Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Pastato elektros energijos suvartojimo adaptyvaus modelio sukūrimas ir tyrimas

Baigiamasis magistro projektas

Stanislav Ignatavičius

Projekto autorius

doc. dr. Gintaras Dervinis

Vadovas

Kaunas, 2019

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Pastato elektros energijos suvartojimo adaptyvaus modelio sukūrimas ir tyrimas

Baigiamasis magistro projektas

Valdymo technologijos (6211EX014)

Stanislav Ignatavičius

Projekto autorius

doc. dr. Gintaras Dervinis

Vadovas

doc. dr. Leonas Balaševičius

Recenzentas

Kaunas, 2019

Kauno technologijos universitetas

Elektros ir elektronikos fakultetas

Stanislav Ignatavičius

Pastato elektros energijos suvartojimo adaptyvaus modelio sukūrimas ir tyrimas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Stanislavo Ignatavičiaus, baigiamasis projektas tema „Pastato elektros energijos suvartojimo adaptyvaus modelio sukūrimas ir tyrimas“ yra parašytas visiškai savarankiškai ir 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)

Ignatavičius, Stanislav. Pastato elektros energijos suvartojimo adaptyvaus modelio sukūrimas ir tyrimas. Magistro baigiamasis projektas / vadovas doc. dr. Gintaras Dervinis; Kauno technologijos universitetas, Elektros ir elektronikos fakultetas.

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

Reikšminiai žodžiai: suvartojama elektros energja, statistiniai prognozavimo modeliai, autoregresiniai prognozavimo modeliai, eksponentinio išlyginimo prognozavimo modelis, dirbtinis neuroninis tinklas, neraiškios logikos reguliatorius, adaptyvus prognozavimo modelis.

Kaunas, 2019. 92 p., priedų 8 p.

Santrauka

Baigiamojo magistro tema – „Pastato elektros energijos suvartojimo adaptyvaus modelio sukūrimas ir tyrimas“. Šio darbo literatūros analizės dalyje yra apžvelgiami statistiniai ir dirbtinio intelekto prognozavimo modeliai ir jų tinkamumas mažai žinomo individualaus pastato suvartojamos elektros energijos prognozei sudaryti. Taip pat teorinėje dalyje yra apžvelgiamas neraiškios logikos reguliatorius, kaip prognozavimo modelio adaptyvumo įrankis, atnaujinant optimalaus prognozavimo modelio parametrų vertes. Metodinėje dalyje yra pateikiama informacija apie pasirinktą individualų pastatą, kuriam yra sudaromas suvartojamos elektros energijos prognozavimo modelis. Taip pat šioje dalyje yra apžvelgiama autoregresinio prognozavimo modelių sudarymo metodika ir naudojami algoritmai modeliams sudaryti. Metodinėje dalyje pateikiama trumpa naudojamos programinės įrangos *Matlab*, ir naudojamų *Matlab* praplėtimų paketų apžvalga. Analitinėje dalyje yra pateikiami sudarytų statistinių ir dirbtinio intelekto prognozavimo modeliai, atliekamas jų tyrimas ir palyginimas, siekiant nustatyti optimalų prognozavimo modelį pasirinktam individualiam pastatui. Taip pat, analitinėje dalyje sudaromas neraiškios logikos reguliatorius, kaip optimalaus prognozavimo modelio parametrų atnaujinimo įrankis, ir atliekas jo tyrimas, nustatant optimalią reguliatoriaus konfigūraciją. Atliekamas palyginimas tarp optimalaus prognozavimo modelio su parametrų atnaujinimu ir be parametrų atnaujinimo.

Ignatavičius, Stanislav. Development and research of adaptive model for building electricity consumption. Master's Final Degree Project / supervisor Dr. Gintaras Dervinis; Faculty of Electrical and Electronics Engineering, Kaunas University of Technology.

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

Keywords: electricity consumption, statistical forecasting models, autoregressive forecasting models, exponential smoothing forecasting model, artificial neural network, fuzzy logic controller, adaptive forecasting model.

Kaunas, 2019. 92 pages.

Summary

The final master thesis - "Development and research of adaptive model for building electricity consumption". In the literature analysis part of this project the overview of statistical and artificial intelligence forecasting models are presented, describing their suitability to forecast the electricity consumption of a little-known individual building. Also, the theoretical part provides an overview of the Fuzzy logic regulator as a adaptability tool for forecasting model to update the values of the optimal forecasting model parameters. The methodology part of this project contains information about the selected individual building, for which electricity consumption forecasting models are created. This section also provides an overview of the methodology used to create autoregressive prediction models as well as an overview of the algorithms which are being used to create forecasting models. The methodology section also provides a brief overview of the used Matlab software and the Matlab extensions packages. In the analytical part of this final project the creation and reasearch of the statistical and artificial intelligence forecasting models are presented. Created forecasting models are compared in order to determine the optimal forecasting model for chosen individual building. Also, in the analytical part, an Fuzzy logic regulator is created as a tool for adapting the parameters of the optimal forecasting model. Analysis of the Fuzzy logic regulator is performed to determine an optimal configuration of the regulator. A comparison is made between an optimal prediction model with parameter update and an optimal prediction model without parameter update.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Įvadas

Elektros energijos suvartojimo prognozė yra ypač paklausi dėl šiais laikais sparčiai didejančio vartotojų skaičiaus, taip pat dėl naujų pramoninių regionų vystymosi Kinijoje ir tolimuosiuose rytuose. Elektros energijos suvartojimo prognozė yra svarbi atokiose vietovėse esantiems individualiems energijos vartotojams, nes vis didėja žmonių gyvenviečių ir įvairių mokslinės paskirties ir ginybos objektų Rusijos ir Kanados šiaurinėse dalyse. Naudojant elektros suvartojimo prognozavimo modelius, užtikrinami mažesni energijos nuostoliai, atsirandantys dėl neteisingo energijos paskirstymo [1,2]. Šie nuostuoliai privalo būti padengti, todėl didėjant nuostoliams tuo pačiu didėja ir kaina už elektros energiją tiek vartotojams, tiek ir energijos tiekėjams. Išankstinis suvartojamos elektros energijos žinojimas tiekėjui leidžia pasiruošti ir sukaupti reikiamą energijos kiekį iš atsinaujinančių energijos šaltinių, taip pat, jeigu yra galimybė, naudoti ir mažesnį kiekį pirminių energijos šaltinių – anglies, naftos, dujų ir kt.

Atokiose vietovėse esantiems naujiems arba atnaujinamiems objektams, sudarant prognozavimo modelius, būtinas modelio prisitaikymo atnaujinimo algoritmas dėl informacijos stokos apie objekto energijos suvartojimą. Taip pat prognozavimo modelio atnaujinimo - prisitaikymo algoritmas yra būtinas, siekiant išlaikyti modelį efektyviai dirbantį ilgesnį laiką be žmogaus įsikišimo, išlaikant objekto efektyvų energijos vartojimą.

Šio tiriamojo darbo tikslas – sukurti optimalų adaptyvų suvartojamos elektros energijos prognozavimo modelį pasirinktam mažai žinomam individualiam pastatui. Darbo uždaviniai:

- remiantis literatūros analize, nustatyti tinkamus statistinius prognozavimo modelius mažai žinomo individualaus pastato suvartojamos elektros energijos prognozei sudaryti;
- sudaryti pasirinktus statistinius prognozavimo modelius ir nustatyti optimalų individualaus, mažai žinomo pastato suvartojamos elektros energijos statistinį prognozavimo modelį;
- sudaryti dirbtinio neuroninio tinklo mažai žinomo individualaus pastato suvartojamos elektros energijos prognozavimo modelį ir nustatyti optimalią dirbtinio neuroninio tinklo struktūrą;
- atlikti palyginimo analizę tarp optimalaus statistinio ir dirbtinio neuroninio tinklo prognozavimo modelių;
- sudaryti neraiškios logikos reguliatorių optimalaus prognozavimo modelio parametrų atnaujinimui ir nustatyti optimalią reguliatoriaus konfigūraciją;
- sudaryti optimalų adaptyvų pasirinkto mažai žinomo individualaus pastato suvartojamos elektros energijos prognozavimo modelį ir atlikti analizę, lyginant optimalaus modelio prognozę su adaptyvaus optimalaus modelio prognoze, esant trikdžiams pastato elektros energijos vartojime.

1. Literatūros analizė

1.1. Teorinis objektas

Šiais laikais atsiranda vis daugiau gyvenviečių ir strateginių mokslinės bei karinės paskirties objektų, esančių atokiose pasaulio vietose. Šios atokios vietos yra Rusijos, Kanados ir Skandinavijos šiaurinės dalys, taip pat atokios salos, kurios neturi tiesioginio susisiekimo su žemynais per sausumą, tačiau yra mokslinių tyrimų arba karinės strategijos objektai. Dėl geografinės padėties elektros energijos tiekimas į šiuos objektus yra sudėtingas ir tik labai retais atvejais yra realizuotas, naudojant elektros paskirstymo tinklus. Jeigu ir naudojamas elektros paskirstymo tinklas, elektros tiekimui į tokius objektus, tai dažniausiai toks tinklas neišduoda reikiamo galios kiekio, kuris užtikrintų naudojamos elektros įrangos korektišką funkcionavimą. Dažniausiai tokiuose objektuose naudojami elektros energijos šaltiniai yra dyzeliniai generatoriai, kurie elektros energijos gamybai naudoja dyzelį arba kitokį pirminės energijos šaltinį. Tokius energijos šaltinius stengiamasi naudoti kuo mažiau, nes jiems reikalingas pastovus pirminės energijos šaltinių tiekimas, bet dėl naftos produktų didėjančios kainos ir kuro transportavimo išlaidų, šių objektų eksploatacija tampa ekonomine našta objektų savininkams. Dėl to šiais laikas, kaip pagrindiniai elektros energijos šaltiniai, vis plačiau yra naudojami atsinaujinantys energijos šaltiniai – vėjo jėgainės ir saulės elementai.

Norint efektyviai išnaudoti atsinaujinančios elektros energijos šaltinius, jų generuojamą energiją būtina paskirstyti tiesioginiam vartojimui, perduodant ją į valdymo ir galios grandinę, arba sugeneruotą energiją kaupti. Norint atlikti tokį energijos paskirstymą, reikia žinoti būsimą elektros energijos vartojimą, kurį galima sužinoti, naudojant suvartojamos elektros energijos prognozavimo modelius. Individualiems objektams sudaryti suvartojamos elektros energijos prognozavimo modelį yra sudėtingiau, nei žemyne esančių gyvenviečių individualiems pastatams, ir žymiai sudėtingiau, nei elektros energijos paskirstymo stotims arba elektrinėms. Pagrindinės tokių objektų prognozavimo modelio sudarymo sudėtingumo priežastys yra:

a) Informacijos trūkumas apie patį objektą ir jo aplinką. Senos statybos mokslinių arba karinių strateginių objektu renkama informacija apie aplinką arba objektą (pastatą) yra bevertė ir nėra naudojama, sudarant objekto elektros energijos suvartojimo prognozavimo modelį. Jeigu objektas yra naujas, tuomet informacijos apie pati objektą arba jo aplinką papraščiausiai nėra.

b) Objekto suvartojamos elektros energijos kiekio nepastovumas. Atokiuose vietovese esantys objektai dažniausiai yra autonominiai arba pusiau autonominiai, tačiau jiems yra būtinas žmogaus įsikišimas. Atlikdami tam tikrus darbus objektuose žmonės gyvena tam pritaikytuose patalpose, kurios naudoja elektros energiją. Toks pokytis objekto aplinkoje į objekto elektros vartojimo sistemą įneša svyravimus.

Lyginant objektą pagal jo dydį ir suvartojamą elektros energijos kiekį, objekte individuali žmogaus veikla yra jaučiama labiau, nei tokia pat veikla yra jaučiama dideliuose ofisų kompleksuose arba gyvenamuosiuose daugiabučiuose. Pagal suvartojamos elektros energijos kiekį aprašytų objektų energijos suvartojimas yra artimas individualių gyvenamųjų pastatų suvartojamam energijos kiekiui.

Remiantis autoriaus patirtimi ir aprašytų objektų prognozavimo modelių realizavimo sudėtingumo priežastimis, buvo sudaryti reikalavimai, kuriuos turi tenkinti pasirinktas suvartojamos elektros energijos kiekio prognozavimo modelis. Šie reikalavimai yra:

1. Minimalus istorinių duomenų kiekis reikalingas sudaryti prognozavimo modelį. Remiantis *a* priežastimi, informacijos kiekis apie naują arba modernizuojamą objektą yra minimalus, todėl pasirinktas prognozavimo modelis turi būti apmokytas arba sudarytas, naudojant duomenis, kurie leistų objektui dirbti be prognozavimo modelio kiek įmanoma mažesnį laiko tarpą.

2. Pasirinktame prognozavimo modelyje privalo būti modelio savikorekcijos algoritmas. Remiantis *b* priežastimi, aprašyto objekto energijos suvartojimas keičiasi priklausomai nuo tam tikrų išorinių veiksnių, oro sąlygų, taip pat nuo objekto atliekamų funkcijų bei nuo objekte esančių žmonių veiklos. Objekto suvartojamos elektros energijos kiekis, priklausantis nuo išorinių veiksnių ir objekto atliekamų funkcijų, teoriškai turi tam tikrą priežastingumą ir nekinta šuoliškai, dėl to yra prognozuojamas lengviau, nei nuo žmogaus veiklos priklausantis elektros energijos suvartojimas. Pastarasis energijos suvartojimas yra apibūdinamas kaip chaotiškas, neturintis tam tikro nuoseklumo, kuris suprastintų prognozavimo procesą. Dėl to pasirinktame prognozavimo modelyje privalo būti savikorekcijos algoritmas, kuris leistų pakeisti prognozavimo modelį, jame įvykus staigiems pokyčiams. Taip pat šis modelio korekcijos algoritmas turi atnaujinti modelį per kiek įmanomą trumpesnį laiką ir išlaikyti norima prognozavimo tikslumą.

Šio baigiamojo projekto teorinėje dalyje yra apžvelgiami prognozavimo modeliai, kurie labiausiai tiktų individualaus pastato suvartojamos energijos prognozei atlikti. Todėl visi apžvelgti prognozavimo modeliai yra vertinami pagal tai, kaip jie gerai galėtų prognozuoti paminėto mažai žinomo individualaus pastato suvartojamą elektros energiją. Parenkant prognozavimo modelius teorinėje šio darbo dalyje yra atsižvelgiama į reikalavimus pateiktus šioje dalyje.

1.2. Bendrosios prognozavimo metodų kryptys – statistiniai ir dirbtinio intelekto modeliai

1.2.1. Statistiniai prognozavimo modeliai

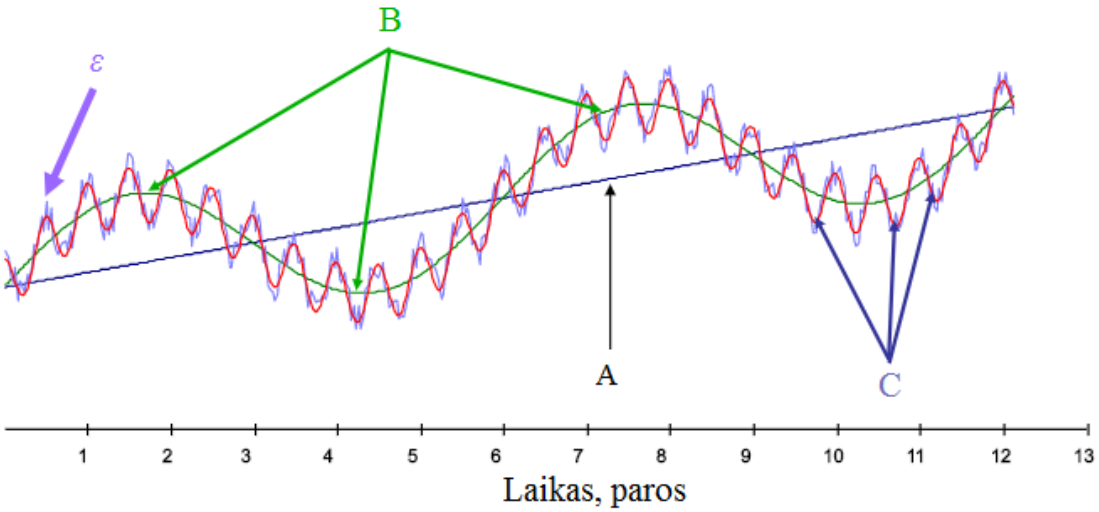
Statistiniai prognozavimo modeliai – tai prognozavimo modeliai, kuriuose suvartojama elektros energija yra aprašoma kaip matematinė funkcija priklausanti nuo tam tikrų parametrų [1]. Statistiniai prognozavimo modeliai yra skirstomi į du pogrupius – sudėties modeliai (angl. *additive models*) ir daugybos modeliai (angl. *multiplicative models*).

Statistiniai modeliai yra priskiriami prie sudėties modelių tuomet, kai modelio suvartojamos elektros energijos prognozė yra suma priklausanti nuo parametrų, naudojamų prognozei sudaryti. Šio modelio supaprastinta matematinė išraiška bendruoju atveju yra pateikiama pirmoje formulėje (žr. 1.2.1 formulę).

$$L = A + B + C + \varepsilon; \tag{1.2.1}$$

L – bendroji suvartojama elektros energija, kurią prognozuoja sudarytas statistinis sudėties prognozavimo modelis. Šiuo atveju bendroji suvartojama elektros energija priklauso nuo *A, B, C* ir ε dedamųjų, kur *A* dedamoji atitinka bazinę suvartojamos elektros energijos dalį, kitaip vadinama tendencija (angl. trend), ir apsprendžia bendrąją visos suvartojamos elektros energijos kryptį – formą. Pateiktos formulės *A* dalis yra naudojama sudaryti prognozę ilgiausiam laikotarpiui – parai, mėnesiui, metams ir t.t. Suvartojamos elektros energijos dalis *B* – tai dalis, priklausanti nuo išorinių reiškinių, kurių negali kontroliuoti vartotojas. Dažniausiai tai yra oro sąlygos, kurios apsprendžia ne pagrindinę, tačiau ženklią suvartojamos elektros energijos dalį. Ši prognozės dalis dar yra vadinama *cikline* dalimi, nes apibūdina prognozės svyravimus apie trendo liniją (*A* dalį). *B* dalis daro

trumpesnio poveikio įtaką bendrai prognozei nei *A* dalis. *C* dalis išreiškia nestandartinę sistemos suvartojamos elektros energijos dalį, kuri remiasi į pasitaikančias šventines dienas arba atostogas. Pateiktos formulės *C* dalis taip pat yra vadinama sezonine dedamąja, ji atitinka svyravimus apie ciklinę dedamąją ir privalo apibūdinti sistemos elektros energijos suvartojimą trumpesniame laiko intervale. Paskutinė sudėties prognozavimo medelio dalis ε atitinka triukšmą arba trikdžius pačioje sistemoje arba objekte. Komponentė ε – tai nepastovioji komponentė, kuri atitinka elektros suvartojimą trumpiausiais įmanomais laiko intervalais ir turinti atsitiktinį pasirodymo dažnumą. Aprašyta sudėties modelio forma pirmoje formulėje (žr. 1.2.1 formulę) yra tik bendroji modelio išraiška. Esant būtinumui, prie modelio galima pridėti papildomų komponenčių, kurios leistų padidinti modelio prognozės tikslumą, tačiau tai priklauso nuo to, kiek gerai yra žinomas nagrinėjamas objektas arba sistema.



1.2.1 pav. Sudėties prognozavimo modelio komponenčių įtaka bendrai modelio prognozei

Daugybos prognozavimo modeliai – tai modeliai, kurie skiriasi nuo sudėties modelių tuo, kad juose tarp naudojamų komponenčių ryšys yra išreiškiamas ne sudėtimi, o daugyba. Tad tokiame modelyje bendroji suvartojamos elektros energijos prognozė yra lygi sandaugai tarp naudojamų modelio komponenčių (žr. 1.2.2 formulę).

$$L = A * B * C * \varepsilon; \qquad (1.2.2)$$

Kaip ir sudėties modelyje, taip ir daugybos prognozavimo modelyje, naudojamos komponentės turi tą pačią prasmę. *L* – bendroji suvartojamos elektros energijos prognozė, *A* dalis atitinka tendencijos dedamąją, *B* dalis atitinka ciklinę dalį, *C* dalis atitinka sezoninę dalį, o ε atitinka nepastoviąją dalį.

Prie statistinių prognozavimo modelių yra priskiriami [1]:

- Panašios dienos paieškos modelis (angl. Similar-Day Method);
- Eksponentinio išlyginimo modelis (angl. Exponential Smoothing);
- Regresiniai modeliai (angl. Regression Methods);
- Autoregresiniai modeliai (angl. Autoregressive Model);

- Autoregresinis su slenkančių vidurkiu modelis (angl. Autoregressive Moving Average Model);
- Autoregresinis integruoto slenkačio vidurkio modelis (angl. Autoregressive Integrated Moving Average Model);
- Autoregresiniai modeliai su išoriniais kintamaisiais (angl. Autoregressive Model with Exogenous Variables);

1.2.2. Prognozavimo modeliai, paremti dirbtiniu intelektu

Dėl sparčiai besivystančių kompiuterinių technologijų, prognozavimo modeliams sudaryti yra naudojami dirbtinio intelekto (angl. artificial intellegence) modeliai. Pati dirbtinio intelekto koncepcija buvo sukurta tuomet, kai kompiuterinės sistemos nusileisdavo savo pajėgumais žmogiškajam intelektui, tačiau šiais laikas sukurtas dirbtinis intelektas yra daug pažangesnis už pačius pirmuosius savo variantus ir gali atlikti kai kurias funkcijas tokiame pačiame lygyje arba net geriau, nei tai gali atlikti žmogiškasis intelektas. Dėl savo pažangumo, dirbtinis intelektas sudaro galingą ir universalų įrankį, kuris leidžia surasti sprendimą problemoms arba uždaviniams kurių negalima išspręsti klasikiniais metodais [3].

Prie dirbtinio intelekto prognozavimo modelių yra priskiriami modeliai, kurie naudoja [1,3]:

- Dirbtinius neuroninius tinklus (angl. *Artificial Neural Network*);
- Neraiškių aibių logiką (angl. *Fuzzy Logic*);
- Ekspertines sistemas (angl. *Expert System*);
- Palaikančias vektorines mašinas (angl. *Support Vector Machines*);
- Genetinius algoritmus (angl. *Genetic Algorithm*).

Dirbtinio intelekto prognozavimo modelis, naudojantis dirbtinius neuroninius tinklus, yra plačiausiai naudojamas dėl prognozavimo uždavinio tiesmukiško sprendimo ir sąlyginai gerų rezultatų. Lyginant dirbtinio neuroninio tinklo prognozavimo modelio ir palaikančios vektorinės mašinos prognozavimo modelio prognozavimo rezultatus, dirbtinio neuroninio tinklo modelis yra tikslesnis nei palaikančios vektorinės mašinos prognozavimo modelis [4]. Kiti dirbtinio intelekto modeliai taip pat yra naudojami, sudarant elektros energijos prognozavimo modelius, tačiau mažiau naudojami, kaip atskiros pavienės sistemos, plačiau – kaip hibridinių prognozavimo modelių komponentai kartu su dirbtiniais neuroniniais tinklais [5,4,6].

1.3. Pagrindiniai statistinių prognozavimo modulių realizacijos metodai

1.3.1. Panašios dienos modelis

Panašios dienos prognozavimo modelis (angl. *similar day approach*) – tai pats paprasčiausias prognozavimo modelis paremtas detalia istorinių duomenų analize. Modelyje, priklausomai nuo turimų objekto parametrų, ieškoma panašios dienos su tokiais pačiais parametrais iš nagrinėjamo objekto istorinių duomenų. Naudojami palyginimo parametrai gali būti savaites diena, mėnesio diena, mėnuo ir orų prognozės parametrai: temperatūra, drėgmė, vėjo greitis, debesuotumas

ir kt. [7]. Suradus panašią dieną pagal aukščiau išvardintus parametrus, jos tikras suvartotas energijos kiekis yra prilyginamas objekto suvartojamos elektros energijos kiekio prognozei [8]. Kartais panašios dienos prognozavimo modelis yra naudojamas, sudarant prognozę specialiems laikotarpiams, šventinėmis dienomis arba savaitgaliais suvartojimai elektros energijai prognozuoti [8], pastarųjų dienų elektros energijos suvartojimas skiriasi nuo elektros suvartojimo darbo dienomis.

Panašios dienos prognozavimo modelis taip pat gali būti naudojamas ne tik kaip atskiras prognozavimo modelis, tačiau ir kaip įrankis, realizuojant pilnesnį prognozavimo modelį [9]. Kaip pavyzdys šis modelis gali būti naudojamas kartu su regresiniais modeliais. Kai panašios dienos modelio rastos kelios panašios valandos, dienos arba net savaitės parametrai sutampa su objekto parametrais ir yra surišami tarpusavyje naudojant regresinius modelius. To pagrindu yra sudaroma nagrinėjamo objekto suvartojamos elektros energijos prognozė [1]. Tomonobu Senjyu ir kitų 2002 m. publikuotame straipsnyje pasiūlytas prognozavimo modelis, paremtas dirbtiniais neuroniniais tinklais, apmokytas, naudojant atrinktas dienas, kurių aplinkybių parametrai yra panašūs į lyginamosios dienos parametrus [10]. Tomonobu Senjyu pasiūlytas prognozavimo modelis sumažino neuroninio tinklo apmokymo trukmę ir struktūra naudojant tik panašias dienas tinklo apmokymui. Sudaryto prognozavimo modelio vidutinė paklaida siekia vos 1,63% nuo realių duomenų.

Aprašytas prognozavimo metodas yra paprastas ir primityvus, tačiau retai yra naudojamas dėl savo ne didelio tikslumo. Panašios dienos modelis yra naudojamas kaip įrankis, leidžiantis pagerinti pagrindinio prognozavimo modelio tikslumą [8,9,10].

Pagal aprašyta teorinį modelį, panašios dienos prognozavimo modelis negali būti naudojamas kaip pagrindinis teorinio objekto suvartojamos elektros energijos prognozavimo modelis. Jis netinka, nes norint sudaryti tokį modelį reikia turėti didelį duomenų kiekį kuris apimtu kelis metus, kurie leistų sudaryti adekvačiai funkcionuojanti prognozavimo modelį. Panašios dienos modelis gali būti taikomas jeigu objektas yra gerai žinomas, tačiau tuomet tokiam objektui yra naudingiau taikyti tikslesnius statistinius arba dirbtinio intelekto prognozavimo modelius.

1.3.2. Regresiniai modeliai

Regresiniai prognozavimo modeliai yra priskiriami prie statistinių prognozavimo modelių. Pastarieji modeliai yra paremti regresiniais analizės metodais. Regresiniai analizės metodai – tai statistiniai procesai, kurie leidžia nustatyti ryšį tarp naudojamų nepriklausomų kintamųjų ir tarp vieno ar kelių priklausomų kintamųjų. Ryšis tarp šių kintamųjų yra atvaizduojamos funkcijos pavidalu (žr. 1.3.1 formulę).

$$Y = f(X, \beta); \qquad (1.3.1)$$

*čia **Y** atitinka priklausomąjį kintamąjį, **X** atitinka nepriklausomą kintamąjį, o **β** atitinka koeficientus, kurie leidžia surišti priklausomą ir nepriklausomą kintamuosius. Regresinis ryšys yra išreiškiama funkcijos pavidalu.*

Regresiniai modeliai leidžia nustatyti, kaip kinta priklausomo kintamojo reikšmė, pakitus vienam iš nepriklausomų kintamųjų, kai kiti nepriklausomi kintamieji nekinta. Statistiniai regresiniai modeliai yra skirstomi į du pagrindinius tipus – tiesiniai regresiniai modeliai (angl. linear regression model), kai ryšį tarp kintamųjų galima aprašyti tiesine funkcija, ir netiesiniai regresiniai modeliai (angl. nonlinear regression model), kai kintamųjų tarpusavio ryšys yra netiesinis. Visų regresinių modeliu tikslas – surasti tokias koeficientų β_1, β_2, ... β_i, vertes, kurios sudarytų minimalų skirtumą tarp apskaičiuoto priklausomo kintamojo ir tikrosios priklausomo kintamojo vertės, gautos eksperimentinių būdu. Beveik visi sudaryti regresiniai modeliai kaip tikslumo kriterijų naudoja

mažiausių kvadratų metodą (angl. least squares), kuris atitinka skirtumo tarp apskaičiuoto ir tikrojo priklausomo kintamojo kvadratų sumą (žr. 1.3.2 ir 1.3.3 formules).

$$e = y - f(x, \beta); \quad (1.3.2)$$

$$S = \sum_{i=1}^{n} e_i^2; \quad (1.3.3)$$

čia ***e*** *atitinka paklaidą tarp tikrosios priklausomo kintamojo vertės* ***y*** *ir apskaičiuotos priklausomo kintamojo vertės, kuri priklauso nuo nepriklausomo kintamojo* ***x*** *ir koeficiento* ***β****. Penktoje formulėje* ***S*** *atitinka paklaidos kvadratų sumą.*

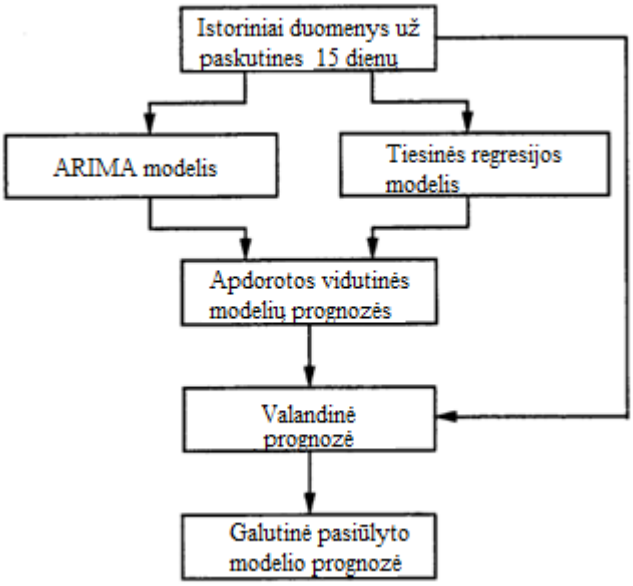
1.3.3. Tiesinės regresijos prognozavimo modeliai.

Tiesinės regresijos prognozavimo modeliai – tai modeliai, kuriuose priklausomas kintamasis yra suvartojama elektros energija, kuri tiesiškai priklauso nuo naudojamų nepriklausomų kintamųjų [8]. Kaip nepriklausomi kintamieji, tiesinės regresijos modeliuose gali būti naudojami įvairiausi kintamieji, kurie turi įtakos objekto energijos suvartojimui, tačiau dažniausiai naudojami kintamieji yra valandos, savaitės diena, elektros energijos kainos ir orų prognozės parametrai: temperatūra, drėgme ir kiti parametrai [1,7]. Bendroji tiesinės regresijos prognozavimo modelio išraiška yra pateikta formulėje žemiau (žr. 1.3.4 formulę).

$$L_t = a_0 + a_1 * B_1 + \cdots + a_n * B_n + \varepsilon; \quad (1.3.4)$$

čia $\mathbf{L_t}$ *– tai bendroji nagrinėjamo objekto suvartojamos elektros energijos prognozė,* $\mathbf{a_0}$ *– tai laisvasis koeficientas,* $\mathbf{a_i}$ *, kai* $\mathbf{i > 0}$*, – koeficientas, kuris atlieka skaliavimą nepriklausomam kintamajam.* $\mathbf{B_i}$ *– tai nepriklausomas kintamasis, kuris turi įtakos* $\mathbf{L_t}$*.* $\boldsymbol{\varepsilon}$ *– tai triukšmas, kuris taip pat daro įtaka modelio prognozei.*

Alex D. Papalexopoulos ir Timothy C. Hesterberg 1990 m. publikuotame straipsnyje sudarė trumpojo laikotarpio modelį, kuris atliko prognozavimą kas-valanda savaitei į priekį [14]. Pasiūlytame prognozavimo modelyje yra naudojamas tiesinės regresijos modelis ir ARIMA prognozavimo modelis, sudarant suvartojamos elektros energijos maksimumų prognozę kas-valanda.



1.3.1 pav. Alex D. Papalexopoulos pasiūlyto prognozavimo modelio algoritmas [11]

Alex D. Papalexopoulos pasiūlytas prognozavimo modelis buvo padalintas į 4 atskirus modelius pagal metų sezonus – vasaros, pavasario rudens ir žiemos. Realizuotas prognozavimo modelis

geriausiai pasirodė žiemos periodu, prognozės vidutinė paklaida nuo tikrųjų duomenų siekia vos 0.02% , vasaros sezonu vidutinė paklaida vos 0.3% [11]. Sudaryto modelio prognozė, dėl dažnai besikeičiančių orų, turėjo didelę paklaidą nuo tikrųjų duomenų rudens ir pavasario sezonais [11]. Naudojamas tiesinis prognozavimo modelis gerai pasirodo, kai naudojami nepriklausomi parametrai keičiasi tendentiškai ir nėra didelių jų verčių šuolių, tačiau jie prastai pasirodo tuomet, kai naudojami nepriklausomi kintamieji keičiasi drastiškai per trumpą laiką [11].

Apel Mahmud 2001 m. publikavo straipsnį, kuriame nurodė, jog naudojant tiesinę regresiją įmanoma sudaryti nežinomo objekto (regiono) suvartojamos elektros energijos prognozavimo modelį, panaudojus kito panašaus objekto nedidelį kiekį duomenų [12]. Remiantis šiuo straipsniu galima teigti, jog norint sudaryti tiesinės regresijos prognozavimo modelį, nebūtina naudoti didelį kiekį duomenų, nebūtina naudoti net to paties objekto duomenų, tačiau nuo dviejų objektų duomenų skirtumo priklauso realizuojamo modelio prognozavimo tikslumas.

Naudojant tiesinės regresijos modelius daroma prielaidą, kad objekto suvartojamos elektros energija tiesiškai priklauso nuo pasirinktų nepriklausomų kintamųjų. Jeigu sudarytas tiesinės regresijos prognozavimo modelis nepasiekia nustatyto tikslumo kriterijaus, galima panaudoti netiesinės regresijos prognozavimo modelį.

Tiesinės regresijos modeliai nėra tinkami naudoti aprašyto teorinio modelio suvartojamos elektros energijos prognozėms sudaryti. Naudojant šį prognozavimo modelį būtina žinoti kurie aplinkos parametrai daro įtaką sistemos suvartojimui ir žinoti koks tiesinis ryšys sieja šiuos išorinius kintamuosius su objekto elektros suvartojimu. Jeigu ryšis tarp išorinių kintamųjų ir objekto suvartojimo nėra nustatytas, tuomet šį modelį negalima naudoti prognozėms sudaryti. Taip pat šiame modelyje nėra savikorekcijos algoritmo kuris leistų koreguoti sudaryto modelio parametrus. Naudojant šį prognozavimo modelį būtina naudoti papildomus metodus kurie leistų atnaujinti modelį pakitus objekto parametrams.

1.3.4. Netiesinės regresijos prognozavimo modeliai.

Netiesinės regresijos elektros energijos prognozavimo modeliai – tai modeliai, kuriuose ryšys tarp priklausomų ir nepriklausomų kintamųjų yra išreiškiamas, naudojant netiesinę funkciją. Netiesiškumo ryšys gali būti išreikštas bet kuria forma, tačiau bendroji netiesinės regresijos forma yra pateikiama formulėje žemiau (žr. 1.3.5 formulę).

$$y \sim f(x, \beta); \tag{1.3.5}$$

*čia **y** atitinka priklausomą kintamąji, **x** atitinka nepriklausomą kintamąjį, o **β** atitinka koeficientą.*

Kaip ir tiesinės regresijos atveju, netiesinės regresijos prognozavimo modeliuose, kaip tikslumo kriterijus, yra naudojamas mažiausių kvadratų metodas (žr. 1.3.3 formulę). Kaip ir tiesinės regresijos modeliuose taip ir netiesinės regresijos modeliuose pagrindinis tikslas yra nustatyti naudojamos funkcijos parametrus, kurie užtikrintų mažiausią įmanomą prognozavimo paklaidą nuo tikrųjų duomenų. Tačiau skirtingai, nei tiesinės, netiesinės regresijos prognozavimo modeliuose būtina iš anksto žinoti, kokiu netiesiškumu tarpusavyje yra surišti naudojami nepriklausomi kintamieji. Nors ir modelio sudarymo metu, pasirinkus tam tikrą netiesiškumo ryšį tarp kintamųjų, nebūtina užtikrinti artimą suvartojamos elektros energijos prognozę tikrajam suvartojamam energijos kiekiui. Nežinant objekto netiesinio ryšio tarp priklausomų ir nepriklausomų kintamųjų, galima naudoti duomenų interpoliaciją ir ekstrapoliaciją. Atlikus literatūros analizę, nustatyta, kad sudarant suvartojamos

elektros energijos prognozavimo modelius, esant netiesiniam ryšiui tarp kintamųjų, autoriai dažniausiai naudoja modelio sudarymo metodus, kurie atleidžia juos nuo būtinybės ieškoti netiesinio ryšio tarp kintamųjų.

Kaip ir panašios dienos modeliai, taip ir regresiniai modeliai dažniau yra naudojami kaip komponentai, sudarant hibridinius elektros energijos suvartojimo prognozavimo modelius. Slobodan Ružic 2003 m. publikuotame straipsnyje pasiūlė prognozavimo modelį, paremta regresiniais metodais, sudarant Serbijos elektrinės apkrovos prognozę trumpuoju laikotarpiu [13]. Pasiūlytas trumpojo prognozavimo laikotarpio modelis adaptyvus ir priklausantis nuo orų prognozės, taip pat modelio algoritmas susidaro iš dviejų žingsnių, kur pirmajame žingsnyje yra atliekama bendra paros suvartojamos elektros energijos prognozavimas, o antrame žingsnyje yra atliekamas kasvalandinis suvartojamos elektros energijos prognozavimas. Realizuotas prognozavimo modelis parodė gerus rezultatus, nes prognozės paklaida siekia vos 1,64% MAEE ir 2,73% MALE atitinkamai [13].

O. Hyde ir P.F. Hodnett savo straipsnyje publikuotame 1997 metais, pasiūlytame prognozavimo modelyje tiesinę regresiją naudojo, kaip papildomą komponentą prognozavimo modelyje [14]. Jų pasiūlytas prognozavimo modelis buvo paremtas tiesine regresija, kurioje kaip nepriklausomi kintamieji buvo naudojami orų prognozės parametrai. Sudarytas trumpojo prognozavimo laikotarpio modelis buvo skirtas prognozuoti Airijos elektros energijos paskirstymo sistemos apkrovą 7-10 dienas į priekį, mėnesį į priekį ir vieną valandą į priekį. Realizuoti prognozavimo modeliai yra tikslūs, kai mėnesiui į priekį prognozės vidutinė paklaida yra iki 2%, o valandą į priekį prognozės nuokrypis nuo tikrųjų duomenų siekia vos 0,7% ir 0,6% [14].

Regresiniai suvartojamos elektros energijos prognozavimo modeliai yra ganėtinai tikslūs modeliai, tačiau vis rečiau yra naudojami kaip savarankiški prognozavimo modeliai ir vis daugiau yra naudojami kaip komponentai tikslesniuose prognozavimo modeliuose. Regresiniai modeliai gerai pasirodo tuomet, kai tarp priklausomų ir nepriklausomų kintamųjų yra tiesinis ryšys, tačiau prastai pasirodo tuomet kai šis ryšys yra netiesinis ir savo tikslumu nusileidžia kitiems prognozavimo modeliams.

Atlikus regresinių modelių analizę paaiškėjo, kad regresiniai prognozavimo modeliai gerai atlieka drastiškai kintančių dydžių prognozę, tuomet kai yra žinomas ryšys tarp priklausomų ir nepriklausomų kintamųjų. Šiuose modeliuose prognozavimo tikslumas priklauso nuo to, kaip gerai yra nustatytas kintamųjų ryšys ir kaip gerai yra nustatyti nepriklausomų kintamųjų koeficientai, nes priklausomo kintamojo vertė yra apskaičiuojama analitiškai pagal nepriklausomų kintamųjų vertes. Tokio modelio privalumas – galimybė sudaryti modelį su nedidelių kiekių duomenų [12]. Regresinio modelio vienas iš trūkumų yra savikorekcijos algoritmo stoka, kuris neleidžia modeliui sau pačiam apsimokyti ir viena karta nustatyti modelio koeficientai lieka nepakitę per visą modelio naudojimo laiką. Dar vienas modelio trūkumas – modelyje sudėtinga įvertinti žmogaus chaotišką elgesį ir tai neįmanoma atlikti, naudojant koeficientus. Regresiniai modeliai yra greiti ir pakankamai tikslūs, tačiau yra netinkami aptarto objekto suvartojamos elektros energijos prognozei sudaryti.

Regresiniai modeliai nėra tinkami, nes modelyje nėra savikorekcijos algoritmo, sunkiai realizuojamas žmogaus veiklos įvertinimas, modelyje yra pasikliaunama tik išorinių kintamųjų įtaka objekto suvartojamos elektros energijos kiekiui.

1.4. Autoregresiniai prognozavimo modeliai

Modeliai, apibūdinantys laike kintančius parametrus, yra sudaromi pagal prielaidą, kad naudojami kintamieji turi tarpusavio ryšį išreiškiamą autokoreliaciją, trendo liniją (anlg. trend) arba priklausomo kintamojo ciklišku. Prognozavimo modeliai yra sudaromi, siekiant surasti tarpusavio ryšį tarp kintamųjų, kuris leistų atlikti ateities suvartojamos elektros energijos prognozę, remiantis turimais istoriniais duomenimis. Jeigu prognozavimo modelio autorius daro prielaidą, kad būsimas suvartojamos elektros energijos kiekis (prognozė) priklauso nuo prieš tai tikrojo suvartoto elektros energijos kiekio, tuomet sudaromas modelis yra priskiriamas prie autoregresinių prognozavimo modelių. Autoregresiniai modeliai – tai statistiniai prognozavimo modeliai, naudojami apibūdinti atsitiktinį procesą, kuris remiasi prielaida, kad ateities prognozė tiesiškai priklauso nuo buvusių tikrųjų verčių. N-tosios eilės autoregresinio modelio matematinė išraiška yra pateikiama formulėje žemiau (žr. 1.4.1 ir 1.4.2 formules).

$$X_t = c + \sum_{i=1}^{N} \varphi_i * X_{t-i} + \varepsilon_t; \quad (1.4.1)$$

$$X_t = c + \sum_{i=1}^{N} \varphi_i * B^i X_{t-i} + \varepsilon_t; \quad (1.4.2)$$

čia $\varphi_1, \varphi_2, \ldots, \varphi_n$ *atitinka autoregresionio modelio nežinomus koeficientus,* X_t *– tai nagrinėjamos sistemos suvartojamos elektros energijos prognozė* t*-uoju laiko momentu.* ε_t *– tai nagrinėjamos sistemos suvartojamos elektros energijos trukdžiai, kurie daro įtaką energijos prognozei.* B^i *– grįžtamumo operatorius.*

Tuomet bendruoju atveju, kai konstanta *c* yra lygi nuliui, sistemos pokytis tarp *t* ir *t-1* laiko momentu yra lygus sistemos trikdžiams ε (žr. 1.4.3 formulę).

$$\phi[B] * X_t = \varepsilon_t; \quad (1.4.3)$$

Naudojamame autoregresiniame modelyje tam tikri parametrai turi būti apriboti tam, kad sudarytas modelis išliktų stacionarus ir išlaikytų priklausomybę nuo prieš tai buvusių verčių. Bendruoju atveju sudarytas autoregresinis modelis yra stacionarus, kai charakteringos lygties poliai, polinėje koordinačių ašyje, yra vienetinio apskritimo viduje ir tenkina sąlyga $|z_i|<1$ (žr. 1.4.4 ir 1.4.5 formules).

$$z^p - \sum_{i=1}^{p} \varphi_i * z^{p-1} = 0; \quad (1.4.4)$$

$$1-\varphi_1 z - \varphi_2 z_2 - \ldots - \varphi_p z_p = 0; \quad (1.4.5)$$

Suvartojamas elektros energijos kiekio prognozavimas yra stacionarus procesas, jeigu modelio statistinės savybės nesikeičia bėgant laikui. Jeigu sudaryto modelio prognozavimo vidurkis, dispersija (angl. *variance*) ir autokoreliacija nekinta bėgant laikui tuomet modelis yra stacionarus. Autokoreliacija, kitaip dar vadinama vėlinimo funkcija – tai funkcija, kuri nurodo signalo arba dydžio koreliaciją su jo paties kopija pavėlinta per tam tikrą žingsnį *L*. Prognozavimo modelio kovariacija yra stacionari tuomet, kai charakteringos lygties poliai patenka į vienetinį apskritimą (žr 1.4.5 formulę) [1]. Kovariacija – tai matas, leidžiantis išreikšti dviejų analizuojamų dydžių tarpusavio priklausomybę kiekybiškai. Priešingu atveju, jeigu charakteringos lygties šaknys yra ant vienetinio apskritimo, tuomet autoregresinio modelio autokovariacija kinta laike.

Autoregresinio modelio eilė parodo, kiek laiko žingsnių atgal sistemos suvartojamos energijos kiekis daro įtaką būsimai sistemos suvartojamos elektros energijos prognozei. Jeigu autoregresinio modelio eilė yra *p*, tuomet sudaryto modelio prognozė priklauso nuo *p* žingsnių atgal tikro suvartoto elektros energijos kiekio. Pats paprasčiausias autoregresinis prognozavimo modelis yra pirmos eilės, kai p=1. Tuomet modelio prognozė priklauso tik nuo prieš tai buvusios tikro elektros energijos suvartojimo kiekio (žr. 1.4.6 formulę) [15].

$$X_t = c + \varphi * X_{t-1} + \varepsilon_t; \quad (1.4.6)$$

Norint apskaičiuoti autoregresinio modelio nežinomus parametrus φ_i, būtina žinoti modelio autokoreliacijos funkciją. Autoregresinio modelio parametrus galima apskaičiuoti, naudojant mažiausių kvadratų metodą, momentų metodą paremtą Yule–Walker lygtimi arba naudojant maksimalios tikimybės įvertinimo (angl. *maximum likelehood*) metodą. Plačiau apie autoregresinių modelių parametrų nustatymo metodus yra patekta metodinėje dalyje.

Autoregresiniai metodai yra plačiai naudojami statistinių procesų analizėse, tokiose kaip ekonomika, skaitmeninių signalų analizė bei plačiai yra taikomi elektros apkrovos prognozavimo modeliuose. Tačiau, kaip ir kiti statistiniai duomenų analizės metodai, autoregresiniai metodai vis dažniau yra naudojami kaip duomenų apdorojimo priemonė, siekiant sudaryti tikslesnį elektros apkrovos prognozavimo modelį. M. T. Hagan ir S. M. Behr 1987 m. publikuotame straipsnyje pasiūlė autoregresinį metodą naudoti, norint patikslinti Bokso ir Jenkinso (angl. Box and Jenkins) tiesinio prognozavimo modelio tikslumą [15]. Bokso ir Jenkinso prognozavimo modelis yra tiesinis modelis, kuris netiksliai atvaizduoja netiesinį ryšį tarp suvartojamos elektros energijos ir temperatūros. Publikuotame darbe autoregresinis metodas yra naudojamas sudaryti ryšį tarp sistemos temperatūros ir elektros energijos suvartojimo bei nustatant realizuojamų ARIMA ir Bokso-Jenkinso modelių struktūras ir eilę [15]. Gauti autorių rezultatai patvirtino, kad naudojant autoregresinį metodą, kaip pagalbinį metodą, sudarant suvartojamos elektros energijos prognozavimo modelį, galima padidinti modelio tikslumą [15].

S.R. Huang 1997 metais publikuotame straipsnyje pasiūlė suvartojamos elektros energijos trumpojo laikotarpio prognozavimo modelį, kuris atliko prognozavimą kas valanda. Pasiūlytas prognozavimo modelis naudoja slenkstinį autoregresinį metodą su stratifikacijos taisykle (angl. *stratification rule*). Realizuotame modelyje yra naudojamas mažesnis kiekis parametrų, reikalingų apibūdinti prognozuojamos apkrovos dinamiką, tokių būdu padidintas naudojamo modelio tikslumas lyginant su klasikinių AR modeliu (žr. 1.4.1 pav.) [16].

Savaitės diena	Slenkstinis AR modelis		AR modelis	
	Maksimumo prognozė	Pavalandinė prognozė	Maksimumo prognozė	Pavalandinė prognozė
	Absoliutinė paklaida (MW)	Absoliutinė paklaida (MW)	Absoliutinė paklaida (MW)	Absoliutinė paklaida (MW)
Pirmadienis	192.85	208.83	482.19	348.22
Antradienis	112.32	202.46	206.36	307.18
Trečiadienis	104.75	192.66	399.58	336.52
Ketvirtadienis	75.18	176.11	372.36	310.26
Penktadienis	80.79	178.60	467.35	362.10
Vidutinė vertė	113.18	191.73	385.57	332.86

1.4.1 pav. Slenkstinio autoregresinio modelio ir paprasto autoregresnio modelio prognozavimo tikslumo palyginimas [16]

Ummuhan Basaran Filik ir Mehmet Kurban 2007-ais metais publikuotame straipsnyje realizavo trumpojo laikotarpio prognozavimo modelius: autoregresinį modelį, prognozavimo modelį, paremtą dirbtiniais neuroniniais tinklais ir hibridinį modelį, kuriame buvo naudojami dirbtinio neuroninio tinklo ir autoregresiniai modeliai, sujungti į vieną prognozavimo modelį [17]. Iš visų pasiūlytų modelių prasčiausiai pasirodė AR modelis (žr. 1.4.2 pav). AR modelis prastai prognozavo

dėl to, jog procesas buvo netiesinis ir jame pasitaikė periodiškumas, kurį autoregresiniams modeliams sunku prognozuoti dėl atsirandančios inercijos modelyje. Autoregresniai modeliai yra tiesiniai ir bet kokie netiesiškumai pasireiškiantys procesuose arba staigus duomenų pokytis sumažina modelio prognozavimo tikslumą [17]. Nagrinėjamuose procesuose netiesiškumai ir staigūs duomenų pokyčiai pasireiškia esant nestandartinėms situacijoms – savaitgaliai, šventinės dienos ir sistemų gedimai.

Modelis	MSE (GW)
AR	39.862
ANN su BP	26.776
ANN su kaskadiniu BP	21.975
Pasūlytas modelis su Bp	20.078
Pasiūtas modelis su kaskadiniu BP	18.085

1.4.2 pav. Skirtingų prognozavimo modelių prognozavimo paklaidos [17]

Autoregresiniai metodai yra vieni iš pirmųjų suvartojamos elektros energijos prognozavimo modelių. Jie buvo plačiai naudojami dėl savo paprastumo, tačiau vis mažiau yra naudojami dėl savo didelių prognozavimo paklaidų. Šie metodai vis rečiau yra naudojami kaip atskiri prognozavimo metodai ir dažniau yra naudojami, kaip papildomas ryšio tarp kintamųjų paieškos metodai, sudarant tikslesnius prognozavimo modelius. AR prognozavimo modeliai gali būti naudojami aptarto objekto suvartojamai elektros energijai prognozuoti, tačiau norint sudaryti AR prognozavimo modelį reikia turėti atitinkamą duomenų kiekį, pagal kurį būtų galima nustatyti nežinomus modelio parametrus. Taip pat, kaip ir regresiniuose modeliuose, autoregresniuose modeliuose nėra savikorekcijos algoritmo, kuris leistų koreguoti modelio parametrus, tačiau priešingai, nei regresiniuose modeliuose, autoregresiniuose modeliuose nepasikliaujama išoriniais faktoriais ir remiamasi tik paties objekto energijos suvartojimo istoriniais duomenimis. Pastaroji modelio savybė leidžia geriau prognozuoti periodišką žmonių veiklą, tačiau būtent ši modelio savybė į modelį įneša tam tikrą prognozavimo inertiškumą [17], kuris mažina modelio prognozės tikslumą, esant staigiems objekto suvartojimo pokyčiams. Bendruoju atveju autoregresinius modelius susidedančius tik iš AR dedamosios galima naudoti aptarto objekto suvartojamos elektros energijos prognozei sudaryti.

1.5. Autoregresinis prognozavimo modelis su judančiu vidurkiu

Autoregresinis prognozavimo modelis su judančiu vidurkiu, sutrumpintai vadinamas ARMA (angl. *autoregressive moving average*), tai modelis, kurio principas yra paremtas autoregresiniu metodu. ARMA modelis yra jungtinis modelis, sudarytas iš autoregresines dalies, kuri yra pažymima AR, ir judančio vidurkio dalies, kuri yra žymima MA. Kadangi autoregresinė modelio dedamoji jau yra apžvelgta, šioje dalyje yra apžvelgiama ARMA modelio judančio vidurkio dedamoji MA. Judančio vidurkio modelis – tai laiko funkcija, kuri apibūdina autoregresinio modelio trikdžių ε pokytį. Judančio vidurkio matematinė q-tosios eilės išraiška yra pateikiama formulėje žemiau (žr. 1.5.1 formulę).

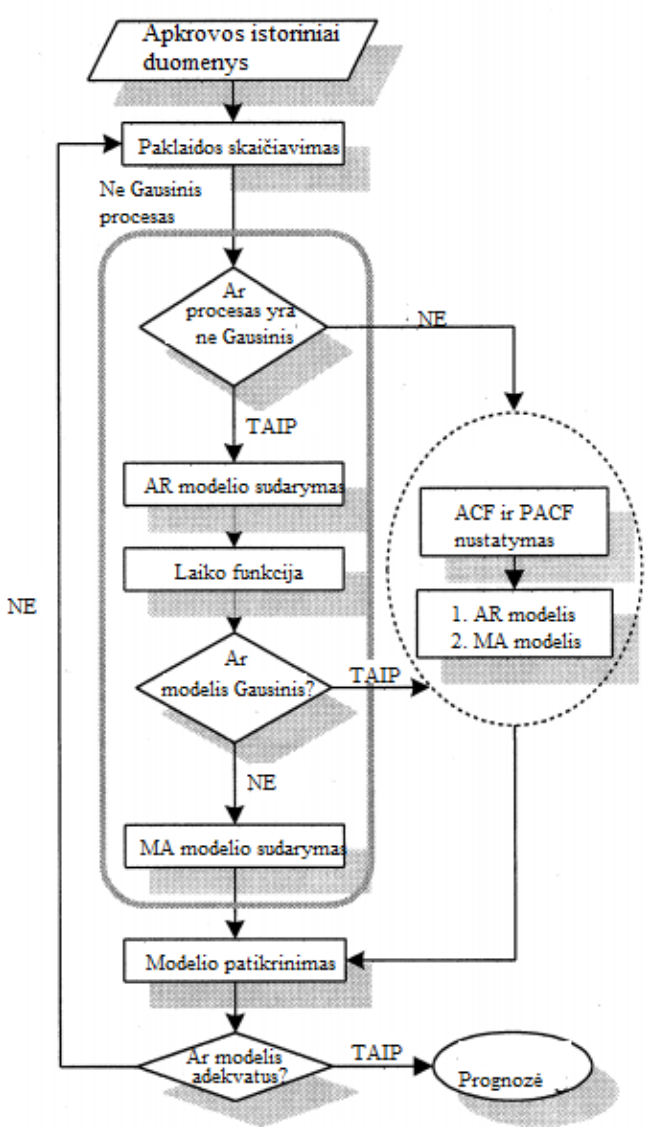
$$X_t = \varepsilon_t + \sum_{i=1}^{q} \theta_i * \varepsilon_{t-i}\,; \quad (1.5.1)$$

čia, θ_i atitinka MA dedamosios parametrus, ε_t – tai modelį t laiko momentu veikiantys trikdžiai. ε_{t-i} tai pavėlinti sistemos trikdžiai, kurie veikė MA modelį praeitais laiko momentais.

Autoregresinis judančio vidurkio prognozavimo modelis ARMA turi dualias savybes, kurios yra apjungtos viename modelyje. Modelio prognozuojamas suvartojamas elektros energijos kiekis laiko momentu *t* priklauso ir nuo prieš tai buvusiu suvartoto elektros energijos kiekio ir nuo prieš tai buvusių trikdžių verčių. ARMA modelyje naudojamos buvusios prognozių ir trikdžių vertės, leidžia išlaikyti sistemos prognozavimo kryptį. Didėjant modelio eilei *t,* prognozuojamas suvartojamas elektros energijos kiekis tampa stabilesnis. Taip pat ARMA modelis leidžia tiksliau prognozuoti suvartojamos energijos minimumus nei naudojant įprastinį AR modelį [16]. Modelio prognozė mažiau iškraipoma dėl į modelį patekančių trikdžių, nes yra priimta, kad bendras objekto suvarotjamas energijos kiekis susidaro iš dviejų dedamųjų – autoregresinės dedamosios, už kurią atsako modelio AR dedamoji, ir triukšmo dedamoji, už kurią atsako MA dedamoji. Sudarant ARMA modelį, priimama prielaida, kad modelyje esantis triukšmas nėra atsitiktinis ir taip pat gali būti prognozuojamas, nes tarp suvartojamos elektros energijos triukšmo yra tarpusavio ryšys išreiškiamas autokoreliacija, kurią apibūdina MA dedamoji. Bendroji ARMA modelio matematinė išraiška yra pateikiama formulėje žemiau (žr. 1.5.2 formulę) [7, 18].

$$X_t = c + \sum_{i=1}^{N} \varphi_i * X_{t-i} + \varepsilon_t + \sum_{i=1}^{q} \theta_i * \varepsilon_{t-i}; \quad (1.5.2)$$

ARMA prognozavimo modelis, kaip ir kiti autoregresiniai modeliai, mažiau naudojamas šiais laikais atsiradus dirbtinio intelekto sistemoms. Tačiau autoregresiniai modeliai gali būti tikslesni už dirbtinio intelekto modelius, jeigu objektas, kuriam yra atliekama prognozė, suvartoja arba paskirsto energiją MW eilės ir apie kurio veikimą turima daug duomenų [18]. S. J. Huang ir K. R. Shih 2003 m. apublikuotame straipsnyje pasiūlė patobulintą ARMA prognozavimo modelį [18]. Pasiūlytas modelis papildė ARMA modelį algoritmu, kuris leido atskirti gausinį ir ne gausinį procesą (žr. 1.5.1 pav.).



1.5.1 pav. Pasiūlyto ARMA modelio papildymo algoritmas [18]

Papildytas modelis yra tikslesnis už paprastą ARMA ir dirbtiniais neuroniniais tinklais paremtus prognozavimo modelius. Nemodifikuoto ARMA modelio prognozavimo paklaida – 1,67%, dirbtiniais neuroniniais tinklais paremto modelio prognozės paklaida – 2,2% ir patobulinto ARMA modelio prognozavimo paklaida – 1,62% [18].

Pagrindinis skirtumas tarp AR ir ARMA prognozavimo modelių – tai priimta prielaida, kad nagrinėjamame procese triukšmas turi tarpusavio ryšį. Šio triukšmo tarpusavio ryšys yra išreiškiamas MA dedamosios pavidalu [16]. Bendruoju atveju, MA dedamoji leidžia tiksliau prognozuoti proceso minimalias vertes, nei AR modeliai, kurie tiksliau prognozuoja proceso maksimalias vertes, todėl ARMA modeliai geriau prognozuoja prognozės išėjimą iš minimalios vertės, kas padidina modelio jautrumą ir sumažina jo inertiškumą. Kitais aspektais AR ir ARMA modeliai yra panašūs, todėl aptartam objektui prognozavimo modelio naudojimas yra patartinas ir yra tinkamesnis už AR modelį. ARMA modelis turi mažesnį inertiškumą, nei AR modelis, tačiau, kaip ir AR modelyje, ARMA modelyje prognozė yra sudaroma, remiantis vien tik proceso istoriniais duomenimis. Taip pat kaip ir kituose autoregresiniuose modeliuose, ARMA modelyje nėra numatytas modelio koeficiento verčių

atnaujinimo algoritmas. Dėl šios priežasties, norint sudaryti savikoreaguojanti ARMA modelį, būtina naudoti papildomus autokerekcijos metodus.

1.6. Autoregresinis integralinis prognozavimo modelis su slenkančiu vidurkiu

Autoregresinis integralinis prognozavimo modelis su slenkančiu vidurkiu – tai apibendrintas ARMA prognozavimo modelis, turintis integruojančią dalį „I“, kuri yra išreiškiama per skirtumo operatorių (angl. difference operator) (žr. 17 formulę) [7]. ARIMA modelis yra naudojamas elektros apkrovos prognozei sudaryti tuomet, kai naudojami duomenys atitinka nestacionarų autoregresinį modelį. Tam, kad pašalinti autoregresinio modelio nestacionarumą, yra pritaikomas skirtumo operatorius. Jeigu skirtuminis operatorius yra lygus 0, tuomet ARIMA modelis tampa ARMA modeliu. ARMA modelio matematinė išraiška yra pateikta formulėje žemiau (žr. 1.6.1 formulę), tuomet *t* laiko momentu autoregresinio modelio pirmos eilės skirtumas yra pateiktas formulėje žemiau (žr. 1.6.2 formulę) [7].

$$X'_t = X_t - X_{t-1}; \tag{1.6.1}$$

Pirmos eilės skirtumo operatorių galima išreikšti matematine išraiška:

$$X'_t = X_t - X_{t-1} = X_t - L * X_t = (1 - L) * X_t; \tag{1.6.2}$$

Gautoje išraiškoje skirtumo operatorius yra *L* (žr. 1.6.2 formulė). Tuomet ARIMA modelio matematinė išraiška yra jungtinis modelis, sudarytas iš 1.6.2, 1.5.2 ir 1.4.1 formulės atitinka ARIMA modelio dalis I, MA ir AR. Bendroji ARIMA modelio išraiška yra pateikiama formulėje žemiau (žr. 1.6.3 formulę).

$$\left(1 - \sum_{i=1}^{p} \phi_i * L^i\right) * (1 - L)^d * X_t = \left(1 + \sum_{i=1}^{q} \theta_i * L^i\right) * \varepsilon_t; \tag{1.6.3}$$

čia ARIMA modelio matematinėje išraiškoje $\boldsymbol{\phi}$ *atitinka modelio autoregresinės dalies parametrus,* $\boldsymbol{\theta}$ *tai aprašyto modelio judančio vidurkio (MA) dedamosios parametrai.* $\boldsymbol{p}$ *atitinka autoregresinės dedamosios eilę,* $\boldsymbol{q}$ *– tai judančio vidurkio dedamosios eilė, o* $\boldsymbol{d}$ *atitinka integruojančios dedamosios eilę.*

ARIMA prognozavimo modelis pakeitė savalaikius regresinius modelius, nes pastarieji yra tikslesni už regresinius modelius ir geriau realizuoja netiesinį ryšį tarp suvartojamos elektros energijos duomenų [19]. I. Moghram ir S. Rahrnan 1989 m. apublikavo straipsnį, kuriame atliko penkių trumpojo laikotarpio prognozavimo modelių tyrimą, kuriame siekė nustatyti tiksliausią modelį. Nustatyta, kad autoregresinis integruojantis modelis su judančiu vidurkiu yra vidutiniškai 2,56% tikslesnis už klasikinį regresinį prognozavimo modelį [19]. Straipsnyje ištirti prognozavimo metodai: regresinis (angl. Multiple Linear Regression), stochastinės laiko sekos (angl. Stochastic Time Series), prie kurių priskiriamas ARIMA modelis, bendras eksponentinio išlyginimo modelis (angl. General Exponential Smoothing), sistemos būsenos modelis (angl. State Space Method), žiniomis paremtos sistemos prognozavimo modelis (angl. Knowledge-Based Approach) [19]. Pagal pristatytus rezultatus straipsnyje, mažiausia prognozės paklaida yra gauta, naudojant ARIMA prognozavimo modelį [19].

Pagrindinis ARIMA prognozavimo modelio trūkumas yra jo reiklumas duomenims. Norint realizuoti šį modelį reikia nemažo kiekio istorinių duomenų. Jeigu nagrinėjamas procesas nėra tiesinis, turi sezoninių ir ciklinių svyravimų, reikia tokio duomenų kiekio, kuris apimtų šiuos svyravimus. ARIMA modelis yra labiau taikomas trumpalaikėms prognozėms sudaryti, nei ilgalaikėms, šis teiginys taip pat atsiremia į tai, jog norint sudaryti modelį būtinas atitinkamas kiekis duomenų gautas tam tikru dažnumu. Lyginant ARIMA prognozavimo modelį su eksponentinio išlyginimo prognozavimo modeliu, standartiniame ARIMA modelyje nėra numatytas automatinis modelio atnaujinimas.

Realizuotas ARIMA modelis liks toks pat ir nepakitęs, nepriklausomai nuo to, kaip pakis prognozuojamas procesas, kai eksponentinio išlyginimo modelyje modelis automatiškai keičiasi, kintant pačiam procesui [19]. Pakitus stebimam procesui, ARIMA modelis turi būti atnaujintas, kitaip jo atliekamas prognozavimas bus nekorektiškas. Šis autoregresinis modelis yra tinkamas prognozuoti aptarto objekto suvartojamos elektros energijos kiekį jeigu bus naudojamas papildomas metodas kuris leistų atnaujinti ARIMA modelio parametrų vertę.

ARIMA (1,1,0) modelis – tai modelis, kuris apibūdina procesą, naudojant pirmos eilės autoregresinę modelio dedamąją AR(1) ir integruojančią pirmos eilės (pirmos eilės skirtumas) dedamąją I(1). Šis ARIMA modelis yra analogiškas pirmos eilės AR(1) modeliui, tačiau skirtingai nuo AR modelio, pastarasis yra sudarytas, remiantis turimų duomenų skirtumams, o ne originalioms suvartoto elektros energijos kiekio vertėmis.

ARIMA (0,1,1) procesas – tai autoregresinis prognozavimo modelis, kuriame nėra autoregresinės dedamosios, $p = 0$. Pagal bendruosius autoregresinių modelio principus, modeliui su MA(q) q-aja eile daro įtaka q kartu pavėlinti triukšmai, tačiau AR(1) modelio prognozei visos praeities dedamosios daro įtaką, bet tik naujausi duomenys daro didžiausią įtaką modelio prognozei. Todėl ARIMA prognozavimo modeliai yra labiausiai tinkami atlikti prognozę trumpuoju laikotarpiu.

ARIMA prognozavimo modelis – tai modifikuotas ARMA modelis, todėl jame yra tokie patys privalumai ir trūkumai, kokie yra ir ARMA modelyje. Autoregresinio modelio integruojanti dedamoji į modelį yra įterpiama tuomet, kai nagrinėjamas procesas nėra stacionarus ir neįmanoma sudaryti ARMA modelio. Bendruoju atveju, standartinio ARIMA modelio pritaikymas aptartam objektui yra yra, nors jame nėra numatytas modelio koeficientų atnaujinimas, tačiau šis modelis yra tikslesnis už kitus regresinius ir stochastinius prognozavimo modelius [19]. Norint panaudoti ARIMA modelį aptarto objekto suvartojimui prognozuoti, būtina naudoti modelio koeficientų verčių atnaujinimo algoritmus, kurie palaikys modelio prognozavimo tikslumą, taip pat sumažins modelio duomenų kiekio poreikį.

1.7. Autoregresniai prognozavimo modeliai su išoriniais kintamaisiais

ARMAX apkrovos prognozavimo modelis – tai autoregresinis prognozavimo modelis su judančiu vidurkiu ir išoriniais kintamaisiais. Modelis yra sudarytas iš trijų dalių – autoregresinės dalies AR, kuri yra aprašyta aukščiau esančiame skyriuje, judančios dedamosios dalies MA, kuri taip pat yra aprašyta aukščiau esančiame skyriuje, bei iš papildomų išorinių kintamųjų. Kiekvienas autoregresinis modelis AR, ARMA, ARIMA turi savo atitinkantį modelį su išoriniais kintamaisiais, kurie yra išreiškiami kaip ARX, ARMAX, ARIMAX. Kitaip sakant, ARMAX modelis – tai ARMA modelis su papildomais išoriniais kintamaisiais. ARMAX modelio matematinė išraiška yra pateikta formulėje žemiau (žr. 1.7.1 formulę) [20].

$$X_t = \varepsilon_t + \sum_{i=1}^{p} \varphi_i * X_{t-1} + \sum_{i=1}^{q} \theta_i * \varepsilon_{t-i} + \sum_{i=1}^{b} \eta_i * d_{t-i}; \quad (1.7.1)$$

čia modelio parametrai yra analogiški 18 – oje formulėje (žr. 18 formulę) pateiktiems parametrams. $\mathbf{d_{t-i}}$ *– tai išorinis kintamasis, naudojamas modelyje,* $\boldsymbol{\eta_i}$ *– tai išorinio kintamojo parametrai.* ***b*** *atitinka išorinio kintamojo eilę, kuri nurodo, kiek žingsnių atgal yra vertinamas parametras, sudarant objekto apkrovos prognozę* ***t*** *laiko momentu.*

ARMAX prognozavimo modelis naudojamas, kai yra papildomi išoriniai kintamieji, kurie gali padidinti įprastinio ARMA prognozavimo modelio tikslumą. Tačiau šiame modelyje, kaip ir ARMA modelyje, nagrinėjamo objekto suvartojamos elektros energijos procesas privalo būti stacionarus, t.y.

proceso vidurkis ir standartinis nuokrypis turi nekisti bėgant laikui. Jeigu objekto procesas yra nestacionarus, tuomet būtina naudoti ARIMAX modelį suvartojamos elektros energijos prognozei. Pagrindinis šio modelio realizacijos sudėtingumas yra parametrų apskaičiavimas. H. T. Yang ir kt. 1996 m. publikuotame darbe pasiūlė ARMAX modelio parametrų paieškos metodą, naudojant evoliucinį programavimą, kai modelis atliko prognozavimo kas valanda nuo vienos dienos iki savaitės į priekį [20]. Sudarytas ARMAX prognozavimo modelis atliko prognozavimą tiksliau, nei komerciškai naudojamas SAS paketas [20].

ARMAX modelis skirtas prognozuoti suvartojamos elektros energijos kiekį yra tikslesnis, nei standartinis ARMA modelis. J. Y. Fan ir J. D. McDonald 1994 m. apublikavo straipsnį, kuriame realizavo realaus laiko trumpojo laikotarpio elektros apkrovos prognozavimo modelį, skirtą elektros paskirstymo sistemoms [16]. Modelis atliko prognozę remiantis orų prognozės parametrais [16]. ARMA modelis atvaizdavo laiko funkciją su netiesinių ryšiu tarp elektros apkrovos ir orų prognozės, o modelio parametrai buvo atnaujinami realiu laiku, naudojant rekursyvinį mažiausių kvadratų algoritmą (angl. Weighted Recursive Least Squares algorithm) [16]. Realizuotas ARMAX modelis yra tikslus su vidutine absoliutine paklaida (angl. Mean absolute error) iki 2%, atliekant prognozę 24 valandoms į priekį arba iki 2,5%, sudarant prognozę 168 valandas į priekį [16]. Pasiūlytame prognozavimo modelyje buvo realizuoti du esminiai modelio komponentai, kurie leido pasiekti tokią minimalią prognozavimo paklaidą. Pirmasis modelio elementas – modelio prognozės priklausomybė nuo išorinių veiksnių, oro temperatūros. Šis elementas modelį pakeitė iš ARMA į ARMAX modelį, tokiu būdu modelio prognozė tapo priklausoma tiek nuo proceso istorinių duomenų, tiek nuo išorinių veiksnių, kurie, be abejones, daro įtaką suvartojamas elektros energijos kiekiui. Šis elemtas eliminuoja iš modelio prognozavimo inertiškumą, kuris atsiranda modelyje tuomet, kai prognozė yra sudaroma, remiantis tik proceso istoriniais duomenimis. Antras pasiūlyto modelio elementas – nuolatinis modelio parametrų verčių atnaujinimas, šis modelio elementas – tai modelio savikorekcijos algoritmas, kuris užtikrina modelio prisitaikymą prie pakitusio proceso. Tokiu būdu sudarant modelį, nėra būtinas didelis kiekis duomenų, užtenka sudaryti pradinę modelio versija, kuri pasiekia tam tikrą prognozavimo tikslumą, bėgant laikui, dėl savikorekcijos algoritmo, modelis labiau turėtų prisitaikyti prie proceso ir, atitinkamai, jo prognozavimo tikslumas turi padidėti [16].

Kaip ir kiti autoregresiniai modeliai, šiame poskyryje aprašytas modelis taipogi nusileidžia tikslumu dirbtinio intelekto modeliams, bet yra naudojamas tuomet, kai nėra didelis duomenų kiekis apie stebimą procesą. Lyginant su standartiniais ARMA ir ARIMA modeliais, ARMAX ir ARIMAX yra labiau tinkami prognozavimo modeliai aptartam objektui, dėl modelyje esančio išorinių kintamųjų poveikio įvertinimo. Kaip jau buvo minėta, ARMAX ir ARIMAX modeliuose prognozės nebe priklauso tik nuo istorinių proceso duomenų, o priklauso ir nuo išorinių veiksnių. Šis modelių išorinių poveikių įvertinimas sumažina modelių inertiškumą ir leidžia greičiau reaguoti į drastiškai pakitusias aplinkos sąlygas, kurios iššaukia staigius suvartojamos elektros energijos pokyčius. Bet kaip ir kituose autoregresiniuose modeliuose, taip ir ARMAX ir ARIMAX modeliuose nėra išspręstas klausimas apie modelio koeficientų atnaujinimo procedūrą, todėl, siekiant atnaujinti modelio koeficientus, būtina naudoti papildomus metodus.

1.8. Eksponentinio išlyginimo prognozavimo modelis

Eksponentinis išlyginimas (angl. Exponential Smoothing) – tai statistinis metodas, kuris elektros energetikoje ir automatikoje yra naudojamas prognozuoti būsimą elektros energijos suvartojimą. Šiame prognozavimo modelyje būsimas elektros energijos suvartojimas yra sudaromas, remiantis

istorinių duomenų eksponentiškai apdorotu (angl. weighted) vidurkiu. Modelio vertės išlyginamos, naudojant išlyginimo koeficientą α. Paprasto eksponentinio išlyginimo (angl. Simple exponential smoothing) vertė S_i laiko momentu i yra gaunama, naudojant formulę, kuri yra pateikta žemiau (žr. 1.8.1 ir 1.8.2 formules) [21].

$$S_{i+1} = S_i + \alpha * (x_i - S_i); \quad (1.8.1)$$

Aukščiau pateikta formulė gali būti išreikšta kitokiu pavidalu (žr. 1.8.2 formulę).

$$S_i = \alpha * x_i + (1 - \alpha) * S_{i-1}; \quad (1.8.2)$$

čia $\boldsymbol{S_i}$ *atitinka išlygintą modelio vertę* ***i*** *–uoju laiko momentu.* $\boldsymbol{0<\alpha<1}$ *atitinka išlyginimo koeficientą.* $\boldsymbol{x_i}$ *sistemos suvartojama elektros energija* ***i*** *–uoju laiko momentu.* $\boldsymbol{S_{i-1}}$ *– tai išlyginta sistemos vertė* $\boldsymbol{i-1}$ *laiko momentu.*

Tuomet bendruoju atveju kiekviena nauja išlyginta proceso vertė yra apskaičiuojama, kaip apdorotas vidurkis iš išlygintų istorinių verčių sumos. Todėl i – uoju laiko momentu išlyginta proceso vertė - tai apdorotas prieš tai buvusių verčių vidurkis. Bendruoju atveju sistemą t laiko mementu galima išreikšti (žr. 1.8.3 -1.8.6 formules):

$$S_{i-1} = \alpha * x_{i-1} + (1 - \alpha) * S_{i-2}; \quad (1.8.3)$$

$$S_{i-2} = \alpha * x_{i-2} + (1 - \alpha) * S_{i-3}; \quad (1.8.4)$$

$$S_{i-3} = \alpha * x_{i-3} + (1 - \alpha) * S_{i-4}; \quad (1.8.5)$$

…

$$S_i = \alpha * x_i + \alpha * (1 - \alpha) * x_{i-1} + \alpha * (1 - \alpha)^2 * x_{i-2} \ldots + (1 - \alpha)^t * S_0; \quad (1.8.6)$$

Tuomet visų išlyginimo koeficientų sumą galima išreikšti viena bendra formule (žr. 1.8.7 formulę) [21].

$$\alpha + \alpha * (1 - \alpha) + \alpha * (1 - \alpha)^2 \ldots \alpha * (1 - \alpha)^{t-1} + (1 - \alpha)^t == \alpha * \left[\frac{1-(1-\alpha)^t}{1-(1-\alpha)}\right] + \quad (1.8.7)$$

$$+(1 - \alpha)^t;$$

Eksponentinio išlyginimo modelyje prognozuojama vertė $i+1$ laiko momentu priklauso nuo pačio nagrinėjamo proceso formos ir nuo išlygintų proceso verčių, tam tikrais laiko momentais. Žemiau yra pateikiami keli pavyzdžiai, priklausantys nuo nagrinėjamo proceso formos (žr. 1.8.8 – 1.8.16 formules).

- Nagrinėjamas procesas yra tiesinis [21]:

$$\hat{X}_{i+1}(x) = a_i + b_i * x; \quad (1.8.8)$$

Tuomet tiesinio proceso koeficientai yra apskaičiuojami pagal formules [21]:

$$a_i = 2 * S_i^{(1)} - S_i^{(2)}; \quad (1.8.9)$$

$$b_i = \frac{\alpha*(S_i^{(1)} - S_i^{(2)})}{1-\alpha}; \quad (1.8.10)$$

- Nagrinėjamo proceso forma yra kvadratinė:

$$\hat{X}_{i+1}(x) = a_i + b_i * x + c_i * x^2; \quad (1.8.11)$$

Tuomet tiesinio proceso koeficientai yra apskaičiuojami pagal formules [21]:

$$a_i = 3 * S_i^{(1)} - 3 * S_i^{(2)} + S_t^{(3)}; \quad (1.8.12)$$

$$b_i = \frac{\alpha}{2*(1-\alpha)^2} * [(6 - 5\alpha)S_i^{(1)} - 2(5 - 4\alpha)S_i^{(2)} + (4 - 3\alpha)S_i^{(3)}]; \quad (1.8.13)$$

$$S_i^{(1)} = \alpha * x_i + (1 - \alpha) * S_{i-1}^{(1)}; \quad (1.8.14)$$

$$S_i^{(2)} = \alpha S_i^{(1)} + (1-\alpha) * S_{i-1}^{(2)}; \quad (1.8.15)$$

$$S_i^{(3)} = \alpha S_i^{(2)} + (1-\alpha) * S_{i-1}^{(3)}; \quad (1.8.16)$$

Išlyginimo koeficientas turi didelę įtaką prognozuojamai vertei. Pastarasis koeficientas gali būti parinktas atsitiktinai, tačiau dažniausiai šis parametras yra parenkamas pagal tai, kad jo vertė sąlygotų minimalią prognozavimo modelio prognozės paklaidą. Naudojami modelio paklaidos įvertinimo metodai yra pasirenkami laisvai ir priklauso nuo modelio projektuotojo, tačiau dažniausiai yra naudojami – MSE, RMSE, MAE, MAPE paklaidos įvertinimo metodai.

Eksponentinio išlyginimo prognozavimo modelis yra naudojamas dažniausiai trumpalaikių prognozių sudaryme, taip pat gali būti naudojamas prognozei sudaryti vidutinės bei ilgo laiko intervaluose [21]. James W. Taylor 2003 metais apublikuotame straipsnyje pasiūlė trumpalaikį elektros apkrovos prognozavimo modelį su dvigubu sezoniniu eksponentiniu išlyginimu (angl. Double Seasonal Exponential Smoothing), kurį palygino su įprastiniu eksponentiniu išlyginimu ir dvigubo sezoniškumo ARIMA modeliu. Aprašytas modelis pasirodė geriau nei kiti du metodai [22]. Tas pats autorius 2006 metais apublikavo kita straipsnį, kuriame atliko to paties dvigubo sezoniškumo eksponentinio išlyginimo modelio palyginimą su dvigubo sezoniškumo ARIMA modeliu, neuroninio tinklo modeliu ir regresiniu modeliu [11]. Pasiūlytas modelis pasirodė gėriau nei kiti palyginimo modeliai [23].

Eksponentinio išlyginimo prognozavimo modelis yra priskiriamas prie statistinių prognozavimo modelių. Kaip ir autoregresiniuose prognozavimo modeliuose, taip ir šiame modelyje, ateities suvartojamas elektros energijos kiekis yra sudaromas priklausomai nuo istorinių duomenų, dėl to modelyje yra inertiškumas, kuris trukdo modeliui staigiai reaguoti į proceso pokytį. Eksponentinio išlyginimo prognozavimo modelyje nenumatytas automatinis koeficientų verčių atnaujinimas, taip pat pastarajame modelyje koeficientų apskaičiavimas yra paprastesnis nei autoregresiniuose prognozavimo modeliuose. Šis prognozavimo modelis gali būti naudojamas aptarto objekto prognozėms sudaryti, tačiau jo reakcijos laikas į drastišką proceso pokytį bus didesnis už atoregresinius modelius su išoriniais kintamaisiais.

1.9. Neraiškios logikos regliatorius kaip prognozavimo modelio adaptivioji dalis

Aptarti prognozavimo modeliai nepasižymi adaptyviomis savybėmis, kurios leistų keisti modelio parametrus staigiai pakitus energijos suvartojimui. Autoregresiniai prognozavimo modeliai į staigius prognozuojamo dydžio pokyčius reaguoja lėtai, priklausomai nuo to kokios AR eilės yra prognozavimo modelis. Norint sudaryti adaptyvų individualaus pastato suvartojamos elektros energijos prognozavimo modelį siūloma naudoti neraiškios logikos reguliatorių. Pastarasis reguliatorius gali padidinti pasirinkto prognozavimo modelio prognozės tikslumą atnaujindamas modelio parametrus [24,25].

Neraiškios logikos reguliatorius – tai priemonė, kuri leidžia atlikti sistemos valdymą arba derinimą, naudojant lingvistinius kintamuosius, pritaikant ekspertines žmogaus žinias, nesudarant tikslios valdymo funkcijos [26]. Kadangi žmogus mąsto abstrakčiais reiškiniais, kuriuos sunku aprašyti logiškai, norint pritaikyti žmogaus ekspertines žinias, yra naudojami lingvistiniai kintamieji. Pastarieji kintamieji suriša žmogaus ekspertines žinias su valdymo sistema per neraiškiąja logiką. Sudaryta sistema – Fuzzy reguliatorius, kuris interpretuodamas išorinius valdomo objekto kintamuosius, naudojant neraiškiąją logiką ir lingvistinius kintamuosius, priima

sprendimus kaip objektą valdytų žmogus ekspertas ir tokiu būdu atlieka objekto valdymą arba derinimą. Šiame baigiamajame darbe bus apžvelgiami Mamdani algoritmas ir Sugeno – Takagi algoritmas, kurie yra naudojami neraiškios logikos reguliatoriams sudaryti [27].

H. Mamdani – anglų matematikas, kuris 1975 metais sukūrė neraiškios logikos reguliatoriaus realizacijos algoritmą [28]. Pagrindiniai Mamdani algoritmo elementai, sudarant neraiškios logikos reguliatorių [27]:

1. Atrenkami reguliatoriaus išėjimo ir įėjimo kintamieji. Išoriniai kintamieji, tai kintamieji, kurie yra naudojami objektui stebėti ir jį valdyti
2. Atrenkami lingvistiniai kintamieji, tai kintamieji, kurie atspindi terminus, kuriais vadovaujasi žmogus ekspertas valdydamas arba derindamas stebimą objektą
3. Lingvistinių kintamųjų priklausomumo funkcijų aprašymas.
4. Neraiškios logikos žinių bazės sudarymas
5. Fuzifikacija
6. Sudėtinių sąlygų sudarymas ir jų teisingumo radimas
7. Implikacija. Nustatomos lingvistinių kintamųjų verčių priklausomybės nuo sudarytų žinių bazės taisyklių tenkinimo laipsnio
8. Agregacija. Sudaromos Fuzzy reguliatoriaus išėjimų agreguota priklausomumo funkcija
9. Defuzifikavimas, pagal lingvistines išėjimo vertes nustatoma išėjimo skaitinė vertė

Sugeno ir Takagi neraiškios logikos reguliatoriaus sudarymo algoritmas skiriasi nuo Mamdani algortimo savo žinių bazės forma, taip pat Sugeno ir Takagi algoritme nėra Mamdani algoritmo 7, 8 ir 9 žingsnių. Sugeno ir Takagi algoritmo žinių bazėje naudojamų taisyklių forma yra pateikiama žemiau (žr. 1.9.1 formulę) [27].

$$IF\ S_1(x_1)\ AND\ S_2(x_2)\ THEN\ z_j = \alpha_0 + \alpha_1 * x_1 + \alpha_2 * x_2; \quad (1.9.1)$$

čia $S_i(x_i)$ – tai sąlygos, kurias tenkina nefuzifikuoto kintamojo skaitinė vertė. Sąlyga nurodo, kuriam neraiškiam lygmeniui priklauso pastarasis x_i kintamasis. α_i - tai vertė, kurią parenka žmogus ekspertas, pagal kurį yra sudaromas neraiškios logikos reguliatorius.

Taip pat Sugeno ir Takagi algoritme po sudėtinių sąlygų sudarymo ir jų teisingumo radimo žingsnio yra atliekama defuzifikacija. Defuzifikacijai yra naudojamas svertinio vidurkio metodas (žr. 37 formulę) arba yra naudojamas svertinės sumos metodas (žr. 1.9.2 ir 1.9.3 formulę) [27].

$$z_0 = \frac{t_1 z_1 + t_2 z_2}{t_1 + t_2}; \quad (1.9.2)$$

$$z_0 = t_1 z_1 + t_2 z_2; \quad (1.9.3)$$

čia z_0 – bendra neraiškaus reguliatoriaus išėjimo vertė. t_j - reguliatoriaus išėjimo teisingumo vertė

Pagrindiniai skirtumai tarp Mamdani ir Sugeno – Takagi algoritmų yra tai, kad Sugeno algoritme neraiškios logikos reguliatoriaus išėjimas yra apskaičiuojamas pagal griežtas matematines formules (žr. 1.9.2 ir 1.9.3 formules) ir nėra naudojamos priklausomumo funkcijos ir defuzifikavimas, kaip tai yra daroma Mamdani algoritme [29]. Taip pat Sugeno – Takagi algoritmų paremtas neraiškios logikos reguliatorius yra naudojamas MISO (angl. Multiple Input Single Output) sistemoms, kai

Mamdani algoritmu paremtas reguliatorius naudojamas MIMO (angl. Multiple Input Multiple Output) sistemoms. Sugeno – Takagi algoritmas yra optimizuotas kompiuteriniam skaičiavimui, tačiau Mamdani algoritmas yra labiau pritaikytas išreikšti žmogaus ekspertines žinias nei tai yra realizuota Sugeno algoritme, kai žinios yra perkeliamos į skaitmeninę aplinką [29].

Sudarant neraiškios logikos reguliatorių pagal Mamdani algoritmą, svarbiausias etapas – parinkti teisingus reguliatoriaus įėjimo ir išėjimo kintamuosius, kuriems vėliau bus pritaikyti lingvistiniai kintamieji. Kintamųjų pasirinkimas priklauso nuo užduoties ir nuo reguliatoriaus autoriaus, tačiau dažniausiai kaip įėjimo kintamasis yra pasirenkama valdomo dydžio paklaida nuo užduotos arba norimos vertės [24,28,30]. O kaip reguliatoriaus išėjimo kintamieji yra pasirenkamos reguliuojamo objekto parametrų kitimo vertės, nuo kurių priklauso stebimo dydžio vertė [30,31,32].

Lingvistiniai kintamieji (angl. term) skiriasi nuo skaitinių kintamųjų. Neraiškios logikos reguliatoriuose, lingvistinių kintamųjų vertės yra ne skaičiai, o jų natūralios vertės, kurios yra išreiškiamos žodžiais arba jų kombinacijomis [33]. Pastarųjų kintamųjų reikšmės aprašomos neraiškiais skaičiais arba neraiškiomis aibėmis [33]. Lingvistiniai kintamieji priklauso nuo sprendžiamo uždavimo ir nuo Fuzzy reguliatoriaus autoriaus.

Lingvistinių kintamųjų funkcijos (angl. membership function) suriša naudojamą įėjimo arba išėjimo kintamąjį su pasirinktais lingvistiniais kintamaisiais. Dažniausiai naudojamos priklausomumo funkcijos yra: *trimf* (žr. 1.9.4 formulę), *trapmf* (žr. 1.9.5 formulę), *gbellmf* (žr. 1.9.6 formulę)*ir gaussmf* (žr. 1.9.7 formulę).

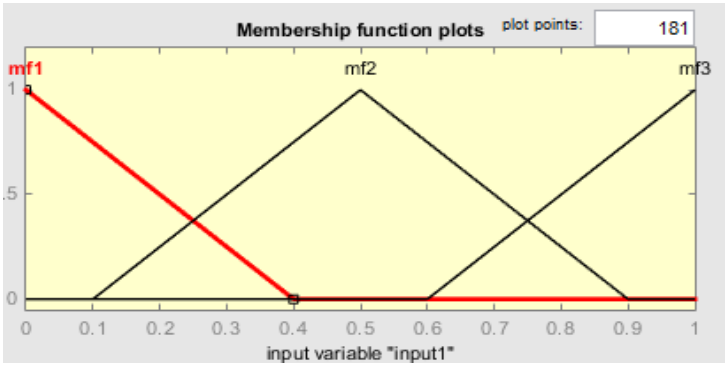
$$trimf(x,a,b,c) = max(min(\frac{x-a}{b-a},\frac{c-x}{c-b},0)); \quad (1.9.4)$$

$$trapmf(x;a,b,c,d) = \left(\left(\frac{x-a}{b-a},1,\frac{d-x}{d-c},0\right)\right); \quad (1.9.5)$$

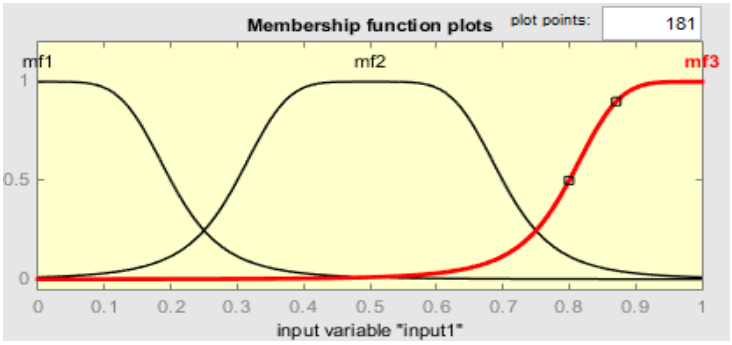
$$gbellmf(x;a,b,c,d) = \frac{1}{1-\frac{|x-c|^{2b}}{a}}; \quad (1.9.6)$$

$$gaussmf(x;\sigma,c) = e^{\frac{-(x-c)^2}{2\sigma^2}}; \quad (1.9.7)$$

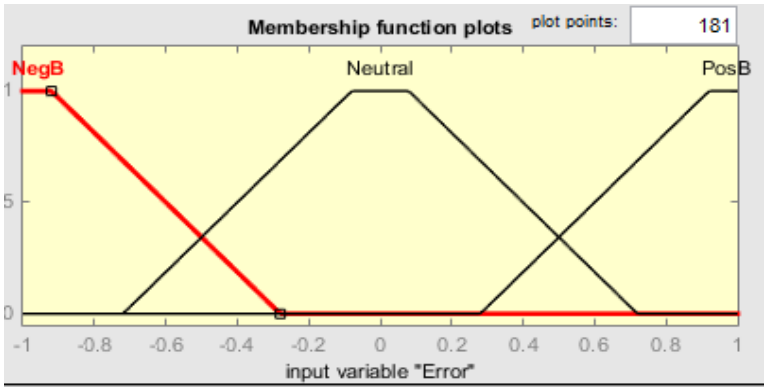
Dažniausiai naudojamos lingvistinių kintamųjų priklausomumo funkcijos - *trimf, trapmf, gbellmf* ir *gaussmf* grafiškai atvaizduojamos žemiau pateikiamuose paveikslėliuose (žr. 1.9.1, 1.9.2, 1.9.3 ir 1.9.4 pav.).



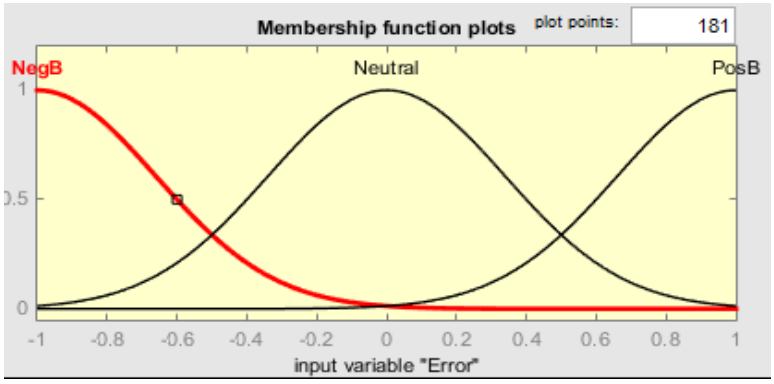
1.9.1 pav. Neraiškios logikos reguliatoriaus įėjimo kintamojo lingvistinių kintamųjų priklausomumo funkcijos *trimf* grafinis atvaizdavimas



1.9.2 pav. Neraiškios logikos reguliatoriaus įėjimo kintamojo lingvistinių kintamųjų priklausomumo funkcijos *gbellmf* grafinis atvaizdavimas



1.9.3 pav. Neraiškios logikos reguliatoriaus įėjimo kintamojo lingvistinių kintamųjų priklausomumo funkcijos *trapmf* grafinis atvaizdavimas

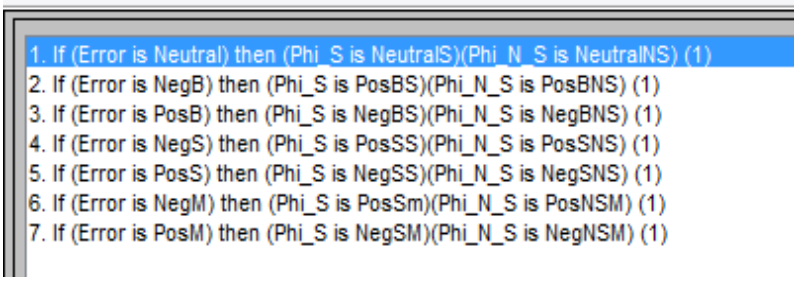


1.9.4 pav. Neraiškios logikos reguliatoriaus įėjimo kintamojo lingvistinių kintamųjų priklausomumo funkcijos *gaussmf* grafinis atvaizdavimas

Neraiškios logikos reguliatoriaus išėjimo kintamųjų lingvistinės vertės yra nustatomos pagal įėjimų kintamųjų lingvistines vertes, kai pastarosios yra surišamos, tarpusavyje naudojant lingvistines taisykles [27]. Kaip ir Mamdani algoritme (žr. 1.9.8 formulę), taip ir Sugeno ir Takagi algoritme (žr. 1.2.1 formulę), naudojamos lingvistinės taisyklės – tai *IF...THEN...* tipo [27]. Pastarųjų taisyklių visuma sudaro žinių bazę, kurioje taisyklių skaičius priklauso nuo naudojamų kintamųjų skaičiaus ir jų neraiškiųjų lygmenų. Žinių bazėje yra naudojami lingvistinių kintamųjų neraiškieji lygmenys, o ne jų skaitinės vertės (žr. 1.9.5 pav.).

$$IF\ S_1(x_1)\ AND\ S_2(x_2)\ THEN\ z_j = y_j\ ; \quad (1.9.8)$$

čia $S_i(x_i)$ – tai tenkinama lingvistinių kintamųjų sąlyga. Sąlyga nurodo, kokiam neraiškumo lygmeniui priklauso pastarasis x_i kintamasis. y_i – lingvistinis išėjimo kintamasis



1.9.5 pav. Fuzzy reguliatoriaus žinių bazės pavyzdys

Fuzifikacija – tai veiksmas, kurio metu raiškus arba neraiškus skaičius yra pakeičiamas į lingvistinį kintamąjį [33].

Sudėtinių sąlygų sudarymas ir jų teisingumo radimas. Lingvistinio kintamojo vertė, gauta pagal konkrečią taisyklę, atitinka tos taisyklės sąlygos teisingumą [27]. Jeigu tenkinamos kelios sąlygos ir tik iš dalies, tuomet sudėtinių sąlygų teisingumas yra randamas pagal metodus:

- minimalios vertės metodas (žr. 1.9.9 formulę),
- sandaugos metodas (žr. 1.9.10 formulę),
- maksimalios vertės metodas (žr. 1.9.11 formulę).

$$t = min\{m_{S1}(x_1); m_{S2}(x_2); \dots m_{Sn}(x_n)\}; \quad (1.9.9)$$
$$t = m_{S1}(x_1) m_{S2}(x_2) \dots m_{Sn}(x_n); \quad (1.9.10)$$
$$t = max\{m_{S1}(x_1); m_{S2}(x_2); \dots m_{Sn}(x_n)\}; \quad (1.9.11)$$

t - tai sudėtinės sąlygos teisingumas, m_{SI} – priklausomumo funkcijos

Implikacija – tai veiksmas, kuris pagal nustatytą tenkinamų sudėtinių sąlygų teisingumą pakeičia lingvistinių kintamųjų priklausomumo funkcijas ir gaunamos aktyvuotos priklausomumo funkcijos [27]. Naudojami implikacijos metodai [27]:

- minimumo (min) metodas (žr. 1.9.12 formulę),
- sandaugos (prod) metodas (žr. 1.9.13 formulę).

$$\hat{m}(y) = min\{t, m(y)\}; \quad (1.9.12)$$
$$\hat{m}(y) = t * m(y); \quad (1.9.13)$$

$\hat{m}(y)$ – tai implikuota, pakeista lingvistinio kintamojo priklausomumo funkcija, t – sudėtinės sąlygos tinkamumo vertė, m(y) – lingvistinio kintamojo priklausomumo funkcija

Agregacija – tai veiksmas, kurio metu, naudojant atitinkamus metodus, yra sudaroma viena bendra priklausomumo funkcija, apjungianti visas gautas aktyvuotas priklausomumo funkcijas. Naudojami agregacijos metodai:

- maksimumo metodas (žr. 1.9.14 formulę),
- sumavimo metodas (žr. 1.9.15 formulę),

- probor metodas (žr. 1.9.16 formulę).

$$m(y) = max\{m_{S1}(y); m_{S2}(y); \dots m_{Sn}(y)\}; \quad (1.9.14)$$
$$m(y) = m_{S1}(y) + m_{S2}(y) + \cdots + m_{Sn}(y); \quad (1.9.15)$$
$$m(y) = m_{S1}(y) + m_{S2}(y) - m_{S1}(y) * m_{S2}(y); \quad (1.9.16)$$

Defuzifikacija – tai procedūra, kurios metu, naudojant specialius metodus, lingvistinis kintamasis pagal bendrą priklausomumo funkciją, gautą agregacijos būdų, sudaromas raiškus, paprastas, skaičius. Dažniausiai naudojami defuzifikavimo metodai Mamdani algoritme:

- svorio centro metodas (žr. 1.9.17 formulę),
- ploto centro metodas (žr. 1.9.18 formulę),
- maksimumų vidurkio metodas (žr. 1.9.19 formulę),
- maksimumų didžiausios vertės metodas (žr. 1.9.20 formulę).

$$y_0 = \frac{\int_{y_{max}}^{y_{min}} y * m(y)dy}{\int_{y_{max}}^{y_{min}} m(y)dy}; \quad (1.9.17)$$
$$\int_{y_{min}}^{y_0} m(y)dy = \int_{y_0}^{y_{max}} m(y)dy; \quad (1.9.18)$$
$$y_0 = mean\{y: m(y) = max\{m(y)\}\}; \quad (1.9.19)$$
$$y_0 = max\{y: m(y) = max\{m(y)\}\}; \quad (1.9.20)$$

Neraiškios logikos reguliatorius yra naudojamas, kaip prognozavimo modelių komponentas, siekiant padidinti pastarųjų modelių prognozavimo tikslumą. Neraiškios logikos reguliatoriai gali būti naudojami modeliuose, kurie prognozuoja stojančiųjų skaičių į universitetą, akcijų kainų pokytį, lauko temperatūros svyravimus [31]. F. M. Tseng ir G. H. Tzeng 2001 – ais metais pristatytame straipsnyje pasiūlė neraiškios logikos ARIMA modelį, prognozuojant kaip keičiasi užsienio valiutos rinka [24]. Jų realizuotam autoregresiniam ARIMA prognozavimo modeliui, neraiškios logikos modelis buvo naudojamas kaip įrankis, atnaujinti parametrų vertes. Prognozavimo modelis neprognozavo konkrečių užsienio valiutos rinkos kainos verčių, tačiau buvo naudojamas prognozuoti valiutos kainos kitimo ribas. Sudarytos prognozės, naudojant pasiūlytą

ARIMA modelį ir neraiškios logikos modelį, beveik atitiko tikrąją užsienio valiutos kainą (žr. 1.9.6 pav.).

Data	Tikroji vertė	Fuzzy ARIMA modelio apatinė riba	Fuzzy ARIMA modelio viršutinė riba
5-Sep	27.54	27.53	27.55
6-Sep	27.55	27.52	27.55
7-Sep	27.54	27.53	27.55
9-Sep	27.55	27.53	27.56
10-Sep	27.54	27.53	27.56
11-Sep	27.55	27.53	27.55
12-Sep	27.54	27.53	27.55
13-Sep	27.54	27.53	27.55
14-Sep	27.54	27.53	27.55
16-Sep	27.55	27.53	27.55

1.9.6 pav. F. M. Tseng ir G. H. Tzeng 2001 – ais metais pasiūlyto Fuzzy Arima modelio valiutos kainos prognozuojamos ribos [24]

Taip pat, lyginant pasiūlytą prognozavimo modelį su standartinių ARIMA modeliu, pasiūlyto modelio patikimumo intervalas (angl. *condence interval*) yra 95% siauresnis nei standartinio ARIMA modelio [24].

F. M. Tseng ir G. H. Tzeng 2002 – ais metais pateikė antrą straipsnį, kuriame realizavo prognozavimo modelį ARIMA su sezonine dedamąja ir kuriame naudojo neraiškios logikos modelį parametrų atnaujinimui. Pastarajame straipsnyje buvo prognozuojama Taivanio mašinų pramonės gamybos apimtis ir gaiviųjų gėrimų pardavimo apimtis [25]. Abiem atvejais pasiūlytas prognozavimo modelis (FSARIMA) buvo lyginamas su standartinių ARIMA modeliu, turinčiu sezoninę dedamąją (SARIMA) [25]. Lyginant SARIMA ir FSARIMA modelių gaiviųjų gėrimų pardavimo prognozę, FSARIMA modelio pardavimų prognozuojama viršutinė ir apatinė ribos yra artimesnės tikrajai pardavimų reikšmei, nei SARIMA modelio prognozės apatinė ir viršutinė riba (žr. 1.9.7 pav.).

Data	Tikroji vertė	SARIMA $(1,1,0)(0,1,1)_{12}$		FSARIMA	
		95% CI Apatinė riba	95% CI Viršutinė riba	Apatinė riba	Viršutinė riba
Jan-75	52	39.12	55.68	35.09	91.62
Feb-75	60	49.32	60.54	39.54	106.38
Mar-75	66	55.502	71.99	42.60	126.94
Apr-75	80	69.002	83.26	50.74	157.46
May-75	85	69.59	100.55	51.62	172.64
Jun-75	95	76.19	95.73	57.60	201.46
Jul-75	100	74.39	107.78	58.69	218.15
Aug-75	104	81.74	113.94	61.86	240.64
Sep-75	101	61.84	117.78	58.99	242.01
Oct-75	94	58.56	116.666	54.86	235.31
Nov-75	81	32.38	108.636	45.68	205.55
Dec-75	70	42.68	99.36	38.10	179.00

1.9.7 pav. F. M. Tseng ir G. H. Tzeng 2002 – ais metais pasiūlyto FSARIMA modelio gaiviųjų gėrimų prognozuojamo pardavimo ribos [25]

Kaip ir pirmajame straipsnyje taip ir antrajame straipsnyje pasiūlytas prognozavimo modelis yra geresnis nei standartinis prognozavimo modelis [25]. Išanalizuoti straipsniai parodė, kad neraiškios logikos reguliatorius gali būti naudojamas,kaip įrankis atnaujinti prognozavimo modelių parametrus, siekiant išgauti didesnį prognozavimo tikslumą.

2. Metodinė dalis

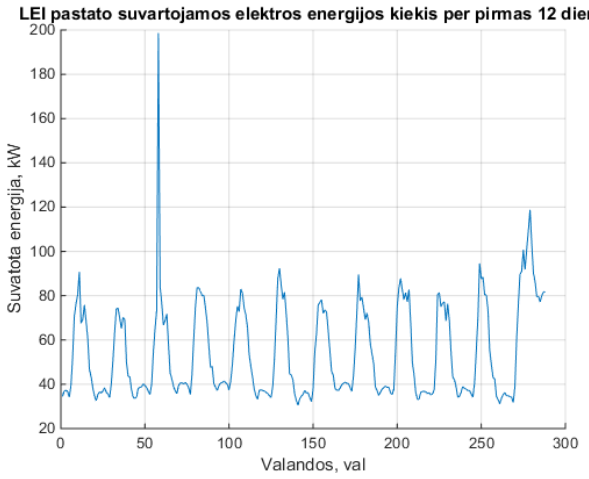
2.1. Individualaus mažai žinomo pastato apžvalga

Šiame baigiamajame darbe buvo sudaryti individualaus pastato adaptyvus suvartojamos elektros energijos prognozavimo modeliai. Kaip nagrinėjamas objektas, buvo pasirinktas individualus pastatas, esantis Kauno mieste, Lietuvoje (žr. 2.1.1 pav.). Pastatas priklauso Lietuvos energetikos institutui, adresu - Breslaujos gatvė 3, Kaunas 44403. Gauti pastato suvartojamos elektros energijos duomenys už laikotarpį nuo 2017-05-18 iki 2018-04-23, suvartojamas elektros energijos kiekis buvo registruojamas fiksuotais laiko intervalais – kas valandą (žr. 2.1.1 lentelę, 2.1.2 pav.). Lentelėje pateiki duomenys yra kW eilės.

2.1.1 lentelė. Nagrinėjamo individualaus pastato suvartojamos elektros energijos kiekis

	2017-05-18	**2017-05-19**	**2017-05-20**	**2017-05-21**	**2017-05-22**	**...**	**2018-04-23**
01:00	34,5	37	38,2	37	33	...	40,2
02:00	37	38,4	37,6	37	32,8	...	40,8
03:00	37,2	36,4	37,2	37,8	32,8	...	40
04:00	36,8	35,6	35,6	35,6	31,6	...	38,4
05:00	34,2	34	33,8	32,6	29,8	...	35,4
06:00	39,2	40,2	33,6	32,8	37,6	...	44,6
07:00	52	51	33,4	32,6	52	...	60,2
...	...	...	...	...	...	...	...
17:00	46,6	43,4	34,2	34,2	51	...	47,6
18:00	43,4	37,4	34	38	49,8	...	48
19:00	38,4	34,2	34	34	41,8	...	40,2
20:00	34,8	33,6	33,8	29,8	37,2	...	38,6
21:00	32,6	34,2	33,8	29	35,6	...	37,2
22:00	35,4	38,2	36,8	32,6	40,2	...	39,4
23:00	36,6	38,6	36,6	32,6	40,2	...	40,6
00:00	36	38,8	37,4	33,2	40,2	...	40,8

2.1.1 pav. Nagrinėjamo objekto, Lietuvos energetikos instituto pastato nuotrauka



2.1.2 pav. LEI pastato suvartotos elektros energijos kiekio grafikas per pirmąsias 12 darbo dienų nuo 2017 05 18 dienos

Prognozavimo modeliai šiame darbe buvo sudaryti tik darbo dienoms, nes, sudarant prognozavimo modelius visai kalendorinei savaitei, įskaitant ir savaitgalius, modelis tampa chaotiškas. Didesnis prognozavimo tikslumas yra pasiekiamas tuomet, kai kalendorinei savaitei yra sudaromi du atskiri prognozavimo modeliai: modelis darbo dienoms, modelis savaitgaliams ir šventinėms dienoms. Šiame darbe yra sukoncentruotas dėmesys į prognozavimo modelį, skirtą prognozuoti suvartojamą elektros energiją darbo dienomis.

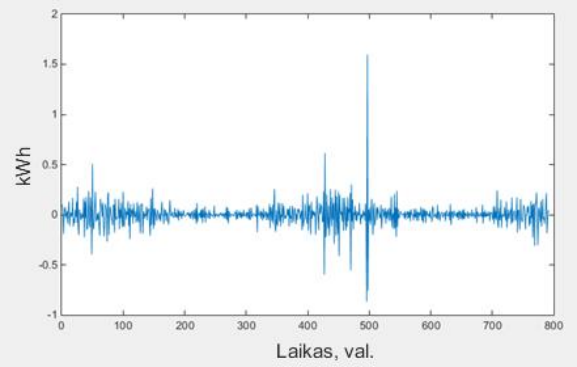
Sudarant autoregresinius ir statistinius prognozavimo modelius, buvo naudojami tik istoriniai suvartojamos elektros energijos duomenys. Siekiant išgauti didesnį prognozavimo modelio tikslumą buvo atliekami pastarųjų duomenų apdorojimai [34].

2.2. Autoregresinių prognozavimo modelių identifikavimo ir realizacijos metodai

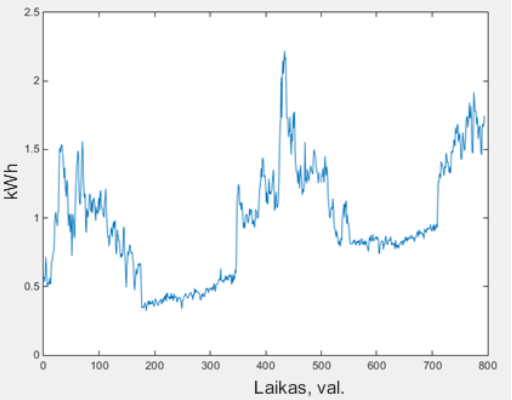
Šiame baigiamajame darbe realizuoti autoregresiniai prognozavimo modeliai. Pastarųjų prognozavimo modelių realizacija buvo atliekama pagal E. P. Box ir M. Jenkins pasiūlytą metodologiją [34]. Autoregresnių modelių realizacija susideda iš kelių pagrindinių žingsnių, kuriuos būtina įvykdyti, norint realizuoti autoregresinį prognozavimo modelį. Sudarant autoregresinį prognozavimo modelį, svarbiausia yra nustatyti modelio struktūrą. Išanalizavus nagrinėjamo objekto duomenis, energijos suvartojimo procesas aprašomas, naudojant AR, MA, ARMA, ARIMA autoregresinius modelius. Sudarant prognozavimo modelį, tikslas – sudaryti tokį modelį, kuris turės mažiausiai parametrų, bet tenktų ir tikslumo kriterijus, o prognozavimo paklaida turėtų neviršyti nustatytos paklaidos ribos. Pagrindiniai modelio sudarymo žingsniai:

- Modelio duomenų vizualinė analizė. Pirmasis ir pagrindinis autoregresinio prognozavimo modelio sudarymo žingsnis – vizualiai atvaizduoti nagrinėjamo proceso duomenis (žr. 2.2.1 pav.). Pagal gautą duomenų grafiką galima iš karto pasakyti, ar procesas yra linijinis (žr. 2.2.1 pav.), turintis trendo liniją, ir kokia yra trendo linija – teigiama (žr. 2.2.2 pav.) ar neigiama. Taip pat galima nustatyti, ar procesas turi sezoniškumų (žr. 2.2.3 pav.), ciklišku mų. Be abejo, vizualinė proceso duomenų analizė neleidžia pasakyti, kokios eilės turi būti modelis arba

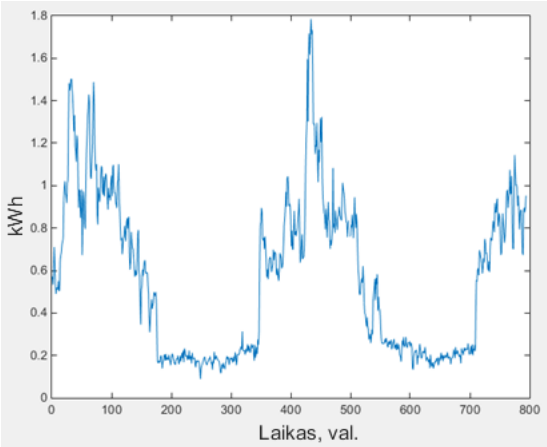
kokios yra jo parametrų vertės, tačiau leidžia nustatyti, kokio maždaug tipo modelis labiausiai atitiks analizuojamus duomenis.



2.2.1 pav. Linijinis procesas



2.2.2 pav. Procesas, turintis teigiamą trendo liniją



2.2.3 pav. Procesas, turintis išreikštą sezoniškumą

- Nagrinėjamo proceso stacionarumo nustatymas. Sudarant autoregresinį prognozavimo modelį, ypač svarbu, ar nagrinėjamas procesas yra stacionarus ar ne. Priklausomai nuo to yra pasirenkami skirtingi metodai, realizuojant prognozavimo modelį. Kaip jau buvo minėta ankščiau teorinėje dalyje, procesas yra stacionarus tuomet, kai jo vidutinė vertė (angl. *mean*), dispersija (angl. *variance*) ir kovariacija (angl. *covariance*) $i + m$ etape yra pastovi ir nekinta, kintant vėlinimo žingsniams. Stacionarumą taip pat galima patikrinti, naudojant papildytą *Dickey-Fuller* testą (angl. *Augmented Dickey–Fuller test*). Šio testo esmė – patvirtinti arba paneigti nulio hipotezę, nagrinėjamo proceso atžvilgiu. Jeigu nulio hipotezė yra paneigiama, tuomet procesas yra stacionarus, jeigu nulio hipotezė yra patvirtinama, tuomet procesas yra nestacionarus [35], plačiau apie *Dickey-Fuller* testo taikymą, parenkant prognozavimo modelius, yra pateikta 17-ame šaltinyje.

  Jeigu nagrinėjamas procesas yra nestacionarus, tuomet negalima sudaryti autoregresinio prognozavimo modelio. Norint sudaryti autoregresnį prognozavimo modelį, būtina apdoroti duomenis taip, kad procesas taptų stacionarus. Yra naudojami skirtingi metodai, leidžiantys apdoroti proceso duomenis ir paversti procesą į stacionarų, pagrindiniai iš jų yra :

- Trendo linijos eliminavimas (angl. *detrending*), duomenų apdorojimas, kurio metu iš proceso duomenų yra pašalinama dedamoji, kuri yra atsakinga už trendo liniją duomenyse.

- Skirtuminis perskaičiavimas (angl. *defferencing*). Šio apdorojimo metu yra perskaičiuojamas proceso duomenų skirtumas, kuris vėliau yra naudojamas kaip proceso duomenys ir, remiantis jais, sudaromas proceso modelis. Skirtuminis skaičiavimas pašalina iš duomenų sezoniškumą. Panaudojus skirtuminį perskaičiavimą, autoregresinis modelis, kuris apibūdins procesą, įgis integralinę dedamąją. Jeigu vėliau atlikti modelio struktūros nustatymo metodai nurodys, kad procesas turi būti aprašytas, naudojant *ARMA* modelį, automatiškai bus naudojamas *ARIMA* modelis. Skirtuminio perskaičiavimo eilė yra lygi modelio integruojančios dedamosios eilei.

- Eksponentinis apdorojimas – tai nagrinėjamo proceso duomenų apdorojimas tuomet, kai vietoj originalių proceso duomenų *X,* yra naudojamas šių duomenų dešimtainis logaritmas *log(X)*. Šis duomenų apdorojimo metodas iš proceso duomenų pašalina kintančia dispersiją.

Atliekant šiuos duomenų apdorojimus, siekiama gauti tokią proceso išraišką, kuri yra pavaizduota aukščiau pateiktose formulėse (žr. 1.4.1 ir 1.4.3 formules). Atlikus bet kokį duomenų apdorojimą, būtina pakartoti papildytą *Dickey-Fuller* testą. Nagrinėjamo proceso duomenys yra apdorojami tol, kol yra gaunamas stacionarus procesas. Apdorojant duomenis, galima maišyti apdorojimo metodus tarpusavyje.

Esant stacionariam nagrinėjamam procesui, galima taikyti sekantį modelio sudarymo žingsnį – nustatyti modelio tipą (AR, MA, ARMA, ARIMA) ir jo vidinę struktūrą.

- Nagrinėjamo proceso struktūros nustatymas. Pirmas žingsnis, nustatant modelio struktūrą - nustatyti modelio suvartojamos elektros energijos autokoreliaciją prie skirtingų žingsnių atgal, tam yra naudojama autokoreliacijos funkcija ACF (angl. *sample autocorrelation function*) (žr. 2.2.1, 2.2.2, 2.2.3 formules).

$$ACF(h) = \hat{\rho} = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)}; \quad (2.2.1)$$

$$ACVF(h) = \hat{\gamma}(h) = \frac{1}{n} * \sum_{t=1}^{n-h}(x_{t+h} - \bar{x}) * (x_t - \bar{x}); \quad (2.2.2)$$

$$\bar{x} = \frac{1}{n} * \sum_{t=1}^{n} x_t; \quad (2.2.3)$$

čia ***h=0, 1, ..., n-1*** *atitinka vėlinimo žingsnių skaičių,* ***n*** *atitinka visą duomenų kiekį, kuris yra skirtas modeliui sudaryti.* ***ACFV*** *atitinka autokovariacijos funkciją (angl. sample autocovariance function), kuri yra reikalinga, norint apskaičiuoti autokoreliacijos funkciją.* $\bar{\boldsymbol{x}}$ *atitinka naudojamų duomenų vidurkį modelio sudarymui.*

Jeigu nagrinėjamo objekto suvartojamos elektros energijos procesas yra visiškai atsitiktinis procesas ir jo duomenys neturi jokios tarpusavio priklausomybės, tuomet *ACF(h)* funkcijos vertė turi būti artima nuliui, esant bet kokiam vėlinimo žingsniui *h*. Jeigu nagrinėjamo proceso duomenys yra tarpusavyje priklausomi, tuomet, esant konkrečiam vėlinimui, proceso autokoreliacijos vertė (žr. 2.2.1 formulę) turi būti didesnė už nulį. Jeigu procesas yra tiesinis, tuomet autokoreliacija, tarp duomenų 0 ir 1 laiko momentu, yra didelė, taip pat ji yra didelė tarp 1 ir 2 laiko momentų ir t.t., dėl to duomenys 0 - uoju ir 2- uoju arba

vėlesniu laiko momentu turėtų turėti tarpusavio ryšį, kuris bus matomas per autokoreliacijos *ACF* vertę. Didėjant vėlinimo žingsniui *h*, autokoreliacija tarp duomenų 0 ir *h* mažėja. Laikoma, kad duomenys neturi tarpusavio tiesinio ryšio tuomet, kai autokoreliacijos vertė yra lygi $\widetilde{N}$ vertei, kurios apskaičiavimas yra išreikštas formulėje žemiau (žr. 2.2.4 formulę). $\widetilde{N}$ vertė priklauso nuo duomenų kiekio, kuris yra naudojamas modeliui sudaryti.

$$\widetilde{N} = \pm\frac{1{,}96}{\sqrt{n}}; \tag{2.2.4}$$

čia ***n*** *atitinka duomenų kiekį, naudojamą autoregresinio modelio sudarymui.*

Dalinė autokoreliacijos funkcija *PACF* (angl. *partial autocorrelation function*) – tai funkcija, kuri yra naudojama, sudarant autoregresinį modelį. Ši funkcija leidžia nustatyti autoregresnio modelio tinkamą vėlinimo žingsnį. Kaip ir *ACF*, *PACF* nurodo autokoreliaciją tarp naudojamų duomenų, esant skirtingam vėlinimo žingsniui, kuo didesnė autokoreliacija, tuo stipresnį ryšį duomenys tarpusavyje turi. Tačiau priešingai, nei *ACF*, *PACF* nurodo tarpusavio ryšį tarp duomenų, nutolusių per 1, 2, ... *n-1* laiko žingsnį, nevertinant autokoreliacijos, esančios tarpinėse grandyse. *PACF* matematine išraiška yra pateikta formulėje žemiau (žr. 2.2.5, 2.2.6 ir 2.2.7 formules).

$$PACF(h) = \begin{cases} 1 & kai\ h = 0 \\ \widehat{\phi_{hh}} & kai\ h \geq 1 \end{cases}; \tag{2.2.5}$$

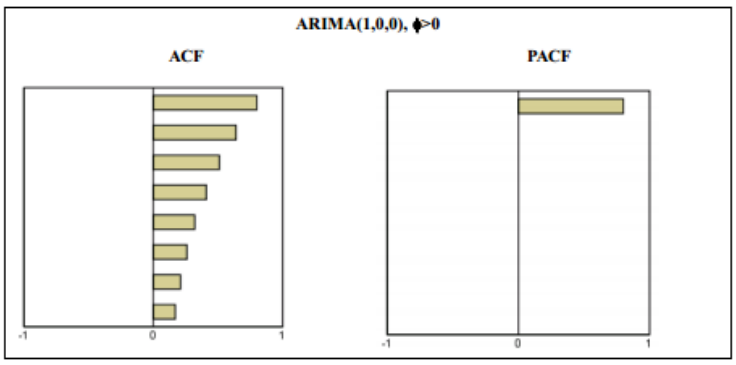
$$\widehat{\Phi_h} = \widehat{\Gamma_h^{-1}}[\hat{\gamma}(1), \hat{\gamma}(2), \dots, \hat{\gamma}(h)]'; \tag{2.2.6}$$

$$\widehat{\Gamma_h} = [\hat{\gamma}(i-j)]_{i,j=1}^{h}; \tag{2.2.7}$$

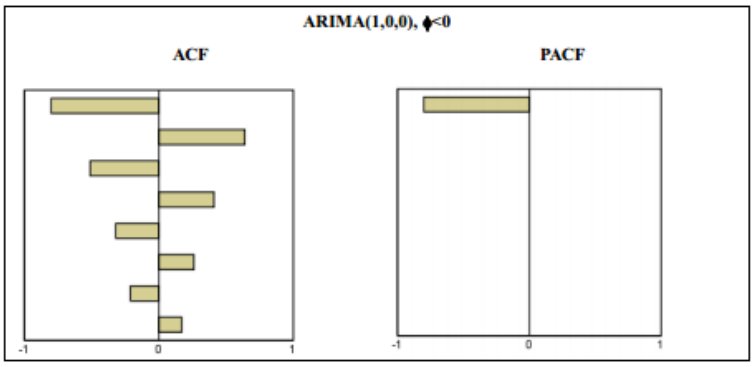
$\boldsymbol{\widehat{\phi_{hh}}}$ - tai koeficientas, kuris apibūdina dalinę autokoreliaciją (žr. 2.2.6 formulę). Šis koeficientas yra lygus paskutiniam koeficientui tiesinėje *h* – osios eilės regresijoje. Šis koeficientas nurodo tiesioginį ryšį tarp x_{t+h} ir x_t duomenų elementų. Kaip *ACF*, taip ir *PACF* kintamieji neturi tarpusavio ryšio tuomet, kai autokoreliacinė vertė yra artima nuliui (žr. 2.2.5 formulę).

Realizuojamo modelio struktūra yra taip pat nustatoma grafiškai. Naudojant *ACF* ir *PACF* grafikus, nustatomos autoregresinio modelio AR(p) ir MA(q) dedamųjų eilės.

- Procesui aprašyti naudojamas AR(p) modelis tuomet, kai proceso duomenų *ACF* grafikas eksponentiškai artėja į nulį, o *PACF* grafikas įgyja nulinę vertę arba yra mažesnis už $\widetilde{N}$ vertę (žr. 2.2.4 ir 2.2.5 pav.). AR modelio koeficientai ϕ yra teigiami tuomet, kai *PACF* grafiko vertės didesnės už $\widetilde{N}$ ir yra teigiamos, ϕ vertės yra neigiamos tuomet, kai *PACF* grafiko vertės yra neigiamos ir *ACF* vertės yra besikeičiančio ženklo.

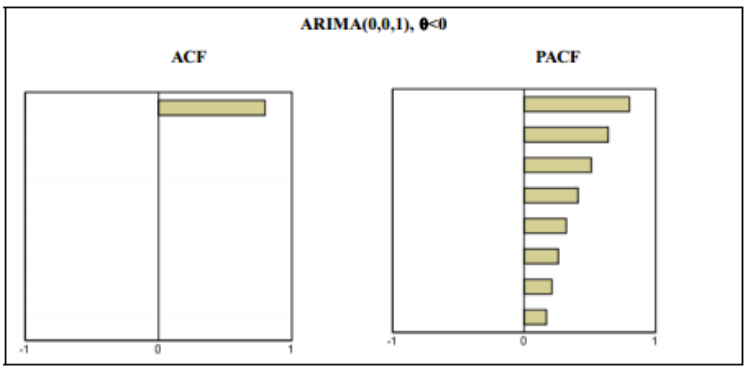


2.2.4 pav. AR(1) modelis, kai ϕ>0

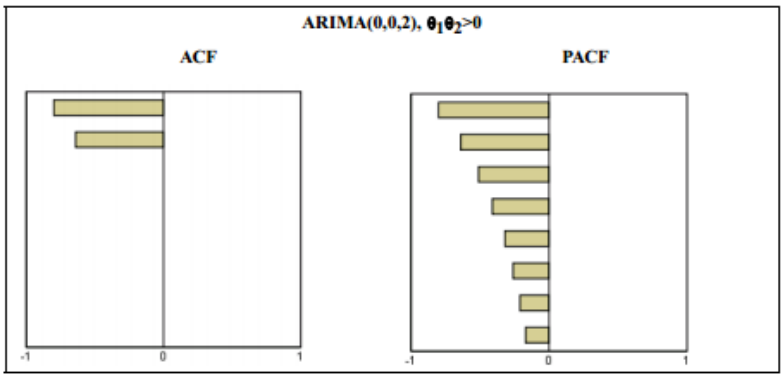


2.2.5 pav. AR(1) modelis, kai ϕ<0

- Procesui aprašyti yra naudojamas MA(q) modelis tuomet, kai proceso duomenų *PACF* grafikas eksponentiškai artėja į nulį, o ACF grafikas įgyja nulinę vertę arba yra mažesnis už $\widetilde{N}$ vertę (žr. 2.2.6 ir 2.2.7 pav.), po *n* vėlinimų. MA modelio koeficientai θ yra teigiami tuomet, kai *PACF* grafiko vertės didesnės už $\widetilde{N}$ ir yra teigiamos, o *ACF* vertės yra neigiamos ir yra mažesnės už $\widetilde{N}$ po *n* vėlinimų (žr. 2.2.6 pav.). MA modelio koeficientai θ yra neigiami tuomet, kai *PACF* grafiko vertės didesnės už $\widetilde{N}$ ir yra neigiamos, o *ACF* vertės yra teigiamos ir yra mažesnės už $\widetilde{N}$ po *n* vėlinimų (žr. 2.2.7 pav.).

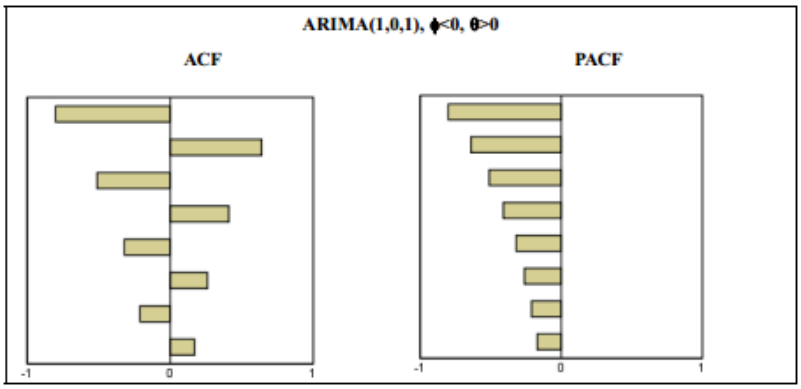


2.2.6 pav. MA(1) modelis kai θ<0



2.2.7 pav. MA(2) modelis kai θ1 ir θ2 > 0

- Procesui aprašyti yra naudojamas ARMA(p,q) modelis tuomet, kai proceso duomenų *PACF* grafikas eksponentiškai artėja į nulį, o *ACF* grafikas taip pat eksponentiškai artėja į nulį. Modelio MA dedamosios koeficiento θ ženklas priešingas *PACF* grafiko vertėms, o AR dedamosios koeficiento ϕ ženklas yra neigiamas tuomet, kai *ACF* grafiko ženklai kinta, keičiantis vėlinimui (žr. 2.2.8 pav.)



2.2.8 pav. ARMA(1,0,1) modelis, kai ϕ<0, θ>0

- Modelio sudarymas, parametrų verčių ieškojimas. Sudarius modelio struktūrą ir nustačius AR, MA ir integruojančios dedamosios eilės yra atliekamas parametrų nustatymas. Parametrų paieškos esmė – surasti tokias ϕ ir θ, kad prognozės nuokrypis nuo tikrųjų verčių būtų minimalus. Parametrams nustatyti yra naudojami metodai – mažiausių kvadratų metodas, Yule ir Walker metodas bei maksimalios tikimybės įvertinimo metodas (angl. *Maximum likehood estimation*).

  - Mažiausių kvadratų metodas naudojamas, ieškant autoregresinio modelio nežinomų parametrų tuomet, kai modelis yra tiesinis. Jeigu nagrinėjamas procesas yra aprašomas kaip AR(p) modelis (žr. 1.4.1 formulę) ir daroma prielaida, kad sistemos nuokrypis ε_t atitinka baltą triukšmą, kai baltas triukšmas $WN(0,\sigma^2)$ yra normaliai pasiskirstęs ir nepriklausomas, tuomet AR(p) modelio parametrai randami yra matricų pavidalu (žr. 2.2.8 – 2.2.11 formules):

$$\boldsymbol{X}_t = (X_t, X_{t-1}, \dots, X_{t-p+1}); \quad (2.2.8)$$

$$\boldsymbol{u}_t = (\varepsilon_t, 0, \dots, 0); \quad (2.2.9)$$

$$\boldsymbol{A} = \begin{pmatrix} \phi_1 \dots & \phi_{p-1} & \phi_p \\ I_1 \dots & I_{p-1} & 0_{(p-1)x1} \end{pmatrix} = \begin{pmatrix} \phi_1 & \phi_2 & \dots & \phi_{p-1} & \phi_p \\ 1 & 0 & & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & & 0 & 1 \end{pmatrix}; \quad (2.2.10)$$

$$\boldsymbol{X}_t = \boldsymbol{A} * \boldsymbol{X}_{t-1} + \boldsymbol{u_t}; \quad (2.2.11)$$

Žinant aukščiau pateiktą išraišką, galima apskaičiuoti nežinomus ϕ parametrų įverčius ir dispersiją σ^2 iš išraiškos pateiktos žemiau (žr. 2.2.12 ir 2.2.13 formules). Pagal žemiau esančia formulę (žr. 2.2.12 formulę), yra surandami parametrų įverčiai, o ne konkrečios parametrų vertės, nes parametrų vertės priklauso nuo naudojamų duomenų kokybės ir kiekio, ir, jeigu procesas yra netiesinis, negalima teigti, kad rasti parametrai yra tikrosios jų vertės. Kai $\hat{\alpha}$ atitinka ieškomų parametrų įverčių matricą.

$$\hat{\boldsymbol{\alpha}} = (\sum_{t=2}^{N} \boldsymbol{X}_{t-1} * \boldsymbol{X}'_{t-1})^{-1} * \sum_{t=2}^{N} \boldsymbol{X}_{t-1} * X_t; \quad (2.2.12)$$

$$\hat{\sigma}^2 = \frac{1}{n-p-1} * \sum_{t=p+2}^{N} (X_t - \hat{\boldsymbol{\alpha}}_1 * X_{t-1} - \hat{\boldsymbol{\alpha}}_2 * X_{t-2} - \hat{\boldsymbol{\alpha}}_p * X_{t-p})^2; \quad (2.2.13)$$

- Maksimalios tikimybės įvertinimo metodas. Šio metodo esmė – nustatyti autoregresinio modelio nežinomus parametrus $\boldsymbol{\varphi}, \boldsymbol{\theta}, \sigma_a$, kurie atitinka maksimalią tikimybės (angl. *likelihood*) funkcijos *LLF()* vertę. Nežinomi parametrai yra surandami iš tikimybės funkcijos ekstremumo kiekvieno parametro atžvilgiu. Autoregresinio modelio maksimalios tikimybės funkcija yra aprašyta žemiau (žr. 2.2.14 formulę). Šiame metode nežinomų parametrų skaičius *M* priklauso nuo naudojamo autoregresinio modelio struktūros (žr. 2.2.16 formulę).

$$LLF(\boldsymbol{\varphi}, \boldsymbol{\theta}, \sigma_a) = -\frac{N}{2} \ln({\sigma_a}^2) - \frac{\boldsymbol{S}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{2\sigma_a^2}; \quad (2.2.14)$$

$$S(\boldsymbol{\varphi}, \boldsymbol{\theta}) = \sum_{i=1}^{N} a_i^2(\boldsymbol{\varphi}, \boldsymbol{\theta}); \quad (2.2.15)$$

$$M = p + q + 1; \quad (2.2.16)$$

Funkcijos $S(\boldsymbol{\varphi}, \boldsymbol{\theta})$ narys a_i^2 priklauso nuo naudojamo autoregresinio modelio struktūros. *LLF()* funkcijos vertė bus maksimali tuomet, kai $\boldsymbol{S}(\boldsymbol{\varphi}, \boldsymbol{\theta})$ funkcijos vertė bus minimali (žr. 2.2.15 formulę). Norint surasti $\boldsymbol{S}$ funkcijos minimumą, yra ieškomas $\boldsymbol{S}$ funkcijos ekstremumas $\boldsymbol{\phi}$ ir $\boldsymbol{\theta}$ koeficientų atžvilgiu. Ekstremumui surasti gali būti naudojami gradiento nusileidimo algoritmas, Levenberg – Marquardt algoritmas ir kiti algoritmai, kurie yra skirti funkcijos eksptremumo paieškai.

- Yule – Walker metodas yra naudojamas autoregresinio proceso parametrams surasti. Kai nagrinėjamas procesas yra išreikštas AR(p) (žr. 1.4.1 formulę), o proceso nuokrypis ε_t yra baltas triukšmas $WN(0,\sigma^2)$, tuomet procesą galima išreikšti forma (žr. 2.2.17 ir 2.2.18 formules):

$$X_t * X_{t-h} = \varphi_1 * (X_{t-1} * X_{t-h}) + \varphi_2 * (X_{t-2} * X_{t-h}) + \dots + \varphi_p * \left(X_{t-p} * X_{t-h}\right) + \varepsilon_t * X_{t-h}; \quad (2.2.17)$$

$$E(X_t, X_{t-h}) = \varphi_1 * E(X_{t-1}, X_{t-h}) + \dots + \varphi_p * E\left(X_{t-p}, X_{t-h}\right) + E(\varepsilon_t, X_{t-h}); \quad (2.2.18)$$

Aukščiau pateiktoje formulėje (žr. 72 formulę), $E(X_t,X_{t-h})$ atitinka matematinę viltį tarp X_t ir X_{t-h}, šiuo atveju matematinė viltis yra kovariacija $\gamma(h)$. Kadangi ε_t neturi ryšio su X_{t-h}, kovariacija tarp šitų dydžių yra *0*, o $E(\varepsilon_t,X_t)=\sigma^2$. Tuomet nagrinėjamo proceso *h* –

osios eilės autokovariacijos funkcija yra išreiškiama kaip (žr. 2.2.19, 2.2.20, 2.2.21 formules):

$$\gamma(h) = \varphi_1 * \gamma(h-1) + \cdots + \varphi_p * \gamma(h-p); \quad (2.2.19)$$

$$\rho(h-1) = \frac{\gamma(h-1)}{\gamma(h)}; \quad (2.2.20)$$

$$\rho(h) = \varphi_1 * \rho(h-1) + \varphi_p * \rho(h-p); \quad (2.2.21)$$

Tuomet Yule – Walker lygtys yra (žr. 2.2.22, 2.2.23 ir 2.2.24 formules):

$$\begin{Bmatrix} \rho(1) = \varphi_1 * \rho(0) + \cdots + \varphi_p * \rho(p-1) \\ \vdots \\ \rho(p) = \varphi_1 * \rho(p-1) + \cdots + \varphi_p * \rho(0) \end{Bmatrix}; \quad (2.2.22)$$

$$\gamma(0) = \varphi_1\gamma(1) + \varphi_p\gamma(p) + \sigma^2; \quad (2.2.23)$$

$$\sigma^2 = \gamma(0) - \varphi_1 * \gamma(1) - \cdots - \varphi_p * \gamma(p); \quad (2.2.24)$$

Yule – Walker metodo lygtyse naudojamas $\rho(h)$ nėra žinomas, todėl jam nustatyti yra naudojami nagrinėjamo proceso istoriniai duomenys. Parametrų įverčiams nustayti yra nadojamos žemiau pateiktos formulės (žr. 2.2.25 – 2.2.29 formules):

$$r(h) = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)}; \quad (2.2.25)$$

$$\hat{\gamma}(h) = \frac{1}{N} * \sum_{t=1}^{N-h}(X_t - \bar{X}) * (X_{t+h} - \bar{X}); \quad (2.2.26)$$

$$\begin{pmatrix} r(1) \\ r(2) \\ \vdots \\ r(p) \end{pmatrix} = \begin{pmatrix} r(0) & r(1) & \cdots & r(p-1) \\ r(1) & r(0) & \cdots & r(p-2) \\ \vdots & \vdots & \ldots & \vdots \\ r(p-1) & r(p-2) & \ldots & r(0) \end{pmatrix} * \begin{pmatrix} \widehat{\varphi_1} \\ \varphi_2 \\ \vdots \\ \varphi_p \end{pmatrix}; \quad (2.2.27)$$

$$\boldsymbol{r_p} = \boldsymbol{R}_p * \widehat{\boldsymbol{\varphi}_p}; \quad (2.2.28)$$

$$\widehat{\boldsymbol{\varphi}_p} = \boldsymbol{R}_p^{-1} * \boldsymbol{r_p}; \quad (2.2.29)$$

Rasti autoregresinio proceso koeficientų įverčiai $\hat{\phi}$ (žr. 2.2.29 formulę) yra naudojami, kaip modelio parametrai, tačiau gali būti perskaičiuoti, jeigu netenkina tikslumo kriterijų. Parametrų vertės gali būti perskaičiuotos, nes nežinomų kintamųjų yra daugiau, nei yra lygčių, todėl parametrai gali turėti daugiau, nei vieną reikšmę.

Visi parametrai aukščiau išvardintuose metoduose yra ieškomi AR(p) modelyje, MA(q) modelyje, taip pat ir ARMA(p,q) modelyje. MA(q) dedamosios dalies parametrai taip pat gali būti nustatyti ir iš ryšio tarp AR ir MA modelio dedamųjų.

- Modelio patikimumo nustatymas. Sudaryto modelio patikimumas yra nustatomas, naudojant tikslumo kriterijus: Aikaike koreguotas informacijos kriterijus *AIC* (angl. *bias-corrected Akaike's Information Criterion*) (žr. 2.2.30 formulę) ir Biaso informacijos kriterijus *BIC* (angl. *Bayesian Information Criterion*) (žr. 2.2.31).

$$AIC = -2 * Log(\psi) + \frac{2dn}{n-d-1}; \quad (2.2.30)$$

$$BIC = -2\log(\psi) + \log(n); \quad (2.2.31)$$

$$\hat{X}_t = \frac{1}{n} * \sum_{k=1}^{t} X_k; \quad (2.2.32)$$

$$\log(\psi) = -\frac{n}{2} * \log(2 * \pi * \sigma^2) - \frac{1}{2} * \log(\Gamma) - \frac{1}{2*\sigma^2} * X' * \Gamma^{-1} * X; \quad (2.2.33)$$

$$d = p + q + 1; \quad (2.2.34)$$

čia $\boldsymbol{n}$ *atitinka naudojamų duomenų kiekį,* $\boldsymbol{d}$ *atitinka ARMA modelio eilę.* $\boldsymbol{log(\psi)}$ *atitinka vėlinimo – panašumo funkciją.* $\boldsymbol{\Gamma}$ *– tai autokovariacijos matrica tarp* $\boldsymbol{X}$ *ir* $\boldsymbol{\widehat{X}}$ *suvartojamos elektros energijos dydžių.*

Naudojant *AIC* kriterijų, pagal Akaike teoremą, modelis yra tinkamesnis naudojamiems duomenims tuomet, kai *AIC* vertė yra mažiausia. Taip pat *AIC* kriterijaus vertė yra bauduojama, padidinus modelio parametrų skaičių. Skirtumas tarp *AIC* ir *BIC* kriterijų yra tas, kad *BIC* kriterijus taiko didesnę kriterijaus vertės baudą tiems modeliams, kurių parametrų skaičius yra didesnis.

Pagal aukščiau pateiktus tinkamumo kriterijus yra atrenkamas modelis, kurio *AIC* ir *BIC* kriterijų vertės yra mažiausios. Tuomet su pasirinktu modeliu yra atliekamas prognozavimas.

- Prognozės sudarymas. Nustačius autoregresinio modelio parametrus ir patikrinus modelio patikimumą, sekančiame žingsnyje yra atliekamas prognozės sudarymas. Jeigu gauta prognozė stipriai skiriasi nuo tikrojo suvartojamos elektros energijos kiekio, būtina perskaičiuoti modelio parametrus arba net pakeisti viso modelio struktūrą.

Autoregresinis modelis yra laikomas sudarytu tuomet, kai yra žinoma modelio vidinė struktūra ir parametrai.

2.3. Autoregresinių prognozavimo modelių ekstremumų paieškos algoritmai

Ieškant autoregresinių prognozavimo modelių parametrų vertės naudojant Maksimalios tikimybės įvertinimo metodą, naudojamos *LLF()* funkcijos vertė bus maksimali tuomet, kai $\boldsymbol{S(\varphi, \theta)}$ funkcijos verė bus minimali. Norint surasti $\boldsymbol{S}$ funkcijos minimumą, yra ieškomas $\boldsymbol{S}$ funkcijos ekstremumas, $\boldsymbol{\phi}$ ir $\boldsymbol{\theta}$ koeficientų atžvilgiu. $\boldsymbol{S(\varphi, \theta)}$ funkcijos ekstremumui surasti šiame darbe yra naudojami algoritmai – gradiento nusileidimo algoritmas (ang. *Gradient Discent*, *GD*), patobulintas gradiento nusileidimo algoritmas ir Levenberg – Marquardt algoritmas.

2.3.1. Gradiento nusileidimo algoritmas

Gradiento nusileidimo algoritmas, kaip klasikinis algoritmas, naudojamas surasti nagrinėjamos funkcijos ekstremumą, tuomet, kai nagrinėjamoje funkcijoje yra daugiau nei vienas koeficientas. Pastarojo algoritmo grubus veikimo principas yra aprašytas žemiau:

1. Pasirenkamos pradinės, nežinomų parametrų $\boldsymbol{\varphi}$ ir $\boldsymbol{\theta}$ vertės.
2. Apskaičiuojamos naudojamų duomenų liekanų vertės a_i. Šios vertės priklauso nuo sudaryto modelio ir jų apskaičiavimo formulė priklauso nuo modelio struktūros.
3. Turint modelio liekanų vertes, apskaičiuojama $\boldsymbol{S_i(\varphi, \theta)}$ funkcijos vertė, esant konkrečioms parametrų vertėms.
4. Pasirinktos modelio optimalios konkrečios iteracijos parametrų vertės yra atnaujinamos, naudojant gradiento nusileidimo (*GD*) algoritmo parametrų atnaujinimo formulę (žr. 2.3.1 formulę).

$$\varphi_{i+1} = \varphi_i - \mu * \frac{dS(\varphi_i,\theta)}{d\varphi_i}; \quad (2.3.1)$$

čia yra nurodomas φ parametro atnaujinimas i – osios iteracijos metu. μ – tai mokymosi koeficientas, kuris nurodo, kuriuo žingsniu yra keičiamas parametras.

5. Jeigu modelio, su neatnaujintomis parametrų vertėmis, $\boldsymbol{S}$ funkcijos vertė yra mažesnė už prieš tai buvusios iteracijos modelio $\boldsymbol{S}$ funkcijos vertę, tuomet parametrų apskaičiavimo procesas yra kartojamas nuo antro žingsnio. Priešingu atveju procesas yra stabdomas.

2.3.2. Patobulintas gradiento nusileidimo algoritmas

Naudojant gradiento nusileidimo algoritmą, $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcija užstrigdavo lokaliame ekstremume. Siekiant išvengti šio reiškinio, parametrams apskaičiuoti papildomai yra naudojamas autoriaus patobulintas gradiento nusileidimo algoritmas. Patobulinto gradiento nusileidimo algoritmo veikimo principas yra pateikiamas žemiau:

1. Pasirenkamos pradinės, nežinomų parametrų vertės.

2. Priklausomai nuo pasirinktų parametrų verčių apskaičiuojamos tarpinės parametrų vertės - X_2 (pasirinkta parametro vertė), $X_3=X_2*1,5$ (pasirinkto parametro vertė padidinta 50%.), $X_1=X_2*0,5$ (pasirinkto parametro vertė sumažinta 50%.).

3. Apskaičiuojamos naudojamų duomenų liekanų vertės a_i. Šios vertės priklauso nuo sudaryto modelio ir jų apskaičiavimo formulė priklauso nuo modelio struktūros.

4. Turint modelio liekanų vertes a_i, apskaičiuojama $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcijos vertė, esant konkrečioms parametrų vertėms. $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcija yra apskaičiuojama su visomis įmanomomis perskaičiuotų modelio parametrų verčių kombinacijomis. Todėl turint $M=q+p$ parametrų, kai kiekvienas parametras turi po 3 vertes, vienos iteracijos metu apskaičiuojamos $(3q)*(3p)$ $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcijos vertės.

5. Gautos $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcijos vertės yra tarpusavyje palyginamos ir pasirenkama $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcija su mažiausia verte. Priklausomai nuo nustatyto minimumo modelio atitinkamos parametrų vertės yra laikomos optimalios toje iteracijoje ir yra naudojamos kaip pagrindinės parametrų vertės.

6. Pasirinkto modelio konkrečios iteracijos parametrų vertės yra atnaujinamos, naudojant gradiento nusileidimo (*GD*) algoritmą. Gradiento nusileidimo algoritmo parametrų vertės atnaujinimo formulė yra pateikta žemiau (žr. 2.3.2 ir 2.3.3 formulės).

$$\boldsymbol{\varphi_k} = \boldsymbol{\varphi_{k-1}} - \left(\mu * \frac{\partial \boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})}{\partial \varphi_{k-1}}\right); \quad (2.3.2)$$

$$\boldsymbol{\theta_k} = \boldsymbol{\theta_{k-1}} - \left(\mu * \frac{\partial \boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})}{\partial \theta_{k-1}}\right); \quad (2.3.3)$$

7. Jeigu modelio, su neatnaujintomis parametrų vertėmis, $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcijos ekstremumo vertė yra mažesnė už prieš tai buvusios iteracijos modelio $\boldsymbol{S}(\boldsymbol{\varphi},\boldsymbol{\theta})$ funkcijos ekstremumo vertę, tuomet parametrų apskaičiavimo procesas yra kartojamas nuo antro žingsnio. Priešingu atveju pracesas yra stabdomas.

Autoriaus sudaryta autoregresinio modelio parametrų apskaičiavimo realizacija buvo atlikta, naudojant Matlab programą ir realizacijos pavyzdys yra pateiktas laikmenoje (žr. „*AutoregParamGD*.m“ ir „*AutoregParamUGD_LM.m*“).

2.3.3. Levenberg – Marquardt ekstremumo paieškos algoritmas

Palyginimui su gradiento nusileidimo algoritmu taip pat yra naudojamas Levenberg – Marquardt ekstremumo paieškos algoritmas. Pastarojo algoritmo koeficientų vertės atnaujinimo formulė yra pateikta žemiau (žr. 2.3.7 ir 2.3.8 formules).

Autoregresinių modelių parametrams nustatyti, naudojant Levenberg – Marquardt ekstremumo paieškos algoritmą, algoritmo veikimo principas yra pateikiamas žemiau:

1. Pradinis sudaryto autoregresinio modelio parametrų parinkimas.

2. Apskaičiuojamos sudaryto modelio liekanų a_i vertės naudojamų duomenų visame intervale.

3. Apskaičiuojama prognozavimo modelio $\boldsymbol{S}(\boldsymbol{\varphi}, \boldsymbol{\theta})$ funkcijos Jakobiano matrica, naudojamų parametrų atžvilgiu. Jakobiano matrica ARMA(2,2) modeliui yra pateikiama formulėje žemiau (žr. 2.3.4 formulę).

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial \boldsymbol{S_1}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \varphi_1} & \frac{\partial \boldsymbol{S_1}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \varphi_2} & \frac{\partial \boldsymbol{S_1}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \theta_1} & \frac{\partial \boldsymbol{S_1}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \theta_2} \\ \frac{\partial \boldsymbol{S_2}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \varphi_1} & \frac{\partial \boldsymbol{S_2}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \varphi_2} & \frac{\partial \boldsymbol{S_2}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \theta_1} & \frac{\partial \boldsymbol{S_2}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \theta_2} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \boldsymbol{S_N}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \varphi_1} & \frac{\partial \boldsymbol{S_N}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \varphi_2} & \frac{\partial \boldsymbol{S_N}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \theta_1} & \frac{\partial \boldsymbol{S_N}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{\partial \theta_2} \end{bmatrix}; \quad (2.3.4)$$

4. Nustatoma sudaryto autoregresinio prognozavimo modelio $\boldsymbol{S}$ funkcijos vertė (žr. 2.3.5 formulę).

$$\boldsymbol{S} = \boldsymbol{e} = \boldsymbol{\Delta} * \boldsymbol{\Delta^T} = \boldsymbol{a} * \boldsymbol{a^T}; \quad (2.3.5)$$

5. Apskaičiuojama modelio Heseno matrica $\boldsymbol{H}$ (žr. 2.3.6 formulę).

$$\boldsymbol{H} = \boldsymbol{J^T J}; \quad (2.3.6)$$

6. Atliekamas pasirinktų parametrų atnaujinimas, remiantis Levenberg – Marquardt ekstremumo paieškos algoritmu (žr. 2.3.7 ir 2.3.8 formules).

$$\varphi_{1up} = \varphi_1 + ((\boldsymbol{H} + \lambda \boldsymbol{I})^{-1}) * (\boldsymbol{J^T}\Delta); \quad (2.3.7)$$

$$\theta_{1up} = \theta_1 + ((\boldsymbol{H} + \lambda \boldsymbol{I})^{-1}) * (\boldsymbol{J^T}\Delta); \quad (2.3.8)$$

7. Atliekamas ekstremumo paieškos algoritmo patobulinimas analogiškas patobulinimui, atliktam gradiento nusileidimo algoritme. Taip ir šioje dalyje patobulinimo žingsniai yra identiški žingsniams patobulinto gradiento nusileidimo algoritme nuo 2 žingsnio iki 5 žingsnio imtinai.

8. Sudaromas autoregresinis modelis su atnaujintais parametrais ir apskaičiuojamos $\boldsymbol{S_{up}}$ funkcijos vertė, esant atnaujintiems φ_{1up} ir θ_{1up} parametrams.

9. Jeigu apskaičiuotoji $\boldsymbol{S_{up}}$ funkcijos vertė su atnaujintais parametrais yra mažesnė už $\boldsymbol{S}$ funkcijos vertę, tuomet mokymosi koeficiento λ yra sumažinama 10 kartų, o $\varphi_1 = \varphi_{1up}$ ir $\theta_1 = \theta_{1up}$ bei

procesas kartojamas nuo 2 žingsnio. Tačiau jeigu $\boldsymbol{S_{up}} > \boldsymbol{S}$, tuomet λ yra padidinama 10 kartų, o φ_1 ir θ_1 lieka tie patys. Procesas yra kartojamas nuo 6 žingsnio

Radus $\boldsymbol{\phi}$ ir $\boldsymbol{\theta}$ vertes, σ_a parametro vertė yra surandama pagal *LLF()* funkcijos ekstremumą (žr. 2.3.9, 2.3.10 ir 2.3.11 formules).

$$\frac{\partial LLF(\boldsymbol{\varphi}, \boldsymbol{\theta}, \sigma_a)}{\partial \sigma_a} = 0; \tag{2.3.9}$$

$$\frac{\partial LLF(\boldsymbol{\varphi}, \boldsymbol{\theta}, \sigma_a)}{\partial {\sigma_a}^2} = (-\frac{N}{2}\ln({\sigma_a}^2) - \frac{\boldsymbol{S}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{2{\sigma_a}^2})' = 0; \tag{2.3.10}$$

$$\frac{\partial LLF(\boldsymbol{\varphi}, \boldsymbol{\theta}, \sigma_a)}{\partial {\sigma_a}^2} = \frac{\boldsymbol{S}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{2({\sigma_a}^2)^2} - \frac{N}{2} = 0; \tag{2.3.11}$$

Tuomet prognozavimo modelio liekanų dispersija yra išreikšta formulėje žemiau (žr. 2.3.12 formulę).

$${\sigma_a}^2 = \frac{\boldsymbol{S}(\boldsymbol{\varphi}, \boldsymbol{\theta})}{2 * N}; \tag{2.3.12}$$

2.4. Eksponentinio išlyginimo prognozavimo modelio ekstremumo paieškos algoritmai

Eksponentinio išlyginimo prognozavimo modelio nežinomo koeficiento vertei nustatyti yra naudojami gradiento nusileidimo, patobulinto gradiento nusileidimo, dalinimo pusiau, auksinio pjūvio, penkių taškų ir Landveberio optimizacijos algoritmai. Gradiento nusileidimo ir patobulinto gradiento nusileidimo algoritmų veikimo principai yra aprašyti aukščiau. Gradiento nusileidimo ir patobulinto gradiento nusileidimo abiejuose algoritmuose pradinė nežinomo koeficiento α vertė yra 0. Žingsnis, su kuriuo yra atnaujinama koeficiento vertė atitinkamai yra 0,00001. Abiejuose algoritmuose maksimalus iteracijų skaičius yra 5000.

2.4.1. Dalinimo pusiau algoritmas

Dalinimo pusiau algoritmas. Naudojant šį algoritmą yra ieškoma vidurkio išlyginimo koeficiento α vertė. Pasirinkta sritis, iš kurios bus nustatoma koeficiento vertė yra nuo -1 iki 2 imtinai. Maksimalus iteracijų skaičius yra 2000. Algoritmo pagrindiniai žingsniai:

1. Eksponentinio išlyginimo algoritmo išlyginimo koeficiento vertė yra prilyginama koeficiento verčių intervalo pradžiai. Pagal pasirinkto koeficiento vertę yra sudaromas prognozavimo modelis ir yra atliekama prognozė bei nustatoma MSE paklaida tarp tikrojo energijos suvartojimo ir prognozuoto.

2. Kartojamas pirmas žingsnis, tik dabar išlyginimo koeficento vertė yra prilyginama koeficiento verčių intervalo pabaigai.

3. Apskaičiuotos paklaidos yra palyginamos tarpusavyje.

4. Jeigu gautoji paklaida n-ojoje iteracijoje sudaro 98% nuo (n-1)-os iteracijos mažiausios prognozavimo paklaidos, tuomet procesas yra stabdomas ir laikoma, kad yra surasta optimali išlyginimo koeficiento vertė.

5. Jeigu procesas pasiekė maksimalų iteracijų skaičių, algoritmas yra stabdomas.

6. Jeigu modelio, su koeficiento verte lygia ieškomo parametro verčių intervalo pradžiai, paklaida yra didesnė už prognozavimo paklaidą modelio su koeficiento verte lygia ieškomo parametro verčių intervalo pabaigai, tuomet nauja verčių intervalo pradžia yra verčių intervalo aritmetinis vidurkis. Procesas yra kartojamas nuo antro žingsnio.

7. Jeigu modelio su koeficiento verte lygia ieškomo parametro verčių intervalo pabaigai, paklaida yra didesnė už prognozavimo paklaidą, kai koeficiento verte lygi ieškomo parametro verčių intervalo pradžiai, tuomet nauja verčių intervalo pabaiga yra verčių intervalo aritmetinis vidurkis. Procesas yra kartojamas nuo antro žingsnio.

2.4.2. Auksinio pjūvio algoritmas

Auksinio pjūvio algoritmo veikimo principas yra analogiškas dalinimo pusiau algoritmui. Skirtumai tarp algoritmų yra – išlyginimo koeficiento verčių ribos, kurios yra nuo 0 iki 2 imtinai. Taip pat, modelyje nauja koeficiento verčių intervalo riba (pradinė arba galinė) yra ne aritmetinis intervalo vidurkis, kaip tai yra realizuota dalinimo pusiau algoritme, bet riba yra apskaičiuojama pagal formulę, pateiktą žemiau (žr. 2.4.1 formulę).

$$Aksinis\ pjūvis = \frac{(-1+\sqrt{5})}{2} * Intervalo\ pabaiga; \tag{2.4.1}$$

2.4.3. Penkių taškų algoritmas

Penkių taškų algoritme optimalaus išlyginimo koeficiento vertės nustatymo principas yra analogiškas auksinio pjūvio arba dalinimo pusiau algoritmams. Algoritmų veikimo principas:

1. Pasirinkti išlyginimo koeficiento vertę iš verčių intervalo

2. Sudaromas modelis ir pagal jį yra sudaroma prognozė. Šiame algoritme koeficiento verčių intervalas yra nuo 0 iki 2 imtinai.

3. Apskaičiuojama paklaida tarp sudarytos prognozės ir tikrojo energijos suvartojimo.

4. Atliekamas koeficiento verčių intervalo ribų pakeitimas.

5. Algoritmo žingsniai yra kartojami kol nebus tenkinamas stabdymo kriterijus ($MSE_n > MSE_{n-1}*0,98$) arba pasiektas maksimalus iteracijų skaičius 2000.

Šiame algoritme, vietoje naudojamų 2 taškų iš koeficiento verčių intervalo, yra naudojami 5 taškai iš verčių intervalo. Taškai yra intervalo pradžia, intervalo pabaiga, intervalo vidurys (žr. 2.4.2 formulę), intervalo pirmas ketvirtis (žr. 2.4.3 formulę) bei intervalo trečias ketvirtis (žr. 2.4.4 formulę).

$$Intervalo\ vidurys = \frac{Intervalo\ pabaiga}{2}; \tag{2.4.2}$$

$$Pirmas\ ketvirtis = \frac{Intervalo\ vidurys}{2}; \tag{2.4.3}$$

$$Trečias\ ketvirtis = \frac{Intervalo\ vidurys}{2} + Intervalo\ vidurys; \tag{2.4.4}$$

Šiame algoritme yra lyginamos tarpusavyje sudarytų prognozavimo modelių prognozavimo paklaidos nuo visų penkių taškų. Intervalo ribų pakeitimas, nustačius minimalią prognozavimo paklaidą:

- Jeigu parametro vertės taškas, atitinkantis minimalią prognozavimo paklaidą, yra intervalo pradžios taškas, tuomet nauja intervalo pradžia yra buvęs pradžios taškas, o intervalo pabaiga tampa intervalo pirmas ketvirtis.

- Jeigu minimalią prognozavimo paklaidą atitinka ieškomo parametro verčių intervalo pabaigos taškas, tuomet naujo parametro verčių intervalo pradžios taškas tampa buvusio intervalo trečio ketvirčio taškas. O naujo intervalo pabaigos taškas nekinta.

- Jeigu minimalią prognozavimo paklaidą atitinka ieškomo parametro verčių intervalo vidurio taškas, tuomet ieškomo parametro naujo verčių intervalo pradžios taškas tampa buvusio verčių intervalo pirmo ketvirčio taškas. Naujo verčių intervalo pabaigos taškas tampa buvusio verčių intervalo trečiojo ketvirčio ženklas.

- Jeigu minimalią prognozavimo paklaidą atitinka ieškomo parametro verčių intervalo pirmo ketvirčio taškas, tuomet naujo intervalo pradžios taškas tampa, buvusio verčių intervalo pradžios taškas. Naujo verčių intervalo pabaigos taškas tampa buvusio verčių intervalo vidurio taškas.

- Jeigu minimalią prognozavimo paklaidą atitinka ieškomo parametro verčių intervalo trečio ketvirčio taškas, tuomet naujo verčių intervalo pradžios taškas tampa buvusio intervalo vidurio taškas. Ieškomo parametro naujo verčių intervalo pabaigos taškas tampa buvusio verčių intervalo pabaigos taškas.

2.4.4. Landveberio optimizacijos algoritmas

Paskutinis algoritmas naudojamas optimalios eksponentinio išlyginimo koeficiento vertes nustatymui yra naudojamas Landveberio optimizacijos algoritmas. Veikimo principas yra analogiškas gradiento nusileidimo algoritmui. Pradinė ieškomo parametro koeficiento vertė yra 0, maksimalus iteracijų skaičius yra 2000. Ieškomo koeficiento vertės atnaujinimo žingsnis yra pasirinktas θ=0,0001; Algoritmo žingsniai:

1. Nuo pasirinktos koeficiento vertės sudaromas prognozavimo modelis
2. Atliekamas prognozavimas pagal pasirinktą modelį
3. Apskaičiuojama prognozavimo paklaida tarp prognozuojamų duomenų ir tikrojo elektros energijos suvartojimo.
4. Apskaičiuojamas prognozavimo paklaidos gradientas
5. Atnaujinama ieškomo parametro vertė
6. Tikrinamas stabdymo kriterijus ir maksimalus iteracijų skaičius. Jeigu nei vienas nei kitas nėra tenkinami, tuomet viskas kartojama nuo pirmo žingsnio, priešingu atveju procesas yra stabdomas.

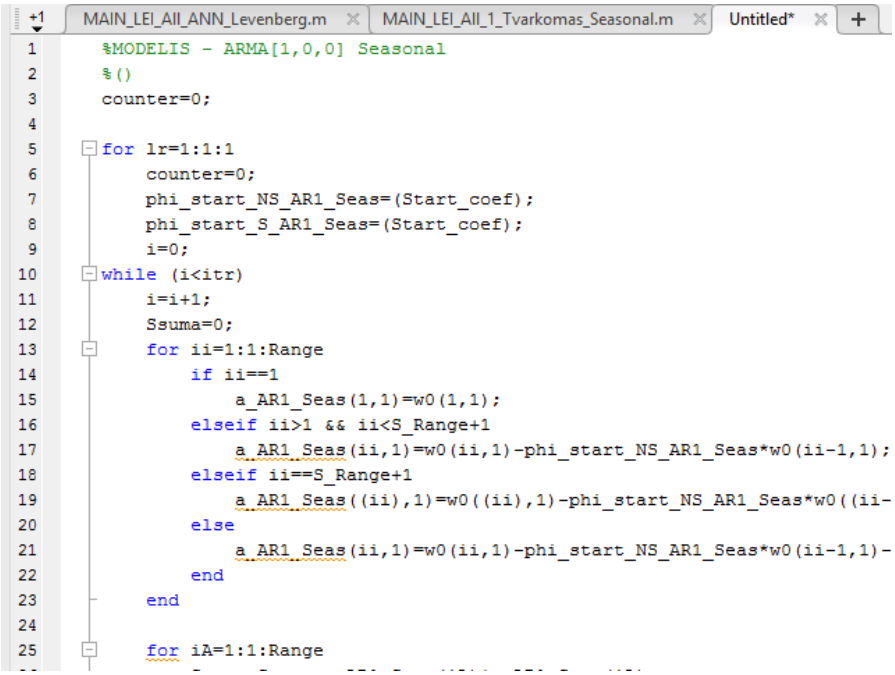
Parametro atnaujinimo formulė yra pateikiama žemiau (žr. 2.4.5 formulę)

$$\alpha_n = \alpha_{n-1} + \theta * \frac{dS(\alpha_{n-1}, x_{n-1})}{d\alpha_{n-1}}; \quad (2.4.5)$$

2.5. Prognozavimo modelių realizavimo aplinka

2.5.1. Statistinių prognozavimo modelių realizacija Matlab aplinkoje

Šiame baigiamajame darbe statistiniai autoregresiniai prognozavimo modeliai ir eksponentinio išlyginimo prognozavimo modelis yra realizuoti Matlab aplinkoje. Visi modeliai realizuoti kodo eilute (žr. 2.5.1 pav.).



2.5.1 pav. Autoregresinio prognozavimo modelio realizacijos fragmentas

Realizuojant statistinius prognozavimo modelius, visi naudojami parametrų nustatymo metodai ir algoritmai yra realizuoti kodo eilute. Nustatant autoregresinių prognozavimo modelių struktūrą, naudojamas *Econometrics Toolbox* praplėtimas. Pastarasis praplėtimas yra reikalingas, nes nustatant modelio struktūrą yra naudojamos autokoreliacijos *autocorr()* ir dalinės autokoreliacijos *parcorr()* funkcijos. Taip pat nustatant, ar naudojami duomenys yra statiniai, yra naudojama *adftest()* funkcija. Statistinių prognozavimo modelių realizacijos programinis kodas yra pateikiamas laikmenoje (žr. „*AutoregParamGD.m*“ ir „*AutoregParamUGD_LM.m*“).

2.5.2. Dirbtinio neuroninio tinklo prognozavimo modelio realizacija Matlab aplinkoje

Dirbtinio neuroninio tinklo prognozavimo modeliai buvo realizuoti Matlab aplinkoje, naudojant *nntoolbox* praplėtimo paketą. Sudarant neuroninį tinklą, tinklas buvo realizuotas kodo eilute, siekiant dinamiškai keisti neuroninio tinklo parametrus be būtinybės naudoti Matlab paketo siūlomą grafinę sąsają. Neuroninio tinklo realizavimo fragmentas yra pateikiamas žemiau (žr. 2.5.2 pav.).

```
counter=0;
mse1=zeros(3,1);
time1=zeros(3,1);
regression1=zeros(3,1);

for i=1:5:300
    counter=counter+1;
    for h=1:1:3

        net = feedforwardnet([i],'trainlm');
        net.adaptFcn='adaptwb';
        net.adaptParam='none';
        net.derivFcn='defaultderiv';
        net.divideFcn='dividerand';
        net.divideParam.trainRatio=0.7;
        net.divideParam.valRatio=0.15;
        net.divideParam.testRatio=0.15;
        net.divideMode='sample';
        net.initFcn='initlay';
        net.performParam.regularization=0;
        net.performParam.normalization='none';
        net.plotFcns={'plotperform', 'plottrainstate', 'plotreg
        net.performFcn='mse';
        net.trainParam.epochs = 1000;
        net.trainParam.mu = Learnin_rate_mu;
        net.trainParam.showWindow = 0;
        net.trainParam.showCommandLine=0;
        net.trainParam.show = 25;
```

2.5.2 pav. Dirbtino neuroninio tinklo prognozavimo modelio realizacijos fragmentas

Sudaryto prognozavimo modelio dirbtinis neuroninis tinklas yra tiesioginio duomenų sklidimo tipo. Neuronionio tinklo tipas yra nurodomas, sudarant *feedforwardnet* modelį. Realizuoto dirbtinio neuroninio tinklo prognozavimo modelio vienas iš pavyzdžių yra pateikiamas laikmenoje (žr. „*ANN_LM.m*“).

2.5.1 lentelė. Dirbtinio intelekto prognozavimo modelio, neuroninio tinklo aktualūs parametrai.

Neuroninio tinklo parametras	**Parametro vertė**	**Parametro paaiškinimas**
net.trainFcn		Neuroninio modelio apmokymo algoritmas
net.layers{N}.transferFcn		Tinklo N-ojo paslėpto sluoksnio neuronų aktyvavimo funkcija
net.adaptFcn	adaptwb	Neuroninio tinklo naudojama prisitaikymo funkcija
net.adaptParam	none	Naudojamos prisitaikymo funkcijos parametrai
net.derivFcn	defaultderiv	Tinklo naudojama išvestinės funkcija, skirta apskaičiuoti paklaidos išvestinę, kuri sudaro Jakobiano matricą
net.divideFcn	dividerand	Naudojamo tinklo apmokymo duomenų paskirstymo funkcija
net.divideParam.trainRatio	0,7	Tinklo apmokymo duomenų dalis, tenkanti tinklo apmokymui
net.divideParam.valRatio	0,15	Tinklo apmokymo duomenų dalis, tenkanti tinklo validacijai
net.divideParam.testRatio	0,15	Tinklo apmokymo duomenų dalis, tenkanti tinklo testavimui
net.divideMode	sample	Duomenų paskirstymo funkcijos parametras, kurio atžvilgiu bus dalinami naudojami duomenys

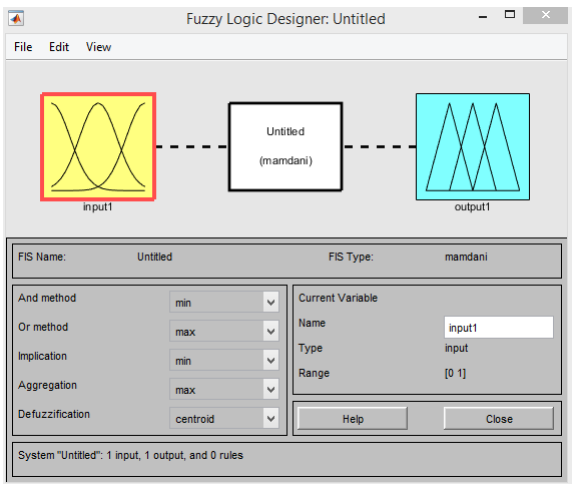
net.initFcn	initlay	Neuroninio tinklo funkcija, kuri inicializuoja tinklo biosų ir svorių pradines vertes
net.performFcn	mse	Neuroninio tinklo prognozavimo tikslumo apskaičiavimo funkcija
net.trainParam.epochs	1000	Tinklui apmokyti naudojamas maksimalus iteracijų skaičius
net.trainParam.mu	Learnin_rate_mu	Tinklo apmokymo, svorių verčių atnaujinimo parametras
net.trainParam.time	Inf.	Neuroninio tinklo maksimali apmokymo trukmė
net.trainParam.goal	0	Tinklo apmokymo tikslas - stabdymo kriterijus
net.trainParam.min_grad	1,00E-07	Minimali priimtina prognozavimo modelio paklaidos vertė
net.trainParam.max_fail	6	Tinklo validacijos maksimalus nesėkmingų badymų skaičius

Sudaryto dirbtinio neuroninio tinklo keičiami parametrai yra nurodyti lentelėje aukščiau (žr. 2.5.1 lentelę). Nustatant tinkamą dirbtinio intelekto prognozavimo modelį, dirbtinio neuroninio tinklo prognozavimo modelis lyginamas su statistiniais prognozavimo modeliais. Aukščiau minėti dirbtinio neuroninio tinklo parametrai buvo keičiami.

Plačiau apie dirbtinio neuroninio tinklo tinkamumą ir teorinį pagrindą, pritaikant jį pastato suvartojamos elektros energijos prognozavimui, yra parašyta trečiajame šaltinyje [36].

2.5.3. Neraiškios logikos reguliatoriaus realizacija Matlab aplinkoje

Neraiškios logikos reguliatoriai, kaip ir prognozavimo modeliai, yra realizuojami Matlab aplinkoje. Reguliatoriui realizuoti yra naudojamas Matlab praplėtimo paketas *Fuzzy Logic Toolbox*. Pirminė neraiškios logikos reguliatoriaus konfigūracija yra sudaroma, naudojant Matlab aplinkos neraiškios logikos reguliatoriaus grafinę sąsają – *Fuzzy Logic Designer*. Grafinė sąsaja yra iškviečiama įrašius į komandinį langą *fuzzy* kreipinį (žr. 2.5.3 pav.).



2.5.3 pav. Neraiškios logikos reguliatoriaus konfigūravimo grafinė sąsaja Matlab aplinkoje

Sudarius pirminę reguliatoriaus konfigūraciją, ji išsaugoma ir vėliau yra perkeliama į programinį kodą, naudojant *readfis()* funkciją. Nustatant optimalų lingvistinių kintamųjų skaičių ir

priklausomumo funkciją, realizuotos neraiškios logikos reguliatoriaus konfigūracijos testavimo metu buvo keičiamos dinamiškai (žr. 2.5.4 pav.).

```
MPE_FuzzyMFArr=zeros(Range_T,1);

%Atrenkama mf funkcijos,  trimf, trapmf,  gbellmf, gaussmf

PHI_NS_Beg=phi_start_NS_AR1_Seas;
PHI_S_Beg=phi_start_S_AR1_Seas;
zzd=0;
counter1=1;
for if1=1:1:4
    for if2=1:1:3
        zzd=zzd+1;
        switch zzd
            case 1
                fis= readfis('FuzzyCode1.fis');
            case 2
                fis= readfis('FuzzyCode1Trap.fis');
            case 3
                fis= readfis('FuzzyCode1Gauss.fis');
            case 4
                fis= readfis('FuzzyCode1Bell.fis');
            case 5
                fis= readfis('FuzzyCode2.fis');
```

2.5.4 pav. Neraiškios logikos reguliatoriaus optimalaus lingvistinių kintamųjų skaičiaus ir priklausomumo funkcijos nustatymo programos fragmentas

Siekiant nustatyti optimalią neraiškios logikos reguliatoriaus žinių bazės konfigūraciją, reguliatoriaus žinių bazės konfigūracija keičiama dinamiškai per kodo eilutę, atsisakant Matlab aplinkos reguliatoriaus konfigūravimo grafinės sąsajos (žr. 2.5.5 pav.).

```
if (rule4(3)==5)%NB tikrinimas ant PS
   rule5(3)=4;%PB papildymas ant NS
elseif (rule4(3)==7)%NB tikrinimas ant PM
    rule5(3)=6;%PB papildymas ant NM
elseif (rule4(3)==3)%NB tikrinimas ant PB
    rule5(3)=1;%PB papildymas ant NB
elseif (rule4(3)==2)
    rule5(3)=2;
end

%Irasymas i fuzzy reguliatoriu
rules=[rule1; rule2; rule3; rule4; rule5; rule6; rule7];
name(ruleCount,1)=addrule(fis_Test,rules);

fis=name(ruleCount,1);
if (ruleCount==3784)
    writefis(fis,'FuzzyBestRules.fis');
end
```

2.5.5 pav. Neraiškios logikos reguliatoriaus optimalios žinių bazės konfigūracijos nustatymo programos fragmentas

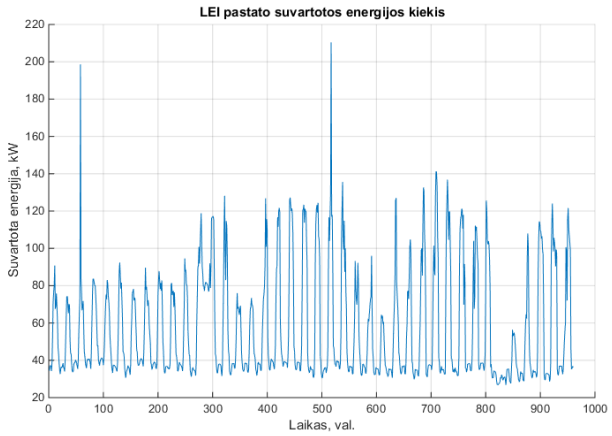
Neraiškios logikos reguliatoriaus optimalios žinių bazės konfigūracijos nustatymo programinis kodas yra pateikiamas laikmenoje (žr. „*FuzzyOptimalKnowBase.m*“).

3. Darbo rezultatai

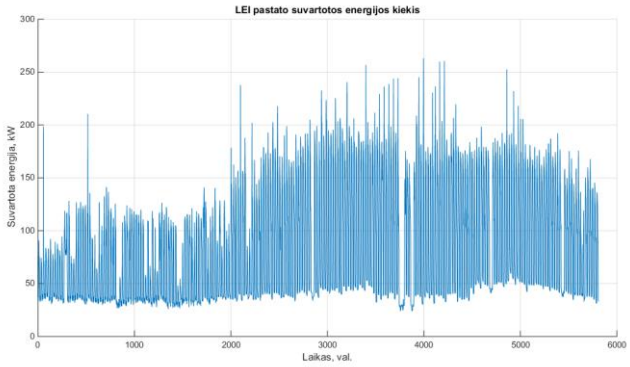
3.1. Optimalaus autoregresinio prognozavimo modelio nustatymas

3.1.1. Vizualinė duomenų analizė

Baigiamajame magistrantūros darbe, prieš sudarant autoregresinius prognozavimo modelius, vizualiai buvo išanalizuota turimo objekto suvartojama elektros energija (žr. 3.1.1 pav.). Sudarant bet kokio tipo prognozavimo modelius (statistinius ir dirbtinio intelekto), buvo naudojamas ribotas informacijos kiekis – 40 darbo dienų, kurios atitinka 940 informacinių taškų.

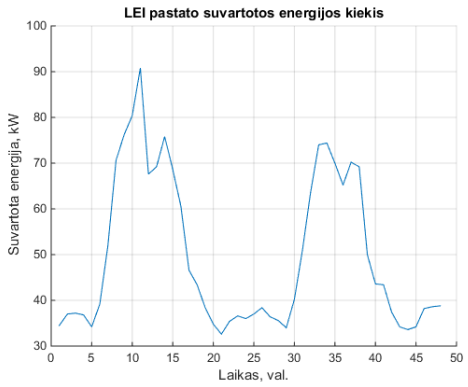


3.1.1 pav. LEI analizuojamo pastato suvartotos elektros energijos kiekis per 40 d.d.



3.1.2 pav. LEI analizuojamo pastato suvartota elektros energija kas valandą, darbo dienomis nuo 2017-05-18 iki 2018-04-23

Atlikus vizualinę individualaus pastato suvartojamos elektros energijos kiekio duomenų analizę, teisinga būtų teigti, kad sistemos elektros energijos suvartojimo grafike yra ryškiai išreikštas sezoniškumas (žr. 3.1.1 ir 3.1.2 pav.). Siekiant sudaryti efektyvų statistinį prognozavimo modelį, ši išreikštą sezoniškumą bus būtina pašalinti. Taip pat iš gautų grafikų galima preliminariai pasakyti duomenų periodiškumo intervalą – pastarasis yra apie 24 val. (žr. 3.1.3 pav.).



3.1.3 pav. LEI analizuojamo pastato suvartota elektros energija kas valandą, darbo dienomis 48 val. bėgyje

Atlikus vizualinę duomenų analizę buvo atlikti papildomi analitiniai skaičiavimai, kurie patvirtino prielaidą, kad duomenyse yra sezoniškumas su 24-uriu valandų periodiškumo intervalu.

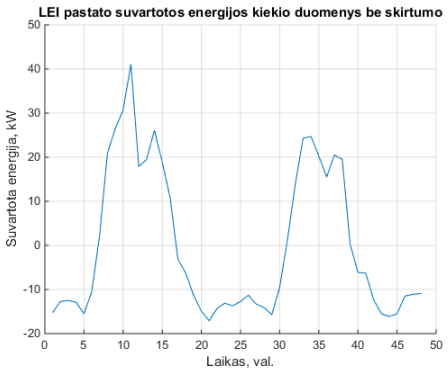
3.1.2. Analitinė duomenų analizė

Prieš sudarant prognozavimo modelius yra atlikta analitinė turimų duomenų analizė, nustatant naudojamų duomenų stacionarumą. Kaip analizės priemonė buvo naudojamas praplėstasis Dickey-Fuller testas (angl. *Augmented Dickey–Fuller test, ADF test*). Remiantis vizualinės analizės rezultatais, norint pašalinti pastebėtą duomenų sezoniškumą, yra apskaičiuojami naudojamų duomenų pirmos eilės skirtumas, naudojant skirtuminį perskaičiavimą. Gauti analizės rezultatai yra pateikiami trečioje lentelėje (žr. 3.1.1 lentelę). Vizualinio naudojamų duomenų atvaizdavimo kodas ir analitinės analizės realizacijos kodas yra pateiktas laikmenoje (žr. „*DataVisual.m*“).

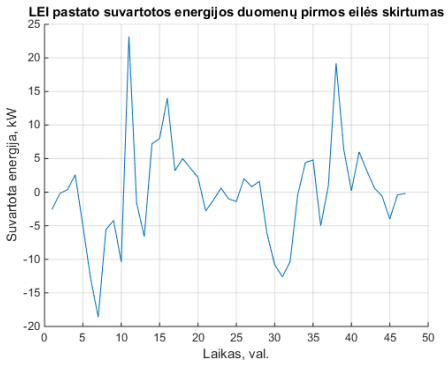
3.1.1 lentelė. Duomenų analitinis tyrimas, naudojant ADF testą

Eilės Nr.	**Duomenys**	**ADF testo rezultatas**	**Stacionarumas**
1	Neapdoroti duomenys	0	Nestacionarus
2	Duomenys be vidurkio	1	Nestacionarus
3	Duomenų pirmos eilės skirtumas	1	Stacionarus

Skirtuminis perskaičiavimas buvo realizuotas kodo eilute, sudarius funkciją *Difference()*, šios funkcijos išeities tekstas yra pateikiamas prieduose (žr. 1 priedą). Gauti perskaičiuoti duomenys yra atvaizduojami paveiksluose žemiau (žr. 3.1.4 ir 3.1.5 pav.).



3.1.4 pav. LEI pastato suvartojama energija, pašalinus vidurkį

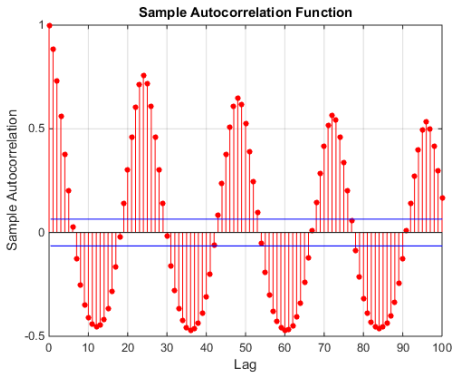


3.1.5 pav. LEI pastato suvartojamos energijos pirmas skirtumas

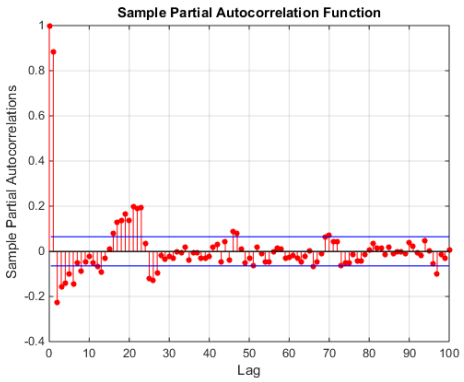
Gauta informacija analitinės analizės metu perša prielaidą, kad autoregresiniai prognozavimo modeliai turėtų būti sudaromi, naudojant tik stacionarių procesų duomenis. Remiantis gauta informacija šioje dalyje negalima tiksliai pasakyti, kuriuos stacionarius duomenis naudojant modelis bus tikslesnis. Prognozavimo modeliai yra realizuojami su kiekviena stacionarią duomenų imtimi.

3.1.3. Prognozavimo modelio struktūros identifikavimas

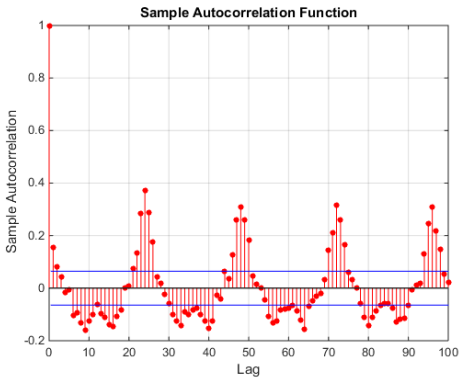
Modelio struktūrai identifikuoti yra naudojamos autokoreliacijos funkcija (*ACF*) ir dalinė autokoreliacijos funkcija (*PACF*). Modelio identifikavimas buvo realizuotas kodo eilute naudojant standartinės *Matlab* funkcijų išraiškas *autocorr(x,y)* ir *parcorr(x,y)*, kai *x* – atitinka analizuojamus duomenis, o *y* – autokoreliacijos eilę. Gauti autokoreliacijos funkcijos ir dalinės autokoreliacijos funkcijos grafikai, atvaizduojami kiekvienai duomenų imčiai (žr. 3.1.6, 3.1.7, 3.1.8 ir 3.1.7 pav.). Apskaičiuojant *ACF* ir *PACF* kiekvienai duomenų imčiai, maksimalus koreliacijos žingsnis tarp duomenų buvo nurodytas 100.



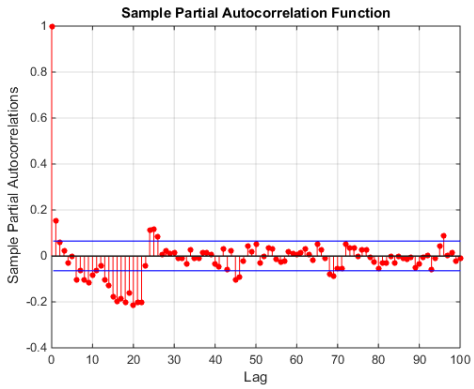
3.1.6 pav. LEI duomenų, su pašalintu vidurkiu, *ACF* funkcijos grafikas



3.1.7 pav. LEI duomenų, su pašalinu vidurkiu, *PACF* funkcijos grafikas



3.1.8 pav. LEI duomenų pirmos eilės skirtumo *ACF* funkcijos grafikas



3.1.9 pav. LEI duomenų pirmos eilės skirtumo *PACF* funkcijos grafikas

Remiantis gautais *ACF* ir *PACF* grafikais bei atlikta literatūros analize, sudarytos išvadas kiekvienai duomenų imčiai:

- Rekomenduotinas autoregresinis modelis neapdorotai duomenų imčiai su pašalintu vidurkiu yra AR(1), su koeficiento verte $\phi>0$. Atsižvelgiant į duomenų išraišką bei *ACF* ir *PACF* grafikus (žr. 3.1.6 ir 3.1.7 pav.), matosi kad *ACF* grafiko vertė eksponentiškai mažėja, kai *PACF* grafiko vertė drastiškai sumažėja iki nejautrumo zonos po pirmos eilės vėlinimo.

- Rekomenduotinas autoregresinis modelis pirmos eilės skirtumo duomenų imčiai yra MA(1) su koeficiento verte $\theta<0$. Modelio struktūra, nustatyta remiantis atlikta vizualine ir analitine analize, taip pat atsižvelgus, kad pastarieji duomenys apibūdina stacionarų procesą bei įvertinus *ACF* ir *PACF* grafikus (žr. 3.1.8 ir 3.1.9 pav.).

3.1.4. Autoregresinių modelių išraiškos sudarymas

Kiekvienai stacionariai duomenų imčiai (žr. 2.5.1 lentelę), buvo sudaryti atitinkamos struktūros autoregresiniai modeliai be išorinio kintamojo dedamosios (žr. 3.1.2 lentelę):

3.1.2 lentelė. Realizuotų autoregresinų modelių struktūru analitinės išraiškos be sezoninių dedamųjų

Modelis	**Pradinė modelio išraiška**	**Galutinė modelio išraiška**
ARMA(1,0,0)	$(1-\varphi B)Z_t = a_t$	$Z_t = a_t + \varphi_1 Z_{t-1}$
ARMA(2,0,0)	$(1-\varphi_1 B - \varphi_2 B^2)Z_t = a_t$	$Z_t = a_t + \varphi_1 Z_{t-1} + \varphi_2 Z_{t-2}$
ARMA(0,0,1)	$Z_t = (1-\theta)a_t$	$Z_t = a_t - \theta_1 a_{t-1}$
ARMA(0,0,2)	$Z_t = (1-\theta_1 B - \theta_2 B^2)a_t$	$Z_t = a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2}$
ARMA(1,0,1)	$(1-\varphi B)Z_t = (1-\theta B)a_t$	$Z_t = a_t + \varphi_1 Z_{t-1} - \theta_1 a_{t-1}$
ARMA(2,0,1)	$(1-\varphi_1 B - \varphi_2 B^2)Z_t = (1-\theta B)a_t$	$Z_t = a_t + \varphi_1 Z_{t-1} + \varphi_2 Z_{t-2} - \theta_1 a_{t-1}$
ARMA(1,0,2)	$(1-\varphi B)Z_t = (1-\theta_1 B - \theta_2 B^2)a_t$	$Z_t = a_t + \varphi_1 Z_{t-1} - \theta_1 a_{t-1} - \theta_2 a_{t-2}$
ARMA(2,0,2)	$(1-\varphi_1 B - \varphi_2 B^2)Z_t = = (1-\theta_1 B - \theta_2 B^2)\, a_t$	$Z_t = a_t + Z_1 a_{t-1} + Z_2 a_{t-2} - \theta_1 a_{t-1} - \theta_2 a_{t-2}$

Autoregresiniai modeliai sudaryti duomenų imčiai su pirmos eilės skirtumu turi tokias pat struktūras kaip ir kitų duomenų imčių autoregresiniai modeliai, tačiau šie modeliai įgyja integralinę dedamąją. Ši dedamoji modelio pavadinime yra išreikšta *I* raide ir eina po AR dalies ir prieš MA dalį (*ARMA* modelis su integralinę dedamąją išreiškiamas kaip *ARIMA* modelis).

Taip pat buvo sudaryti autoregresiniai modeliai kiekvienai stacionariai duomenų imčiai, naudojant papildomą autoregresinį sezoniškumo ryšį išreiškiantį narį, kuris apibūdindavo prognozavimo modelio prognozės priklausomybę nuo duomenų tam tikrais intervalais. Šie papildomi koeficientai buvo išreiškiami kaip θ_S ir Φ_S, koeficientai. Pasirinkti sezoniškumo intervalai: 23val., 24 val. ir 25 val. Šie intervalai pasirinkti neatsitiktinai. Atlikus vizualinę duomenų analizę, matosi kad duomenys laiko atžvilgiu atkartoja tą pačią energijos suvartojimo grafiko formą kas 24 val. (žr. 3.1.4 ir 3.1.5 pav.). Autoregresinių modelių, su įtrauktais papildomais nariais, bazinė nesezoninė modelio struktūros dedamoji atitinka aukščiau aprašytas struktūras (žr. 3.1.2 lentelę). Gautos naujos ARMA modelių išraiškos yra pateikiamos žemiau (žr. 3.1.3 lentelę)

3.1.3 lentelė. Realizuotų autoregresinų modelių struktūra ir analitinės išraiškos su sezoniniais dedamaisiais

Modelio Sezoninė dedamoji	**Modelio Nesezoninė dedamoji**	**Modelio analitinė išraiška**
AR(1)	AR(1)	$(1-\Phi B^T)(1-\varphi B)Z_t = a_t$
		$Z_t = a_t + \Phi Z_{t-T} + \varphi Z_{t-1} - \Phi\varphi Z_{t-1-T}$
MA(1)	MA(1)	$Z_t = a_t(1-\Theta B^T)(1-\theta B)$
		$Z_t = a_t - \Theta a_{t-T} - \theta a_{t-1} + \Theta\theta a_{t-T-1}$
ARMA(1,0,1)	ARMA(1,0,1)	$(1-\Phi B^T)(1-\varphi B)Z_t = a_t(1-\Theta B^T)(1-\theta B)$
		$Z_t = a_t - \Theta a_{t-T} - \theta a_{t-1} - \theta\Theta a_{t-1-T} + \Phi Z_{t-T} + \varphi Z_{t-1} - \Phi\varphi Z_{t-1-T}$

Sudarytiems prognozavimo modeliams parametrų vertės yra ieškomos, naudojant Maksimalios tikimybės įvertinimo metodą (angl. *Maximum Likelyhood Estimation MLE*). Kiti parametrų paieškos algoritmai, Yule – Walke ir mažiausių kvadratų, nėra naudojami autoregresinių modelių parametrų paieškai [37]. Parametrų nustatymas, esant nedideliam duomenų kiekiui, naudojant pastaruosius algoritmus gali sąlygoti nestabilų prognozavimo modelį [37,38].

3.1.5. Autoregresinių prognozavimo modelių parametrų paieškos algoritmo pasirinkimas

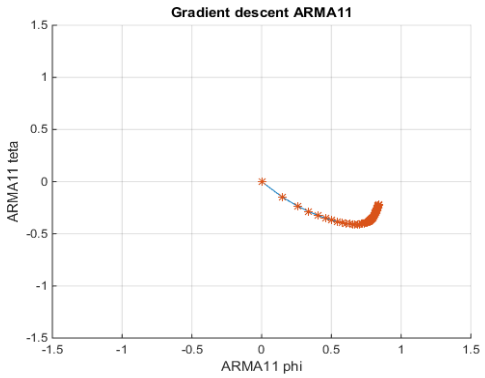
Ieškant statistinių prognozavimo modelių parametrų, buvo naudojamas Maksimalios tikimybės įvertinimo metodas. Norint surasti paramentrus pagal *MLE* metodą, būtina surasti metodo naudojamos *LLF()* (angl. *log likelihood function*) funkcijos ekstremumą. Šiame darbe *LLF()* funkcijos ekstremumui nustatyti yra naudojami trys pagrindiniai algoritmai: gradiento nusileidimo algoritmas, autoriaus patobulintas gradiento nusileidimo algoritmas, taip pat Levemberg Marquardt algoritmas. Šiame etape algoritmai tarpusavyje yra palyginami pagal iteracijų skaičių, reikalingą parametrams nustatyti, taip pat pačio algoritmo robastiškumą, gebėjimą surasti globalų minimumą, pasikeitus pradinėms sąlygoms, taip pat yra vertinamas laikas, per kurį buvo surastos parametrų vertės. Kaip etaloninis modelis yra naudojamas ARMA(1,0,1) prognozavimo modelis, kuriam bus ieškomi parametrai ir tarpusavyje lyginami trys ekstremumo paieškos algoritmai. Siekiant nustatyti ekstremumo paieškos algoritmų robastiškumą, kiekvienas algoritmas ieško ekstremumo verčių, esant skirtingoms pradinėms sąlygoms, kai pradinės parametrų vertės yra: 0, (-1) ir 1. Esant kiekvienai

pradinei sąlygai, skaičiavimai yra pakartojami tris kartus ir ekstremumo paieškos algoritmai tarpusavyje bus lyginami pagal trijų bandymų rezultatų aritmetinį vidurkį.

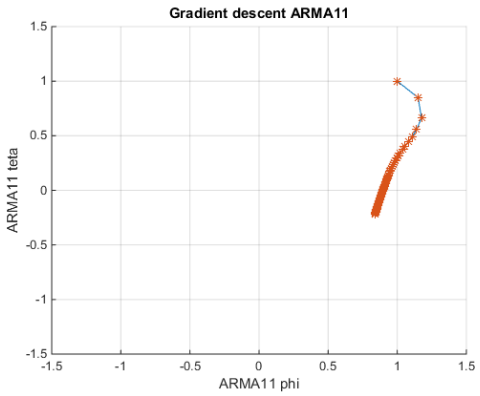
Atlikti skaičiavimo rezultatai yra pateikiami lentelėse žemiau (žr. 3.1.4, 3.1.5 ir 3.1.6 lenteles). Šiose lentelėse yra pateikiama kiekvieno paieškos algoritmo bandymo rezultatai bei surastos parametrų vertės. Taip pat parametrų paieškos procesai yra atvaizduojami grafiškai, pateikiant kiekvieno paieškos algoritmo antrąjį bandymą, esant skirtingoms pradinėms sąlygoms (žr. 3.1.10, 3.1.11, 3.1.12, 3.1.13, 3.1.14, 3.1.15, 3.1.16, 3.1.17 ir 3.1.18 pav.).

3.1.4 lentelė. Gradiento nusileidimo algoritmo ekstremumo paieškos rezultatai

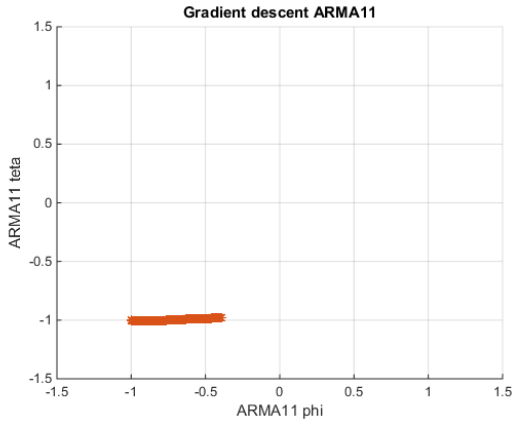
Bandymas	**Parametras**	**Parametro pradinė vertė**	**Algorimtas**		
			Gradiento nusileidimas		
			Parametro vertė	Įteracijų skaičius	Skaičiavimo laikas, *s*
1	ϕ	0	0,8371	1791	0,3451
	θ	0	-0,2220		
2	ϕ	0	0,8371	1791	0,3582
	θ	0	-0,2220		
3	ϕ	0	0,8371	1791	0,3497
	θ	0	-0,2220		
Vidurkis				1791	0,351
1	ϕ	1	0,8371	5000	0,8018
	θ	1	-0,2220		
2	ϕ	1	0,8371	5000	0,8412
	θ	1	-0,2220		
3	ϕ	1	0,8371	5000	0,8766
	θ	1	-0,2220		
Vidurkis				5000	0,83986
1	ϕ	-1	-0,3902	5000	0,7985
	θ	-1	-0,9797		
2	ϕ	-1	-0,3902	5000	0,7939
	θ	-1	-0,9797		
3	ϕ	-1	-0,3902	5000	0,8026
	θ	-1	-0,9797		
Vidurkis				5000	0,7983



3.1.10 pav. Standartinio Gradiento nusileidimo algoritmo koeficientu paieškos grafikas, kai parametrų pradinė vertė yra 0



3.1.11 pav. Standartinio Gradiento nusileidimo algoritmo koeficientu paieškos grafikas, kai parametrų pradinė vertė yra 1

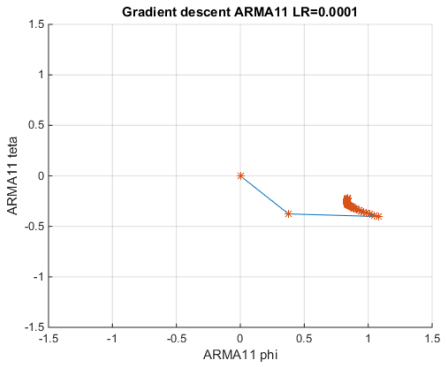


3.1.12 pav. Standartinio Gradiento nusileidimo algoritmo koeficientų paieškos grafikas, kai parametrų pradinė vertė yra -1

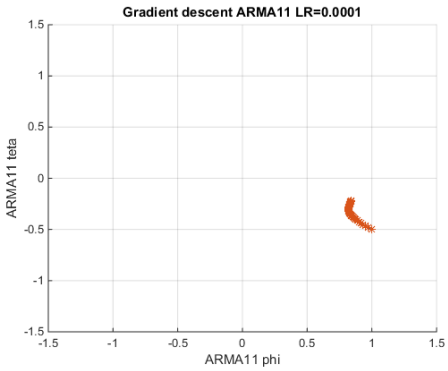
3.1.5 lentelė. Patobulinto gradiento nusileidimo algoritmo ekstremumo paieškos rezultatai

Bandymas	**Parametras**	**Parametro pradinė vertė**	**Algorimtas**		
			Patobulintas Graidinto nusileidimas		
			Parametro vertė	Iteracijų skaičius	Skaičiavimo laikas
1	ϕ	0	0.8371	918	0,7358
	θ	0	-0.2220		
2	ϕ	0	0.8371	918	0,7349
	θ	0	-0.2220		
3	ϕ	0	0.8371	918	0,7539
	θ	0	-0.2220		
Vidurkis				918	0,74153
1	ϕ	1	0.8371	925	0,7418
	θ	1	-0.2220		

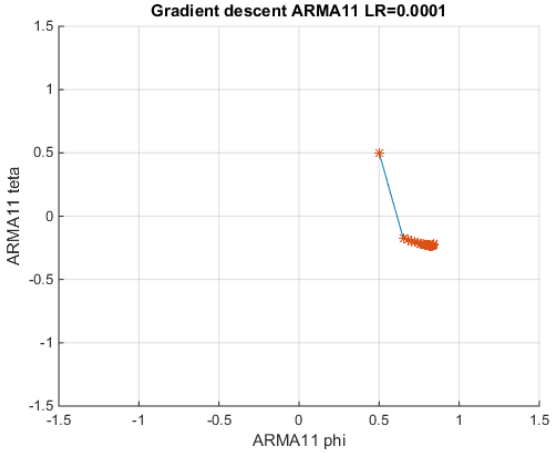
2	ϕ	1	0.8371	925	0,7206
	θ	1	-0.2220		
3	ϕ	1	0.8371	925	0,7645
	θ	1	-0.2220		
Vidurkis				925	0,7423
1	ϕ	-1	0.8371	860	0,7138
	θ	-1	-0.2220		
2	ϕ	-1	0.8371	860	0,6856
	θ	-1	-0.2220		
3	ϕ	-1	0.8371	860	0,6846
	θ	-1	-0.2220		
Vidurkis				860	0,69466



3.1.13 pav. Patobulinto Gradiento nusileidimo algoritmo koeficientu paieškos grafikas, kai parametrų pradinė vertė yra 0



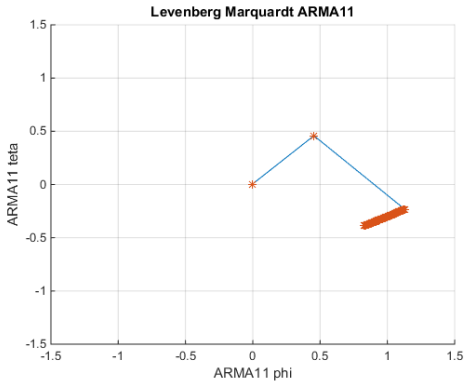
3.1.14 pav. Patobulinto Gradiento nusileidimo algoritmo koeficientu paieškos grafikas, kai parametrų pradinė vertė yra 1



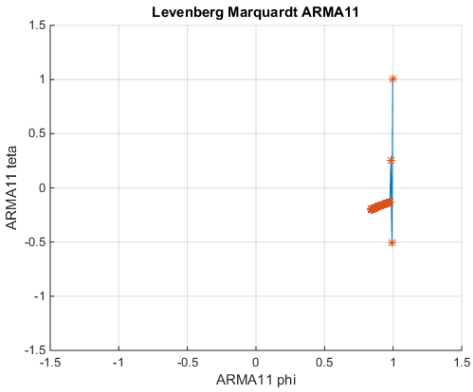
3.1.15 pav. Patobulinto Gradiento nusileidimo algoritmo koeficientų paieškos grafikas, kai parametrų pradinė vertė yra (-1)

3.1.6 lentelė. Levenberg – Marquardt algoritmo ekstremumo paieškos rezultatai

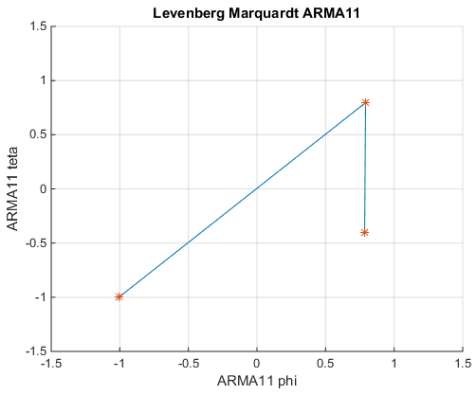
Bandymas	Parametras	Parametro pradinė vertė	Algorimtas		
			Lvenberg Marquardt		
			Parametro vertė	Įteracijų skaičius	Skaičiavimo laikas
1	ϕ	0	0,8324	100	0,3305
	θ	0	-0,3855		
2	ϕ	0	0,8324	100	0,3301
	θ	0	-0,3855		
3	ϕ	0	0,8324	100	0,3628
	θ	0	-0,3855		
Vidurkis				100	0,34113
1	ϕ	1	0,8460	48	0,2864
	θ	1	-0,1970		
2	ϕ	1	0,8460	48	0,2914
	θ	1	-0,1970		
3	ϕ	1	0,8460	48	0,2571
	θ	1	-0,1970		
Vidurkis				48	0,2783
1	ϕ	-1	0,7916	8	0,0535
	θ	-1	-0,3980		
2	ϕ	-1	0,7916	8	0,1235
	θ	-1	-0,3980		
3	ϕ	-1	0,7916	8	0,0821
	θ	-1	-0,3980		
Vidurkis				8	0,0803



3.1.16 pav. Levenberg Marquardt algoritmo koeficientų paieškos grafikas, kai parametrų pradinė vertė yra 0



3.1.17 pav. Levenberg Marquardt algoritmo koeficientų paieškos grafikas, kai parametrų pradinė vertė yra 1



3.1.18 pav. Levenberg Marquardt algoritmo koeficientų paieškos grafikas, kai parametrų pradinė vertė yra -1

Pagal gautus rezultatus matyti, kad standartinio gradiento nusileidimo algoritmas nėra robastinis, kaip ir Levenberg – Marquardt algoritmas. Šių abiejų algoritmų parametrų vertės kinta, pakitus ieškomų parametrų pradinei vertei (žr. 3.1.4 ir 3.1.6 lenteles). Vienintelis patobulintas gradiento nusileidimo algoritmas yra robastinis, jo apskaičiuojamos parametrų vertės nepakinta, keičiantis parametrų pradinėms vertėms (žr. 3.1.5 lentelę). Vertinant algoritmus pagal jų greitaveiką, greičiausias algoritmas, suradęs ekstremumą ir jį atitinkančiu parametrų vertes ir patenkinęs stabdymo kriterijus, yra Levenberg Marquardt algoritmas. Lėčiausiai surastas ekstremumas šiuo algoritmu yra 0,34113 s. Lėčiausias algoritmas yra standartinis gradiento nusileidimo algoritmas, kuris lėčiausiai nustatė optimalius parametrus per 0,83986 s., kai patobulintas gradiento nusileidimo algoritmas lėčiausiai apskaičiavo optimalius parametrus per 0,7423 s. Vertinant įteracijų skaičių, tik standartinio gradiento nusileidimo algoritmas pasiekė maksimalų iteracijų skaičių, kuris yra 5000 iteracijų, ir kiekvieno bandymo metu iteracijų skaičius buvo virš 1000. Kai patobulinto gradiento nusileidimo algoritmo iteracijų skaičius kiekvieno bandymo metu nesiekė 1000, o Levenberg – Marquardt algoritmas, ieškant parametrų, neperžengė 100 iteracijų ribos.

Įvertinant visas ankščiau išvardintas priežastis, nuspręsta, kad parametrams surasti bus naudojamas patobulintas gradiento nusileidimo algoritmas. Nors Levenberg – Marquardt algoritmas yra greičiausias ir suranda *LLF()* funkcijos ekstremumą per mažiausią iteracijų skaičių, tačiau jis nėra robastinis ir optimalių parametrų vertės keičiasi priklausomai nuo pasirinktų pradinių parametrų verčių.

Nustatytas optimalus prognozavimo modelių parametrų paieškos algoritmas negali užtikrinti, kad nustatytos prognozavimo modelio parametrų vertės sąlygos mažiausią prognozavimo paklaidą. Pastarasis algoritmas užtikrina, kad bus sunaudota mažiausiai laiko ir resursų parametrų paieškai bei bus surastas globalus parametrų optimumas, remiantis prognozavimo modelio mokymo duomenimis.

3.1.6. Autoregresinių prognozavimo modelių apskaičiuoti parametrai

Šioje dalyje yra pateikiami apskaičiuotos autoregresinių prognozavimo modelių parametrų vertės, pagal kurias sudaromi prognozavimo modeliai ir atliekamas individualaus pastato suvartojamos elektros energijos prognozavimas. Kaip jau buvo minėta ankščiau, visiems statistiniams prognozavimo modelių parametrams apskaičiuoti yra naudojamas patobulintas gradiento nusileidimo algoritmas. Gauti rezultatai yra pateikiami prieduose žemiau (žr. 2 priedą).

Apskaičiavus nežinomų parametrų vertes pastebėta, kad tuomet, kai modelyje yra daugiau nei 2 nežinomi parametrai, modelio parametrų vertės priklauso nuo jų pradinių verčių (žr. 2 priedą). Dėl šios priežasties modelių, kurių parametrai kinta, kintant jų pradinėms vertėms, nustatant tiksliausią prognozavimo modelį, nagrinėjamos visos trys parametrų vertes.

3.1.7. Autoregresinių modelių patikimumo skaičiavimas, naudojant *AIC* ir *BIC* kriterijus

Modelio tinkamumas yra nustatomas, naudojant *AIC* ir *BIC* kriterijus. Autoregresinio modelio patikimumo įvertinimo kriterijus *AIC* apskaičiuojamas, naudojant standartinę *Matlab* paketo funkciją *aic()*. Kitas patikimumo kriterijus – *BIC* apskaičiuojamas, naudojant *Matlab* standartinę funkciją *bic()*. Aukščiau minėti kriterijai buvo apskaičiuoti visiems sudarytiems modeliams. Penki autoregresiniai prognozavimo modeliai su mažiausiomis *AIC* ir *BIC* kriterijų vertėmis yra pateikiami lentelėje žemiau (žr. 3.1.7 lentelę). Visų sudarytų autoregresinių prognozavimo modelių apskaičiuotų *AIC* ir *BIC* kriterijų vertės yra pateikiamos prieduose (žr. 3 priedą).

3.1.7 lentelė. Realizuotų modelių patikimumo kriterijų AIC ir BIC vertės

Modelis	AIC	BIC
AR(1)xAR$_{24}$(1)	5,04261801921986	5,05782718573381
AR(1)xAR$_{25}$(1)	5,12037039590225	5,13557956241619
AR(1)xAR$_{23}$(1)	5,12238327111553	5,13759243762947
AR(2)	5,18477198474381	5,19998115125776
IMA(1,1)	5,27149748597430	5,28164532971754

Remiantis gautais *AIC* kriterijais, patikimiausias modelis yra AR(1)xAR$_{24}$(1), kai *AIC* vertė yra – 5,0426. Remiantis *BIC* kriterijumi, patikimiausias modelis yra AR(1)xAR$_{24}$(1), kai *BIC* vertė yra – 5,0578. Įvertinus tiek *AIC*, tiek *BIC* kriteriju vertes, nustatyta, kad patikimiausias modelis yra AR(1)xAR$_{24}$(1). Tačiau nustatytas modelio patikimumas nereiškia, kad modelis yra tiksliausias, todėl, siekiant nustatyti tiksliausią modelį, buvo realizuoti visi aukščiau pateikti autoregresiniai prognozavimo modeliai.

3.1.8. Pastato suvartojamos elektros energijos prognozės sudarymas, naudojant autoregresinius prognozavimo metodus

Realizuotiems modeliams nustatyti, nežinomi parametrai yra pateikiami prieduose (žr. 2 priedą). Remiantis gautais rezultatais, buvo sudaryta nagrinėjamo objekto elektros energijos suvartojimo prognozė dešimčiai darbo dienų, kai prognozė buvo sudaroma valandą į priekį. Sudarytų autoregresinių prognozavimo modelių prognozės paklaidos išraiškos MSE ir MPE yra pateikiamos žemiau (žr. 3.1.8 lentelę) (žr. 3.1.1 ir 3.1.2 formules). ARMA modeliams, kuriuose nežinomų parametrų skaičius yra ne mažesnis, nei 3, prognozavimas yra atliekamas, esant skirtingoms pradinėms vertėms. Bendras prognozių skaičius per 10 parų yra 240.

$$MSE = \frac{\sum_{i=1}^{N}(y_i - x_i)^2}{N}; \quad (3.1.1)$$

$$MPE = \frac{\sum_{i=1}^{N}\frac{|y_i - x_i|}{x_i} * 100\%}{N}; \quad (3.1.2)$$

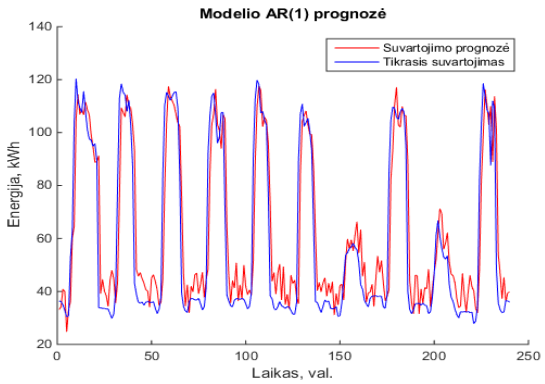
čia MSE – angl. mean square error, MPE – mean procent error. N – atliktų prognozių skaičius, y_i- energijos suvartojimo prognozė, x_i – tikrasis energijos suvartojimas.

Penki autoregresiniai prognozavimo modeliai su mažiausia prognozavimo paklaida yra atvaizduojami lentelėje žemiau (žr. 3.1.8 lentelę). Visų sudarytų autoregresinių prognozavimo modelių prognozavimo paklaidos yra pateikiamos prieduose (žr. 4 priedą).

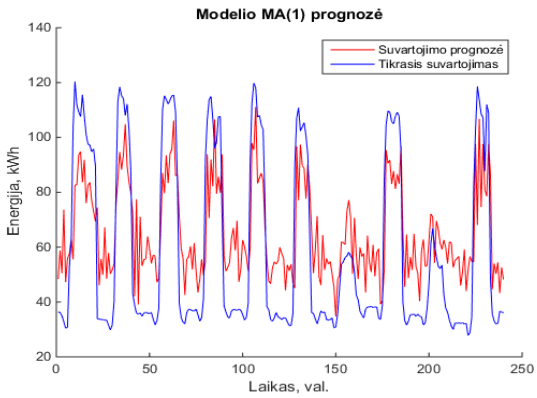
3.1.8 lentelė. AR, MA ir ARMA prognozavimo modelių prognozės paklaida

Modelis	**Parametras**	**Parametro vertė**	**Prognozavimo paklaida MSE %**	**Prognozavimo paklaida MPE %**
AR(1)xAR$_{24}$(1)	ϕ_s	0,8039	188,0376	17,3096
	Φ_{NS}	0,456		
ARMA(2,1)	ϕ_1	1,0898	200,7419	18,4191
	ϕ_2	-0,2098		
	θ_1	-0,1455		
ARMA (2,2)	ϕ_1	0,5751	201,0782	18,4741
	ϕ_2	0,1917		
	θ_1	-0,5636		
	θ_2	-0,2087		

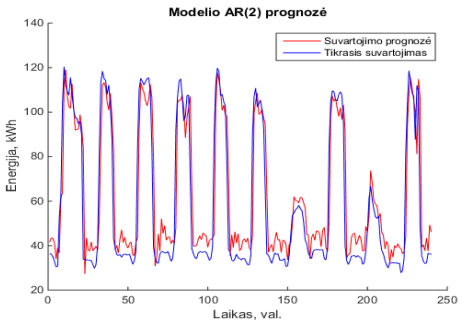
Gautos prognozės grafiškai yra atvaizduojamos žemiau pateiktuose grafikuose (žr. nuo 3.1.19 pav. iki 3.1.37 pav.). Statistinių prognozavimo modelių su integraline dedamąja gauti prognozavimo grafikai nėra atvaizduojami dėl didelės prognozavimo paklaidos, lyginant su prognozavimo modeliais be integralinės dedamosios (žr. 3.1.8 lentelę).



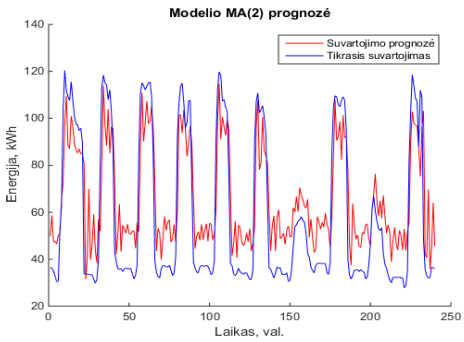
3.1.19 pav. AR(1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas



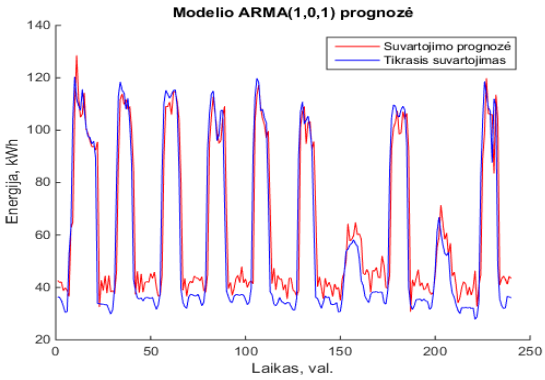
3.1.20 pav. MA(1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas



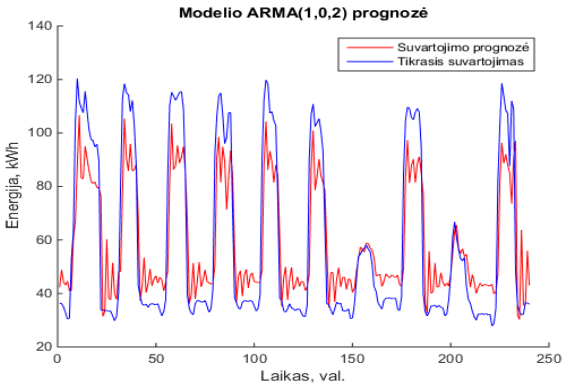
3.1.21 pav. AR(2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas



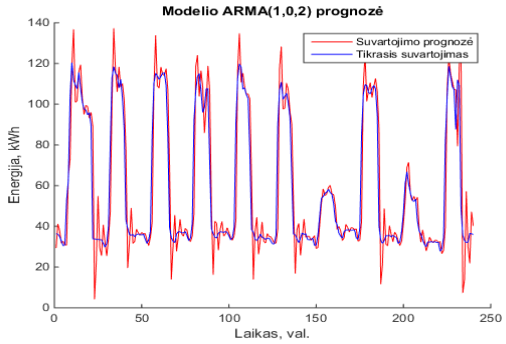
3.1.22 pav. MA(2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas



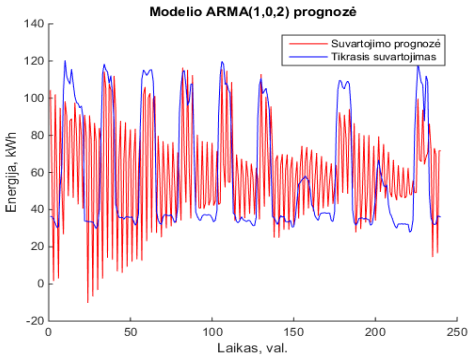
3.1.23 pav. ARMA(1,0,1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas



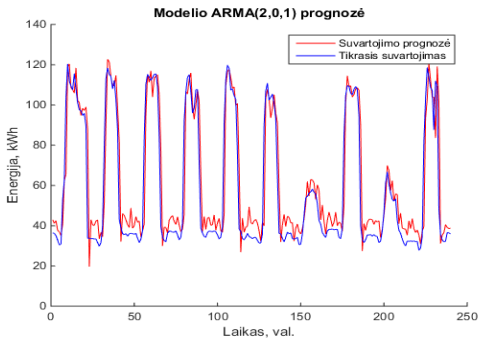
3.1.24 pav. ARMA(1,0,2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 0



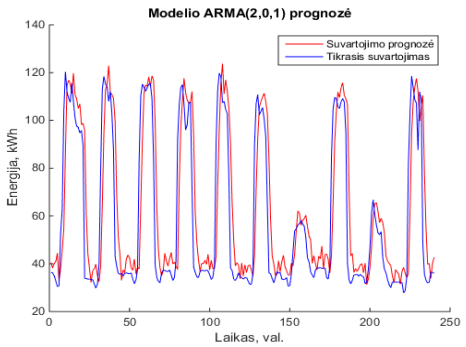
3.1.25 pav. ARMA(1,0,2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 1



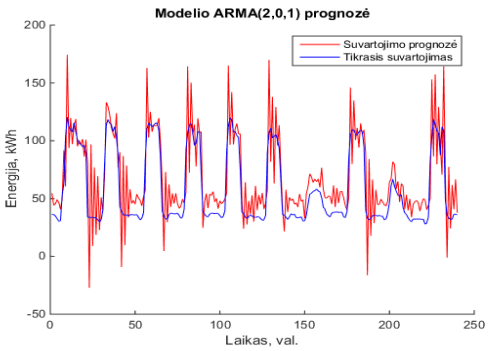
3.1.26 pav. ARMA(1,0,2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi -1



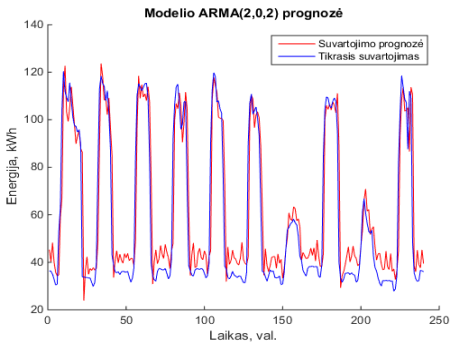
3.1.27 pav. ARMA(2,0,1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 0



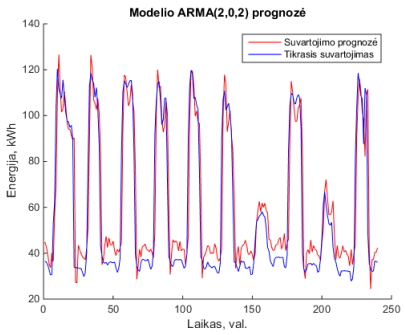
3.1.28 pav. ARMA(2,0,1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 1



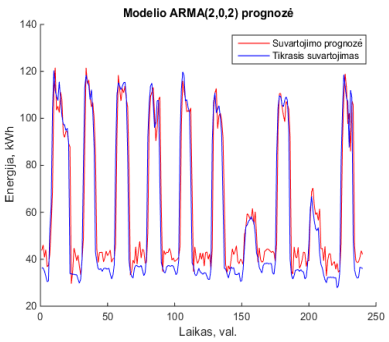
3.1.29 pav. ARMA(2,0,1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi -1



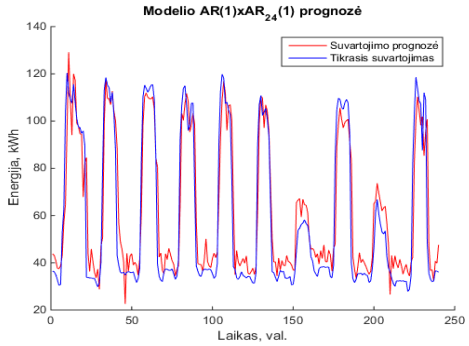
3.1.30 pav. ARMA(2,0,2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 0



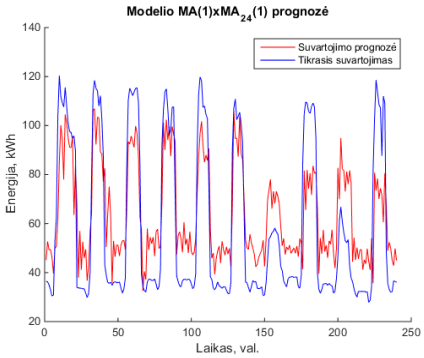
3.1.31 pav. ARMA(2,0,2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 1



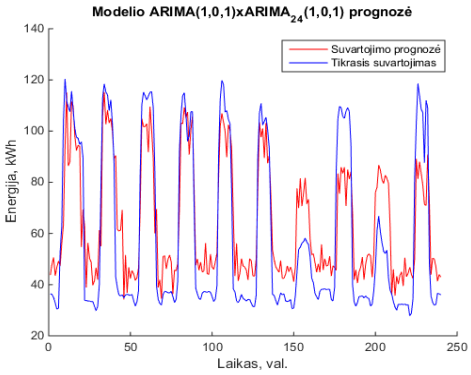
3.1.32 pav. ARMA(2,0,2) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi -1



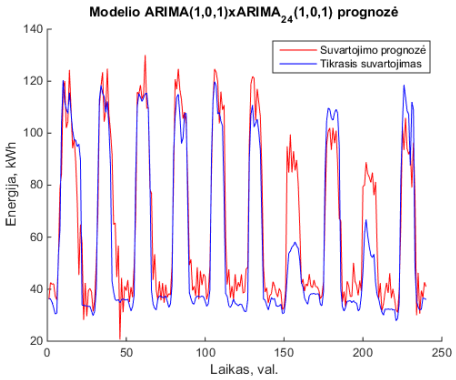
3.1.33 pav. AR(1)xAR$_{24}$(1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas



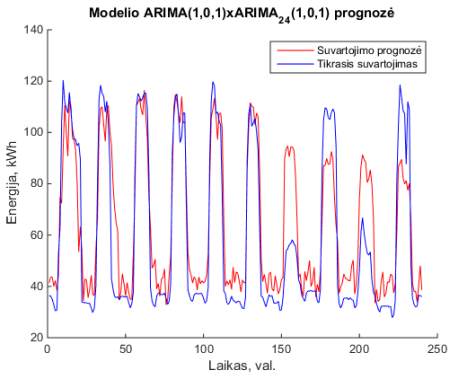
3.1.34 pav. MA(1)xMA$_{24}$(1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas



3.1.35 pav. ARMA(1,0,1)xARMA$_{24}$(1,0,1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi -1



3.1.36 pav. ARMA(1,0,1)xARMA$_{24}$(1,0,1) modelio energijos suvartojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 1



3.1.37 pav. ARMA(1,0,1)xARMA$_{24}$(1,0,1) modelio energijos suvatojimo prognozės ir tikrojo suvartojimo grafikas, kai parametrų pradinė vertė lygi 0

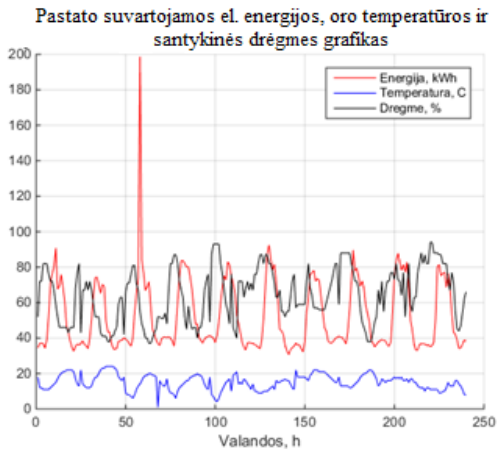
Pagal gautas prognozavimo paklaidas matosi, kad mažiausia prognozavimo paklaida susidaro tuomet, kai yra naudojamas AR(1)xAR_{24}(1) prognozavimo modelis, turintis sezoninę 24 val. dedamąją (žr. 3.1.8 lentelę).

Vertinant gautą prognozavimo paklaidą ir apskaičiuotą modelio patikimumą, daroma išvada, kad optimalus autoregresinis prognozavimo modelis yra AR(1)xAR_{24}(1), turintis sezoninę 24 val. dedamąją ir kurio parametrai rasti, naudojant maksimalios tikimybės metodą ir patobulintą gradiento nusileidimo algoritmą.

3.2. Prognozavimo modelis su išoriniais kintamaisiais

Siekiant sudaryti prognozavimo modelį su išoriniais kintamaisiais (ARMAX ir ARIMAX), būtina ištirti nagrinėjamo objekto suvartojamos elektros energijos koreliaciją su išoriniais kintamaisiais, kurie bus naudojami modelyje. Šiame darbe tiriamas ryšys tarp pastato suvartotos elektros energijos kiekio ir oro prognozės duomenų – temperatūros ir santykinės drėgmės. Koreliacija tarp išvardintų duomenų buvo nustatoma per 40 darbo dienų. Koreliacijai nustatyti buvo naudojamas *Matlab* programavimo paketas.

Atlikus tyrimą, nustatyta, kad koreliacija tarp pastato suvartojamos elektros energijos ir oro temperatūros yra 0,193551. Tuo pačiu koreliacijos koeficientas tarp pastato suvartotos elektros energijos kiekio ir oro santykinės drėgmės yra -0,0440654. Temperatūra, aplinkos santykinė drėgme ir pastato suvartojamos elektros energijos kiekis per 40 darbo dienų yra atvaizduojami grafiškai (žr. 3.2.1 pav.).



3.2.1 pav. LEI pastato suvartojamos elektros energijos, aplinkos oro temperatūros ir santykinės drėgmės grafikas

Pagal gautas koreliacijos koeficientų vertes galima teigti, kad nagrinėjamo objekto suvartojamos elektros energijos kiekis turi mažą ryšį su išoriniais parametrais, tokiais kaip oro temperatūra ir oro santykinė drėgmė. Dėl mažo ryšio tarp prognozuojamo dydžio ir išorinių kintamųjų. buvo atsisakyta realizuoti ARMAX ir ARIMAX statistinius prognozavimo modelius.

3.3. Optimalaus eksponentinio išlyginimo prognozavimo modelio nustatymas

Atlikus literatūros analizę, eksponentinio išlyginimo prognozavimo modelis yra tinkamas modelis teorinio objekto energijos suvartojimui atlikti. Siekiant nustatyti optimalų eksponentinio išlyginimo modelį, būtina nustatyti modelio parametro vertę, kuri atitiks minimalią prognozavimo paklaidą. Taip

pat būtina nustatyti, kokį koeficiento paieškos algoritmą naudojant koeficiento vertė bus rasta per mažiausią iteracijų skaičių.

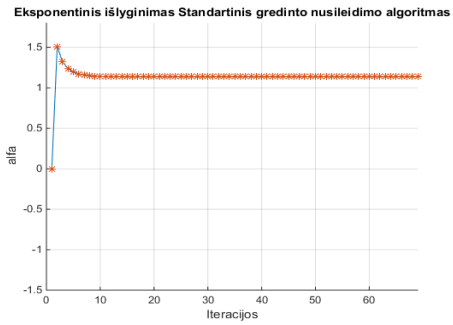
Eksponentinio modelio realizacija yra pateikiama laikmenoje (žr. „*EksModel.m*“). Kaip ir kiti statistiniai prognozavimo modeliai, taip ir eksponentinio išlyginimo modelis buvo realizuojamas prognozuoti tik darbo dienomis suvartojamą elektros energiją. Taip pat, ieškant modelio nežinomo parametro vertės, buvo naudojamas maksimalios tikimybės metodas ir tie patys duomenys, kaip ir kitose prognozavimo modeliuose.

Kadangi *LLF()* funkcijos vertė priklauso tik nuo išlyginimo koeficiento α vertės, tai sprendžiamas paprastas optimizavimo uždavinys. *LLF()* funkcijos ekstremumui surasti yra naudojami keli standartiniai optimizavimo algoritmai: gradiento nusileidimo (angl. *gradient descent*) algoritmas, dalinimo pusiau (angl. *half spliting*) algoritmas, auksinio pjūvio (angl. *golden section*) algoritmas, penkių taškų algoritmas, Landveberio optimizacijos (angl. *Landweber iteration*) algoritmas. Taip pat buvo naudojamas autoriaus sudarytas algoritmas, kuris yra paremtas gradiento nusileidimo algoritmu. Realizuoti algoritmai yra pateikiami laikmenoje (žr. „*EksModel.m*“). Gradiento nusileidimo algoritmo ir autoriaus patobulinto gradiento nusileidimo algoritmo koeficiento α pradinė vertė yra 0, o žingsnis, su kuriuo yra atnaujinama koeficiento vertė, atitinkamai yra 0,00001. Abiejose algoritmuose maksimalus iteracijų skaičius yra 5000.

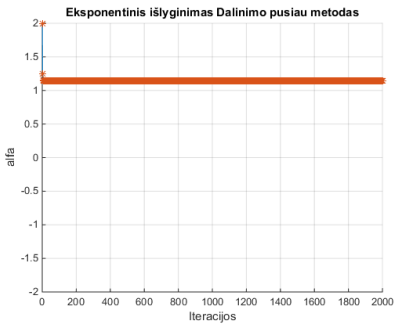
Visi aukščiau paminėti ekstremumo paieškos algoritmai buvo tarpusavyje palyginami pagal modelio ekstremumo suradimo laiką ir iteracijų skaičių. Gauti rezultatai yra pateikiami lentelėje žemiau (žr. 3.3.1 lentelę). Parametro α keitimosi grafikai, pagal skirtingus paieškos algoritmus, yra pateikiami žemiau (žr. 3.3.1, 3.3.2, 3.3.3, 3.3.4, 3.3.4 ir 3.3.5 pav.) .

3.3.1 lentelė. Eksponentinio išlyginimo parametrų paieškos algoritmų palyginimas

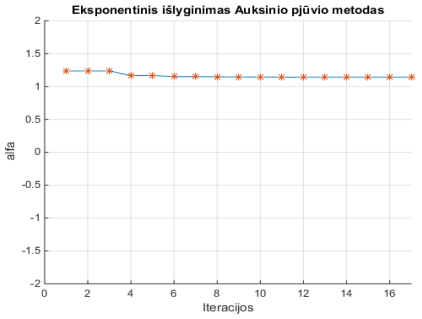
Algoritmai	**Parametro vertė**	**Iteracijos**	**Skaičiavimo laikas, *s***
Gradiento nusileidimo	1,1374	69	0,0361
Pjūvio pusiau	1,1378	2000	0,2313
Auksinio pjūvio	1,1409	17	0,2265
Penkių taškų	1,1377	11	1,4015
Landveberio optimizacijos	1,1385	88	0,0760
Patobulinto gradiento nusileidimo	1,1385	119	0,0289



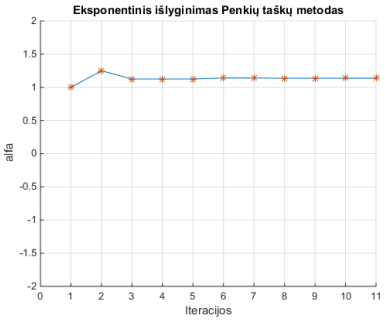
3.3.1 pav. Eksponentinio išlyginimo α parametro pokytis, naudojant standartinį gradiento nusileidimo algoritmą



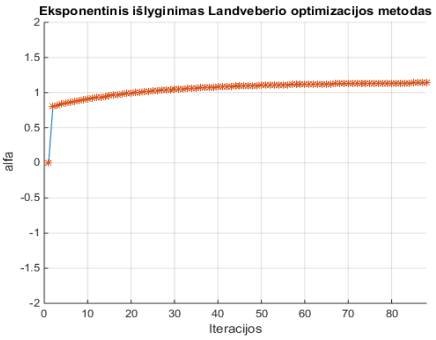
3.3.2 pav. Eksponentinio išlyginimo α parametro pokytis, naudojant dalinimo pusiau algoritmą



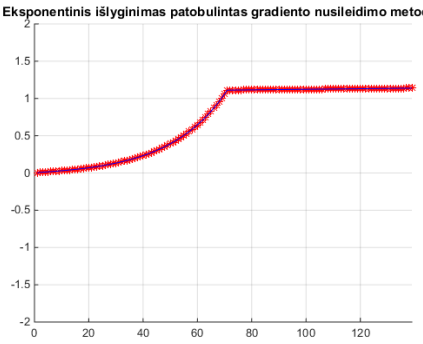
3.3.3 pav. Eksponentinio išlyginimo *α* parametro pokytis, naudojant auksinio pjūvio algoritmą



3.3.4 pav. Eksponentinio išlyginimo *α* parametro pokytis, naudojant penkių taškų algoritmą



3.3.5 pav. Eksponentinio išlyginimo *α* parametro pokytis, naudojant Landveberio optimizacijos algoritmą

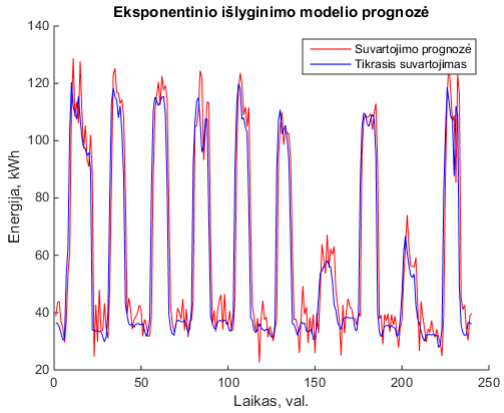


3.3.6 pav. Eksponentinio išlyginimo *α* parametro pokytis, naudojant patobulintą gradiento nusileidimo algoritmą

Remiantis gautais rezultatais, eksponentinio išlyginimo prognozavimo modelio nežinomam kintamajam α nustatyti bus naudojamas patobulinto gradiento nusileidimo algoritmas, nes naudojant pastarąjį algoritmą ekstremumas rastas greičiausiai. Optimalus ekstremumo paieškos algoritmas nėra parenkamas pagal iteracijų skaičių, nes modeliai su mažiausių iteracijų skaičiumi užtruko ilgiausiai

(žr. 3.3.1 lentelę). Patobulinto gradiento nusileidimo algoritmas rado ieškomo koeficiento vertę, artimą parametro vertėms, kurias rado kiti ekstremumo paieškos algoritmai.

Nustatyta eksponentinio išlyginimo prognozavimo modelio parametro vertė bus naudojama, sudarant pastato suvartojamos elektros energijos prognozę. Optimalaus eksponentinio išlyginimo prognozavimo modelio su pasirinktu α parametru prognozavimo paklaida per 240 prognozavimo taškus MSE yra 242,8796 kWh, kai MPE yra 22,8123%. Gauta prognozė per 240 taškus yra atvaizduojama kartu su realiu pastato elektros energijos suvartojimu žemiau (žr. 3.3.7 pav.).



3.3.7 pav. Eksponentinio išlyginimo modelio energijos suvartojimo prognozė ir tikrojo suvartojimo grafikas

Optimalus eksponentinio išlyginimo prognozavimo modelis – tai modelis, kurio optimali α parametro vertė yra nustatoma, naudojant patobulintą gradiento nusileidimo algoritmą, kai parametro vertė yra 1,1385. Nustatyto optimalaus eksponentinio prognozavimo modelio prognozavimo paklaida, prognozuojant suvartojimą 10 – čiai d. d., yra 242,8796 kWh, kai MPE yra 22,8123%.

3.4. Pastato suvartojamos elektros energijos prognozės sudarymas, naudojant dirbtinio intelekto prognozavimo metodus

Kaip jau buvo minėta ankščiau, šiame baigiamajame darbe palyginimui su statistiniais prognozavimo modeliais buvo sudarytas ir dirbtinio intelekto prognozavimo modelis – dirbtinis neuroninis tinklas, detalesnė informacija apie naudojamą neuroninį tinklą yra pateikta 37-ame šaltinyje [36]. Šiame darbe buvo sudaryti dirbtiniai neuroniniai tinklai, kurie buvo apmokyti: Levenberg – Marquardt algoritmu, jungtinio gradiento nusileidimo algoritmu su Reeves – Fletcher atnaujinimo metodu ir gradiento nusileidimo algoritmu. Taip pat buvo atliekami bandymai su skirtingomis tinklo struktūromis, keičiant paslėptų sluoksnių skaičių bei keičiant paslėptuose sluoksniuose esančių neuronų skaičių. Neuroninių tinklų struktūra buvo keičiama, keičiant tinklo įėjimų skaičių, naudojant tris įėjimus – paros valanda, temperatūra ir drėgmė. Taip pat buvo naudojamas keturių įėjimų konfigūracijos tinklas, kur įėjimai yra: paros valanda, temperatūra, drėgmė ir praeitos valandos energijos suvartojimas. Tinklai tarpusavyje buvo lyginami pagal prognozavimo paklaidą, mokymo trukmę ir regresiją (žr. 5 priedą). Prognozavimas buvo atliekamas tokiam pat laikotarpiui kaip ir naudojant statistinius prognozavimo modelius.

Siekiant nustatyti optimalią dirbtinio neuroninio tinklo konfigūraciją, taip pat buvo keičiama dirbtinio neuroninio tinklo paslėptų neuronų aktyvacijos funkcija. Buvo testuojamos neuronų aktyvacijos funkcijos – tangentinis sigmoidas (*tansig*) ir logaritminis sigmoidas (*logsig*). Nustatant optimalią paslėptų neuronų aktyvacijos funkciją, buvo atliekami bandymai su neuroninių tinklų apmokytu

Levenberg – Marquardt algoritmu. Pastarasis eksperimentas buvo atiekamas su vieno paslėpto sluoksnio neuroninių tinklu kuriame yra 10 paslėptų neuronų ir su dvieju paslėptų sluoksniu neuroniniu tinklu, kuriame kiekviename paslėptame sluoksnyje buvo po 10 neuronų. Gauti rezultatai yra pateikiami lentelėje žemiau (žr. 3.4.1 lentelę).

3.4.1 lentelė. Optimali dintinio neuroninio tinklo paslėptų neuronų aktyvacijos funkcija

Aktyvacijos funkcija	**MPE,%**	**Laikas, s**	**Regresija, %**	**ANN struktūra**
logsig	5,810	0,3648	0,93246	10
logsig	5,747	0,1334	0,92727	10 ir 10
tansig	5,237	0,1708	0,93041	10
tansig	5,466	0,3260	0,93639	10 ir 10

Remiantis gautais rezultatais (žr. 3.4.1 lentelę), optimali paslėptų neuronų aktyvacijos funkcija yra tangentinio sigmoido funkcija (*tansig*). Pastaroji funkcija yra optimali, nes esant vieno ir dvejų paslėptų neuronų sluoksnio neuroniniai tinklai yra tikslesni, nei neuroniniai tinklai su logaritminio sigmoido funkcija. Toliau nustatant optimalią dirbtinio neuroninio tinklo optimalią konfigūraciją, paslėptų neuronų aktyvacijos funkcija bus naudojama tangentinio sigmoido aktyvacijos funkcija.

Penki tiksliausi prognozavimo modeliai, esant skirtingiems tinklų apmokymo algoritmams ir skirtingam įėjimo kintamųjų skaičiui yra pateikiami prieduose (žr. 5 priedą). Esant konkrečiai tinklo struktūrai, tinklas yra apmokomas tris kartu ir prognozavimas yra atliekamas tris kartus, todėl prieduose pateikiamos prognozavimo rezultatų vidutinės trijų bandymų vertės.

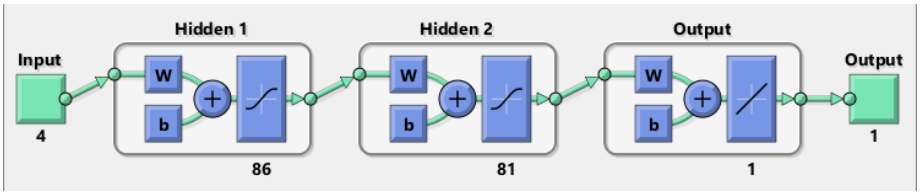
Bendram palyginimui tarp dirbtinio intelekto prognozavimo modelių mažiausios prognozavimo paklaidos yra pateiktos lentelėje žemiau, kai tinklai apmokyti skirtingais algoritmais (žr. 3.4.2 lentelę).

3.4.2 lentelė. Dirbtinių Fneuroninių tinklų mažiausią prognozavimo paklaida, esant skirtingiems tinklo mokymo algoritmams

Tinklo apmokymo Algritmas	**Neuronų skaičius pirmajame sluoksnyje**	**Neuronų skaičius antrajame sluoksnyje**	**MPE, %**	**MSE, *kWh***	**Regresija, *%***	**Mokymo trukmė, *s***	**Tinklo įėjimų skaičius**
Levenber - Marquardt	86	81	1,914	20,386	0,92721	0,7368	4
Gradiento nusileidimo algoritmas	91	6	1,930	20,556	0,92553	0,36595	4
Jungtinio gradiento algoritmas su Reeves – Fletcher atnaujinimo metodu	41	11	1,935	20,606	0,92194	0,20937	4

Iš gautų rezultatų matosi, kad mažiausią prognozavimo paklaida susidaro tuomet, kai dirbtinio neuroninio tinklo struktūra yra sudaryta iš dviejų paslėptų sluoksnių, kai pirmajame sluoksnyje yra

86 neuronai, o antrajame sluoksnyje yra 81 neuronas, taip pat, kai tinklas yra apmokytas Levenber - Marquardt algoritmu ir yra naudojami keturi įėjimo kintamieji. Tokios struktūros tinklo prognozavimo paklaida MSE yra 20,38696 kWh, o MPE yra 1,9148%, kai tinklo struktūra yra atvaizduojama žemiau pateiktame grafike (žr. 3.4.1 pav.).



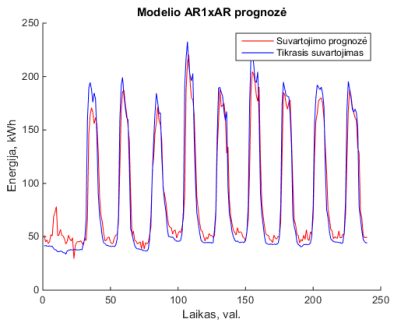
3.4.1 pav. Dirbtinio neuroninio tinklo prognozavimo modelio konfigūracija atitinkanti mažiausią prognozavimo paklaidą

3.5. Statistinių prognozavimo modelių ir dirbtinio intelekto prognozavimo modelio prognozių palyginimas

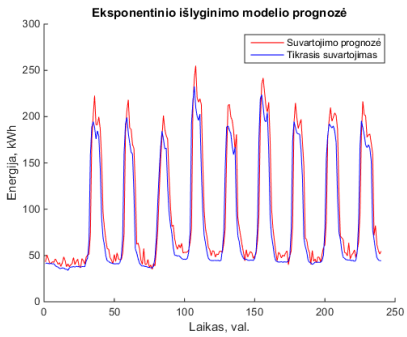
Vertinant statistinių ir dirbtinio intelekto prognozavimo modelius, galima vienareikšmiškai teigti, kad, esant vienodoms tyrimo sąlygoms, dirbtinio intelekto prognozavimo modelis yra tikslesnis už statistinius prognozavimo modelius, tuomet kai prognozuojamas suvartojimas nėra nutolęs nuo apmokymo duomenų daugiau nei 10 d.d. Tiksliausio statistinio prognozavimo modelio prognozavimo paklaida MSE yra 188,0376 kWh, o MPE yra 17,3096%, kai modelis yra AR(1)xAR_{24}(1) su sezonine dedamąja. Eksponentinio išlyginimo prognozavimo modelio mažiausia paklaida MSE yra 242,8796 kWh, o MPE yra 22,8123%. Tiksliausio dirbtinio neuroninio tinklo prognozavimo modelio mažiausia paklaida MSE yra 20,38696 kWh, o MPE yra 1,9148% (žr. 3.4.1 pav.). Tačiau norint pilnai įvertinti prognozavimo modelių tinkamumą, būtina įvertinti modelio apmokymo arba koeficientų radimo laiką. Iš lentelės aukščiau (žr. 3.4.2 lentelę) matosi, kad dirbtinio neuroninio tinklo prognozavimo modelio atitinkančio mažiausią prognozavimo paklaidą, apmokymas užtruko 0,7368 s., kai AR(1)xAR_{24}(1) prognozavimo parametrų paieška užtruko 0,4153 s., o eksponentinio išlyginimo prognozavimo modelio parametro α radimas užtruko 0,0289 s. Remiantis gautais rezultatais, dirbtinio neuroninio tinklo modelio apmokymas įvyko ilgiausiai.

3.5.1. Statistinių prognozavimo ir dirbtinio intelekto modeliu energijos suvartojimo prognozavimas po 3 mėnesių nuo modelio sudarymo

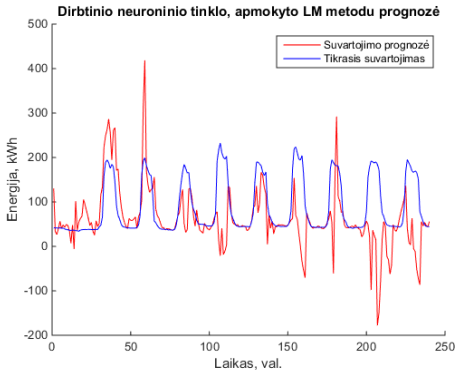
Lyginant statistinius prognozavimo modelius su dirbtinio neuroninio tinklo prognozavimo modeliais, būtina įvertinti modelių gebėjimą prognozuoti energijos suvartojimą, nutolus nuo duomenų pagal kuriuos buvo apmokyti modeliai. Patikrinti šią modelių savybę, prognozavimo modeliai atliko prognozę, remiantis duomenimis, nutolusiais per 3 mėnesius (laiko atžvilgiu) nuo tų duomenų, pagal kuriuos buvo apmokyti prognozavimo modeliai. Aukščiau pateiktiems tiksliausiems modeliams ištirti, buvo naudojama duomenų imtis, susidedanti iš 240 taškų (t.y. 10 darbo dienų) nuo 2017 11 01 iki 2017 11 14. Sudarant AR(1)xAR_{24}(1) modelį, taip pat yra naudojami 24 papildomi taškai, kurie yra reikalingi modelio sezoninei dedamajai. Gauti rezultatai yra pateikiami žemiau esančiuose grafikuose (žr. 3.5.1, 3.5.2 ir 3.5.3 pav.).



3.5.1 pav. AR1xAR1 modelio prognozė ir tikrojo suvartojimo grafikas nuo 2017 11 01 iki 2017 11 14



3.5.2 pav. Eksponentinio išlyginimo modelio prognozė ir tikrojo suvartojimo grafikas nuo 2017 11 01 iki 2017 11 14



3.5.3 pav. Dirbtinio neuroninio tinklo modelio prognozė ir tikrojo suvartojimo grafikas nuo 2017 11 01 iki 2017 11 14

Statistinio prognozavimo modelio AR(1)xAR$_{24}$(1) su sezonine dedamąja prognozavimo paklaida MSE per 240 taškų yra 275,7267 kWh, kai MPE yra 25,8975%. Eksponentinio išlyginimo prognozavimo modelio prognozavimo paklaida MSE 364,7084 kWh, kai MPE yra 34,2550%. Dirbtinio neuroninio tinklo prognozavimo paklaida MSE 1323,705 kWh, kai MPE yra 124,3283%. Iš gautų rezultatų matosi, kad statistinis prognozavimo modelis AR(1)xAR$_{24}$(1) su sezonine dedamąja yra tiksliausias. Taip pat remiantis gautais rezultatais galima daryti išvadą, kad tolstant (laiko atžvilgiu) prognozuojamam dydžiui nuo duomenų, pagal kuriuos yra apmokomi prognozavimo modeliai, prognozavimo paklaida didėja ir labiausiai daro įtaką dirbtinio neuroninio tinklo prognozavimo tikslumui. Nutolus prognozuojamam dydžiui nuo duomenų, pagal kuriuos buvo apmokyti modeliai per 3 mėnesius, dirbtinio neroninio tinklo prognozės paklaida padidėjo per 6394,82 %, statistinio AR(1)xAR$_{24}$(1) modelio prognozės paklaida padidėjo per 136,4858 %, o eksponentinio išlyginimo modelio prognozės paklaida padidėjo per 150,1602 %.

3.5.2. Statistinių prognozavimo modelių ir dirbtinio intelekto prognozavimo modelio prognozavimas 2, 5, 12 ir 24 val. į priekį

AR(1)xAR$_{24}$(1), eksponentinio išlyginimo ir dirbtinio neuroninio tinklo prognozavimo modeliu prognozavimo rezultatai yra pateikiami lentelėje žemiau (žr. 3.5.1 ir 3.5.2 lenteles). Prognozavimo į

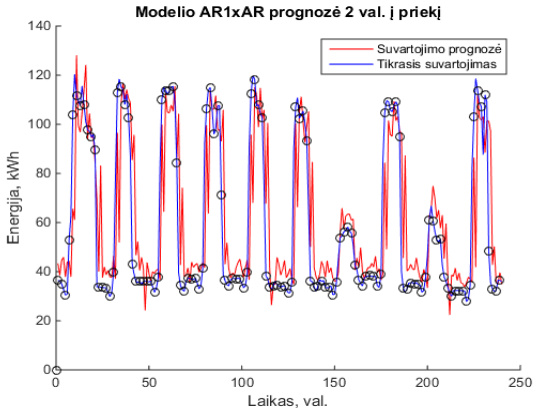
priekį realizacijos kodas yra pateiktas laikmenoje (žr. „*ModelCompareThirdExper.m*“). Atliktų prognozių ir tikrojo suvartojimo grafikai yra pateikiami žemiau (žr. nuo 3.5.4 iki 3.5.15 pav.)

3.5.1 lentelė. Prognozavimo modelių prognozių paklaida MSE, prognozuojant kelioms valandoms į priekį

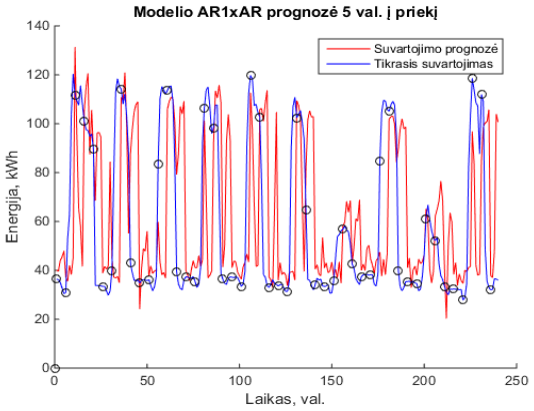
Prognozavimo modelis	**Prognozavimo žingsnis, val**			
	2	**5**	**12**	**24**
	MSE, kWh	**MSE, kWh**	**MSE, kWh**	**MSE, kWh**
Statistinis AR(1)xAR$_{24}$(1)	320,5940	574,0796	335,1506	389,4418
Eksponentinis išlyginimas	287,7151	484,4495	844,3708	600,4520
Dirbtinis neuroninis tinklas	748,1740	630,3133	744,6505	936,7530

3.5.2 lentelė. Prognozavimo modelių prognozių paklaida MPE, prognozuojant kelioms valandoms į priekį

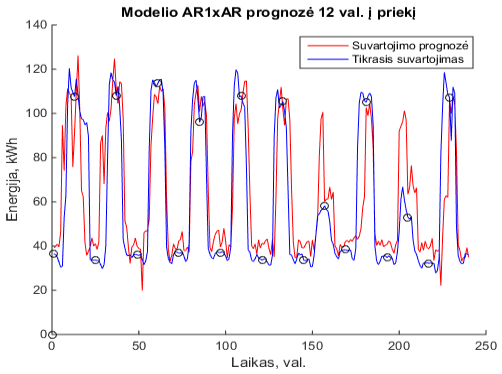
Prognozavimo modelis	**Prognozavimo žingsnis, val**			
	2	**5**	**12**	**24**
	MPE,%	**MPE,%**	**MPE,%**	**MPE,%**
Statistinis AR(1)xAR$_{24}$(1)	30,1116	53,9201	31,4788	36,5781
Eksponentinis išlyginimas	27,0235	45,5017	79,3071	56,3971
Dirbtinis neuroninis tinklas	70,2718	59,2018	69,9409	87,9840



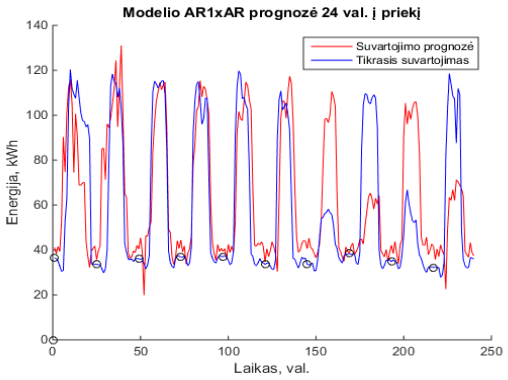
3.5.4 pav. Medelio AR(1)xAR24(1) prognozavimas 2 val. į priekį ir tikrojo suvartojimo grafikas



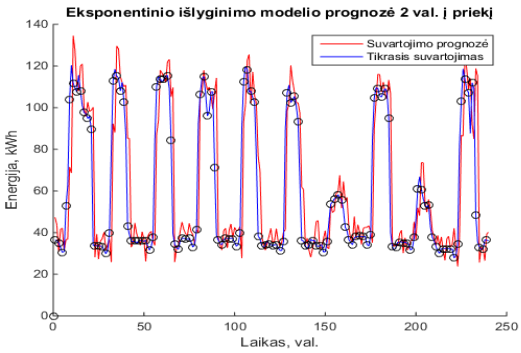
3.5.5 pav. Medelio AR(1)xAR24(1) prognozavimas 5 val. į priekį ir tikrojo suvartojimo grafikas



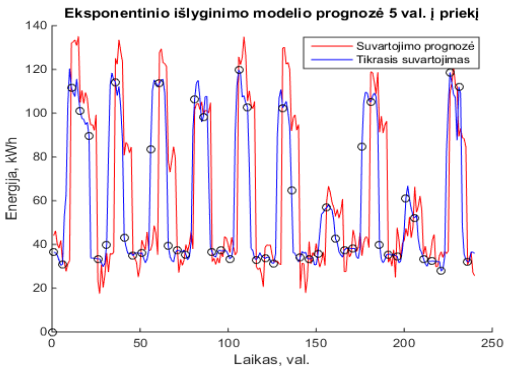
3.5.6 pav. Medelio AR(1)xAR$_{24}$(1) prognozavimas 12 val. į priekį ir tikrojo suvartojimo grafikas



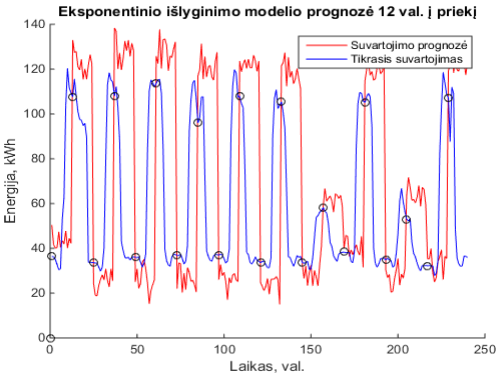
3.5.7 pav. Medelio AR(1)xAR$_{24}$(1) prognozavimas 24 val. į priekį ir tikrojo suvartojimo grafikas



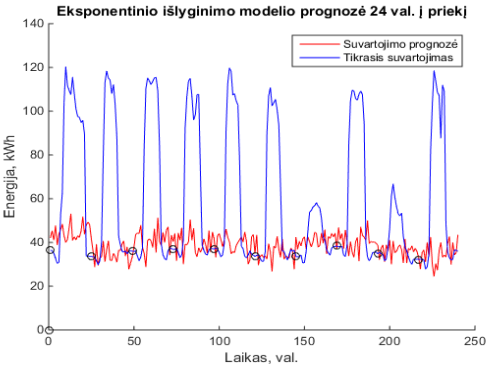
3.5.8 pav. Eksponentinio išlyginimo modelio prognozavimas 2 val. į priekį ir tikrojo suvartojimo grafikas



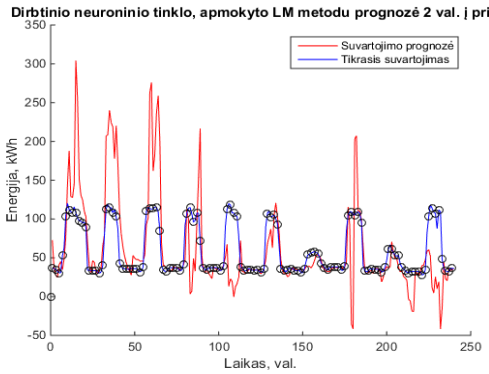
3.5.9 pav. Eksponentinio išlyginimo modelio prognozavimas 5 val. į priekį ir tikrojo suvartojimo grafikas



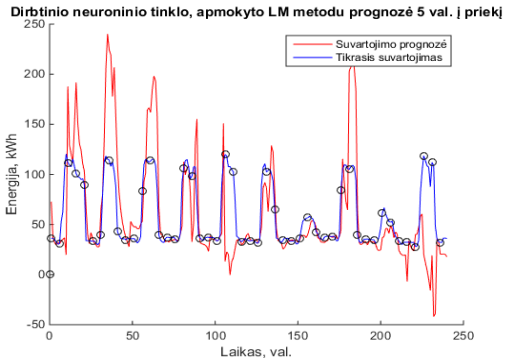
3.5.10 pav. Eksponentinio išlyginimo modelio prognozavimas 12 val. į priekį ir tikrojo suvartojimo grafikas



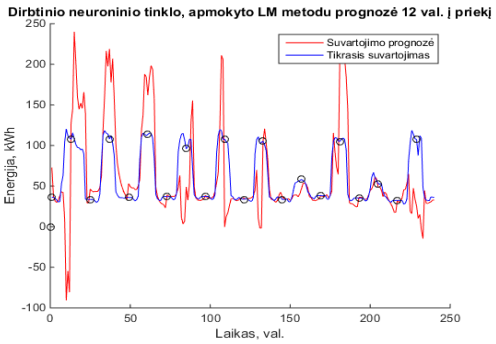
3.5.11 pav. Eksponentinio išlyginimo modelio prognozavimas 24 val. į priekį ir tikrojo suvartojimo grafikas



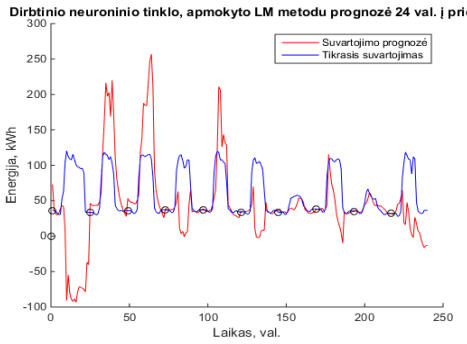
3.5.12 pav. Dirbtinio neuroninio tinklo modelio prognozavimas 2 val. į priekį ir tikrojo suvartojimo grafikas



3.5.13 pav. Dirbtinio neuroninio tinklo modelio prognozavimas 5 val. į priekį ir tikrojo suvartojimo grafikas



3.5.14 pav. Dirbtinio neuroninio tinklo modelio prognozavimas 12 val. į priekį ir tikrojo suvartojimo grafikas



3.5.15 pav. Dirbtinio neuroninio tinklo modelio prognozavimas 24 val. į priekį ir tikrojo suvartojimo grafikas

Pagal gautus rezultatus atliekant prognozę kelioms valandoms į priekį, prognozavimo paklaida didėja visų modelių prognozėse, kai prognozavimo žingsnis didėja (žr. 3.5.1 ir 3.5.2 lenteles). Tačiau prognozavimo paklaida, didėjant prognozavimo žingsniui, didėja lėčiausiai, naudojant statistinį AR(1)xAR$_{24}$(1) su sezoninę dedamąja. Prasčiausiai į tokį prognozavimo aplinkybių pasikeitimą sureagavo dirbtinio neuroninio tinklo prognozavimo modelis (žr. 3.5.1 ir 3.5.2 lenteles). Iš to seka išvada, kad statistinis AR(1)xAR$_{24}$(1) prognozavimo modelis yra labiau nepriklausomas ir universalus, nei eksponentinio išlyginimo ir dirbtinio neuroninio tinklo prognozavimo modeliai, nes gali prognozuoti energijos suvartojimą nepriklausomai nuo išorinių aplinkybių ir net su tam tikrais informaciniais tarpais.

3.6. Optimalaus neraiškios logikos reguliatoriaus nustatymas.

Šiame baigiamajame darbe surasto optimalaus prognozavimo modelio parametrai atnaujinami naudojant neraiškios logikos reguliatorių. Tam, kad parametrai būtų atnaujinti efektyviai, nustatoma optimali neraiškios logikos reguliatoriaus konfigūracija.

Kadangi nustatytas optimalus prognozavimo modelis turi daugiau nei vieną parametrą, pagal kurį yra sudaroma prognozė, tai neraiškios logikos reguliatorius yra sudaromas pagal Mamdani neraiškios logikos reguliatoriaus realizavimo algoritmą. Mamdani algoritmas yra pasirinktas, nes pastarasis

algoritmas gali realizuoti neraiškios logikos reguliatorių su keliais išėjimo kintamaisiais, ko negalima padaryti naudojant Sugeno neraiškios logikos reguliatoriaus realizavimo algoritmą.

Nustatant optimalų neraiškios logikos reguliatorių sudaryta pagal Mamdani algoritmą, buvo nustatomi optimalūs reguliatoriaus parametrai:

- Optimalus lingvistinių kintamųjų skaičius. Testuojami lingvistiniai kintamieji keičiami vienodai fuzzy reguliatoriaus įėjimo ir išėjimo kintamiesiems. Testavimas atliekamas bandant reguliatorių esant trims, penkiems ir septyniems lingvistiniams kintamiesiems.

- Optimali priklausomumo funkcija. Ieškant optimalios priklausomumo funkcijos testuojamos priklausomumo funkcijos - *trimf, trapmf, gaussmf* ir *gbellmf* .

- Optimali žinių bazės konfigūracija. Žinių bazė taisyklų skaičius priklauso nuo naudojamų lingvistinių kintamųjų skaičiaus. Pastarasis neraiškios logikos parametras yra pagrindinis elementas, perteikiantis žmogaus ekspertines žinias į skaitmeninę sistemą, pagal kurią yra sudaromas optimalus prognozavimo modelio parametrų atnaujinimas. Testuojamų žinių bazių konfigūracijos buvo sudarytos taip, kad esant skirtingam įėjimo kintamojo ženklui, išėjimo kintamieji kistų simetriškai.

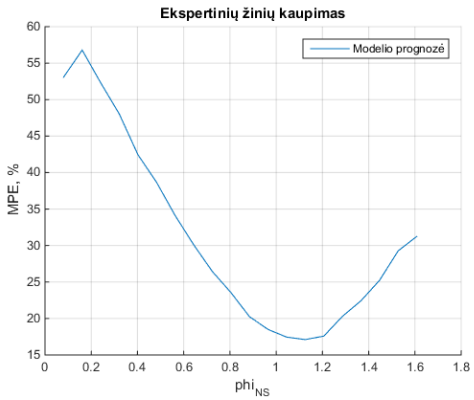
Siekiant sukaupti teisingas ekspertines žinias, kurios bus perteiktos į neraiškios logikos reguliatorių, atliekamas tyrimas, kaip keičiant optimalaus prognozavimo modelio parametrus, keičiasi optimalaus prognozavimo modelio prognozavimo paklaida. Atliekamas bandymas keičiant parametro vertes tam tikru fiksuotu žingsniu ir stebima kaip keičiasi prognozavimo paklaida. Remiantis gautais rezultatais sukauptos ekspertinės žinios yra perkeliamos į neraiškios logikos reguliatorių naudojant lingvistinius kintamuosius ir žinių bazę.

Testuojamame neraiškios logikos reguliatoriuje taip pat buvo parametrų, kurie nebuvo keičiami testavimo metų, nes pastarieji parametrai turi mažesne įtaka neraiškios logikos reguliatoriaus veikimui. Todėl reguliatoriaus sudėtinės sąlygos buvo gaunamos pagal *IR* minimumo bei *ARBA* maksimumo metodus. Visuose neraiškios logikos reguliatoriuose naudojami metodai – minimumo implikacijos metodas, maksimumo agregacijos metodas bei kintamųjų defuzifikavimui naudojamas masės centro metodas.

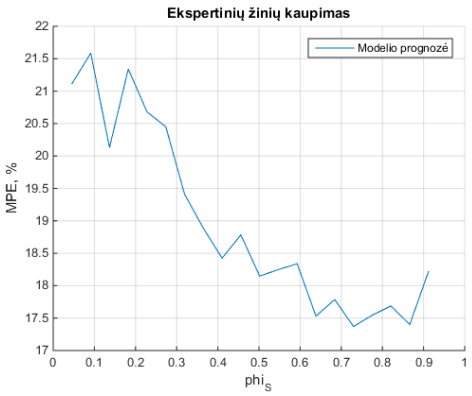
Sudaryti neraiškios logikos reguliatoriai lyginami tarpusavyje pagal pasirinkto prognozavimo modelio prognozavimo MPE.

3.6.1. Ekspertinių žinių kaupimas.

Kaupiamos ekspertinės žinios, kaip keičiasi optimalaus prognozavimo modelio prognozavimo paklaida, kečiant parametrus. Nustatyto optimalus prognozavimo modelio $AR(1)xAR_{24}(1)$ parametras φ_{NS} yra keičiamas pagal fiksuotą žingsnį 0,0456, kai Φ_S yra keičiamas pagal fiksuotą žingsnį 0,0804, kai parametro ribos yra nuo $2*\alpha/20$ iki $2*\alpha$, kai α – tai pradinė parametro vertė paskaičiuota pagal maksimalio tikimybės metodą. Pradinė Φ_S vertė yra 0,4560, kai φ_{NS} pradinė vertė yra 0,8039. Gauti rezultatai kaip kinta prognozavimo paklaida keičiantis parametrų vertėms yra pateikiami paveiksluose žemiau (žr. 3.6.1 ir 3.6.2 pav.).



3.6.1 pav. Optimalaus prognozavimo modelio prognozavimo paklaidos MPE kitimas keičiant φNS parametrą



3.6.2 pav. Optimalaus prognozavimo modelio prognozavimo paklaidos MPE kitimas keičiant ΦS parametrą

Remiantis gautais rezultatais galima teigti, kad didinant φ_{NS} parametrą nuo pradinės vertės 0,8039 prognozavimo paklaida mažėja iki tam tikros parametro vertės, o po to vėl didėja, kaip ir tuomet kai parametro vertė mažėja nuo pradinės jo vertės. Analogiškai didinant Φ_S parametrą nuo jo pradinės vertės 0,4560, prognozavimo modelio paklaida mažėja iki tam tikros parametro vertės, o po to didėja kaip ir mažinant parametro vertę nuo pradinės jo vertės. Žinių bazės sudarymo realizacijos kodas yra pateikiamas laikmenoje (žr. „*FuzzyExpertKnow.m*“).

3.6.2. Optimalus neraiškios logikos reguliatoriaus sudarymas

Nustatant optimalų neraiškios logikos reguiatorių, kaip reguliatoriaus įėjimas yra naudojamas pasirinkto AR(1)xAR$_{24}$(1) optimalaus statistinio prognozavimo modelio prognozavimo paklaida MPE. Kadangi neraiškios logikos reguliatorius skirtas atnaujinti pasirinkto prognozavimo modelio parametrus, tai neraiškios logikos regulaitoriaus išėjimai yra prognozavimo modelio parametrų Φ_S ir φ_{NS} pokyčiai. Prognozavimo modelio prognozės paklaida yra kintamas dydis, todėl norint sudaryti universalų neraiškios logikos reguliatorių, reguliatoriaus įėjimo kintamasis yra normalizuojamas pagal žemiau patekta formulę (žr. 3.6.1 formulę). Normalizavimas buvo atliekamas režiuose tarp -1 ir 1, kai maksimalį prognozavimo paklaida reikalinga normalizavimui atlikti yra $\mp$150 kWh.

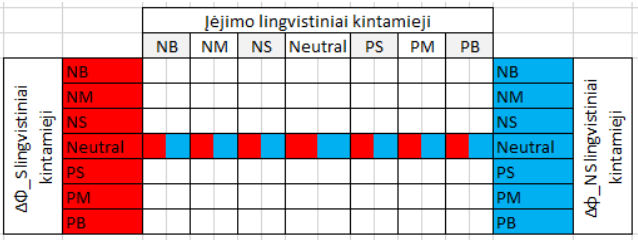
$$y = \frac{((X_{max} - X_{min}) * ((X - Y_{min}))}{(Y_{max} - Y_{min})} + X_{min}; \quad (3.6.1)$$

čia Y – normalizuota įėjimo vertė, Ymax normalizuoto dydžio maksimali vertė, Ymin normalizuoto dydžio minimali vertė, Xmax nenormalizuoto dydžio maksimali vertė, Xmin nenormalizuoto dydžio maksimali vertė.

Pasirinkus neraiškios logikos reguliatoriaus įėjimo ir išėjimo kintamuosius, buvo sudaryti lingvistiniai kintamieji, kurie perteikė sukauptas ekspertinės žinias į skaitmeninę aplinką. Beieškant optimalios reguliatoriaus konfigūracijos buvo nuspręsta atlikti testavimus su 3-mis, 5-iais ir 7-iais lingvistiniais kintamaisiais:

- Trys kintamieji – *NegB, Neutral, PosB*
- Penki kintamieji – *NegB, NegS, Neutral, PosS, PosB*
- Septyni kintamieji – *NegB, NegM, NegS, Neutral, PosS, PosM, PosB*

Aukščiau išvardinti lingvistiniai kintamieji buvo naudojami tiek reguliatoriaus įėjimo, tiek išėjimo kintamiesiems nusakyti. Nustatant optimalų lingvistinių kintamųjų skaičių buvo sudaroma prognozė 240 – ams taškų, kurie kalendoriškai sekė, po tų duomenų, pagal kuriuos buvo nustatomas optimalus prognozavimo modelis. Per tą patį eksperimentą, buvo nustatomas optimalus lingvistinių kintamųjų skaičius bei optimali lingvistinių kintamųjų priklausomumo funkcija. Testuojamos priklausomumo funkcijos – *trimf, trapmf, gaussmf ir gbellmg*. Kiekviena lingvistinių kintamųjų imtis buvo testuojamas su visomis aukščiau išvardintomis priklausomumo funkcijomis. Atliekant pastarąjį eksperimentą naudojama žinių bazė yra pateikiama žemiau (žr. 3.6.3 pav.).



3.6.3 pav. Naudojama žinių bazė nustatant optimalų lingvistinių kintamųjų skaičių ir priklausomumo funkciją, esant septyniems lingvistinėms kintamiesiems

Atlikto eksperimento rezultatai yra pateikiami žemiau (žr. 3.6.1 lentelę.).

3.6.1 lentelė. Prognozavimo modelio paklaidos MPE %, matrica

	Priklausomumo funkcijos			
	trimf	**trapmf**	**gaussmf**	**gbellmf**
3 lingvistiniai kintamieji	16,9571	19,1094	17,0745	17,2438
5 lingvistiniai kintamieji	18,6209	16,9570	18,8736	16,4257
7 lingvistiniai kintamieji	17,0745	18,6209	15,8758	15,7991

Vertinant gautus rezultatus, optimalus lingvistinių kintamųjų skaičius yra 7, nes esat šiam lingvistinių kintamųjų skaičiui, prognozavimo paklaida MPE buvo mažiausia. Taip pat vertinant atlikto eksperimento rezultatus, optimali priklausomumo funkcija yra *gbellfm*. Todėl, tolimesniuose eksperimentuose bus naudojamas optimalus lingvistinių kintamųjų skaičius 7 ir *gbellmf* optimali priklausomumo funkcija. Neraiškios logikos optimalios priklausomumo funkcijos ir optimalaus lingvistinių kintamųjų skaičiaus nustatymo realizacijos kodas yra pateikiamas laikmenoje (žr. „*OptimalFuzzy.m*“).

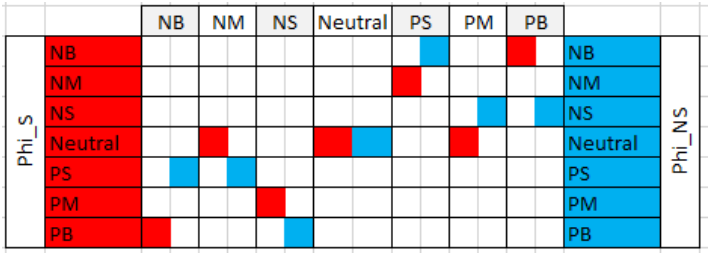
3.6.3. Optimalios žinių bazės nustatymas.

Nustatant optimalią žinių bazę, naudojami septyni lingvistiniai kintamieji reguliatoriaus įėjimui ir septyni reguliatoriaus išėjimui. Kaip jau buvo minėta anksčiau, žinių bazė sudaroma tokių būdu, kad sudarytos taisyklės būtų simetriškos reguliatoriaus įėjimo ženklo atžvilgiu. Kitaip sakant, jeigu įėjimo kintamojo lingvistinis kintamasis yra *NM*, o reguliatoriaus išėjimo kintamųjų lingvistiniai kintamieji $\Delta\Phi_{NS}$ yra *NM* ir $\Delta\varphi_S$ yra *PM*, tuomet kai įėjimo kintamojo lingvistinis kintamasis yra *PM*, tai išėjimo kintamųjų lingvistiniai kintamieji $\Delta\Phi_{NS}$ yra *PM*, o $\Delta\varphi_S$ yra *NM*. Taip pat buvo priimta taisyklė, kad esant reguliatoriaus įėjimo vertei *Neutral*, reguliatoriaus išėjimo $\Delta\Phi_{NS}$ vertė yra *Neutral*, o išėjimo $\Delta\varphi_S$ vertė yra *Neutral*. Įvertinant visas aukščiau išvardintas taisykles nustatant optimalią neraiškios logikos reguliatoriaus žinių bazę, bendruoju atveju buvo ištestuotos 4096 žinių bazės konfigūracijos.

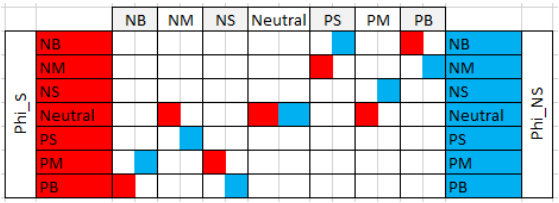
Penkios žinių bazės konfigūracijos, kurios atitinka mažiausią prognozavimo paklaidą yra pateikiamos paveikslėliuose žemiau (žr. 3.6.4, 3.6.5, 3.6.6, 3.6.7 ir 3.6.8 pav.). Taip pat pastarųjų žinių bazių prognozavimo paklaidų skaitinės vertės ir galutinės optimalaus prognozavimo modelio parametrų skaitinės vertės yra pateikiamos žemiau (žr. 3.6.2 lentelę).

3.6.2 lentelė. Tiksliausių žinių bazių konfigūraciju prognozavimo paklaidos ir parametru vertės

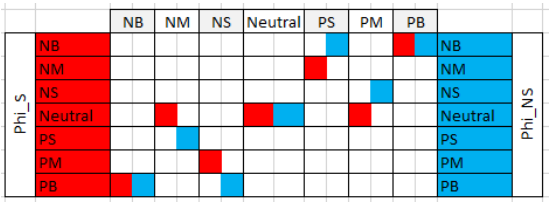
Eilės Nr.	φ_S	Φ_{NS}	MPE %
1	0,3176	1,062	14,58432053
2	0,3174	1,0506	14,62900605
3	0,3055	1,0172	14,68827836
4	0,456	1,0671	14,69941525
5	0,456	1,0869	14,70251057



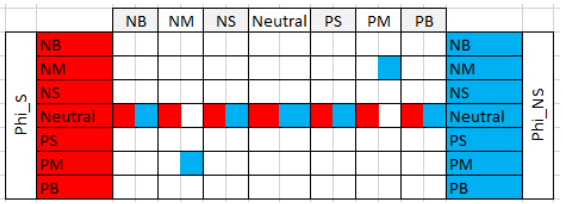
3.6.4 pav. Optimali neraiškios logikos reguliatoriaus žinių bazės konfigūracija



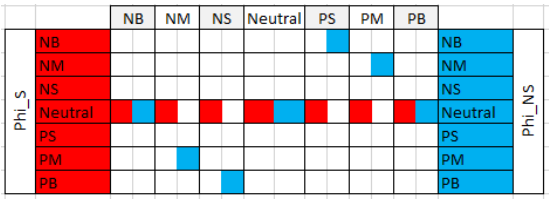
3.6.5 pav. Antra tiksliausia neraiškios logikos reguliatoriaus žinių bazės konfigūracija



3.6.6 pav. Trečia tiksliausia neraiškios logikos reguliatoriaus žinių bazės konfigūracija



3.6.7 pav. Ketvirta tiksliausia neraiškios logikos reguliatoriaus žinių bazės konfigūracija



3.6.8 pav. Penkta tiksliausia neraiškios logikos reguliatoriaus žinių bazės konfigūracija

Gauta optimali žinių bazės konfigūracija yra naudojama kaip pagrindinė žinių bazė galutiniame prognozavimo modelyje su parametrų atnaujinimu. Optimalios žinių bazės konfigūracijos nustatymo realizacijos kodas yra pateikiamas laikmenoje (žr. „*FuzzyOptimalKnowBase.m*“).

3.6.4. Optimalios parametrų atnaujinimo duomenų imties nustatymas

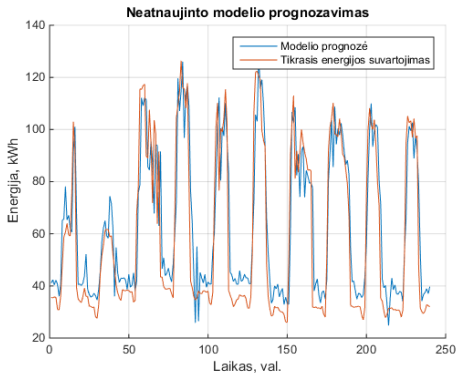
Norint užtikrinti optimalų programuojamo loginio valdiklio resursu išnaudojimą, būtina ištirti kokią įtaką prognozavimo modelio prognozės tikslumui daro duomenų kiekis kuris yra naudojamas parametrų atnaujinimui. Šiame eksperimente siekiama nustatyti ribinę duomenų imtį, kuri bus saugoma PLV vidinėje atmintyje ir naudojama parametrams atnaujinti padidėjus prognozavimo paklaidai.

Šiame eksperimente pasirinkto optimalaus prognozavimo modelio parametrai atnaujinami pagal – 48 informacinius taškus, 96 informacinius taškus, 144 informacinius taškus, 192 ir 240 informacinius taškus. Prognozavimo modeliai su skirtingais parametru atnaujinimo informaciniu tašku skaičiumi, tarpusavyje palyginami pagal prognozavimo modelio prognozavimo paklaidą MPE, kai prognozė yra sudaroma 240 informacinėms taškams, kurie yra nutolę per 240 taškų nuo modelio apmokymo imties. Taip pat prognozavimo modelių prognozės yra palyginamos su prognozavimo modelio prognozę, kurio parametrai nebuvo atnaujinti. Gauti rezultatai yra pateikiami lentelėje ir paveikslėliuose žemiau (žr. 3.6.3 lentelę) (žr. 3.6.9, 3.6.10, 3.6.11, 3.6.12, 3.6.13 ir 3.6.14 pav.).

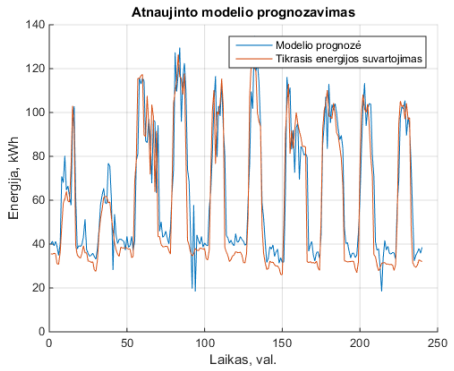
3.6.3 lentelė. Pasirinkto optimalaus prognozavimo modelio prognozavimo paklaida, kintant parametrų atnaujinimo duomenų imčiai

	Informacinių taškų skaičius				
	48	**96**	**144**	**192**	**240**
MPE, %	19,22731	18,70786	18,49785	18,03708	18,01893

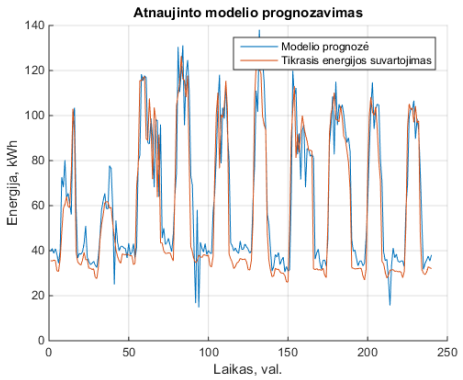
Prognozavimo modelio be parametrų atnaujinimo prognozavimo paklaida yra 20,6455% (žr. 3.6.11 pav.).



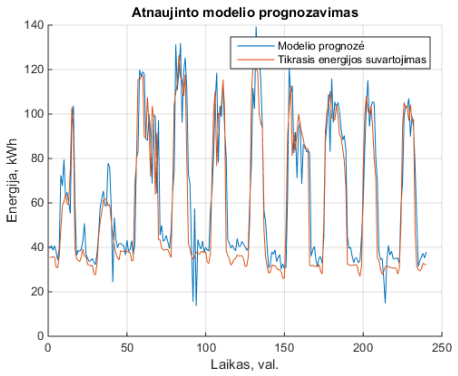
3.6.9 pav. Prognozavimo modelio be parametrų atnaujinimo prognozės grafikas



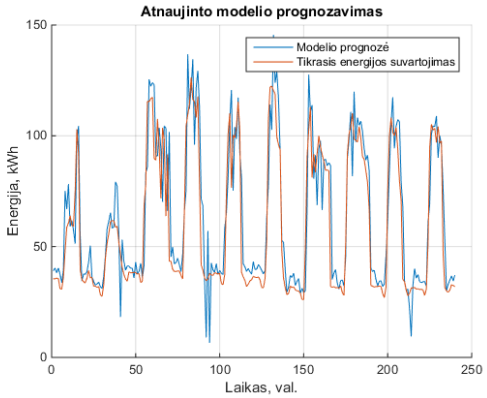
3.6.10 pav. Prognozavimo modelio su 48 taškų parametrų atnaujinimo imtimi, prognozės grafikas



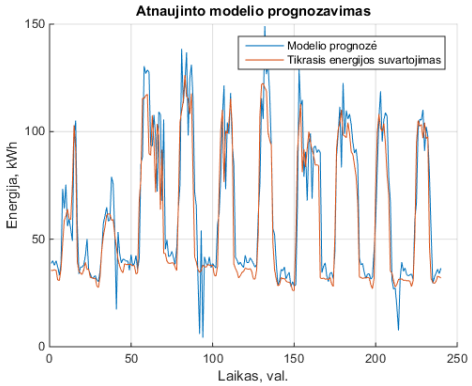
3.6.11 pav. Prognozavimo modelio su 96 taškų parametrų atnaujinimo imtimi, prognozės grafikas



3.6.12 pav. Prognozavimo modelio su 144 taškų parametrų atnaujinimo imtimi, prognozės grafikas



3.6.13 pav. Prognozavimo modelio su 192 taškų parametrų atnaujinimo imtimi, prognozės grafikas



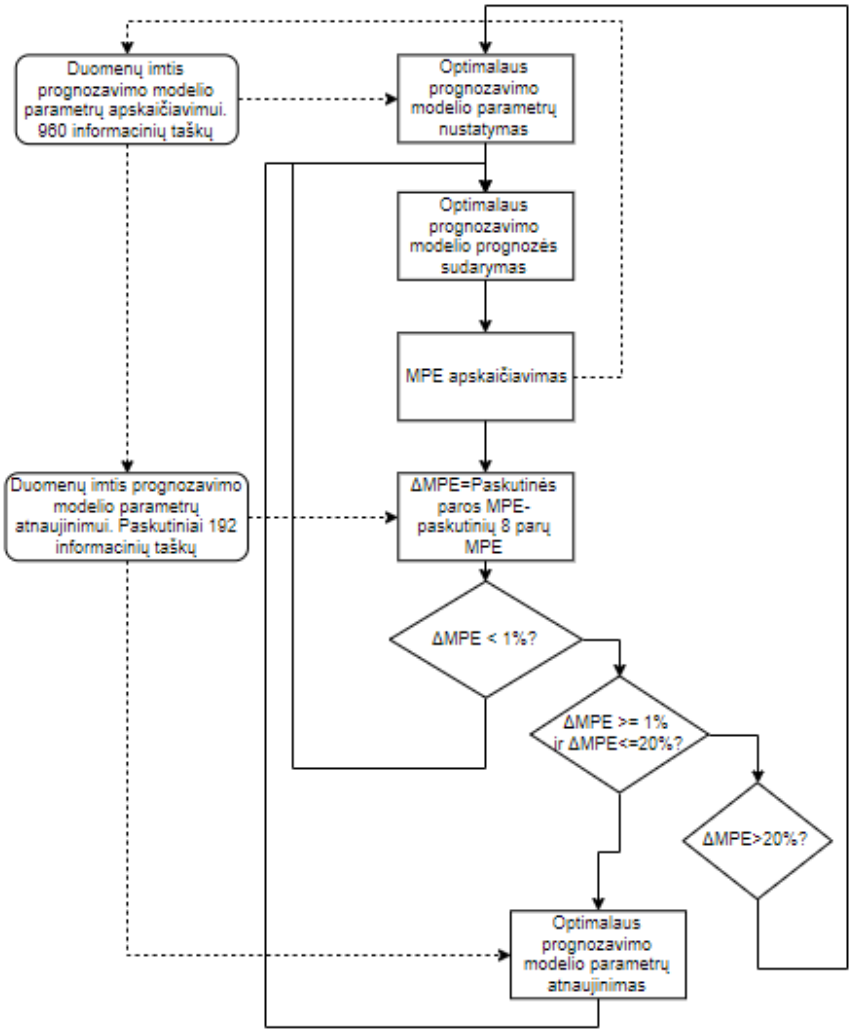
3.6.14 pav. Prognozavimo modelio su 240 taškų parametrų atnaujinimo imtimi, prognozės grafikas

Vertinant gautus rezultatus galima teigti, kad ribinis informacinių tašku skaičius yra 192, kuriam esant pasirinkto prognozavimo modelio paklaida sumažėjo per 2,6084%, lyginant su modelio prognozę, kurio parametrai nėra atnaujinami. Modelio su 240 informaciniais taškais skirtais parametru atnaujinimui, prognozavimo paklaida skiriasi vos per 0,0181%, nuo modelio su 192 informaciniais taškais. Optimalios duomenų imties skirtos parametrų atnaujinimui nustatymo realizacijos programinis kodas yra pateikiamas laikmenoje (žr. „*FinalOptimalCompare.m*“).

3.6.5. Optimalaus prognozavimo modelio su parametrų atnaujinimu prognozavimas

Galutinis prognozavimo modelis atitinka pasirinktą optimalų statistinį prognozavimo modelį, kuriame yra realizuotas optimalus neraiškios logikos reguliatorius, skirtas pasirinkto modelio parametrų atnaujinimui. Pastarajame modelyje parametru atnaujinimas vyksta ne nuolatos o pagal sudarytą parametrų atnaujinimo algoritmą. Sudarytame algoritme parametrai atnaujinami tuomet, kai vienos paros vidutinė prognozavimo paklaida (MPE) yra didesnė kaip 1% ir ne didesnė kaip 20% nei prieš tai esančių aštuonių dienų (192 informacinių taškų) vidutinė procentinė prognozavimo paklaida. Taip pat jeigu paskutinės paros prognozavimo paklaida yra didesnė nei 20% lyginant praeitų aštuonių dienų vidutine procentine prognozavimo paklaida, tuomet yra atliekamas modelio permokinimas

pagal paskutinius išsaugotus 960 informacinius taškus. Jeigu prognozavimo modelio paskutinės paros vidutinė procentinė prognozavimo paklaida yra iki 1%, tuomet prognozavimo modelio parametrai nėra atnaujinami. Sudarytas parametrų atnaujinimo algoritmas yra pateikiamas žemiau (žr. 3.6.15 pav.). Galutinio prognozavimo modelio su parametru atnaujinimu pagal pasiūlyta algoritmą programinis kodas yra pateikiamas laikmenoje (žr. „*FinalOptimalCompare.m*“).



3.6.15 pav. Optimalaus prognozavimo modelio parametrų atnaujinimo algoritmas [sudaryta autoriaus]

Sudaryto galutinio prognozavimo modelio su parametru atnaujinimu pagal pasiūlyta algoritmą veikimas patikrintas atliekant prognozavimą šiame darbe nagrinėto individualaus pastato energijos suvartojimo esant energijos suvartojimo trikdžiams. Realizuoti energijos suvartojimo trikdžiai yra kintami ir pastovūs. Pastovūs trikdžiai, tai šuoliškai padidėjęs nagrinėjamo objekto energijos suvartojimas per (-15) kWh, kai kintantys trikdžiai tai šuoliškai pakitęs objekto energijos suvartojimas per $\mp$10 kWh, kai trikdžių kitimo intervalas yra 5 val. Testuojant prognozavimo modelį buvo sudaroma prognozė 240 – čiai informacinių taškų arba 10 d.d. Prognozuojami taškai tai taškai einantys po duomenų imties pagal kurią buvo nustatytos pirminės prognozavimo modelio parametrų vertės. Gauti rezultatai yra pateikiami lentelėse žemiau (žr. 3.6.4 ir 3.6.5 lenteles). Lentelėse yra pateikiama prognozavimo modelio vidutinė prognozavimo paklaida modelio su parametrų atnaujinimu ir be, galutinės parametrų vertės, taip pat nurodyta kiek kartų modelyje buvo atnaujinami parametrai arba buvo atliekamas pakartotinis parametrų verčių nustatymas.

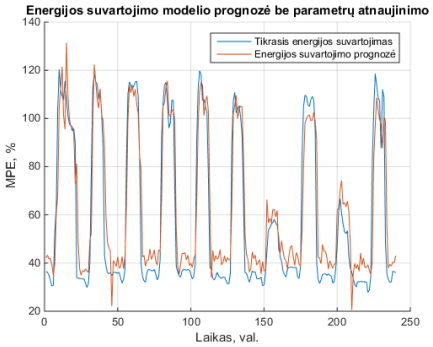
3.6.4 lentelė. Optimalaus prognozavimo modelio be parametrų atnaujinimo prognozavimo rezultatai

	Modelis be parametr atnaujinimo		
	MPE, %	**Φ_S**	**φ_{NS}**
Be trikdžio	18,6552	0,456	0,8039
Statinis trikdys	34,0851	0,456	0,8039
Kintantis trikdys	17,7647	0,456	0,8039

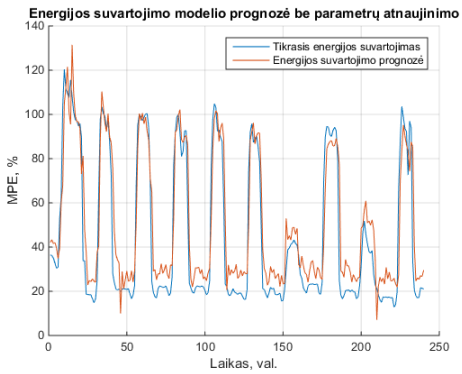
3.6.5 lentelė. Optimalaus prognozavimo modelio su parametrų atnaujinimu prognozavimo rezultatai

	Modelis su parametrų atnaujinimu				
	MPE, %	**Φ_S**	**φ_{NS}**	**Permokinimu skaičus**	**Atnaujinimu skaičius**
Be trikdžio	15,7713	0,6622	1,584	0	2
Statinis trikdys	26,8765	0,507	0,8436	2	4
Kintantis trikdys	16,1373	0,6732	1,9855	0	3

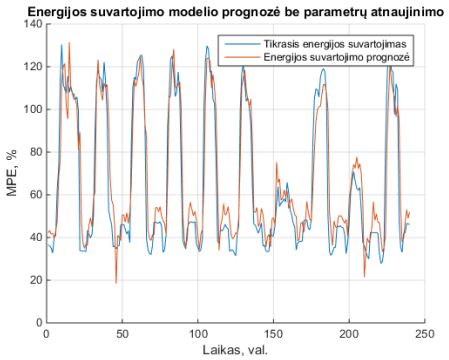
Palyginamų prognozavimo modelių prognozių grafikai yra atvaizduojami paveikslėliuose žemiau (žr. 3.6.16, 3.6.17, 3.6.18, 3.6.19, 3.6.20 ir 3.6.21 pav.).



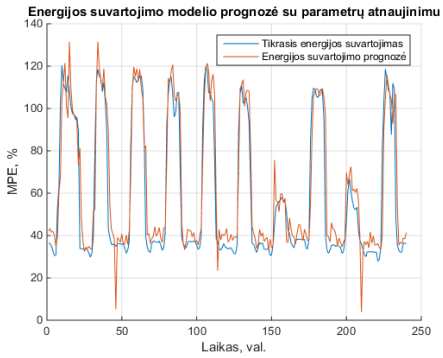
3.6.16 pav. Modelio be parametrų atnaujinimo prognozės grafikas duomenims be trikdžio



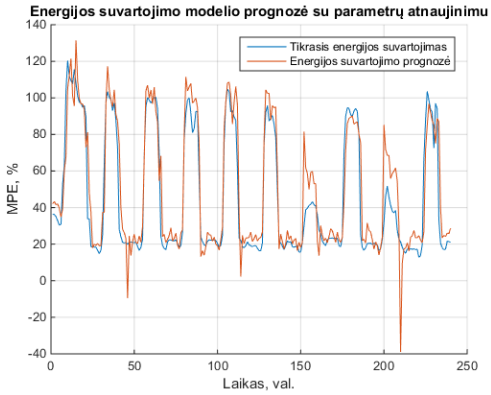
3.6.17 pav. Modelio be parametrų atnaujinimo prognozės grafikas duomenims su statinių trikdžiu



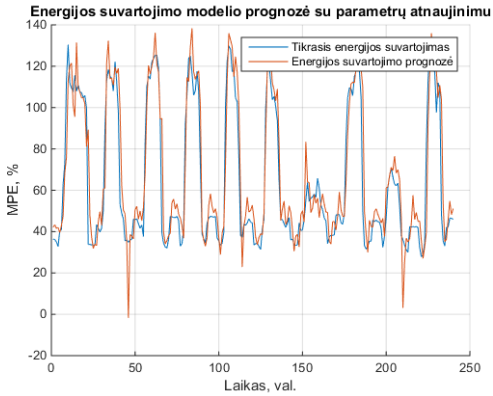
3.6.18 pav. Modelio be parametrų atnaujinimo prognozės grafikas duomenims su kintamu trikdžiu



3.6.19 pav. Modelio su parametrų atnaujinimu prognozės grafikas duomenims be trikdžio



3.6.20 pav. Modelio su parametrų atnaujinimu prognozės grafikas duomenims su statinių trikdžiu



3.6.21 pav. Modelio su parametrų atnaujinimu prognozės grafikas duomenims su kintamu trikdžiu

Pagal gautus rezultatus galima teigti, kad prognozavimo modelių su parametrų atnaujinimu bendra MPE per 240 informacinių taškų yra mažesnė, nei prognozavimo modelių be parametrų atnaujinimo. Prognozuojant energijos suvartojimą duomenims be trikdžių, modelio su parametrų atnaujinimu prognozavimo paklaida MPE yra 2,8839% mažesnė, nei modelio be parametrų atnaujinimo. Modelio su parametrų atnaujinimu energijos suvartojimo prognozė duomenims su pastovių trigdžiu MPE yra 7,2086% mažesnė, nei modelio be parametrų atnaujinimo. Taip pat esant pastoviam trikdžiui parametrų atnaujinimas buvo atliekamas 4 kartus, o parametrų pakartotinis nustatymas buvo atliekamas 2 kartus. Esant kintamam trikdžiui modelio su parametrų atnaujinimu MPE per 240 informacinius taškus yra 1.6274% mažesnis, nei modelio be parametrų atnaujinimo. Taip pat esant kintamam trikdžiui modelio parametrų atnaujinimas buvo atliekamas 3 kartus, o modelio parametrų pakartotinas nustatymas nebuvo atliekamas.

Išvados

1. Remiantis atliktos literatūros analizės rezultatais, AR, ARMA, ARIMA, ARIMAX autoregresiniai ir eksponentinio išlyginimo prognozavimo modeliai yra tinkami sudaryti mažai žinomo individualaus pastato suvartojamos elektros energijos prognozę. Panašios dienos modelis, tiesinės ir netiesinės regresijos modeliai nėra tinkami prognozei sudaryti, nes jiems realizuoti būtinas didelis duomenų kiekis apie pastato elektros energijos suvartojimą. Išnagrinėtuose visuose statistiniuose prognozavimo modeliuose nėra numatytas parametrų atnaujinimo algoritmas.

2. Remiantis vizualine pasirinkto objekto suvartojamos elektros energijos duomenų analize, pastebėtas duomenų sezoniškumas, turintis 24 valandų pasikartojimo intervalą bei, remiantis atlikto praplėsto Dickey – Fuller testo rezultatais, pastebėta, kad pasirinkto objekto neapdoroti suvartojamos elektros energijos duomenys yra nestacionarūs. Duomenų pirmos eilės skirtumas ir neapdoroti duomenys su pašalintu vidurkiu yra stacionarūs.

3. Remiantis autoregresinių modelių sudarymo metodika [34] bei autokoreliacijos ir dalinės autokoreliacijos funkcijų rezultatais, duomenims be vidurkio rekomenduojamas AR(1) prognozavimo modelis, kai $\varphi>0$, o duomenų pirmos eilės skirtumui - MA(1) modelis, kai $\theta<0$.

4. Remiantis atliktos literatūros analizės rezultatais [37,38], statistinių autoregresinių modelių parametrų paieškai naudojamas maksimalios tikimybės įvertinimo metodas. Pagal atliktų eksperimentų rezultatus, maksimalios tikimybės įvertinimo metodo ekstremumui surasti pasirinktas autoriaus patobulintas gradiento nusileidimo algoritmas. Pastarasis algoritmas pasirinktas, nes vienintelis surado globalų minimumą pakitus pradinėms parametrų vertėms, taip pat paieška užtruko mažiau nei 1000 iteracijų ir vidutiniškai apie 0,74 s. Šis algoritmas yra greitesnis už standartinį gradiento nusileidimo algoritmą, bet lėtesnis už Levenberg – Marquardt algoritmą.

5. Remiantis atlikto prognozavimo modelių patikimumo nustatymo eksperimento rezultatais, naudojant AIC ir BIC kriterijus, nustatyta, kad AR(1)xAR24(1) modelis yra patikimiausias. Pastarojo modelio gautų kriterijų AIC - 5,0426 ir BIC - 5,0578 vertės yra mažiausios, kas parodo jo patikimumą.

6. Atlikus realizuotų prognozavimo modelių prognozės analizę, nustatyta, jog optimalus autoregresinis prognozavimo modelis yra AR(1)AR24(1). Pastarojo modelio prognozavimo paklaida 10-čiai darbo dienų yra mažiausia tarp visų autoregresinių modelių prognozių ir atitinka MSE-188,0376 kWh ir MPE-17,3096%.

7. Remiantis atlikto tyrimo rezultatais, nustatyti koreliaciniai ryšiai tarp pastato suvartojamos elektros energijos ir išorinių kintamųjų – oro temperatūros ir drėgmės, autoregresiniai modeliai su išoriniais kintamaisiais nėra sudaromi. Modeliai nėra sudaromi dėl mažo koreliacinio ryšio tarp pastato suvartojamos elektros energijos ir išorinių kintamųjų. Koreliacijos koeficiento vertės tarp suvartojamos energijos ir lauko temperatūros yra 0,193551, o tarp energijos suvartojimo ir oro drėgmes yra (-0,0440654).

8. Remiantis literatūros analize [37, 38], eksponentinio išlyginimo prognozavimo modelio parametrai ieškomi, naudojant maksimalios tikimybės įvertinimo metodą. Pasirinktas

ekstremumo paieškos algoritmas – patobulintas gradiento nusileidimo algoritmas, buvo greičiausias, ieškant modelio parametro vertės ir užtruko vos 0,0289 s. ir 139 iteracijas. Eksponentinio išlyginimo prognozavimo modelio prognozavimo paklaidą atitinka MSE - 242,8796 kWh ir MPE - 22,8123%, sudarant prognozę 10-čiai darbo dienų.

9. Remiantis atlikto tyrimo rezultatais, optimalų dirbtinio neuroninio tinklo prognozavimo modelį sudaro du paslėpti sluoksniai atitinkamai su 86-iais ir 81-u paslėptais neuronais, o neuronų aktyvacijos funkcija yra *tansig*. Modelyje yra naudojami keturi įėjimo kintamieji, kai tinklas buvo apmokytas, naudojant Levenberg – Marquardt algoritmą. Pastaroji dirbtinio neuroninio tinklo konfigūracija yra optimali, nes atitinka mažiausią prognozavimo paklaidą MSE-20,38696 kWh, o MPE-1,9148%, kai modelis prognozuoja 10-čiai darbo dienų.

10. Remiantis sudarytų optimalių prognozavimo modelių prognozėmis, AR(1)xAR24(1) modelis yra optimalus individualaus pastato suvartojamos elektros energijos prognozavimo modelis. Sudarant prognozę 10-čiai darbo dienų duomenims, nenutolusiems nuo apmokymo duomenų, AR(1)xAR24(1) modelis tikslumu nusileidžia tik dirbtinio neuroninio tinklo prognozavimo modeliui. Sudarant prognozę 10-čiai darbo dienų duomenims, nutolusiems per 3 mėnesius nuo duomenų, pagal kuriuos buvo apmokyti modeliai, AR(1)xAR24(1) modelio prognozavimo paklaida yra mažiausia, kai MSE-275,726 kWh, o MPE-25,8974%. Tai nurodo, kad sudarant adaptyvų prognozavimo modelį, AR(1)xAR24(1) modeliui bus reikalinga mažiausia parametrų adaptacija prie pakitusio suvartojimo. Taip pat, remiantis prognoze 2, 5, 12 ir 24 valandoms į priekį nuo vieno informacinio taško, AR(1)xAR24(1) modelio prognozavimo paklaida išauga mažiausiai, kai yra atliekama prognozė 12-ai ir 24-oms val. į priekį, kai eksponentinio išlyginimo modelio prognozė išauga mažiausiai, sudarant prognozę 2-iem ir 5-ioms valandoms į priekį.

11. Remiantis atlikto tyrimo rezultatais, optimalus neraiškios logikos reguliatoriaus lingvistinių kintamųjų skaičius yra 7 - NegB, NegM, NegS, Neutral, PosS, PosM, PosB, kai optimali priklausomumo funkcija yra – gbellmf. Ši reguliatoriaus konfigūracija atitinka minimalią AR(1)xAR24(1) modelio prognozavimo MPE-15,7991%. Neraiškios logikos reguliatoriaus optimali žinių bazė atitinka minimalią autoregresnio AR(1)xAR24(1) modelio prognozavimo paklaidą MPE-14,58432053%. Optimalios žinių bazės konfigūracija yra pateikta aukščiau (žr. 3.6.6 pav). Optimali duomenų imtis autoregresinio AR(1)xAR24(1) modelio parametrams atnaujinti yra 192 val. arba 8 d.d., kai modelio MPE yra 18,03708%. Pastarasis duomenų kiekis yra optimalus, išlaikant modelio prognozavimo tikslumą, esant parametrų atnaujinimui ir tausojant PLV atminties resursus.

12. Remiantis atlikto tyrimo rezultatais, sudaryto optimalaus AR(1)xAR24(1) prognozavimo modelio su parametrų atnaujinimu, prognozavimo paklaidos sumažėjo, lyginant su modeliu, kurio parametrai nėra atnaujinami [24,25]. Sudarant prognozę 10-čiai darbo dienų energijos suvartojimui be trikdžių, modelio su parametrų atnaujinimu MPE yra 2,8839% mažesnė, nei modelio be parametrų atnaujinimo. Sudarant prognozę 10-čiai darbo dienų energijos suvartojimui su statiniu trikdžiu MPE yra 7,2086% mažesnė, nei modelio be parametrų atnaujinimo. Sudarant prognozę 10-čiai darbo dienų energijos suvartojimui su kintamu trikdžiu MPE yra 1.6274% mažesnė, nei modelio be parametrų atnaujinimo.

Literatūros sąrašas

1. WERON, Rafal. *Modeling and forecasting electricity loads and prieces a statistical approach [interaktyvus]*. John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, 2006, pp. 67 – 100 [Žiūrėta 2018-03-14]. ISBN-13: 978-0-470-05753-7. Prieiga per: doi: https://doi.org/10.1002/9781118673362.ch3.
2. HAO, Z. and SHAO, D. and LU, L. Power load forecasting. [Study Group Report]. Power load forecasting. In: *Workshop on Industrial Applications* [interaktyvus]. Hong Kong, 2006 [žiūrėta 2018-03-14]. Prieiga per: http://www.maths-in-industry.org/miis/524/1/Power-load-forecasting.pdf
3. ALFARES H. K. and NAZEERUDDIN, M., (2002). Electric load forecasting: literature survey and classification of method. In: *International Journal of Systems Science* [interaktyvus]. Taylor & Francis Ltd, 2002, volume 33, number 1, pp. 23 – 34 [žiūrėta 2018-03-14]. ISSN 1464-5319. Prieiga per: https://www.researchgate.net/publication/242705103_Electric_load_forecasting_Literature_survey_and_classification_of_methods
4. AHMAD, A. S., HASSAN, M.Y., ABDULLAH, M.P. RAHMAN, H.A., HUSSIN, F., ABDULLAH, H., SAIDUR, R. A review on applications of ANN and SVM for building electrical energy consumption forecasting. In: *Renewable and Sustainable Energy Reviews* [interaktyvus]. Elsevier Science, 2014, volume 33, pp. 102-109 [žiūrėta 2018-04-10]. ISSN :1364-0321. Prieiga per: doi: https://doi.org/10.1016/j.rser.2014.01.069
5. KHOTANZAD, A., ZHOU, E. and ELRAGAL, H. A Neuro-Fuzzy Approach to Short-Term Load Forecasting in a Price-Sensitive Environment. In: *IEEE Transactions on Power Systems* [interaktyvus]. 2002, 17(4), pp. 1273 - 1282 [žiūrėta 2018-03-14]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/TPWRS.2002.804999
6. KIM, K. H., PARK, J. K., HWANG, K. J. and KIM, S. H. Implementation of Hybrid Short-term Load Forecasting System Using Artificial Neural Networks and Fuzzy Expert Systems. In: *IEEE Transactions on Power Systems* [interaktyvus]. 1995, 10(3), pp. 1534 - 1539 [žiūrėta 2018-04-10]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/59.466492
7. TAKIYAR, S. and SINGH, V. Trend analysis and evolution of Short Term Load Forecasting Techniques. In: *Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), 2015 4th International Conference* [interaktyvus]. 2015 [žiūrėta 2018-04-15]. ISBN:978-1-4673-7231-2. Prieiga per: doi: https://doi.org/10.1109/ICRITO.2015.7359233
8. SONG, K. B., BAEK, Y. S., HONG, D. H. and JANG. G. Short-Term Load Forecasting for the Holidays Using Fuzzy Linear Regression Method. In: *IEEE Transactions on Power Systems* [interaktyvus]. 2005, 20 (1), pp. 96 – 101 [žiūrėta 2018-04-21]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/TPWRS.2004.835632
9. FEINBERG, E. A. and GENETHLIOU, D. Load Forecasting. In: *Applied Mathematics for Restructured Electric Power Systems* [interaktyvus]. Springer, Boston, MA, 2005, pp. 269-285 [žiūrėta 2018-04-21]. ISBN:978-0-387-23471-7. Prieiga per: doi: https://doi.org/10.1007/0-387-23471-3_12
10. SENJYU, T., TAKARA, H., UEZATO, K. and FUNABASHI, T. One-Hour-Ahead Load Forecasting Using Neural Network. In: *IEEE Transactions on Power Systems* [interaktyvus].

2002, 17 (1), pp. 113 - 118 [žiūrėta 2018-04-21]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/59.982201

11. PAPALEXOPOULOS, A. D and HESTERBERG, T. C. A regression-based approach to short-term system load forecasting. In: *IEEE Transactions on Power Systems* [interaktyvus]. 1990, 5 (4), pp. 1535 – 1547 [žiūrėta 2018-03-14]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/59.99410
12. Rafał WERON, R. MISIOREK, A. Forecasting Spot Electricity Prices With Time Series Models. In: *International Journal of Forecasting* [interaktyvus]. 2008, 24(4), pp. 744-763 [žiūrėta 2018-05-15]. Prieiga per: doi: https://doi.org/10.1016/j.ijforecast.2008.08.004
13. RUŽIC S., VUCKOVIC, A. and NIKOLIC, N. Weather Sensitive Method for Short Term Load Forecasting in Electric Power Utility of Serbia. In: *IEEE Transaction on power systems* [interaktyvus]. 2003, vol. 18, No. 4, pp. 1581-1586 [žiūrėta 2018-04-22]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/TPWRS.2003.811172
14. HYDE, O. and HODNETT, P.F. An Adaptable Automated Procedure for Short-Term Electricity Load Forecasting. In: *IEEE Transactions on Power Systems* [interaktyvus]. 1997, vol: 12, No. 1, pp. 84 – 94 [žiūrėta 2018-04-22]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/59.574927
15. HAGAN, M. T. and BEHR, S. M. The Time Series Approach to Short-Term Load Forecasting. In: *IEEE Power Engineering Review* [interaktyvus]. 1987, vol. 7, No. 8, pp. 56-57 [žiūrėta 2018-04-22]. ISSN: 1558-1705. Prieiga per: doi: https://doi.org/10.1109/MPER.1987.5527072
16. FAN, J. Y. and MCDONALD, J. D. A real-time implementation of short-term load forecasting for distribution power systems. In: *IEEE Transactions on Power Systems* [interaktyvus]. 1994, vol. 9, No. 2, pp. 988-994 [žiūrėta 2018-04-22]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/59.317646
17. FILIK, U. B. and KURBAN, M. A New Approach for the Short-Term Load Forecasting with Autoregressive and Artificial Neural Network Models. In: *International Journal of Computational Intelligence Research* [interaktyvus]. 2007 Vol.3, No.1, pp. 66–71 [žiūrėta 2018-05-15]. ISSN 0973-1873 Prieiga per internetą: https://www.researchgate.net/publication/228625376_A_New_Approach_for_the_Short-Term_Load_Forecasting_with_Autoregressive_and_Artificial_Neural_Network_Models
18. HUANG, S. J. and SHIH, K. R. Short-Term Load Forecasting Via ARMA Model Identification Including Non-Gaussian Process Considerations. In: *IEEE Transactions on Power Systems* [interaktyvus]. 2003, vol. 18, pp. 673-679. [žiūrėta 2018-04-22]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/TPWRS.2003.811010
19. MOGHRAM, I. and RAHRNAN, S. Analysis and evaluation of five short-term load forecasting techniques. In: *IEEE Transactions on Power Systems* [interaktyvus]. 1989, vol. 4, No. 4,pp. 1484 – 1491 [žiūrėta 2018-04-22]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/59.41700
20. YANG, H. T., HUANG, C. M. and HUANG, C. L. Identification of ARMAX Model for Short Term Load Forecasting: An Evolutionary Programming Approach. In: *IEEE Transactions on Power Systems* [interaktyvus]. 1996, vol. 11, No. 1, pp. 403 – 408. [žiūrėta 2018-04-22]. ISSN: 1558-0679. Prieiga per: doi: https://doi.org/10.1109/59.486125

21. PEIRONG, J., DI, X., PENG, W. and JUAN, C. A Study on Exponential Smoothing Model For Load Forecasting. In: *Power and Energy Engineering Conference (APPEEC), 2012 Asia-Pacific* [interaktyvus]. *2012,* [žiūrėta 2018-04-22]. ISSN: 2157-4847 Prieiga per: doi: https://doi.org/10.1109/APPEEC.2012.6307555
22. TAYLOR, J. W. Short-Term Electricity Demand Forecasting Using Double Seasonal Exponential Smoothing. In: *Journal of the Operational Research Society* [interaktyvus]. 2003, vol. 54, No. 8, pp. 799-805 [žiūrėta 2018-04-22]. Prieiga per: doi: https://doi.org/10.1057/palgrave.jors.2601589
23. TAYLOR, J. W., de MENEZES L. M. and MCSHARRY, P. E. A comparison of univariate methods for forecasting electricity demand up to a day ahead. In: *International Journal of Forecasting* [interaktyvus]. 2006, vol. 22, pp. 1-16 [žiūrėta 2018-04-22]. Prieiga per: doi: https://doi.org/10.1016/j.ijforecast.2005.06.006
24. TSENG, F. M., TZENG, G. H., YU, H. C. and YUAN, B. J. C. Fuzzy ARIMA model for forecasting the foreign exchange market. In: *Fuzzy Sets and Systems* [interaktyvus]. 2001, vol. 118, pp. 9–19 [žiūrėta 2019-03-10]. Prieiga per: doi: https://doi.org/10.1016/S0165-0114(98)00286-3
25. TSENG, F. M. and TZENG, G. H. Afuzzy seasonal ARIMAmodel for forecasting. In: *Fuzzy Sets and Systems* [interaktyvus]. 2002, vol 126, pp. 367–376 [žiūrėta 2019-03-10]. Prieiga per: doi: https://doi.org/10.1016/S0165-0114(01)00047-1
26. KOTHAMASU, R. and HUANG, S. H. Adaptive Mamdani fuzzy model for condition-based maintenance. In: *Fuzzy Sets and Systems* [interaktyvus]. 2007, vol. 158, pp. 2715 – 2733 [žiūrėta 2019-03-10]. Prieiga per: doi: https://doi.org/10.1016/j.fss.2007.07.004
27. PURVINIS, O. ir ŠUKYS, P. *Neraiškioji logika ir jos praktiniai taikymai* [interaktyvus]. Kaunas: Technologija, 2012 [žiūrėta 2019-03-10]. ISBN 978-609-02-0733-8. Prieiga per: https://www.ebooks.ktu.lt/eb/952/neraiskioji-logika-ir-jos-praktiniai-taikymai/
28. MAMDANI, E.H. and ASSILIAN, S. An experiment in linguistic synthesis with a fuzzy logic controller. In: *International Journal of Man-Machine Studies* [interaktyvus]. 1975, vol. 7, No. 1, pp. 1-13 [žiūrėta 2019-03-10]. Prieiga per: doi: https://doi.org/10.1016/S0020-7373(75)80002-2
29. KAUR, A. and KAUR, A. Comparison of Mamdani-Type and Sugeno-T ype Fuzzy Inference Systems for Air Conditioning System. In: *International Journal of Soft Computing and Engineering* [interaktyvus]. 2012, vol. 2, No. 2, pp. 323- 325 [žiūrėta 2019-03-10]. ISSN: 2231-2307. Prieiga per: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.486.1238&rep=rep1&type=pdf
30. GOLÉA, N. GOLÉA, A. and BENMAHAMMED, K. Fuzzy Model Reference Adaptive Control. In: *IEEE Transactions on Fuzzy Systems* [interaktyvus]. 2002, vol. 10, No. 4, pp. 436-444 [žiūrėta 2019-03-10]. ISSN: 1941-0034. Prieiga per: doi: https://doi.org/ 10.1109/TFUZZ.2002.800694
31. CHEN, T. L.,CHENG, C. H. and TEOH, H. J. High-order fuzzy time-series based on multi-period adaptation model for forecasting stock markets. In: *Physica A: Statistical Mechanics and its Applications* [interaktyvus]. 2008, vol. 387, No. 4, pp. 876-888 [žiūrėta 2019-03-10]. Prieiga per: doi: https://doi.org/10.1016/j.physa.2007.10.004

32. KOTHAMASU, R. and HUANG, S. H. Adaptive Mamdani fuzzy model for condition-based maintenance. In: *Fuzzy Sets and Systems* [interaktyvus]. 2007, vol. 158, pp. 2715 – 2733 [žiūrėta 2019-03-10]. Prieiga per: doi: https://doi.org/doi:10.1016/j.fss.2007.07.004
33. BAGDONAS, V. *Intelekto modeliai ir jų taikymas valdymui*. Kaunas: Technologija, 2003, pp 35-41. ISBN 9955-09-559-8.
34. BOX, G. E. P., JENKINS, G. M., REINSEL, G. C. and LJUNG, G. M. *Time series analysis : forecasting and control*. Wiley, 5 laida, 2015. ISBN: 978-1-118-67502-1.
35. DIEBOLD, F. X. and KILIAN, L. Unit Root Tests are Useful for Selecting Forecasting Models. In: *Journal of Business & Economic Statistics* [interaktyvus]. 2000, vol. 18, No. ,3 pp. 265-273 [žiūrėta 2018-05-15]. Prieiga per: doi: https://doi.org/10.2307/1392260
36. IGNATAVIČIUS, S. Pastato suvartojamos elektros energijos prognozavimo modelis, paremtas dirbtiniais neuroniniais tinklais. Iš: *E2ta–2018 elektronika, elektra, telekomunikacijos, automatika. 15-osios studentų mokslinės konferencijos pranešimų medžiaga* [interaktyvus]. Kauno Technologijos Universitetas, Kaunas 2018, pp 12 – 16 [žiūrėta 2018-05-15]. ISSN 2351-6275. Prieiga per: http://eta.ktu.edu/public/e2ta/Proceedings_E2TA_2018.pdf
37. de Hoon, M.J.L.,.van der Hagen, T.H.J.J , Schoonewelle, H. and van Dam, H. Why Yule-Walker should not be used for autoregressive modelling. In: *Annals of Nuclear Energy* [interaktyvus]. 1996, vol. 23, No. 15, pp. 1219-1228 [žiūrėta 2018-05-15]. Prieiga per: doi: https://doi.org/10.1016/0306-4549(95)00126-3
38. AL-OSH, M. A. and ALZAID, A. A. First-Order integer-valued autoregressive (INAR(1)) process. In: *Journal of Time Series Analysis* [interaktyvus]. 1987, vol. 8, No. 8, pp. 261-275 [žiūrėta 2018-05-15]. Prieiga per: doi: https://doi.org/10.1111/j.1467-9892.1987.tb00438.x

Priedai

1 priedas. Skirtuminis duomenų perskaičiavimas

```
function [ Y ] = Difference( X )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
    [n,m]=size( X );
    for i=2:1:(n)
        Y(i,1)=X(i,1)-X(i-1,1);
    end
end
```

2 priedas. Realizuotų prognozavimo modelių nustatytos parametrų vertės, esant skirtingoms pradinėms vertėms

Prognozavimo modelis	Modelio parametras	Parametro vertė		
		Pradinė parametro vertė		
		0	1	(-1)
AR(1)	ϕ_1	0,8839	0,8839	0,8839
ARI(1,1)	ϕ_1	0,1337	0,1337	0,1337
AR(2)	ϕ_1	1,0833	1,0833	1,0833
	ϕ_2	-0,2257	-0,2257	-0,2257
ARI(2,1)	ϕ_1	0,1459	0,1459	0,1459
	ϕ_2	0,0591	0,0591	0,0591
MA(1)	θ_1	-0,7666	-0,7664	-0,7666
IMA(1,1)	θ_1	-0,1482	-0,1482	-0,1482
MA(2)	θ_1	-0,7855	-0,7428	-0,9109
	θ_2	-0,6331	-0,6635	-0,8236
IMA(1,2)	θ_1	-0,1443	-0,1443	-0,1443
	θ_2	-0,0783	-0,0783	-0,0783
ARMA(1,0,1)	ϕ_1	0,8371	0,8371	0,8371
	θ_1	-0,2220	-0,2220	-0,2220
ARIMA(1,1,1)	ϕ_1	0,4545	0,4545	0,4545
	θ_1	0,3085	0,3085	0,3085
ARMA(1,0,2)	ϕ_1	$-3,0989*10^{-4}$	1,0006	-1,1058
	θ_1	-0,7810	-0,4997	-0,9785
	θ_2	-0,6341	-0,4997	-0,9546
ARIMA(1,1,2)	ϕ_1	0,0056	0,4724	0,3624
	θ_1	-0,1388	0,1976	0,1715
	θ_2	-0,0735	0,0117	-0,0315
ARMA(2,0,1)	ϕ_1	1,0898	0,8337	1,5921
	ϕ_2	-0,2098	0,1623	-0,8810
	θ_1	-0,1455	0,2906	-0,7360
ARIMA(2,1,1)	ϕ_1	0,2920	-0,4605	0,3034
	ϕ_2	0,0395	0,1956	-0,7454
	θ_1	-0,1456	-0,7826	-0,5744
ARMA(2,0,2)	ϕ_1	0,5751	1,1738	1,1909
	ϕ_2	0,1917	-0,3545	-0,3166
	θ_1	-0,5636	0,1404	0,1289
	θ_2	-0,2087	-0,2305	-0,0612
ARIMA(2,1,2)	ϕ_1	0.1163	0,6280	0,3073
	ϕ_2	0.0720	0,2537	-0,6459
	θ_1	-0.0356	0,6223	0,1908

	θ_2	0.0110	0,2467	-0,6032
AR(1)x AR_{24}(1)	ϕ_1	0,8039	0,8039	0,8039
	φ	0,4560	0,4560	0,4560
AR(1)x AR_{23}(1)	ϕ_1	0,8262	0,8262	0,8262
	φ	0,3556	0,3556	0,3556
AR(1)x AR_{25}(1)	ϕ_1	0,8250	0,8250	0,8250
	φ	0,3591	0,3591	0,3591
MA(1)x MA_{24}(1)	θ_1	0,6327	0,6266	0,6161
	Θ	0,5352	0,6175	0,5635
MA(1)x MA_{23}(1)	θ_1	0,6346	0,6181	0,6080
	Θ	0,5063	0,6184	0,5533
MA(1)x MA_{25}(1)	θ_1	0,6293	0,7255	0,6045
	Θ	0,5037	0,3520	0,5478
ARMA(1,0,1)x $ARMA_{24}$(1,0,1)	ϕ_1	$4{,}3625*10^{-6}$	0,0625	0,1250
	φ	0,9230	0,9993	0,5010
	θ_1	0,3465	0,6245	0,5001
	Θ	-0,4245	-0,4999	0,1250
ARMA(1,0,1)x $ARMA_{23}$(1,0,1)	ϕ_1	$3{,}2975*10^{-5}$	0,0625	0,1250
	φ	0,8635	0,9994	0,5010
	θ_1	0,3425	0,6245	0,5001
	Θ	-0,1711	-0,4999	0,1250
ARMA(1,0,1)x $ARMA_{25}$(1,0,1)	ϕ_1	$2{,}4471*10^{-5}$	0,0625	0,1252
	φ	0,8552	0,9997	0,5008
	θ_1	0,3425	0,6248	0,5003
	Θ	-0,1694	-0,5000	0,1251

3 priedas. Realizuotų modelių patikimumo kriterijų AIC ir BIC vertės

Modelis	AIC	BIC
AR(1)	5,23492835763527	5,24506780197790
ARI(1,1)	5,27111705367653	5,28633881929139
AR(2)	5,18477198474381	5,19998115125776
ARI(2,1)	5,26714730760451	5,28236907321937
MA(1)	5,90812659803705	5,91826604237967
IMA(1,1)	5,27149748597430	5,28164532971754
MA(2)	5,59116042430928	5,60636959082322
IMA(1,2)	5,27046998098949	5,29076566847597
ARMA(1,0,1)	5,19886050400082	5,21406967051476
ARMA(1,0,1)	5,26933220527658	5,28962789276306
ARMA(2,0,1)	5,59480187010677	5,61508075879203
ARIMA(2,1,1)	5,26899261119414	5,28928829868062
ARMA(1,0,2)	5,20352073583944	5,22379962452469
ARIMA(1,1,2)	5,27249015397681	5,29785976333491
ARMA(2,0,2)	5,20515435137782	5,23050296223439
ARIMA(2,1,2)	5,27350721327390	5,30395074450362
$AR(1)xAR_{24}(1)$	5,04261801921986	5,05782718573381
$AR(1)xAR_{23}(1)$	5,12238327111553	5,13759243762947
$AR(1)xAR_{25}(1)$	5,12037039590225	5,13557956241619
$MA(1)xMA_{24}(1)$	5,62999855538054	5,64520772189448
$MA(1)xMA_{23}(1)$	5,66676767638294	5,68197684289688
$MA(1)xMA_{25}(1)$	5,67654353634797	5,69175270286192
$ARMA(1,0,1)xARMA_{24}(1,0,1)$	5,40844459951032	5,42872348819558
$ARMA(1,0,1)xARMA_{23}(1,0,1)$	5,64837489165751	5,66865378034277
$ARMA(1,0,1)xARMA_{25}(1,0,1)$	5,62271505656413	5,64299394524939

4 priedas. AR, MA ir ARMA prognozavimo modelių prognozės paklaida

Modelis	Parametro pradinė vertė	Prognozavimo paklaida MSE, kWh	Prognozavimo paklaida MPE, %
AR(1)	0	221.2798	20,7835
ARI(1,1)	0	538,8278	50,6091
AR(2)	0	205,8063	18,9453
ARI(2,1)	0	609,0450	57,2042
MA(1)	0	324,8489	42,0513
IMA(1,1)	0	581,9680	54,6610
MA(2)	0	275,9755	34,3692
IMA(1,2)	0	562,5355	52,8358
ARMA(1,0,1)	0	206,007	18,6827
ARIMA (1,1,1)	0	605,6811	56,8883
ARMA (1,2)	0	252,6528	24,8498
	1	217,0261	19,9781
	(-1)	537,8032	56,3768
ARIMA (1,1,2)	0	548,7365	51,5398
	1	829,6058	77,9203
	(-1)	687,1555	64,5407
ARMA(2,1)	0	200,7419	18,4191
	1	291,7467	23,9270
	(-1)	348,6897	37,8837
ARIMA(2,1,1)	0	581,1901	54,5880
	1	528,5212	49,6410
	(-1)	522,1862	49,0461
ARMA (2,2)	0	201,0782	18,4741
	1	206,0971	19,2154
	(-1)	204,7545	17,9382
ARIMA (2,1,2)	0	601,7677	56,5207
	1	550,4333	51,6991
	(-1)	632,3359	59,3918
AR(1)xAR_{24}(1)	0	188,0376	17,3096
AR(1)xAR_{23}(1)	0	205,3591	19,6109
AR(1)xAR_{25}(1)	0	211,6605	19,8801
MA(1)xMA_{24}(1)	0	300,1178	28,1884
MA(1)xMA_{23}(1)	0	295,4977	35,2156
MA(1)xMA_{25}(1)	0	305,7912	28,7213
ARMA (1,0,1)x$ARMA_{24}$(1,0,1)	0	242,3139	22,7592
	1	213,5304	20,0557
	(-1)	257,1320	24,1510

ARMA (1,0,1)xARMA$_{23}$(1,0,1)	0	294,4698	30,2923
	1	266,4486	28,7754
	(-1)	272,3587	31,6034
ARMA (1,0,1)xARMA$_{25}$(1,0,1)	0	313,7267	29,4666
	1	292,2737	27,4516
	(-1)	279,2290	26,2264

5 priedas. Dirbtinio neuroninio tinklo prognozavimo modelių prognozavimo paklaidos, esant skirtingiems tinklo apmokymo algoritmams ir konfigūracijoms

Neuroninio tinklo apmokymo algoritmas	**Paslėptų sluoksnių skaičius**	**Neuronų skaičius pirmajame sluoksnyje**	**Neuronų skaičius antrajame sluoksnyje**	**Įėjimų skaičius**	**MPE, %**	**MSE, *kWh***	**Regresija, *%***	**Mokymo trukmė, *s***
Levenberg – Marquardt	1	6	0	3	3,0858	32,8539	0,80505	0,2422
	1	31	0	3	3,1561	33,6026	0,80023	0,22037
	1	11	0	3	3,1881	33,94372	0,82116	0,3646
	1	36	0	3	3,2569	34,67551	0,76744	0,2394
	1	21	0	3	3,2606	34,7152	0,81694	0,3216
Levenberg – Marquardt	2	1	6	3	2,8233	30,05918	0,80242	0,49015
	2	1	21	3	2,8559	30,40625	0,70639	0,14856
	2	1	26	3	2,8773	30,63394	0,80141	0,57794
	2	1	11	3	2,8786	30,64787	0,79476	0,16673
	2	1	36	3	2,8888	30,75715	0,77372	0,16793
Levenberg – Marquardt	1	6	0	4	1,9158	20,39686	0,93061	0,17326
	1	11	0	4	1,9335	20,58614	0,92706	0,23709
	1	26	0	4	1,9553	20,81761	0,93834	0,16768
	1	16	0	4	1,9944	21,23435	0,93297	0,20533
	1	21	0	4	2,0286	21,59827	0,8981	0,37436
Levenberg – Marquardt	2	86	81	4	1,9148	20,38696	0,92721	0,7368
	2	6	46	4	1,9204	20,44653	0,92599	0,47311
	2	6	81	4	1,9215	20,4582	0,92402	0,64863
	2	6	16	4	1,9322	20,57142	0,9315	0,15708
	2	11	11	4	1,9334	20,58458	0,93629	0,2974
Gradiento nusileidimo algoritmas	1	6	0	3	3,006	32,00496	0,79119	0,22418
	1	11	0	3	3,0067	32,01198	0,7857	0,19407
	1	16	0	3	3,0577	32,55496	0,71752	0,18596
	1	26	0	3	3,0822	32,81616	0,75072	0,17082
	1	46	0	3	3,1381	33,41081	0,78119	0,39968
Gradiento nusileidimo algoritmas	2	1	6	3	2,8828	30,69292	0,78797	0,069702
	2	66	6	3	2,8945	30,81679	0,80699	0,33247
	2	71	6	3	2,8948	30,82037	0,80355	0,23973
	2	26	11	3	2,9034	30,91227	0,80499	0,20477
	2	31	11	3	2,9456	31,36129	0,79156	0,088911
Gradiento nusileidimo algoritmas	1	6	0	4	2,0705	22,04379	0,92024	0,089657
	1	21	0	4	2,0893	22,24402	0,90697	0,16551
	1	11	0	4	2,0925	22,27886	0,91336	0,073634
	1	16	0	4	2,1137	22,50374	0,90951	0,21269

	1	51	0	4	2,1238	22,61199	0,91569	0,60042
Gradiento nusileidimo algoritmas	2	91	6	4	1,9308	20,55669	0,92553	0,36595
	2	26	6	4	1,9353	20,60464	0,92691	0,2148
	2	31	26	4	1,9548	20,81276	0,92706	0,22771
	2	61	16	4	1,9554	20,81916	0,91811	0,32825
	2	36	36	4	1,9649	20,91989	0,92144	0,43346
Gradiento nusileidimo algoritmas su Reeves – Fletcher atnaujinimo metodu	1	26	0	3	3,0159	32,10962	0,79555	0,27031
	1	16	0	3	3,0553	32,52957	0,79426	0,23117
	1	11	0	3	3,0606	32,58596	0,79206	0,18122
	1	46	0	3	3,1302	33,32691	0,7067	0,36793
	1	41	0	3	3,1676	33,72531	0,79336	0,40203
Gradiento nusileidimo algoritmas su Reeves – Fletcher atnaujinimo metodu	2	11	76	3	2,8689	30,54496	0,79268	0,50356
	2	61	6	3	2,8758	30,6185	0,80045	0,22293
	2	11	11	3	2,8967	30,84061	0,79357	0,13698
	2	91	6	3	2,9316	31,21223	0,81281	0,71748
	2	96	11	3	2,9457	31,36246	0,79917	0,27139
Gradiento nusileidimo algoritmas su Reeves – Fletcher atnaujinimo metodu	1	41	0	4	1,992	21,20883	0,91458	0,36474
	1	76	0	4	2,0396	21,71547	0,92145	0,25785
	1	6	0	4	2,0664	22,00022	0,90397	0,26849
	1	96	0	4	2,1255	22,62984	0,90753	0,17094
	1	26	0	4	2,1785	23,19438	0,89902	0,35957
Gradiento nusileidimo algoritmas su Reeves – Fletcher atnaujinimo metodu	2	41	11	4	1,9354	20,60642	0,92194	0,20937
	2	76	11	4	1,9394	20,64825	0,92343	0,4372
	2	6	41	4	1,9479	20,73941	0,9143	1,6409
	2	96	16	4	1,9539	20,80306	0,92015	0,4553
	2	26	16	4	1,9571	20,83657	0,92436	0,42797