KAUNO TECHNOLOGIJOS UNIVERSITETAS

FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS

MATEMATINĖS SISTEMOTYROS KATEDRA

Miglė Drūlytė

RADIALINĖS BAZĖS FUNKCIJOS NEURONINIO TINKLO TAIKYMAS LAIKO EILUČIŲ, PAGRĮSTŲ NEREGULIARIU REKONSTRAVIMU, PROGNOZAVIME

Magistro darbas

Vadovas
lekt.dr. Kristina Lukoševičiūtė

KAUNAS, 2013

KAUNO TECHNOLOGIJOS UNIVERSITETAS

FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS

MATEMATINĖS SISTEMOTYROS KATEDRA

TVIRTINU
Katedros vedėjas
prof. habil.dr. V.Pekarskas
2013 06 02

RADIALINĖS BAZĖS FUNKCIJOS NEURONINIO TINKLO TAIKYMAS LAIKO EILUČIŲ, PAGRĮSTŲ NEREGULIARIU REKONSTRAVIMU, PROGNOZAVIME

Taikomosios matematikos magistro baigiamasis darbas

Vadovas
_____lekt.dr.
K.Lukoševičiūtė
2013 06 01

Recenzentas
_____dr. M.Kavaliauskas
2013 06 01

Atliko
FMMM-1 gr. stud.
_____ M.Drūlytė
2013 05 30

KAUNAS, 2013

KVALIFIKACINĖ KOMISIJA

Pirmininkas:	Rimantas Rudzkis, profesorius (VU)
Sekretorius:	Eimutis Valakevičius, docentas (KTU)
Nariai:	Jonas Valantinas, profesorius (KTU)
	Vytautas Janilionis, docentas (KTU)
	Vidmantas Povilas Pekarskas, profesorius (KTU)
	Zenonas Navickas, profesorius (KTU)
	Arūnas Barauskas, dr., direktoriaus pavaduotojas (UAB „Danet Baltic“)

Drūlytė M. Radial basis function neural network application to time series prediction, which are based on an non-regular embedding: Master's work in applied mathematics / supervisor lekt. dr. K. Lukoševičiūtė: Department of Mathematical Research in Systems, Faculty of Fundamental Science, Kaunas University of Technology. – Kaunas, 2013. – 70.

SUMMURY

Time series analysis gives the opportunity to choose a model that can be used for time series prediction, i.e. construction of a time series generating model and use of that model for the last time series values extrapolation to the future. Our goal is to find a model that describes the observations and use of the model to forecast the future time series values from the past values. There are numbers of time-series forecasting methods and models. The most common time series forecasting methods are the moving average method, exponential smoothing methods, random walk and trend patterns, methods based on neural, fuzzy logic networks.

The goal of this work is to show that the time series prediction using radial basis function neural network can be improved by using time series reconstruction into time delay space with non-uniform time delays. The time lags and embedding dimension are two factors that determine the attractor reconstruction with delay coordinate method. The embedding dimension describes the dimension of the time delay space.

Identification of embedding parameters includes not only optimization of time lags but also determination of optimal dimension of the reconstructed phase space. The forecasting of reconstructed time series is based on RBF (radial basis function) neural network. The experiments in this work with time series shows that proposed method can significantly improve the prediction accuracy.

TURINYS

PAVEIKSLŲ SĄRAŠAS

LENTELIŲ SĄRAŠAS

ĮVADAS

Laiko eilutė yra duomenų, surinktų per laiko periodą – savaitę, mėnesį, ketvirtį ar metus, seka. Laiko eilutė gali būti naudojama vykdant dabarties sprendimus ar įgyvendinant planus, grindžiamus ilgalaikėmis prognozėmis. Priimta, kad praeities modeliai tęsis toliau į ateitį. Ilgalaikės prognozės tęsias ilgiau kaip vienerius metus į ateitį; yra dažnos ir 5, 10, 15 ir 20 metų prognozės. Ilgo laikotarpio prognozės yra pagrindas, suteikiantis pakankamą laiką pirkimų, gamybos, pardavimų, finansų ir kitiems įmonės departamentams parengti planus dėl galimų naujų įrenginių įsigijimo, finansavimų, naujų produktų, modelių kūrimo.

Laiko eilučių analizė suteikia galimybę pasirinkti modelį, kuris gali būti naudojamos laiko eilutės prognozavime, t.y. laiko eilutę generuojančio modelio sukonstravimas ir to modelio panaudojimas paskutinių laiko eilutės reikšmių prognozavimui į ateitį. Laiko eilutės analizės tikslas yra nustatyti modelį, kuris apibūdina stebėjimus ir panaudojant tą modelį ekstrapoliuoti paskutines laiko eilutės reikšmes į ateitį. Dažniausiai pasitaikantys laiko eilutės prognozavimo modeliai yra slenkančio vidurkio modelis, eksponentinio glodinimo modeliai, atsitiktinio klaidžiojimo ir trendo modeliai, modeliai pagrįsti neuroniniais, neraiškios logikos tinklais.

Tyrimų objektas laiko eilutės, pagrįstos optimaliu nereguliariu rekonstravimu į laiko vėlinimų erdvę prognozavimo modelis. Laiko eilutės prognozavimas paremtas radialinės bazės funkcijos neuroniniu tinklu. Rekonstravimo parametrų nustatymas apima ne tik laiko vėlinimų optimizavimą, bet ir laiko vėlinimų erdvės optimalios dimensijos nustatymą.

Darbo tikslas – sudaryti laiko eilučių, kurios yra grįstos rekonstravimu į laiko vėlinimų erdvę su laiko vėlinimais, dinaminį modelį paremtą radialinės bazės neuronui tinklu. Pritaikyti šį modelį eilutės reikšmių ekstrapoliavimui į ateitį. Palyginti šią prognozavimo metodikos funkcionalumą su kitais laiko eilučių prognozavimo metodais.

Šiame darbe pirmiausia yra sukonstruojamas laikų eilučių rekonstravimo į nereguliarių laiko vėlinimų erdvę matematinis modelis. Šio modelio pagrindinis kriterijus turi atspindėti turimos laiko eilutės savybes. Tuomet pritaikomas šis rekonstravimo į laiko vėlinimų erdvę nereguliariais laiko vėlinimais modelis radialinės bazės funkcijos neuroniniam tinklui. Toliau yra ekstrapoliuojamos laiko eilutės reikšmės į ateitį, panaudojant modifikuotą radialinės bazės funkcijos neuroninį tinklą. Modelio prognozės efektyvumas palyginamas su etaloninėmis laiko eilutėmis bei realaus pasaulio laiko eilutėmis. Galiausiai, gauti rezultatai palyginami su kitų modelių prognozių rezultatais.

Šia tema buvo skaitytas pranešimas tema „Laiko eilučių, pagrįstų nereguliariu rekonstravimu, prognozavimas RBF tinklais“ („Time Series Based On Non-Uniform Embedding Forecasting Using RBF Network“) konferencijoje „Matematika ir matematinis modeliavimas 2013“ bei pateiktas

straipsnis tema „Keleivių atvykimo į metro stotį laiko eilutės, pagrįstos nereguliariu rekonstravimu, prognozavimas RBF tinklais“ leidiniui „Taikomoji matematika“ 2013.

Magistrinio darbo struktūra. Darbą sudaro įvadas, trys skyriai, išvados, padėka, literatūros sąrašas ir priedai.

Pirmas skyrius skirtas prognozavimo modelių, neuroninių tinklų, rekonstravimo į laiko vėlinimų erdvę aprašymui, antras skyrius- rezultatų aptarimui, o trečias- programiniam algoritmų įgyvendino aptarimui.

1. BENDROJI DALIS

1.1. LAIKO EILUČIŲ PROGNOZAVIMAS

Laiko eilutė yra atsitiktinių kintamųjų stebėjimų seka. Vadinasi, ji yra stochastinis procesas.[2] Pavyzdžiui mėnesinė produkto paklausa, metinis pirmakursių priėmimas į universitetą ir t.t. Laiko eilučių prognozavimas yra ypač svarbus uždavinys daugelyje tyrimų sričių. Laiko eilučių analizė suteikia galimybę pasirinkti modelį, kuris gali būti naudojamos laiko eilutės prognozavime, t.y. laiko eilutę generuojančio modelio sukonstravimas ir to modelio panaudojimas paskutinių laiko eilutės reikšmių prognozavimui į ateitį. Yra įvairių laiko eilutės prognozavimo modelių: slenkančio vidurkio modelis, eksponentinio glodinimo modelis, atsitiktinio klaidžiojimo ir trendo modeliai, modeliai paremti neuroniniais, neraiškios logikos tinklais.

Taigi, mūsų tikslas yra nustatyti modelį, kuris apibūdina stebėjimus ir panaudojant tą modelį ekstrapoliuoti paskutines laiko eilutės reikšmes į ateitį. Aptarsime kelis laiko eilutės prognozavimo modelius.

Paprasčiausias modelis, naudojamas kaip "statybos plytos" daugelyje kitų modelių, yra atsitiktinis procesas.[3] Šis procesas gali būti apibrėžiamas kaip seka nekoreliuotų, vienodai pasiskirsčiusių atsitiktinių dydžių su vidurkiu lygiu nuliui ir pastovia dispersija. Šis procesas yra stacionarus su autokoreliacijos funkcija:

$$x_k = \begin{cases} 1, & k = 0 \\ 0, & \text{kitu atveju} \end{cases} \tag{1.1}$$

Atsitiktinis procesas yra vadinamas įvairiai: (nekoreliuotas) baltas triukšmas, inovacijos procesas arba „klaidos“ procesas.[3] Modelis yra retai naudojamas apibūdinti duomenis tiesiogiai, bet dažnai naudojamas sukonstruoti atsitiktinius sutrikimus sudėtingesniuose procesuose.

Atsižvelgiant į sekos nepriklausomumą, vietoje nekoreliuotų, atsitiktinių dydžių, kai kurie autoriai vartoja terminą „grynai atsitiktinis procesas“.

Vienas iš paprasčiausių modelių yra atsitiktinio klaidžiojimo (angl. Random Walk) modelis [1, 4, 8]. Šiuo modeliu tariama, kad dabartinė reikšmė X_t yra laikoma kaip pirmoji prognozės reikšmė, t.y. $X_t = X_{t+1}$. Tuomet atsitiktinio klaidžiojimo modelis apibrėžiamas:

$$X_t = X_{t-1} + \varepsilon_t \tag{1.2}$$

čia $\{\varepsilon_t\}$ yra atsitiktinis procesas, vadinamas triukšmu. Triukšmui vidurkis yra lygus nuliui, o dispersija yra pastovi. X_{t-1} yra eilutės reikšmė esanti prieš reikšmę X_t. Šis modelis gali būti naudojamas kaip pirmoji aproksimacija ekonominių ir finansinių laiko eilučių [3]. Pavyzdžiui,

kažkokios akcijos kaina dabartiniu momentu yra lygi tos pačios akcijos ankstesnės prekybos dienos kainai, prie tos kainos pridėjus ar atėmus tam tikrą akcijų kainų pokytį.[3]

Procesas apibrėžtas 1.2 formule nėra stacionarus, tai parodo laike didėja dispersija. Tačiau pirmosios eilės skirtumai, t.y. ($\Delta X_t = X_t - X_{t-1} = \varepsilon_t$ [9]), sudaro stacionarų procesą.

Kitas, šiek tiek sudėtingesnis, laiko eilučių modelis yra tiesinio trendo modelis. Šis modelis yra aprašomas lygtimi:

$$X_t = b_0 + b_1 t + \varepsilon_t \qquad \textbf{(1.3)}$$

čia X_t žymi tiriamos laiko eilutės reikšmę laiko momentu *t*, b_0 ir b_1 yra nežinomi parametrai ir ε_t žymi triukšmą, atitinkantį laiko eilutės dalį, kuri negali būti sutapatinama su trendo linija. [6] Kai yra padaromos atitinkamos prielaidos apie triukšmo pobūdį, mes galime įvertinti nežinomus parametrus b_0 ir b_1. Jei parametrai b_0 ir b_1 išlieka pastovūs, tai turime globalaus trendo modelį. Tokiu atveju eilutes reikšmes galima prognozuoti naudojant tiesinės regresijos modelį. Jei šie parametrai laikui bėgant keičiasi, tai turime lokaliojo trendo modelį. Šiam modeliui dažniausiai naudojamas dvigubas eksponentinis glotninimas.

Yra dviejų tipų glotninimo modeliai:

- Vidurkinimo modelis.
- Eksponentinio glotninimo modelis.

Paprasčiausias būdas suglotninti eilutę yra suskaičiuoti paprastą arba nesvertinį vidurkį [3]. Suglotninta statistika $\{S_t\}$ yra paprasčiausias vidurkis paskutinių *k* reikšmių. Slenkančio vidurkio modelis aprašomas formule:

$$S_t = \frac{1}{k}\sum_{n=0}^{k-1} x_{t-n} = S_{t-1} + \frac{X_t - X_{t-k}}{k} \qquad \textbf{(1.4)}$$

čia $k > 1$ yra sveikasis skaičius. Mažas *k* turės mažesnį glotninimo efektą ir tokiu atveju bus labiau atsižvelgiama į paskutinius stebėjimų pokyčius. Tuo tarpu didesnė *k* reikšmė turės didesnį glotninimo poveikį. Vienas šio modelio trūkumas yra, kad jis negali būti naudojamas pirmoms laiko eilutės $k-1$ reikšmėms [3].

Šiek tiek sudėtingesnis laiko eilutės $\{X_t\}$ glotninimo modelis yra skaičiavimas svertinio slenkančio vidurkio, pirmiausia pasirenkant svorių seką $\{w_1, w_2, \ldots, w_k\}$, tokią kad

$$\sum_{n=1}^{k} w_n = 1 \qquad \textbf{(1.5)}$$

tuomet panaudojant tuos svorius skaičiuojant suglotnintą statistiką $\{S_t\}$. Taigi, svertinis slenkančio vidurkio modelis aprašomas:

$$S_t = \sum_{n=1}^{k} w_n X_{t+1-n} \quad \textbf{(1.6)}$$

Praktikoje svoriai dažniausiai pasirenkami suteikti didesnį svorį paskutinėms laiko eilutės reikšmėms ir mažesnius svorius senesnėms laiko eilutės reikšmėms [3]. Reiktų atkreipti dėmesį į tai, kad šis modelis turi tą patį trūkumą kaip paprastas slenkamasis vidurkis, ir kad glotninimo procedūros apskaičiavimas kiekviename žingsnyje yra kur kas sudėtingesni nei paprasto slenkančio vidurkio.

Laiko eilutės duomenis galima analizuoti ir tokiais statistiniais modeliais kaip eksponentinio glodinimo modelis, dvigubo eksponentinio glodinimo modelis, AR , ARMA bei ARIMA, kurie yra gana detaliai aprašyti autorių straipsniuose [5, 7, 10-11, 13-17, 29-30]. Šiame darbe šių modelių nenagrinėsime.

1.2. DIRBTINIAI NEURONINIAI TINKLAI (ANN)

Dirbtinis neuroninis tinklas (ANN) yra sistema, pagrįsta biologinių neuroninių tinklų veikimu, kitaip tariant, yra biologinės nervų sistemos emuliacija. Emuliacija - galimybė visą sistemą arba jos dalį imituoti kitos sistemos priemonėmis neprarandant imituojamos sistemos funkcinių galimybių ir neiškreipiant gaunamų rezultatų. Programinė neuroninių tinklų realizacija gali turėti privalumų ir trūkumų [18, 19, 20]

Privalumai:

- Neuroninis tinklas gali atlikti užduotis, kurių negali atlikti linijinė programa.
- Jei koks nors neuroninio tinklo elementas sugenda, sistema be jokių problemų gali veikti ir toliau, nes ji yra lygiagretaus pobūdžio.
- Neuroninis tinklas turi savybę būti apmokomas ir dėl to jo nereikia perprogramuoti.
- Dirbtinis neuroninis tinklas gali būti įgyvendintas bet kuriame programiniame pakete.
- Neuroninis tinklas gali būti įgyvendintas be jokių problemų.

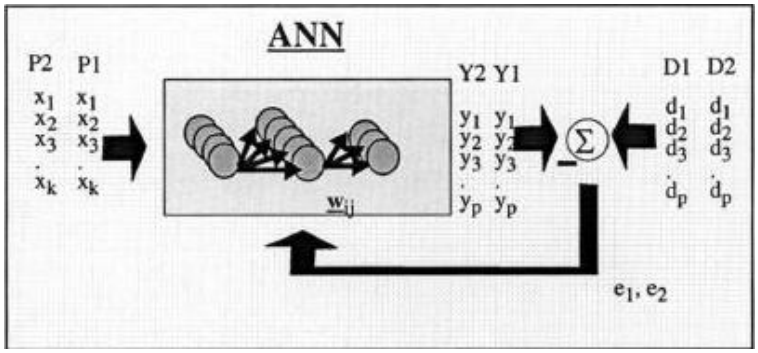
Trūkumai:

- Tam kad neuroniniu tinklu būtų galima naudotis, jį reikia apmokyti.
- Neuroninių tinklų architektūra skiriasi nuo mikroprocesorių architektūros todėl neuroninį tinklą reikia emuliuoti.
- Dideliam neuroniniam tinklui reikalinga didelis apdorojimo laikas.

Kitas dirbtinių neuroninių tinklų aspektas yra toks, kad yra skirtingos neuroninių tinklų struktūros, todėl yra reikalingi skirtingi tų struktūrų algoritmai. Esant akivaizdžiai sudėtingai neuroninio tinklo sistemai, tinklo įgyvendinimas yra gana paprastas [18].

Dirbtiniai neuroniniai tinklai turi dvi pagrindines funkcijas: struktūros klasifikatorius ir nelinijinius adaptacinius filtrus. Kaip ir biologinis pirmtakas, dirbtinio neuroninio tinklo sistema yra prisitaikoma. Prisitaikoma sistema reiškia tai, kad kiekvienas parametras sistemos veikimo metu yra atnaujinamas. Tokie veiksmai vadinami mokymo etapu [19].

Dirbtinis neuroninis tinklas yra sukurtas su sisteminga, žingsnis po žingsnio, procedūra, kuri optimizuoja kriterijų, paprastai vadinamą kaip mokymosi taisyklę. Šių neuroninių tinklų pagrindas yra įvesties/išvesties apmokymo duomenys, nes jie perteikia informaciją, kuri yra būtina tinklo tinkamam veikimui.



1.1 pav. ANN tinklo schema [18]

Iš esmės, dirbtinis neuroninis tinklas yra sistema, t.y. struktūra, kuri gauna pradinius duomenis, juos apdoroja ir pateikia rezultatą. Paprastai, įvestis susideda iš duomenų masyvo, pavyzdžiui, duomenys iš vaizdinio failo, garso bangų duomenys ar bet kokios rūšies duomenys, kurie gali būti pateikti masyvu [18]. Kai įvestis $(P1, P2)$ (žr. 1.1. pav.) yra pateikiama neuroniniam tinklui, yra nustatomas atitinkamas pageidaujamas $(D1, D2)$ atsakas (angl. target), suskaičiuojama klaida $(e1, e2)$, kurią sudaro norimo atsako ir gauto rezultato skirtumas $(Y1, Y2)$.

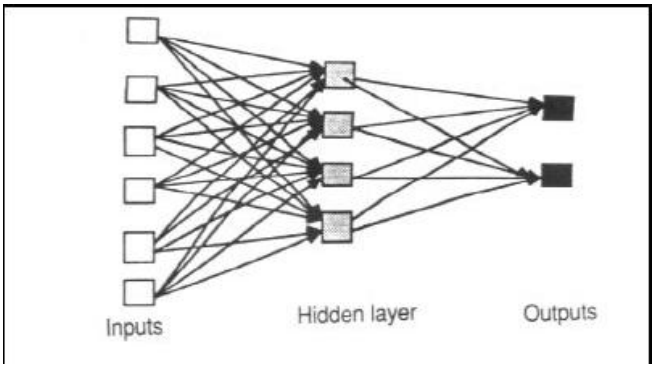
Klaidos informacija yra grąžinama atgal į sistemą, kuri leidžia atnaujinti visus sistemos parametrus (w_{ij}). Šis procesas kartojamas tol, kol yra gaunamas norimas rezultatas. Svarbu pastebėti, kad tinklo atlikimas labai priklauso nuo pateiktų duomenų [18].

Kuriant neuroninį tinklą yra reikalinga parinkti tinklo tipą, veikimo funkciją, mokymo taisyklę ir kriterijų reikalingą mokymo etapo sustabdymui. Taigi, yra sudėtinga nustatyti tinklo dydį ir parametrus, kai nėra jokios tai aprašančios taisyklės ar formulės.

Modeliuodami dirbtinį funkcinį modelį iš biologinio neurono, turime atsižvelgti į tris pagrindinius komponentus. Pirmiausia, biologinio neurono sinapsės yra modeliuojamos kaip svoriai. Dirbtinio neurono atveju svoris yra skaičius, o atstovauja sinapsę. Neigiamas svoris atspindi slopinantį ryšį, o teigiamos reikšmės atspindi sužadinamąją jungtį. Visos įvestys yra modifikuojamos svoriais. Toks procesas vadinamas tiesine kombinacija [20].

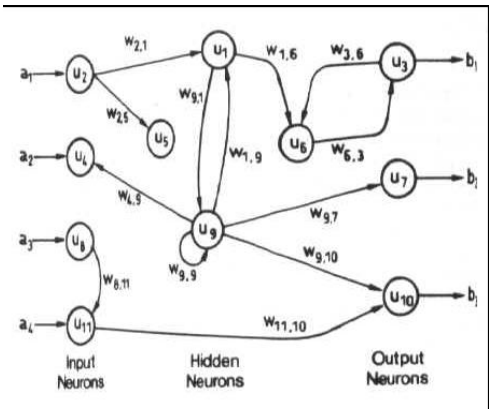
Yra dviejų tipų dirbtiniai neuroniniai tinklai:

- Tiesioginio ryšio (angl. feedforward) dirbtiniai neuroniniai tinklai leidžia signalui keliauti tik viena kryptimi; iš įvesties iki rezultato. Nėra grįžtamų ryšių (kilpų), t.y. bet kurio sluoksnio rezultatas neturi įtakos tam pačiam sluoksniui. Tiesioginio ryšio neuroninis tinklas pasižymi tiesiu keliu, kuris sieja įvestis su išvestim (žr. 1.2. pav.). Šie tinklai yra plačiai naudojami struktūrų atpažinime [22].



1.2 pav. Tiesioginio ryšio dirbtinio neuroninio tinklo schema [22]

- Dirbtiniai neuroniniai tinklai turintys grįžtamąjį sąryšį. Šie tinklai gali turėti signalus , kurie gali keliauti, kitaip tariant, tinklas su kilpomis. Grįžtamojo ryšio tinklai yra labai sudėtingi. Grįžtamojo ryšio tinklai yra dinamiški, jų „būsenos“ nuolat keičiasi tol, kol yra pasiekiamas pusiausvyros taškas. Jie lieka pusiausvyros taško padėtyje iki tol, kol yra pakeičiami įvesties duomenys, tuomet vėl ieškomas naujas pusiausvyros taškas [22].



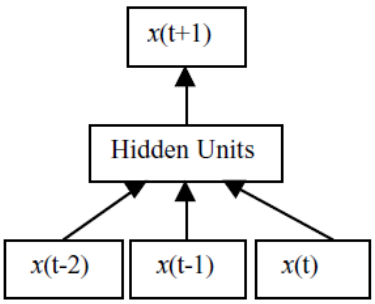
1.3 pav. Grįžtamojo ryšio dirbtinio neuroninio tinklo schema [22]

Klasikiniai tiesioginio ryšio neuroninių tinklų pavyzdžiai yra Perceptronas ir Adaline (Prisitaikantis tiesioginio ryšio neuroninis tinklas). „Grįžtamųjų ryšių neuroninių tinklų pavyzdžiai buvo pateikti Anderson‘o (Anderson, 1977), Kohonen‘o (Kohonen, 1977) ir Hopfield‘o (Hopfield, 1982)“ [23].

Dirbtiniai neuroniniai tinklai yra plačiai naudojami laiko eilučių prognozavime: dažniausiai naudojami tiesioginio ryšio neuroniniai tinklai. Tipiškais pavyzdžiais galėtų būti rinkos prognozės, meteorologinės ir tinklo srauto prognozavimas [22,23,24]. Tokiose sistemose turi būti apsvarstomos

dvi svarbios problemos: koks turi būti paimtas nagrinėjamų duomenų dažnis, ir duomenų taškų skaičius.

Standartinis neuroninių tinklų modelis, naudojamas laiko eilučių prognozavimui, yra funkcijos f paveikimas naudojant bet kokią tiesioginio ryšio funkciją, kuri aproksimuoja neuroninių tinklų architektūrą, pavyzdžiui, standartinis MLP (daugiasluoksnis perceptronas), RBF (radialinės bazės funkcija) architektūra ar „Cascade“ koreliacijos modelis [25], naudojant N – elementų seką kaip įėjimus ir vienintelę išeitį, kaip neuroninio tinklo norimą rezultatą. Šis modelis dažnai vadinamas „slankiojančių langų modeliu“ kai N – elementų įvestis pereina per visą norimą atsaką (angl. target). Schemoje pateikiama pagrindinė architektūra.



1.4 pav. Laiko eilučių prognozavimo standartinis modelis naudojant „slankiojančius langus“; trys laiko žingsniai [25]

Esant prognozavimo problemai, neuroninio tinklo įvestys (angl. input) paprastai yra nepriklausomi arba prognozuoti kintamieji. Funkcinis ryšys, rastas naudojant dirbtinius neuroninius tinklus, gali būti aprašomas:

$$y = f\left(x_1, x_2, \ldots, x_p\right) \tag{1.7}$$

čia $x_1, x_2, \ldots, x_p$ yra p nepriklausomų kintamųjų ir y yra priklausomas kintamasis [19]. Šia prasme, neuroninis tinklas yra funkcionaliai lygiavertis netiesinės regresijos modeliui. Įvertimo arba laiko eilutė prognozavimo problemoms spręsti, įvestys paprastai būna laiko eilutės praeities reikšmės ir rezultatas paprastai būna suprognozuota reikšmė. Dirbtinis neuroninis tinklas aprašomas:

$$y_{t+1} = f\left(y_t, y_{t-1}, \ldots, y_{t-p}\right) \tag{1.8}$$

čia y_t yra stebėjimas laiko momentu t [19].

1.2.1. RADIALINĖS BAZĖS FUNKCIJOS NEURONINIS TINKLAS

Radialinės bazės funkcijos (trump. RBF) neuroninis tinklas atsirado iš daugiamačių interpoliacijos modelių ir yra aprašomas literatūroje nuo 1985 metų [37, 38, 39, 40].

Radialinės bazės funkcijos neuroninis tinklas gali būti apibūdinamas kaip parametrizuotas modelis, naudojamas sutartinės funkcijos aproksimacijai, naudojant bazinės funkcijos tiesinę kombinaciją [41]. RBF tinklas priklauso branduolio funkcijų tinklų klasei, kur modelio įvestys yra paveiktos branduolio funkcijos. Kiekvienai bazinės funkcijos išvesčiai yra priskiriamas svoris [41].

Kadangi yra platus netiesinių modelių panaudojimas, galima iškelti tokius klausimus: „Kokie yra Radialinė bazinės funkcijos neuroninio tinklo panaudojimo privalumai?“, „Kodėl, mes turėtume naudoti RBF neuroninį tinklą iš daugelio kitų neuroninių tinklų tipų?“. Norint atsakyti į šiuos klausimus yra naudinga apsvarstyti priežastis, kodėl paprasti tiesiniai modeliai yra taip dažnai naudojami.

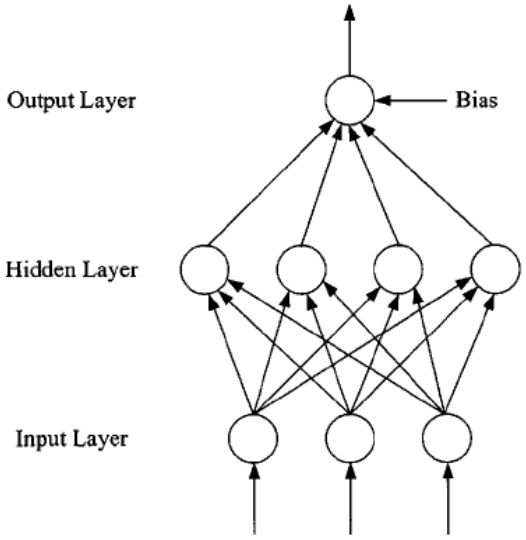
Tiesinės nekintančios laike perdavimo funkcijos turi pranašumą suteikiančias savybes kaip [41]:

- Suprantamumas;
- Lengvas FIR (baigtinių impulsų atsakų) modelių apmokymas;
- Nedidelis skaičiavimo sudėtingumas;
- Nedidelis atminties reikalavimas;
- Stabilumas;
- Atsparumas.

Nors netiesinių neuroninių tinklų modelių kūrimas buvo sėkmingas, ypatingai daugiasluoksnių tinklų apmokytų atgal skleidimu, mokslininkai nustatė, kad kai kuriais atvejais gali būti sunku apmokyti daugiasluoksnį tinklą ir gauti gerus rezultatus [41].

Pagrindinė priežastis daugiasluoksnio tinklo sunkaus apmokymo yra tokia, kad tinklai yra netiesiniai tinklo parametrams. Todėl norima gauti tokius modelius, kurie gali įveikti šią mokymo problemą. Norint išvengti šios problemos yra naudojamas RBF neuroninis tinklas [41].

Radialinės bazės funkcijos neuroniniu tinklu (trump. RBFNN), naudojant chaotinių laiko eilučių prognozavime, siekiama nustatyti laiko eilutės dinaminį modelį [28]. RBFNN yra neuroninio tinklo rūšis naudojama spręsti keletą problemų, tokių kaip modeliavimas klasifikavimas ir prognozavimas [32, 33, 34 35]. Radialinės bazės funkcijos neuroninio tinklo struktūra pateikta 1.5 paveiksle:



1.5 pav. Radialinės bazės funkcijos neuroninio tinklo schema [19]

RBF neuroninis tinklas susideda iš trijų sluoksnių (1.5 paveikslas): įvesties (angl. input) sluoksnio, paslėpto (angl. hidden) sluoksnis ir išėjimo (angl. output) sluoksnio. Įvesties sluoksnis informacijos neapdoroja, o tik paskirsto pradinius vektorius paslėptam sluoksniui [32]. RBFNN paslėptas sluoksnis susideda iš RBF vienetų (n_k) ir poslinkių (angl. bias) (b_k). Dažniausiai naudojama radialinės bazės funkcija yra Gauso funkcija, kuri yra apibūdinama centrais (c_j) ir pločiu (r_j) [42]. Šiame darbe naudojama Gauso funkcija su pločiu, kuris yra konstanta. RBF funkcijos matuojančios Euklido atstumus tarp įėjimo vektorių (X) ir radialinės bazės funkcijos centrų (c_j) ir atliekančios netiesines transformacijas su RBF paslėptajame sluoksnyje aprašomos formule:

$$h_j(X) = \exp\left(-\left\|x - c_j\right\|^2 / r_j^2\right) \quad \textbf{(1.9)}$$

kurioje h_j j-tajo RBF vieneto rezultato žymėjimas. Atitinkamai j-tajo RBF centras ir plotis yra (c_j) ir (r_j). Išėjimo sluoksnio operacija yra tiesinė ir aprašoma lygtimi:

$$y_k(X) = \sum_{j=1}^{n_h} w_{kj} h_j(X) + b_k \quad \textbf{(1.10)}$$

kur y_k yra k-tasis išvesties vienetas įėjimo vektoriui x, w_{kj} svoris ryšys tarp k-tojo išvesties vieneto ir j-tojo paslėpto sluoksnio vieneto ir b_k yra poslinkis.

Mokymo procedūrą, kai yra naudojama RBF, sudaro centrų, pločių ir svorių skaičiavimas.

MATLAB (*2011a, Neural Network Toolbox*) pakete RBF neuroninio tinklo skaičiavimas: Radialinės bazinės funkcija, kaip aktyvacijos funkcija, kuri paprastai yra Gauso funkcija, apskaičiuojama pagal formulę:

$$radbas(n) = e^{-n^2} \quad \textbf{(1.11)}$$

čia *radbas* yra MATLAB paketo funkcija. Kai Gauso funkcijos įvestis n yra netoli centrinės zonos, paslėptieji taškai pateiks didesnį rezultatą. Radialinės bazės funkcijos neuroninio tinklo modelis gali būti išreikštas lygtimis (1.16 – 1.18).

Įvestis n yra atstumas tarp svorių vektoriaus $w1$ (c_j) ir įvesties vektoriaus p (X), ir yra padaugintas iš rezultatų sluoksnio neuronų poslinkio $b1$ (b_k). Kai atstumas sumažės, rezultatas padidės. Todėl tinklas gali lokaliai aproksimuoti.

$$n = \|w1 - p\| b1 \quad \textbf{(1.12)}$$

Gauso funkcijos rezultatas yra:

$$a = radbas(\|w1 - p\| b1) \quad \textbf{(1.13)}$$

Išvesties neuronas naudoja tiesinę funkciją. Įvestis yra paslėptojo sluoksnio išvestis. Tai reiškia:

$$y = purelin(w2 \cdot a + b2) \quad \textbf{(1.14)}$$

čia *pureline* yra MATLAB funkcija, $w2$ yra svorinis vektorius tarp paslėpto sluoksnio ir rezultatų sluoksnio ir $b2$ yra išvesčių neuronų poslinkiai. $b1$ gali būti naudojamas reguliuoti funkcijos jautrumą. Paprastai $b1 = 0.8326 / C$, čia C yra išsibarstymo konstanta. Prieš apmokymą turi būti pateikti įvesties vektorius, norimo atsako (angl. target) vektorius ir išsibarstymo konstanta C.

1.2.2. FUNKCIJOS PRITAIKYMO NEURONINIS TINKLAS

Funkcijos pritaikymo neuroninis tinklas (trumpi. FITNET) yra naudojamas palygintini gautus rezultatus su RBF neuroniniu tinklu. FITNET neuroninis tinklas priklauso tiesioginio ryšio neuroninių tinklų klasei [53].

FITNET tinklas susideda iš sluoksnių serijos. Pirmasis sluoksnis turi ryšį su tinklo įvestimi. Kiekvienas sekantis sluoksnis turi ryšį su ankstesniu sluoksniu. Paskutinis sluoksnis pateikia tinklo išvestį. FITNET tinklai gali būti naudojami nustatyti ryšį bet kokios įvesties su išvestimi. Funkcijos pritaikymo neuroninis tinklas su vienu paslėptuoju sluoksniu ir pakankamai neuronų paslėptajame sluoksnyje, gali būti pritaikomas bet kokiam baigtiniai įvesties-išvesties atvaizdavimo problemai.

Funkcijos pritaikymo neuroninis tinklas gali būti apibūdinamas kaip analitinis atvaizdavimas tarp realių reikšmių įvesties X_k ($k = 1, 2, \ldots, N$) ir išvesties reikšmių y_t ($t = 1, 2, \ldots, M$). Įvesties kintamieji yra dauginami iš svorinių ryšių w_{jk}. FITNET tinklo funkcija paslėptajame sluoksnyje aprašomos taip [53]:

$$h_j = f\left(\sum_{k=1}^{N} w_{jk} + \theta_j\right),\ (j = 1,\ldots,J) \qquad \textbf{(1.15)}$$

čia J yra skaičius vienetų paslėptajame sluoksnyje ir θ_j yra riba. Funkcija $f(\cdot)$ yra sigmoidinė funkcija ir yra apibrėžiama:

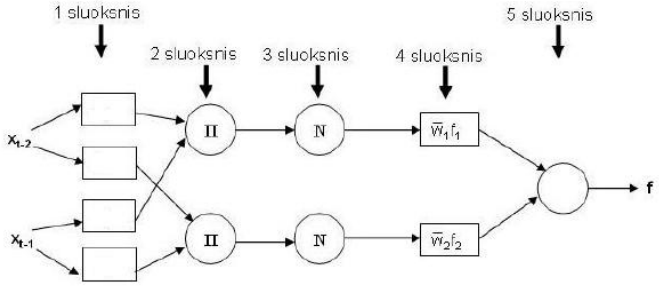
$$f(x) = \frac{1}{1+e^{-x}} - \frac{1}{2} \qquad \textbf{(1.16)}$$

Paslėptojo sluoksnio išvestis yra padauginama iš antrojo rinkinio svorinių ryšių $\hat{w}_{tj}$ ir yra pridedama riba $\hat{\theta}_t$ prie gauto vektoriaus bei sugeneruojama išvestis:

$$y_t = \sum_{j=1}^{J} \hat{w}_{tj} h_j + \hat{\theta}_t,\ t = 1,\ldots,M \qquad \textbf{(1.17)}$$

1.2.3. NEURONINĖS NERAIŠKIOS LOGIKOS SISTEMOS (ANFIS)

ANFIS architektūroje panaudota ANN ir neraiškių sprendimų sistemos (FIS) modeliai. FIS optimalaus sprendinio paieškai naudoja tam tikrą pradinę patirtį ir sugeneruotas žinias. ANFIS naudoja daugiareikšmę loginę sistemą, vadinamą neraiškia logika, su kuria siekiama atsižvelgti į paslėptą netikslumą duomenyse ir atitinkamai tuos duomenis tiksliai atvaizduoti. Tikslas pasiekiamas atliekant įėjimo duomenų fuzifikaciją per sąryšio funkcijas [57]. ANFIS gali gana greitai pritaikyti sąryšio funkcijų parametrus, ir priklausomai nuo įėjimo duomenų juos adaptyviai optimizuoti. ANFIS architektūra pateikta 1.6 paveiksle. Neuroninis neraiškių sistemų modelis yra penkių sluoksnių neuroninis tinklas, kurio pirmasis yra įėjimų sluoksnis. Pirmojo sluoksnio išėjimams yra priskiriamos sąryšio funkcijos antrajame sluoksnyje (vyksta duomenų fuzifikacija). Trečiasis sluoksnis yra taisyklių sluoksnis. Ketvirtasis sluoksnis yra normalizacijos sluoksnis, o paskutinysis sluoksnis yra defuzifikacijos sluoksnis. Paskutiniojo sluoksnio išėjimas ir yra prognozuojama laiko eilutės reikšmė.



1.6 pav. ANFIS tinklo architektūra

Neraiškių logikų sistemos prognozės uždavinyje įėjimų vektorius yra rekonstruotos laiko eilutės reikšmės, o išėjimo reikšmė $\hat{x}(t+\tau_{d-1})$ yra prognozuojama laiko eilutės reikšmė:

$$\hat{x}\left(t+\tau_{d-1}\right)=f\left(y_t'\right) \tag{1.18}$$

čia

$$y_t'=\left(x\left(t-\sum_{i=1}^{d-2}\tau_i\right),x\left(t-\sum_{i=2}^{d-2}\tau_i\right),\ldots,x\left(t\right)\right). \tag{1.19}$$

Bendru atveju neraiškios logikos išėjimas yra funkcija [57]:

$$f\left(y_t'\right)=\frac{\sum_{k=1}^{L}w_k\left(y'\right)f_k}{\sum_{k=1}^{L}w_k} \tag{1.20}$$

čia L – neraiškios logikos sistemos taisyklių skaičius, kuris priklauso nuo įėjimų vektoriaus ilgio ir užsibrėžtų sąryšio funkcijų skaičiaus; f_k - *k*-osios taisyklės išeities parametras; *k*-osios taisyklės svoris $w_k\left(y_t'\right)=\prod_{i=1}^{d-1}w_{ik}\left(y_{ti}'\right)$, $k=1,2,\ldots,L$; $w_{ik}\left(\cdot\right)$ *i*-ojo elemento ir *k*-osios taisyklės sąryšio funkcija. Tokio modelio architektūros ir apmokymo bei naudojimo detalius aprašymus galima rasti [57].

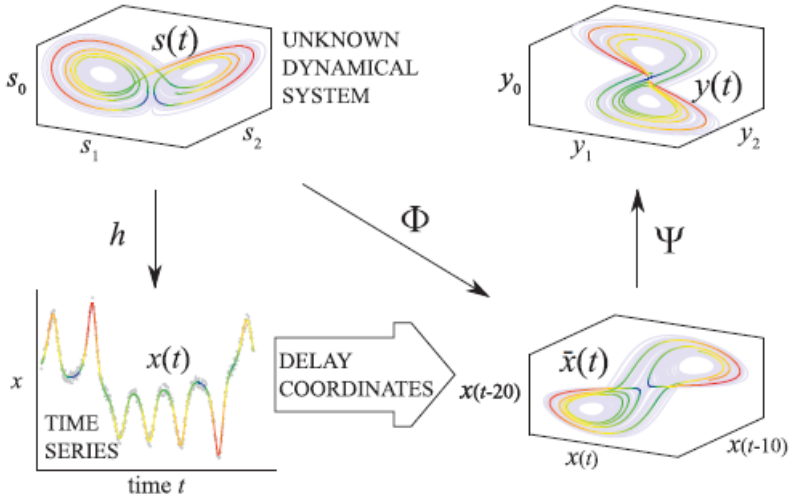
1.3. LAIKO EILUTĖS REKONSTRAVIMAS

Norint prognozuoti chaotines laiko eilutes, pirmiausia reikia bandyti rekonstruoti fazinę erdvę [28]. Dažniausiai duomenys yra gaunami nagrinėjant dinamikas, kurios yra „laiko eilučių“ formos, t.y. sekos reikšmės imamos reguliariais intervalais. Packard [29] ir Takens [30] naudojo tokias laiko eilutes rekonstruoti į parinktos sistemos laiko evoliucijos baigtinio matavimo laiko vėlinimų erdvę panaudojant laiko vėlinimų techniką. Rekonstruotos laiko vėlinimų erdvės ir originalios dinamikos topologinės savybės ir geometrinė struktūra yra visiškai tokios pačios [36]. Skaliarinės laiko eilutės dinamika $\left\{x(t),\ t=1,2,\ldots,N\right\}$ yra integruota į d -tosios dimensijos vėlinimų erdvę, kurioje laiko tarpai yra τ . Atsižvelgiant į laiko vėlinimo techniką, vėlinimų erdvė yra išreikšta [29]:

$$X\left(t\right)=\left(x(t),x(t-\tau),\ldots,x\left(t-(d-1)\tau\right)\right) \tag{1.21}$$

čia $t=(d-1)\tau+1,\ldots,N$, τ yra laiko vėlavimas, d yra rekonstravimo (angl. embedding) dimensija, ir N yra laiko eilutės ilgis. d -tosios dimensijos vėlinimų erdvėje, pradinės charakteristikos ir dinamikos reguliarumas gali būti apibrėžiami atsižvelgiant į d -tosios dimensijos rekonstravimo taškų trajektoriją. Laiko vėlinimo ir rekonstravimo dimensija yra svarbūs parametrai rekonstruojant erdvę [31].

Paveiksle 1.7 pateikiamas vėlinimų erdvės rekonstravimo procesas, pradedant nuo išmatuoto proceso. Bendriausia forma, vėlinimų erdvės vektoriaus $s(t)$ rekonstravimas gali būti apibūdinimas *d*-dimensijos vektoriumi $y(t)=\Psi\left[\bar{x}(t)\right]$, čia $\bar{x}(t)=\left\{x(t),x(t-\tau),\ldots,x\left[t-(d-1)\tau\right]\right\}$ yra vėlinimų vektorius laiko momentu *t* ir $\Psi:\mathbb{R}^d\to\mathbb{R}^n$ yra tolimesnė rekonstrukcija, kuri apsvarsto bendresnės rekonstrukcijos galimybę (nereguliarūs laiko vėlinimai, globalus ar lokalus reikšmės suskaidymas ar triukšmo šalinimo algoritmai) [52]. 1.7 paveikslo viršutinis grafikas kairėje yra tikroji sistema. Jeigu mes galime išmatuoti tik vieno sistemos kintamojo reikšmes (1.7 paveikslo apatinis grafikas kairėje), jo rekonstravimas praktiškai atkartoja pačios sistemos dinamiką (lyginami abu dešinės pusės grafikai).

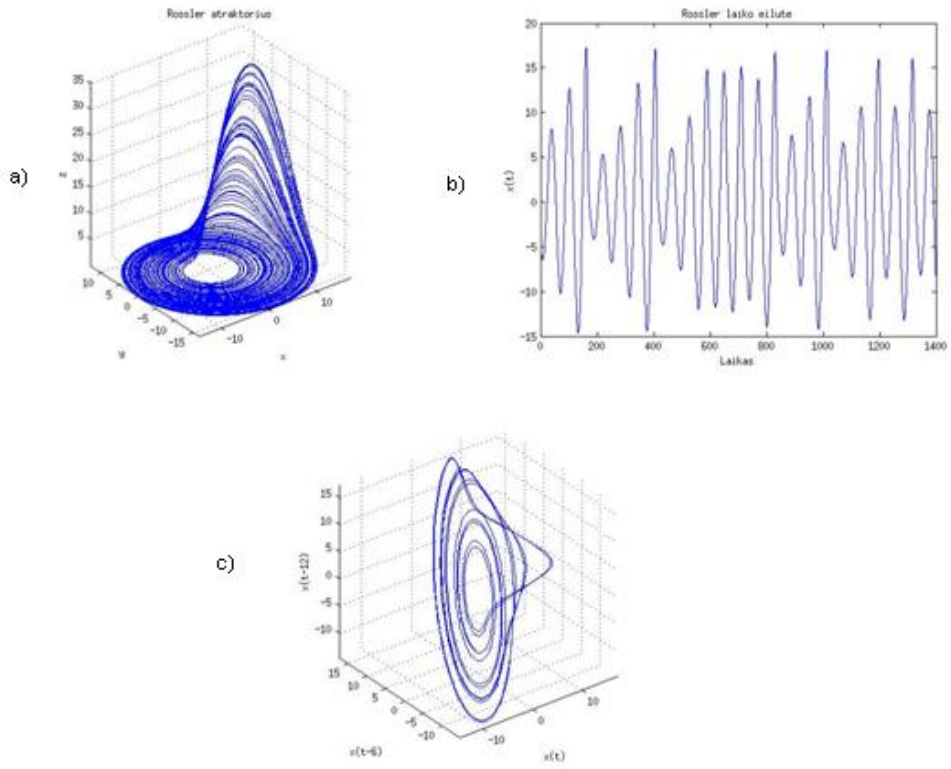


1.7 pav. Matavimo proceso ir dinaminės sistemos atraktoriaus rekonstravimo schema [52]

Tarkime turime Rossler atraktorių. Rossler sistema aprašoma trimis diferencialinėmis lygtimis:

$$\begin{aligned}\frac{dx}{dt}&=-y-z\\ \frac{dy}{dt}&=x+az\\ \frac{dz}{dt}&=b+z(x-c)\end{aligned}\qquad\textbf{(1.22)}$$

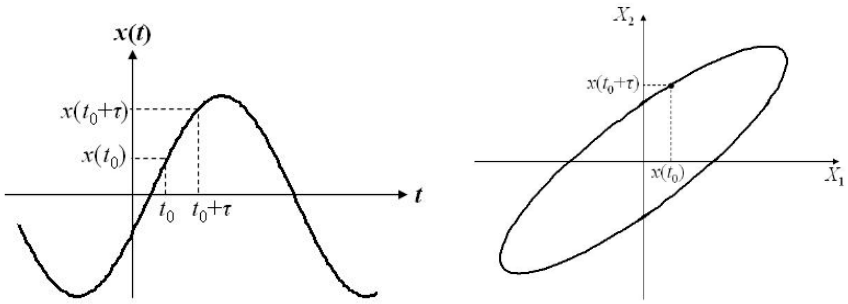
čia *a*, *b* ir *c* yra konstantos ir *t* yra laikas. Laiko eilutė yra sugeneruojama kai $a=0.15$, $b=0.2$ ir $c=10$. Rossler atraktoriaus dinaminės sistemos grafinis vaizdavimas pateiktas 1.8 paveikslo a dalyje, kintamojo *x* laiko eilutės grafinis vaizdavimas 1.8 paveikslo b dalyje. Rekonstruoto atraktoriaus, iš vieno kintamojo $x(t)$ į laiko eilutės vėlinimų erdvę, dinamika vaizduojama 1.8 paveikslo c dalyje.



1.8 pav. a) Rossler atraktorius; b) Rossler laiko eilutė; c) Laiko eilutė rekonstruota į laiko vėlinimų erdvę su laiko vėlinimu 6

Šiame darbe naudojama laiko eilutės rekonstravimo į laiko vėlinimų erdvę metodika, kuri grindžiama maksimaliu atraktoriaus išpūtimu laiko vėlinimų erdvėje [Ragulskio ir Lukoševičiūtės str. 44-45]. Toliau pateikiame šios metodikos aprašymą.

Tarkime nagrinėjame dvimatę vėlinimų koordinačių erdvę ir tiriama harmoninės laiko eilutės sugeneruoto atraktoriaus geometrinę formą, kuri gaunama eilutę rekonstruojant į vėlinimų erdvę [44]:



1.9 pav. Laiko eilutės rekonstravimo schema į dvimatę laiko vėlinimų erdvę [44]

Diskrečioji harmoninė laiko eilutė gali būti išreiškiama tokiu pavidalu:

$$x_n = A\cdot\sin\left(\omega\delta\left(s-1\right)+\varphi\right);\ n=1,2,\ldots,N \qquad \textbf{(1.23)}$$

čia A yra amplitudė, ω - ciklinis dažnis, φ - harmoninės funkcijos fazė, o δ – laiko intervalas tarp dviejų gretimų taškų skaliarinėje laiko eilutėje. Funkcijos reikšmės x_i ir $x_{i+\tau}$ $\left(i=1,2,\ldots,N-\tau\right)$

vėlinimų plokštumoje X_1OX_2 pažymimos tašku $\left(x_i, x_{i+\tau}\right)$, kur τ yra laiko vėlinimas, o X_1 ir X_2 yra vėlinimų koordinačių plokštumos ašys. Harmoninė laiko eilutė rekonstruotoje laiko vėlinimų plokštumoje atvaizduojama į elipsę, kurios lygtis turi išraišką:

$$X_2 = X_1 \cos\left(\omega\tau\delta\right) + \sqrt{A^2 - X_1^2}\sin\left(\omega\tau\delta\right) \quad \textbf{(1.24)}$$

Pagrindinės elipsės ašys yra $X_2 = X_1$ ir $X_2 = -X_1$ (žiūrėti 1.9 paveikslą). Jeigu rekonstravimo į laiko vėlinimų erdvę langas lygus

$$\tau\delta = \frac{2\pi}{\omega}\left(n-1\right),\ n = 1, 2, \ldots; \quad \textbf{(1.25)}$$

elipsė yra suspaudžiama į liniją ant diagonalės $X_2 = X_1$. Lygiai taip pat, kai

$$\tau\delta = \frac{\pi}{\omega} + \frac{2\pi}{\omega}\left(n-1\right),\ n = 1, 2, \ldots; \quad \textbf{(1.26)}$$

elipsė yra suspaudžiama ant diagonalės $X_2 = -X_1$. Kai

$$\tau\delta = \frac{\pi}{2\omega} + \frac{2\pi}{\omega}\left(n-1\right),\ n = 1, 2, \ldots; \quad \textbf{(1.27)}$$

laiko eilutė laiko vėlinimų plokštumoje yra atvaizduojama į apskritimą.

Rekonstruojant signalą į dvimatę vėlinimų erdvę siekiama, kad gautos elipsės plotas būtų kuo didesnis. Taigi, yra apibrėžiamas parametras Q_1, charakterizuojantis atraktoriaus dinamiką rekonstruotoje vėlinimų erdvėje [44]:

$$Q_1 = \frac{S_E}{\pi A^2} \quad \textbf{(1.28)}$$

čia S_E rekonstruotos dvimatės erdvės plokštumoje atvaizduotas elipsės plotas. Čia Q_1 yra $0 \le Q_1 \le 1$. Jei $Q_1 = 1$, tai laiko eilutė atvaizduojama į apskritimą. Blogiausias atvejis gaunamas su $Q_1 = 0$, kai elipsė yra suspaudžiama į liniją. Atvaizduotos elipsės pusašės R_1 ir R_2 yra lygios:

$$\begin{aligned} R_1 &= A\sin\left(\omega\tau\delta\right) / \sqrt{1-\cos\left(\omega\tau\delta\right)} \\ R_2 &= A\sin\left(\omega\tau\delta\right) / \sqrt{1+\cos\left(\omega\tau\delta\right)} \end{aligned} \quad \textbf{(1.29)}$$

Tuomet elipsės plotas:

$$E = \pi \cdot R_1 R_2 = \pi A^2 \left|\sin\left(\omega\tau\delta\right)\right| \quad \textbf{(1.30)}$$

Kokybės parametras Q_1 įgyja pavidalą:

$$Q_1 = \left|\sin\left(\omega\tau\delta\right)\right|. \quad \textbf{(1.31)}$$

1.3.1. LAIKO EILUTĖS REKONSTRAVIMAS REGULIARIAIS LAIKO VĖLINIMAIS

Vėlinimo koordinačių erdvės išmatavimas yra d ir turime $\frac{d(d-1)}{2}$ skirtingų rekonstruoto atraktoriaus plokštuminių projekcijų [44]. Reguliarus laiko eilutės rekonstravimas į laiko vėlinimų erdvę gaunamas tuomet, kai

$$\tau = \tau_1 = \ldots = \tau_{d-1} \tag{1.32}$$

Apibrėžiame rekonstravimo į laiko vėlinimų erdvę kokybės funkciją kaip visų galimų plokštuminių projekcijų rekonstravimo kokybės parametrų vidurkį [44,45]:

$$Q(\tau,\omega) = \frac{2}{d(d-1)}\sum_{k=1}^{d-1}(d-k)Q_k = \frac{2}{d(d-1)}\sum_{k=1}^{d-1}(d-k)\left|\sin(k\omega\delta\tau)\right| \tag{1.33}$$

čia $\frac{2}{d(d-1)}$ atvirkštinis skaičius visų galimų dvimačių plokštumų kombinacijų d- dimensinėje erdvėje, $(d-k)$ rodo kiek yra tokių pačių plotų Q_k , o Q_k charakterizuoja atraktoriaus dinamiką rekonstruotoje erdvėje. Matome, kad $0 \le Q(\tau,\omega) \le 1$. Lygybė $Q = 0$ reiškia, kad eilutė yra suspaudžiama į atkarpą visose galimose projekcijose.

1.3.2. LAIKO EILUTĖS REKONSTRAVIMAS NEREGULIARIAIS LAIKO VĖLINIMAIS

Tarkime turime daugiamatį atvejį ir laiko eilutė yra nebūtinai harmoninė. Skirtingų projekcijų skaičius *d*-matėje vėlinimo koordinačių erdvėje yra $\frac{d(d-1)}{2}$. Rekonstravimo į vėlinimų erdvę kokybės funkcija yra priklausoma nuo parametrų $\tau_1,\ldots,\tau_{d-1},\omega$ [44, 45]. Tuomet

$$Q(\tau_1,\tau_2,\ldots,\tau_{d-1},\omega) = \frac{2}{d(d-1)}\left(\sum_{i=1}^{d-1}\left|\sin(\omega\delta\tau_i)\right| + \sum_{i=1}^{d-2}\left|\sin\left(\omega\delta(\tau_i+\tau_{i+1})\right)\right| + \ldots + \left|\sin\left(\omega\delta\sum_{j=1}^{d-1}\tau_j\right)\right|\right) \tag{1.34}$$

ciklinis dažnis, δ laiko intervalas tarp dviejų gretimų taškų skaliarinėje laiko eilutėje.

Jeigu būtų nagrinėjamas kurios nors laiko eilutės Furjė amplitudinis spektras, būtų galima pastebėti, kad tas signalas susideda iš daugelio harmoninių komponenčių. Visą informaciją apie atraktoriaus formą teikia rekonstruojamos harmoninės eilutės amplitudė, o ne fazė. Kiekvieną iš atskirai paimtų harmoninių komponenčių prie atitinkamai parinktų laiko vėlinimų $\tau_1,\tau_2,\ldots,\tau_{d-1}$ specifiškai paveikia rekonstravimo kokybės funkcija $Q(\tau_1,\tau_2,\ldots,\tau_{d-1},\omega)$. Konkretaus dažnio

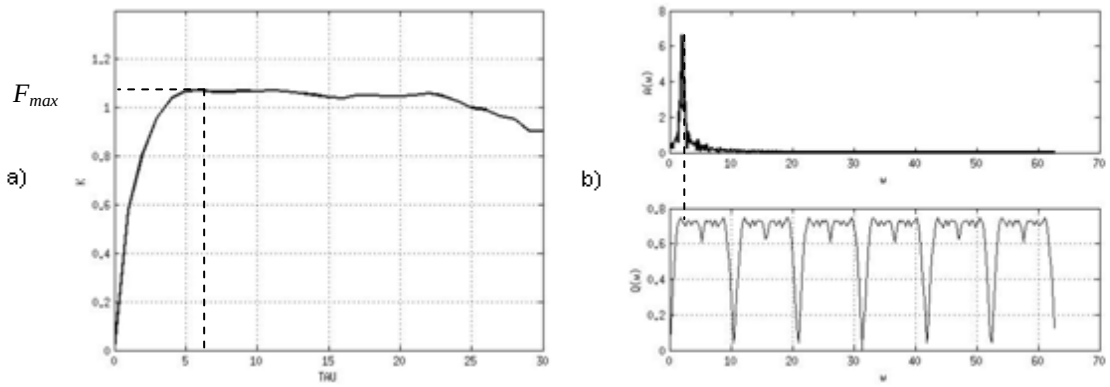
harmoninis signalas atitinkamai atvaizduojamas į tam tikras elipses visose skirtingose projektavimo plokštumose. Jeigu nagrinėjamą signalą galima vertinti kaip tiesinį darinį iš įvairių harmoninių komponenčių, tai galima pritaikyti šį optimalaus harmoninės eilutės rekonstravimo įvertį būtent tam sudėtingo signalo rekonstrukcijos įverčiui kurti, t.y. konstruojama tokia tikslo funkcija, kuri įvertina kaip atskiros diskrečiosios amplitudinio spektro komponentės yra paveikiamos rekonstrukcijos kokybės funkcijos prasme. Tai reprezentuoja netiesioginis integralas šio įverčio skaitiklyje, kai perbėgami visi dažniai ir suskaičiuojama kokias Furjė amplitudinio spektro komponentes labiausiai paveikė kokybės funkcija Q . Laikantis šių prielaidų apibrėžiama tikslo funkcija F , kuri charakterizuoja atraktoriaus vidutinį tūrį rekonstruotoje vėlinimų erdvėje [44]:

$$F\left(\tau_1,\tau_2,\ldots,\tau_{d-1}\right)=\frac{\pi}{2}\frac{\int_0^\infty A(\omega)Q\left(\tau_1,\tau_2,\ldots,\tau_{d-1},\omega\right)d\omega}{\int_0^\infty A(\omega)d\omega} \qquad \textbf{(1.35)}$$

kur $A(\omega)$ pradinės laiko eilutės Furjė amplitudinis spektras. Skaičiuojama netiesioginiai integralai visame dažnių diapazone. Vardiklyje $A(\omega)$ integralas naudojamas tik tam, kad būtų normuojama tikslo funkcijos reikšmė (šis integralas gali būti skaičiuojamas vieną kartą prieš optimizacijos procedūrą). Daugiklis $\frac{\pi}{2}$ yra naudojamas tam, kad tikslo funkcijos reikšmė, rekonstruojant baltąjį triukšmą į d-matę vėlinimų erdvę, būtų lygi vienetui.

Taigi tikslo funkcijos uždavinys yra parinkti tokį laiko vėlinimų rinkinį, su kuriuo tikslo funkcijos reikšmė bus maksimizuojama, t.y. rekonstruotas atraktorius duos turtingiausią vaizdą laiko vėlinimų erdvėje, t.y. šioje erdvėje atraktoriaus tūris bus didžiausias. Tokiu būdu yra gaunama daugiausia informacijos apie tiriamos sistemos kompleksiškumą. Minimizavimo uždavinys nesprendžiamas dėl to, kad minimalaus tūrio suspausta sistemos evoliucija bus beveik identiška visose koordinatėse, todėl informacija apie sistemos kompleksiškumą bus prarasta. Tikslo funkcijos savybės yra aprašytos Minvydo Ragulskio ir Kristinos Lukoševičiūtės straipsnyje [44].

Tarkime, kad Rossler atraktorius rekonstruojamas į laiko vėlinimų erdvę su reguliariais laiko vėlinimais ir tarkime, kad rekonstruotos erdvės išmatavimas yra $d=6$. Tikslo funkcijos įgyjamų reikšmių grafikas vaizduojamas 1.10 paveikslo a dalyje, kai tau kinta nuo 1 iki 30. Kokybės funkcija įgyja pavidalą, demonstruojama 1.10 paveikslo b dalyje, pasirinkus optimalų laiko vėlinimą.



1.10 pav. a) Tikslo funkcijos kitimas keičiantis laiko vėlinimo reikšmiai; b) Furjė amplitudinis spektras (viršuje) ir kokybės funkcijos kreivė, kai parenkamas laiko vėlinimas esant didžiausiai tikslo funkcijos reikšmei

1.3.3. LAIKO EILUTĖS REKOSTRAVIMO DIMENSIJOS PARINKIMAS

Vienas iš svarbiausių atraktoriaus savybių yra ta, kad atraktorius būna dažnai suspaustas objektas vėlinimų erdvėje. Taigi, atraktoriaus taškai įgyja „kaimynus“ šioje erdvėje. Šių „kaimynų“ nauda, yra tokia, kad jie leidžia informaciją, kaip fazinės erdvės kaimynai plėtojasi, panaudoti naujų taškų, esančių ant ar šalia atraktoriaus, laiko evoliucijos prognozavimo lygčių generavimui. Jie taip pat leidžia tiksliai apskaičiuoti sistemos Liapunovo eksponentes [46].

Rekonstravimo dimensijai, kuri yra per maža atraktoriaus atskleidimui, ne visi taškai, kurie išsidėstę šalia vienas kito bus kaimynai dėl jų dinamikos [46]. Kai kurie iš jų bus toli vienas nuo kito ir atrodys kaip „kaimynai“, nes geometrinė atraktoriaus struktūra buvo rekonstruota į mažesnę erdvę. Jei turima dimensija yra d ir $y(n)$ r - tasis artimiausias kaimynas pažymimas $y^{(r)}(n)$, tada Euklidinio atstumo, tarp taško $y(n)$ ir šio kaimyno, kvadratas:

$$R_d^2(n,r) = \sum_{k=0}^{d-1} \left[x(n+kT) - x^{(r)}(n+kT) \right]^2 \qquad \textbf{(1.36)}$$

Pereinant nuo dimensijos d iki dimensijos $d+1$ pridedame $(d+1)$ - ąją koordinatę kiekvienam $y(n)$ vektoriui. Po naujos $(d+1)$ - osios koordinatės pridėjimo atstumas tarp $y(n)$ ir to paties r - tojo artimiausio kaimyno yra:

$$R_{d+1}^2(n,r) = R_d^2(n,r) + \left[x(n+kT) - x^{(r)}(n+kT) \right]^2 \qquad \textbf{(1.37)}$$

Kriterijus surandantis rekonstravimo klaidas, kad didėjant atstumui tarp $y(n)$ ir $y^{(r)}(n)$ yra didelis, kai pereinama iš dimensija d iki dimensijos $d+1$. Sukonstruojamas šis kriterijus paskiriant bet kokį „kaimyną“ kaip „klaidingą kaimyną“, kuriam:

$$\left(\frac{R_{d+1}^2(n,r)-R_d^2(n,r)}{R_d^2(n,r)}\right)^{1/2}=\frac{x(n+dT)-x^{(r)}(n+dT)}{R_d(n,r)}>R_{tol} \tag{1.38}$$

čia R_{tol} yra tam tikra riba. Pakanka atsižvelgti tik į artimiausią kaimyną ($r=1$) ir ištirti kiekvieną tašką ($n=1,2,\ldots,N$) ir nustatyti „klaidingų kaimynų“ skaičių iš N.

Jei $y(n)$ „artimiausias kaimynas“ nėra jam artimas $\left[R_d(n)\approx R_A\right]$ ir yra „klaidingas artimiausias kaimynas“, tada atstumas $R_{d+1}(n)$ pridėjus $(d+1)$ komponentes bus $R_{d+1}(n)\approx 2R_A$.

$$R_A^2=\frac{1}{N}\sum_{n=1}^{N}\left[x(n)-\bar{x}\right]^2,\ \text{o}\ \bar{x}=\frac{1}{N}\sum_{n=1}^{N}x(n) \tag{1.39}$$

Antras kriterijus atrodo taip:

$$\frac{R_{d+1}(n)}{R_A}>A_{tol} \tag{1.40}$$

Rekomenduojama riba A_{tol} yra lygi $A_{tol}=2$ [47].

1.4. PROGNOZAVIMO KOKYBĖS TIKRINIMAS

Norint patikrinti prognozavimo kokybę yra įvedamos įvairios paklaidų matavimo metrikos. Paprasčiausias paklaidų matavimo metrikos pavyzdys yra paklaidų vidurkis [43]:

$$ME=\frac{1}{N}\sum_{t=1}^{N}(x_t-y_t) \tag{1.41}$$

čia x_t ir y_t yra laiko eilutės tikroji reikšmė ir prognozės reikšmė laiko momentu t, kai laiko eilutėje yra N duomenų.

Dažnai lyginant modelių prognozės tikslumus yra pasirenkamos metrikos, kuriose vertinamas paklaidos santykis su tikrosios eilutės reikšmėmis. Viena tokių metrikų yra vidutinė procentinė paklaida:

$$MAE=\frac{1}{N}\sum_{t=1}^{N}\left|y_t-x_t\right| \tag{1.42}$$

Kita dažnai pasitaikanti metrika yra vidutinė absoliučioji procentinė paklaida (angl. Mean Absolute Percentage Error):

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\frac{|x_t - y_t|}{x_t} \tag{1.43}$$

2. TIRIAMOJI DALIS

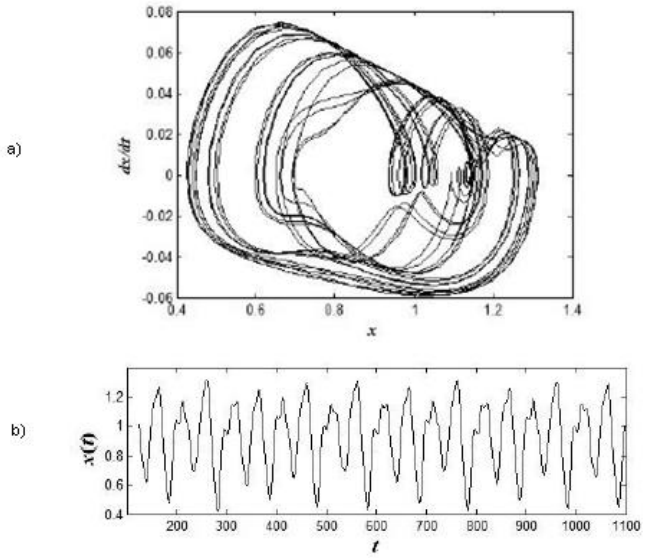
2.1. MACKEY GLASS LAIKO EILUTĖS PROGNOZAVIMAS

Laiko eilutės prognozavimas, remiantis chaotine Mackey-Glass diferencialine lygtimi yra standartinė problema srityse tokiose kaip neuroniniai tinklai, neraiškios logikos sistemos ir hibridinėse sistemose lyginant skirtingų algoritmų mokymąsi ir apibendrinimą [48,49,50,51]. Laiko eilutė, kuri yra sukuriama pasinaudojant Mackey-Glass diferencialine lygtimi apibrėžiama taip:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx(t) \tag{2.1}$$

čia a, b ir τ yra konstantos ir t yra laikas. Laiko eilutė yra sugeneruojama kai $a = 0.2$, $b = 0.1$ ir $\tau = 17$.

Šios diferencialinės lygties dalinio sprendinio grafinis vaizdavimas laiko vėlinimų plokštumoje pateiktas 2.1 paveikslo a dalyje ir laiko eilutės grafinis vaizdavimas 2.1 paveikslo b dalyje.

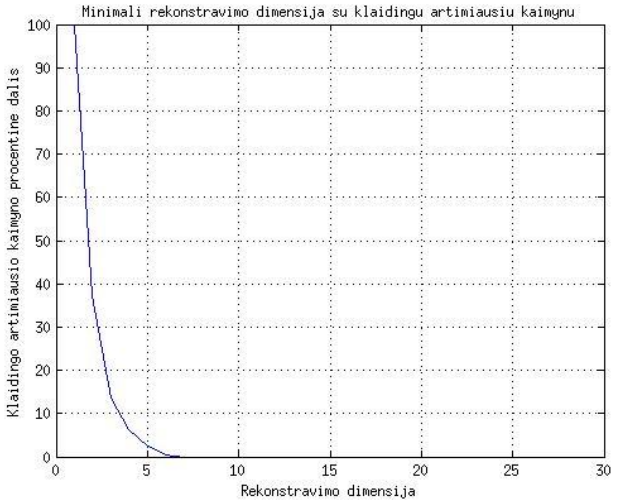


2.1 pav. a) Mackey-Glass diferencialinės lygties dalinio sprendinio grafinis vaizdavimas; b) Mackey – Glass laiko eilutės grafinis vaizdavimas

2.1.1. MACKEY-GLASS LAIKO EILUTĖS REKONSTRAVIMAS REGULIARIAI LAIKO VĖLINIMAIS IR RBF TINKLO APMOKYMAS

Pirmiausia panagrinėkime atvejį kai Mackey – Glass laiko eilutę rekonstruosime į laiko vėlinimų erdvę su reguliariais laiko vėlinimais. Tolimesniuose skyreliuose laiko eilutės bus rekonstruojamos į laiko vėlinimų erdvę tik su nereguliariais laiko vėlinimais. Kad eilutė būtų optimaliai išskleista reikia nustatyti į kokio išmatavimo laiko vėlinimų erdvę rekonstruosime laiko eilutę.

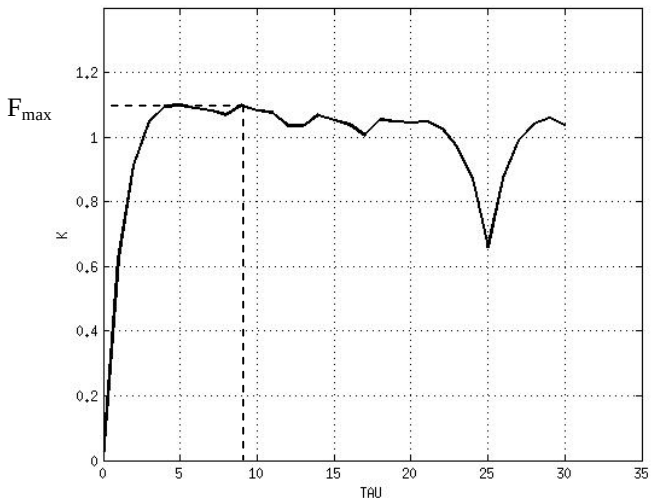
Pasinaudojant „klaidingo artimiausio kaimyno“ algoritmo (1.37-1.41) formulėmis randama minimali rekonstravimo dimensija. Dimensijų grafikas pateiktas 2.2 paveiksle.



2.2 pav. Mackey-Glass laiko eilutės rekonstravimo dimensijų grafikas

Pagal turimą grafiką galime apytiksliai spręsti, kad rekonstruojamos erdvės išmatavimo matas yra $d = 6$.

Turėdami erdvės išmatavimą, galime nustatyti optimalų reguliaraus rekonstravimo į laiko vėlinimų erdvę laiko vėlinimą. Iš 2.3 paveikslo galima susidaryti vaizdą, kad tikslo funkcijos maksimali reikšmė gaunama kai $\tau = 4$.



2.3 pav. Mackey - Glass atraktoriaus rekonstravimo į šešiamatę laiko vėlinimų erdvę optimalaus vėlinimo radimas

Lentelėje pateikiama keletas tikslo funkcijos reikšmių esant rekonstruojamos erdvės išmatavimui $d=6$.

2.1 lentelė.

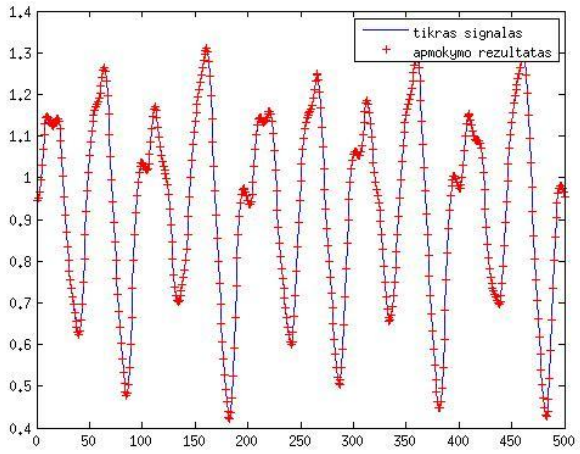
Laiko vėlinimų rinkiniai ir tikslo funkcijos reikšmės

Rekonstravimo būdas	Laiko vėlinimų erdvės matavimas *d*	Reguliarus rekonstravimas	$F(\tau_1,\ldots,\tau_{d-1})$
Reguliarus	6	(4,4,4,4,4)	1.0943
		(5,5,5,5,5)	1.0982
		(9,9,9,9,9)	1.0990
		(10,10,10,10,10)	1.0812
		(11,11,11,11,11)	1.0759

Taigi, iš pateiktos lentelės matoma, kad tikslo funkcijos reikšmė yra didžiausia kai $\tau=9$, t.y. $F(9,9,9,9,9)=1.0990$. Ši nustatyta laiko vėlimo reikšmė yra geriausia esant išmatavimui $d=6$, tačiau tai nereiškia, kad ji bus tinkama esant kitam rekonstruojamos erdvės išmatavimui, pavyzdžiui esant rekonstruojamos erdvės išmatavimui $d=5$.

Mes prognozuojame $x(t+9)$ reikšmę iš praeities reikšmių $x(t-36)$, $x(t-27)$, $x(t-18)$, $x(t-9)$ ir $x(t)$.

Prognozavimui naudojamas RBF neuroninis tinklas. Pirmiausia yra surandami RBF neuroninio tinklo parametrai *spread* ir *MSEgoal*. Suradus parametrus, toliau RBF tinklas yra apmokomas. Taigi, 500 eilutės narių yra naudojami neuroninio tinklo apmokymui, likę eilutės nariai yra naudojami testavimui. Tikras signalas ir gautos reikšmės jau apmokius tinklą vaizduojamos 2.4 paveiksle.

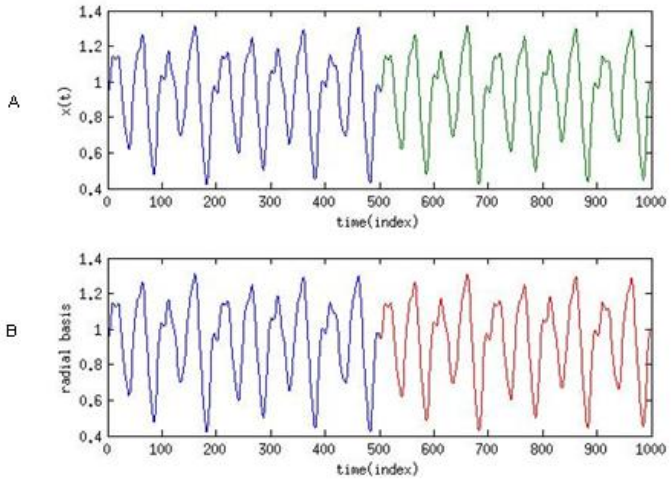


2.4 pav. Mackey - Glass laiko eilutės 500 reikšmių prieš ir po apmokymo

Iš 2.4 paveikslo matoma, kad parametrai parinkti tinkamai ir kad skirtumai tarp turimos laiko eilutės ir po apmokymo gautos eilutės yra nežymūs. Mackey – Glass eilutei *spread* parametras yra lygus 3.4755 ir $MSEgoal=2.5690e-05$. Pasiekus šias konstantas modelis sustoja skaičiavęs.

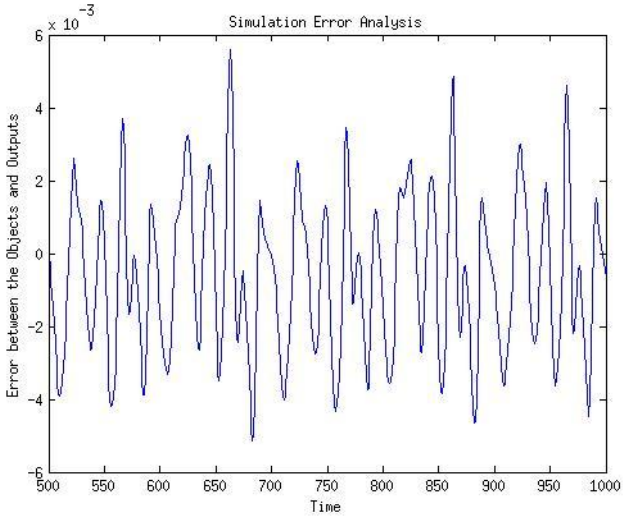
2.1.2. MACKEY-GLASS LAIKO EILUTĖS, PAGRĮSTOS REGULIARIU REKONSTRAVIMU, PROGNOZAVIMAS

Konstruojamas laiko eilutės 1000 duomenų vektorius; pirmieji 500 naudojami MATLAB‘o funkcijos RBF apmokymui, o kiti duomenys naudojami prognozės tikslumui tikrinti. RBF neuroninio tinklo prognozės rezultatai pavaizduoti 2.5 paveikslo B dalyje. Mėlynos spalvos grafikas vaizduoja prognozės rezultatus apmokymo metu.



2.5 pav. Viršutinis grafikas žymi tikrą signalą, antras grafikas rodo: 500 RBF tinklo apmokymo reikšmes ir 500 suprognozuotas reikšmes

Norint patikrinti ar modelis gerai prognozuoja, bus stebimos prognozės skirtuminės paklaidos, kurios demonstruojamos 2.6 paveiksle.



2.6 pav. RBF neuroninio tinklo paklaida

2.1.3. MACKEY-GLASS LAIKO EILUTĖS, PAGRĮSTOS NEREGULIARIU REKONSTRAVIMU, PROGNOZAVIMAS

Tęsiamas darbas su Mackey – Glass laiko eilute, tik dabar ji rekonstruojama į laiko vėlinimų erdvę nereguliariais laiko vėlinimais. Šios eilutės laiko vėlinimų erdvės išmatavimas yra $d = 6$. Laiko vėlinimų deriniui rasti naudojant genetinius algoritmus. Genetiniai algoritmai yra labai globalios, lygiagrečios ir atsitiktinės paieškos algoritmai, kurie remiasi evoliucijos teorija, natūralia atranka. Tai modelis pritaikantis evoliucijos teoriją optimizavimo uždaviniams. Detaliau šie algoritmai aprašyti autorių straipsniuose [55-56]. Darbe naudojami jau sukurta genetinių algoritmų *Matlab* paketo programa.

Surandama optimali laiko vėlinimų aibė, perrinkus visus įmanomus laiko vėlinimų rinkinius intervale [1, 30]. Lentelėje 2.2 pateikta keletas laiko vėlinimų derinių ir jų tikslo funkcijos reikšmių:

2.2 lentelė.

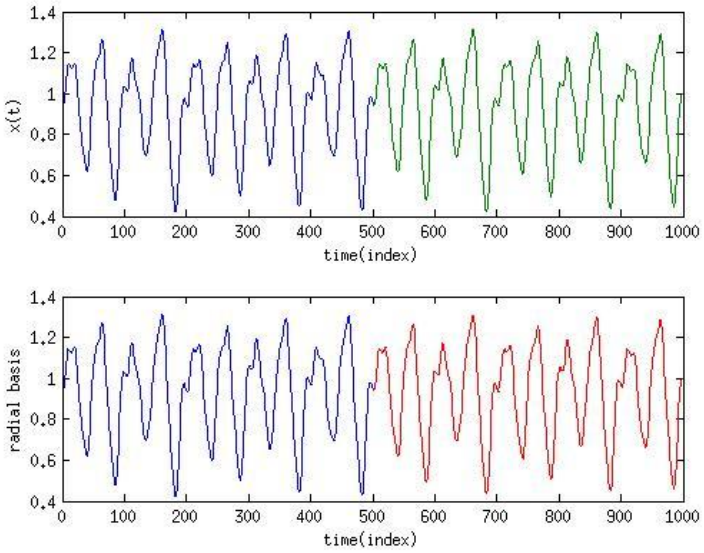
Laiko vėlinimų rinkiniai ir tikslo funkcijos reikšmės

Laiko vėlinimų erdvės matavimas *d*	Nereguliarus rekonstravimas	$F(\tau_1,\ldots,\tau_{d-1})$
6	(6,4,5,5,6)	1.1044
	(7,5,4,4,7)	1.1023
	(5,5,5,4,5)	1.1010
	(6,5,5,4,7)	1.1049
	(7,5,5,5,6)	1.1013
	(7,4,5,4,7)	1.1056

Iš lentelės matoma, kad didžiausia tikslo funkcijos reikšmė gaunama, esant laiko vėlinimų rinkiniui $\{7,4,5,4,7\}$, t.y. $F(7,4,5,4,7) = 1.1056$.

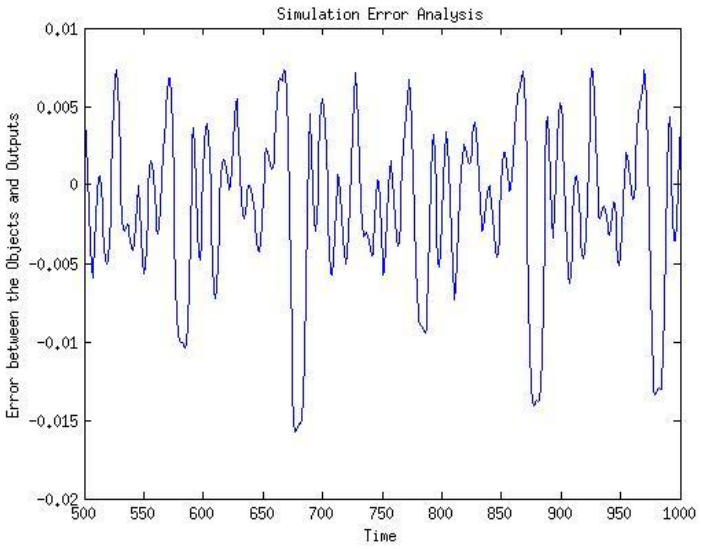
Mes prognozuojame $x(t+7)$ reikšmę iš praeities reikšmių $x(t-20)$, $x(t-13)$, $x(t-9)$, $x(t-4)$ ir $x(t)$. Radialinės bazės funkcijos neuroninio tinklo parametrai gaunami tokie: $spread = 2.9188$ ir $MSEgoal = 2.5690e-05$.

Mackey – Glass diferencialinės lygties sprendinio *x(t)* dinamika laike vaizduojama 2.7. paveikslo A dalyje. Konstruojamas laiko eilutės 1000 duomenų vektorius; pirmieji 500 naudojami Matlab‘o funkcijos RBF apmokymui, o kiti duomenys naudojami prognozės tikslumui tikrinti. RBF neuroninio tinklo prognozės rezultatai pavaizduoti 2.7. paveikslo B dalyje.



2.7 pav. Viršutinis grafikas žymi tikrą signalą, antras grafikas rodo: 500 RBF tinklo apmokymo reikšmes ir 500 suprognozuotas reikšmes

Tikrinant ar modelis gerai prognozuoja, stebimos prognozės skirtuminės paklaidos, kurios demonstruojamos 2.8. paveiksle. Skirtuminių paklaidų grafikai, naudojant ANFIS bei FITNET prognozavimo modelius, pateikti 3 priede.



2.8 pav. RBF neuroninio tinklo paklaida

Tolesnių skyrelių skaičiavimo struktūra susideda iš tokių žingsnių: pirmiausia nustatomas rekonstruojamos vėlinimų erdvės matavimas, tada randamas laiko vėlinimų rinkinys; trečiame žingsnyje RBF neuroninis tinklas adaptuojamas uždavinio sprendimui. Galiausiai, kai jau į RBF neuroninį tinklą adaptuojama laiko vėlinimų aibė, galima prognozuoti laiko eilutės ateities reikšmes.

Kad aiškiau būtų galima pastebėti prognozavimo kokybę vertinamas MAPE metrika, kuri yra pateikiama 2.3 lentelėje. Ši metrika taip pat rodo prognozės tikslumo pagerėjimą.

2.3 lentelė.

Mackey - Glass laiko eilutės RBF neuroninio tinklo prognozės paklaidų palyginimas

Rekonstravimo būdas	Vėlinimų erdvės matavimas d	Laiko vėlinimų rinkinys $(\tau_1, \tau_2, \ldots, \tau_{d-1})$	$F(\tau_1, \tau_2, \ldots, \tau_{d-1})$	Prognozavimo paklaidų MAPE (%)
Nereguliarus	5	$\{10,10,9,2\}$	1.1225	0.3345
Reguliarus	6	$\{9,9,9,9,9\}$	1.0990	0.2178
Nereguliarus	6	$\{7,4,5,4,7\}$	1.1056	0.5220

Prognozavimo tikslumui nustatyti šio modelio rezultatai palyginami su FITNET neuroninio tinklo prognozėmis, kuris aprašytas 1.2.2 skyrelyje, bei su MATLAB įrankiuose pateikiamo adaptyvaus neuroninio tinklo, paremto neraiškių išvadų sistemos funkcija ANFIS, prognozėmis. ANFIS architektūros trumpą aprašymą galima rasti 1.2.3 skyrelyje, o detalesnį [57]. Skirtingų modelių MAPE paklaidos bei skaičiavimo greičiai pateikti 2.4 lentelėje:

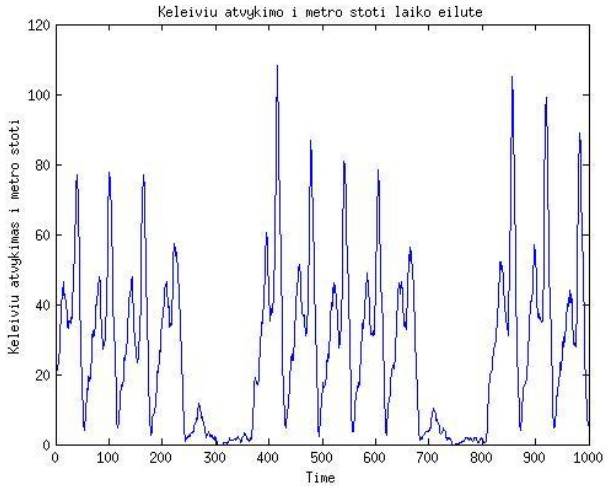
2.4 lentelė.

Mackey-Glass laiko eilutės prognozės paklaidų palyginimas

	Mackey-Glass laiko eilutė	
	MAPE paklaida (%)	Skaičiavimo greičiai (sekundės)
RBF neuroninių tinklų prognozė	0.5220	82.821864
ANFIS tinkle prognozė	0.0435	21.597643
FITNET neuroninis tinklas (be rekonstravimo)	0.9944	46.296269
FITNET neuroninis tinklas (su rekonstravimu)	0.7188	47.670482

2.2. PROGNOZĖS MODELIO TAIKYMAI KELEIVIŲ ATVYKIMO Į METRO STOTĮ LAIKO EILUTEI.

Prognozuojama realaus pasaulio laiko eilutė, kuri yra apibūdinama keleivių atvykimo į San Diego metro stotį 15 minučių laiko intervalu. Šios laiko eilutės vaizdas pateiktas 2.9 paveiksle. Imami šios laiko eilutės duomenys nuo 11 iki 1010. Laiko eilutė yra normuojama.



2.9 pav. Keleivių atvykimo į San Diego metro stotį laiko eilutė

2.2.1. KELEIVIŲ ATVYKIMO Į METRO STOTĮ LAIKO EILUTĖS PROGNOZAVIMAS, PAGRĮSTAS LAIKO EILUTĖS NEREGULIARIU REKONSTRAVIMU

Analogiškai, kaip ir Mackey-Glass laiko eilutės atveju nustatoma, į kokio matavimo erdvę reikia rekonstruoti tą laiko eilutę. Vėl pasinaudojant „klaidingo artimiausio kaimyno“ algoritmu nustatoma, kad optimalus vėlinimų erdvės matavimas yra $d = 7$ (grafikas pateiktas 1 priede).

Lentelėje pateikta keletas laiko vėlinimų derinių ir jų tikslo funkcijos reikšmių:

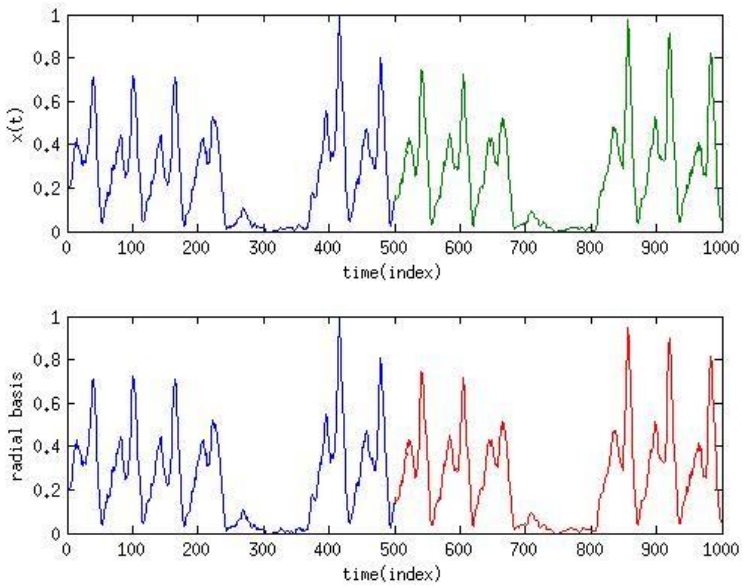
2.5 lentelė.

Realaus pasaulio laiko eilutės laiko vėlinimų deriniai ir tikslo funkcijos reikšmės

Laiko vėlinimų erdvės matavimas *d*	Nereguliarus rekonstravimas	$F(\tau_1,\ldots,\tau_{d-1})$
7	$(22,24,21,20,24,26)$	1.0383
	$(19,18,20,18,20,19)$	1.0409
	$(23,17,26,22,21,23)$	1.0457
	$(18,21,19,19,19,19)$	1.0437
	$(20,24,24,22,23,25)$	1.0386

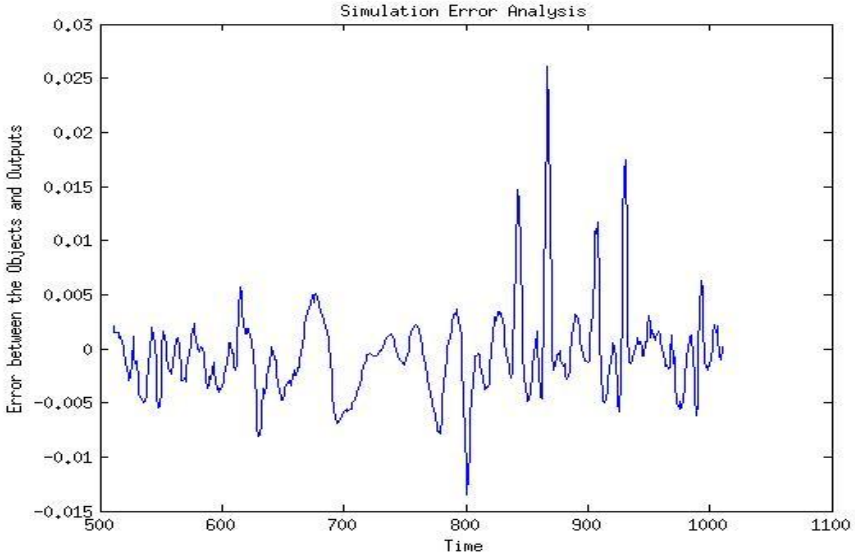
Optimalus laiko vėlinimų rinkinys $\{23,17,26,22,21,23\}$ geriausiai reprezentuoja rekonstruojamą atraktorių septynmatėje vėlinimų erdvėje. Tikslo funkcijos reikšmė gaunama $F(23,17,26,22,21,23)=1.0457$. Taigi, prognozuojama $x(t+23)$ reikšmė iš praeities duomenų $x(t-109)$, $x(t-86)$, $x(t-69)$, $x(t-43)$, $x(t-21)$ ir $x(t)$. Konstruojamas 1000 duomenų vektorius. Pirmi 500 duomenų panaudojami RBF neuroninio tinklo apmokymui, likę prognozės

tikslumo nustatymui. Radialinės bazės funkcijos neuroninio tinklo parametrai gaunami tokie: $spread = 2.9650$ ir $MSEgoal = 2.1228e-05$. RBF neuroninio tinklo apmokymui panaudojami 8 neuronai. Prognozės rezultatai pateikiami 2.10 paveiksle.



2.10 pav. Viršutinis grafikas žymi tikrą signalą, antras grafikas rodo: 500 RBF tinklo apmokymo reikšmes ir 500 suprognozuotas reikšmes

Prognozės skirtuminės paklaidos demonstruojamos 2.11 paveiksle. Skirtuminių paklaidų grafikai, naudojant ANFIS bei FITNET prognozavimo modelius, pateikti 2 priede.



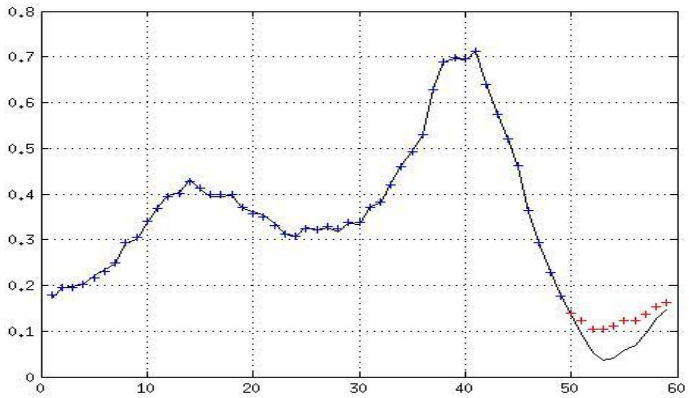
2.11 pav. RBF neuroninio tinklo paklaida

2.6 lentelė.

Keleivių atvykimo į metro stotį prognozės paklaidų palyginimas

	Keleivių atvykimas į metro stotį laiko eilutė	
	MAPE paklaida (%)	Skaičiavimo greičiai (sekundės)
RBF neuroninių tinklų prognozė	0.0755	83.376030
ANFIS tinkle prognozė	1.9291	132.254658
FITNET neuroninis tinklas (be rekonstravimo)	47.450	47.281390
FITNET neuroninis tinklas (su rekonstravimu)	0.0232	48.179175

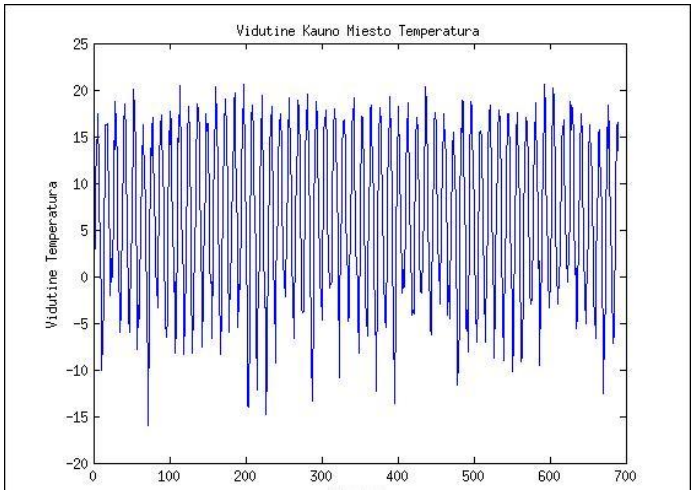
Toliau prognozuojama tą pačią realaus pasaulio laiko eilutę, kai pirmi 50 nariai yra naudojami RBF tinklo apmokymui. Didžiausia tikslo funkcijos reikšmė gaunama *F(10,7,11,11,7,11)=1.1197*. RBF neuroninio tinklo parametrai gaunami tokie: $spread = 0.8754$ ir $MSEgoal = 1.0299e-05$. Apmokant RBF neuroninį tinklą yra panaudojami 9 neuronai. Tokio prognozavimo atveju MAPE paklaida gaunama kur kas didesnė. *MAPE=3.1926* ir skaičiavimo greitis yra *15.2872* sekundės. Prognozės rezultatai pateikti 2.12 paveiksle. Palyginimui buvo naudojamas svertinis slenkamojo vidurkio modelis, kurio atveju paklaida *MAPE=76.9479* [54].



2.12 pav. Realaus pasaulio eilutės prognozė RBF tinklais. Juoda linija vaizduoja realią eilutę, mėlyni pliusai vaizduoja apmokymo reikšmes, raudoni pliusai prognozes.

2.3. PROGNOZĖS MODELIO TAIKYMAI VIDUTINĖS KAUNO MIESTO TEMPERATŪROS LAIKO EILUTEI.

Prognozuojama realaus pasaulio laiko eilutė, kuri yra apibūdinama vidutine Kauno miesto temperatūra laikotarpiu nuo 1922m. gegužės mėn. iki 1980m. rugpjūčio mėn. Šios laiko eilutės vaizdas pateiktas 2.13 paveiksle. Imami šios laiko eilutės duomenys nuo 11 iki 610. Laiko eilutė yra normuojama.



2.13 pav. Vidutinė Kauno miesto temperatūra 1922m. gegužės mėn. - 1980m. rugpjūčio mėn.

2.3.1. VIDUTINĖS TEMPERATŪROS KAUNO MIESTE LAIKO EILUTĖS PROGNOZAVIMAS, PAGRĮSTAS LAIKO EILUTĖS NEREGULIARIU REKONSTRAVIMU

Pirmiausia nustatoma, į kokio matavimo erdvę reikia rekonstruoti šią laiko eilutę. Optimalus vėlinimų erdvės matavimas yra $d = 6$ (grafikas 1 priede).

Lentelėje pateikta keletas laiko vėlinimų derinių ir jų tikslo funkcijos reikšmių:

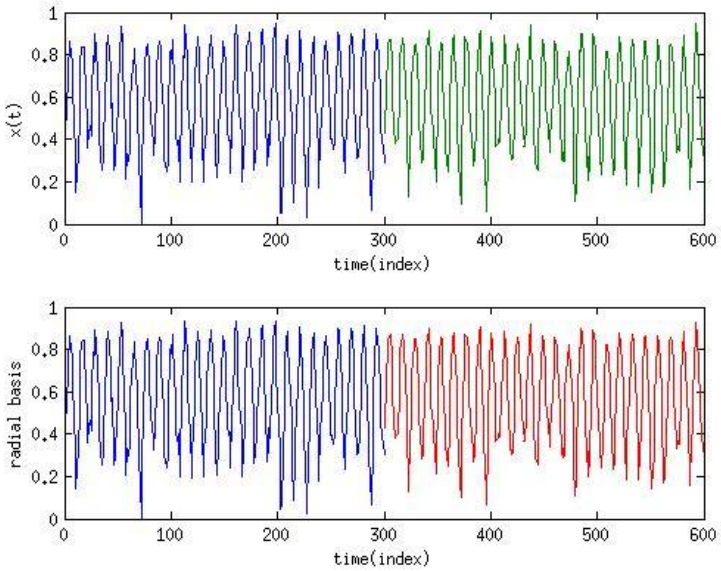
2.7 lentelė.

Vidutinės oro temperatūros laiko eilutės rekonstravimo į laiko vėlinimų erdvę laiko vėlinimų rinkiniai

Laiko vėlinimų erdvės matavimas d	Nereguliarus rekonstravimas	$F(\tau_1,\ldots,\tau_{d-1})$
6	$(8,14,9,16,10)$	1.0546
	$(9,10,13,9,17)$	1.0551
	$(9,7,10,9,14)$	1.0547
	$(16,16,17,14,20)$	1.0542
	$(9,14,9,17,21)$	1.0573

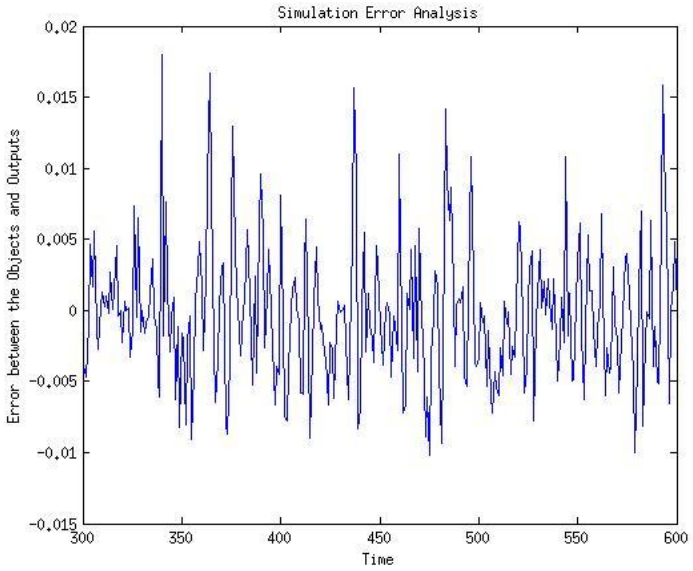
Surandama, kad optimalus laiko vėlinimų rinkinys $\{9,14,9,17,21\}$ geriausiai reprezentuoja rekonstruojamą atraktorių šešiamatėje vėlinimų erdvėje. Tikslo funkcijos reikšmė gaunama

$F(9,14,9,17,21)=1.0573$. Taigi prognozuojama $x(t+21)$ reikšmė iš praeities duomenų $x(t-49)$, $x(t-40)$, $x(t-26)$, $x(t-17)$ ir $x(t)$. Konstruojamas 600 duomenų vektorius, kai laikas kinta nuo $t = 11$ iki 610. Pirmi 300 duomenų panaudojami RBF neuroninio tinklo apmokymui, likę prognozės tikslumo nustatymui. RBF neuroninio tinklo parametrai gaunami tokie: $spread = 1.9187$ ir $MSEgoal = 2.5764e-05$. Prognozės rezultatai pateikiami 2.14. paveiksle.



2.14 pav. Viršutinis grafikas žymi tikrą signalą, antras grafikas rodo: 300 RBF tinklo apmokymo reikšmes ir 300 suprognozuotas reikšmes

Stebimos prognozės skirtuminės paklaidos, kurios demonstruojamos 2.15 paveiksle.



2.15 pav. RBF neuroninio tinklo paklaida

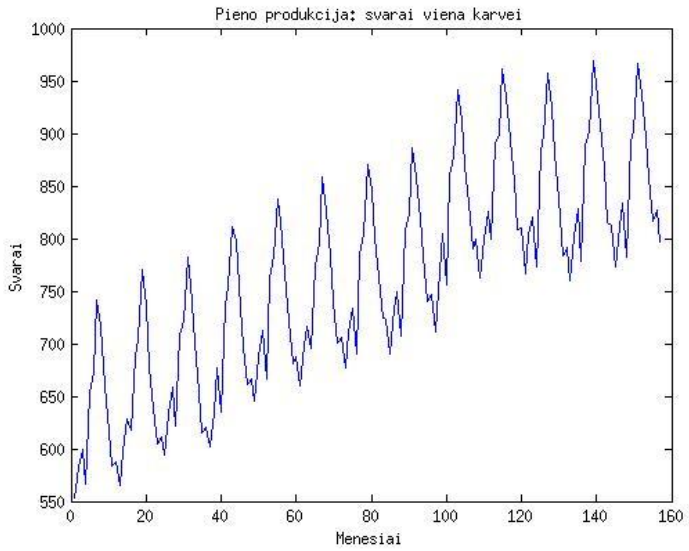
2.8 lentelė.

Vidutinės oro temperatūros laiko eilutės prognozės paklaidų palyginimas

	Vidutinės temperatūros Kauno mieste laiko eilutė	
	MAPE paklaida (%)	Skaičiavimo greičiai (sekundės)
RBF neuroninių tinklų prognozė	0.7214	54.565187
ANFIS tinklo prognozė	20.9351	14.394825
FITNET neuroninis tinklas (be rekonstravimo)	10.5277	30.426879
FITNET neuroninis tinklas (su rekonstravimu)	0.1903	35.023313

2.4. PROGNOZĖS MODELIO TAIKYMAI PIENO PRODUKCIJOS LAIKO EILUTEI.

Prognozuojama realaus pasaulio laiko eilutė, kuri yra apibūdinama mėnesine pieno produkcija (svarai vienai karvei) laikotarpiu nuo 1962m. sausio mėn. iki 1975m. gruodžio mėn. Šios laiko eilutės vaizdas pateiktas 2.16 paveiksle. Imami šios laiko eilutės duomenys nuo 11 iki 40. Laiko eilutė yra normuojama.

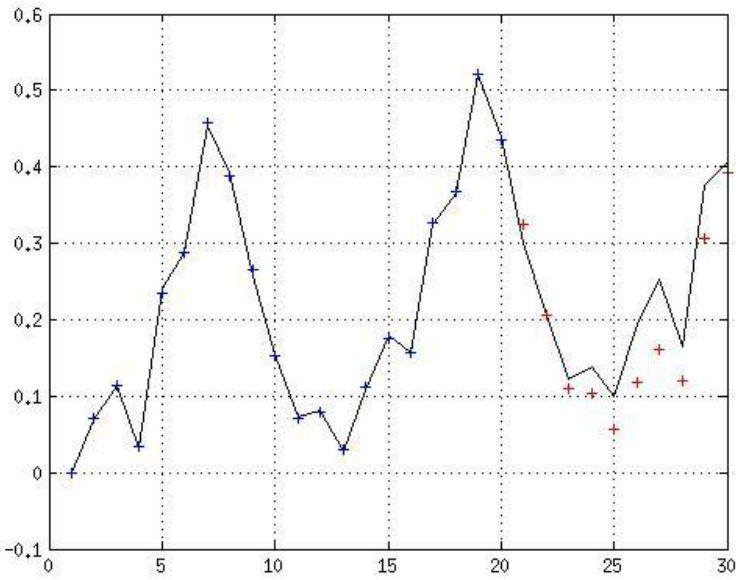


2.16 pav. Pieno produkcija 1962m. sausio mėn. - 1975m. gruodžio mėn.

2.4.1. PIENO PRODUKCIJOS LAIKO EILUTĖS PROGNOZAVIMAS, PAGRĮSTAS LAIKO EILUTĖS NEREGULIARIU REKONSTRAVIMU

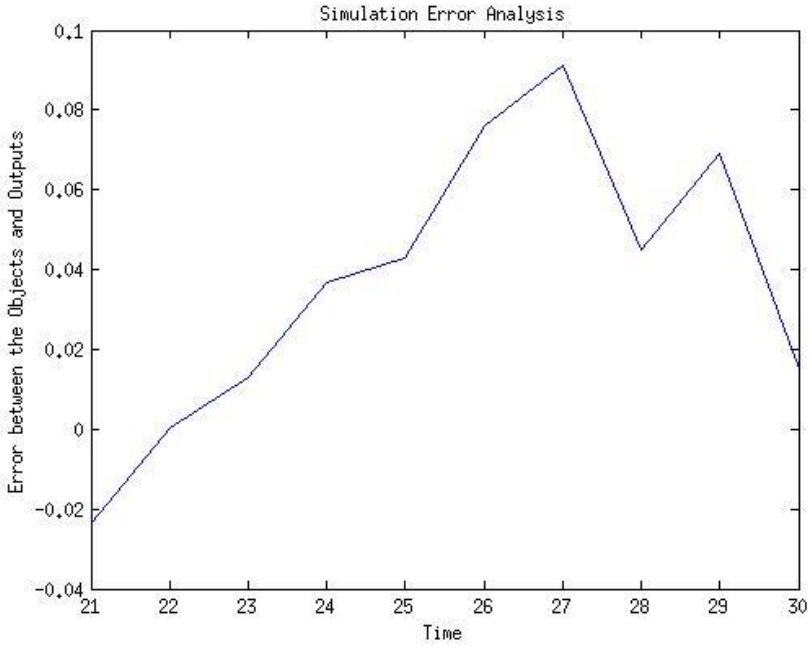
Analogiškai, kaip ir prieš tai buvusių laiko eilučių atveju nustatoma, į kokio matavimo erdvę reikia rekonstruoti šią laiko eilutę. Nustatoma, kad optimalus vėlinimų erdvės matavimas yra $d = 6$ (grafikas 1 priede).

Pasinaudojus genetiniais algoritmais surandama, kad optimalus laiko vėlinimų rinkinys $\{17,12,19,18,9\}$ geriausiai reprezentuoja rekonstruojamą atraktorių šešiamatėje vėlinimų erdvėje. Tikslo funkcijos reikšmė gaunama $F(17,12,19,18,9) = 1.0991$. Prognozuojama $x(t+9)$ reikšmė iš praeities duomenų $x(t-66)$, $x(t-49)$, $x(t-37)$, $x(t-18)$ ir $x(t)$. Konstruojamas 30 duomenų vektorius, kai laikas kinta nuo $t = 11$ iki 40. Pirmi 20 duomenų panaudojami RBF neuroninio tinklo apmokymui, likę prognozės tikslumo nustatymui. RBF neuroninio tinklo parametrai gaunami tokie: $spread = 0.5619$ ir $MSEgoal = 1.1993e-05$. Apmokant šį tinklą yra panaudojami 15 neuronai. Prognozės rezultatai pateikiami 2.17. paveiksle .



2.17 pav. Juoda linija žymi tikrą signalą, mėlyni pliusai žymi 20 RBF tinklo apmokymo reikšmes, o raudoni pliusai žymi 10 suprognozuotas reikšmes

Stebimos prognozės skirtuminės paklaidos, kurios demonstruojamos 2.18 paveiksle.



2.18 pav. RBF neuroninio tinklo paklaida

2.9 lentelė.

Pieno produkcijos laiko eilutės prognozės paklaidų palyginimas

	Pieno produkcijos laiko eilutė	
	MAPE paklaida (%)	Skaičiavimo greičiai (sekundės)
RBF neuroninių tinklų prognozė	21.3355	15.058114
FITNET neuroninis tinklas (su rekonstravimu)	32.4376	12.707906
Svertinis slenkamas vidurkis	58.3192	20.15746

3. PROGRAMINĖ REALIZACIJA IR INSTRUKCIJA VARTOTOJUI

Programa kurta MATLAB R2011a paketu. Šia programine įranga buvo sukurta radialinės bazės funkcijos neuroninio tinklo prognozavimo programa, pavadinimu *RBF_prognoze.m*.

Prieš pradedant prognozavimą, pirmiausia yra nustatoma eilutės rekonstravimo dimensija, pasinaudojant *Dimensija_fnn.m* . Tuomet pasinaudojant genetiniu algoritmu (*pagrindine.m*) surandami laiko vėlinimai. Tuomet yra atliekama prognozė su RBF neuroniniu tinklu.

Pirmiausia yra nuskaitomi duomenų failai, kurių formatai aprašomi formatu *.txt*, naudojant Matlab‘o funkciją *load* (mgdata.txt, passenge_rarrivals_1201.txt, average_monthly_temp_Kaunas.txt, monthly_milk_production.txt), tuomet jei reikia duomenys yra normuojami ir nubraižoma mūsų turima laiko eilutė:

```
load mgdata.txt
a = mgdata;
duom = a(:,2);
time = a(:,1);
%length(a)
P = duom';
%Duomenu normavimas
%P=(duom-min(duom))/(max(duom)-min(duom));
%duom1
%P=P';
index=101:1200;
figure(4);
plot(1:1100,duom(index));
xlabel('Laikas');
ylabel('x(t) ');
title('Mackey - Glass laiko eilute')
```

Paruošiame duomenis taip, kad galėtume juos panaudoti RBF neuroninio tinklo prognozavimui. Tarkime prognozuojame Mackey-Glass laiko eilutę su reguliariais laiko vėlinimais. Pagal žemiau pateiktą MATALB kodą mes prognozuojame $x(t+9)$, kai turime eilutės praeities duomenis $x(t-36),\ x(t-27), x(t-18), x(t-9)$ ir $x(t)$. Toks duomenų rinkinys yra gaunamas rekonstruojant laiko eilutę į $d=6$ mato erdvę su reguliariais laiko vėlinimais tarp gretimų duomenų.

Iš 1000 duomenų vektoriaus mes pirmuosius 500 duomenų panaudojame RBF tinklo apmokymui:

```
P1=zeros(6,N);   %ivestis P = RxQ matrica
Start=Start1;
P1(1,:)=P(Start:Start+N-1);
Start=Start+tau1;
P1(2,:)=P(Start:Start+N-1);
Start=Start+tau2;
P1(3,:)=P(Start:Start+N-1);
Start=Start+tau3;
P1(4,:)=P(Start:Start+N-1);
Start=Start+tau4;
P1(5,:)=P(Start:Start+N-1);
Start=Start+tau5;
P1(6,:)=P(Start:Start+N-1);
%Start=Start+tau6;
%P1(7,:)=P(Start:Start+500-1);
```

Tarkime norime padidinti dimensijų skaičių nuo $d=6$ iki $d=7$. Tuomet RBF procedūroje `P1` turės 7, o ne 6 stulpelius.

Tuomet yra apskaičiuojami RBF neuroninio tinklo parametrai bei sukuriamas pats neuroninis tinklas, pasinaudojus MATLAB paketo pateikta neuroninio tinklo funkcija, kuri yra aprašoma eilute:

$$newrb(P1,\ T,\ MSEgoal,\ spread,\ mn,\ df) \qquad \textbf{(2.2)}$$

čia $P1$ yra įvesties vektoriai, T yra testavimo vektorius, *MSEgoal* – kvadratinė paklaida, kurią pasiekus, modelis sustoja skaičiavęs, *spread* yra išsibarstymo konstanta, *mn* – maksimalus neuronų skaičius, *df* – kas kiek neuronų yra išvedami rezultatai ekrane. *MSEgoal* ir *spread* yra apskaičiuojami MATLAB pakete pagal formules:

$$spread = 0.5 \cdot mean\left(median\left(dist\left(P1, P1'\right)\right)\right); \qquad \textbf{(2.3)}$$

Išsibarstymo konstanta apskaičiuojama taip: pirmiausia randamas Euklido atstumas tarp įvesties vektoriaus ir jo transponuoto vektoriaus. Euklido atstumas skaičiuojamas pagal formulę:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \quad \textbf{(2.4)}$$

Tuomet randama medianos ir galiausiai randamas medianų vidurki, kuris yra padauginamas iš 0.5. O *MSEgoal* apskaičiuojamas:

$$MSEgoal = MSEtrain00 / 2000 \quad \textbf{(2.5)}$$

čia *MSEtrain*00 randama pagal formule:

$$MSEtrain00 = mse\left(T - \left(repmat\left(mean(T,2),1,Ntrain\right)\right)\right) \quad \textbf{(2.6)}$$

kur *Ntrain* yra testavimo vektoriaus dydis. Norint apskaičiuoti *MSEgoal*, pirmiausia yra surandamas testavimo vektoriaus dydis. *MSEgoal* parametro suradimui naudojamos MATLAB funkcijos „*mean*" – vidurkis ir „*repmat*" – sukuria naują matricą .

Tuomet yra sukuriami testavimui ir prognozavimui naudojami vektoriai, bei atliekama prognozė:

```
Pt2=zeros(6,1);
Start = Start2;
Pt2(1,:)=P(Start:Start+1-1);
Start=Start+tau1;
Pt2(2,:)=P(Start:Start+1-1);
Start=Start+tau2;
Pt2(3,:)=P(Start:Start+1-1);
Start=Start+tau3;
Pt2(4,:)=P(Start:Start+1-1);
Start=Start+tau4;
Pt2(5,:)=P(Start:Start+1-1);
Start=Start+tau5;
Pt2(6,:)=P(Start:Start+1-1);
%Start=Start+tau6;
%Pt2(7,:)=P(Start:Start+1-1);

for i = 0:N

if i~=0
Start=Start2+i;
%tau=9;
Pt2(1,:)=P(Start:Start+1-1);
Start=Start+tau1;
Pt2(2,:)=P(Start:Start+1-1);
Start=Start+tau2;
Pt2(3,:)=P(Start:Start+1-1);
Start=Start+tau3;
Pt2(4,:)=P(Start:Start+1-1);
Start=Start+tau4;
Pt2(5,:)=P(Start:Start+1-1);
Start=Start+tau5;
Pt2(6,:)=yPred;
Pt3(i)=yPred;
end

yPred = net(Pt2);

end
```

Norint padidinti dimensijų skaičių nuo $d = 6$ iki $d = 7$ RBF procedūroje `P2` turės 7, o ne 6 stulpelius.

Tuomet yra apskaičiuojama MAPE paklaida bei nubraižomi rezultatų grafikai.

3.1 lentelė.

Programos ir jų paskirtis

Programos	Paskirtis
RBF_prognoze.m	Apskaičiuoja prognozes pasinaudojant RBF neuroniniu tinklu.
Dimensija_fnn.m	Surandą minimalią dimensiją, naudojantis klaidingo artimiausio kaimyno algoritmu.
ANFIS_prognoze.m	Apskaičiuoja prognozes pasinaudojant ANFIS tinklu.
FITNET_prognoze.m	Apskaičiuoja prognozes pasinaudojant FITNET neuroninius tinklu.
FITNET_rekonstruotas_prognoze.m	Apskaičiuoja prognozes pasinaudojant FITNET neuroninius tinklu, kai eilutė yra rekonstruojama.
pagrindine.m	Genetinis algoritmas surandantis laiko vėlinimų derinius.
Tikslo_Kokybes_reguliarus.m	Programa surandantį laiko vėlinimą esant laiko eilutės rekonstravimui su reguliariais laiko vėlinimais.

IŠVADOS

1. Nereguliarių laiko vėlinimų atrinkimo algoritmas buvo pritaikytas radialinės bazės funkcijos neuroniniam tinklui. Laiko eilučių prognozavimas šiuo neuroniniu tinklu turi daug pranašumų paklaidų atžvilgiu lyginant su kitais prognozavimo modeliais. Šio darbo tikslas buvo parodyti, kad radialinės bazės funkcijos neuroniniais tinklais pagrįstas laiko eilutės prognozavimas gali būti pagerintas laiko eilutę rekonstruojant į nereguliarią laiko vėlinimų erdvę su optimaliu laiko vėlinimo rinkiniu.
2. Naudojant radialinės bazės funkcijos neuroninius tinklus prognozavimui reikalaujama kur kas mažiau duomenų modelio apmokymui lyginat su kitais prognozavimo modeliais. Tokiu būdu galima prognozuoti ir trumpas laiko eilutės. Iš pateiktų pavyzdžių matoma, kad RBF neuroninis tinklas tiksliau prognozavo trumpas laiko eilutes: Keleivių atvykimo į San Diego metro stotį laiko eilutė net 24,1 karto tiksliau skaičiavo prognozes lyginant su svertinio slenkamojo vidurkio prognozėmis. Pieno produkcijos laiko eilutė buvo tiksliau prognozuota su RBF tinklu, nors paklaidos gaunamos nemažos: 1,5 karto lyginant su FITNET (su rekonstravimu) ir 2,7 lyginat su svertiniu slenkamuoju vidurkiu.
3. Prognozuojant laiko eilutes su RBF neuronui tinklu gali prireikti didelių laiko sąnaudų. Jei prognozuojama trumpa laiko eilutė ir ieškoma keletas reikšmių į ateitį, tai modelis skaičiuos ganėtinai greita. Jei laiko eilutė yra ilga ir daugiau jos reikšmių naudojama apmokymui, prognozavimas apims didesnį skaičiavimo laiką. Pieno produkcijos laiko eilutės prognozei prireikė tik 15,05 sekundės, o Mackey-Glass laiko eilutės prognozavimas užtruko net 82.82 sekundes.
4. Mackey – Glass laiko eilutė buvo tiksliau prognozuota su ANFIS tinklu. Tačiau realaus pasaulio eilutės tiksliau prognozuojamos su radialinės bazės funkcijos neuroniniu tinklu. Tai parodo gauti rezultatai. Tai pat radialinės bazės neuroninis tinklas kur kas tiksliau prognozuoja rekonstruotas laiko eilutes lyginant su nerekonstruotų laikų eilučių prognozėmis, naudojant kitą neuroninį tinklą.
5. Mackey – Glass laiko eilutė buvo 12,7 kartų geriau prognozuota naudojant ANFIS tinklus lyginat su RBF neuroniniu tinklu, net 22,9 kartus lyginant su FITNET neuroniniu tinklu (be rekonstravimo) ir 16,5 karto tiksliau prognozavo nei FITNET neuroniniu tinklu (su rekonstravimu). Keleivių atvykimo į San Diego metro stotį laiko eilutė 25,6 karto tiksliau skaičiavo prognozes lyginant su ANFIS tinklo prognoze, net 628,5 kartus tikslesnis lyginant su FITNET (be rekonstravimo) ir 3,3 karto prasčiau prognozavo nei FITNET (su rekonstravimu) neuroninis tinklas. Vidutinės Kauno miesto temperatūros laiko eilutės prognozės RBF neuronui

tinklu buvo tikslesnės 29 kartus lyginant su ANFIS tinklu, 14,6 kartus lyginant su FITNET neuroniniu tinklu be rekonstravimo ir 3,8 karto blogiau prognozavo lyginant su rekonstruotu FITNET tinklu.

PADĖKA

Gerb. lekt.dr.
Kristinai Lukoševičiūtei

Dėkoju magistrinio darbo vadovei lekt.dr. Kristinai Lukoševičiūtei, kuri yra puiki darbo vadovė. Vadovė intensyviai konsultuodavo ne tik susitikimo metu universitetuose, bet ir elektroniniu paštu. Nuoširdžiai dėkoju darbo vadovei už patarimus ir pagalbą įgyvendinant kai kurias baigiamojo magistrinio darbo idėjas.

Su didžiausia pagarba,
Miglė Drūlytė

LITERATŪROS SĄRAŠAS

1. Mills, T. C. *The* Econometric Modeling of Financial Time Series, *Cambridge University Press*, Cambridge, 2003, 372 p., ISBN: 0-521-624134.
2. Time Series and Forecasting [interaktyvus] [žiūrėta 2013 01 23]. Prieiga per internetą: *http://www.me.utexas.edu/~jensen/ORMM/supplements/units/time_series/time_series.pdf*
3. Chatfield Chris, *Time Series Forecasting*, 2000, USA.
4. Sitte, R.; Sitte, J. Neural Network Approach to the Random Walk Dilemma of Financial Time Series, *Applied Intelligence*, 2002, vol. 16, no. 3, p. 163-171.
5. Exponential smoothing / Wikipedia [interaktyvus] [žiūrėta 2013 01 24]. Prieiga per internetą: http://en.wikipedia.org/wiki/Exponential_smoothing.
6. Forecasting trends: Exponential smoothing [interaktyvus] [žiūrėta 2013 01 24]. Prieiga per internetą: http://www.lums.lancs.ac.uk/files/mansci/18735.pdf.
7. Double exponential smoothing [interaktyvus] [žiūrėta 2013 01 24]. Prieiga per internetą: http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc433.htm.
8. E. M. Azoff, Neural Network Time Series: Forecasting of Financial Markets (John Wiley & Sons, Chichester, 1994).
9. Gardner E. S., Exponential smoothing: The state of the art – Part II, [interaktyvus] [žiūrėta 2013 01 24]. Prieiga per internetą: http://www.bauer.uh.edu/gardner/docs/pdf/Exponential-Smoothing.pdf.
10. Bourke P., Autoregression analysis, [interaktyvus] [žiūrėta 2013 01 24]. Prieiga per internetą: http://paulbourke.net/miscellaneous/ar/.
11. Autoregressive model / Wikipedia [interaktyvus] [žiūrėta 2013 01 24]. Prieiga per internetą: http://en.wikipedia.org/wiki/Autoregressive_model.
12. Autoregressive models [interaktyvus] [žiūrėta 2013 01 24]. Prieiga per internetą: http://www.vosesoftware.com/ModelRiskHelp/index.htm#Time_series/Autoregressive_models.htm.
13. Moving-average model / Wikipedia [interaktyvus] [žiūrėta 2013 01 25]. Prieiga per internetą: http://en.wikipedia.org/wiki/Autoregressive%E2%80%93moving-average_model
14. Autoregressive integrated moving average [interaktyvus] [žiūrėta 2013 01 25]. Prieiga per internetą: http://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average.
15. Hag H. M. A. El, Sharif S. M., An Adjusted ARIMA Model for Internet Traffic, Africon, 2007.
16. G. E. Box and M. G. Jenkins, Time series analysis forecasting and control, 2nd ed. San Francisco: Holden-Day, 1976.

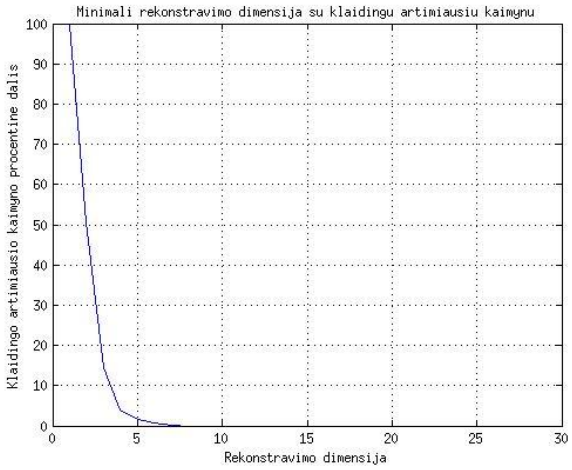
17. J. Johnston and J. DiNardo, Econometric methods, 4th ed. Singapore: McGraw-Hill, 1997.
18. Neural networks: A requirement for intelligent systems, [interaktyvus] [žiūrėta 2013 01 29]. Prieiga per internetą: http://www.learnartificialneuralnetworks.com/.
19. Zhang G., Patuwo B. E., Hu M. Y., *Forecasting With Artificial Neural Networks: The State of the Art*, International Journal of Forecasting, 1998, vol. 14, p. 35-62.
20. Gershenson C., Artificial Neural Networks for Beginners, [interaktyvus] [žiūrėta 2013 01 29]. Prieiga per internetą: http://arxiv.org/ftp/cs/papers/0308/0308031.pdf.
21. Stergiou C., Siganos D., Neural networks, [interaktyvus] [žiūrėta 2013 01 30]. Prieiga per internetą: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html.
22. Edwards, T., Tansley, D, S. W., Davey, N., Frank, R. J. (1997). Traffic Trends Analysis using Neural Networks. *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications 3*. pp. 157-164.
23. Patterson D W, Chan K H, Tan C M. 1993, Time Series Forecasting with neural nets: a comparative study. *Proc. the international conference on neural network applications to signal processing*. NNASP 1993 Singapore pp 269-274.
24. Bengio, S., Fessant F., Collobert D. A Connectionist System for Medium-Term Horizon Time Series Prediction. *In Proc. Intl. Workshop Application Neural Networks to Telecoms* pp308-315, 1995.
25. Gershenfeld N. A. and A. S. Weigend, The Future of Time Series. In Time Series Prediction: Forecasting the Future and Understanding the Past, Gershenfeld A. N. and A. S. Weigen, eds, pp 1-70. 1993.
26. Khashei, M.; Bijari, M.; Ardali, G. A. R. Improvement of auto-regressive integrated moving average models using fuzzy logic and artificial neural network (ANNs), *Neurocomputing*, 2009, vol. 72, p. 956-967.
27. Lee, Ch. M.; Ko, Ch. N. Short-term load forecasting using lifting scheme and ARIMA models, *Expert Systems with Applications*, 2011, vol. 38, p. 5902-5911.
28. Tao D., Hongfei X., Chaotic Time Series Prediction Based on Radial Basis Function Network, *Eighth ACIS International Conference*, 2007.
29. Packard N. H., Crutch J. eld, Farmer J., Shaw R., Geometry from a time series, *Phy. Rev. Lett*, 1980, vol. 45, p. 712-716.
30. Takens F., Detecting strange attractors in turbulence, *Lecture Notes in Mathematic*, 1980, 898, p. 366-381
31. Kim H. S., Eykholt R., Salas J. D., Nonlinear dynamics, delay times, and embedding windows. *Physica D*, 1999, 127, p. 48-60.

32. Yao X., Zhang X., Zhang R., Liu M., Hu Z., Fan B., Prediction of Gas Chromatographic Retention Indices by the Use of Radial Basis Function Neural Networks, *Talanta*, 2002, vol. 52, p. 297-306/
33. Xia C., Wang J., McMenemy K., Short, Medium and Long Term Load Forecasting Model and Virtual Load Forecaster Based on Radial Basis Function Neural Network, *Electrical Power and Energy Systems*, 2010, vol. 32, p. 743-750.
34. Harpham C., Dawson C. W., The Effect of Different Basis Functions on a Radial Basis Function Network for Time Series Prediction: A Comparative Study, *Neurocomputing*, 2006, vol. 69, p. 2161-2170.
35. Al-Haddad L., Morris C. W., Boddy L., Training radial basis function neural networks: effects of training set size and imbalanced training sets, *Journal of Microbiological Methods*, 2000, vol. 43, p. 33-44.
36. Ma N., Lu C., Zhang W. J., Wu X. H., Application of Parallel RBF Network on Iterative Prediction of Chaotic Time Series, International Workshop on Chaos-Fractal Theory and its Applications, 2010.
37. Broomhead D.S., Lowe D., Multivariable Functional Interpolation and Adaptive Networks, *Complex Systems*, 1988, 2:321–355.
38. Michelli C.A., Interpolation of Scattered Data: Distance Matrices and Conditionally Positive Definite Functions, *Constructive Approximation*, 1986., 2:11–22
39. M.J.D. Powell. Radial basis functions for multivariable interpolation: a review. In Proceedings of the IMA Conference on Algorithms for the Approximation of Functions and Data, Shrivenham, 1985. RMCS.
40. M.J.D. Powell. Radial basis functions for multivariable interpolation: a review. In Algorithms for Approximation, 1987, p. 143–167. Clarendon Press, Oxford.
41. Hu J. H., Hwang J. N., Handbook of NEURAL NETWORK SIGNAL PROCESSING, USA, 2002.
42. Ripley B.D., Pattern Recognition and Neural Networks. *Cambridge University Press*, Cambridge, MA, 1995.
43. Brandimarte P., Zotteri G., Introduction to distribution logistics. John Wiley & Sons. Hoboken, New Jersey, 2007, p. 587. ISBN 978-0-471-75044-4.
44. Ragulskis M., Lukoševičiūtė K., Non-uniform attractor embedding for time series forecasting by fuzzy inference systems, *Neurocomputing*, 2009, vol. 72, p. 2618-2626.
45. Lukoševičiūtė K., Ragulskis M., Evolutionary algorithm for the selection of time lags for time series forecasting by fuzzy inference systems, *Neurocomputing*, 2010, vol. 73, p. 2077-2088.

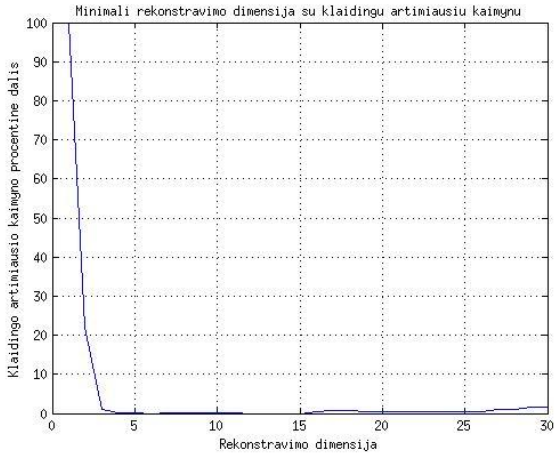
46. Kennel M. B., Brown R., Abarbanel H. D. I., Determining embedding dimension for phase space reconstruction using a geometrical construction, *Physical review A*, 1992, vol. 46, p. 3403-3411.
47. Rhodes C., Morari M., False-nearest-neighbors algorithm and noise-corrupted time series, *Physical review E*, 1997, vol. 55, nr. 5, p. 6162-6170.
48. Du H., Zhang N., Time Series Prediction Using Evolving Radial Basis Function Networks With New Encoding Scheme, *Neurocomputing*, 2008, vol. 71, p. 1388-1400.
49. Harpham C., Dawson C.W., The Effect of Different Basis Functions on a Radial Basis Function Network For Time Series Prediction: A Comparative Study, *Neurocomputing*, 2006, vol. 69, p. 2161-2170.
50. Leung H., Lo T., Wang S., Prediction of Noisy Chaotic Time Series Using an Optimal Radial Basis Function Neural Network, *IEEE Transactions on Neural Network*, 2001, vol. 12, No. 5, p. 1163-1172.
51. Li C., Ye H., Wang G., Nonlinear Time Series Modeling and Prediction Using RBF Network with Improved Clustering Algorithm, *IEEE International Conference on Systems*, Man and Cybernetics, 2004, p. 3513-3518.
52. Uzal L.C., Grinblat G.L., Verder P. F., Optimal reconstruction of dynamical systems: A noise amplification approach, *Physical Review E 84*, 2011.
53. Bishop C.M., Roach C.M., Fast curve fitting using neural network, 1992.
54. Forecasting by smoothing techniques, [interaktyvus] [žiūrėta 2013 05 02]. Prieiga per internetą: http://home.ubalt.edu/ntsbarsh/Business-stat/otherapplets/ForecaSmo.htm.
55. Shin G.W., Song Y.J., Lee T.B., Choi H.K, Genetic Algorithm for Identification of Time Delay Systems from Step Responses, *International Journal of Control, Automation and Systems*, 2007, vol. 5, no. 1, p. 79-85.
56. Mayra O., Ahola T., Leiviska K., Time delay estimation and variable grouping using genetic algorithm, *Control Engineering Laboratory* ,2006, Report A no. 32.
57. Atsalakis G.S., Dimitrakakis E.M., Zopounidis C.D., Elliott wave theory and neuro-fuzzy systems with applications, 2011, vol. 38, p. 9196-9206.

1 PRIEDAS. DIMENSIJŲ GRAFIKAI

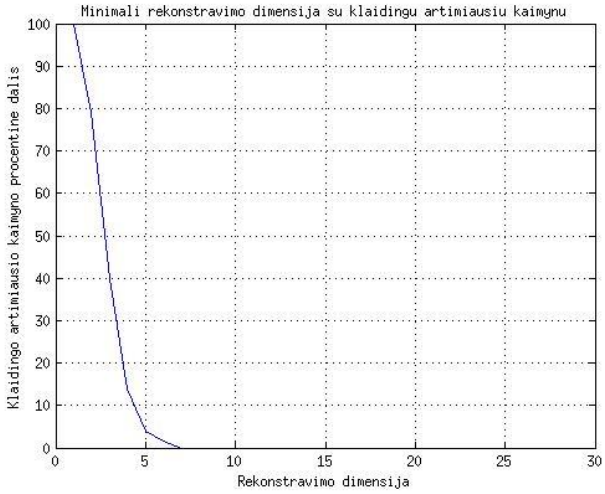
Keleivių atvykimo į San Diego metro stotį laiko eilutės dimensijos grafikas:



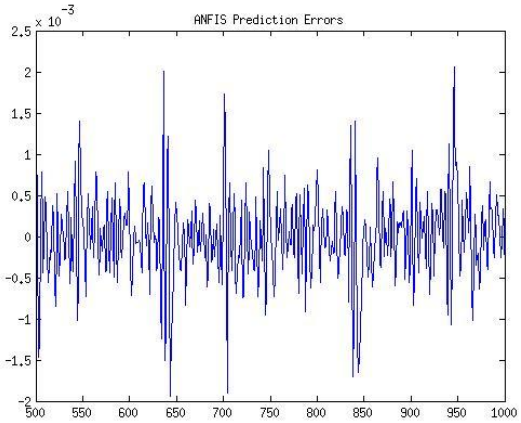
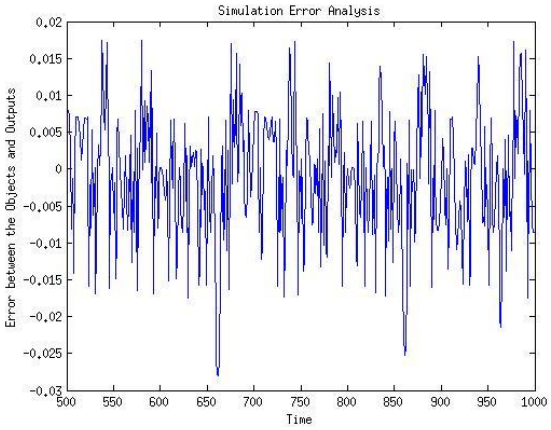
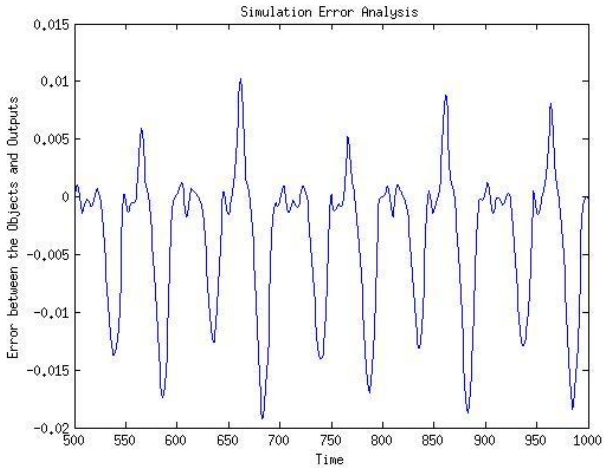
Vidutinės Kauno miesto temperatūros laiko eilutės dimensijos grafikas:

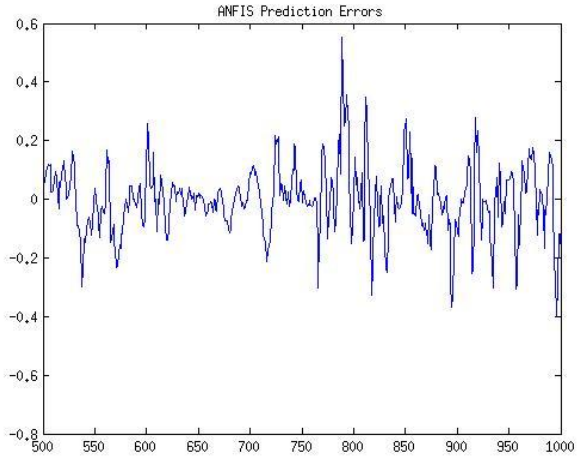
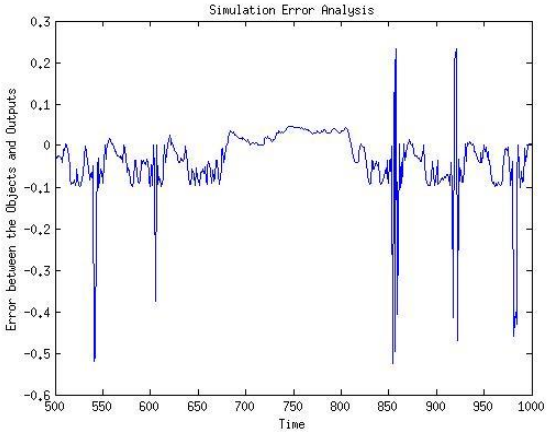


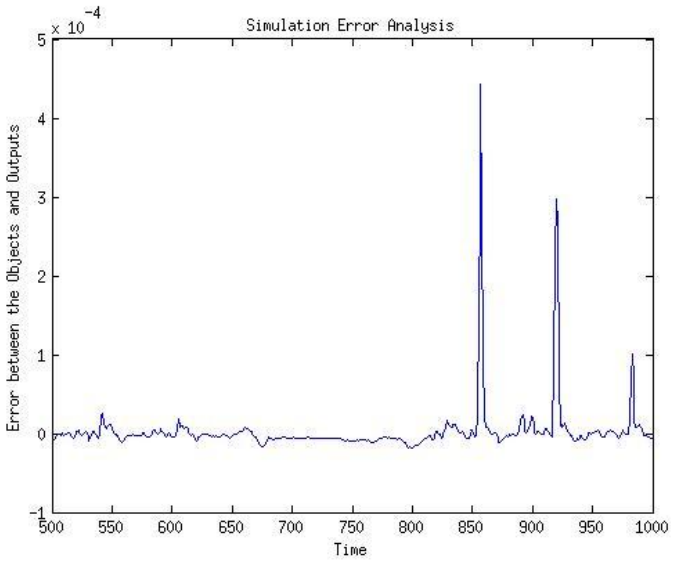
Pieno produkcijos laiko eilutės dimensijos grafikas:

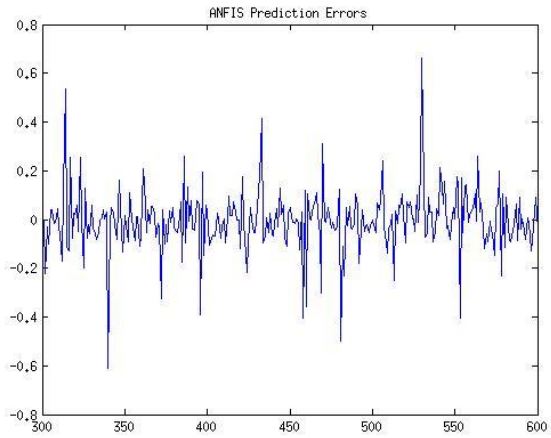
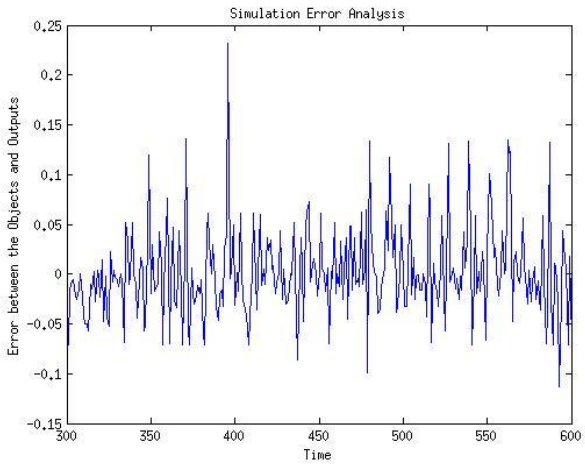
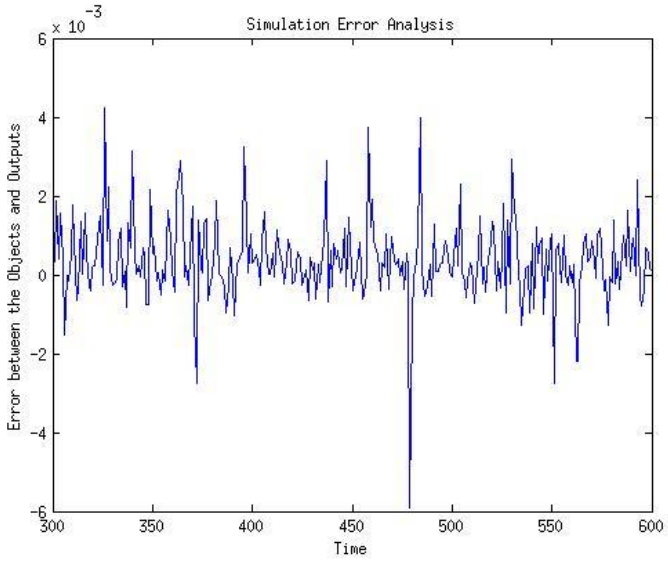


2 PRIEDAS. SKIRTUMINĖS PAKLAIDOS

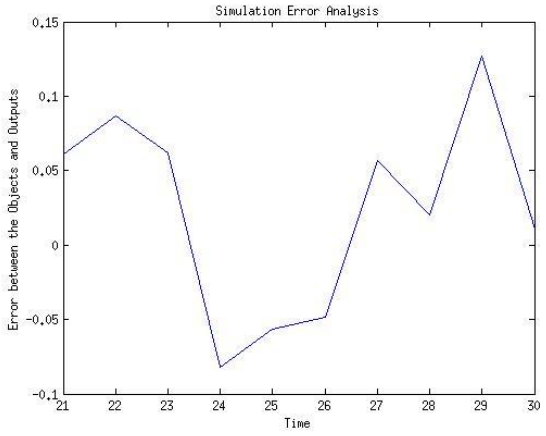
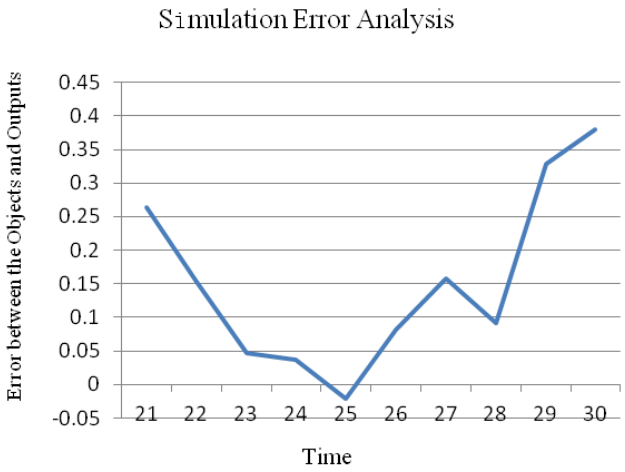
Mackey – Glass laiko eilutė	
Anfis prognozės skirtuminių paklaidų grafikas	Fitnet prognozės skirtuminių paklaidų grafikas
	
Fitnet rekonstruotos laiko eilutės prognozės skirtuminių paklaidų grafikas	
	

Keleivių atvykimo į San Diego metro stotį laiko eilutė	
Anfis prognozės skirtuminių paklaidų grafikas	Fitnet prognozės skirtuminių paklaidų grafikas
	
Fitnet rekonstruotos laiko eilutės prognozės skirtuminių paklaidų grafikas	



Vidutinės Kauno miesto temperatūros laiko eilutė	
Anfis prognozės skirtuminių paklaidų grafikas	Fitnet prognozės skirtuminių paklaidų grafikas
	
Fitnet rekonstruotos laiko eilutės prognozės skirtuminių paklaidų grafikas	
	

Pieno produkcijos laiko eilutė

Fitnet rekonstruotos laiko eilutės prognozės skirtuminių paklaidų grafikas	Svertinio slenkančiojo vidurkio prognozės skirtuminių paklaidų grafikas
	

3 PRIEDAS. RBF NEURONINIO TINKLO PROGNOZĖS MATLAB REALIZACIJA

```
clc;
workspace;
clear all;
tic
load mgdata.dat
a = mgdata;
duom = a(:,2);
time = a(:,1);
%length(a)
P = duom';

%Duomenu normavimas
%P=(duom-min(duom))/(max(duom)-min(duom));
%P=P';

Start1 = 101;
tau1 = 7;
tau2 = 4;
tau3 = 5;
tau4 = 4;
tau5 = 7;
%tau6 = 7;
N = 500;
Start2 = 601;

index=101:1200;
figure(4);
plot(1:1100,duom(index));
xlabel('Laikas');
ylabel('x(t)');
title('Mackey - Glass laiko eilute')


%apmokymo duomenys
P1=zeros(6,N);   %ivestis P = RxQ matrica
Start=Start1;
P1(1,:)=P(Start:Start+N-1);
Start=Start+tau1;
P1(2,:)=P(Start:Start+N-1);
Start=Start+tau2;
P1(3,:)=P(Start:Start+N-1);
Start=Start+tau3;
P1(4,:)=P(Start:Start+N-1);
Start=Start+tau4;
P1(5,:)=P(Start:Start+N-1);
Start=Start+tau5;
P1(6,:)=P(Start:Start+N-1);
%Start=Start+tau6;
%P1(7,:)=P(Start:Start+500-1);

T = P(101:600); %Norimas vektorius

mn = 500;     % max neuronu skaicius mn = Ntrn  (No. of training vectors)
df = 1;      % Number of neurons to add between displays
mean(var(T));
size(P1);
minmax(P1);
size(T);
minmax(T);
y = mean(T);
y00 = repmat(mean(T,2),1,size(T,2));
e00 = T-y00;
SSE00 = sse(e00);
```

```
SSEgoal = SSE00/100;
spread = 0.5*mean(median(dist(P1,P1')))    % issibarstymo konstanta
MSE0 = (N-1)*mean(var(T'))/N;
MSEi = (N-1)*var(T)/N;

size(T)
[ O Ntrain ] = size(T);
Ytrain00 = repmat(mean(T,2),1,Ntrain);
MSEtrain00 = mse(T-Ytrain00);
MSEgoal = MSEtrain00/2000 % R2train >= 0.995;    % sum-squared error goal
MSE00 = mean(var(T'));
net = newrb(P1,T,MSEgoal,spread,mn,df);
%view(net)
ytrain = net(P1);

%Laiko eilutes prognozavimas
Pt2=zeros(6,1);
Start = Start2;
Pt2(1,:)=P(Start:Start+1-1);
Start=Start+tau1;
Pt2(2,:)=P(Start:Start+1-1);
Start=Start+tau2;
Pt2(3,:)=P(Start:Start+1-1);
Start=Start+tau3;
Pt2(4,:)=P(Start:Start+1-1);
Start=Start+tau4;
Pt2(5,:)=P(Start:Start+1-1);
Start=Start+tau5;
Pt2(6,:)=P(Start:Start+1-1);
%Start=Start+tau6;
%Pt2(7,:)=P(Start:Start+1-1);


for i = 0:N

if i~=0
Start=Start2+i;
%tau=9;
Pt2(1,:)=P(Start:Start+1-1);
Start=Start+tau1;
Pt2(2,:)=P(Start:Start+1-1);
Start=Start+tau2;
Pt2(3,:)=P(Start:Start+1-1);
Start=Start+tau3;
Pt2(4,:)=P(Start:Start+1-1);
Start=Start+tau4;
Pt2(5,:)=P(Start:Start+1-1);
Start=Start+tau5;
Pt2(6,:)=yPred;
Pt3(i)=yPred;

end

yPred = net(Pt2);

end

Tt=P(601:1100);
Yt = Pt3;

%output=Yt*(max(duom)-min(duom))+min(duom);
%MSE=mse(e);

%R2 = 1-MSE/MSE00   %>0.95

mape_res = (abs(Tt-Yt))./Tt;
mape = (sum(mape_res))*(1/N)*100;
mape
%end
```

```
train=duom(101:600)';

e=Tt-Yt;


ytrainn=ytrain*(max(duom)-min(duom))+min(duom);

index=601:1100;
figure(1);
subplot(2,1,1)
plot(Tt);
xlabel('time(index)'); ylabel('x(t)');
subplot(2,1,2)
plot(Yt);
xlabel('time(index)'); ylabel('radial basis');

figure(2);
plot(501:1000,e);
xlabel('Time');
ylabel('Error between the Objects and Outputs');
title('Simulation Error Analysis')

figure(3);
plot(1:500,T,'b-',501:1000,Tt,'r-')
grid on
hold on
plot(1:500,ytrain,'g+',501:1000,Yt,'k+');
%legend('Time','x(t)')
hold off

index=101:1101;
figure(5);
subplot(2,1,1)
%plot(index, Tt(index+100));
plot(1:500,T,501:1000,Tt);
xlabel('time(index)'); ylabel('x(t)');
subplot(2,1,2)
plot(1:500,ytrain,'b-',501:1000,Yt,'r-');
xlabel('time(index)'); ylabel('radial basis');

% Isvedami svoriai ir poslinkiai
W1=net.IW{1,1}
b1=net.b{1,1}
W2=net.LW{2,1}
b2=net.b{2,1}

toc
```

4 PRIEDAS. KLAIDINGO ARTIMIAUSIO KAIMYNO ALGORITMO MATLAB REALIZACIJA

```
function [FNN] = knn_denemem(duom,delay,maxd,Rtol,Atol)
clc;
clear all;
workspace;

load mgdata.txt
a = mgdata;
duom = a(:,2);
time = a(:,1);

maxd = 30;
delay = 9;
Rtol = 10;
Atol=2;

N=length(duom);
Ra=std(duom,1);

for m=1:maxd
    M=N-m*delay;
    Y=psr_deneme(duom,m,delay,M);
    FNN(m,1)=0;
    for n=1:M
        y0=ones(M,1)*Y(n,:);
        atstumas=sqrt(sum((Y-y0).^2,2));
        [neardis nearpos]=sort(atstumas);

        D=abs(duom(n+m*delay)-duom(nearpos(2)+m*delay));
        R=sqrt(D.^2+neardis(2).^2);
        if D/neardis(2) > Rtol || R/Ra > Atol
             FNN(m,1)=FNN(m,1)+1;
        end
    end
end
FNN=(FNN./FNN(1,1))*100;
figure
plot(1:length(FNN),FNN)
grid on;
title('Minimali rekonstravimo dimensija su klaidingu artimiausiu kaimynu')
xlabel('Rekonstravimo dimensija')
ylabel('Klaidingo artimiausio kaimyno procentine dalis')

function Y=psr_deneme(duom,m,tao,npoint)
% Fazines erdves rekonstravimas
N=length(duom);
if nargin == 4
    M=npoint;
else
    M=N-(m-1)*tao;
end
Y=zeros(M,m);
for i=1:m    Y(:,i)=duom((1:M)+(i-1)*tao)';
end
```

5 PRIEDAS. ANFIS TINKLO PROGNOZĖS MATLAB REALIZACIJA

```
clc;
clear all
workspace;
tic
%load duomenys
load mgdata.txt
a=mgdata;
time=a(:,1);
duom=a(:,2);
length(a)

%P=(duom-min(duom))/(max(duom)-min(duom));
%duom=P;

P1=zeros(500,6);
Pt2=zeros(500,6);
start=101;
P1(:,1)=duom(start:start+500-1);
start=start+7;
P1(:,2)=duom(start:start+500-1);
start=start+4;
P1(:,3)=duom(start:start+500-1);
start=start+5;
P1(:,4)=duom(start:start+500-1);
start=start+4;
P1(:,5)=duom(start:start+500-1);
start=start+7;
P1(:,6)=duom(start:start+500-1);
%start=start+18;
%P1(:,7)=duom(start:start+500-1);
%start=start+20;
%P1(:,8)=duom(start:start+500-1);

start=601;
Pt2(:,1)=duom(start:start+500-1);
start=start+7;
Pt2(:,2)=duom(start:start+500-1);
start=start+4;
Pt2(:,3)=duom(start:start+500-1);
start=start+5;
Pt2(:,4)=duom(start:start+500-1);
start=start+4;
Pt2(:,5)=duom(start:start+500-1);
start=start+7;
Pt2(:,6)=duom(start:start+500-1);
%start=start+18;
%Pt2(:,7)=duom(start:start+500-1);
%start=start+20;
%Pt2(:,8)=duom(start:start+500-1);

index=121:1120;
figure(1);
plot(time(index), duom(index));
xlabel('Time(sec)'); ylabel('x(t)');

fismat=genfis1(P1);
figure(2);
for input_index=1:5
    subplot(4,2,input_index)
    [x,y]=plotmf(fismat,'input',input_index);
    plot(x,y)
    axis([-inf inf 0 1.2]);
    xlabel(['input ', int2str(input_index)]);
end
```

```
[trn_fismat, trn_error, stepsize, chk_fismat,chk_error]=anfis(P1, fismat, [], [], Pt2);
figure(3);
for input_index=1:5
    subplot(4,2,input_index)
    [x,y]=plotmf(trn_fismat,'input',input_index);
    plot(x,y)
    axis([-inf inf 0 1.2]);
    xlabel(['input ', int2str(input_index)]);
end

figure(4);
epoch_n=10;
tmp=[trn_error chk_error];
subplot(1,1,1)
plot(tmp);
hold on; plot(tmp, '+'); hold off;
xlabel('epochs'); ylabel('RMSE(Root Mean Squared Error)');

input=[P1(:, 1:5); Pt2(:, 1:5)];
anfis_output=evalfis(input, trn_fismat);
k=anfis_output(501:1000);
Train=duom(121:620);

index=128:1127;
figure(5);
subplot(2,1,1)
plot (501:1000, Train); xlabel('time(index)'); ylabel('x(t)');
%axis([501 1000 0 1])
subplot(2,1,2)
plot(501:1000, anfis_output(501:1000)); xlabel('time(index)'); ylabel('anfis_output');
%axis([501 1000 0 1])

index=628:1127;
duom2=duom;
diff=duom2(index)-anfis_output(501:1000);
figure(6);
plot(diff);
title('ANFIS Prediction Errors');
diff;
l=duom2(index);
length(diff);

length(duom2(index))
mod = abs(diff);
mape_res = mod./l;
N = 500;
mape = (sum(mape_res))*(1/N)*100;
mape

toc
```

6 PRIEDAS. FITNET NEURONINIO TINKLO PROGNOZĖS MATLAB REALIZACIJA

```
clc;
clear all;
workspace;
tic
load mgdata.txt
a = mgdata;
duom = a(:,2);
time = a(:,1);

P = duom;
%Normalized by the folowing function
%P=(duom-min(duom))/(max(duom)-min(duom));

x = P(101:600)';
t = P(601:1100)';

net = fitnet(15);
% Train the Network
net = train(net, x, t); % Iskaidoma i train val test viduje!

test = P(51);

for i = 0:40

if i~=0
test = P(51+i);
output(i)=yPred;
end
yPred = net(test);
end

% Test
Yt = output;
diff = Yt-t;

figure(1)
subplot(2,1,1)
plot(t);
xlabel('time(index)'); ylabel('x(t)');
subplot(2,1,2)
plot(Yt);
xlabel('time(index)'); ylabel('fitnet');
%mse = mse(gsubtract(testTargets, outputs));

figure(2);
plot(diff);
xlabel('Time');
ylabel('Error between the Objects and Outputs');
title('Simulation Error Analysis')

N = 40;
mape_res = (abs(t-Yt))./t;
mape = (sum(mape_res))*(1/N)*100;
mape
%end

toc
```

7 PRIEDAS. FITNET NEURONINIO TINKLO PROGNOZĖS MATLAB REALIZACIJA (REKONSTRUOTOS EILUTĖS)

```
tic
clc;
workspace;
clear all;
load mgdata.txt
a = mgdata;
duom = a(:,2);
time = a(:,1);
length(a)
index=101:1100;
figure(4);
plot(1:1000,duom(index));
xlabel(Laikas);
ylabel('x(t)');
title('Mackey-Glass eilute')
P = duom;
%Normalized by the folowing function
%P=(duom-min(duom))/(max(duom)-min(duom));
P=P';
%training set
P1=zeros(6,500);
Start=11;
%tau=28;
P1(1,:)=P(Start:Start+500-1);
Start=Start+7;
P1(2,:)=P(Start:Start+500-1);
Start=Start+4;
P1(3,:)=P(Start:Start+500-1);
Start=Start+5;
P1(4,:)=P(Start:Start+500-1);
Start=Start+4;
P1(5,:)=P(Start:Start+500-1);
Start=Start+7;
P1(6,:)=P(Start:Start+500-1);
%Start=Start+11;
%P1(7,:)=P(Start:Start+500-1);
%Start=Start+20;
%P1(8,:)=P(Start:Start+500-1);
T = P(101:600);
net = fitnet(15);
%view(net)
net = train(net,P1,T);
ytrain = net(P1);
Pt2=zeros(6,1);
Start=601;
Pt2(1,:)=P(Start:Start+1-1);
Start=Start+7;
Pt2(2,:)=P(Start:Start+1-1);
Start=Start+4;
Pt2(3,:)=P(Start:Start+1-1);
Start=Start+5;
Pt2(4,:)=P(Start:Start+1-1);
Start=Start+4;
Pt2(5,:)=P(Start:Start+1-1);
Start=Start+7;
Pt2(6,:)=P(Start:Start+1-1);
%Start=Start+11;
%Pt2(7,:)=P(Start:Start+1-1);
%Start=Start+20;
%Pt2(8,:)=P(Start:Start+1-1);
%Pt2 = Pt2';

for i = 0:500
 if i~=0
```

```
  Start=601+i;
  Pt2(1,:)=P(Start:Start+1-1);
  Start=Start+7;
  Pt2(2,:)=P(Start:Start+1-1);
  Start=Start+4;
  Pt2(3,:)=P(Start:Start+1-1);
  Start=Start+5;
  Pt2(4,:)=P(Start:Start+1-1);
  Start=Start+4;
  Pt2(5,:)=P(Start:Start+1-1);
  %Start=Start+7;
  %Pt2(6,:)=P(Start:Start+1-1);
  Start=Start+21;
  Pt2(6,:)=yPred;
  Pt3(i)=yPred;
 end
 yPred = net(Pt2);
end
Tt=P(311:610) ;
Yt = Pt3;
e=Tt-Yt;
N = 500;
mape_res = (abs(e))./Tt;
mape = (sum(mape_res))*(1/N)*100;
mape
%end
index=601:1100;
figure(1);
subplot(2,1,1)
plot(501:1000,Tt);
xlabel('time(index)'); ylabel('x(t)');
subplot(2,1,2)
plot(501:1000,Yt);
xlabel('time(index)'); ylabel('fitnet network');

figure(2);
plot(501:1000,e);
xlabel('Time');
ylabel('Error between the Objects and Outputs');
title('Simulation Error Analysis')

figure(3);
plot(1:500,T,'k-',501:1000,Tt,'k-')
grid on
hold on
plot(1:500,ytrain,'b+',501:1000,Yt,'r+');
legend('Time','x(t)')
hold off
index=101:1100;
figure(5);
subplot(2,1,1)
%plot(index, Tt(index+100));
plot(1:500,T,501:1000,Tt);
axis([0 1000 0 1])
xlabel('time(index)'); ylabel('x(t)');
subplot(2,1,2)
plot(1:500,ytrain,'b-',501:1000,Yt,'r-');
axis([0 1000 0 1])
xlabel('time(index)'); ylabel('fitnet network');
% Display weights and biases
W1=net.IW{1,1} ;
b1=net.b{1,1} ;
W2=net.LW{2,1} ;
b2=net.b{2,1} ;
toc
```

8 PRIEDAS. LAIKO VĖLINIMŲ RADIMO MATLAB REALIZACIJA

```
disp('maksimali reiksme maxval:')
maxval=30
disp('kintamuju kiekis')
bits=5
disp('populiacijos dydis')
pop=24
disp('generaciju skaicius')
gens=20
disp('kryzminimo laipsnis')
matenum=0.8
disp('mutacijos intensyvumas')
mu=0.002
disp('dominavimo koeficientas')
dom=2
fkritmax=0;
kritinis=1.1026;
tau=[];
N=[];
K(1:100)=0;
F=[];
Tau=[];
FULLSORF=[];
for gens=1:100
for i=1:100
 [x,f]=optga('spygliaiNU',maxval,bits,pop,gens,mu,matenum,dom);
 if f>fkritmax
     fkritmax=f;
     tau=x;
 end;
 if f>=kritinis
     K(gens)=K(gens)+1;
 end;
N(i)=i;
F(i)=f;
Tau(i,:)=x;
end;
end;
FF=F';
 fkritmax
 tau
```

spygliaiNU.m

```
function K=spygliaiNU(tau)
global FTRAP F w lew;
n=length(tau);
delta=0.05;
d=n+1;
K=0;
Q=0;
Qsin=0;
sumtau=0;
for j = 1:d-1
    for i=1:d-j
        for k=1:j
            sumtau=sumtau+tau(i+k-1);
        end;
        Qsin = Qsin + abs(sin(delta*sumtau*w));
        sumtau=0;
    end;
   Q=Q+Qsin;
   Qsin=0;
end;
    Q = Q*2/(d*(d-1));
```

```
    FQ = F.*Q;
    K = (pi*trapz(FQ))/(FTRAP*2);
```

optga.m

```
function[xval,maxf]=optga(fun,maxval,bits,pop,gens,mu,matenum,dom)
%optimizavimas paremtas genetiniais algoritmais
%fun - tikslo funkcija
%bits - bitu skaicius
%pop - chromosomu skaicius populiacijoje
%gens-generaciju (kartu) skaicius
%mu- mutaciju dydis
init;
newpop=[];
newpop=genbin(bits,pop,maxval);
for i=1:gens
    selpop=selectga(fun,newpop);
    N=size(selpop);
    newgen=matesome(selpop,matenum,dom);
    M=size(newgen);
    newgen1=mutate(newgen,mu,maxval);
    newpop=newgen1;
end
[fit,fitot]=fitnes(fun,newpop);
[maxf,mostfit]=max(fit);
xval=newpop(mostfit,:);
maxf=feval(fun,newpop(mostfit,:));
```

init.m

```
global FTRAP F w lew;
delta=0.05;
load mgdata.txt
a=mgdata;
time=a(:,1);
duom=a(:,2);
[n,m]=size(a);
for i=101:1100
   duomen(i-100)=duom(i);
end
y = fft(duomen,500);
ley = length(y);
lew = round(ley/2);
pyy = 2*abs(y)/ley;
F = pyy(1:lew);
w = 2*pi/(delta*ley) * (0:(lew-1));
F(1)=0;
FTRAP = trapz(F);
```

selectga.m

```
function newchrom=selectga(kriter,chrom)
%parenka geriausias chromosomas sekanciai kartai pagal tikslo funkcija
%-'kriter'
[pop bitlength]=size(chrom);
fit=[];
%skaiciuojam chromosomu geruma
[fit fitot]=fitnes(kriter,chrom);
for chromnum=1:pop
    sval(chromnum)=sum(fit(1,1:chromnum));
end
parname=[];
for i=1:pop
    rval=floor(fitot*rand);

    if rval<sval(1)
        parname=[parname 1];
    else
```

```
        for j=1:pop-1
            sl=sval(j);
            su=sval(j)+fit(j+1);
            if (rval>=sl)&(rval<=su)
                parname=[parname j+1];
            end
        end
    end
end
newchrom(1:pop,:)=chrom(parname,:);
```

matesome.m

```
function chrom1=matesome(chrom,kiek,dom)
% genu kryzminimo procedura
% chrom - pradinis genu rinkinys
% chrom1 - genai po sukryzminimo
% kiek - kryzminimo laipsnis nuo 0 iki 1
% dom - vieno geno dominavimo dydis, dom>0
mateind=[];
chrom1=chrom;
[pop bitlength]=size(chrom);
ind=1:pop;
u=floor(pop*kiek);
if floor(u/2)~=u/2;
    u=u-1;
end
%atsitiktinis kryzminimas -sukonstruoja matrica mateind
while length(mateind)~=u
    i=round(rand*pop);
    if i==0
        i=1;
    end
    if ind(i)~=-1
        mateind=[mateind i];
        i=-1;
    end
end
%informacijos kryzminimas atsitiktiniame taske
for i=1:2:u-1
   splitpos=floor(rand*(bitlength-1));
   if splitpos==0
       splitpos=1;
   end
   splitpos=splitpos+1;
   i1=mateind(i);
   i2=mateind(i+1);
   domgene1=dom*chrom(i1,:);
   domgene2=dom*chrom(i2,:);
   chrom1(i1,1:splitpos-1)=domgene1(1:splitpos-1)+chrom(i2,1:splitpos-1);
   chrom1(i1,splitpos:bitlength)=domgene2(splitpos:bitlength)+chrom(i1,splitpos:bitlength);
   chrom1(i2,1:splitpos-1)=domgene2(1:splitpos-1)+chrom(i1,1:splitpos-1);
   chrom1(i2,splitpos:bitlength)=domgene1(splitpos:bitlength)+chrom(i2,splitpos:bitlength);
   chrom1(i1,:)=round(chrom1(i1,:)/(dom+1));
   chrom1(i2,:)=round(chrom1(i2,:)/(dom+1));
end
```

mutate.m

```
function chrom=mutate(chrom,mu,maxval)
%chromosomu mutacija
%mu - mutacijos stiprumo parametras, paprastai nedidelis teig. sk.
[pop bitlength]=size(chrom);
for i=1:pop
    for j=1:bitlength
    if rand<=mu
        chrom(i,j)=chrom(i,j)+round(rand*(maxval-1));
        if chrom(i,j)>maxval
            chrom(i,j)=chrom(i,j)-maxval;
```

```
        end
    end
    end
end
```

genbin.m

```
function chromosome=genbin(bitl,numchrom,maxval)
%generuojama desimtaine(dvejetaine) populiacija
%populiacijos dydis numchrom
maxchros=2^bitl;
if numchrom>=maxchros
    numchrom=maxchros;
end
chromosome=round(maxval*rand(numchrom,bitl));
```

fitnes.m

```
function [fit,fitot]=fitnes(kriter,chrom)
%Fitness (gerumas) chromosomu rinkiniui chrom
%'kriter'- isorines tikslo funkcijos pavadinimas
[pop bitl]=size(chrom);
for k=1:pop
    fit(k)=feval(kriter,chrom(k,:));
end
fitot=sum(fit);
```

9 PRIEDAS. REGULIARAUS REKONSTRAVIMO MATLAB REALIZACIJA

```
clear all;
close all;
clc
d = 6;  %dimensija
delta = 0.05;
load mgdata.txt  % laiko eilute
a=mgdata;
time=a(:,1); duom=a(:,2);
%duom=(duom-min(duom))/(max(duom)-min(duom)); %Normuojami duomenys
for i=101:1100
   duomenys(i-100)=duom(i);
end
y = fft(duomenys,500);  %Furje transformacija
ley = length(y);
lew = round(ley/2);
pyy = 2*abs(y)/ley;
F = pyy(1:lew);
w = 2*pi/(delta*ley) * (0:(lew-1));
F(1)=0;
FTRAP = trapz(F);
T(1) = 0; TAU(1) = 0;
taumax=0; Kkritmax=0; Kkr=0;
for tau = 0:30
Q = zeros(1,lew);
    for k = 1:(d-1)
        Q = Q + (d-k)*abs(sin(k*tau*delta*w));
    end
    Q = Q*2/(d*(d-1));
    FQ = F.*Q;
    T(tau+1) = (pi*trapz(FQ))/(FTRAP*2);
    TAU(tau+1) = tau;
    if T(tau+1) > Kkritmax
        Kkritmax=T(tau+1);
        taumax=tau;

    end
end
 mmQ(tau)=max(Q);

figure(1)
hold on
plot(w,F,'k','LineWidth',2);
plot(w,Q,'k','LineWidth',1);
xlabel('w'); ylabel('A(w), Q(w)');

figure(2);
hold on
subplot(2,1,1)
plot(w,F,'k','LineWidth',2);  %Fourier amplitude
xlabel('w'); ylabel('A(w)');
subplot(2,1,2)
plot(w,Q,'k','LineWidth',1);  %Kokybes funkcija
xlabel('w'); ylabel('Q(w)');
grid on
hold off;

figure(3);
plot(TAU,T,'k','LineWidth',2)  %Tikslo funkcija
axis([0 30 0 1.4])
xlabel('TAU'); ylabel('K');
grid on

 Kkritmax   %Optimali tikslo funkcijos reiksme
 taumax   % reguliariaus rekonstravimo laiko velinimas
```