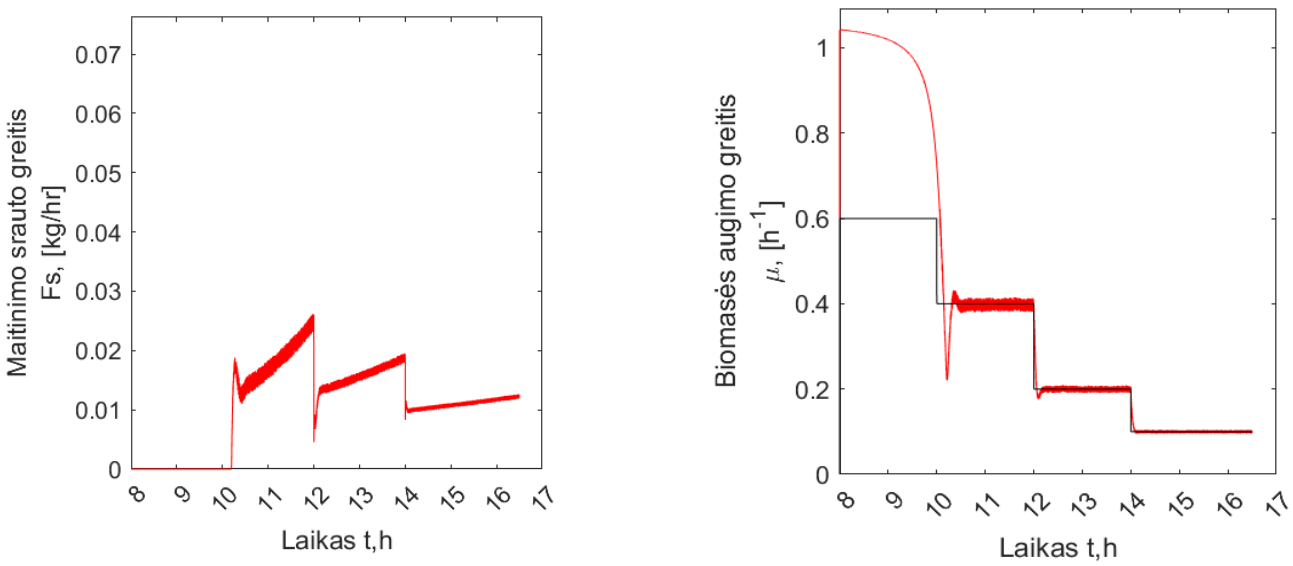
MAE: 0,0067;
MAPE: 2,4006;
RMSE: 0,0250.



29 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant ANN valdiklį be triukšmo pakeitus pradinius parametrus

Atliekamas dar vienas bandymas siekiant patikrinti valdiklio proceso valdymo kokybę sukeliant didesnį triukšmą. Rezultatai yra pateikti (30 pav.). Pradiniai duomenys yra paimti iš pirmojo bandymo, o triukšmas padidinamas dvigubai - 4%. Gaunamos paklaidos yra tokios:

MAE: 0,0053;
MAPE: 1,8357;
RMSE: 0,0232.



30 pav. Pamaitinimo greičio srauto ir specifinio augimo greičio kartu su nustatyta nuostata grafikai naudojant ANN valdiklį su padidintu triukšmu

Galima pastebėti, kad esant triukšmui, maitinimo srauto greitis kiek skiriasi (24, 27 pav.) ir atsiranda minimalūs svyravimai. Vis dėlto, specifinis augimo greitis yra arti nustatytos nuostatos. Triukšmo įtaka yra gana menka: MAE paklaidos skirtumas sudaro 0,0003, MAPE – 0,0226, o RMSE – lieka nepakitęs. Galime daryti prielaidą, kad net veikiant triukšmui, ANN valdiklis beveik tiksliai nustato norimą specifinį augimo greitį, tačiau vis tiek yra tam tikrų paklaidų. Norint pasiekti dar geresnių rezultatų, reikėtų atlikti geresnį ANN suderinimą. Visos bandymų metu gautos paklaidos yra pateiktos (1) lentelėje.

Rezultatai ir išvados

1. Apžvelgta literatūra apie biotechnologinius procesus, *E. Coli* bateriją, bioreaktorių tipus, jų valdymo metodus. Apžvelgtos PID reguliatoriaus ir neuroninio valdiklio valdymo ypatybės.
2. Pasirinktas ir sukurtas matematinis *E. coli* periodinio su pamaitinimu kultivavimo proceso modelis, naudojant „Matlab“ aplinką.
3. Neuroninio valdiklio algoritmas buvo sukurtas „Matlab“ aplinkoje. Valdiklis, kaip ir reguliatorius, koreguoja santykinio augimo greitį periodinėje su pamaitinimu mikroorganizmų kultūroje.
4. PI reguliatoriaus parametrų tikslumui įvertinti, procesas buvo vykdomas naudojant du skirtingus PI reguliatoriaus parametrų rinkinius. Pirmojo bandymo metu pradinės vertės buvo P=1 ir I=0,5. Bandymo metu gauti grafikai (10-15) paveikslėliuose, gautos paklaidos, kurios įrašytos (1) lentelėje. Triukšmas pirmajame bandyme turėjo sąlyginai mažą įtaką: esant triukšmui ir be jo, MAE paklaidų skirtumas buvo 0,0001, MAPE – 0,1162, RMSE – 0,001. Antrasis bandymas buvo atliktas su kitomis pradinėmis vertėmis: P=2, I=1. Keičiant pradinius duomenis, buvo gauti (16-17) paveikslėliuose pavaizduoti grafikai ir paklaidos, kurios taip pat pateiktos (1) lentelėje. Kaip ir pirmojo bandymo metu, triukšmas antrajame bandyme turėjo sąlyginai mažą įtaką: esant triukšmui ir be jo, MAE paklaidų skirtumas buvo 0,0002, MAPE – 0,129, RMSE – 0,003. Palyginus pirmojo ir antrojo bandymo triukšmo įtakos paklaidų duomenis, galima pastebėti, kad pirmojo bandymo paklaidos yra mažesnės. Paklaidų skirtumai yra tokie: MAE – 0,0001, MAPE – 0,1033, RMSE – 0,002. Nors paklaidos nėra didelės, jų skirtumas yra aiškiai matomas. Iš šių duomenų galima daryti prielaidą, kad pirmojo bandymo pradiniai parametrai buvo efektyvesni. Be to, antrojo bandymo metu pastebėti ypač dideli svyravimai, kai nuostata buvo pakeista į 0,1 laiko momentu t=14 (17 pav.). Įvertinus visus duomenis, galima daryti išvadą, kad pirmojo bandymo parametrų rinkinys buvo efektyvesnis, nes jis sukėlė mažesnes paklaidas. Tačiau reikia pažymėti, kad nors ir antrojo bandymo paklaidos buvo didesnės, jos vis tiek buvo gana mažos ir gali būti laikomos priimtinomis daugumoje situacijų.

4.1. Dviejų bandymų metu, kurie atlikti siekiant išsiaiškinti švytavimų atsiradimo priežastis, buvo pastebėta, kad nuostatos pakeitimas į 0,1 (18-19 pav.) gali sukelti didesnius švytavimus, ypač remiantis antrojo bandymo pradiniais PI reguliatoriaus parametrais (17 pav.). Tai rodo, kad procesui efektyviai valdyti, reikia tinkamai parinkti reguliatoriaus parametrus.

4.2. Atliktas bandymas padidinus triukšmo vertę dvigubai, naudojant pirmojo bandymo pradinius duomenis. Po bandymo gauti grafikai yra pateikti (20 pav.). Gautos paklaidos kurios pateiktos (1) lentelėje. Paklaidų skirtumas tarp skirtingų triukšmo lygių: MAE: 0,005, MAPE: 1,2915; RMSE: 0,0032. Pastebėta, kad padidinus triukšmą dvigubai, paklaidų skirtumas tapo žymiai didesnis, tai rodo, kad triukšmo įtaka reguliatoriaus veikimui stiprėja, ir dėl to reguliatorius sunkiau valdo procesą.

4.3. Neuroninio valdiklio parametrų tikslumui įvertinti buvo atlikti du bandymai su skirtingais parametrais. Pirmojo bandymo metu pradiniai parametrai buvo šie: Eta – 0,2, Kc – 0,12, neuronų skaičius – 5. Po bandymo gauti grafikai yra pateikti (21-26 pav.). Gautos paklaidos yra pateiktos (1) lentelėje. Pastebėta, jog esant triukšmui, paklaidos buvo mažesnės, tai galima matyti iš paklaidų skirtumų esant triukšmui ir be jo: MAE – 0,0003, MAPE – 0,0226, RMSE – 0. Iš šių duomenų galima daryti išvadą, kad pagal pirmojo bandymo pradinius parametrus neuroninis valdiklis efektyviau veikia esant triukšmui, kas yra naudinga, nes realiose situacijose ar gamyboje dažnai atsiranda pašalinių efektų ar triukšmų. Antrasis bandymas buvo atliktas su kitais pradiniais parametrais: Eta – 0,05, Kc – 0,1, neuronų skaičius – 20. Po bandymo gauti grafikai yra pateikti

(27-28 pav.). Gautos paklaidos yra pateiktos (1) lentelėje. Kaip ir pirmojo bandymo metu, paklaidos esant triukšmui buvo mažesnės, paklaidų skirtumai esant triukšmui ir be jo yra: MAE – 0,0001, MAPE – 0,0084, RMSE – 0. Palyginus pirmojo ir antrojo bandymo paklaidų skirtumus, pastebima, kad paklaidos buvo mažesnės pirmojo bandymo metu, tai reiškia, kad pirmojo bandymo pradinių parametrų parinkimas buvo efektyvesnis.

4.4. Atliktas bandymas padidinus neuroninio valdiklio triukšmo vertę dvigubai. Šis triukšmas buvo generuotas naudojant pirmojo bandymo pradinius duomenis, kadangi šie duomenys parodė geriausius rezultatus. Po bandymo gauti grafikai yra pateikti (30 pav.). Gautos paklaidos yra pateiktos (1) lentelėje. Paklaidų skirtumas esant skirtingiems triukšmo lygmenims yra: MAE - 0,001, MAPE - 0,0349, RMSE - 0,0001. Padidinus triukšmą dvigubai, pastebėta, kad šis triukšmas valdikliui daro gana mažą įtaką, paklaidos yra beveik identiškos.

5. PI reguliatoriaus ir neuroninio valdiklio gautos paklaidos, esant didesniam triukšmui, buvo palygintos. Pastebėta, kad neuroninis valdiklis veikia žymiai geriau esant didesniam triukšmui - triukšmo įtaka beveik nesiskiria nuo bandymo, kai triukšmas yra lygus 2%. PI reguliatoriaus rezultatai iš (1) lentelės, palyginti su neuroniniu valdikliu, yra daug prastesni, o tai aiškiai rodo neuroninio valdiklio pranašumą.

5.1. PI reguliatoriaus ir neuroninio valdiklio geriausių bandymų paklaidos buvo palygintos, o jų skirtumas yra: MAE – 0,0028, MAPE – 0,8023, RMSE – 0,0049. Atsižvelgiant į šį nemažą paklaidų skirtumą, galima daryti išvadą, kad neuroninis valdiklis kur kas efektyviau nustato biomasės augimo greitį, valdant maitinimo srauto greitį. Net palyginus maitinimo srautų grafikus be triukšmo (15 ir 26 pav.) naudojant PI reguliatorių ir neuroninį valdiklį, matomas akivaizdus skirtumas - neuroninio valdiklio grafike matyti mažesni švytavimai ir aiškesnė maitinimo srauto greičio kreivė laiko atžvilgiu.

6. Atlikus bandymus, neuroninio valdiklio pranašumas tapo akivaizdus. Iš rezultatų matyti, kad neuroninis valdiklis yra atsparesnis triukšmui nei PI reguliatorius. Nepaisant triukšmo, neuroninis valdiklis geba tiksliau suvokti valdymo charakteristikas ir atitinkamai reguliuoti procesą. Kitas svarbus pranašumas - mokymosi galimybė. Aprašytas neuroninis valdiklis mokosi iš savo praeities klaidų, o PI reguliatoriaus struktūra yra paprastesnė, jis reikalauja išankstinio parametrų nustatymo ir nepasirodo toks lankstus kaip neuroninis valdiklis. Neuroninis valdiklis yra galingas įrankis, galintis valdyti sudėtingas sistemas ir netiesinius įvesties ir išvesties santykius, o PI reguliatorius yra geriau pritaikytas paprastesnių sistemų parametrų valdymui.

6.1. Siekiant gauti dar aukštesnio tikslumo rezultatus ir mažesnes paklaidas, vertėtų apsvarstyti neuroninio valdiklio struktūros tobulinimą ir adaptaciją sudėtingesnėms situacijoms. Taip pat būtina atlikti daugiau bandymų, kad būtų galima nustatyti, su kokiais parametrais sistema veiktų efektyviausiai. Be to, vertėtų sukurti papildomus triukšmo variantus, kad būtų galima išsamiau patikrinti valdiklio darbo kokybę, nes sukurtas biomasės augimo greičio triukšmas turėjo tik nedidelę įtaką proceso valdymui.

Literatūros sąrašas

1. Anita Sangwan, Escherichia Coli (E. Coli): Meaning, Morphology and Characteristics. [žiūrėta: 2023 m. Gegužės 06d.] Prieiga internetu: http://www.biologydiscussion.com/bacteriology/systematic-bacteriology/escherichia-coli-e-colimeaning-morphology-and-characteristics/30821Advances in recombinant protein expression for use in pharmaceutical research. Current Opinion in Structural Biology, 23(3), 393-402.
2. Nataro, J. P. (2005). Escherichia coli: Pathotypes and Principles of Pathogenesis. Clinical Microbiology Reviews, 18(2), 214-226. doi: 10.1128/CMR.18.2.214-226.2005.
3. V. Lubenova, I. Rocha, E.C. Ferreira, Estimation of multiple biomass growth rates and biomass concentration in a class of bioprocesses, [žiūrėta 2023 m. gegužės 5d.] Prieiga internetu :https://link.springer.com/article/10.1007/s00449-003-0325-1.
4. Talaat E. Shehata, Allen G. Marr. Effect of Nutrient Concentration on the Growth of Escherichia coli. [žiūrėta 2023 m. gegužės 6d.]. Prieiga internetu : https://journals.asm.org/doi/abs/10.1128/jb.107.1.210-216.1971.
5. Cong T. Trinh, Ross Carlson, Aaron Wlaschin, Friedrich Srienc, Design, construction and performance of the most efficient biomass producing E. coli bacterium. [žiūrėta 2023 m. gegužės 6d.]. Prieiga internetu : https://www.sciencedirect.com/science/article/abs/pii/S109671760600067X?casa_token=p6DfXXXxOzMAAAAA:AofG3EmxWCii6WQEzh_22QCkKDydc0IgdXR05Dd1nwjKsfR1oKbDQQOTdgJsogy6wp2yjl25wyc.
6. Galvanauskas, V., & Levišauskas, D. (2008). Biotechnologinių procesų modeliavimas, optimizavimas ir valdymas. [žiūrėta 2023 m. gegužės 6d.]. Prieiga internetu : https://doi.org/10.5755/e01.9786090203903.
7. Schirmer, C., Maschke, R.W., Pörtner, R. et al. An overview of drive systems and sealing types in stirred bioreactors used in biotechnological processes. [žiūrėta 2023 m. gegužės 7 d.] Prieiga internetu : https://link.springer.com/article/10.1007/s00253-021-11180-7.
8. Alfred Elikem Kwami Afedzi, Kittipong Rattanaporn, Pramuk Parakulsuksatid, Impeller selection for mixing high-solids lignocellulosic biomass in stirred tank bioreactor for ethanol production. [žiūrėta 2023 m. gegužės 6d.]. Prieiga internetu : https://www.sciencedirect.com/science/article/abs/pii/S2589014X21003133?casa_token=aPqG_C6r-TAAAAAA:GycMUv15JiJwoOyF1o96y7XQWHh188q34sXn-uaurXGNFk9jS_YdenHqPcwoKMTGwo-EaktgTzo
9. George Skouteris, Daphne Hermosilla, Patricio López, Carlos Negro, Ángeles Blanco. Anaerobic membrane bioreactors for wastewater treatment: A review. [žiūrėta 2023 m. gegužės 7d.]. Prieiga internetu : https://www.sciencedirect.com/science/article/abs/pii/S1385894712006420?casa_token=WN9kN0mosfIAAAAA:_VPJbyTykNwk9X9WYSmcFVF2tfp_OpTCarfzs4Zkx8LUZBVNvvGPVoNG6JRBlV_B-mbHj1nH5w.
10. Verena Kastner, Walter Somitsch, Wolfgang Schnitzhofer, The anaerobic fermentation of food waste: a comparison of two bioreactor systems. [žiūrėta 2023 m. gegužės 7d.]. Prieiga internetu : https://www.sciencedirect.com/science/article/abs/pii/S0959652612001515?casa_token=C6LVAtsqWkgAAAAA:4YZpGPBJJUf7sa62y8kT5BVVHTDi7O21Un-7RItJtYmv7wdUgfFCkouruv-C840zfgaqEPhknJg

11. Galvanauskas, V., & Levišauskas, D. (2008). Biotechnologinių procesų modeliavimas, optimizavimas ir valdymas. [žiūrėta 2023 m. gegužės 7 d.]. Prieiga internetu : https://doi.org/10.5755/e01.9786090203903.

12. A fundamental analysis of continuous flow bioreactor models and membrane reactor models to process industrial wastewaters. [žiūrėta 2023 m. gegužės 7 d.]. Prieiga internetu : https://www.sciencedirect.com/science/article/abs/pii/S1385894707007735.

13. DOCHAIN Denis. *Automatic Control of Bioprocesses* [Automatique des bioprocédés]. 2 – oji laida. Antony Rowe Ltd, Chippenham, Wiltshire, 2008. ISBN: 978-1-84821-025-7.

14. Rimvydas Simutis, Andreas Lübbert. Bioreactor control improves bioprocess performance. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://onlinelibrary.wiley.com/doi/full/10.1002/biot.201500016?casa_token=Dzo3tIXsI_8AAAAA%3A1v2wrUk4M1xzqHlrQPhJOO5iCYBkzkpP4TDaPDwS5nBmSXHOmjcrXiVdksyWXe5l3zoogLptxaHlZha2.

15. P. R. Davidson, R. D. Jones, J. H. Andreae and H. R. Sirisena. Simulating closed- and open-loop voluntary movement: a nonlinear control-systems approach, [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://ieeexplore.ieee.org/abstract/document/1046932?casa_token=hPjsiSSqvgAAAAAA:oz83jDwqSCwreuYpCBMeMa84LHDG14iV1D7CDfLcsGCJ23ai_DvBFE5KPkWVutFF4KLrpehnL0k.

16. Galvanauskas, V.; Simutis, R.; Levišauskas, D.; Urniežius, R. Practical Solutions for Specific Growth Rate Control Systems in Industrial Bioreactors. Processes 2019, 7, 693. https://doi.org/10.3390/pr7100693.

17. IB Levitan, LK Kaczmarek. The neuron: cell and molecular biology. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://books.google.lt/books?hl=lt&lr=&id=yTXS_OAkUmcC&oi=fnd&pg=PA3&dq=neuron+cell+structure&ots=9f5x81V8VU&sig=BfdJChR_yNk5zDyZhz8KFzrZgsg&redir_esc=y#v=onepage&q=neuron%20cell%20structure&f=false.

18. Chris M. Bishop. Neural networks and their applications. . [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://pubs.aip.org/aip/rsi/article/65/6/1803/682910/Neural-networks-and-their-applicationsNeural.

19. Ian Goodfellow, Yoshua Bengio, Aaron Courville. Deep Learning. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://www.google.lt/books/edition/Deep_Learning/Np9SDQAAQBAJ?hl=lt&gbpv=1&dq=Ian+Goodfellow,+Yoshua+Bengio,+Aaron+Courville&printsec=frontcover.

20. Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., & Khudanpur, S. Recurrent neural network based language model. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://www.fit.vutbr.cz/research/groups/speech/publi/2010/mikolov_interspeech2010_IS100722.pdf

21. Krizhevsky, A., Sutskever, I., & Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://papers.nips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

22. Hinton, G. E., Osindero, S., & Teh, Y. W. A fast learning algorithm for deep belief nets. Neural computation. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://www.cs.toronto.edu/~hinton/absps/fastnc.pdf

23. Sepp Hochreiter, Jürgen Schmidhuber . Long Short-Term Memory. Neural Comput. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://doi.org/10.1162/neco.1997.9.8.1735.

24. TAKIALDDIN, AL SMADI, AL-AGHA, OSMAN IBRAHIM, ALSMADI, KHALID ADNAN. Overview of model free adaptive (MFA) control technology. IAES International Journal of Artificial Intelligence [interaktyvus]. 2018. 165-169, 7(4) [žiūrėta 2023 m. gegužės 3 d.]. ISSN 22528938. Prieiga per: doi: 10.11591/ijai.v7.i4.pp165-169 Techniques for Adaptive Control – knyga 145-155 psl.

25. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://doi.org/10.1038/nature14539.

26. Ian Goodfellow, Yoshua Bengio, Aaron Courville. Deep Learning. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://www.google.lt/books/edition/Deep_Learning/Np9SDQAAQBAJ?hl=lt&gbpv=1&dq=Ian+Goodfellow,+Yoshua+Bengio,+Aaron+Courville&printsec=frontcover.

27. Diederik P. Kingma, Jimmy Ba. Adam: A Method for Stochastic Optimization . [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://arxiv.org/pdf/1412.6980.pdf.

28. Tieleman, T., & Hinton, G. (2012). Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 4(2), 26-31.

29. Andrew Y. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://icml.cc/Conferences/2004/proceedings/papers/354.pdf

30. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov . Dropout: A Simple Way to Prevent Neural Networks from Overfitting. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf.

31. Sergey Ioffe, Christian Szegedy.Batch normalization: Accelerating deep network training by reducing internal covariate shift. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://arxiv.org/abs/1502.03167.

32. Peter F. Stanbury, Allan Whitaker, Stephen J. Hall. Principles of fermentation technology. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://www.google.lt/books/edition/Principles_of_Fermentation_Technology/yOKoBAAAQBAJ?hl=lt&gbpv=1&dq=Principles+of+fermentation+technology&printsec=frontcover.

33. C Altamirano, C Paredes, A Illanes, J.J Cairó, F Gòdia, Strategies for fed-batch cultivation of t-PA producing CHO cells: substitution of glucose and glutamine and rational design of culture. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://www.sciencedirect.com/science/article/pii/S0168165604000926.

34. V. Galvanauskas, R. Simutis & A. Lübbert. Hybrid process models for process optimisation, monitoring and control. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://link.springer.com/article/10.1007/s00449-004-0385-x.

35. V. Galvanauskas, R. Simutis, N. Volk & A. Lübbert. Model based design of a biochemical cultivation process. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://link.springer.com/article/10.1007/s004490050435.

36. Urniezius, R., Survyla, A., Paulauskas, D. et al. Generic estimator of biomass concentration for Escherichia coli and Saccharomyces cerevisiae fed-batch cultures based on cumulative oxygen consumption rate. Microb Cell Fact 18, 190 (2019). https://doi.org/10.1186/s12934-019-1241-7

37. Galvanauskas, V.; Simutis, R.; Vaitkus, V. Adaptive Control of Biomass Specific Growth Rate in Fed-Batch Biotechnological Processes. A Comparative Study. [žiūrėta 2023m. gegužės 7 d.]. Prieiga internetu: https://www.mdpi.com/2227-9717/7/11/810.

Priedai

1 priedas. Balanso proceso .m failo kodas

```
function dc_dt = BalanceProcess(x,s,a,w,Fs)
global qSmax Ks Ki Ysx m miu_crit alpha_ap alpha_ac miu_max o1 o2 c1 c2  b_1
global Yax sf e1 Fsmp Sf Yas F
global qS rac rap OUR CER
%
%=====================================================================
% Model/process parameters
% -------------------------
qSmax       = 2.125;         % Maximal glucose specific uptake rate, [g/g/h]
Ks          = 0.119;         % Monod constant for glucose, [g/l]
Ki          = 17.64;         %
Ysx         = 1.9075;        %
m           = 0.0089;        % Maintenance term, [g/g/h]
miu_crit    = 0.52;          % Critical specific growth rate [1/hr]
alpha_ap    = 0.53;
alpha_ac    = 0.885;
miu_max     = 0.394;
o1          = 0.746;
o2          = 0.0306;
c1          = 1;
c2          = 0.055;
Yax         = 3.868;         % Conversion yield acetate consumption/biomass, [g/g]
b_1         = 0.0025;
Yas         = 0.994;         % Conversion yield acetate/glucose, [g/g
sf          = 300;
e1          = 0.015;         % Evaporation rate
Fsmp        = 0.05;
%=====================================================================
if(x < 0),x = 0; end
if(s < 0),s = 0; end
if(a < 0),a = 0; end
if(w < 0),w = 0; end
if(Fs < 0),Fs = 0; end

% Specific glucose consumption rate
=====================================================================
qS=qSmax*s/(s+Ks)*Ki/(a+Ki);

% Specific growth rate
miu = qS/Ysx-m;

% Acetate production / consumption rap, rac
=====================================================================
rap = 0;
rac = 0;

if(miu >= miu_crit)              %acetatai kaupiasi kai miu daugiau arba lygu už
kritinę miu
    rap = alpha_ap*(miu-miu_crit);
end
if(miu < miu_crit && a > 0)      %acetatų suvartojimas kai mažiau už kritinę miu ir
taip pat kad būtų acetatų daugiau už 0, kad būtų iš ko vartoti
    rac = miu_max - alpha_ac*miu;
end

% Oxygen consumption and carbon dioxide emmision OUR, CER
=====================================================================
OUR=o1*(miu + (rac/Yax))*x+o2*x;
```

```
CER=c1*(miu + (rac/Yax))*x+o2*x;
if(OUR < 0), OUR = 0; end
if(CER < 0), CER = 0; end

% Solutions
Fb = b_1*x*(miu+rac/Yax)*w/1000; % Base additon rate (for pH control)
F_cl = (CER-OUR)*w*0.001;                % Mass lost due CO2 evolution
F=Fs+Fb-F_cl-e1;                 % Total mass flow

% Mass balance equations
========================================================================
%x= x + (miu+(rac/Yax)-(F/w))*x * c;
%if(x < 0), x = 0; end

%s= s + (-(qS+(rap/Yas))*x-(F/w)*s+(Fs/w)*sf) * c;
%if(s(i) < 0), s(i) = 0; end

%a= a + ((rap-rac)*x-(F/w)*a) * c;
%if(a(i) < 0), a(i) = 0; end

%w(i)= w + (F-Fsmp) * c;
%if(w(i) < 0), w(i) = 0; end

% Mass balance equations
dc_dt(1,1)=(miu+(rac/Yax)-(F/w))*x;
dc_dt(2,1)=(-(qS+(rap/Yas))*x-(F/w)*s+(Fs/w)*sf);
dc_dt(3,1)=((rap-rac)*x-(F/w)*a);
dc_dt(4,1)=(F-Fsmp);
dc_dt(5,1)=miu;
dc_dt(6,1)=OUR;
dc_dt(7,1)=CER;

%q=[q; t miu qS qA qL qP FS FL OUR CPR Fb];
end


% ========================================================================
```

2 priedas. ANN reguliatoriaus .m failo kodas

```
global x s w a CER OUR miu time FS miu_setpoint

%=======================================================================
% Initial conditions
c = 0.001;
tstart = 8;
t = 16.5;

FSmax = 5.0;           % Maximal glucose feeding rate, [kg/h]
F0S = 0.1;         % Glucose initial feed rate,[kg/h]
miu0 = 0.375;           % Glucose feed rate expon.,[kg/h]

miu1 = 0.6;
t2 = 10;
miu2 = 0.4;
t3 = 12;
miu3 = 0.2
t4 = 14;
miu4 = 0.1;
t5 = 17;
```

```
iterations = (t-tstart)/c;
x = zeros(iterations + 1,1); x(1) = 0.13;
s = zeros(iterations + 1,1); s(1) = 2;
a = zeros(iterations + 1,1); a(1) = 0;
w = zeros(iterations + 1,1); w(1) = 4.65;
OUR = zeros(iterations + 1,1); OUR(1) = 0;
CER = zeros(iterations + 1,1); CER(1) = 0;
miu = zeros(iterations + 1,1); miu(1) = 0.375;
FS  = zeros(iterations + 1,1); FS(1) = 0;
miu_setpoint = zeros(iterations + 1,1); miu_setpoint(1) = 0;
time = [tstart:c:t]';

MAE_ANN = 0;
MAPE_ANN = 0;
RMSE_ANN = 0;
count_ANN = 0;

% ANN parameter ============================
Eta = 0.2;
Kc = 0.12;
Neurons = 5;
e = zeros(Neurons + 1,1);
e(1) = 1;%1.4135;
weight = zeros(Neurons + 1,Neurons);
h = zeros(Neurons,1);
index = 2;

%=========================================================================
% Process
for i=2:length(time)

    if (c<=0 || t<=0)
        break
    end

%Setpoint miu ============================================
miu_setpoint(i) = miu1;
if time(i) > t2 && time(i) <= t3, miu_setpoint(i) = miu2; end
if time(i) > t3 && time(i) <= t4, miu_setpoint(i) = miu3; end
if time(i) > t4 && time(i) <= t5, miu_setpoint(i) = miu4; end
%=========================================================
% ANN regulator ===========================================
if(s(i - 1) < 0.15)
error = miu_setpoint(i) - miu(i - 1);

MAE_ANN = MAE_ANN + abs(error);
MAPE_ANN = MAPE_ANN + abs(error/miu(i - 1));
RMSE_ANN = RMSE_ANN + error^2;
count_ANN = count_ANN + 1;

miu_2proc_down = miu(i - 1) - miu(i - 1)* 0.02;
miu_2proc_up = miu(i - 1) + miu(i - 1)* 0.02;
noise = miu_2proc_down + (miu_2proc_up-miu_2proc_down)*rand();
miu(i - 1) = noise;
error = miu_setpoint(i) - noise;

e(index) = error;
index = index + 1;
if(index > Neurons + 1),index = 2; end

p = zeros(Neurons,1);
```

```
q = zeros(Neurons,1);

for j=1:Neurons
    for l=1:Neurons + 1
        p(j) = p(j) + weight(l,j)*(e(l)^2*0.5);
    end
end

for j=1:Neurons
    q(j) = 1 / (1 + exp(-p(j)));
end

sum_h = 0;
o = 0;
for j=1:Neurons
    sum_h = sum_h + h(j);
    o = o + h(j).*q(j);
end

for j=1:Neurons
    for z=1:Neurons+1
        weight(z,j) = weight(z,j) + Eta*Kc*e(z)*q(j)*(1-q(j))*(e(z))^2*0.5*sum_h;
    end
    h(j) = h(j) + Eta*Kc*e(j+1)*q(j);
end

FS(i) = Kc*(error + o);
if FS(i)<0, FS(i)=0; end % Min feed rate limitation
if FS(i)>=FSmax, FS(i)=FSmax; end       % Max feed rate limitation
end
%============================================================
%============================================================
dc_dt = BalanceProcess(x(i - 1),s(i - 1),a(i - 1),w(i - 1),FS(i));
x(i) = x(i - 1) + dc_dt(1,1) * c;
s(i) = s(i - 1) + dc_dt(2,1) * c;
a(i) = a(i - 1) + dc_dt(3,1) * c;
w(i) = w(i - 1) + dc_dt(4,1) * c;
miu(i) = dc_dt(5,1);
OUR(i) = dc_dt(6,1);
CER(i) = dc_dt(7,1);
end

MAE_ANN = MAE_ANN/count_ANN;
MAPE_ANN = MAPE_ANN*100/count_ANN;
RMSE_ANN = sqrt(RMSE_ANN/count_ANN);

eps=0.5;
xw = x.*w;
%=========================================================================
% Plotting results
cexp = zeros(1,6);
figure (1)
subplot (2,4,2)
%title ('Fed-batch E.coli B pUBS520 p12023 process for production of recombinant MAK33
light chain protein'), hold on

subplot (2,4,1)
plot (time,xw,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ({'Biomasės koncentracija', 'X, [g ]'})
axis ([time(1) time(end)+eps 0 xw(end)+1])
set(gca, 'xtick', time(1):1:time(end)+eps);
set(gca, 'ytick', 0:5:xw(end)+1);
```

```
subplot (2,4,2)
plot (time,s,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ({'Gliukozės koncentracija', 'S, [g kg^-^1]'})
axis ([time(1) time(end)+eps 0 max(s)+eps])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,3)
plot (time,a,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ({'acetatų koncentracija','A, [g kg^-^1]'})
axis ([time(1) time(end)+eps 0 max(a)+eps])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,4)
plot (time,w,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ({'bioreaktoriaus svoris', 'W, [kg]'})
axis ([time(1) time(end)+eps 0 max(w)+eps])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,5)
plot (time,miu_setpoint,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ({'Biomasės augimo greičio nuostata','\mu , [h^-^1]'})
axis ([time(1) time(end)+eps 0 max(miu_setpoint)+0.05])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,6)
plot (time,miu,'k',0,cexp(:,1),['w' '.']), hold on
plot (time,miu_setpoint,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ({'Biomasės augimo greitis','\mu, [h^-^1]'})
axis ([time(1) time(end)+eps 0 max(miu)+0.05])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,7)
plot (time,FS,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ({'Maitinimo srauto greitis','Fs, [kg/hr]'})
axis ([time(1) time(end)+eps 0 max(FS)+0.05])
set(gca, 'xtick', time(1):1:time(end)+eps);
set(gca, 'ytick', FS(1):0.1:FS(end)+0.05);
```

3 priedas. PI reguliatoriaus .m failo kodas

```
global x s w a CER OUR miu time FS miu_setpoint

%=========================================================================
% Initial conditions
c = 0.001;
tstart = 8;
t = 16.5;

FSmax = 5.0;           % Maximal glucose feeding rate, [kg/h]
F0S = 0.1;         % Glucose initial feed rate,[kg/h]
miu0 = 0.375;           % Glucose feed rate expon.,[kg/h]

miu1 = 0.6;
t2 = 10;
miu2 = 0.4;
t3 = 12;
miu3 = 0.2
t4 = 14;
```

```
miu4 = 0.1;
t5 = 17;

MAE_PID = 0;
MAPE_PID = 0;
RMSE_PID = 0;
count_PID = 0;

iterations = (t-tstart)/c;
x = zeros(iterations + 1,1); x(1) = 0.13;
s = zeros(iterations + 1,1); s(1) = 2;
a = zeros(iterations + 1,1); a(1) = 0;
w = zeros(iterations + 1,1); w(1) = 4.65;
OUR = zeros(iterations + 1,1); OUR(1) = 0;
CER = zeros(iterations + 1,1); CER(1) = 0;
miu = zeros(iterations + 1,1); miu(1) = 0.375;
FS  = zeros(iterations + 1,1); FS(1) = 0;
miu_setpoint = zeros(iterations + 1,1); miu_setpoint(1) = 0;
time = [tstart:c:t]';

% PID parameter ============================
P = 1;
I = 0.5;
error_sum = 0;

%=========================================================================
% Process
for i=2:length(time)
    if (c<=0 || t<=0)
        break
    end

%Setpoint miu ============================================
miu_setpoint(i) = miu1;
if time(i) > t2 && time(i) <= t3, miu_setpoint(i) = miu2; end
if time(i) > t3 && time(i) <= t4, miu_setpoint(i) = miu3; end
if time(i) > t4 && time(i) <= t5, miu_setpoint(i) = miu4; end
%==========================================================

% PID regulator ==========================================
if(s(i - 1) < 0.15)
error = miu_setpoint(i) - miu(i - 1);

MAE_PID = MAE_PID + abs(error);
MAPE_PID = MAPE_PID + abs(error/miu(i - 1));
RMSE_PID = RMSE_PID + error^2;
count_PID = count_PID + 1;

miu_2proc_down = miu(i - 1) - miu(i - 1)* 0.02;
miu_2proc_up = miu(i - 1) + miu(i - 1)* 0.02;
noise = miu_2proc_down + (miu_2proc_up-miu_2proc_down)*rand();
miu(i - 1) = noise;
error = miu_setpoint(i) - noise;

error_sum = error_sum + (error * c);
FS(i) =  error * P + error_sum * I;
if FS(i)<0, FS(i)=0; end % Min feed rate limitation
if FS(i)>=FSmax, FS(i)=FSmax; end       % Max feed rate limitation
end

%==========================================================
dc_dt = BalanceProcess(x(i - 1),s(i - 1),a(i - 1),w(i - 1),FS(i));
```

```
x(i) = x(i - 1) + dc_dt(1,1) * c;
s(i) = s(i - 1) + dc_dt(2,1) * c;
a(i) = a(i - 1) + dc_dt(3,1) * c;
w(i) = w(i - 1) + dc_dt(4,1) * c;
miu(i) = dc_dt(5,1);
OUR(i) = dc_dt(6,1);
CER(i) = dc_dt(7,1);
end

MAE_PID = MAE_PID/count_PID;
MAPE_PID = MAPE_PID*100/count_PID;
RMSE_PID = sqrt(RMSE_PID/count_PID);

eps=0.5;
xw = x.*w;
%=========================================================================
% Plotting results
cexp = zeros(1,6);
figure (1)
subplot (2,4,2)
title ('Fed-batch E.coli B pUBS520 p12023 process for production of recombinant MAK33
light chain protein'), hold on

subplot (2,4,1)
plot (time,xw,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ('XW, [g ]')%kg^-^1]')
axis ([time(1) time(end)+eps 0 xw(end)+1])
set(gca, 'xtick', time(1):1:time(end)+eps);
set(gca, 'ytick', 0:5:xw(end)+1);

subplot (2,4,2)
plot (time,s,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ('S, [g kg^-^1]')
axis ([time(1) time(end)+eps 0 max(s)+eps])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,3)
plot (time,a,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ('A, [g kg^-^1]')
axis ([time(1) time(end)+eps 0 max(a)+eps])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,4)
plot (time,w,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ('W, [kg]')
axis ([time(1) time(end)+eps 0 max(w)+eps])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,5)
plot (time,miu_setpoint,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ('\mu setpoint, [h^-^1]')
axis ([time(1) time(end)+eps 0 max(miu_setpoint)+0.05])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,6)
plot (time,miu,'k',0,cexp(:,1),['w' '.']), hold on
plot (time,miu_setpoint,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ('\mu, [h^-^1]')
axis ([time(1) time(end)+eps 0 max(miu)+0.05])
set(gca, 'xtick', time(1):1:time(end)+eps);

subplot (2,4,7)
```

```
plot (time,FS,'k',0,cexp(:,1),['w' '.']), hold on
ylabel ('Fs, [kg/hr]')
axis ([time(1) time(end)+eps 0 max(FS)+0.05])
set(gca, 'xtick', time(1):1:time(end)+eps);
set(gca, 'ytick', FS(1):0.1:FS(end)+0.05);
```