KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS

Juozas Kovaliovas

BIOKURO PAKUROS MODELIO SUDARYMAS IR DARBO REŽIMŲ TYRIMAS

Baigiamasis magistro projektas

Vadovas
Doc. dr. Kęstutis Brazauskas

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS

AUTOMATIKOS KATEDRA

BIOKURO PAKUROS MODELIO SUDARYMAS IR DARBO REŽIMŲ TYRIMAS

Baigiamasis magistro projektas

Valdymo technologijos (kodas M3066L21)

Vadovas

Doc. dr. Kęstutis Brazauskas

Recenzentas

Doc. dr. .....

Projektą atliko

Juozas Kovaliovas

KAUNAS, 2015

TVIRTINU:
KTU Automatikos katedros vedėjas
doc. Gintaras Dervinis

20….. m. .... d.

2 METŲ MAGISTRO STUDIJŲ BAIGIAMOJO DARBO
UŽDUOTIS

Magistrantas ***Juozas Kovaliovas***

Magistro tiriamojo darbo tema ***Biokuro pakuros modelio sudarymas ir darbo režimų tyrimas***

Magistro baigiamojo darbo **užduotis**:

1. Darbo tikslas (aiškiai nurodyti, kokiam tolimesniam tikslui pasiekti atliekami tyrimai)
Sudaryti metodiką, kuri labiausiai tiktų biokuro pakuros darbo režimo monitoringui ir analizei.

2. Literatūros apžvalga ir analizė
Išnagrinėti vandens šildymo katilų tipus, kuro rūšis, nubraižyti jų principines schemas. Apžvelgti biokuro pakuroje naudojamas valdymo sistemas ir jų tarpusavio ryšius. Pagal turimus duomenis ištirti kaip dirbo biokuro pakura, apibrėžti pagal kuriuos duomenis ir kokiose ribose jiems kintant bus sprendžiama apie pakuros darbo režimą, kaip jis bus klasifikuojamas.

3. Individualių tyrimų turinys
Sudaryti biokuro pakuros technologinę schemą. Surinkti ir apdoroti archyvinius pakuros darbo duomenis. Sudaryti pakuros matematinį modelį, atitinkantį archyvinius duomenis. Sukurti algoritmą, kuris naudodamas matematinį modelį, identifikuotų matavimo signalų neatitikimus su juos atitinkančiais modelio išėjimais.

4. Reikalavimai ir apribojimai
Darbas turi atitikti magistro darbams keliamus reikalavimus.

5. Iki 2015 m. gegužės 30 d. pateikti magistratūros studijų kvalifikacinei komisijai baigiamąjį darbą, tenkinantį šią užduotį ir bendruosius keliamus reikalavimus.
Papildomai pateikiama

Užduoties davimo data: 2015 m. vasario 12 d.

Magistrantas ***Juozas Kovaliovas***

(*Vardas, pavardė, parašas*)

Baigiamojo darbo vadovas ***Kęstutis Brazauskas***

(*Vardas, pavardė, parašas*)

KAUNO TECHNOLOGIJOS UNIVERSITETAS

Elektros ir elektronikos fakultetas
(Fakultetas)

Juozas Kovaliovas
(Studento vardas, pavardė)

Valdymo technologijos (M3066L21)
(Studijų programos pavadinimas, kodas)

Baigiamojo projekto

„ BIOKURO PAKUROS MODELIO SUDARYMAS IR DARBO REŽIMŲ TYRIMAS “

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 ___ m. _____ ____ d.
Kaunas

Patvirtinu, kad mano **Juozo Kovaliovo** baigiamasis projektas tema „Biokuro pakuros modelio sudarymas ir darbo režimų tyrimas“ yra parašytas visiškai savarankiškai, o visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

_____ _____
(vardą ir pavardę įrašyti ranka) *(parašas)*

Kovaliovas, J. Biokuro pakuros modelio sudarymas ir darbo režimų tyrimas. Magistro baigiamasis projektas / vadovas doc. dr. Kęstutis Brazauskas; Kauno technologijos universitetas, elektros ir elektronikos fakultetas, automatikos katedra.

Kaunas, 2015. 72 psl.

SANTRAUKA

Biokuro deginimas – sparčiai šalyje populiarėjantis šilumos energijos gamybos šaltinis. Valstybėje pastatyta dešimtys biokurą deginančių katilinių, vykdoma tolesnė plėtra. Dėl kuro specifikos, šiuolaikines katilines galima laikyti didelio masto pramoniniais objektais. Norint užtikrinti patikimą jų funkcionavimą būtinas pastovus įrangos aptarnavimas ir techninė priežiūra.

Siekiama sukurti pavyzdinį katilinės modelį, kurį lyginant su eksploatuojamu objektu būtų galima pastebėti atsirandančius technologinių parametrų nukrypimus, realizuoti išankstinį galimų gedimų diagnozavimą, bei šalinimą.

Darbe apžvelgiami tinkamiausi modeliavimo metodai, jų pritaikymo galimybės. Išskiriami svarbiausi parametrai, kurie apibrėžia pakuros darbo režimą, analizuojama 5,3MWh galios katilinė su kondensaciniu ekonomaizeriu, dirbanti ištisus metus, skirtingo apkrovimo sąlygomis. Surinkti ir apdoroti kelių mėnesių istoriniai duomenys. Neuroninių tinklų metodu sukuriamas pakuros modelis. Naudojantis juo realizuojama katilinės darbo režimo stebėjimo sistema, kuri praneša naudotojui apie pastebėtus nukrypimus, ir leidžia prognozuoti galimai juos sukėlusius įėjimo parametrus. Pateikti sistemos bandymų rezultatai.

Reikšminiai žodžiai:biokuras, pakura, neuroniniai tinklai, katilinė, modelis.

Kovaliovas, Juozas. Simulation of biomass combustion chamber and investigation of operation modes. Final project of *masters qualification degree* / supervisor doc. dr. Kęstutis Brazauskas; Kaunas University of Technology, Faculty of Electrical and Electronics Engineering, department of Automation

Kaunas, 2015. 72 psl.

SUMMARY

The use of biomass for energy production is becoming increasingly popular in Lithuania. There are tens of already operating power plants, and the expansion of this energy sector is still present. Because biomass is not of volatile nature, the machinery required for automated burning process is more complex compared to traditional fossil fuels. In order to ensure proper function of these plants, regular checks and maintenance must be performed.

The goal of this paper is to create an accurate model of the biomass furnace, working in optimal conditions, which could be compared later, to the same furnace, during its operation. This would give the possibility for the responsible personnel to look for deviation in output parameters, which would allow early diagnosis of upcoming problems, thus providing an addition to existing maintenance procedures.

The paper describes several possible modeling methods and provides analysis of their suitability for the task. The analysed plant features a condensing economizer and the maximum power capacity is 5,3MWh. Heat is produced all year round, with varying loads on the system. The most important technological parameters, which describe operation of the furnace, are selected and their historical values are collected. A neural network based model is created using the data, together with a special system, which informs the user of parameter deviations. Furthermore, there is a possibility to identify the input parameters, which could have caused the deviation, using evolutional programming method based approach.

Keywords:biomass, power plant, neural network, model.

Turinys

Įvadas

Temos aktualumas: Biokuro deginimas – sparčiai plinantis šilumos energijos gamybos šaltinis. Kadangi kasmet taikomi vis griežtesni aplinkosaugos reikalavimai, 2020 metais Lietuva įsipareigojusi 20% energijos poreikio patenkinti iš atsinaujinančių šaltinių. Dėl išmetamųjų dujų balanso išlaikymo ekosistemoje, vienu iš tokių šaltinių laikomas biokuras. Valstybėje pastatyta dešimtys biokurą deginančių katilinių, vykdoma tolesnė plėtra.

Kadangi biokuras nepasižymi lakumu kaip naftos produktai ar dujos, jo deginimas palyginti sudėtingas procesas, jam taikomos kardinaliai skirtingos technologijos bei sprendimai. Šiuolaikines katilines galima laikyti didelio masto pramoniniais objektais, jose dirba didelis įvairių elektros, mechanikos, automatikos įrenginių skaičius. Norint užtikrinti patikimą jų funkcionavimą būtinas pastovus aptarnavimas ir techninė priežiūra. Esant ribotam aptarnavimo specialistų skaičiui ir nevienodam darbo intensyvumui kintant metų laikams, atsiranda poreikis sistemos, kuri galėtų prognozuoti artėjančius gedimus ar bent informuoti specialistą apie normalioje proceso būsenoje atsiradusius neatitikimus. Sistema padėtų išvengti netikėtų gedimų, kurie galėtų sąlygoti didelius ekonominius nuostolius, bei, šildymo sezonu turėti įtaką žmonių gyvenimo kokybei.

Darbo tikslas: Sudaryti metodiką, kuri tiktų biokuro pakuros darbo režimo monitoringui ir analizei. Sukurti sistemą, kuri leistų palyginti katilinės darbo parametrus su modeliu, informuotų naudotoją apie atsiradusius nukrypimus, bei identifikuotų įėjimo parametrus galimai sukėlusius nuokrypį.

Darbo uždaviniai:

- Išanalizuoti biokuro pakuros technologiją, naudojamas automatinio valdymo sistemas ir jų tarpusavio ryšius;
- Apžvelgti galimus modeliavimo metodus ir jų tinkamumą nagrinėjamai problemai spręsti;
- Surinkti istorinius katilinės darbo duomenis ir juos apdoroti;
- Sukurti stebėjimo sistemą, kuri automatiškai lygintų modelio išėjimus su realaus objekto išėjimais ir fiksuotų atsirandančias paklaidas. Taip pat, sistemoje įdiegti galimybę identifikuoti paklaidas sukėlusius įėjimo parametrus;
- Ištirti realizuotos sistemos veikimo tikslumą.

1. Bendros žinios apie biokuro sistemas

Medienos ir kito biokuro deginimas šilumos gavimui - pati seniausia technologija. Nors tokio tipo energijos populiarumas XX amžiuje buvo stipriai sumažėjęs, ji užima vis didesnę rinkos dalį paskutiniais dešimtmečiais. To priežastis – siekis naudoti atsinaujinantį, lengvai prieinamą ir pigų energijos šaltinį, bei vis griežtėjantys tarptautiniai aplinkosaugos reikalavimai.

Biokuro deginimo sistemų privalumai:

- Kuro kaina yra palyginti maža su iškastiniu kuru(nafta, dujos, anglis);
- Biomasė gali būti auginama lokalioje rinkoje, skatinama vietinė ekonomika;
- Kuras yra plačiai prieinamas ir atsinaujinantis;
- Granules deginančios krosnys pasižymi maža aplinkos tarša;
- Modernios biokuro deginimo sistemos išmeta mažiau teršalų į aplinką, nei anglies ir naftos deginimas;
- Miško atliekų panaudojimas, teikia naudą pačių miškų ekosistemoms.

Trūkumai:

- Degant medienai į aplinką išmetamos kenksmingos dujos(natrio oksidai, anglies monoksidas), bei suodžiai;
- Didelių medienos deginimo sistemų įrengimo kaina palyginti aukšta;
- Reikalingas pastovus medienos ar biomasės tiekimas;
- Reikalinga vieta kuro sandėliavimui;
- Po degimo proceso lieka pelenų, kurie turi būti tinkamai utilizuoti;
- Didelio masto katilinėms reikalinga operatorių priežiūra, dėl sistemų sudėtingumo;
- Biokuro kokybė gali kisti, taigi reikia daugiau pastangų siekiant išlaikyti stabilų darbą.

Efektyviausios biokuro jėgainės tos, kurios kartu gamina ir elektrą. Dažniausiai Lietuvoje statomose katilinėse naudojamas biokuras – medienos čipsai. Tai miškų kirtimo, valymo atliekos, stambiai smulkinta mediena. Nors deginant šį kurą į aplinką išskiriamas CO_2, tačiau biokuras laikomas atsinaujinančiu energijos šaltiniu, nes išskirta anglis nėra iškasta, o yra esamos ekosistemos dalis[1].

Pagrindinės biokuro katilinės sudedamosios dalys:

- Kuro sandėlis – patalpa, kurioje saugomas deginamas biokuras. Turi būti pritaikyta sunkiojo transporto privažiavimui, bei sutalpinti tokį kuro kiekį, kuris būtų pakankamas katilinei be perstojo dirbti pilnu pajėgumu 5 paras;
- Kuro transportavimo ūkis – techninių priemonių visuma, kuriomis kuras iš sandėlio perduodamas į pakurą. Taikomi įvairūs techniniai sprendimai, didesnėse katilinėse, kurą iš sandėlio į specialų bunkerį tiekia automatizuotas kranas. Mažesnės apimties katilinėse tokio tipo sprendimas palyginti brangus, taigi naudojami įvairių tipų transporteriai;
- Pakura – speciali kuro deginimui skirta patalpa. Kuras specialiais hidrocilindrais tiekiamas ant laiptuoto ardyno. Ventiliatoriais tiekiamas reikiamas degimui oras, dūmus ištraukia specialus dūmsiurblis, o po degimo proceso likusius pelenus surenka specialus transporteris. Su pakura dažnai komplektuojamas ir dūmų recirkuliacijos siurblys;
- Katilas – iš pakuros išeinantys dūmai eina per katilą ir atiduodą didžiąją dalį savo šilumos kitai terpei(dažniausiai vandeniui). Patenkančių dūmų temperatūra gali būti iki 1200ºC, išeinančių iš katilo siekia apie 300ºC;
- DKE(dūmų kondensacinis ekonomaizeris) – po katilo išeinantys dūmai vis dar turi perteklinės energijos. DKE specialios technologijos dėka padeda šią energiją pasisavinti;
- Multiciklonas, elektrostatinis filtras ir kiti dūmų valymo įrenginiai – degant biokurui susidaro didelis kiekis smulkiųjų pelenų, kurie nematomi plika akimi ir gali būti pavojingi sveikatai. Šių prietaisų tikslas maksimaliai sumažinti šių dalelių kiekį dūmuose;
- Kaminas – pagal aplinkosauginius reikalavimus, statomas tam tikro aukščio kaminas, siekiant užtikrinti kenksmingų cheminių medžiagų nepatekimą į gyvenamąsias zonas.

Šiame darbe gilinamasi tik į pakuros darbą, taigi plačiau apžvelgiamas šios sistemos dalies funkcionavimas.

2. Biokuro pakuros paskirtis ir veikimo principas

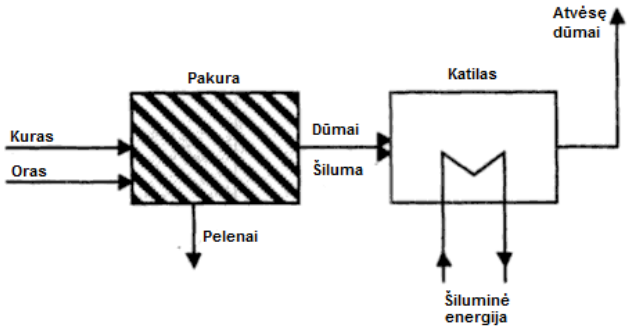
Kadangi biokuras iš prigimties nėra lakus, kaip, pavyzdžiui, dujos, jo sudeginimui reikalinga daug sudėtingesnė sistema. Jis deginamas specialioje patalpoje, kuri vadinama pakura. Joje įrengtas ardynas, ventiliatoriai, pelenų transporteris ir kiti būtini procesui įrengimai. Mūsų analizuojama – tai neaušinamo, judančio ardyno pakura. Pilnam ir kokybiškam kuro sudegimui būtinos salygos[2]:

- Tikslus kuro padavimo greičio valdymas;

- Tikslaus oro padavimo greičio valdymas į skirtingas pakuros zonas;
- Pakuros sugebėjimas atlaikyti aukštas temperatūras;
- Tikslus pakuros išplanavimas, kuris leistų pilnai sudegti visoms išsiskiriančioms cheminėms medžiagoms;
- Galimybė efektyviai šalinti pelenus.

Kuo geresnes degimo sąlygas siekiama palaikyti ne vien dėl ekonominių rodiklių, tačiau kartu ir dėl nuo reakcijos kokybės stipriai priklausančių susidarančių teršalų kiekių. Vykstant nesureguliuotam degimo procesui susidaro kenksmingos medžiagos[3]: anglies monoksidas(CO) – bespalvės, bekvapės, labai žmogui pavojingos dujos, susidarančios dėl nepakankamo oro kiekio degimo procese, bei natrio oksidai(NO_x) – gali sąlygoti rūgštaus lietaus susidarymą, kenkia ozono sluoksniui. Kai kuriuose katilinėse išmetimų kontrolei naudojamos specialios automatizuotųjų matavimų sistemos.

Degimo proceso efektyvumą sąlygoja keletas faktorių – šilumos praradimas per pakuros sienas, perteklinė šiluma pelenuose, nesudegusios dalelės dūmų sraute. Galutinis naudingumo koeficientas gali kisti nuo 65 – 95% ribose.



2.1 pav. Principinė katilinės schema.

2.1. Pagrindinių įrenginių ir parametrų aprašymai

Kuro tiekimas - vykdomas iš specialaus kuro bunkerio, į kurį kuras paduodamas transporteriais iš kuro ūkio. Į pakurą kuras beriamas hidrocilindrų judinamais žertuvais.

Pirminio oro srautas – vienas ar keli ventiliatoriai vykdo sustumto kuro intensyvų džiovinimą, pagrindinio degimo ant ardyno palaikymą. Tiekiamo deguonies kiekis visada stipriai viršija teorinį kiekį, reikalingą tik degimo cheminei reakcijai[4]. Oras pučiamas po ardynu,

paskirstomas į tris zonas. I.I ventiliatorius pučia pirmoje zonoje, kurioje vyksta pagrindinis kuro džiovinimas. Medienoje esančios drėgmės išgarinimui sunaudojama didžioji dalis energijos. Taip pat pirminio oro srautas netiesiogiai atlieka ardyno aušinimo funkciją. Antroje zonoje dirba I.II ventiliatorius – kuras toliau džiūna, išsiskiria įvairios dujos, vyksta degimo procesas. I.III ventiliatorius dirba trečioje zonoje, esant drėgnesniam kurui joje dar būna anglies likučių ir reikalinga pabaigti jų deginimą. Taip pat sudeginamos išsiskyrusios dujos. Visus ventiliatorius valdo atskiri PID reguliatoriai.

Antrinio oro ventiliatoriaus srautas – tiekia į pakurą orą, reikalingą biokuro degimui, nevisiško sudegimo produktams sudeginti.

Tretinio oro ventiliatoriaus srautas – tiekia į pakurą orą, skirtą visiškam degimo produktų sudeginimui. Jis atsakingas už pagrindinio degimo proceso kokybės rodiklio - tinkamo deguonies kiekio dūmuose palaikymą. Jei tretinio oro ventiliatoriaus našumo nepakanka norint sureguliuoti deguonies kiekį, tam pasitelkiamas antrinio oro ventiliatorius.

Trauka. Trauka pakūroje reguliuojama dūmsiurbiu atsižvegiant į esamo traukos daviklio parodymus. Dūmsiurblis valdomas dažnio keitikliu, dažniausiai taikomas metodas – tam tikru komunikacijų kanalu valdiklis siunčia besikeičiančią užduotį dažnio ketikliui.

Dūmų recirkuliacija – sumažina tam tikrų pakuros zonų temperatūrą, kiek įmanoma mažiau įtakojant degimo procesą. Dūmai išėję už katilo(arba multiciklono ar ESP) turi keliais šimtais laipsnių mažesnę temperatūrą, bei nedidelį deguonies kiekį, taigi gali būti efektyviai panaudoti vėsinimui. Speciliais ventiliatoriais jie gražinami į pakurą, priklausomai nuo konstrukcijos į vieną ar kelias zonas. Dėl mažo deguonies kiekio, degimo procesas neįtakojamas.

Katilas. Sudegus kurui, dūmai patenka į katilą – uždarą vamzdinę sistemą. Jame šiluma atiduodama kitam pernešėjui. Dažniausiai tai būna vanduo, kuris vėliau tiekiamas atlikti pasirinktoms užduotims.

2.2. Analizuojamos katilinės apibūdinimas

Darbe analizuojama katilinė turi 5MW vandens šildymo katilą ir iki 1,3MW galingumą galintį pasiekti dūmų kondensacinį ekonomaizerį. Ji skirta nedidelio miestelio šilumos tiekimo užtikrinimui, dirba skirtingu pajėgumu visus metus kartu su papildomais dujiniais katilais. Katilo galingumas parinktas taip, kad jį būtų galima panaudoti kaip vienintelį šilumos šaltinį vasaros metu, kūrenant minimaliu pajėgumu. Šalčiausiu žiemos laikotarpiu, biokuro katilas negali

patenkinti viso atsirandančio šilumos poreikio, įjungiami dujiniai katilai. Galios paskirstymas pasirinktas remiantis ekonominiais skaičiavimais.

Analizuojamos pakuros technologinė schema pateikiama priede numeris 4.

3. Katilinių aptarnavimas

Šiuolaikinės katilinės – didelio mąsto pramoniniai objektai, kuriuose veikia tūkstančiai prietaisų. Joms būtina pastovi operatorių priežiūra ir techninis aptarnavimas. Praktikoje, aptarnaujantis personalas dažniausiai turi prižiūrėti kelis katilus, šalinti atsirandančius smulkius gedimus ir taip užtikrinti nepertraukiamą sistemų funkcionavimą. Sudėtingesnius gedimus šalina ar profilaktinius patikrinimus atlieka samdomi specialistai ir technologai, turintys žinių specifinėse srityse. Dažniausiai atliekami gedimų šalinimai pagal iškvietimą ir techninė priežiūra pagal iš anksto nustatytą grafiką.

Esama daugybė įrenginių, kuriems tesingai neatliekant savo funkcijos, gali tekti stabdyti visos katilinės darbą. Kiekviena sistema turi jai būdingus eksploatacinius niuansus, kurie priklauso nuo konkrečios katilinės įrengimų išdėstymo, susidėvėjimo, oro temperatūros ir kitų veiksnių. Priklausomai nuo metų laiko, gedimo pasėkmė gali turėti didelę įtaką žmonių gyvenimo kokybei(centralizuoto šildymo katilinių atveju), bei atnešti didelius ekonominius nuostolius. Siekiant to išvengti, aktualu turėti analizavimo sistemą, kuri galėtų informuoti operatorius ir kitą aptarnaujantį personalą dar prieš įvykstant gedimui - kitaip tariant, pranešti apie atsirandantį parametrų ryšių neatitikimą, išrinkti galimai pakitusį dydį, apie kurį sužinojęs specialistas galėtų imtis atitinkamų veiksmų.

Apie konkrečius įrenginių gedimus informuoja SCADA sistema, tačiau ilgu eksploatacijos laikotarpiu palaipsniui atsirandantis mechanizmų susidėvėjimas, dilimas, užsinešimas ir panašūs pokyčiai nėra fiksuojami ir gali likti nepastebėti. Tokio tipo gedimų šalinimas apsiriboja profilaktiniais detalių keitimais, pagal nustatytus intervalus. Eksploatavimo metu, netikėtai atsiradus tokio tipo gedimams kyla įvairios problemos susijusios su jų šalinimu – dalių tiekimo terminai, specialistų trūkumas, dažnai reikalingas proceso stabdymas.

3.1. Darbo režimo stebėjimo sistemos veikimo principas

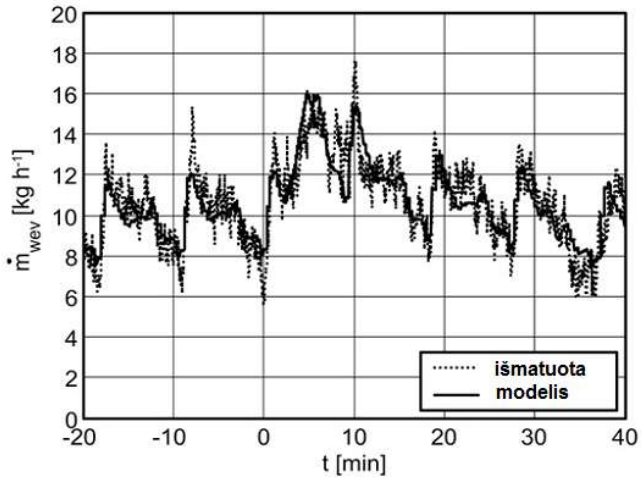
Kuriama sistema pagrįsta išsamiu eksploatuojamo objekto modeliu, skirtu atvaizduoti idealias eksploatacines sąlygas. Sukurtas modelis, atspindi optimalų katilinės darbo režimą, turint darbo parametrus jis lyginamas su eksploatuojamu objektu ir pagal atsirandančius neatitikimus padaromos išvados. Parametrų neatitikimas leidžia numanyti, kad tam tikra sistemos dalis savo funkciją atlieka netinkamai, kitaip nei fiksuotame modelyje. Priėjęs tokios išvados, technologas ar aptarnavimo inžinierius, remdamasis turimais duomenimis ir eksploatuojančio personalo žiniomis gali prognozuoti galimus gedimus ir imtis atitinkamų prevencinių veiksmų.

Kai kuriose katilinėse jau veikia nuotolinio stebėjimo sistemos, leidžiančios pasiekti pagrindinius darbo parametrus tinklu iš nutolusio serverio, taigi tyrimui davus teigiamus rezultatus nesunkiai galima realizuoti sistemą kelių katilinių stebėjimui nuotoliniu būdu per internetą.

4. Modeliavimo metodų apžvalga

4.1. Analitiniais metodais paremtas biokuro degimo procesų modeliavimas

Galima rasti nedidelių biokuro sistemų modeliavimo pavyzdžių pasitelkiant analitinius modelius. Pavyzdžiui R. Bauer, M. Golles, T. Brunner, N. Dourdumas publikacijoje „Modelling of grate combustion in a medium scale biomass furnace for control purposes“[5] pateikia ardyninio biokuro katilo, kūrenamo tam tikro drėgnumo kuru modelį, paremtą masės ir energijos balanso lygtimis. Modeliavimo ir eksperimentinių rezultatų palyginimas pateikiamas iliustracijoje:



4.1 pav. Modeliavimo rezultatai pritaikius analitinį modelį.

Dažniausiai tokio tipo modeliavimas realizuojamas išskaidant procesą į keletą atskirų dalių. Kaip pavyzdį panagrinėsime modelio struktūrą siūlomą „Mathematical model of gasification and combustion of biomass“[6] straipsnyje. Visų pirma, išskiriamas kaitinimo modelis, kuris aprašomas remiantis konvekcijos ir šilumos laidumo dėsniais. Vandens garavimo modelis grindžiamas eksperimentiniais vandens garavimo greičio duomenimis ir glaudžiai susijęs su katinimo modeliu. Terminio skilimo modelis aktualus, kai temperatūra viršija 300ºC. Šiai daliai taip pat pasitelkiami aproksimuoti eksperimentiniai rezultatai. Toliau seka cheminiai ryšiais paremti kietojo kuro degimo, lakaus kuro degimo modeliai. Abiejuose minėtuose straipsniuose sistemos modeliuotos griežtai apibrėžtoje aplinkoje su nustatyto drėgnumo kuru.

Didelėse katilinėse darbo sąlygos dažnai kinta, neapibrėžtas ne tik kuro drėgnumas, bet ir daugelis kitų veiksnių. Taip pat, eksperimentų atlikimas dažnai nepageidaujamas, nes gali sutrikdyti normalų sistemos darbą ir yra nuostolingas. Šie veiksniai lėmė analitinio modeliavimo metodo atmetimą, kaip tinkamo modelio kūrimui.

4.2. Neuroniniais tinklais pagrįstas modeliavimas

Daugelyje Lietuvoje esančių katilinių yra kaupiami dideli kiekiai istorinių eksploatacijos duomenų. Jie registruojami nepriklausomai nuo darbo režimo, leidžia susidaryti pilną objekto vaizdą ir plačiose ribose analizuoti proceso darbą. Būtent dėl lengvo duomenų prieinamumo pasirinktas modeliavimo neuroniniais tinklais metodas.

Siekiama išrinkti periodus, kuriuose procesas dirba teisingai, bei sukurti iš tų periodų išrinktais duomenimis pagrįstą modelį, kurį vėliau lyginant su esamais eksploataciniais duomenis būtų galima patikrinti ar šie telpa į iš anksto užsibrėžtus „teisingo darbo režimo“ rėmus. Duomenų išrinkimas vykdomas remiantis eksploatuojančio personalo žiniomis, apie proceso būklę įvairiais laiko momentais.

5. Eksploatacinių duomenų surinkimas

5.1. Katilinės SCADA sistema

Dažniausiai eksploataciniai duomenys saugomi kompiuteriniuose archyvuose, kuriuos tvarko proceso valdymui skirta SCADA(Supervisory Control And Data Acquisiton[7]) sistema. Jos dėka, už procesą atsakingas operatorius gali nesunkiai matyti scheminį proceso planą, prižiūrėti

įrenginius, bei koreguoti sistemos būseną atsižvelgdamas į poreikius(pavyzdžiui, trikdžių kompensavimas, didesnio katilo galingumo nustatymas). Istorinis parametrų registravimas leidžia atlikti ekonominius skaičiavimus, vertinti pamainų darbą, bei pastebėti atsirandančius proceso pokyčius.

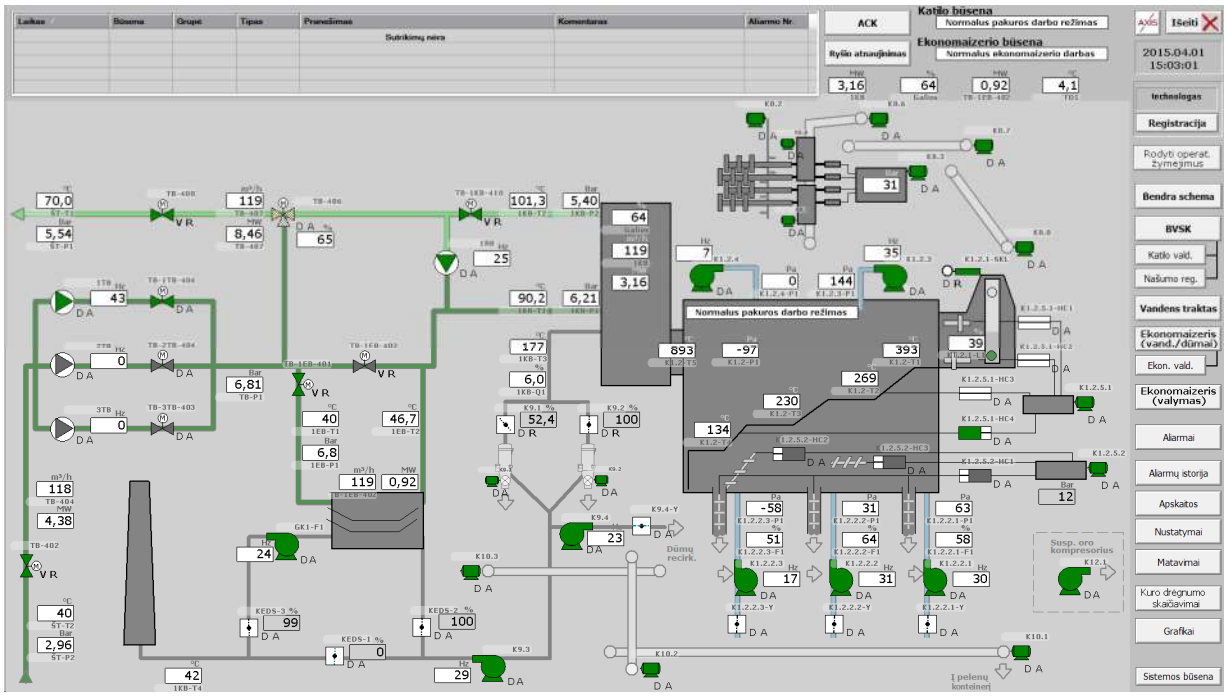
Darbe analizuojamoje katilinėje, procesą valdo atskiras kompiuteris, kuriame įdiegta tokia SCADA programinė įranga:

- Wonderware Archestra IDE(Integrated Development Environment), versija 3.6
- Wonderware WindowViewer 10.6
- Wonderware WindowMaker 10.6
- Pagalbinės programos paketo veikimui.

Pagrindiniai procesą atvaizduojantys langai:

- Bendra katilinės schema;
- Vandens šildymo katilas;
- Vandens traktas;
- Ekonomaizeris.

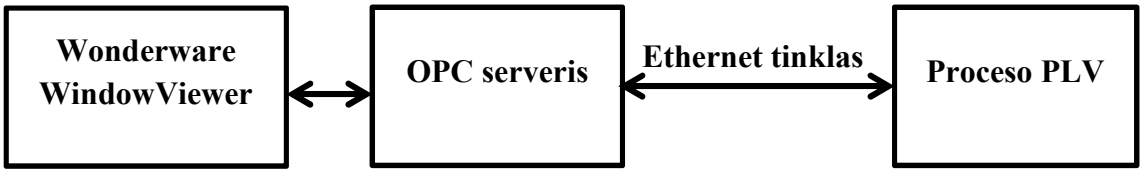
Šiuos langus papildo, kiti iššokantys langai su smulkesniais nustatymais.



5.1 pav. Pagrindinis SCADA sistemos langas.

Siekiant užtikrinti stabilumą ir saugumą, sistemoje yra naudojami keli prisijungimo lygiai, leidžiantys atskiroms vartotojų grupėms prieiti prie skirtingų parametrų rinkinių. Taip apsaugomos svarbiausių parametrų vertės nuo tyčinio ar netyčinio pakeitimo. Technologiniams rodikliams viršijus nustatytas ribines vertes, atvaizduojami aliarmo pranešimai, bei girdimas garsinis signalas, perspėjantis operatorius.

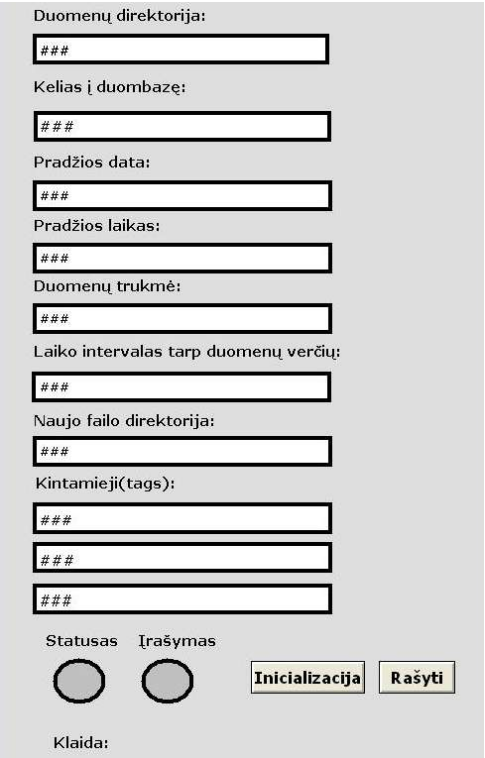
Duomenys į SCADA sistemą perduodami Ethernet tinklu, tiesiai iš valdiklio. Perdavimo struktūra:



5.2 pav. Duomenų į SCADA sistemą perdavimo struktūra.

Iš viso perduodama virš 1000 parametrų. Kai kurių iš jų verčių istorija saugoma naudojant InTouch programos standartinę istorinio registravimo(historical logging) funkciją. Duomenų vertės įrašomos į „*.lgh“ ir „*.idx“ failus, uždaru formatu, kurį galima tiesiogiai iššifruoti tik specialiais programiniais paketais. Kadangi darbe naudojamas Matlab programavimo paketas tiesiogiai nepalaiko šio duomenų formato atliktas duomenų išėmimas iš archyvo, naudojantis pagalbine Historical Data Manager programa[8]. Ši programa leidžia istorinius duomenis pateikti plačiai naudojamame Comma Seperated Values tipo faile, tačiau norint išskirti keletos kintamųjų vertes reikalingos tam tikros sistemos modifikacijos.

SCADA sistemoje sukurtas specialus papildomas langas ir 14 papildomų kintamųjų(tags), kurie reikalingi darbinių paprogramės parametrų išsaugojimui. Pagal pateiktą pavyzdį[9] aprašyti inicializavimo ir rašymo komandas atliekantys mygtukai. Duomenų nuskaitymui būtina paleisti HistData įrankį, kuris veikia atskirai nuo pagrindinės WindowViewer programos, bei inicializuoti parametrų vertes. Nustačius tinkamas vertes, spaudžiamas įrašymo mygtukas – programa pagal užduotus kriterijus išrenka duomenų vertes ir įrašo juos į nurodytą *.csv failą.



5.3 pav. Papildomas langas duomenų išėmimui.

Pagal nutylėjimą, duomenys sistemoje saugomi 90 dienų laikotarpiu, tačiau aptarnaujantis personalas pagal poreikį atlieka rankines duomenų kopijas. Bendras tyrimui naudotų duomenų intervalas 128 dienos. Išskirti keli intervalai:

- Nuo 2015 metų vasario 4 dienos iki 2015 metų kovo 31 dienos;
- Nuo 2014 metų lapkričio 6 dienos iki 2014 metų lapkričio 22 dienos;
- Nuo 2014 metų birželio 18 dienos iki 2014 metų rugsėjo 14 dienos.

Kiekviename iš šių intervalų užfiksuoti istoriniai duomenys išsaugomi „*.csv“ failuose, išrenkant reikiamus kintamuosius tinklo apmokymui. Papildomai atmesti periodai kuomet katilinė nedirba. Duomenų fiksavimo periodas – 30 minučių. Iš viso gauta daugiau nei 7000 matavimų rinkinių tam tikrais laiko momentais.

Analizuojamos katilinės darbo pradžia – 2013 metai. Atsižvelgiant į tai, kad vidutiniškai tokio tipo katilinei prognozuojamas 20 metų tarnavimo laikas, laikoma, kad tai sąlyginai naujas pramonės objektas. Gauti duomenys priimami kaip teisingi, parinkti esant nežymiam mechanizmų susidėvėjimui. Duomenų atitikimą reikalavimams patvirtino katilinę eksploatuojantis personalas.

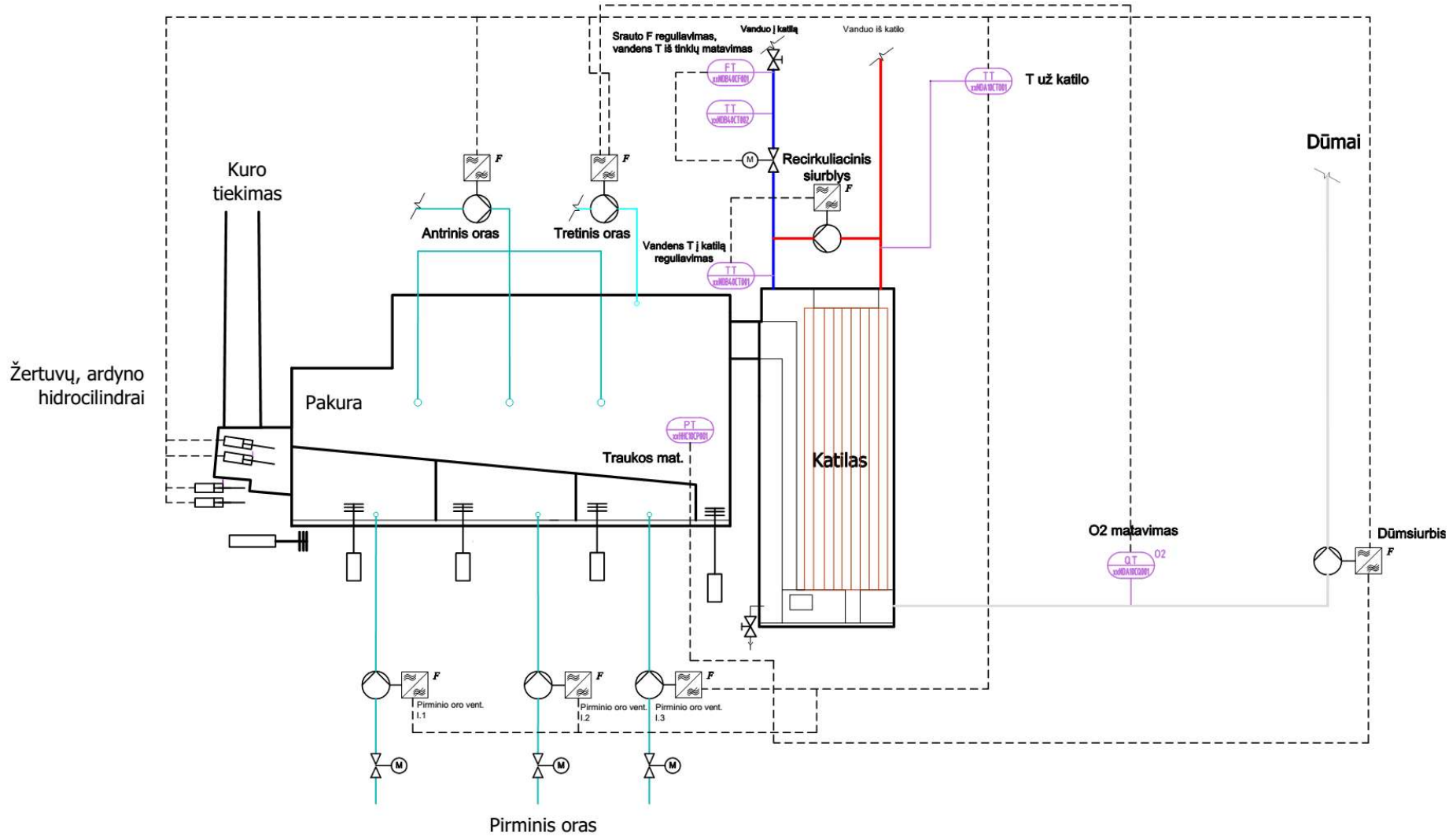
5.2. Technologinių parametrų išrinkimas

Biokuro pakuros ir katilo darbo stebėjimui naudojama daugiau nei 90 parametrų. Konsultuojantis su technologais pasirinkta 14 svarbiausių parametrų, kurie pilnai apibrėžia pakuros darbo režimą.

5.1 lentelė. Svarbiausi technologiniai parametrai.

Parametras	SCADA kintamojo vardas	Matavimo ribos	Technologinės ribos
I.1 oro ventiliatoriaus slėgis	K1221_P1_VALUE	-500...500 Pa	-100...100 Pa
I.2 oro ventiliatoriaus slėgis	K1222_P1_VALUE	-500...500 Pa	-100...100 Pa
I.3 oro ventiliatoriaus slėgis	K1223_P1_VALUE	-500...500 Pa	-100...0 Pa
II oro ventiliatoriaus slėgis	K123_P1_VALUE	0...1000 Pa	0...200 Pa
III oro ventiliatoriaus slėgis	K124_P1_VALUE	0...2000 Pa	0...600 Pa
Trauka pakuroje	K12_P1_VALUE	-250...250 Pa	-120...-70 Pa
Recirkuliacinio siurblio dažnis	1RB_DK_HZ_PV_VALUE	0...50 Hz	0...50 Hz
Dūmsiurbio dažnis	K93_DK_HZ_PV_VALUE	0...50 Hz	0...50 Hz
Vandens temperatūra po ekonomaizerio	1EB_T2_VALUE	0...120 ˚C	35...60 ˚C
Deguonies kiekis dūmuose	1KB_Q1_VALUE	0...21 %	4...12 %
Vandens temperatūra už katilo	1KB_T2_VALUE	0...150 ˚C	95...110 ˚C
Momentinė katilo galia	1KB_MWTH_MOM_VALUE	-	0...6 MWh
Srautas per katilą	1KB_Q_MOM_VALUE	0...200 m^3/h	20...180 m^3/h

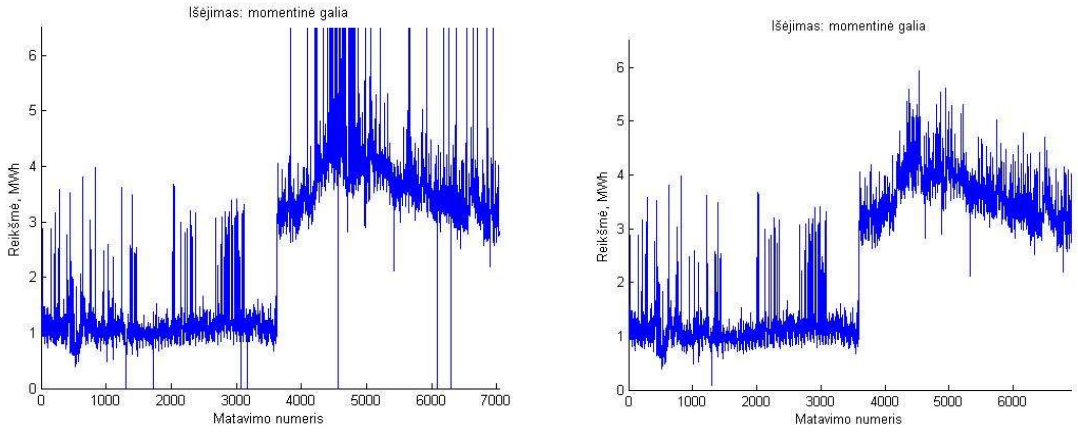
Sekančiame puslapyje pateikiama pakuros technologinė schema, bei atvaizduoti pasirinkti parametrai. Punktyrinės linijos žymi juos siejančius valdymo ryšius.



5.4 pav. Pakuros technologinė schema.

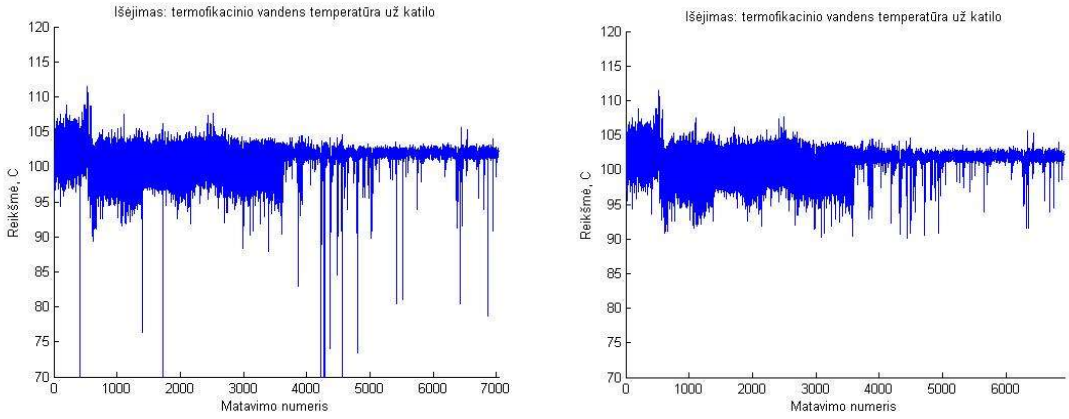
Pradiniuose duomenyse didelis triukšmo, neteisingų reikšmių kiekis(priedas nr. 1), taigi duomenų imtis analizuojama ir išrenkami neteisingi matavimai. Kadangi darbe analizuojamas ilgalaikis sistemos darbas, modelis orientuotas į stabilaus darbo proceso modeliavimą, o ne į dinaminių sistemos pokyčių fiksavimus, atmetamos parametrų vertės, pagal kurias galima spręsti, kad pakura ir katilas dirbo ne stabiliu, o pereinamuoju režimu.

Momentinės galios matavimuose dažnai pasitaiko reikšmių didesnių už maksimalią pasiekiamą katilinės galią, bei reikšmių žemesnių už nulį. Tokie duomenų rinkiniai fiksuojami ir pašalinami:



5.5 pav. Momentinės galios grafikai prieš ir po netinkamų duomenų pašalinimo.

Taip pat pašalinamos duomenų vertės, kai termofikacinio vandens temperatūra žemesnė nei 90 laipsnių Celsijaus. Esant tokiai parametro vertei, pakura arba nedirba, arba vyksta pereinamasis procesas.

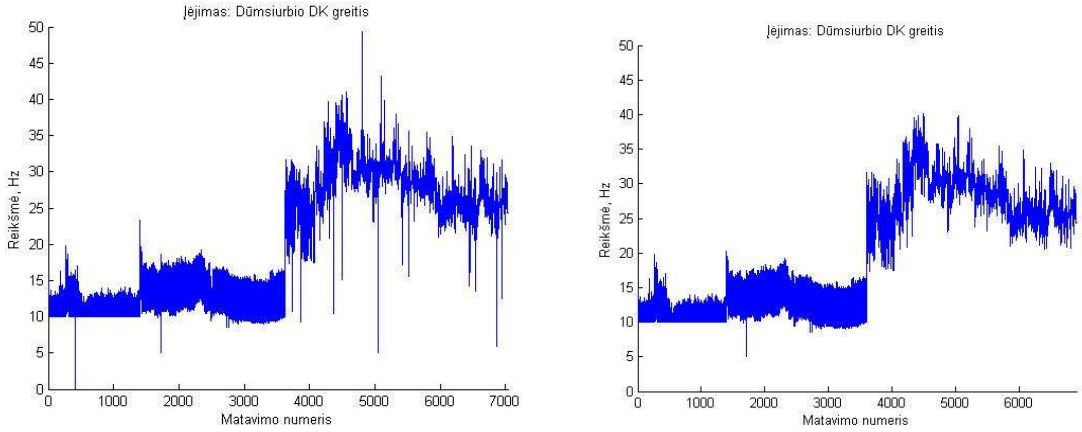


5.6 pav. Vandens temperatūros už katilo grafikai prieš ir po netinkamų duomenų pašalinimo.

Likusių duomenų verčių šalinimas atliekamas naudojant modifikuotą Tomsono tau metodą[10]:

1. Apskaičiuojamas duomenų sekos vidurkis x ir standartinis nuokrypis S;
2. Kiekvienam duomenų taškui apskaičiuojamas absoliutinis nuokrypio dydis:
$$\delta_i = |x_i - x| \quad (1)$$
3. Išrenkamas duomenų taškas turintis didžiausią δ_i reikšmę;
4. Apskaičiuojama τ reikšmė:
$$\tau = \frac{t_{\frac{\alpha}{2}} * (n-1)}{\sqrt{n} * \sqrt{n-2+t_{\frac{\alpha}{2}}^2}}. \quad (2)$$
Čia: n – duomenų taškų kiekis; tα/2 – kritinė t vertė(parenkama iš lentelių, pagal imties dydį);
5. Paeiliui tikrinama po vieną tašką. Kai $\delta_i > \tau * S$, taškas laikomas netinkamu. Kai $\delta_i < \tau * S$, taškas tinkamas.
6. Radus netinkamą tašką, jis pašalinamas ir procedūra kartojama nuo pirmo žingsnio, kol tokių taškų neberandama.

Mūsų atveju, siekiant pašalinti tik didžiausią nuokrypį turinčius taškus, naudota τ reikšmė τ=3,2. Gauti duomenys po filtravimo:



5.7 pav. Dūmsiurbio DK grafikai prieš ir po netinkamų duomenų pašalinimo.

Iš viso pašalinama 114 neteisingų parametrų rinkinių.

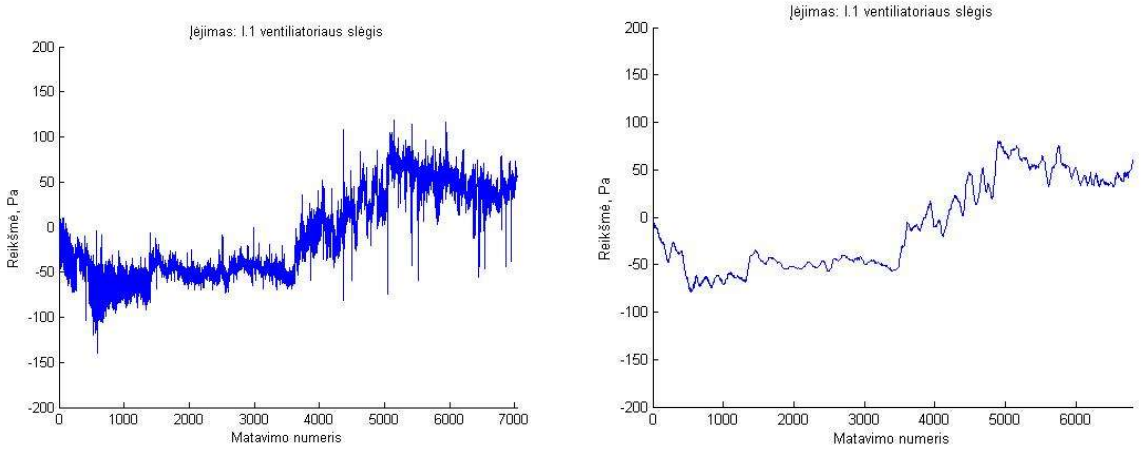
Parašyta Tomsono Tau metodu paremta pašalinių taškų aptikimo funkcija, kuri leidžia automatiškai išanalizuoti tam tikrą pasirinktą parametrą ir jo duomenų imties diapazoną. Išėjimo duomenų imties parametrai numeruojami paeiliui, pridedant įėjimo parametrų skaičių.

```
function [ Iejimai, Rezultatai ] = duomenu_filtravimas( Iejimai, Rezultatai )

parametru_vekt = [ 7; 7; 6; 7; 9; 13]; % Į vektorių įvedami parametrai,
kuriuos norima filtruoti
intervalu_vekt = [ 6300 6900; 4300 5200; 3650 4000; 1 3590; 5300 5700; 1 400]
; % Paeiliui, pagal įvestus parametrus įvedami filtravimo diapazonai,
kiekvienam iš jų
```

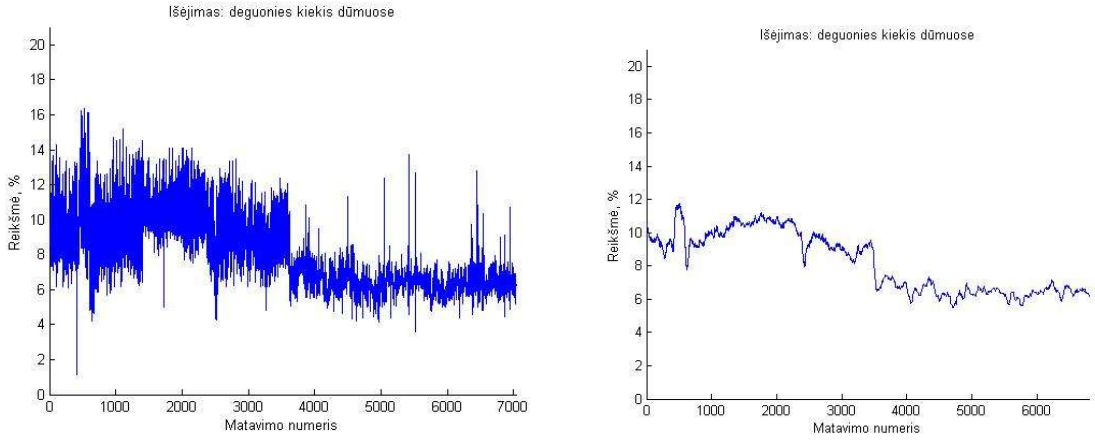
5.7 pav. Duomenų filtravimo funkcijos kodo fragmentas.

Pašalinus neteisingus matavimus, duomenys filtruojami naudojant Savitzky-Golay filtrą[11]. Pasirinkto ilgio duomenų imčiai, naudojant vidutinę kvadratinę paklaidą, kaip tikslo funkciją, apskaičiuojama pasirinkto laipsnio polinomo funkcija. Viduryje intervalo randama polinomo reikšmė ir ji priimama kaip filtruoto signalo dėmuo. Tuomet analizuojama imtis perkeliama tolyn per vieną narį ir procedūra kartojama.

Duomenys po filtravimo(visų parametrų grafikai pateikiami priede nr. 2):



5.8 pav. I.1 ventiliatoriaus slėgio grafikas prieš ir po duomenų filtravimo.



5.9 pav. Deguonies kiekio dūmuose vertė prieš ir po filtravimo.

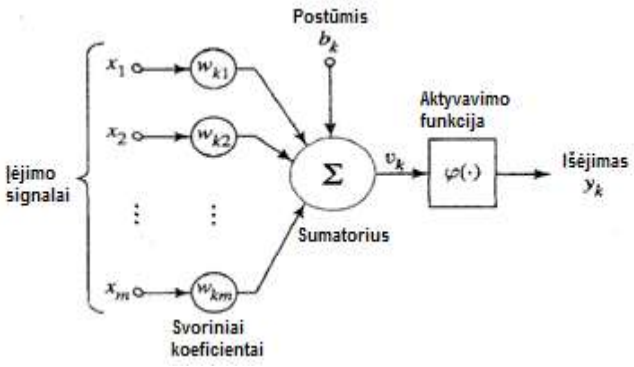
6. Pakuros modelio kūrimas

6.1. Dirbtinių neuroninių tinklų modeliavimo metodo apžvalga

Dirbtiniai neuroniniai tinklai – tai matematinės struktūros, siekiančios atkartoti žmogaus smegenų darbą[12]. Dirbtinis neuronas – tai biologinio neurono abstrakcija. Nors tai nėra preciziškiausias metodas įvairių uždavinių sprendimui, jis pasižymi gebėjimu nesudėtingai aproksimuoti kompleksiškas netiesines matematines sistemas. Dirbtinis neuronas turi keletą įėjimų(dažniausiai žym. x_1, x_2, ..., x_n) ir vieną išėjima(y), kurio reikšmė gaunama pagal formulę:

$$y = \phi\left(\sum_{j=0}^{n} w_j x_j\right) \tag{3}$$

Koeficientai w vadinami įėjimo svoriais, o funkcija ϕ - aktyvavimo funkcija. Įėjimų reikšmės dauginamos iš svorių, pridedamas postūmis ir gauta reikšmė siunčiama į perdavimo funkciją. Skiriamos kelių rūšių perdavimo funkcijos – sigmoidė, šuolinė funkcija, tiesinė funkcija. Atlikus skaičiavimus gauta reikšmė priskiriama neurono išėjimui. Atskiri dirbtiniai neuronai jungiami į tinklus, kuriuose neuronai skirstomi į sluoksnius[13].



6.1 pav. Dirbtinio neurono principinė schema.

Iš dirbtinių neuronų sudarytas tinklas apmokomas, pateikiant jam eksperimentinius duomenis. Pagal juos keičiami neuronus siejančių jungčių svoriai, kol pasiekiamas tam tikras užduotas tikslumo kriterijus.

6.2. Neural network toolbox įrankio apžvalga

Dėl plataus funkcijų spektro skirto neuroninių tinklų kūrimui, pasirinktas Matlab programavimo paketo Neural network toolbox įrankis.

Įrankis leidžia lengvai apdoroti surinktus duomenis, sukurti tinklą, jį apmokyti. Skiriami keturi įrankio naudojimo lygiai[14]:

1. Grafinių vartotojo sąsajų naudojimas – greitas būdas apdoroti duomenis, turintis ribotą funkcijų skaičių;
2. Tekstinių komandų naudojimas – leidžia pasiekti visas gamintojo siūlomas funkcijas, pakeisti pradines parametrų reikšmes, siekiant pritaikyti įrankio darbą;
3. Neural network toolbox pritaikymas – įrankio funkcijų ir programų tekstų redagavimas, išlaikant originalias kopijas;
4. Neural network tooblox keitimas - įrankio funkcijų ir programų tekstų redagavimas.

Palaikomi tokio tipo tinklai:

- Mokymosi su mokytoju tinklai – kartu pateikiamos ir įėjimo ir išėjimo reikšmės. Tinklo svoriai keičiami taip, kad išėjimo reikšmės sutaptų su užduoties reikšmėmis;
- Mokymosi be mokytojo tinklai – turima tik įėjimo imtis, formuojami tinklo svoriai, ir įėjimo duomenys suskirstomi į klases ar klasterius.

Mūsų atveju, sistemoje bus naudojamas mokymas su mokytoju, naudojant tekstines komandas tinklo sukūrimui ir panaudojimui.

Skiriami septyni tinklo kūrimo žingsniai:

1. Duomenų surinkimas;
2. Tinklo sukūrimas;
3. Tinklo konfigūracija;
4. Pradinių parametrų nustatymas;
5. Tinklo apmokymas;
6. Tinklo testavimas;
7. Tinklo panaudojimas.

Pirmasis žingsnis aprašytas praeituose skyriuose.

Dažniausiai mokymosi su mokytoju autoasociatyviniams neuroniniams tinklas siūloma naudoti sigmoidės tipo perdavimo funkcijas vidiniame sluoksnyje ir tiesinę perdavimo funkciją išėjimo sluoksnyje. Keli sluoksniai neuronų turinčių netiesiškas perdavimo funkcijas, leidžia tinklui sėkmingai aproksimuoti netiesines priklausomybes[15].

6.3. Neuronio tinklo apmokymas

Prieš pateikiant duomenis į testavimui skirtą tinklą, jie sunormuojami pagal technologines ribas į 0...1 intervalą.

Atliekamas tinklo apmokymas su filtruotais ir nefiltruotais duomenimis.

Duomenys padalinami į tris imtis: 70% tinklo apmokymui, 15% validavimui, 15% testavimui. Duomenų padalinimas vykdomas naudojant funkciją dividerand – atsitiktinis parinkimas. Naudotas trainlm algoritmas, paslėptame sluoksnyje 10 neuronų.

6.1 lentelė. Apmokymo nefiltruotais duomenimis rezultatai.

Apmokymo numeris	Iteracijų skaičius	Vidutinė kvadratinė paklaida
1	48	$9{,}7904*10^{-4}$
2	32	$9{,}9945*10^{-4}$
3	100	$9{,}4773*10^{-4}$
4	131	$9{,}4724*10^{-4}$
5	67	$9{,}7285*10^{-4}$
Vidutinės vertės	75,6	$9{,}6926*10^{-4}$

6.2 lentelė. Rezultatai gauti apmokant tinklą filtruotais duomenimis.

Apmokymo numeris	Iteracijų skaičius	Vidutinė kvadratinė paklaida
1	55	$3{,}3769*10^{-4}$
2	65	$4{,}2991*10^{-4}$
3	30	$3{,}8386*10^{-4}$
4	70	$5{,}0332*10^{-4}$
5	73	$3{,}8063*10^{-4}$
Vidutinės vertės	58,6	$4{,}0708*10^{-4}$

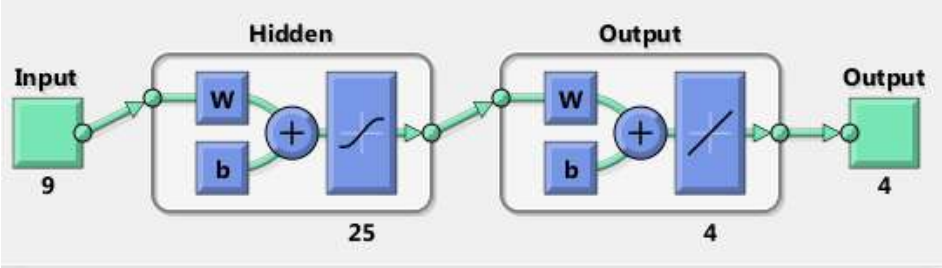
Gauti rezultatai rodo, kad daug didesnis tikslumas pasiekiamas naudojant filtruotus duomenis. Siekiant parinkti tinkamiausią neuronų skaičių, atlikti tinklo apmokymai palaipsniui keičiant parametro vertę.

6.3 lentelė. Paklaidų priklausomybė nuo paslėpto sluoksnio neuronų skaičiaus.

Neuronų skaičius	**5**	**10**	**15**	**20**	**25**	**30**	**35**
Apmokymų paklaidos ($*10^{-4}$)	8,2644	3,3769	2,9576	1,5679	1,5032	1,1223	1,1726
	8,6606	4,2991	2,7040	1,6305	1,2361	1,501	1,4388
	9,5267	3,8386	2,6586	1,5618	1,1844	1,3842	1,153
	9,4321	5,0332	2,8248	1,5636	1,4201	1,0923	1,4042
	8,6570	3,8063	2,6145	1,6566	1,1936	1,6331	1,4661
Vidurkis	8,90816	4,07082	2,7519	1,59608	1,30784	1,34658	1,32694

Kadangi neuronų skaičiaus didinimas virš 25 neuronų pastebimo paklaidos sumažėjimo nesuteikia, nuspręsta naudoti 25 neuronų tinklą modeliavimui.

Siekiant surasti greičiausiai veikiantį ir geriausią tikslumą pasiekiantį apmokymo algoritmą, išbandyti Levenbergo-Markardo, Kvazi-Niutono ir Bajeso algoritmai. Jų pasirinkimą lėmė analogiškų uždavinių sprendimo pavyzdžiai literatūroje[14]. Testavimui naudoti tinklai su 25 neuronais paslėptame sluoksnyje.



6.2 pav. Neuroninio tinklo schema.

„Trainlm“ funkcija naudojanti Levenbergo-Markardo algoritmą.

6.4 lentelė. Apmokymo Levenbergo-Markardo algoritmu rezultatai.

Apmokymo numeris	Iteracijų skaičius	Vid. kv. paklaida	Laikas
1	45	$1,2153*10^{-4}$	74
2	63	$1,1939*10^{-4}$	101
3	40	$1,6687*10^{-4}$	66
4	53	$1,2634*10^{-4}$	80
5	49	$1,1969*10^{-4}$	79
Vidurkis	50	$1,30764*10^{-4}$	80

Trainbr funkcija naudojanti Bajeso algoritmą.

6.5 lentelė. Apmokymo Bajeso algoritmu rezultatai.

Apmokymo numeris	Iteracijų skaičius	Paklaida	Laikas
1	1000	$1{,}3153*10^{-4}$	1544
2	500	$1{,}1796*10^{-4}$	787
3	250	$1{,}1618*10^{-4}$	392
4	250	$1{,}3196*10^{-4}$	460
5	350	$1{,}2161*10^{-4}$	591
Vidurkis	470	1,23848	754,8

Trainbfg funkcija naudojanti Kvazi-Niutono algoritmą:

6.6 lentelė. Apmokymo Kvazi-Niutono algoritmu rezultatai.

Apmokymo numeris	Iteracijų skaičius	Paklaida	Laikas
1	95	$3{,}2482*10^{-4}$	20
2	112	$3{,}5658*10^{-4}$	23
3	143	$2{,}5854*10^{-4}$	28
4	77	$4{,}3803*10^{-4}$	17
5	141	$3{,}7057*10^{-4}$	29
Vidurkis	113,6	$3{,}49708*10^{-4}$	23,4

Remiantis eksperimentiniai duomenimis, pasirinktas Bajeso algoritmas.

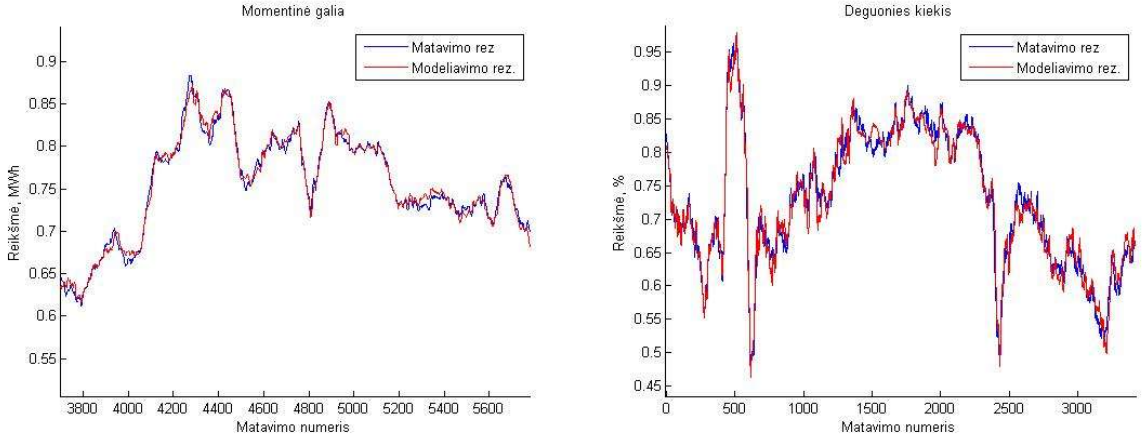
Siekiant didesnio modeliavimo paklaidų stabilumo, apmokytas iš trijų neuroninių tinklų sudarytas kolektyvas. Kiekvienas tinklas turi vienodą svorį, gaunami rezultatai sumuojami ir dalijami iš kolektyvo narių skaičiaus.

6.7 lentelė. Naudojant 3 tinklų kolektyvą gautos paklaidos.

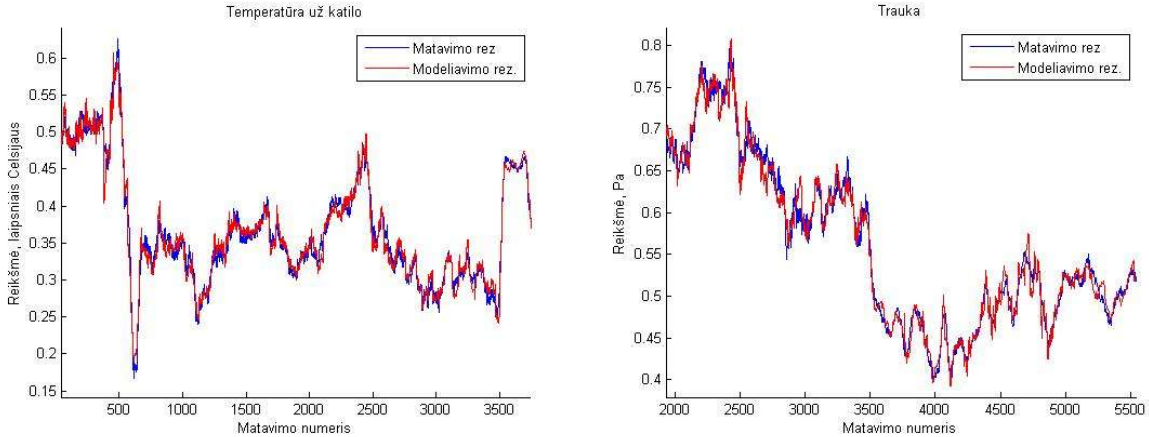
Apmokymo numeris	Paklaida
1	$1{,}1221*10^{-4}$
2	$1{,}2941*10^{-4}$
3	$1{,}2310*10^{-4}$
4	$1{,}3267*10^{-4}$
5	$1{,}0063*10^{-4}$
Vidutinė vertė	$1{,}2160*10^{-4}$

Naudojant 3 tinklų kolektyvą gaunamas mažesnis paklaidų pasiskirstymas, taigi stabilesnis modeliavimas[16]. Naudojant vieną neuroninį tinklą paklaidų standartinis nuokrypis lygus 0,1451, naudojant 3 tinklų kolektyvą – 0,1317.

Gauti modeliavimo grafikai(mastelis pakeistas siekiant geresnio atvaizdavimo, pilno mastelio grafikai pateikti priede nr. 3):



6.3 pav. Momentinės galios ir deguonies kiekio modeliavimo rezultatai.



6.4 pav. Vandens temperatūros už katilo ir traukos pakuroje modeliavimo rezultatai.

Katilinės darbas reguliuojamas pagal termofikacinio vandens temperatūrą už katilo, kuriai leistina paklaida yra ±3˚C. Vidutinė absoliutinė modeliavimo paklaida šiam parametrui siekia 0,47%. Pagal bendrą matuojamą interval tai sudaro 0,47%*15/100%=0,0705 ˚C. Laikome, kad tinkle pasiekiami rezultatai yra pakankamai tikslūs nagrinėjamai problemai spręsti.

7. Darbo režimo stebėjimo sistemos realizavimas

Sukuriama speciali stebėjimo sistema, kuri iteraciškai apdoroja jai pateikiamus eksploatacinius duomenis ir lygina objektą su modeliu. Stebimas neatitikimas tarp modelio ir objekto atskirų išėjimų parametrų verčių:

$$e = |y_{i,tinklo} - y_{i,sistemos}| \tag{4}$$

Paklaidos filtruojamos ir gauta vertė lyginama su tinklo apmokymo metu gauta modeliavimo paklaidos tam parametrui standartinio nuokrypio verte. Filtravimas realizuotas naudojant slenkančio vidurkio filtrą[17]:

$$e_{filtruota}(i) = \frac{1}{M} \sum_{j=k-(M-1)}^{k} |y_{i,tinklo} - y_{i,sistemos}| \tag{5}$$

Čia x – įėjimo signalas, y_i – pasirinktas išėjimo signalas, M – vidurkiui skaičiuoti reikalingų reikšmių skaičius, k – parametrų rinkinio numeris. Nagrinėjamu atveju, eksperimentiniu būdu pasirinktas 100 paskutinių verčių filtras, kadangi stebimi ilgainiui atsitinkantys gedimai.

Jei bent vieno iš išėjimų nuokrypis viršija 3*σ vertę, nagrinėjamas parametrų vektorius laikomas neatitinkančiu teisingo katilinės darbo režimo. Tokiu atveju, sistema registruoja viršytų taškų-pusvalandžių skaičių.

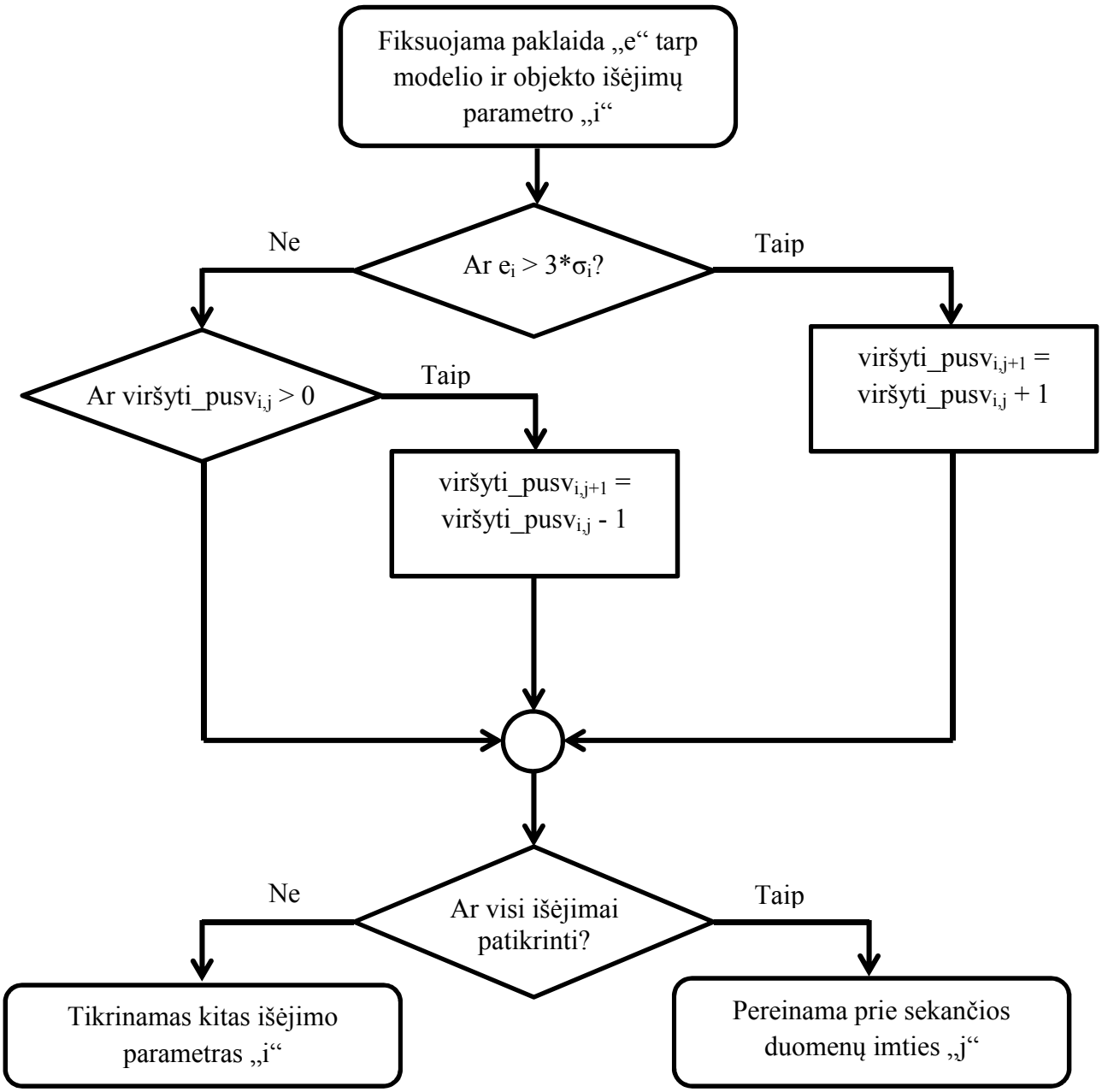
```
    if Filtruota_paklaida(j,i) > verte_ispejimui(j)

virsyti_pusv(j,i+1) = virsyti_pusv(j,i) + 1;

    else
        if virsyti_pusv(j,i) > 0
        virsyti_pusv(j,i+1) = virsyti_pusv(j,i) - 1;
        else
        virsyti_pusv(j,i+1) = 0;
        end
    end
```

7.1 pav. Programos kodo dalis, kurioje sumuojamas parametro viršytų pusvalandžių skaičius.

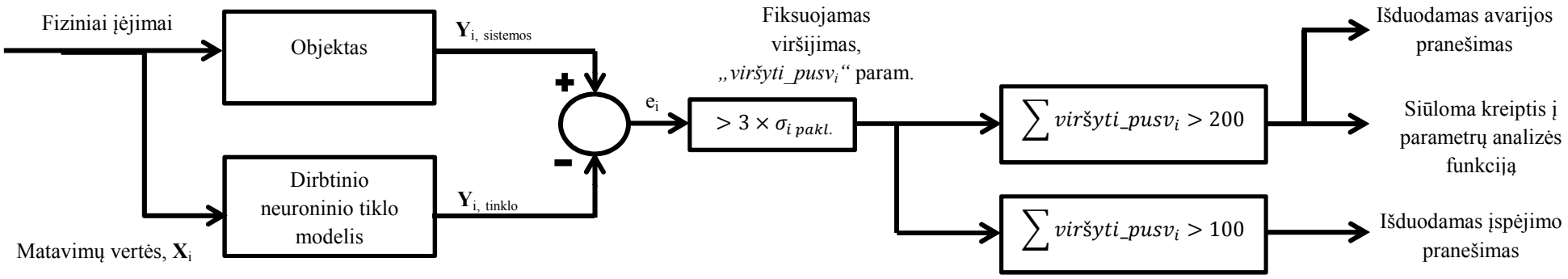
Apskaičiuotos standartinio nuokrypio vertės išėjimo parametrams:

Parametro numeris	σ_i
1	0,0113
2	0,0109
3	0,0078
4	0,0054
Bendrai	0,0089

Taikomos taisyklės:



Sistemos funkcinė schema:



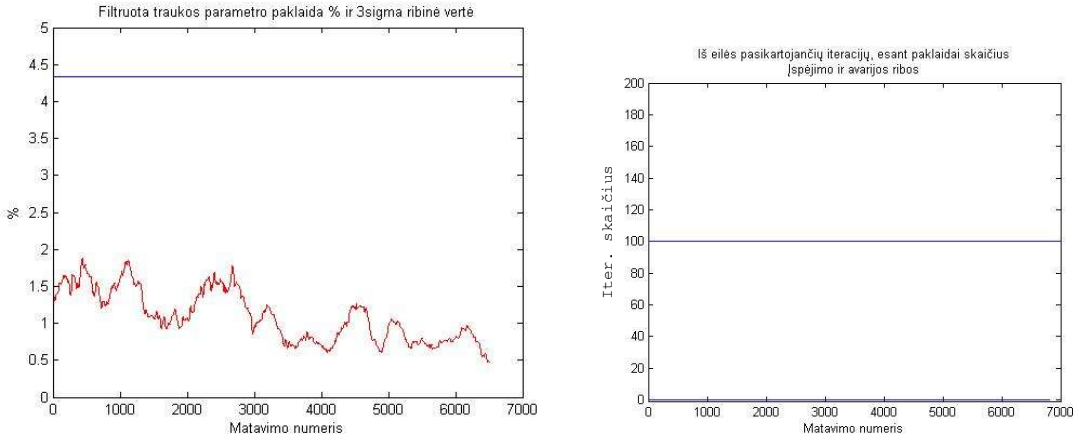
Tik atsiradus viršijimui, naudotojas nėra įspėjamas, nes galimi proceso pereinamieji režimai, kurie sąlygoja laikiną paklaidų padidėjimą. Remiantis technologinėmis žiniomis, pasirinktas 100 viršytų pusvalandžių skaičius, kaip riba, kurią gali peržengti tik nuolatos nukrypęs procesas. Įprastai pasitaikantys trikdžiai(techninė priežiūra, prastas kuras, įrenginių sustojimai ir kita) negali sukelti 50 valandų trunkančio parametrų nuokrypio.

Viršytų pusvalandžių skaičiui nustatomos dvi ribos: įspėjimas ir avarija. Įspėjimo riba – 100 viršytų pusvalandžių vertės. Avarijos riba – 200 viršytų pusvalandžių. Viršijus nustatytas vertes, į ekraną išvedami pranešimai, bei, avarinės ribos atveju, pasiūloma iškviesti analizės funkciją, kuri leistų prognozuoti galimus įėjimo parametrų nuokrypius.

```
    if virsyti_pusv(i+1) == 100 && virsyti_pusv(i) > virsyti_pusv(i-1) %
kokiam viršytų iteracijų skaičiui esant bus įspėjimas
    z = msgbox({'SVARBU!' 'Didelis parametrų nuokrypis!' });
    end

    if virsyti_pusv(i+1) == 200 && virsyti_pusv(i) > virsyti_pusv(i-1)
    z = msgbox({'ĮSPĖJIMAS!' 'Labai didelis parametrų nuokrypis!' });

    prompt = 'Ar kreiptis į analizės funkciją? Y/N [Y]: ';
```

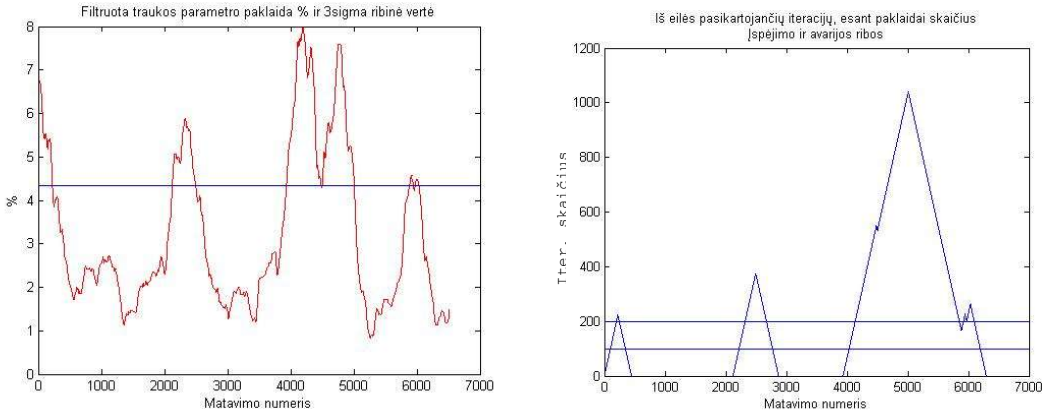
7.2 pav. Įspėjimų išvedimo programos kodo dalis.

Stebėjimo sistema išbandoma pateikiant jai turimus eksploatacinius duomenis.



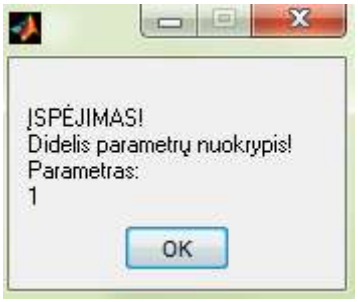
7.3 pav. Paklaidos ir jos pasikartojimo grafikai, gauti pateikus sistemai nepakeistus duomenis.

Pirmo įėjimo parametrui suteikiamas 10% nuokrypis. Gauta paklaidos atvaizduojamos:



7.4 pav. Paklaidos ir jos pasikartojimo grafikai, gauti pateikus sistemai pakeistus duomenis.

Kai viršytų pusvalandžių skaičius pasiekia 100, programos naudotojui pateikiamas įspėjimo iššokantis pranešimas:



7.5 pav. Iššokančio pranešimo langas.

Analogiškas pranešimas pateikiamas ir viršytų pusvalandžių skaičiui viršijus du šimtus. Tuomet vartotojui siūloma kreiptis į parametrų analizės funkciją, kuri naudodamasi modeliu, įėjimų ir išėjimų parametrų rinkiniu, bei turima technologine informacija bando nustatyti galimai nukrypusį parametrą.

7.1. Parametrų analizės funkcija

Atsiradus dideliam modelio ir realaus objekto parametrų nesutapimui iškviečiama parametrų analizės funkcija. Funkcija gali atlikti dviejų skirtingų tipų analizę:

- Pavienio nukrypusio parametro paieška;
- Įėjimo parametrų rinkinio paieška, remiantis ryšių matricomis.

Analizės funkcijos programiniai įėjimai:

- Paskutinių penkių pusvalandžių išėjimo parametrų verčių vektorius (Isejimai = $\mathbf{y}_{i-5,j}...\mathbf{y}_{i,j}$);
- Paskutinių penkių pusvalandžių įėjimo parametrų verčių vektorius(Iejimai = $\mathbf{x}_{i-5,j}...\mathbf{x}_{i,j}$);
- Dirbtinis neuroninis tinklas;
- Tėvų skaičius(evoliucinio programavimo algoritmui);
- Įėjimų ir išėjimų koreliacijos matrica;
- Paskutinių penkių pusvalandžių paklaidų vertės (Filtruota_paklaida = $\mathbf{e}_{i-5,j}...\mathbf{e}_{i,j}$);
- Neuroninio tinklo modeliavimo paklaidų standartinių nuokrypų vertės(standartinis_nuokrypis = $\sigma_1...\sigma_i$).

Funkcijos išėjimai – surastas įėjimų rinkinys ir gauti įėjimai. Abiejų tipų paieškose, parametrų optimizavimui naudojamas evoliucinio programavimo algoritmas.

Bendrai galima teigti, jog funkcija yra rekomendacinio pobūdžio įrankis aptarnaujančiam specialistui, kuris gali pateikti tik orientacinius gedimo paieškos taškus. Dėl proceso sudėtingumo, galimų pokyčių pačiame įrenginyje, bei nedidelių turimų duomenų, funkcija negali tiksliai identifikuoti gedimo vietos.

7.1.1. Evoliucinio programavimo metodo apžvalga

Evoliuciai algoritmai paremti natūralioje evoliucijoje pasitaikančiomis idėjomis. Dažniausiai tokio tipo metodai taikomi netiesinėms sistemoms, bei turint triukšmingus duomenis.

Visi evoliuciniai algoritmai paremti populiacijos prisitaikymu. Bendrai algoritmą galima apibūdinti keturiais žingsniais[18]:

1. Iteracijos skaičius nustatomas i=0;
2. Atsitiktinai sukuriama pradinė populiacija P(i);
3. Kartojami žingsniai:
    a. Apskaičiuojama tikslo funkcijos vertė kiekvienam individui;
    b. Pagal tikslo funkcijos vertę iš P(i) populiacijos atrenkami sekančios populiacijos „tėvai“;
    c. Pritaikoma mutacijos funkcija ir sukuriama populiacija P(i+1);
4. Algoritmas baigiamas pasiekus užduotą tikslo funkcijos vertę arba iteracijų skaičių.

Mutacija dažnai išreiškiama pridedant prie tėvų vektoriaus Gauso pasiskirstymo atsitiktinius dydžius. Nagrinėjamu atveju naudojama formulė:

$$\boldsymbol{x}_i^* = \boldsymbol{x}_i + \Delta\boldsymbol{x} * RAND \quad (6)$$

Čia RAND – atsitiktinis Gauso pasiskirstymo kintamasis, su nuline matematine viltimi M{RAND}=0 ir vienetine dispersija D{RAND}=1. Δx_j, j=1,...,n – parametrai nustatantys „mutacijų“ laipsnį. $\boldsymbol{x}_i$ – tėvų vektorius, $\boldsymbol{x}_i^*$ - naujai sukurtas „palikuonių“ vektorius. Nagrinėjamu atveju $\Delta x = 0{,}05$, tėvų skaičius – 10.

Tikslo funkcija – tai paklaida tarp nustatytų ir tinklo duodamų įšėjimų, kurią stengiamasi sumažinti keičiant įėjimų vertes:

$$f(x) = \sum |\boldsymbol{y}_{\boldsymbol{nust}} - \boldsymbol{y}_{\boldsymbol{tinklo}}| \quad (7)$$

$$J = f(x) \to min$$

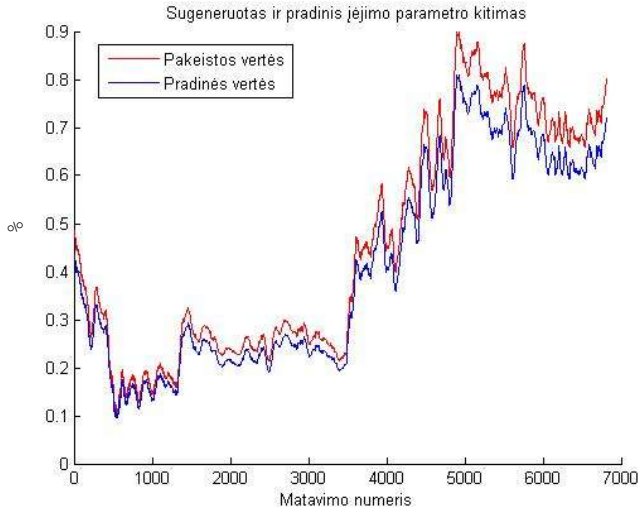
Evoliuciniui algoritmui taikomi apribojimai – gauti duomenys gali būti tik 0...1 ribose, nes būtent tokių ribų duomenimis apmokytas tinklas. Jei gaunamas palikuonis, kurio bent viena dedamoji yra išeinanti už ribų, jo „mutacija“ atliekama iš naujo, kol gaunamas į ribas papuolantis vektorius. Algoritmas kartojamas tam tikrą iteracijų skaičių arba kol pasiekiama nustatyta paklaida.

7.1.2. Pavienio nukrypusio parametro paieška

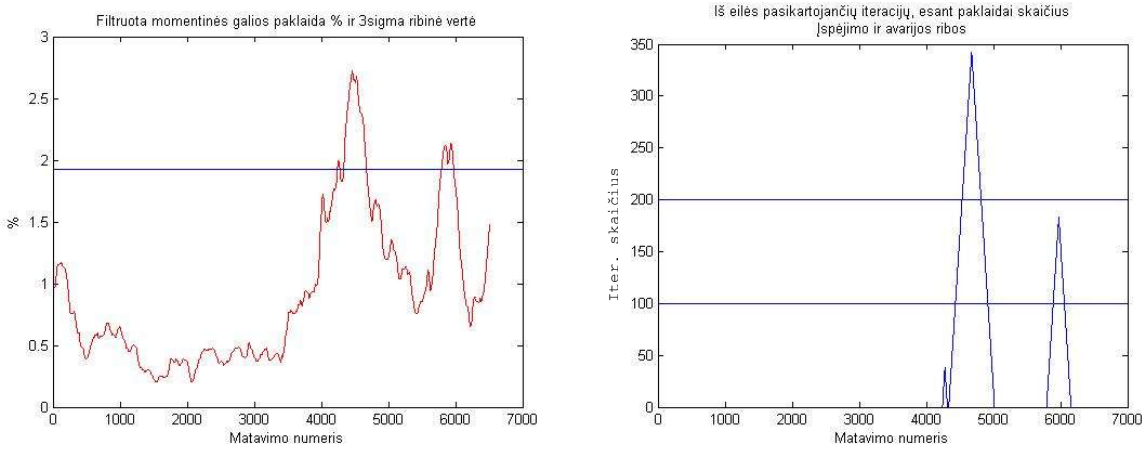
Gana dažnai praktikoje pasitaiko matavimo prietaisų gedimų. Nesant lengvo būdo atlikti paralelius matavimus stebimiems parametrams, aptarnaujančiam personalui ne visada gali būti lengva įsitikinti tam tikro prietaiso funkcijos sutrikimu. Šis algoritmas leistų aptikti tokio tipo gedimus.

Naudojant šią funkcijos dalį, evoliucinis algoritmas paeiliui keičia kiekvieną įėjimo parametrą $x_{i,j}$ ir siekia prilyginti tinklo išėjimo vertes $\mathbf{y}_{i,\ tinklo}$ funkcijos iškvietimo metu gautoms realaus objekto išėjimų vertėms $\mathbf{y}_{i,\ sistemos}$. Minimalios, kiekvienam įėjimui gautos paklaidos įrašomos į matricą(„paklaida_isejimu“), taip pat užfiksuojami evoliucinio algoritmo sugeneruoti įėjimo parametrų rinkiniai(„vekt_iejimu_bendras“). Mažiausią paklaidą pasiekęs parametras turi didžiausią tikimybę būti nukrypęs.

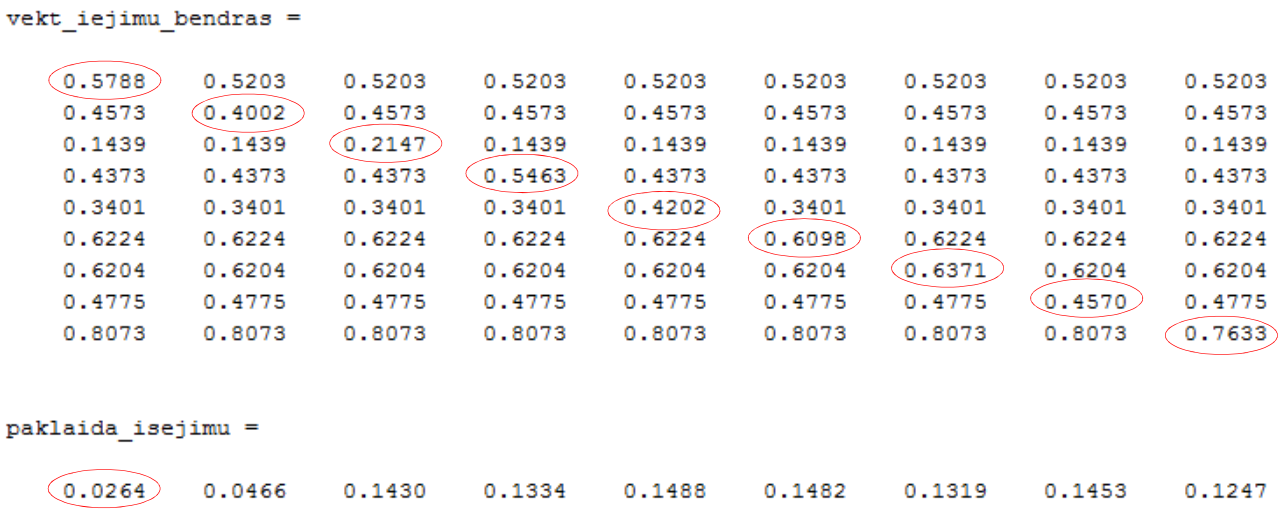
Funkcija išbandyta pateikiant nekeistus išėjimo parametrus ir modifikuotas įėjimo parametrų vertes -20% ir -10%, paeiliui kiekvienam įėjimui. Kiekvienam parametrui atliekami trys bandymai, kuriuos atitinka lentelės eilutės.



7.6 pav. Pakeisto įėjimo parametro grafikas.



7.7 pav. Paklaidos ir jos pasikartojimo grafikai, gauti pateikus sistemai pakeistus duomenis..



7.8 pav. Programos vaizdas darbo metu.

7.8 paveiksle pavaizduotu atveju pakeistas buvo pirmas įėjimo parametras. Matyti, jog algoritmui keičiant įėjimų vertes, mažiausia paklaida gauta parinkinėjant pirmąjį parametrą. Tolesnėse, dviejose lentelėse pateikiami atlikti pakartotinių bandymų rezultatai ir atvaizduojama gauta pakeisto parametro išėjimų paklaidos vieta, bendroje išėjimų paklaidų eilėje(1 reiškia teisingą parametro identifikavimą). Bandymai atlikti atsitiktinai išsirenkant funkcijos iškvietimo laiką.

7.1 lentelė. Nukrypusio parametro vieta eilėje pagal mažiausią paklaidą, kai įėjimai pakeisti 20%.

Įėjimo parametras	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1 bandymas	1	1	2	1	1	1	1	1	1
2 bandymas	1	1	1	1	2	1	1	1	1
3 bandymas	1	1	1	1	1	1	1	1	1
4 bandymas	1	1	2	1	1	1	1	1	1
5 bandymas	1	1	1	2	1	1	1	1	2

Gaunamas apytiksliai 90% tikslumas išrenkant nukrypusį parametrą. Eksperimentas pakartojamas, įvedant 10% nuokrypį, paeiliui kiekvienam parametrui.

7.2 lentelė. Nukrypusio parametro vieta eilėje pagal mažiausią paklaidą, kai įėjimai pakeisti 10%.

Parametrai	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1 bandymas	1	3	3	2	4	1	2	1	1
2 bandymas	1	3	1	1	2	1	2	1	1
3 bandymas	2	1	4	2	1	1	1	1	1
4 bandymas	4	1	2	1	2	3	1	3	1
5 bandymas	1	3	5	1	3	1	2	3	2

Šiuo atveju gaunamas tik 51% tikslumas išrenkant nukrypusį parametrą. Esant mažesniam parametro nuokrypiui, gaunama mažesnė paklaida tarp modelio ir objekto. Ją evoliucinis algoritmas kartais geba kompensuoti keisdamas kitų, nekeistų parametrų vertes. Tuo tarpu, esant dideliam nuokrypiui, kitų parametrų keitimas, sukelia išėjimų rinkinio išsiderinimą. Taigi, šios funkcijos veikimas tuo efektyvesnis, kuo labiau nuo reikiamos vertės yra nukrypęs parametras.

Visiškai tikslus išėjimų suderinimas, bei parametro aptikimo tikslumas negalimas, nes paieškai naudojamos realaus objekto išėjimo parametrų vertės, kurios turi tam tikrą paklaidą nuo modelio išėjimo verčių.

7.1.3. Įėjimo parametrų rinkinio paieška, remiantis įėjimų ir išėjimų ryšių matricomis

Tais atvejais, kai nėra aišku kiek parametrų sukėlė nuokrypį, ar pačiame procese galimai įvyko pakitimai, naudojama viso parametrų rinkinio paieška, kuri identifikuoja daugiausiai nukrypusius išėjimo parametrus ir pateikia didžiausią įtaką jiems darančius įėjimus. Šiuo atžvilgiu nagrinėjami įėjimai, tik tie, kuriuos analizavo neuroninis tinklas. Reikia nepamiršti, jog įvairius proceso pakitimus gali sukelti parametrai, kurie nėra modeliuojami ir tinklas tiesioginio ryšio su jais neturi.

Funkcija, ieškodama galimai problemą sukėlusių įėjimų, atsižvelgia į nukrypusius išėjimus, jų ryšius su įėjimo parametrais, bei kiekvieno iš jų nuokrypio dydį.

Procedūra vykdoma tokiais žingsniais:

1. Surandamos vidutinės įėjimo ir išėjimo parametrų rinkinių vertės

$$\boldsymbol{x}_{analizei} = {}^{1}/_{n} * \sum_{j=1-n}^{i} \boldsymbol{x}_{i,j}\,;\ \boldsymbol{y}_{analizei} = {}^{1}/_{n} * \sum_{j=1-n}^{i} \boldsymbol{y}_{i,j} \qquad (8)$$

2. Lyginant realų procesą su modeliu, kiekvienam išėjimo parametrui atskirai skaičiuojamos paklaidos - jos perduotos į funkciją vektoriuje „Filtruota_paklaida“. Iš jų panariui atimama neuroninio tinklo modeliavimo paklaidų, tam pačiam parametrui gauta standartinio nuokrypio vertė padauginta iš dviejų:

$$\Delta_i = e_{j,i} - 2 * \sigma_i \quad (9)$$

   Normaliomis sąlygomis, tokiu atveju gauta reikšmė Δ, didesnė už nulį turėtų būti ne daugiau kaip apytiksliai 5% visų iteracijų(paklaidų skirstinys normalusis);
3. Pagal nuokrypio Δ dydį, parametrai suskirstomi į eilę nuo didžiausio link mažiausio. Parametrai, kurie netenkina sąlygos $\Delta_i > 0$, į eilę nepatenka;
4. Programos naudotojas išsirenka kokią įėjimų-išėjimų parametrų ryšių matricą naudoti. Galimi du variantai: tiesinės koreliacijos matrica arba remiantis technologinėmis žiniomis sudaryta parametrų ryšių matrica;
5. Kiekvienam eilėje esančiam išėjimo parametrui, pagal naudojamą ryšių matricą, išrenkami labiausiai susiję įėjimo parametrai(taikoma 0,5 slenkstinė vertė koreliacijos matricai);
6. Sukuriama matrica, atvaizduojanti eilėje esančius parametrus ir su jais susijusių įėjimų ryšių pasikartojimą:

```
koef_be_koreliacijos2 =

     1     1     0     1     0     1     1     0     1
     1     1     0     1     0     0     1     0     1
     0     0     1     0     0     0     0     0     0
     1     1     0     1     1     1     1     0     1
```

7.9 pav. Matrica, atvaizduojanti susijusius įėjimo parametrus.

   Šiuo atveju nukrypę visi keturi išėjimo parametrai. Matricoje matome atvaizduotus labiausiai su jais susijusius įėjimus(pagal eilės numerį).
7. Prieš tai sukurta matrica, padauginama iš tiems ryšiams atitinkamų koreliacijos koeficientų. Gaunama matrica, kurioje matomi ne tik įėjimo parametrai, kurie labiausiai susiję su nukrypusiais išėjimais, bet ir jų koreliacijos/ryšių stiprumas:

```
koef_su_koreliacija2 =

    0.8705    0.8099         0    0.8745         0    0.9227    0.8933         0    0.9394
    0.6040    0.6164         0    0.5272         0         0    0.5188         0    0.5337
         0         0    0.5057         0         0         0         0         0         0
    0.8996    0.8382         0    0.9426    0.6187    0.9871    0.9878         0    0.9967
```

7.10 pav. Įėjimo parametrų matrica, padauginus iš koreliacijos koeficiento.

8. Visų įėjimo parametrų vertės, kurios susijusios su tam tikru išėjimo parametru, padauginamos iš jo nuokrypio dalies procentais.

9. Kiekvieno ryšį turinčio įėjimo parametro sudaugintos vertės susumuojamos(pagal visus išėjimus) ir gaunama bendra įėjimų eilė. Pagal ją sprendžiame, kokie įėjimai turi didžiausią ryšį su šiuo metu esančiu įšėjimo parametrų nuokrypiu:

```
koef_be_koreliacijos_sumuota =

   72.1153   72.1153   72.1153   72.1153   72.1153   35.6207   72.1153   35.6207   35.6207
```

7.11 pav. Bendrai gauta įėjimo parametrų įtaka esamai paklaidai.

10. Parametrai, kurie viršija vartotojo nustatytą ribą išrenkami;
11. Atliekama įėjimo parametrų paieška, keičiant įėjimo parametrus, kurie buvo išrinkti praeitame žingsnyje(kaip svarbiausius ir didžiausią įtaką turinčius).

```
Nuokrypis =

    0.0188    0.0268    0.0104    0.0132


eile =

     2     1     4     3

Parametrai pagal koreliaciją(1), technologiją(2): 2
```

7.12 pav. Išėjimo parametrų surikiavimas į eilę, pagal nuokrypį.

Siekiant nustatyti ryšį tarp parametrų skaičiuojami tiesinės koreliacijos koeficientai tarp parametrų.

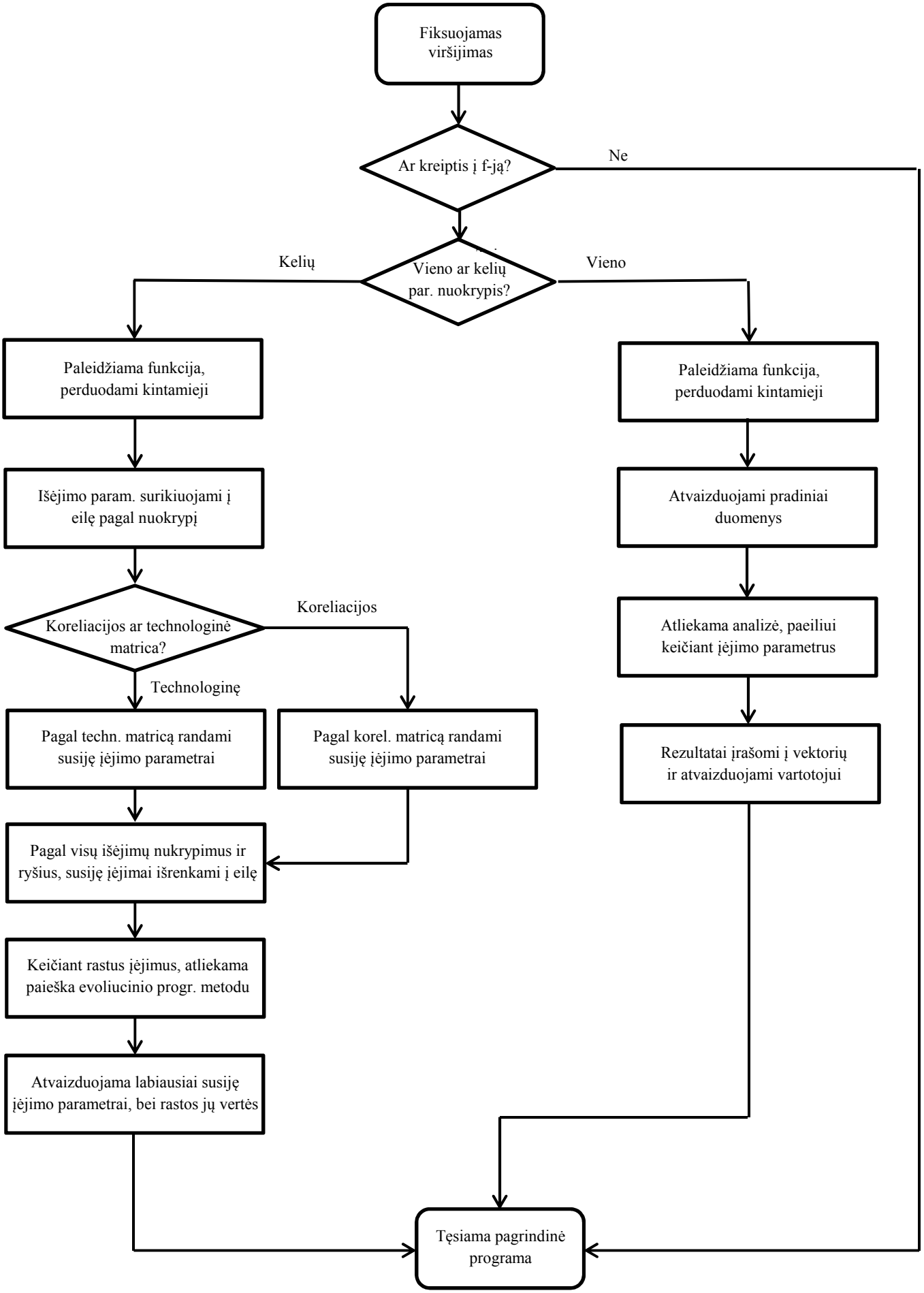
7.3 lentelė. Tiesinės koreliacijos koeficientai.

	I1	I2	I3	I4	I5	I6	I7	I8	I9
O1	-0,092	-0,012	0,505	-0,195	-0,237	-0,298	-0,182	-0,039	-0,248
O2	-0,870	-0,809	0,184	-0,874	-0,491	-0,922	-0,893	-0,356	-0,939
O3	0,603	0,616	0,001	0,527	0,272	0,429	0,518	-0,025	0,533
O4	0,899	0,838	-0,250	0,942	0,618	0,987	0,987	0,294	0,996

Taip pat, konsultuojantis su eksploatuojančiu katilinės personalu, sudaryta matrica, apibrėžianti įėjimo-išėjimo parametrų priklausomybes:

7.4 lentelė. Koeficientai atspindintys ryšius tarp parametrų.

	I1	I2	I3	I4	I5	I6	I7	I8	I9
O1	0,5	0,5	0,5	0,5	0,5	0	1	0	0
O2	0,3	0,3	0,3	0,3	1	0	0,3	0	0
O3	0	0	0	0	0	0,5	0	0,5	1
O4	0,5	0,5	0,5	0,5	0,5	0,5	0,5	1	1


Fiksuojamas viršijimas
Ar kreiptis į f-ją?
Ne
Vieno ar kelių par. nuokrypis?
Kelių
Vieno
Paleidžiama funkcija, perduodami kintamieji
Išėjimo param. surikiuojami į eilę pagal nuokrypį
Koreliacijos ar technologinė matrica?
Koreliacijos
Technologinę
Pagal techn. matricą randami susiję įėjimo parametrai
Pagal korel. matricą randami susiję įėjimo parametrai
Pagal visų išėjimų nukrypimus ir ryšius, susiję įėjimai išrenkami į eilę
Keičiant rastus įėjimus, atliekama paieška evoliucinio progr. metodu
Atvaizduojama labiausiai susiję įėjimo parametrai, bei rastos jų vertės
Paleidžiama funkcija, perduodami kintamieji
Atvaizduojami pradiniai duomenys
Atliekama analizė, paeiliui keičiant įėjimo parametrus
Rezultatai įrašomi į vektorių ir atvaizduojami vartotojui
Tęsiama pagrindinė programa

Metodo tikslumas vieno parametro nuokrypiui išbandomas, paeiliui kiekvieno įėjimo signalo reikšmei įvedant skirtingas paklaidas ir stebint ar pakeistas parametras bus išrinktas į galimai nuokrypį sukėlusių parametrų eilę. Bandymai atliekami po 3 kartus, sumažinant įėjimų vertes 10%:

7.5 lentelė. Pakeisto įėjimo parametro priklausymas išrinktų parametrų eilei ir vieta joje. Naudojama technologinė matrica.

Parametras	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1 bandymas	TAIP, 2	TAIP, 3	-	-	TAIP, 1	TAIP, 2	TAIP, 1	TAIP, 4	TAIP, 2
2 bandymas	TAIP, 2	TAIP, 3	-	-	-	TAIP, 2	TAIP, 1	TAIP, 4	TAIP, 7
3 bandymas	TAIP, 3	TAIP, 3	-	-	-	TAIP, 3	TAIP, 2	TAIP, 6	TAIP, 1

* Brūkšneliai pažymėtuose parametruose, 10% pakeitimo nepakanka analizės funkcijos iškvietimo sąlygoms įvykdyti.

7.6 lentelė. Pakeisto įėjimo parametro priklausymas išrinktų param. eilei ir vieta. Naudojama koreliacijos matrica.

Parametras	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1 bandymas	NE, 0	NE, 0	-	-	NE, 0	NE, 0	NE, 0	NE, 0	TAIP, 1
2 bandymas	TAIP, 1	TAIP, 4	-	-	-	NE, 0	NE, 0	NE, 0	TAIP, 1
3 bandymas	TAIP, 4	TAIP, 3	-	-	-	NE, 0	NE, 0	NE, 0	TAIP, 1

* Brūkšneliai pažymėtuose parametruose, 10% pakeitimo nepakanka analizės funkcijos iškvietimo sąlygoms įvykdyti.

Eksperimentas pakartojamas įvedant 20% nuokrypį.

7.7 lentelė. Pakeisto įėjimo parametro priklausymas išrinktų parametrų eilei ir vieta joje. Naudojama technologinė matrica.

Parametras	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1 bandymas	TAIP, 3	TAIP, 3	TAIP, 2	TAIP, 3	TAIP, 1	TAIP, 4	TAIP, 4	TAIP, 7	TAIP, 7
2 bandymas	TAIP, 3	TAIP, 3	-	TAIP, 3	-	TAIP, 4	TAIP, 4	TAIP, 7	TAIP, 7
3 bandymas	TAIP, 3	TAIP, 3	-	TAIP, 3	-	TAIP, 4	TAIP, 4	TAIP, 7	TAIP, 7

* Brūkšneliai pažymėtuose parametruose, 20% pakeitimo nepakanka analizės funkcijos iškvietimo sąlygoms įvykdyti.

7.8 lentelė. Pakeisto įėjimo parametro priklausymas išrinktų param. eilei ir vieta joje. Naudojama koreliacijos matrica.

Parametras	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1 bandymas	TAIP, 2	TAIP, 5	TAIP, 5	TAIP, 4	NE, 0	NE, 0	TAIP, 5	NE, 0	TAIP, 1
2 bandymas	TAIP, 2	TAIP, 5	-	NE, 0	-	NE, 0	TAIP, 5	NE, 0	TAIP, 2
3 bandymas	TAIP, 2	TAIP, 6	-	TAIP, 3	-	NE, 0	TAIP, 5	NE, 0	TAIP, 2

* Brūkšneliai pažymėtuose parametruose, 20% pakeitimo nepakanka analizės funkcijos iškvietimo sąlygoms įvykdyti.

Rezultatai leidžia spręsti, kad naudojant technologiniais ryšiais pagrįstą ryšių matricą galima gana teisingai išskirti susijusius parametrus. Naudojant technologinę matricą, reikiamas įėjimas priklauso išrinktų parametrų eilei 100% tirtų atvejų Deja, metodas nėra tikslus nustatant, kuris būtent parametras sukėlė nuokrypį.

Prastesni rezultatai gauti naudojant koreliacijos matricą ryšiams tarp įėjimo ir išėjimo parametrų fiksuoti. Esant 10% procentų paklaidai, rasti įėjimai priklausė išrinktų parametrų eilei tik 37% tirtų atvejų. Esant 20% paklaidai, įėjimai priklauso eilei 65% atvejų.

Atlikti eksperimentai, įvedant 20% nuokrypį kelių parametrų rinkiniams. Kadangi 1-5 parametrai savo poveikiu procesui panašūs, nagrinėtos kombinacijos tik tarp 1, 6, 7. 8, 9 parametrų. Naudota technologinė ryšių matrica. Gauti rezultatai:

7.9 lentelė. Pakeistų įėjimo parametrų priklausymas išrinktų parametrų eilei.

Parametrai	**1 bandymas**	**2 bandymas**	**3 bandymas**
1, 6	TAIP, ABU	TAIP, ABU	TAIP, ABU
1, 7	VIENAS	VIENAS	VIENAS
1, 8	TAIP, ABU	TAIP, ABU	TAIP, ABU
1, 9	VIENAS	VIENAS	VIENAS
6, 7	TAIP, ABU	TAIP, ABU	TAIP, ABU
6, 8	TAIP, ABU	TAIP, ABU	TAIP, ABU
6, 9	TAIP, ABU	TAIP, ABU	TAIP, ABU
7, 8	VIENAS	VIENAS	VIENAS
7, 9	VIENAS	VIENAS	VIENAS
8, 9	NEPRIKLAUSO	NEPRIKLAUSO	NEPRIKLAUSO

Bandymas pakartotas parametrams taikant 10% nuokrypį:

Parametrai	**1 bandymas**	**2 bandymas**	**3 bandymas**
1, 6	VIENAS	VIENAS	TAIP, ABU
1, 7	TAIP, ABU	TAIP, ABU	TAIP, ABU
1, 8	TAIP, ABU	TAIP, ABU	VIENAS
1, 9	VIENAS	VIENAS	VIENAS
6, 7	TAIP, ABU	TAIP, ABU	VIENAS
6, 8	VIENAS	VIENAS	VIENAS
6, 9	TAIP, ABU	TAIP, ABU	TAIP, ABU
7, 8	VIENAS	VIENAS	VIENAS
7, 9	VIENAS	VIENAS	VIENAS
8, 9	NEPRIKLAUSO	NEPRIKLAUSO	NEPRIKLAUSO

Esant 20% nuokrypiui, abu nukrypę parametrai, susijusių įėjimų eilei priskirti 50% atvejų, o esant 10% nuokrypiui – 36% atvejų. Naudojantis funkcija pastebėta, kad esant dviejų parametrų nuokrypai dažnai, kaip galintys būti nuokrypio priežastimi, identifikuojami visi įėjimai. Galima teigti, jog funkcija tik nedideliu tikslumu gali identifikuoti keletą nukrypusių parametrų.

Išvados ir rezultatai

1. Atliekant literatūros analizę apžvelgtas biokuro panaudojimo aktualumas, bei standartinės dūmais aušinamos judančio ardyno pakuros funkcionavimas, pateiktos pagrindinių įrenginių funkcijos.
2. Sudarytas esminių technologinių parametrų rinkinys. Surinkti ir apdoroti kelių mėnesių eksploatacijos istoriniai duomenys.
3. Sukurtas dirbtiniu neuroniniu tinklu paremtas pakuros modelis, kurio vidutinė kvadratinė paklaida siekia apie 0,12%, o standartinis nuokrypis 0,89%.
4. Realizuota stebėjimo sistema, kuri perspėja apie modelio ir objekto parametrų nesutapimą. Viršytas pusvalandis fiksuojamas esant didesnei nei $3*\sigma_{paklaidos}$ nuokrypai tarp objekto ir modelio išėjimo parametrų. Įspėjimas išduodamas tik praėjus nustatytam 100 viršytų pusvalandų periodui, o avarija signalizuojama po 200 pusvalandžių.
5. Realizuotos analizės funkcijos, kurios leidžia naudotojui nustatyti galimai nuokrypį sukėlusius įėjimo parametrus. Atliktų bandymų metu nustatyta, jog ieškant vieno parametro, esant daugiau nei 20% nuokrypiui, jis aptinkamas 90% atvejų. Ieškant parametrų rinkinio ir naudojant technologinę matricą, esant daugiau nei 20% vieno parametro nuokrypiui, susijusių įėjimų eilei jis priskiriamas 100% atvejų.

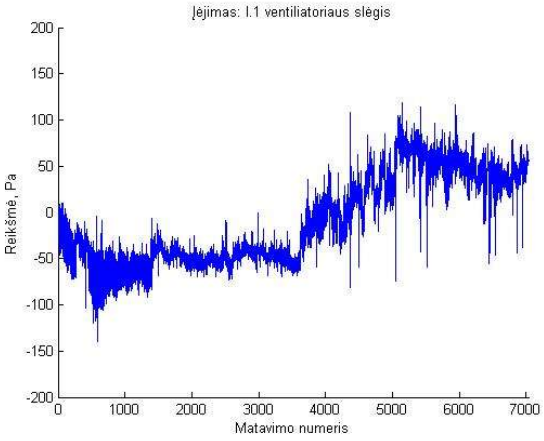
Informacijos šaltinių sąrašas

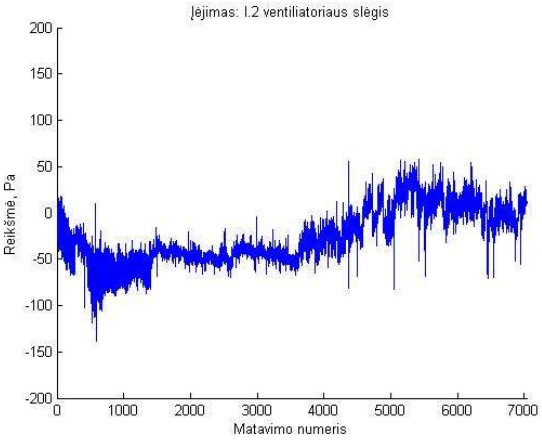
1. Loeser M. „Overview of biomass conversion and generation technologies“. Padova, 2008. Prieiga per internetą: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=4651566&pageNumber%3D3%26queryText%3Dbiomass . Žiūrėta 2014.06.01.
2. Maker, Timothy M. „Wood-chip heating systems“. Prieiga per internetą: http://www.biomasscenter.org/pdfs/Wood-Chip-Heating-Guide.pdf. Žiūrėta 2014.06.02.
3. Quaak Peter, Knoef Harrie, Stassen Hubert. „Energy from biomass: a review of combustion and gasification technologies“. Prieiga per internetą: http://books.google.lt/books?id=M2WMrePlIxkC&lpg=PA41&ots=mVKp1mqq6J&dq=biomass%20burning%20furnace&pg=PP1#v=onepage&q&f=false. Žiūrėta 2014.05.27.
4. Brunch Christian, Peters Bernhard, Naussbaumer Thomas. „Modelling wood combustion under fixed bed conditions“. Prieiga per internetą: http://www.verenum.ch/Publikationen/Bruch_FUEL_82(2003).pdf. Žiūrėta 2014.06.08.
5. Bauer R., Golles M., Brunner T., Dourdumas N.. „Modelling of grate combustion in a medium scale biomass furnace for control purposes“. Prieiga per internetą: http://www.sciencedirect.com/science/article/pii/S096195340900244X . Žiūrėta 2014.06.03.
6. Žecova Monika, Terpak Jan, Dorwak Lubomir. „Mathematical model of gasification and combustion of biomass“. Prieiga per internetą: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6228753. Žiūrėta 2014.06.07.
7. Šarkauskas Kastytis. „Kompiuterinės mašinų valdymo sistemos“. Vilniaus pedagoginio universiteto leidykla, 2008, 186psl..
8. Balaševičius Leonas, Dervinis Gintaras, Mačerauskas Vidmantas. „Supervizorinio valdymo ir duomenų surinkimo sistema InTouch“. Technologija, 2005, 16psl..
9. „InTouch HMI Data Managemnet Guide“, Invensys Systems, Inc. Prieiga per internetą: http://platforma.astor.com.pl/files/getfile/id/3885 . Žiūrėta: 2015.02.06
10. Dieck Ronald H.. „Measurement Uncertainty– Methods and Applications“. ISA, 2006, 169psl..
11. Felinger A.. „Data Analysis and Signal Processing in Chromatography“. Elsevier, 1998, 159psl..
12. Kriesel David. „A brief introduction to neural networks“. Prieiga per internetą: http://www.dkriesel.com/_media/science/neuronalenetze-en-zeta2-1col-dkrieselcom.pdf . Žiūrėta: 2014.10.16.

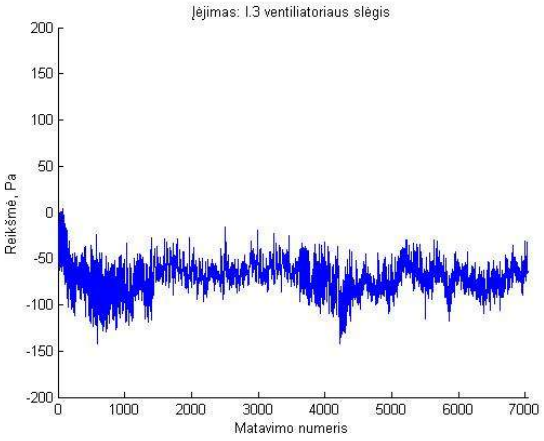
13. Haykin Simon O.. „Neural networks and learning machines“. Prentice hall, 2008. 165psl.
14. Beale M. Hudson, Hagan Martin T., Demuth Howard B.. „Neural network toolbox user‘s guide“. Prieiga per internetą: https://www.mathworks.com/help/pdf_doc/nnet/nnet_ug.pdf . Žiūrėta: 2014.11.03.
15. Simutis Rimvydas. „Sistemų modeliavimas ir identifikavimas“. Vilniaus pedagoginio universiteto leidykla, 2008, 140psl..
16. Yang Shuang , Browne Anthony. „Neural network ensembles: combining multiple models for enchanced performance using multistage approach“. Department of Computing, University of Surrey, 2004. Prieiga per internetą: http://onlinelibrary.wiley.com/doi/10.1111/j.1468-0394.2004.00285.x/pdf . Žiūrėta 2015.02.03.
17. Droke Cliff. „Moving averages simplified“. Marketplace books, 2001.
18. Sarker Ruhul, Mohammadian Masoud, Yao Xin. „Evolutionary Optimization“. Kluwer Academic Publishers, 2002.

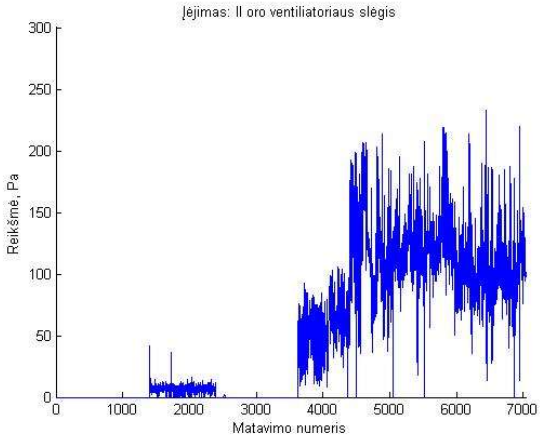
Priedai

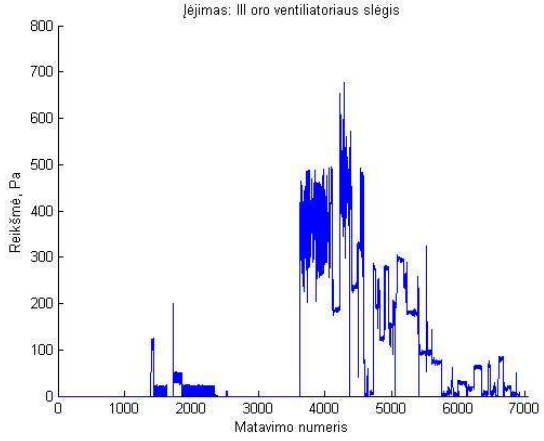
Priedas 1. Pradinių duomenų grafikai

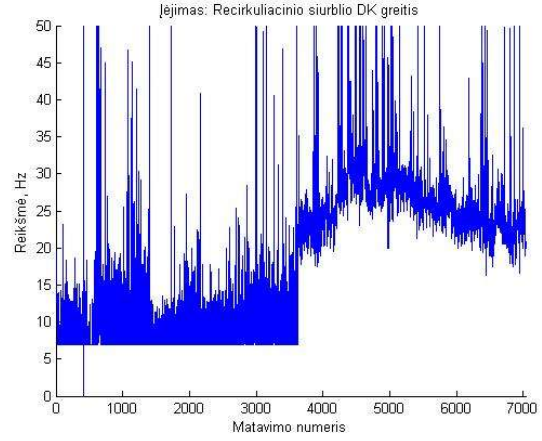


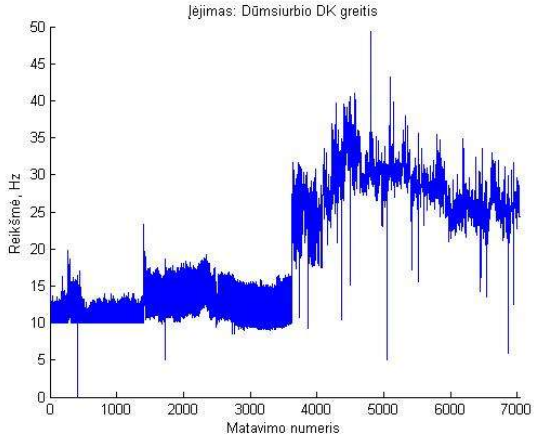


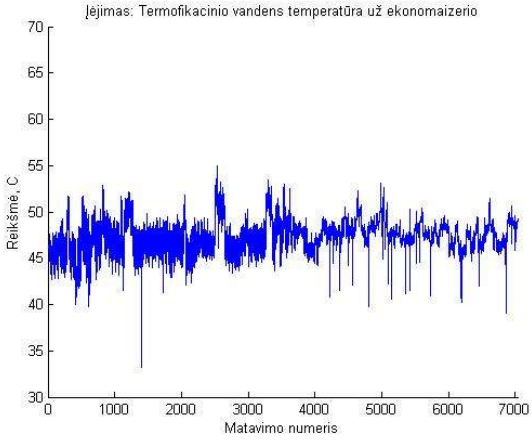


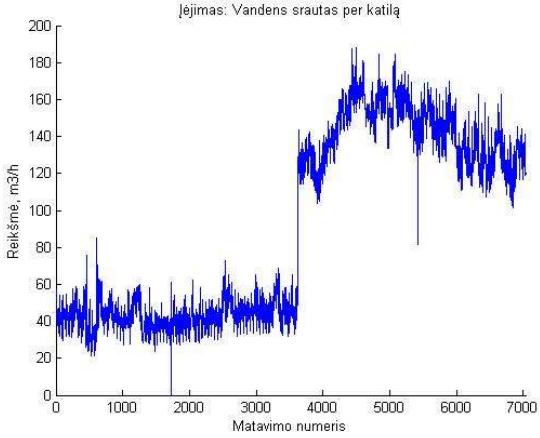


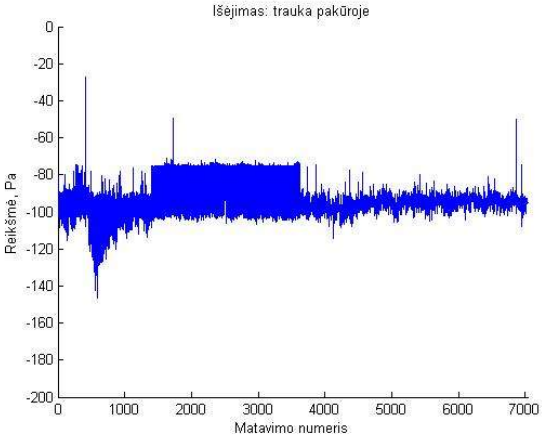


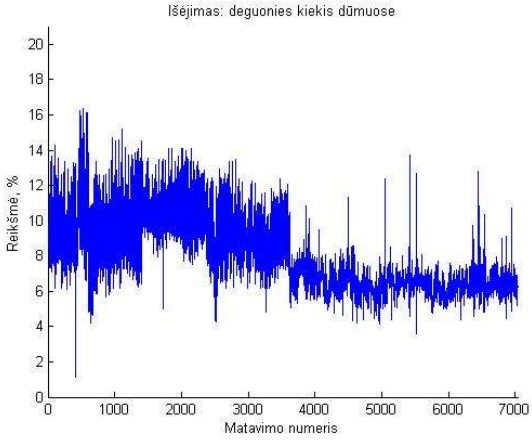


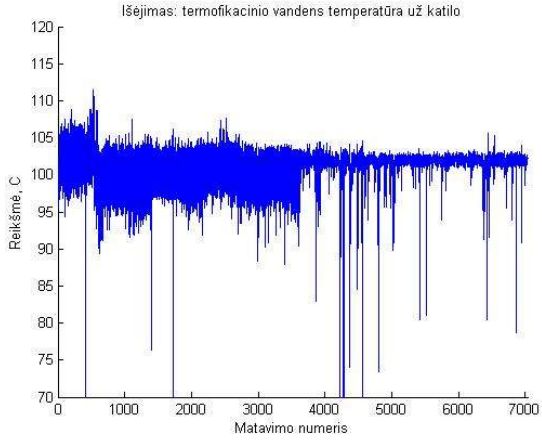

Įėjimas: Dūmsiurbio DK greitis
Reikšmė, Hz
Matavimo numeris

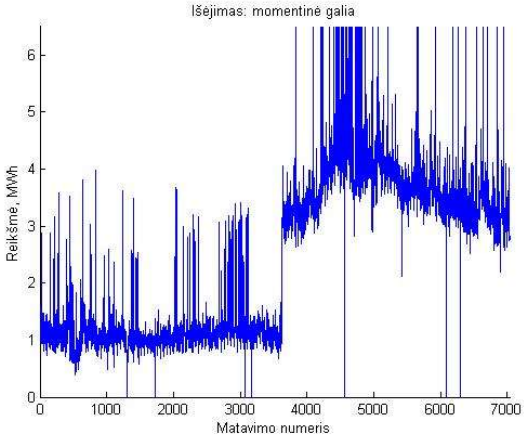

Įėjimas: Termofikacinio vandens temperatūra už ekonomaizerio
Reikšmė, C
Matavimo numeris


Įėjimas: Vandens srautas per katilą
Reikšmė, m3/h
Matavimo numeris

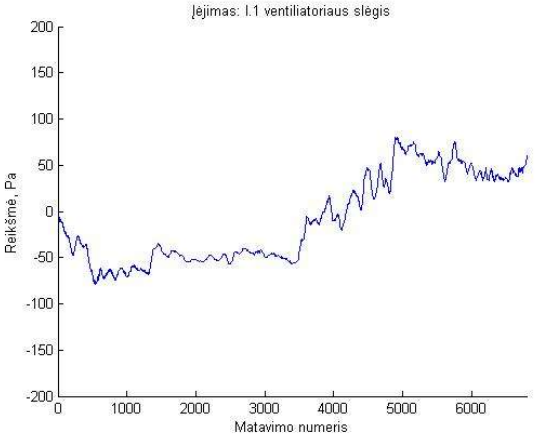

Išėjimas: trauka pakūroje
Reikšmė, Pa
Matavimo numeris

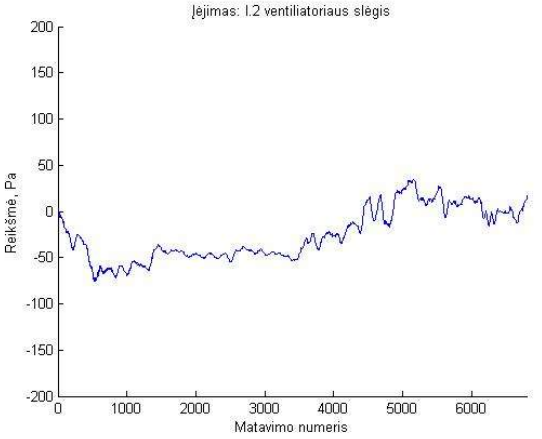

Išėjimas: deguonies kiekis dūmuose
Reikšmė, %
Matavimo numeris

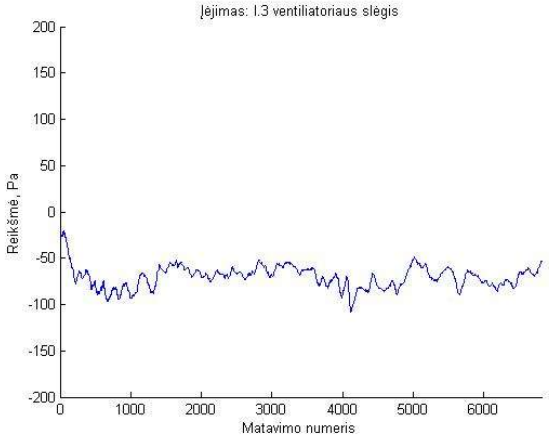


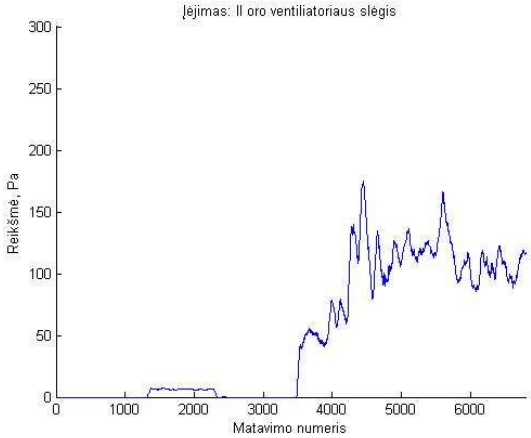


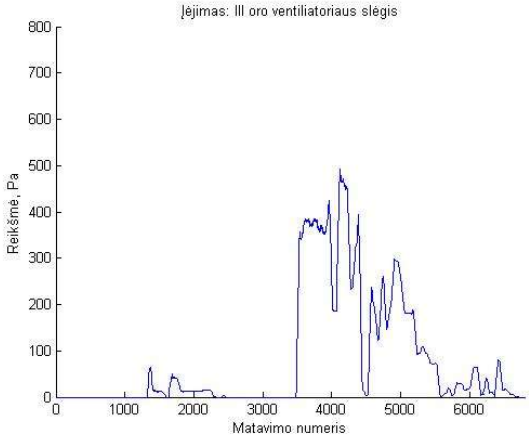
Priedas 2. Apdorotų duomenų grafikai

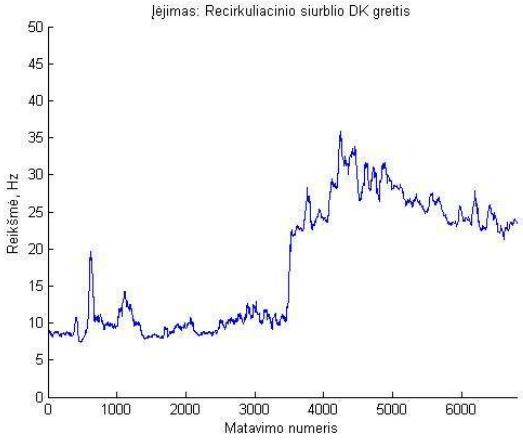


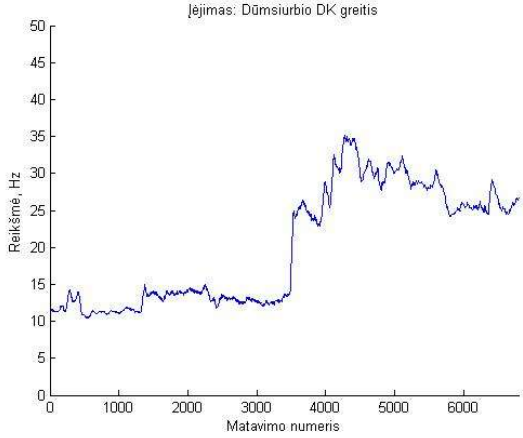


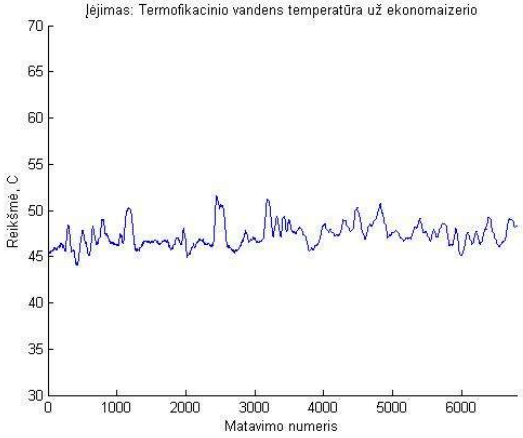


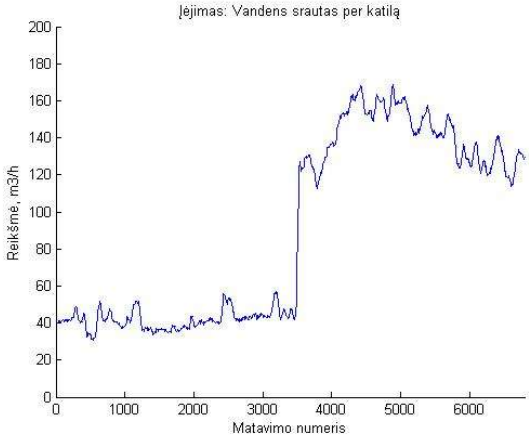


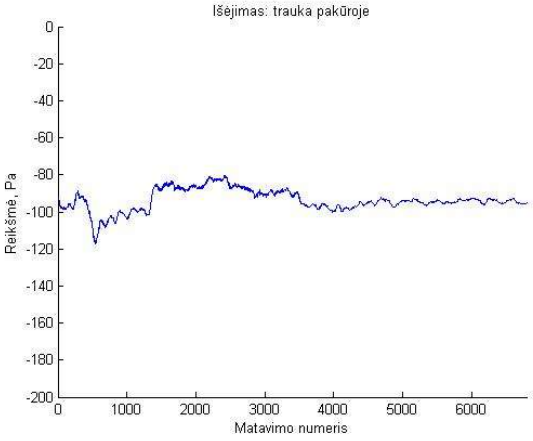

Įėjimas: III oro ventiliatoriaus slėgis
Reikšmė, Pa
Matavimo numeris

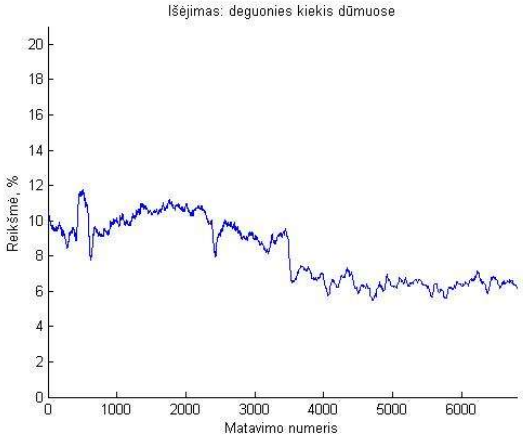

Įėjimas: Recirkuliacinio siurblio DK greitis
Reikšmė, Hz
Matavimo numeris

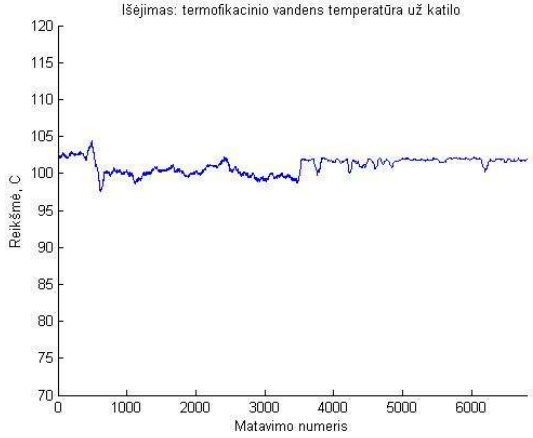

Įėjimas: Dūmsiurbio DK greitis
Reikšmė, Hz
Matavimo numeris

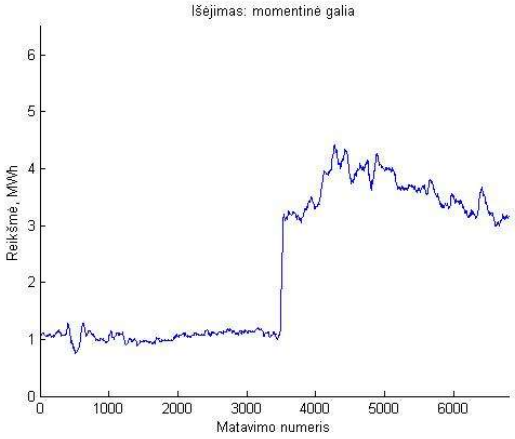

Įėjimas: Termofikacinio vandens temperatūra už ekonomaizerio
Reikšmė, C
Matavimo numeris


Įėjimas: Vandens srautas per katilą
Reikšmė, m3/h
Matavimo numeris

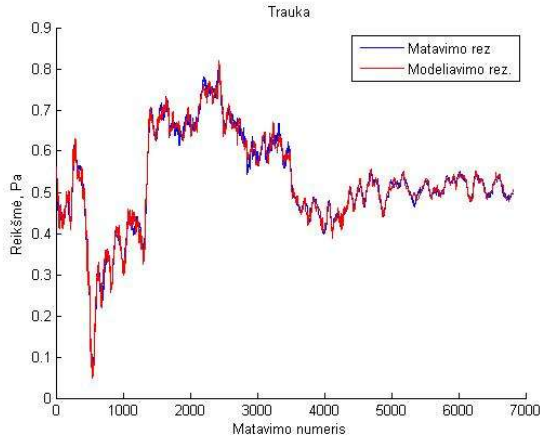

Išėjimas: trauka pakūroje
Reikšmė, Pa
Matavimo numeris

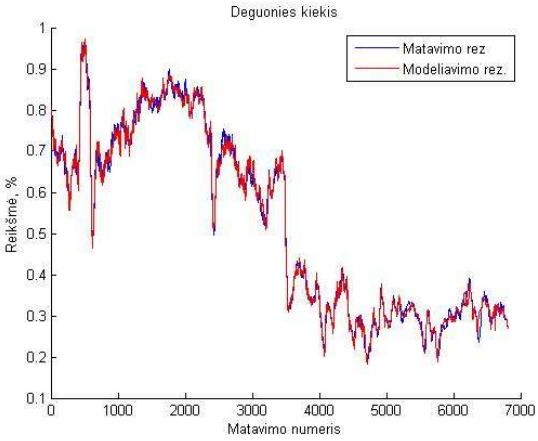

Išėjimas: deguonies kiekis dūmuose
Reikšmė, %
Matavimo numeris

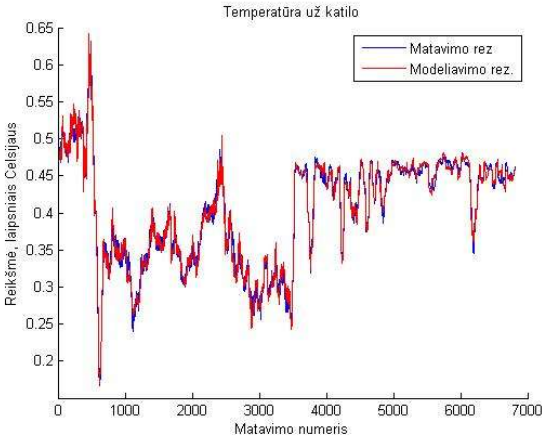

Išėjimas: termofikacinio vandens temperatūra už katilo
Reikšmė, C
Matavimo numeris

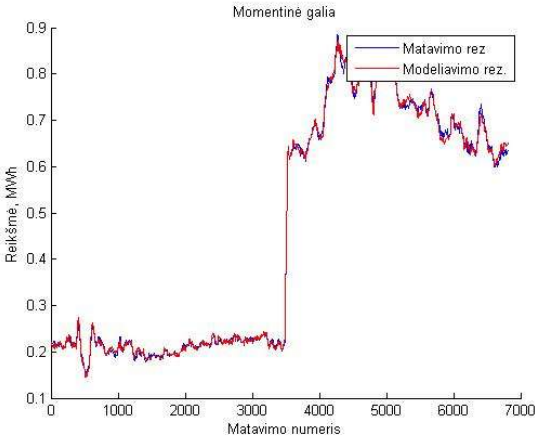

Išėjimas: momentinė galia
Reikšmė, MWh
Matavimo numeris

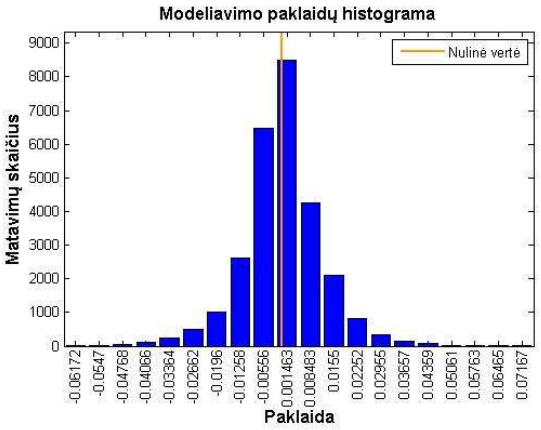
Priedas 3. Modelio ir realaus objekto išėjimų palyginimas



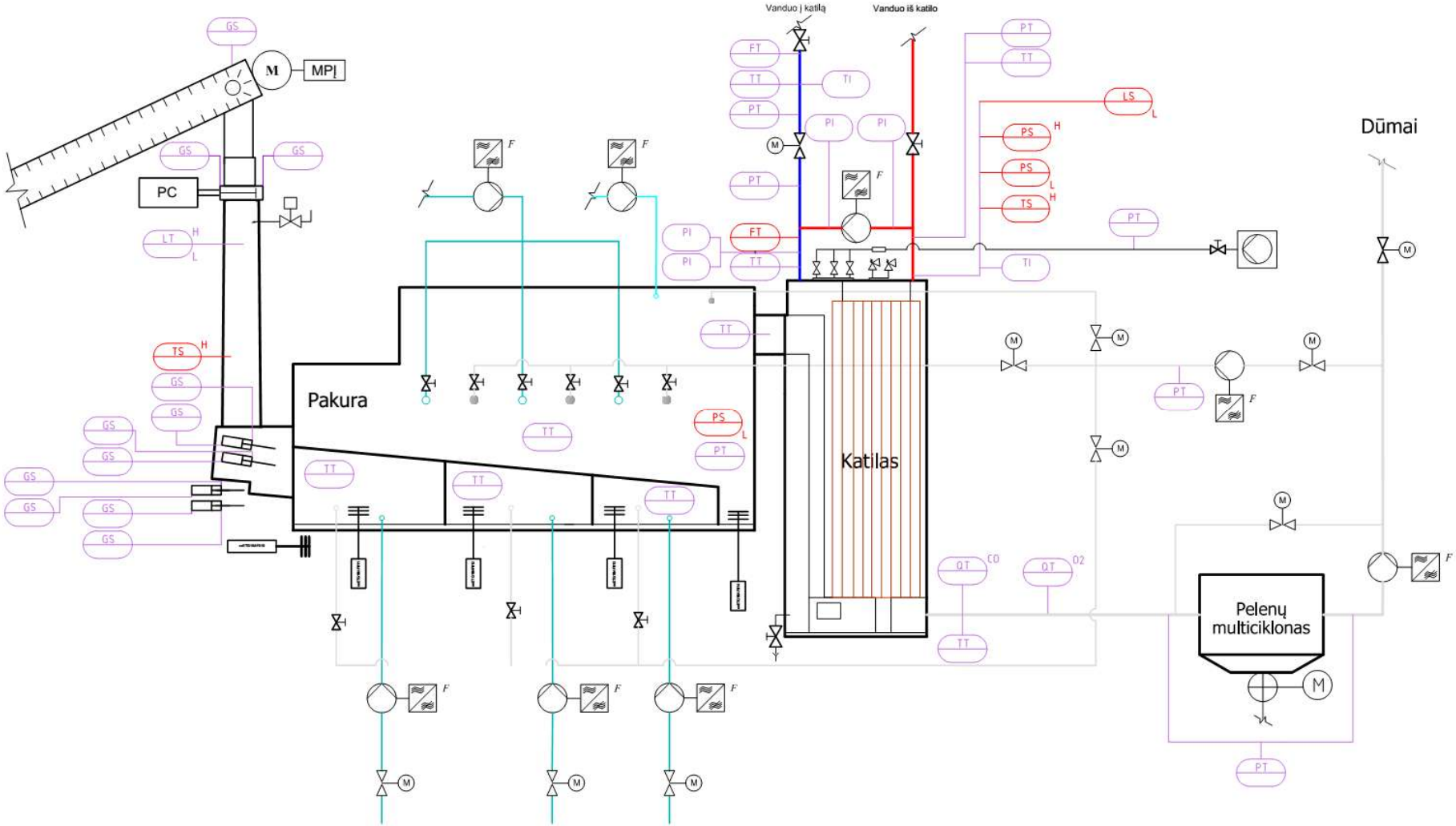








Priedas 4. Pakuros technologinė schema



Priedas 5. MATLAB programų kodai

Pagrindinės programos kodas

```
% Neuroniniu tinklu paremtas pakuros modelis
%
%

%% TINKLŲ APMOKYMAS


prompt = 'Ar išvalyti programos terpę? Y/N [Y]: ';
str = input(prompt,'s');
if isempty(str)
    str = 'Y';
end

if str == 'Y'
% Išvaloma programos terpė
clc, clear all, close all
end

failas = 'nauji.mat';
paslepti_sluoksniai = 25;

pries = cputime;
[ net, net2, net3, y, e, e1, e2, e3, sigma, Iejimai, Rezultatai ] =
apmokymas(paslepti_sluoksniai, failas);

po = cputime - pries



%% GRAFIKŲ IŠVEDIMAS Į EKRANĄ
grafikai( Rezultatai, y, e )


%% KORELIACIJOS KOEFICIENTŲ SKAIČIAVIMAS

clear i
clear j

for i = 1:size(Rezultatai,1)

    for j = 1:size(Iejimai,1)
        kor = corrcoef(Iejimai(j,:),Rezultatai(i,:));
        koreliacija(i,j) = kor(1,2);

    end
end



%% PAKEISTŲ DUOMENŲ SUKŪRIMAS

% Perrašome turimus duomenis, eksperimentų atlikimui

realtime_In = Iejimai;
realtime_Out = Rezultatai;

Model_rec_In = realtime_In;   % REALIOS SISTEMOS ĮĖJIMAI
Model_rec_Out = realtime_Out; % REALIOS SISTEMOS IŠĖJIMAI


i = 0;
ilgis = length(Model_rec_In)
```

```
%{
% Įėjimo parametro keitimas palaipsniui

for i = 1:1:ilgis
    if i < ilgis*0.5
    Model_rec_In(9,i) = Model_rec_In(9,i)*0.1*(i/ilgis*0.5) + Model_rec_In(9,i) ;
    else
        Model_rec_In(9,i) = Model_rec_In(9,i)*1.1;
    end

    if Model_rec_In(9,i) > 1
        Model_rec_In(9,i) = 1;
    end

end
%}

%{
% Išėjimo parametro keitimas palaipsniui

for i = 1:1:ilgis
    if i < ilgis*0.5
    Model_rec_Out(1,i) = Model_rec_Out(1,i)*0.2*(i/ilgis*0.5) + Model_rec_Out(1,i) ;
    else
        Model_rec_Out(1,i) = Model_rec_Out(1,i)*1.2;
    end

    if Model_rec_Out(1,i) > 1
        Model_rec_Out(1,i) = 1;
    end

end
%}

Model_rec_In(2,:) = Model_rec_In(2,:)*0.9; % Pakeičiamas įėjimas

% Atvaizdavimo dalis

figure(8)
hold on;
title('Sugeneruotas ir pradinis įėjimo parametro kitimas')
xlabel('Matavimo numeris')
plot(realtime_In(2,:).', 'r')
plot(Model_rec_In(2,:).', 'b')
legend('Pakeistos vertės','Pradinės vertės')
hold off;


% figure(9)
% hold on;
% title('Sugeneruotas ir pradinis išėjimo parametro kitimas')
% xlabel('Matavimo numeris')
% plot(Model_rec_Out(1,:).', 'b')
% plot(realtime_Out(1,:).', 'r')
% hold off;


%% STEBĖJIMO SISTEMA


ilgis = size(realtime_In, 2);

i = 0;

Filtruota_paklaida(1) = 0;

for i = 1:1:ilgis

%     SU GERAIS DUOMENIMIS, apskaičiuojami tinklo išėjimai
%     Isejimas_model_1(:,i) = net(realtime_In(:,i));
```

```
%     Isejimas_model_2(:,i) = net2(realtime_In(:,i));
%     Isejimas_model_3(:,i) = net3(realtime_In(:,i));

%     SU PAKEISTAIS DUOMENIMIS, apskaičiuojami tinklo išėjimai
    Isejimas_model_1(:,i) = net(Model_rec_In(:,i));
    Isejimas_model_2(:,i) = net2(Model_rec_In(:,i));
    Isejimas_model_3(:,i) = net3(Model_rec_In(:,i));


    Isejimas_model_all = cat(3, Isejimas_model_1(:,i), Isejimas_model_2(:,i),
Isejimas_model_3(:,i));
    Isejimas_model(:,i) = mean(Isejimas_model_all, 3);
    Isejimas_model(:,i) = Isejimas_model_1(:,i);

end


%%

i = 0;
j = 0;

virsyti_pusv = zeros(size(realtime_Out,1), ilgis); % Nustatomas kintamojo dydis
tevu_skaicius = 10; % Nurodoma kiek tėvų turės evoliucinis algoritmas

% Apskaičiuojami neuroninio tinklo modeliavimo paklaidų
% standartiniai nuokrypiai, bei vertės naudojamos parametrų viršijimo
% fiksavimui

stand_nuokrypis = std(e.');
verte_ispejimui = (stand_nuokrypis * 3).';


for i = 1:1:ilgis



    Error(:,i) = abs(gsubtract(Isejimas_model(:,i),Model_rec_Out(:,i)));
    % Iš modelio išėjimų atimami realūs išėjimai

    for j = 1:size(realtime_Out,1)

    if i > 100  % Paklaidos filtravimas

    Filtruota_paklaida(j,i) = sum(Error(j,i-99:i))/100;
    else
        Filtruota_paklaida(i) = sum(sum(Error(1:3,:).'))/i;
    end
    end


    for j = 1:size(realtime_Out,1)

    if Filtruota_paklaida(j,i) > verte_ispejimui(j)

virsyti_pusv(j,i+1) = virsyti_pusv(j,i) + 1;

    else
        if virsyti_pusv(j,i) > 0
        virsyti_pusv(j,i+1) = virsyti_pusv(j,i) - 1;
        else
        virsyti_pusv(j,i+1) = 0;
        end
    end

    end


    for j = 1:size(realtime_Out,1)

    clear str
```

```
    if virsyti_pusv(j, i+1) == 100 && virsyti_pusv(j, i) > virsyti_pusv(j, i-1) %
kokiam viršytų iteracijų skaičiui esant bus įspėjimas
    z = msgbox({'ĮSPĖJIMAS!' 'Didelis parametrų nuokrypis!', 'Parametras:' num2str(j)
});
    end

    if virsyti_pusv(j, i+1) == 500 && virsyti_pusv(j, i) > virsyti_pusv(j, i-1)

    z = msgbox({'AVARIJA!' 'Labai didelis parametrų nuokrypis!', 'Parametras:'
num2str(j)  });

    prompt = 'Ar kreiptis į analizės funkciją? Y/N [Y]: ';
    str = input(prompt,'s');

        if isempty(str)
        str = 'Y';
        end

        if str == 'Y'
            %%
  virsyti_pusv(j, i+1) = 0;

[ kokie_rasti_iejimai, programos_iejimai ] = analize(Model_rec_Out(:,i-5:i),
Model_rec_In(:,i-5:i), net, tevu_skaicius, koreliacija, Filtruota_paklaida(:,i-5:i),
stand_nuokrypis);

% Analizės funkcija pavieniams parametrams
% [ kokie_rasti_iejimai, programos_iejimai ] = analize_pavieniui(Model_rec_Out(:,i-
5:i), Model_rec_In(:,i-5:i), net, tevu_skaicius, koreliacija, Filtruota_paklaida(:,i-
5:i), stand_nuokrypis);

        end

    end

    end

end
%% IŠSKIRIAMI DAUGIAUSIAI NUOKRYPĘ PARAMETRAI
% Lyginama su evoliucinio algoritmo pateiktais įėjimais

 for i = 1:size(kokie_rasti_iejimai,2)
palyg(:,i) = abs(kokie_rasti_iejimai(:,i) - mean(programos_iejimai,2))
 end

 dydis = abs(sum(palyg,1))

 vieta = find(dydis == min(dydis));

 palyg(:,vieta)
 vieta2 = find(palyg(:,vieta) > 0.01)

%% Grafikai

figure(10)
plot(Isejimas_model.')
title('Modeliuotos sistemos išėjimai')
xlabel('Matavimo numeris')


figure(11)
plot(realtime_Out.')
title('Realios sistemos išėjimai')
xlabel('Matavimo numeris')

%%
figure(13)
plot(Filtruota_paklaida(1,301:end).'*100,'r')
refline(0,verte_ispejimui(1,:)*100)
title('Filtruota traukos parametro paklaida % ir 3sigma ribinė vertė')
xlabel('Matavimo numeris')
```

```
ylabel('%')

figure(14)
plot(Filtruota_paklaida(2,301:end).'*100,'r')
refline(0,verte_ispejimui(2,:)*100)
title('Filtruota deguonies kiekio paklaida % ir 3sigma ribinė vertė')
xlabel('Matavimo numeris')
ylabel('%')

figure(15)
plot(Filtruota_paklaida(3,301:end).'*100,'r')
refline(0,verte_ispejimui(3,:)*100)
title('Filtruota vandens T už katilo paklaida % ir 3sigma ribinė vertė')
xlabel('Matavimo numeris')
ylabel('%')

figure(16)
plot(Filtruota_paklaida(4,301:end).'*100,'r')
refline(0,verte_ispejimui(4,:)*100)
title('Filtruota momentinės galios paklaida % ir 3sigma ribinė vertė')
xlabel('Matavimo numeris')
ylabel('%')
%%

for i = 1:size(virsyti_pusv,1)

figure(16+i)
plot(virsyti_pusv(i,301:end))
ispejimo_linija = refline(0,100)
avarijos_linija = refline(0,200)
title({'Iš eilės pasikartojančių iteracijų, esant paklaidai skaičius' 'Įspėjimo ir
avarijos ribos'})

xlabel('Matavimo numeris')


end
```

Duomenų filtravimo funkcijos kodas

```
function [ Iejimai, Rezultatai ] = duomenu_filtravimas( Iejimai, Rezultatai )
%   Pirminių duomenų filtravimas

% Filtruojamos momentinės galios vertės, viršijančios galimas.
nuorodos = find(Rezultatai(4,:) >= 6.5);

ilgis = size(nuorodos,2);
for i = ilgis:-1:1


    Iejimai(:,nuorodos(:,i))=[];
    Rezultatai(:,nuorodos(:,i))=[];


end
clear nuorodos;

% Filtruojamos momentinės galios vertės, mažesnės už 0.
nuorodos = find(Rezultatai(4,:) <0);

ilgis = size(nuorodos,2);
for i = ilgis:-1:1


    Iejimai(:,nuorodos(:,i))=[];
    Rezultatai(:,nuorodos(:,i))=[];


end
```

```
clear nuorodos;

% Filtruojamos termofikacinio vandens T vertės mažesnės už 90C.
% Technologiškai tokiu atveju vyksta pereinamasis procesas arba katilas
% nedirba.
nuorodos = find(Rezultatai(3,:) <= 90);

ilgis = size(nuorodos,2);
for i = ilgis:-1:1


    Iejimai(:,nuorodos(:,i))=[];
    Rezultatai(:,nuorodos(:,i))=[];

end
clear nuorodos;
clear ilgis;


% Tomsono tau metodo taikymas neteisingų duomenų taškų suradimui

% Kintamieji ir jų filtravimo intervalai


% Į vektorių įvedami parametrai, kuriuos norima filtruoti
% Paeiliui, pagal įvestus parametrus įvedami filtravimo diapazonai, kiekvienam iš jų
parametru_vekt = [ 7; 7; 7; 7; 9; 13; 13; 13; 13; 13; 13]; % Į vektorių įvedami
parametrai, kuriuos norima filtruoti
intervalu_vekt = [ 6300 6900; 4300 5200; 3650 4000; 1 3590; 5300 5700; 2600 3100 ;
1900 2400; 1200 1500; 580 1100; 380 540; 1 400] ;



iter_skaicius = size(parametru_vekt, 1);

tau = 3.2;


for j = 1:iter_skaicius

    if parametru_vekt(j,:) < 10

data_In = Iejimai(:,intervalu_vekt(j,1):intervalu_vekt(j,2));
data_Out = Rezultatai(:,intervalu_vekt(j,1):intervalu_vekt(j,2));
data = Iejimai(parametru_vekt(j,:),intervalu_vekt(j,1):intervalu_vekt(j,2));

stand_nuokrypis = std(data);
testavimui = stand_nuokrypis * 5;
kiek_ismesta = 0;

    else

        data_In = Iejimai(:,intervalu_vekt(j,1):intervalu_vekt(j,2));
data_Out = Rezultatai(:,intervalu_vekt(j,1):intervalu_vekt(j,2));
data = Rezultatai(parametru_vekt(j,:)-9,intervalu_vekt(j,1):intervalu_vekt(j,2));

stand_nuokrypis = std(data);
testavimui = stand_nuokrypis * 5;
kiek_ismesta = 0;
    end



while testavimui > tau*stand_nuokrypis

vidurkis = mean(data);
stand_nuokrypis = std(data);

ilgis = size(data,2);
```

```
clear mod_nuokrypio

for i = 1:ilgis
    mod_nuokrypio(1,i) = abs(vidurkis - data(1,i));
end


testavimui = max(mod_nuokrypio);
stebejimui = testavimui - tau*stand_nuokrypis;
if testavimui > tau*stand_nuokrypis

    nuorodos = find(mod_nuokrypio == testavimui);

    if size(nuorodos,2) > 1
        for k = 1:size(nuorodos,2)
            Iejimai(:,nuorodos(1,k)) = [];
    Rezultatai(:,nuorodos(1,k)) = [];
    kiek_ismesta = kiek_ismesta + 1;
        end
    else


    data(:,nuorodos) = [];
    data_In(:,nuorodos) = [];
    data_Out(:,nuorodos) = [];
    kiek_ismesta = kiek_ismesta + 1;
        end
end

end

verciu_skaic = size(data,2);

Iejimai(:,intervalu_vekt(j,1):intervalu_vekt(j,2)) = zeros;
Rezultatai(:,intervalu_vekt(j,1):intervalu_vekt(j,2)) = zeros;

Iejimai(:,intervalu_vekt(j,1):(intervalu_vekt(j,1)+verciu_skaic-1)) =
data_In(:,1:verciu_skaic);
Rezultatai(:,intervalu_vekt(j,1):(intervalu_vekt(j,1)+verciu_skaic-1)) =
data_Out(:,1:verciu_skaic);

for g = 1:kiek_ismesta
Iejimai(:,(intervalu_vekt(j,1)+verciu_skaic)) = [];
Rezultatai(:,(intervalu_vekt(j,1)+verciu_skaic)) = [];
end

clear data
clear data_In
clear data_Out
clear verciu_skaic

end

Iejimai =  sgolayfilt(Iejimai.',1,51).';
Rezultatai =  sgolayfilt(Rezultatai.',1,51).';

end
```

Neuronių tinklų apmokymo funkcijos kodas

```
function [ tinklas1, y, e, e1, sigma, Iejimai, Rezultatai ] = apmokymas(
paslepti_sluoksniai, failas )

%  Dirbtinio neuroninio tinklo apmokymo funkcija pakuros modeliavimui
```

```
% Užkraunami eksploatavimo duomenys
load(failas)

prompt = 'Ar atvaizduoti duomenis? Y/N [Y]: ';
str = input(prompt,'s');
if isempty(str)
    str = 'Y';
end

if str == 'Y'
    pradiniu_duomenu_atvaizdavimas(Iejimai, Rezultatai);
end


prompt = 'Ar vykdyti duomenų filtravimą? Y/N [Y]: ';
str = input(prompt,'s');
if isempty(str)
    str = 'Y';
end

if str == 'Y'
    [ Iejimai, Rezultatai] = duomenu_filtravimas(Iejimai, Rezultatai);
end


% Duomenų normavimas į 0...1 ribas, pagal darbines

Iejimai(1,:) = (Iejimai(1,:)+100)/200;     % Slėgis -100...100
Iejimai(2,:) = (Iejimai(2,:)+100)/200;     % Slėgis -100...100
Iejimai(3,:) = (Iejimai(3,:)+100)/100;     % Slėgis -100...0
Iejimai(4,:) = (Iejimai(4,:))/200;         % Slėgis 0...200
Iejimai(5,:) = Iejimai(5,:)/600;           % Slėgis 0...600
Iejimai(6,:) = Iejimai(6,:)/50;             % DK greitis 0...50Hz
Iejimai(7,:) = Iejimai(7,:)/50;             % DK greitis 0...50Hz
Iejimai(8,:) = (Iejimai(8,:)-35)/25;            % Temperatūra 35...60C
Iejimai(9,:) = (Iejimai(9,:)-20)/160;            % Srautas 20...180m3/h


Rezultatai(1,:) = (Rezultatai(1,:)+120)/50;        % Trauka -120...-70Pa
Rezultatai(2,:) = (Rezultatai(2,:)-4)/8;               % Deguonis 4...12
Rezultatai(3,:) = (Rezultatai(3,:)-95)/15;              % Temperatūra 95...110C
Rezultatai(4,:) = Rezultatai(4,:)/5;       % Galia 0...6MWh



% Duomenų normavimas į 0...1 ribas, pagal daviklių ribas
%
% Iejimai(1,:) = (Iejimai(1,:)+500)/1000;     % Slėgis -500...500
% Iejimai(2,:) = (Iejimai(2,:)+500)/1000;     % Slėgis -500...500
% Iejimai(3,:) = (Iejimai(3,:)+500)/1000;     % Slėgis -500...500
% Iejimai(4,:) = (Iejimai(4,:))/1000;         % Slėgis 0...1000
% Iejimai(5,:) = Iejimai(5,:)/2000;           % Slėgis 0...2000
% Iejimai(6,:) = Iejimai(6,:)/50;             % DK greitis 0...50Hz
% Iejimai(7,:) = Iejimai(7,:)/50;             % DK greitis 0...50Hz
% Iejimai(8,:) = Iejimai(8,:)/120;            % Temperatūra 0...120C
% Iejimai(9,:) = Iejimai(9,:)/200;            % Srautas 0...200m3/h
%
%
% Rezultatai(1,:) = (Rezultatai(1,:)+250)/500;        % Trauka 250...-250Pa
% Rezultatai(2,:) = Rezultatai(2,:)/21;               % Deguonis 0...21
% Rezultatai(3,:) = Rezultatai(3,:)/150;              % Temperatūra 0...150C
% Rezultatai(4,:) = (Rezultatai(4,:)+1.087)/13;       % Galia 0...6MWh


%% Su vienu tinklu
%{

prompt = 'Ar vydyti tinklų apmokymą? Y/N [Y]: ';
str = input(prompt,'s');
if isempty(str)
    str = 'Y';
```

```
end

if str == 'Y'


% Sukuriami tinklai
tinklas1 = feedforwardnet(paslepti_sluoksniai);
tinklas1 = configure(tinklas1, Iejimai, Rezultatai);


% Iėjimų/išėjimų normavimas ir apdorojimas
tinklas1.input.processFcns = {'removeconstantrows','mapminmax'};
tinklas1.output.processFcns = {'removeconstantrows','mapminmax'};



% Duomenys padalijami į apmokymo, validavimo ir testavimo imtis
tinklas1.divideFcn = 'dividerand';  % Atsitiktinio padalijimo funkcija
tinklas1.divideMode = 'sample';
tinklas1.divideParam.trainRatio = 70/100;
tinklas1.divideParam.valRatio = 15/100;
tinklas1.divideParam.testRatio = 15/100;


% Nustatoma Levenbergo-Markardo algoritmo funkcija
tinklas1.trainFcn = 'trainlm'



% Tinklo darbo kokybės kriterijus
tinklas1.performFcn = 'mse';  % Vidutinė kvadratinė paklaida


% Apmokymas
[tinklas1,tr1] = train(tinklas1,Iejimai,Rezultatai);

end
%% Tinklo testavimas


% PIRMAS TINKLAS
y = tinklas1(Iejimai);
e1 = gsubtract(Rezultatai,y); %e1 = gsubtract(Rezultatai(1:3,:),y(1:3,:));
e11 = sum((e1.^2).')/size(e1,2);
e111 = sum(e11,2)/size(e11,2)
e1_sigma = sqrt(e111);
performance1 = perform(tinklas1,Rezultatai,y)

% paklaida_pradine1 = mse(e1);

% Paklaidų sumos skaičiavimas
% s = abs(e);
% suma = sum(s(:))




performance = performance1;
e = e1;
sigma = e1_sigma

%}



%% SU TRIJŲ TINKLŲ KOLEKTYVU

prompt = 'Ar vydyti tinklų apmokymą? Y/N [Y]: ';
str = input(prompt,'s');
if isempty(str)
    str = 'Y';
```

```
end

if str == 'Y'


% Sukuriami tinklai
tinklas1 = feedforwardnet(paslepti_sluoksniai);
tinklas1 = configure(tinklas1, Iejimai, Rezultatai);
tinklas2 = feedforwardnet(paslepti_sluoksniai);
tinklas2 = configure(tinklas2, Iejimai, Rezultatai);
tinklas3 = feedforwardnet(paslepti_sluoksniai);
tinklas3 = configure(tinklas3, Iejimai, Rezultatai);

% Iėjimų/išėjimų normavimas ir apdorojimas
tinklas1.input.processFcns = {'removeconstantrows','mapminmax'};
tinklas1.output.processFcns = {'removeconstantrows','mapminmax'};
tinklas2.input.processFcns = {'removeconstantrows','mapminmax'};
tinklas2.output.processFcns = {'removeconstantrows','mapminmax'};
tinklas3.input.processFcns = {'removeconstantrows','mapminmax'};
tinklas3.output.processFcns = {'removeconstantrows','mapminmax'};


% Duomenys padalijami į apmokymo, validavimo ir testavimo imtis
tinklas1.divideFcn = 'dividerand';  % Atsitiktinio padalijimo funkcija
tinklas1.divideMode = 'sample';
tinklas1.divideParam.trainRatio = 70/100;
tinklas1.divideParam.valRatio = 15/100;
tinklas1.divideParam.testRatio = 15/100;
tinklas2.divideFcn = 'dividerand';  % Atsitiktinio padalijimo funkcija
tinklas2.divideMode = 'sample';
tinklas2.divideParam.trainRatio = 70/100;
tinklas2.divideParam.valRatio = 15/100;
tinklas2.divideParam.testRatio = 15/100;
tinklas3.divideFcn = 'dividerand';  % Atsitiktinio padalijimo funkcija
tinklas3.divideMode = 'sample';
tinklas3.divideParam.trainRatio = 70/100;
tinklas3.divideParam.valRatio = 15/100;
tinklas3.divideParam.testRatio = 15/100;

% Nustatoma Levenbergo-Markardo algoritmo funkcija
tinklas1.trainFcn = 'trainlm'
tinklas2.trainFcn = 'trainlm'
tinklas3.trainFcn = 'trainlm'


% Tinklo darbo kokybės kriterijus
tinklas1.performFcn = 'mse';  % Vidutinė kvadratinė paklaida
tinklas2.performFcn = 'mse';
tinklas3.performFcn = 'mse';

% Apmokymas
[tinklas1,tr1] = train(tinklas1,Iejimai,Rezultatai);
[tinklas2,tr2] = train(tinklas2,Iejimai,Rezultatai);
[tinklas3,tr3] = train(tinklas3,Iejimai,Rezultatai);

end
%% Tinklo testavimas


% PIRMAS TINKLAS
y = tinklas1(Iejimai);
e1 = gsubtract(Rezultatai,y);
e11 = sum((e1.^2).')/size(e1,2);
e111 = sum(e11,2)/size(e11,2)
e1_sigma = sqrt(e111);
performance1 = perform(tinklas1,Rezultatai,y)

% paklaida_pradine1 = mse(e1);

% Paklaidų sumos skaičiavimas
% s = abs(e);
```

```
% suma = sum(s(:))


% ANTRAS TINKLAS
y2 = tinklas2(Iejimai);
e2 = gsubtract(Rezultatai,y2);
e22 = sum((e2.^2).')/size(e2,2);
e222 = sum(e22,2)/size(e22,2)
e2_sigma = sqrt(e222);
performance2 = perform(tinklas2,Rezultatai,y2)

% paklaida_pradine2 = mse(e2);

% Paklaidų sumos skaičiavimas
% s = abs(e);
% suma = sum(s(:))



% TREČIAS TINKLAS
y3 = tinklas3(Iejimai);
e3 = gsubtract(Rezultatai,y3);
e33 = sum((e3.^2).')/size(e3,2);
e333 = sum(e33,2)/size(e33,2)
e3_sigma = sqrt(e333);
performance3 = perform(tinklas3,Rezultatai,y3)

% paklaida_pradine3 = mse(e3);

% Paklaidų sumos skaičiavimas
% s = abs(e);
% suma = sum(s(:))


e = mean(cat(3, e1, e2, e3), 3);
performance = (performance1 + performance2 + performance3)/3
sigma = (e1_sigma + e2_sigma + e3_sigma)/3
% e = e1;
% sigma = e1_sigma
%}


end
```

Pavienio nukrypusio parametro paieškos funkcijos kodas

```
function [ vekt_iejimu_bendras, Gauti_iejimai ] = analize2( Isejimas_model, Iejimai,
net, tevu_skaicius, j, koreliacija, Filtruota_paklaida, standartinis_nuokrypis)

% Evoliuciniu programavimu paremta paieskos funkcija vienam kintamąjam
% Duomenys turi būti pateikti 0...1 ribose.
close all

% Sukuriami reikiami kintamieji
g = 1;
minimumas = ones(1,g);
minimumo_vekt = zeros(size(Iejimai,1),g);
In_param_nr = zeros(size(koreliacija));
pakartojimai = 0;


% Išvedami įvairūs parametrai stebėjimui
Gauti_isejimai = Isejimas_model
Isejimo_verte_paieskai = mean(Isejimas_model,2)
```

```
Gauti_iejimai = Iejimai
Iejimu_vidurkis = mean(Iejimai,2)

Isejimo_verte_paieskai = mean(Isejimas_model,2)
Tinklo_isejimas_su_gautais_iejimais = net(Iejimu_vidurkis)



for s = 1:size(Iejimai,1)

In_param_nr = s

% pause

for l = 1:g

pakartojimai = 0;

while minimumas(:,1) > 0.02 && pakartojimai < 3
    clear tevai; % Išvalomi kintamieji, naujam ciklui
    clear atsakymai;
    clear palikuonys;
    clear test;
    clear test2;
    clear atsakymai_2;
    clear rez;
    clear bendras;
    clear rez2;
    clear B;
    clear didziausias;
    clear nuorodos;


    ciklas = 1;
    pakartojimai = pakartojimai + 1;

for i = 1:500 % Iteracijų skaičius

if ciklas == 1
% Sukuriami "tėvų" vektoriai, paieškai

for o = 1:tevu_skaicius % Užpildomas tėvų vektorius pagal įėjimus
    tevai(:,o) = mean(Iejimai,2);
end

for p = 1:size(In_param_nr,2)
    % Atsitiktinai parenkama keičiamo parametro vertė
tevai(In_param_nr(p),:) = rand(size(Iejimai(p,:), 1), tevu_skaicius);

end

tevai(:,1) = mean(Iejimai,2); % Į tėvų vektorių įvedamas gautų įėjimų vektorius

% Tikrinama ar tėvų vektoriaus reikšmės papuola į 0...1 intervalą
test_tev = find(tevai < 0);
test2_tev = find(tevai > 1);
test_tev = sum(test_tev);
test2_tev = sum(test2_tev);

j = 0;
while test_tev > 0 || test2_tev > 0
    for p = 1:size(In_param_nr,2)

tevai(In_param_nr(p),:) = rand(size(Iejimai(p,:), 1), tevu_skaicius);

end
    tevai(:,1) = mean(Iejimai,2);
    test_tev = find(tevai < 0);
test2_tev = find(tevai > 1);
test_tev = sum(test_tev);
test2_tev = sum(test2_tev);
```

```
j = j + 1;
if j > 100
    break;
end

end

atsakymai = net(tevai);
ciklas = ciklas + 1;

else

atsakymai = net(tevai);

%% Sukuriami palikuonys

palikuonys = tevai;


for p = 1:size(In_param_nr,2)
% Keičiamas reikiamas parametras
palikuonys(In_param_nr(p),:) = tevai(In_param_nr(p),:) + 0.01*rand(size(tevai(p,:),
1), tevu_skaicius);

end

% Tikrinama ar palikuonių vektoriaus nariai patenka į 0...1 ribas
test = find(palikuonys < 0);
test2 = find(palikuonys > 1);
test = sum(test);
test2 = sum(test2);

j = 0;
while test > 0 || test2 > 0
    for p = 1:size(In_param_nr,2)

palikuonys(In_param_nr(p),:) = tevai(In_param_nr(p),:) + 0.01*rand(size(tevai(p,:),
1), tevu_skaicius);

    end

    test = find(palikuonys < 0);
test2 = find(palikuonys > 1);
test = sum(test);
test2 = sum(test2);
j = j + 1;
if j > 100
    break;
end

end


atsakymai_2 = net(palikuonys);


% Sudedame paklaidų vektorius ir tėvų/palikuonių vektorius
for j = 1:tevu_skaicius

    rez(:,j) = Isejimo_verte_paieskai - atsakymai(:,j);
bendras(:,j) = tevai(:,j);

end


for j = (tevu_skaicius+1):(2*tevu_skaicius)

    rez(:,j) = Isejimo_verte_paieskai - atsakymai_2(:,j-tevu_skaicius);
    bendras(:,j) = palikuonys(:,j-tevu_skaicius);

end
```

```
bendras;
rez;

% Randame dalyvius turinčius mažiausais paklaidas

if size(rez, 1) > 1
rez = abs(rez);
rez2 = sum(rez);

B = sort(rez2);
didziausias = B(:,tevu_skaicius);
nuorodos = find(rez2 <= didziausias);

nuoroda_min = find(rez2 == min(B));


else

rez = abs(rez);

B = sort(rez);
didziausias = B(:,tevu_skaicius);
nuorodos = find(rez <= didziausias);


nuoroda_min = find(rez == min(B));

end


if B(1,1) < minimumas(l)
   minimumas(:,l) = B(1,1)
   minimumo_vekt(:,l) = bendras(:,nuoroda_min)
end

for k = 1:tevu_skaicius
    tevai(:,k) = bendras(:,nuorodos(:,k));
end
tevai;

ciklas = ciklas + 1;

end
end


end

end

% Atvaizduojamos įėjimų vertės, bei gautos paklaidos

isejimo_vertes_paieskai = mean(Isejimas_model,2)
isejimo_vertes_su_generuotais_iejimais = net(minimumo_vekt)

vekt_iejimu_bendras(:,s) = minimumo_vekt
paklaida_isejimu(:,s) = minimumas
clear minimumo_vekt
minimumas = ones(1,g);


end


end
```

Įėjimo parametrų rinkinio paieškos funkcijos kodas

```
function [ minimumo_vekt, Gauti_iejimai ] = analize( Isejimas_model, Iejimai, net,
tevu_skaicius, koreliacija, Filtruota_paklaida, standartinis_nuokrypis)

% Evoliuciniu programavimu paremta paieskos funkcija parametrų rinkiniui
% Duomenys turi būti pateikti 0...1 ribose.
close all

% Sukuriami reikiami kintamieji
g = 3;
minimumas = ones(1,g);
minimumo_vekt = zeros(size(Iejimai,1),g);
In_param_nr = zeros(size(koreliacija));
pakartojimai = 0;

 % Užkraunama matrica parodanti technologinius ryšius tarp parametrų
load('matrix.mat')


% Išvedami įvairūs parametrai stebėjimui
Gauti_isejimai = Isejimas_model
Isejimo_verte_paieskai = mean(Isejimas_model,2)

Gauti_iejimai = Iejimai
Iejimu_vidurkis = mean(Iejimai,2)

Isejimo_verte_paieskai = mean(Isejimas_model,2)
Tinklo_isejimas_su_gautais_iejimais = net(Iejimu_vidurkis)


abs_koreliacija = abs(koreliacija);
standartinis_nuokrypis = standartinis_nuokrypis.'


% Peržiūrima, kurie išėjimai, be pagrindinio turi didžiausias paklaidas

Vid_filtruotas_nuokrypis = mean(Filtruota_paklaida,2)

% Paskaičiuojama kiek bendrai nukrypę išėjimo parametrai nuo standartinio nuokrypio.

for i = 1:size(Vid_filtruotas_nuokrypis,1)
   Nuokrypis(i) = Vid_filtruotas_nuokrypis(i) - 2*standartinis_nuokrypis(i);
end

Nuokrypis

% Parametrai surikiuojami į eilė pagal savo nuokrypas

Tarpinis = sort(Nuokrypis, 'descend');
for i = 1:size(Tarpinis,2)

    if Tarpinis(i) > 0
    eile(i) = find(Nuokrypis == Tarpinis(i));
    end

end
eile

% Naudotojas išsirenka pagal kokią matricą vertinti parametrus
prompt = 'Parametrai pagal koreliaciją(1), technologiją(2): ';
    str = input(prompt,'s');

if isempty(str)
        str = '1';
        end

% Surandamas bendras nuokrypis ir kiekvieno IŠĖJIMO parametro svoris
% procentais
```

```
Nuokrypis_procentai = zeros(size(eile));

Sum_nuokrypis = sum(Nuokrypis(eile));
for i = 1:size(eile,2)

    Nuokrypis_procentai(i) = (Nuokrypis(eile(i))*100)/Sum_nuokrypis;

end

Nuokrypis_procentai


% Apibrėžimas parametrų masyvo dydis
mas_param_nr = zeros(size(eile,2),size(koreliacija,2));
mas_abs_kor_isrinkta = zeros(size(eile,2),size(koreliacija,2));

% Surandami ĮĖJIMO parametrai, turintys didžiausią koreliaciją
for i = 1:size(eile,2)

    if str == '1'
        % Taikant koreliacijos matricą
    In_param_nr = find(abs_koreliacija(eile(i),:) > 0.5)
    else
        % Taikant technologinių ryšių matricą
    In_param_nr = find(matrix(eile(i),:) > 0.1)

    end


if sum(In_param_nr) == 0
    In_param_nr = 1:size(Iejimai,1);
end

   for j = 1:size(In_param_nr,2)
       mas_param_nr(i,j) = In_param_nr(1,j);
   end

if str == '1'
abs_kor_isrinkta = abs_koreliacija(eile(i), In_param_nr);
else
abs_kor_isrinkta = matrix(eile(i), In_param_nr);
end

for j = 1:size(abs_kor_isrinkta,2)
       mas_abs_kor_isrinkta(i,j) = abs_kor_isrinkta(1,j);
   end

end


% Sudėliojamos parametrų matricos, su kuriomis bus skaičiuojamas galutinis
% parametrų svarbumas
for i = 1:size(Iejimai,1)

    for j = 1:size(mas_param_nr,1)

        if find(mas_param_nr(j,:) == i) > 0
koef_be_koreliacijos2(j,i) = 1;

        end

    end

end

koef_su_koreliacija2 = koef_be_koreliacijos2;

for j = 1:size(mas_abs_kor_isrinkta,1)
    vieta = find(koef_be_koreliacijos2(j,:) == 1);
for i = 1:size(vieta,2)
```

```
   koef_su_koreliacija2(j,vieta(i)) = koef_be_koreliacijos2(j,vieta(i)) *
mas_abs_kor_isrinkta(j,i);

end
end

    koef_su_koreliacija2
    koef_be_koreliacijos2

% Apskaičiuojama parametrų nuokrypio tikimybė, pagal nuokrypio dalį procentais ir
ryšių
% matricas
for i = 1:size(koef_be_koreliacijos2,1)

koef_be_koreliacijos(i,:) = koef_be_koreliacijos2(i,:) * Nuokrypis_procentai(i);
koef_su_koreliacija(i,:) = koef_su_koreliacija2(i,:) * Nuokrypis_procentai(i);

end

koef_be_koreliacijos = sum(koef_be_koreliacijos,1);
koef_su_koreliacija = sum(koef_su_koreliacija,1);


koef_be_koreliacijos
koef_su_koreliacija

% Nustatoma riba virš kurios parametrai laikomi nukrypusias ir išrenkami
% naudojimui paieškos algoritme

In_param_nr = find(koef_be_koreliacijos > 35);
if sum(In_param_nr) == 0
    In_param_nr = 1:size(Iejimai,1);
end

In_param_nr


pause


% Evoliucinio programavimo algoritmas
for l = 1:g

pakartojimai = 0;

while minimumas(:,l) > 0.02 && pakartojimai < 3
    clear tevai; % Išvalomi kintamieji, naujam ciklui
    clear atsakymai;
    clear palikuonys;
    clear test;
    clear test2;
    clear atsakymai_2;
    clear rez;
    clear bendras;
    clear rez2;
    clear B;
    clear didziausias;
    clear nuorodos;


    ciklas = 1;
    pakartojimai = pakartojimai +1;

for i = 1:500

if ciklas == 1

% Sukuriami "tėvų" vektoriai, paieškai

for o = 1:tevu_skaicius % Užpildomas tėvų vektorius pagal įėjimus
```

```
    tevai(:,o) = mean(Iejimai,2);
end

for p = 1:size(In_param_nr,2)
    % Atsitiktinai parenkama keičiamo parametro vertė
tevai(In_param_nr(p),:) = rand(size(Iejimai(p,:), 1), tevu_skaicius);

end

tevai(:,1) = mean(Iejimai,2); % Į tėvų vektorių įvedamas gautų įėjimų vektorius

% Tikrinama ar tėvų vektoriaus reikšmės papuola į 0...1 intervalą
test_tev = find(tevai < 0);
test2_tev = find(tevai > 1);
test_tev = sum(test_tev);
test2_tev = sum(test2_tev);

j = 0;
while test_tev > 0 || test2_tev > 0
    for p = 1:size(In_param_nr,2)

tevai(In_param_nr(p),:) = rand(size(Iejimai(p,:), 1), tevu_skaicius);

end
    tevai(:,1) = mean(Iejimai,2);
    test_tev = find(tevai < 0);
test2_tev = find(tevai > 1);
test_tev = sum(test_tev);
test2_tev = sum(test2_tev);
j = j + 1;
if j > 100
    break;
end

end

atsakymai = net(tevai);
ciklas = ciklas + 1;

else



atsakymai = net(tevai);

%% Sukuriami palikuonys


palikuonys = tevai;


for p = 1:size(In_param_nr,2)
% Keičiamas reikiamas parametras
palikuonys(In_param_nr(p),:) = tevai(In_param_nr(p),:) + 0.01*rand(size(tevai(p,:),
1), tevu_skaicius);

end

test = find(palikuonys < 0);
test2 = find(palikuonys > 1);
test = sum(test);
test2 = sum(test2);

j = 0;
while test > 0 || test2 > 0
    for p = 1:size(In_param_nr,2)
% Tikrinama ar palikuonių vektoriaus nariai patenka į 0...1 ribas
palikuonys(In_param_nr(p),:) = tevai(In_param_nr(p),:) + 0.01*rand(size(tevai(p,:),
1), tevu_skaicius);

    end
```

```
    test = find(palikuonys < 0);
test2 = find(palikuonys > 1);
test = sum(test);
test2 = sum(test2);
j = j + 1;
if j > 100
    break;
end

end


atsakymai_2 = net(palikuonys);



% Sudedame paklaidų vektorius ir tėvų/palikuonių vektorius
for j = 1:tevu_skaicius

    rez(:,j) = Isejimo_verte_paieskai - atsakymai(:,j);
bendras(:,j) = tevai(:,j);

end


for j = (tevu_skaicius+1):(2*tevu_skaicius)

    rez(:,j) = Isejimo_verte_paieskai - atsakymai_2(:,j-tevu_skaicius);
    bendras(:,j) = palikuonys(:,j-tevu_skaicius);

end
bendras;
rez;

% % Randame 5 dalyvius turinčius mažiausais paklaidas

if size(rez, 1) > 1
rez = abs(rez);
rez2 = sum(rez);

B = sort(rez2);
didziausias = B(:,tevu_skaicius);
nuorodos = find(rez2 <= didziausias);

nuoroda_min = find(rez2 == min(B));


else

rez = abs(rez);

B = sort(rez);
didziausias = B(:,tevu_skaicius);
nuorodos = find(rez <= didziausias);


nuoroda_min = find(rez == min(B));

end


if B(1,1) < minimumas(l)
   minimumas(:,l) = B(1,1)
   minimumo_vekt(:,l) = bendras(:,nuoroda_min)
end

for k = 1:tevu_skaicius
    tevai(:,k) = bendras(:,nuorodos(:,k));
end
tevai;
```

```
ciklas = ciklas + 1;


end
end


end

end

% Gautų įėjimų verčių ir paklaidų atvaizdavimas

minimumas
minimumo_vekt

isejimo_vertes_paieskai = mean(Isejimas_model,2)
isejimo_vertes_su_generuotais_iejimais = net(minimumo_vekt)


end
```