View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by DSpace at Tartu University Library

TARTU ÜLIKOOL
Loodus- ja täppisteaduste valdkond
Füüsika instituut

Rando Avarmaa

TEHISNÄRVIVÕRKUDE RAKENDATAVUSE HINDAMINE FOUKG SIGNAALI TÖÖTLEMISEKS

Bakalaureusetöö (12EAP)

Juhendajad:
Jüri Vedru
Vahur Zadin

TARTU 2017

Tehisnärvivõrkude rakendatavuse hindamine FouKG signaali töötlemiseks

Käesolevas töös uuriti võimalust parandada FouKG ja ruumalasignaali sarnasust tehisnärvivõrgu abil ning ehitati selle jaoks vajaliku südamepiirkonda imiteeriva süsteemi ehk fantoomi. Tulemuseks saadi, et tehisnärvivõrk suurendab valminud fantoomilt registreeritud FouKG- ja ruumalasignaalide sarnasusindeksit ning korrelatsiooni, vähendab keskmist ruuthälvet. Seetõttu on tehisnärvivõrk sobilik meetod töös esitatud probleemi lahendamiseks. Selgus, et konstrueeritud fantoomi käitumine ruumalasignaalide tekitamisel ei ole piisavalt hästi kontrollitav, et selle kaudu hinnata FouKG signaali täpsust. Saadud tulemused on paljulubavad, kuid veenvamate järelduste saamiseks on vajalik nii fantoomi kui ka närvivõrgu edasist täiendamist.

Märksõnad: tehisnärvivõrgud – Foucault' kardiograafi - mittelineaarne regressioon

CERCS: T115, B115

Evaluation of applicability of artificial neural network for FouCG signal processing

In this paper, a possibility to improve the similarities of FouKG and volume signals with neural network was investigated. For this purpose, a physical model of human cardiac area was built. It was found, that neural network increases the similarity indices and correlations of the signals that were measured on the cardiac model. Furthermore, the mean squared errors of the signals decreased significantly. Therefore, neural network proved to be applicable to solve the problem introduced in the thesis. The volume signal of the physical model of human cardiac area turned out to be too poorly controllable, to accurately evaluate the current precision of FouKG. Obtained results are promising, but further development of the physical cardiac model and neural network is needed to come to more convincing conclusions.

Keywords: artificial neural networks - Foucault' cardiography - non-linear regression

CERCS: T115, B115

Kasutatud terminid ja tähistused

Kopsutekkeline komponent – Foucault' kardiogrammi komponent, mis on põhjustatud kopsude tegevusest.

Südametekkeline komponent – Foucault' kardiogrammi komponent, mis on põhjustatud südame tegevusest.

Foucault' kardiograaf – Foucault' vooludel baseeruv andur inimese südame ruumala monitoorimiseks.

Foucault' kardiogramm – Foucault' kardiograafi väljundsignaal.

Fantoom – inimese rindkere südamepiirkonda imiteeriv süsteem.

Tehisnärvivõrk – arvutusmeetod, milles kasutatakse väljundi saamiseks suurel hulgal tehisneuroneid.

Treeningandmed – andmete komplekt, mis sisaldab nii sisendit, kui ka etalonväljundit. Seab iga sisendi vastavusse etalonväljundiga, mille järgi närvivõrk treenib.

Etalon – fantoomilt registreeritud ruumalasignaal.

Tank – fantoomi korpus.

Kontraktsiooniliikumine – südame ruumala muutus kontraktsioonitsükli käigus.

Täielikult ühendatud närvivõrk – närvivõrk, milles iga kihi kõik neuronid on ühendatud järgmise kihi kõikide neuronitega.

Ruumalavõnkumine – fantoomi südameballooni ruumala muutumine ajas.

Pendeldamine – fantoomi südameballooni pendlitaoline liikumine kinnitustoru otsas.

Sisukord

Sissejuhatus

Tänapäevases maailmas tõuseb tempokalt nõudlus kehategevuse monitoorimisseadmete järgi. Suurt huvi pakub just südametegevuse jälgimine, mis on eelkõige vajalik südame- ja veresoonkonna haigustega patsientide ravimiseks. Ohutuma ja efektiivsema treenimise jaoks on südame parameetrite jälgimine oluline ka sportlastele. Lisaks tuntakse järjest rohkem huvi südame tegevuse jälgimise vastu sõduritel. Üks olulist informatsiooni andev parameeter on südame löögimaht, mille hindamiseks arendatakse Tartu Ülikoolis meetodit nimega Foucault' kardiograafia. Meetod kuulub niinimetatud elektriliste bioimpedantsmeetodite hulka. Foucault' kardiograafiaga on tegeldud Tartu Ülikoolis juba aastakümneid. Esmakordselt asus seda meetodit Tartus uurima 1960. aastatel tollases TRÜ biofüüsika laboratooriumis Leo-Henn Humal. Foucault' kardiograafia arendamiseks andis aluse arstide vajadus südame mehaanilise tegevuse kestvaks mitteinvasiivseks monitoorimiseks. Esialgsed ideed meetodist on Tartus tekkinud 1960. aastatel, suurem osa uurimistööst toimunud pärast 1990ndaid. Selle aja jooksul on tegeletud meetodi rakendamiseks vajaliku riistvara konstrueerimisega, kardiogrammi tekke uurimisega, signaalist kopsutekkelise komponendi eraldamisega ning rindkere ja Foucault' kardiograafi vastasmõju arvutisimulatsioonidega. Praegune arendustegevus on suunatud rindkere simulatsioonide täiustamisele, signaalitöötlustarkvara uuendamisele ning Foucault' signaali päritolu kindlakstegemisele. Viimasele on pühendatud ka käesolev bakalaureusetöö. Tänaseks on teada, et Foucault' kardiograafi signaal on tugevasti seotud südame mehaanilise tegevusega. Sellest hoolimata ei ole meetodile saavutatud igapäevast rakendust.

1. Kirjanduse ülevaade

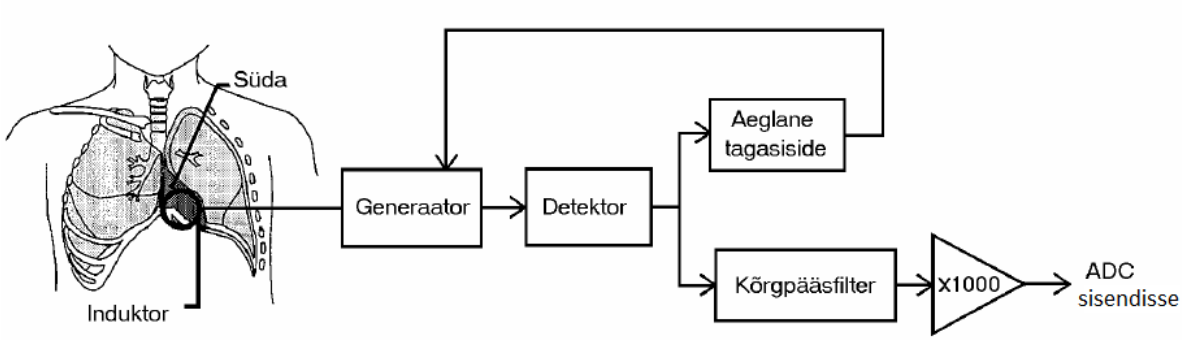
Ideaalne südame mehaaniliste protsesside jälgimise meetod peaks suutma jälgida südame kambrite verega täitumise ajalist kulgu, määrata südame iga löögiga väljutatavat vere hulka, olema odav, mugav ja lihtne kasutada ning võimaldama ööpäevi kestvaid pidevaid mõõtmisi ehk *monitooringut*. Südame elektrilise tegevuse jälgimise meetoditest omab nimetatud omadustele lähimaid elektrokardiograafia (EKG). Vaja oleks meetodit, mis oleks EKG-sarnaste omadustega, kuid võimaldaks jälgida südame elektriliste tegevuste asemel mehaanilisi.

Kasutaja mugavusest lähtuvalt on oluline, et mõõtevahend ei segaks patsiendi ega arsti tavapäraseid toiminguid. Nendes aspektides on silmapaistvad kõik elektrilised bioimpedants meetodid (EBI-meetodid). Seniste EBI-meetodite üldisteks puudusteks raskused signaali interpreteerimisega ning sellest tulenev madal usaldusväärsus.

Nimetatud kategooriast (EBI) on enim levinud impedantskardiograafia, mis põhineb suuresti Kubiceki meetodil [1]. Meetodit on aastate jooksul täiustatud [2][3], kuid kindlat seisukohta selle usaldusväärsuse kohta ei ole. Osa uurimusi väidab Ficki meetodiga ja termolahjendusega võrreldavat täpsust [4] , kuid on ka neid, mille puhul tulemustega arvestada ei saa [5]. Mõõtmise teostamiseks peab patsiendi kehale kinnitama elektroodid, mille vahendusel lastakse inimese rindkerest läbi kõrgsageduslikku nõrka elektrivoolu. Südames ja veresoontes oleva vere hulga pulseerimise tõttu muutub pidevalt ka rindkere elektriline impedants, mis kajastub ümber patsiendi kaela ja rindkere paigutatud elektroodide vahelise pinge muutusena. [6]

Foucault' kardiograafia on mitteinvasiivne EBI-meetod inimese või imetajate südame mehaaniliste protsesside jälgimiseks. See põhineb südamepiirkonna sondeerimisel raadiosageduslike pöörisvooludega, mis tekitatakse induktori vahendusel vahelduvmagnetvälja abil. Pöörisvoolude energia, mis muundub inimkehas soojuseks, on anduriga mõõdetav.

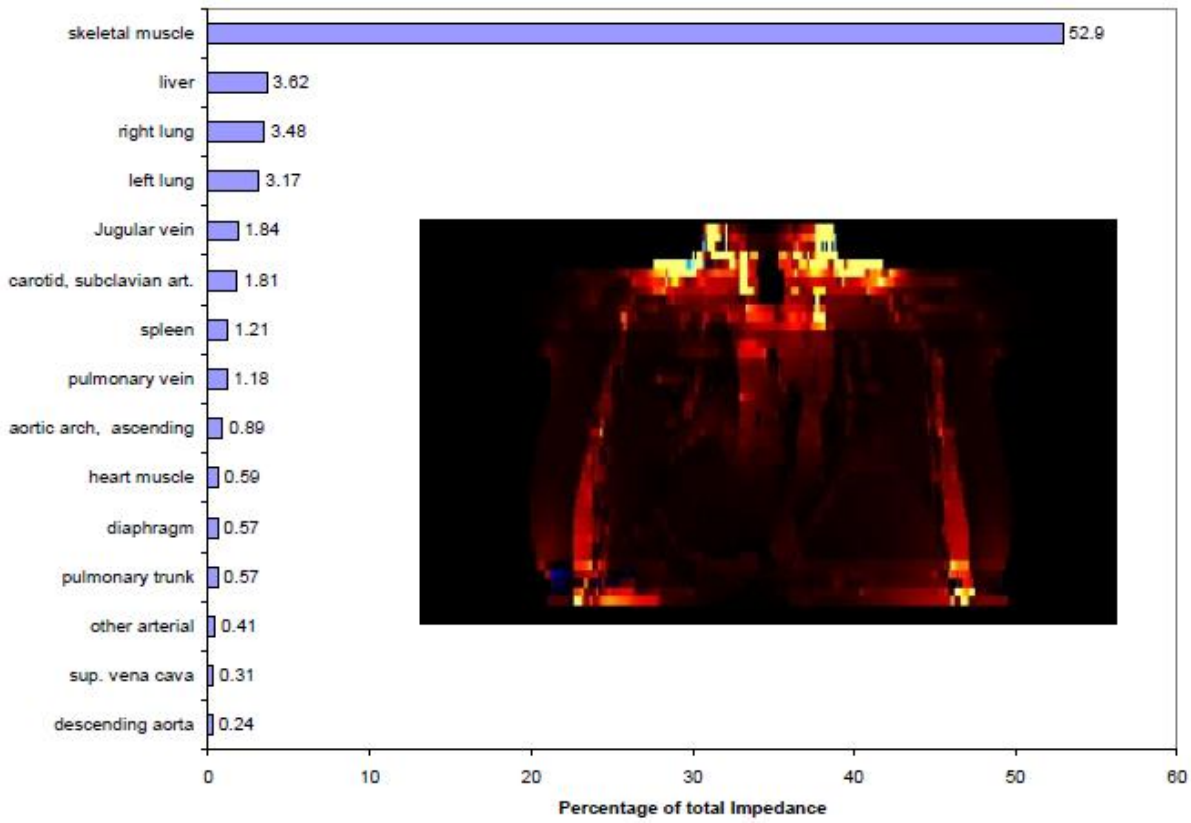
Praegu Tartus kasutatava süsteemi skeem (Joonis 1) sisaldab induktiivselt tagasisidestatud LC-generaatorit; induktorit, mis on ühtlasi generaatori induktiivsuseks; generaatorit marginaalselt kadudetundlikus võnkerežiimis hoidvat negatiivse tagasiside võimendi; generaatori võnkumiste amplituuddetektorit ning viimasest saadava südamesagedusliku signaali võimendit.



Joonis 1. Tartu Foucault' kardiograafi käesoleva versiooni plokkskeem. Joonis magistritööst [7].

Induktorpoolis tekkiva pingevõnkumise amplituud sõltub induktori lähedases objektis, näiteks inimkehas, neeldunud energiast. Foucault' meetodiga on sobiv sondeerida elektriliselt hästi juhtivat objekti halvasti juhtivas keskkonnas, näiteks südant, mis asub kopsude vahel. Induktorpool on väljastpoolt varjestatud, et vältida atrefaktse signaali teket kardiograafi ja mõõteobjekti vahelise mahtuvuse muutumise toimel. Sellises skeemis on induktorpool ühtlasi nii signaali tekitaja kui ka vastuvõtja. Induktori sattudes objekti lähedusse neeldub selles osa energiat. Neeldumiskaod moduleerivad generaatori võnkumiste amplituudi. Üldjuhul on väljundsignaalis (FouKG signaalis) mitu sõltumatut komponenti, millest domineerivad kopsu- ja südametekkeline signaal. Kopsutekkeline signaal tekib kopsude õhu ja vere läbivoolust tingitud ruumala muutusega. Südametekkeline signaal sõltub vere kogusest südames, südame kujust ja anduri asendist südame suhtes. Väljundsignaal sõltub lisaks anduri liigutamisest keha suhtes ning elektrivõrgust tingitud mürast.

Foucault' kardiograafia põhiline erinevus teistest EBI-meetoditest on see, et signaal tekitatakse induktiivselt põhiliselt südmepiirkonnas [8]. Teiste EBI-meetodite korral moodustab aorti impedants alla 1% koguimpedantsist [6]. Teiseks erinevuseks on see, et FouKG registreerimiseks on vaja kasutaja kerele kinnitada vaid üks seade, mis võib olla mõõtmise ajal ka riiete peal.



Joonis 2. Erinevate kehapiirkondade ja organite panus kogu bioimpedantsi, rakendades impedantskardiograafiat (heledad kollased alad on elektroodide kinnituspunktid). Tulemused on arvutatud mudeli järgi, heledam toon näitab suuremat tundlikkust lokaalsele elektrilisele impedantsile (Joonis ja seletus artiklist [6]) .

Südame- ja veresoonkonna tegevuse monitoorimine on südamehaigetel patsientidel eluliselt tähtis. Südame väljutuse ja löögimahu pideval jälgimisel on kliiniliselt palju rakendusi [9]. Ometi ei ole tänaseni olemas nende suuruste hindamiseks üldiselt tunnustatud ja standardset meetodit. FouKG võiks täita meditsiinis just sellise koha.

Foucault' kardiograafi on võimalik muuta väga kompaktseks ning mikroskeemi kujul mahutada plaastrisse või ümber rindkere kinnitatud vöösse [10]. See asjaolu loob meetodile lisaks kliinilisele kasutusele palju teisi rakendusi, näiteks sportlaste ja sõdurite südametegevuse monitoorimine. Artikli [11] järgi on tulevikusõduril küljes mitmeid erinevaid andureid ja sensoreid, mis muuhulgas peaksid registreerima sõduri pulssi kui ka vererõhku. Südame väljutust teades on võimalik arvutada kudede oksüdatsioonitase, mis võimaldab hinnata keha väsimust. On teada, et treenitud inimese südame löögimaht on suurem [12]. Sellest tulenevalt

on võimalik löögimahtu monitoorides hinnata jälgitava isiku vastupanuvõimet kehalisele koormusele või treenitust. Selline informatsioon on kasulik ka sportlaste seas.

FouKG on alles arengustaadiumis, mistõttu on meetodil siiani nii mõnedki puudused või uurimata omadused, mis takistavad selle pakkumist arstidele ning teistele kasutajatele. On teada, et hingamine mõjutab südame tööd ning lisab Foucault' kardiogrammi lisakomponendi [13], kuid siiani ei ole suudetud kopsutekkelisi komponente rahuldaval viisil signaalist eraldada. Kalibratsioonita Foucault' kardiogrammi kaudu ei ole võimalik hinnata südame ruumala muutusi absoluutses skaalas. Ilma kalibratsioonita saab vaid suhtelise muutuse, mis aga ei pruugi olla piisavalt informatiivne. Samuti on kontrollimata hüpotees seosest südame vatsakeste ruumala muutumise ja FouKG anduriga registreeritava lainekuju ja vahel. Siiani on vaid tõdetud, et FouKG signaal on südame kontraktsioonikõveraga sarnane [13], kuid see ei ole katseliselt kontrollitud.

Nimetatud seost on vajalik enne edaspidiseid uuringuid kontrollida. Üks võimalus on teha loomkatsed ning kontrollida tulemust mõne verise meetodiga [14]. Teine võimalus on ehitada fantoom ehk inimese südame piirkonda imiteeriv süsteem, milles on võimalik tekitada kontrollitud ruumalamuutusi. Samaaegselt peab mõõtma tekitatud muutusi ka Foucault' kardiograafiga, ning kahte tulemust omavahel võrreldes andma hinnang nende sarnasusele. Esimene variant annab täpsema hinnangu reaalsemas situatsioonis, kuid on raskemini kättesaadav. Teine võimalus annab teadmise, kui hästi FouKG-ga ruumalamuutusi registreerida saab. Käesolevas bakalaureusetöös just viimast lähenemist kasutataksegi.
Eelduste kohaselt on Foucault' kardiogramm heas kooskõlas fantoomis tekitatud ruumalamuutusega. Signaalide lahknevuste korral on vajalik uurida võimalusi tulemuste parandamiseks.

Keeruliste mittelineaarsete süsteemidega opereerimisel on silmapaistvad oma võimekuse, robustsuse ja adaptiivsuse poolest tehisnärvivõrgud [15]. Neid kasutatakse paljude silmapaistvate ülesannete lahendamiseks – meditsiinialastest rakendustest näiteks kopsuvähi diagnoosimiseks [16] ja pärasoolevähi ravitavuse ennustamiseks (täpsem kui teised kliinilised meetodid) [17]. Tehisnärvivõrkude kasutamiseks peab olema hulk teadaolevaid sisend- ja väljundvektori väärtusi, mida kasutatakse närvivõrgu treenimiseks. Paremate tulemuste saamiseks peab olema teadaolevaid sisend-väljund paare võimalikult palju, et närvivõrk saaks piisavalt erinevate olukordadega kohaneda ning teha vastavaid üldistusi. [18]

Nimetatud omaduste tõttu sobivad tehisnärvivõrgud ka meie probleemi lahendamiseks. Meie juhul on etaloniks fantoomis teadaolevad ruumalamuutused ning närvivõrgu sisendiks FouKG signaal. Kui need omavahel erinevad, õpib tehisnärvivõrk tegema vastavaid korrektsioone.

1.1 Töö eesmärk ja ülesanded

Käesoleva bakalaureusetöö eesmärgiks on uurida võimalusi südame ruumala Foucault' kardiogrammi põhise monitooringu parandamiseks tehisnärvivõrgu abil.

Selle eesmärgi saavutamiseks tuleb lahendada järgmised ülesanded:

1) Ehitada inimese rindkere fantoom võimaldamaks tekitada kontrollitavat südame ruumalasignaali imiteerimist ja närvivõrgu treenimiseks vajalike andmete registreerimist.
2) Hinnata FouKG signaali sarnasust imiteeritud südame ruumalasignaaliga.
3) Tuvastada eelmainitud signaalide võimalikud lahknemise põhjused.
4) Teostada tehisnärvivõrgu sobivuse pilootkatsetus FouKG signaali ja imiteeritud ruumalasignaali sarnasuse parandamiseks.
5) Registreerida fantoomilt tehisnärvivõrgu rakendamiseks vajalikud signaalid.
6) Hinnata närvivõrgu võimekust probleemi lahendamiseks.

2. Materjalid ja meetodid

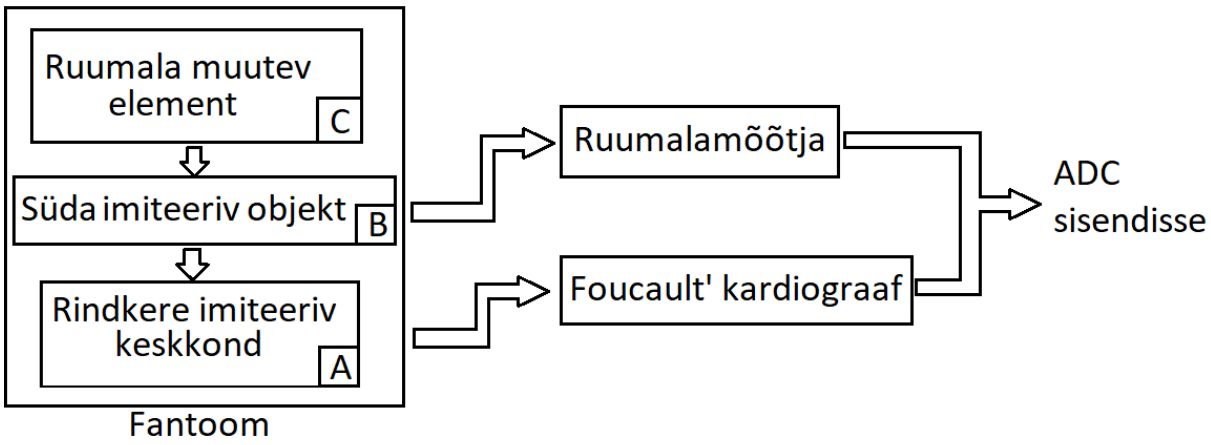
2.1 Inimese rindkere südamepiirkonna füüsiline modelleerimine

Fantoomi otstarve on imiteerida inimkeha rindkere piirkonna FouKG signaali teket põhjustavaid ja mõjutavaid omadusi. Seetõttu peavad fantoomi vajalikud parameetrid olema piisavalt sarnased tegelikkusele. Käesolevas töös pakub meile huvi, millise täpsusega suudab FouKG registreerida ruumalamuutusi hästi kontrollitavas, kuid inimkerele võimalikult sarnases keskkonnas. Fantoomi (Joonis 9) põhikomponentideks (vt Joonis 3) on:

A. Inimese (keskmise mehe) rindkerele lähedase kuju ja suurusega vedelikutank mahtuvusega 25 dm^3. Tanki otstarve on tekitada fantoomile inimese rindkerega sarnane keskkond. Tank on valmistatud 10 mm paksusest pleksiklaasist, mis imiteerib inimese rindkere väliskihti (nahk ja rasvkude). Anum on täidetud NaCl lahusega, mille kontsentratsioon (1,25 g/kg, vt Lisa 1) on valitud nii, et keskkonna elektriline eritakistus kardiograafi töösagedusel oleks sarnane kopsude omale (süda asub kopsude vahel). Meie kasutatavail sagedustel on keha impedantsid ja eriti nende muutuvkomponendid lähedased puhtaktiivsetele [19]. Selline konstruktsioon tagab teatud sarnasuse fantoomi ja inimese rindkere vahel - dielektriline väliskiht ning sarnase juhtivusega sisu. Tank on soolalahusega ääreni täidetud ning hermeetiliselt pealt kaanega suletud (vt Joonis 9).
B. Südant imiteeriv õhukese elastse seinaga lateksballoon. Seda otstarvet täidab lateksist preservatiiv, mis on piisavalt elastne ja vastupidav. Südame paremaks imitatsiooniks on membraan täidetud soolalahusega, mille eritakistus Foucault' kardiograafiga tekitatavate pöörisvoolude sagedusel on lähedane südame keskmisele eritakistusele (1,25 g/kg, vt Lisa 1). Südameballooni ümbritseb perikardi imiteeriv võrkkott, mis osutus vajalikuks, et vähendada südameballooni füsioloogilisest liikumisest erinevat horisontaaltasandis toimunud võnkumist.
C. Süsteem südametöö imiteerimiseks. Süda imiteeriv balloon on ühendatud plasttoruga (ø40 mm, PVC), mis väljub läbi tanki kaane. Toru välimise otsa külge on kinnitatud lehter, mille peale on venitatud membraan (kirurgi kummikinnas). Membraani läbib veekindlalt kolb, mis on ühe otsaga lehtrist väljas ning teisega torus. Kolb ei ole toru sees ideaalselt tihendatud, st laseb vett läbi. Selline süsteem võimaldab tekitada piisava keerukusega võnkumisi, et imitatsioonilaine vastaks keerukuselt reaalsele südame ruumala lainele.

D. Ületõusutoru. Kui tankis südameballoon laieneb, tõuseb osa vedelikust tanki kaant läbivasse ületõusutorusse (ø32 mm, PVC).

E. Kunstsüdant toestav alus, mis vähendab tuiklemist ning imiteerib bioloogilise südame toetumist diafragmale.



Joonis 3. Fantoomi plokkskeem.

2.2 Signaali ja võrdlussignaali registreerimine

2.2.1 Foucault' kardiograaf

Foucault' signaali registreerimiseks kasutasime Jaanus Trolla poolt magistritöö [7] raames ehitatud pt 1-s kirjeldatud Foucault' kardiograafi. Selle signaal muundati firma National Instruments analoog-digitaalmuundiga NI-6215 ja saadeti andmeid koguvasse ja töötlevasse arvutisse. Arvutis kasutasime andmete registreerimiseks National Instrumentsi tarkvara LabView 2010 (vt Lisa 2).

2.2.2 Rõhuandur

Südameballooni paisumine ja kahanemine tekitab ületõusutorus ajas muutuva vedelikusamba kõrguse. Mõõtes vedelikusamba poolt tekitatud rõhku, saame signaali, mis on

$$p = \rho g h = \rho g \frac{4V}{\pi D^2} \quad (1)$$

järgi võrdeline ruumalaga (D – ületõusutoru sisemine läbimõõt). Rõhu mõõtmiseks kasutame firma Honeywell ASDX seeria rõhuandurit, mille mõõtepiirkond on 0 kuni 25 mbar,

reaktsiooniaeg 1 ms, resolutsioon 12 biti ja mõõteriista määramatus ±2% kogu skaala maksimumist. Andur on mõeldud rõhu mõõtmiseks gaasis (õhus), tema sisend ei saa olla vedelikuga vahetus kontaktis. Rõhuanduriga vedeliku rõhu mõõtmiseks ühendati rindkeretankiga läbi selle kaane rõhukonverter vedeliku rõhu muutmiseks õhu rõhuks (Joonis 8). Tanki kaanest väljus PVC-voolik, mis sisestati hermeetiliselt -0,2 dm^3 mahuga klaaspurki. Nii voolik kui ka purk olid täidetud kopsu imiteeriva lahusega. Purgis asus väike lehter (nn Marey kapsel), mille laiemale otsale liimiti lateksist pingul membraan ning peenemast otsast suunati rõhuanduri sisendisse teine, õhuga täidetud PVC-voolik. Paigutades rõhukonverteri sobivale kõrgusele, oli võimalik seadistada rõhuanduri signaali null sobivale hüdrostaatilisele rõhule. Ka rõhuanduri väljundpinge salvestati analoog-digitaalmuunduri vahendusel arvutis programmiga LabView.

2.2 Signaalide sarnasuse hindamine

Signaalide lainekuju võrdlemiseks on vajalik nende sarnasuse mõõt, mille konstrueerimiseks on palju võimalusi. Käesolevas töös on kasutatud sünkroniseeritud signaalide võrdlemiseks korrelatsioonikordajat ning sarnasusindeksit, mis esmakordselt võeti kasutusele artiklis [20] ja on kasutatud ka magistritöös [21].

Sarnasusindeks R sõltub signaali nullindat, esimest ja teist järku tuletiste korrelatsioonikordajatest ning signaalide algfunktsioonide (ehk -1. järku tuletiste) korrelatsioonikordajast.

$$R = [\alpha\rho_{-1}^2 + (1-\alpha)] \cdot [\beta\rho_0^2 + (1-\beta)] \cdot [\gamma\rho_1^2 + (1-\gamma)] \cdot [\delta\rho_2^2 + (1-\delta)]. \qquad (2)$$

Muutujad $\rho_{0,1,2}$ tähistavad vastavat järku tuletise ning ρ_{-1} algfunktsioonide korrelatsiooni. Suurused α, β, γ ja δ on vastavate komponentide kaalud. Sarnasusindeks suureneb võrreldavate signaalide sarnasuse suurenedes ning muutub vahemikus 0 kuni 1.

Käesolevas töös on kaalude väärtused samasugused nagu eelmainitud töödes, st $\alpha = 1$, $\beta = (1/3)^{\frac{1}{2}}$, $\gamma = (1/3)^{\frac{2}{2}}$ ning $\delta = (1/3)^{\frac{3}{2}}$.

Selline indeks on siiani vähe kasutust leidnud, mistõttu kasutan tulemuste esitamisel lisaks sarnasusindeksile ka korrelatsioonikordajat ning keskmist ruuthälvet. Töös esitatakse sarnasushinnangud 2 sekundi pikkustele signaalilõikudele.

2.3 Tehisnärvivõrgud

Tehisnärvivõrgud loovad aluse arvutipõhisele arvutusmeetodile, milles kasutatakse probleemi lahendamiseks suurt hulka tehisneuroneid. Meetod on saanud inspiratsiooni bioloogiliselt ajult, milles bioloogilised neuronid on ühendatud sünapside abil. Iga tehisneuron on ühendatud paljude teistega. Tehisneuronite vahelised ühendusi iseloomustab seevastu ainult üks number, mis näitab ühenduse mõju ehk kaalu. Kaal võib olla nii positiivne kui ka negatiivne, sõltuvalt sellest, kas üks tehisneuron võimendab või vähendab teiste neuronite mõju. Mida suurem on ühenduse kaal absoluutväärtuselt, seda suurem on selle mõju.

Kõige lihtsama ja enamkasutatava närvivõrgu struktuur koosneb kolmest osast: sisendkiht, varjatud kihid ning väljundkiht. Sealjuures on sisend- ja väljundkihte alati üks, kuid varjatud kihte võib keerulisemate struktuuride loomiseks olla ka mitu. Igas kihis võib olla sõltuvalt probleemist üks kuni mitusada või enam neuronit. Enim levinud närvivõrgu tüüp on pärilevivõrk, kus informatsiooni edasikandumine võrgusiseselt ei moodusta tsüklit. Käesolevas töös on kasutatud ainult pärilevivõrke ning kõik kirjeldused käivad seda tüüpi võrkude kohta.
Allpool oleval Joonisel 5 on esitatud näide informatsiooni liikumisest läbi neuroni. Sisendid $x_{1,2,\dots,K}$ korrutatakse kaaludega $w_{1i,2i,\dots,Ki}$ ning summeeritakse kokku vabaliikmetega θ_i. Vabaliikmete otstarve on anda neuronitele treenitavad konstantsed lisaparameetrid. Neuroni sisendid n_i on seega määratud nii kaaludega ning vabaliikmetega ning on antud kujul

$$n_i = \sum_{j=1}^{K} w_{ji} x_j + \theta_i. \tag{3}$$

Neuroni väljundi mittelineaarsuse tagamiseks kasutatakse aktiveerimisfunktsiooni $g(n_i)$. Enim kasutatavad aktiveerimisfunktsioonid on hüperboolne tangens (tanh*)* ning sigmoidfunktsioon (S), käesolevas töös kasutatakse neist viimast (vt Joonis 4).

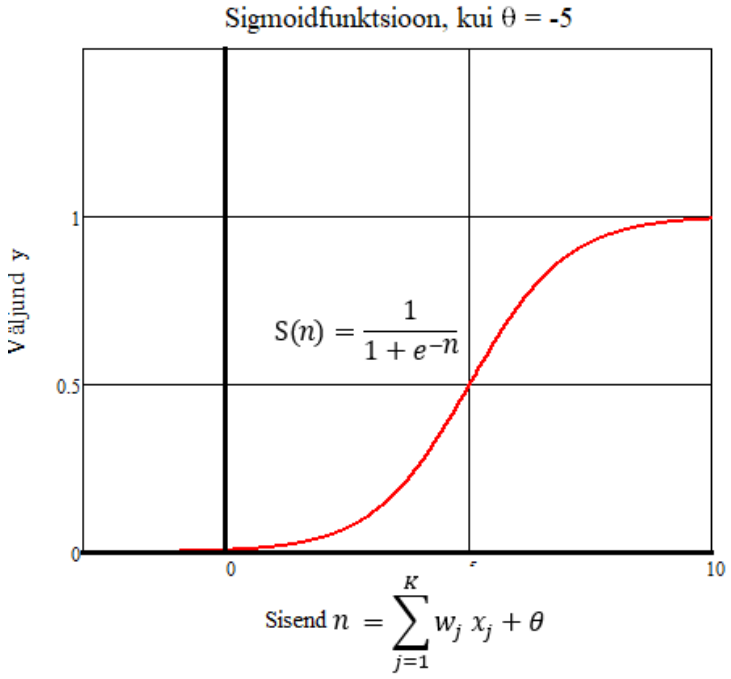
$$\tanh(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{4}$$

$$\mathrm{S(x)} = \frac{1}{1 + e^{-x}} \tag{5}$$

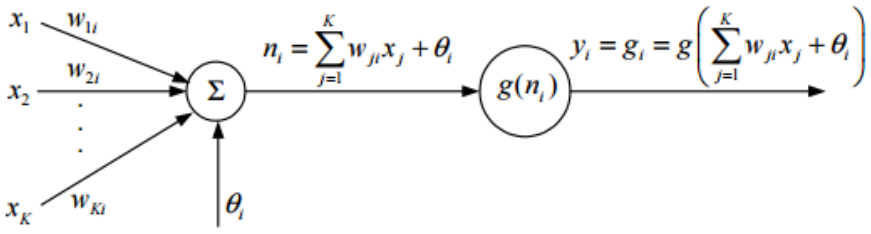
Aktiveerimisfunktsiooni $g(n_i)$ saab seega kirja panna kujul

$$y_i = g_i = g(\sum_{j=1}^{K} w_{ji} x_j + \theta_i), \tag{6}$$

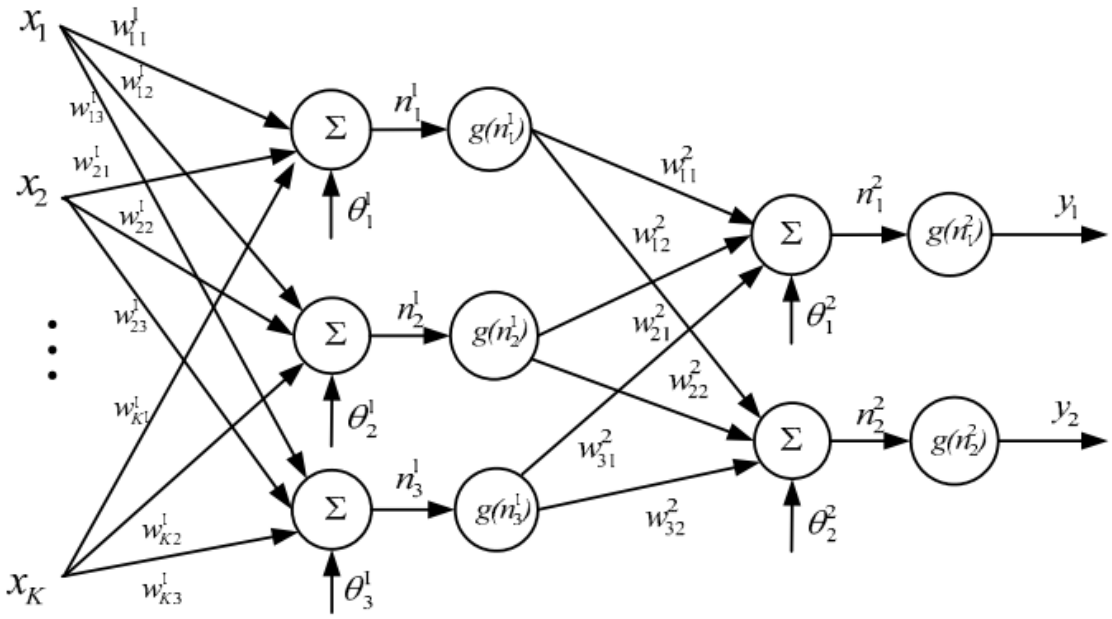
mis on ühtlasi ka sisendiks y_i järgmistele neuronitele. Seejärel käitub eelmise kihi väljund y_i järgmise kihi sisendina x_s ning allpool (Joonis 6) toodud skeemi järgi liigub informatsioon kuni väljundkihini. Sedasi formeerub tavaline närvivõrk, mille skeem on kujutatud allpool (Joonis 6). [22]



Joonis 4. Neuroni väljundi sõltuvus sisendite ja kaalude korrutiste summast. Kui summa on väga väike või väga suur, siis väljund on praeguses näites vastavalt ligikaudu 0 või 1. Aktiveerimisfunktsioon ei lase seega väljundit üle satureerida. Vabaliige θ määrab sigmoidi nihke sisendteljel.



Joonis 5. Skeem ühest neuronist. Selgitused ja skeem võetud õpikust [22].



Joonis 6. Ühe varjatud kihiga närvivõrgu skeem. Skeem võetud õpikust [22].

See, kuidas neuronid omavahel sõlmedega seotud on, sõltub konkreetsest ülesandest ja närvivõrgu struktuurist. Tavaliselt on kihid omavahel sõlmedega täielikult ühendatud, st iga eelneva kihi neuron on ühendatud iga järgneva kihi neuroniga. Selliselt on konstrueeritud ka Joonisel 6 olev närvivõrk. Samuti on kõik käesolevas töös kasutatavad närvivõrgud täielikult ühendatud.

Treenimismeetodit, kus näidiskomplektide $(x, y), x \in X, y \in Y$ kaudu leitakse funktsioon $f: X \to Y$, nimetatakse valvatud õppimiseks. Suurus x kujutab endast treeningandmete sisendväärtusi (meie juhul treenimiseks mõeldud FouKG signaal) ning y vastavaid etalonväärtusi (FouKG-le vastavad rõhusignaalid). Närvivõrgu eesmärgiks on minimeerida x ja y vaheline keskmine ruuthälve. Sisendid x liiguvad ülal kirjeldatud skeemi järgi läbi närvivõrgu, ning väljundisse jõudes võrreldakse neid vastavate etalonväärtustega y. Seejärel arvutatakse väljundi ja etalonväärtuse vaheline keskmine ruuthälve ning tehakse kaaludes w ning vabaliikmetes θ väikesed muudatused. Selliselt muudetakse närvivõrgu parameetreid nii kaua, kuni on jõutud soovitud tulemuseni. Esimese iteratsiooni jaoks valitakse parameetrid juhuslikult. Selline protsess on närvivõrkude treenimisel kõige levinum ning seda nimetatakse tagasilevi meetodiks. Käesolevas töös kasutatakse kõikide närvivõrkude treenimiseks tagasilevi meetodit.

Närvivõrkude kasutamisel mittelineaarse regressiooni ülesande lahendamiseks on üks levinud probleem ületreenimine. See tähendab, et närvivõrk ei õpi ära treeningandmete üldiseid seaduspärasusi, vaid üritab need lihtsalt „meelde jätta“. Selle tagajärjel võib sisendsignaalis olnud müra ja muu ebaoluline informatsioon jõuda ka väljundisse. Käesolevas töös kasutatati üleõppimise vältimiseks treeningprotsessi nimetusega Bayesian regularization backpropagation. Närvivõrkude ehitamiseks Python 2.7 teeki PyBrain, ning MATLAB-i Neural Network *toolbox*-i.

Varasemalt oli teada, et Foucault' kardiograafist saadud signaal sõltub induktori asukohast rindkerel [8] ning tehisnärvivõrkude abil peaks saama seda efekti vähendada [23]. Ideaalsel juhul peaksime närvivõrgu väljundina saama rõhuanduriga mõõdetava signaaliga identse kujuga kõvera. Töös lahendatavat probleemi tuntakse üldiselt kui mittelineaarset regressiooni, mille lahendamisel kasutatakse närvivõrke.

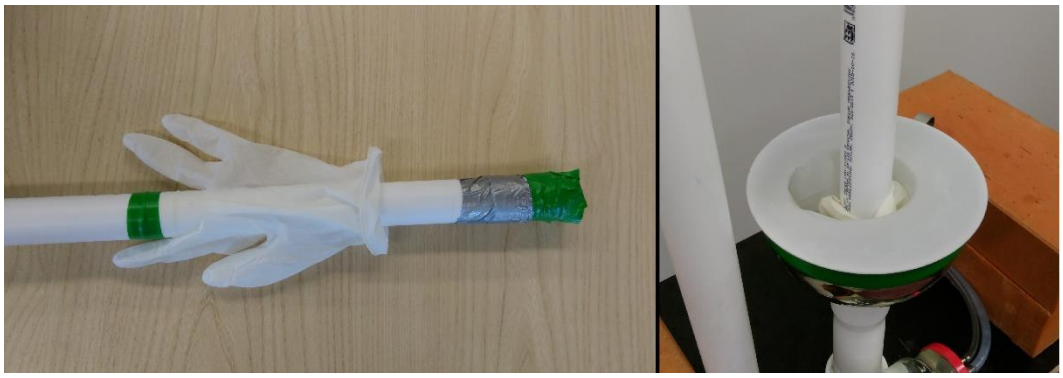
3. Tehtud töö ja tulemused

3.1 Fantoomi valmistamine

Fantoomi ehitamise puhul lähtuti olemasoleva tanki mõõtmetest ja eripäradest. Esimesena valmis tankile kaas. Materjaliks valiti 18 mm paksune veekindel vineer, mille sisse tehti veekindlad avad torude läbiviimiseks ning ühendus rõhu mõõtmise süsteemiga (vt pt 2.2.2).

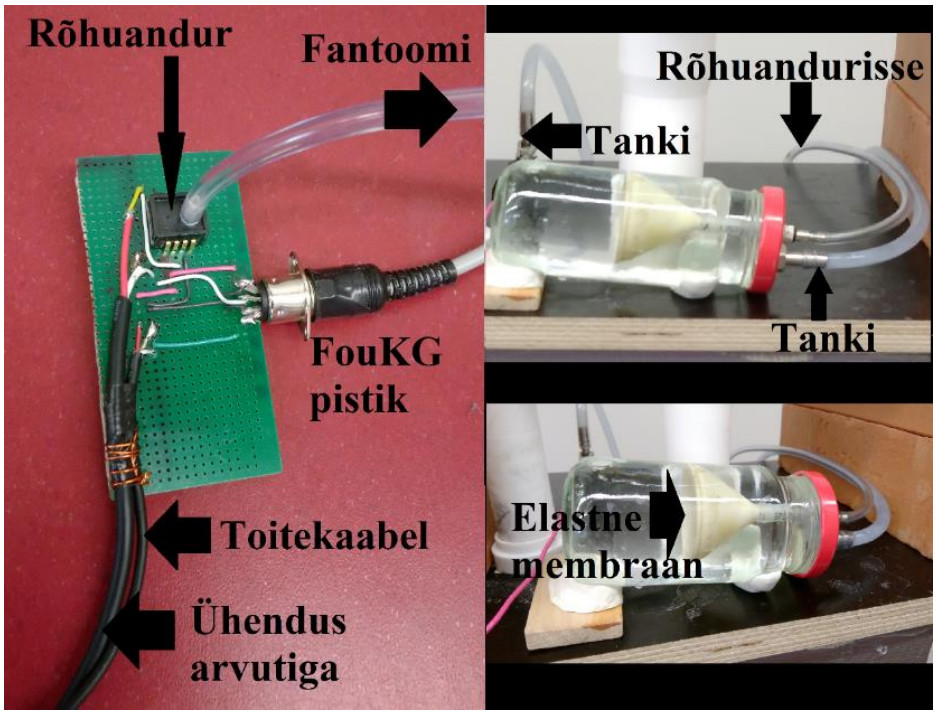
Järgnevalt valiti südant imiteeriv objekt (B - Joonis 9), milleks eeldati sobivat lateksist preservatiiv. Bioloogilise südamega sarnase mehaanilise käitumise tagamiseks, loodeti imitatsiooni elastsus saada võrreldavaks päris südamega. Osutus, et sarnase elastsuse saamine ei olnud käesoleva katse juures määrava olulisusega.

Järgmisena määrati kasutatavate torude optimaalsed läbimõõdud. Liiga peenikeses torus oli takistus suur, kuid liiga jämedas võivad hakata vedeliku pinnalained tekitama väljundisse lisakomponente. Esimese vedeliku taseme mõõtjana kasutati 12 tollist eTape *Liquid Level Sensorit.* Ületõusutoru paksus pidi olema selline, et kunstsüdame löögimaht muudaks veetaset ligikaudu 70% ulatuses sensori mõõtepiirkonnast. Kõige paremini sobis selleks 32mm läbimõõduga PVC toru. Kunstsüdame ruumala muutmiseks sobis 40mm läbimõõduga PVC toru. Nimetatud toru teise otsa kinnitasime lehtri, mille peale fikseeriti kirurgi kummikinnas, mis täitis membraani eesmärki. Selline süsteem võimaldas eelduste kohaselt tekitada vajaliku kujuga ruumalasignaale, järgides täpselt survet membraanile. Selgus, et süsteem on suure inertsiga, mistõttu domineeris membraanile surve avaldamise viisist ühe sagedusega ruumalasignaali komponent. Selle probleemi lahendamiseks lisati lisaks kummikindale toru sisse kolb, mis ei oma erinevalt membraanist elastsust ning summutab kiiresti kõik teised komponendid peale nende, mis kolviga tekitatud.



Joonis 7. Kolb koos kummikinnas-membraaniga.

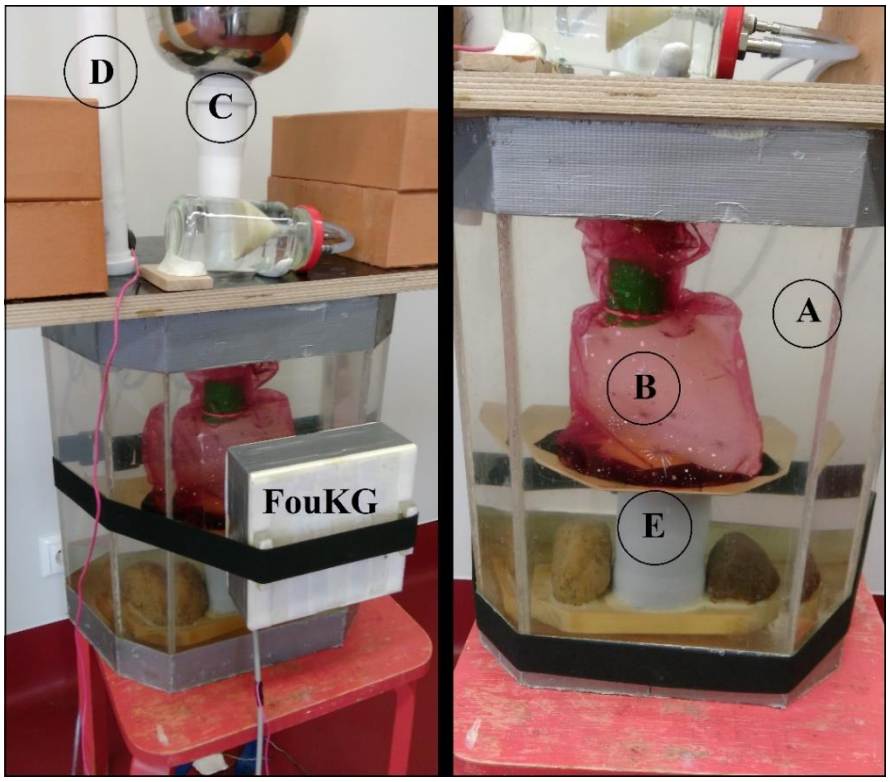
Järgnevalt oli vaja registreerida kunstsüdame ruumalaga võrdeline signaal. Selgus, et eTape veetasememõõtja ei talu kiireid vedelikutaseme muutuseid. Järgmisena proovisime võrdluskõvera saamiseks pt 2.2.2 kirjeldatud rõhuandurit.



Joonis 8. Vasakul elektriskeem rõhuanduriga mõõtmiseks. Paremal rõhu mõõtmiseks valmistatud lisaseade.

Esimeste mõõtmiste tulemusena paistsid FouKG ja rõhusignaalid suures osas erinevad. Seda põhjustas kunstsüdame tuiklemine – FouKG arvestab nii ruumala- kui ka asendimuutust, kuid rõhuandur ainult ruumalamuutust. Selleks, et tuiklemist vähendada, oli vaja kunstsüda altpoolt toestada. Ka bioloogiline süda toetub diafragma peale, mis on väikese kaldenurga all. Toestava

aluse ehitamisel arvestati ka seda, et selle kaldenurk oleks võrreldav reaalsusega. Kunstsüdame edasise fikseerimise ja tuiklemise vähendamise eesmärgil pandi see võrkkoti sisse, mis on alumisest otsast kinnitatud aluse ja ülemisest otsast kunstsüdame toru külge.

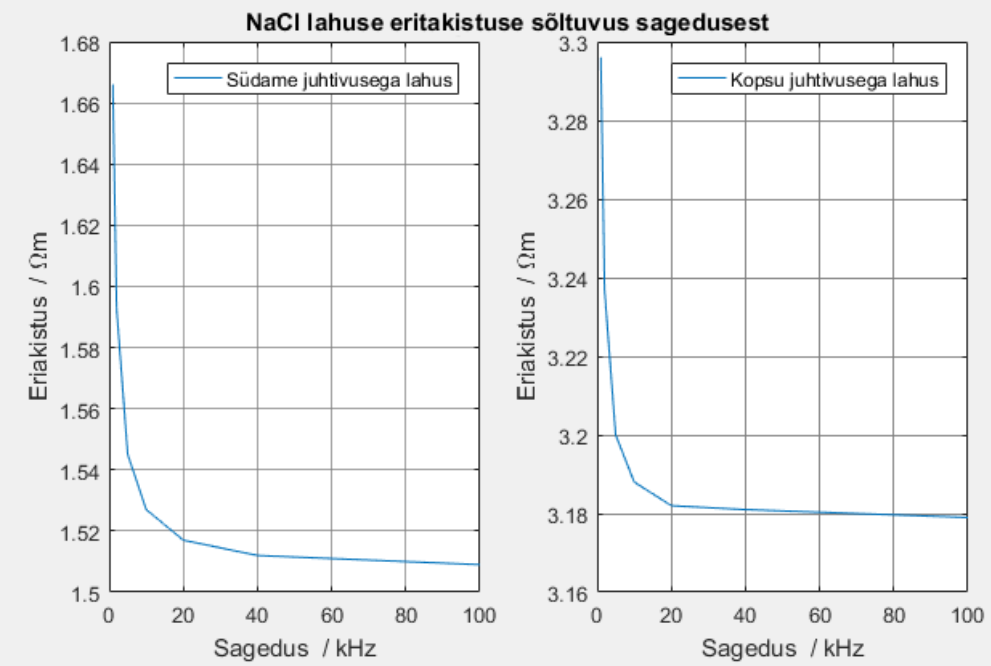


Joonis 9. Paremal tankfantoom ning vasakul fantoom koos mõõtmisasendis oleva FouKG anduriga. Komponentide kirjeldused pt 2.1.

3.1.1 Eritakistuste hindamine

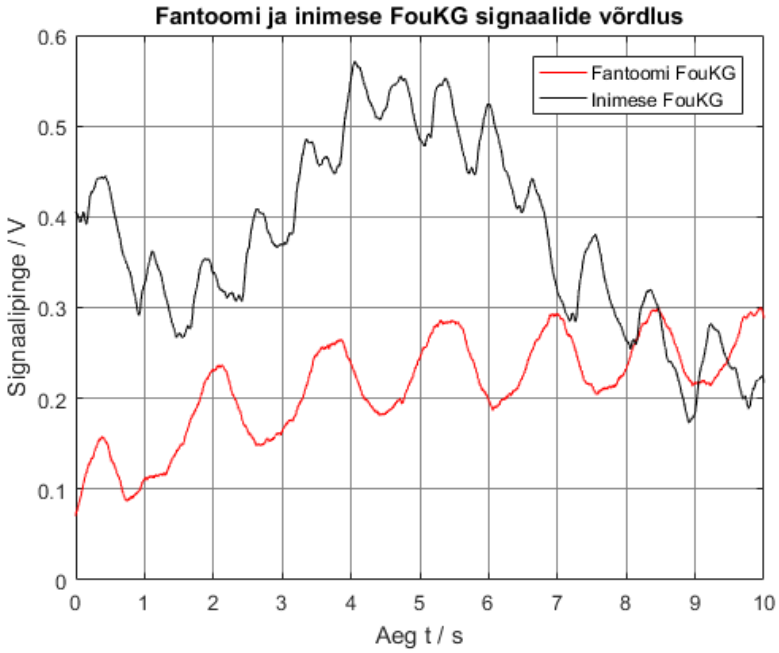
Viimasena määrati tanki ja kunstsüdame sees oleva vedeliku soolasus nii, et nende eritakistused generaatori sagedusel (8 MHz) oleksid sama bioloogilistele vastetele (süda ja kops). Südame ja kopsu eritakistused olid varasemalt teada - keskmisteks väärtusteks südame puhul 1,50 Ωm ning kopsul 3,48 Ωm [7]. Vastavateks NaCl kontsentratsioonideks vesilahuses on vastavalt ligikaudu 3,30 g/kg ja 1,25 g/kg [24]. Saadud kontsentratsioonidega lahuste eritakistuse kontrollimiseks mõõdeti GW Instek LCR-819 mõõturiga takistusi sagedusvahemikus 1 kHz kuni 100 kHz. Seda põhjusega, et mõõtja suuremaid sagedusi ei genereerinud. Takistuse mõõtmiseks kasutati rööptahuka kujulist vanni, mille kahes otsas on elektroodid. Vanni ja elektroodide kõrgus 14.7mm, laius 51mm, elektroodide vahekaugus 84,5mm.

Kontrollimiseks, kas lahuste kontsentratsioonid tagasid FouKG signaalide võrreldavuse, registreeriti kardiogrammid nii inimeselt kui ka fantoomilt (vt Joonis 11). Eritakistuse sõltuvus takistusest on kujutatud allpool oleval joonisel.



Joonis 10. Töös kasutatud vedelike eritakistuste sõltuvused sagedusest vahemikus 1 kHz kuni 100 kHz.

Saadud lahuste ja vastavate organite eritakistused on võrreldavas suurusjärgus. Seda kinnitab ka Joonis 11.



Joonis 11. Fantoomi ja inimese FouKG signaalide mastaabi ja kuju võrdlus.

3.2 Tehisnärvivõrgu rakendamine genereeritud signaalidel

Veendumiseks, et närvivõrk suudab meie probleemi lahendada, rakendati seda esmalt genereeritud signaalidega. Ruumalasignaalide imiteerimiseks kasutati sumbuvaid sinusoide.

Esimeseks sammuks genereeriti treenimiseks ja testimiseks vajalikud andmepunktid. Selle jaoks arvutati vastavad funktsiooni

$$f(t) = Ae^{-t}\cos(2\pi ft) \quad (7)$$

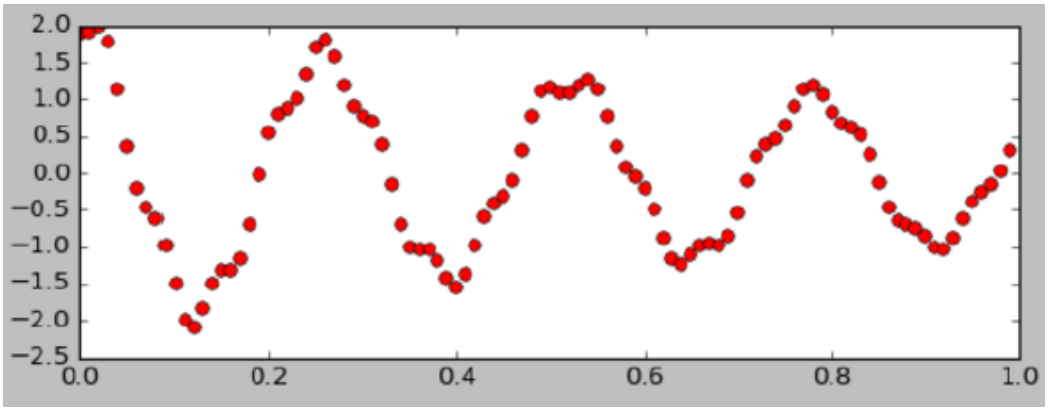
väärtused iga t jaoks vahemikus 0; 0,01…1. See tähendab, et kokku kirjeldab ühte signaali 100 punkti. Amplituudi võimalikud väärtused olid vahemikus 5 ja 10 ja sagedusel 1 kuni 5 (ühikuta). Reaalne signaal ei ole kunagi müravaba, seega lisasin ka selle. Esialgu katsetasin valge müraga, hiljem liitsin kõrge sagedusega siinuse ja kosiinuse summa

$$h(t) = Be^{-t}\sin(2\pi f_B t) + Ce^{-t}\sin(2\pi f_C t), \quad (8)$$

kuid tulemusi see märgatavalt ei mõjutanud. Müra komponentide sagedused f_B ja f_C muutusid vahemikus 0 kuni 50 (ühikuta) ning amplituudid B ja C juhuslikult kuni 10% signaalist. Kõige lõpuks liitsin esialgsele funktsioonile $f(t)$ ja mürale $h(t)$ juurde funktsiooni, mille kuju sõltub amplituudist mittelineaarselt. Sellise tingimuse põhjuseks on reaalse südamesignaali kuju mittelineaarne sõltuvus asukohast. Nimetatud kaalutlulustel valisin signaali kuju muutvaks funktsiooniks

$$g(t) = De^{-t}\cos(2\pi\sqrt{D}t), \quad (9)$$

kus D muutub vahemikus 0 kuni 5. Selliselt saadakse mitmest komponendist koosnev signaal mille ühks näide on kujutatud allpool oleval joonisel.



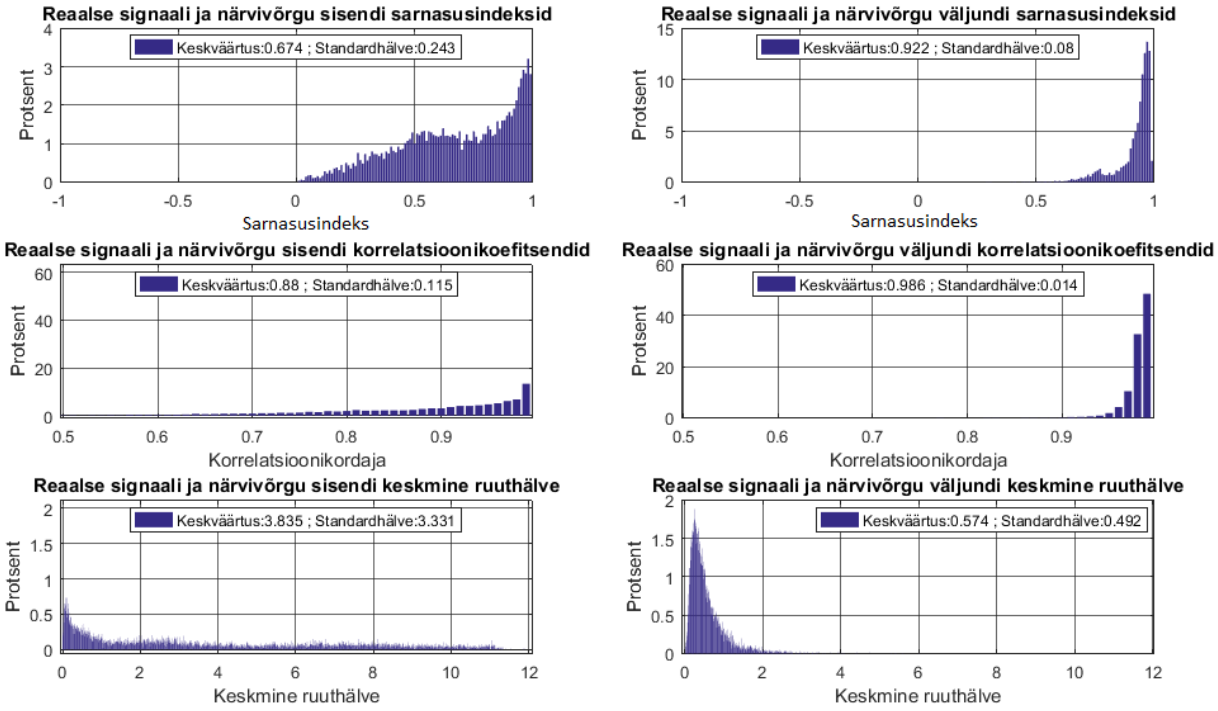
Joonis 12. Näide genereeritud signaalist.

Järgmiseks sammuks oli närvivõrgu loomine. Lihtsuse mõttes kasutati tüüpilist 3-kihilist võrku, milles on üks sisendkiht, üks varjatud kiht ning üks väljundkiht. Kihid on omavahel täielikult ühendatud ning varjatud kihi käivitusfunktsioonina kasutati sigmoidfunktsiooni. Genereeritud signaali kirjeldas 100 andmepunkti, mistõttu on sisend- ja väljundkihis samuti 100 neuronit. Varjatud kihi neuronite arv määrati proovimise teel. Jälgides väljundi paranemise suunda, valiti lõpuks varjatud kihti 75 neuronit.

Närvivõrgu treenimiseks kasutati tüüpilist vea tagasilevi meetodit (vt pt 2.3). Kokku kasutas närvivõrk treenimiseks 500000 andmepunktide komplekti (vt Joonis 12) ning läbis 40 iteratsiooni. Ühe iteratsiooni all on mõeldud kõikide treeningandmete ühekordne edasi ja tagasi arvutamine läbi närvivõrgu. Võrgu sisendiks oli seega funktsioon $f(t) + h(t) + g(t)$ ning väljundiks pidi tulema kõver, mis oleks võimalikult sarnane funktsiooniga $f(t)$. Treenimise järgselt testiti närvivõrgu tulemusi 10000 test-andmete komplekti peal.

Test-andmete tulemustest arvutati keskmine sarnasusindeksi, korrelatsioonikordaja, keskmine ruuthälve ning kõikidel juhtudel ka standardhälbed (vt Joonis 13). Arvutustehnilise vahendina kasutasin selleks katseks Python 2.7 teeki PyBrain (vt Lisa 6).

Genereeritud signaalide töötlemisel muutus sarnasusindeksi keskväärtus 0,674-lt 0,922-le, korrelatsioonikoefitsendi keskväärtus 0,88-lt 0,986le ning keskmine ruuthälve 3,835-lt 0,574-le. Vastavad standardhälbed kahanesid 0,243-lt 0,08-le, 0,155-lt 0,014-le ja 3,331-lt 0,492-le (vt Joonis 13).



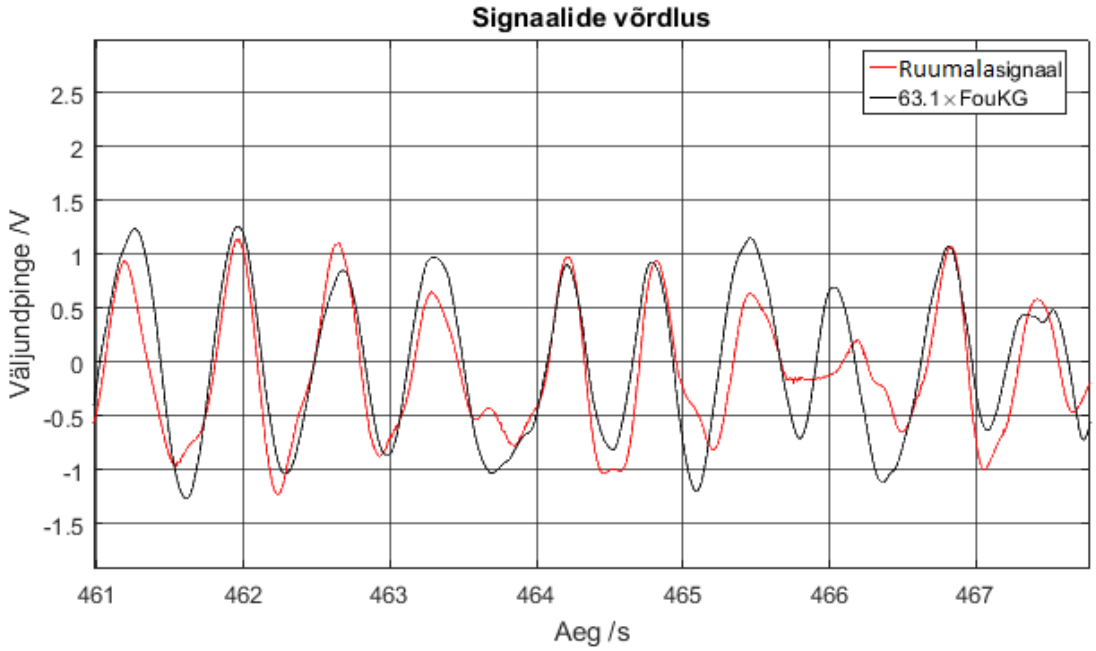
Joonis 13. 10 000 simuleeritud kõvera sarnasusindeksite, korrelatsioonikordajate ja keskmise ruuthälbe protsentuaalsed jaotused (iga sammas vastab vahemikule 0,01) enne ja pärast närvivõrgu rakendamist.

3.3 Tehisnärvivõrgu rakendamine fantoomil

Järgenvalt tehti närvivõrkude treenimiseks vajalikud mõõtmisseeriad. Signaalide tekitamisel peeti silmas, et need oleksid võimalikult südame kontraktsiooniliikumise sarnased, kuid samas ebakorrapärased, et näha, kas meetod töötab ka komplitseeritud olukorras. Selgus, et vajaliku kujuga liikumisi on sellise süsteemiga raske tekitada. Seetõttu on tekitatud rõhulained küll südame tööga sarnase sagedusega, kuid erineva kujuga. Selline asjaolu ei ole takistuseks närvivõrkude rakendamiseks.

Järgmine probleem oli andmete esitamine närvivõrgule - millise pikkusega signaalitükid on kõige optimaalsem närvivõrkude treenimiseks esitada, kas ja kuidas enne treenimist peab sisendit mingil muul moel töötlema. Esimene küsimus sai lahendatud proovimise teel. Kõige sobivamaks osutusid lühikesed, 100 ms kuni 200 ms pikkused signaalilõigud. See tähendab, et kui diskreetimissagedus on 250 Hz, siis signaalilõigud koosnesid 25 kuni 50 punktist. Treenimiseks mõeldud signaali kogukestvus on ligikaudu üks tund.

Närvivõrguga paremate tulemuste saamiseks prooviti signaalilõigud enne ära normeerida, kuid see omas vastupidist efekti. Normeeritavad signaalilõigud peavad olema paraja pikkusega, sest ruumalasignaali keskväärtus muutus pidevalt, sõltuvalt kolvi algkõrgusest. Erinevalt normeeritud lõigud ei anna tagasi kokku pannes enam õiget signaali kuju. Lisaks ei pruugi normeerimisjärgsed lõigud enam kokku sobida, mis tekitab signaalis katkevusi. Sisendi töötlusena kasutati ainult silumist. Allpool oleval pildil on näide registreeritud FouKG signaalist ning vastavast ruumalakõverast.

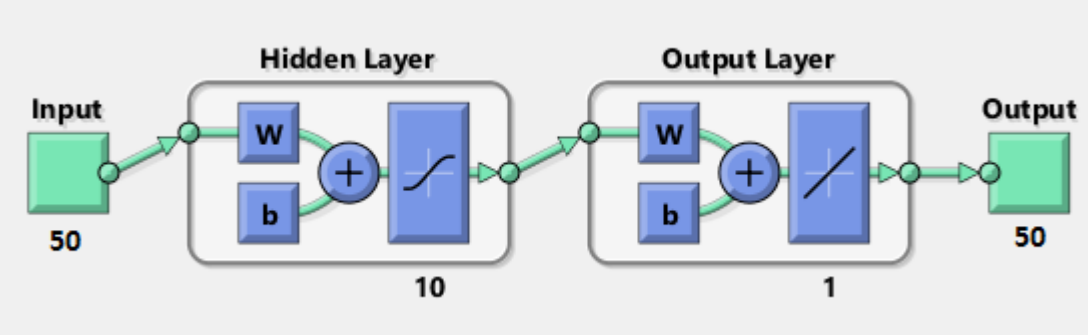


Joonis 14. Näide fantoomil registreeritud Foucault' kardiogrammist ning vastavast ruumala signaalist.

Järgmiseks etapiks oli närvivõrgu treenimine. Esimese närvivõrgu treenimiseks kasutasin sarnast Pythoni skripti, millega tehti katsed simuleeritud kõveratega (vt Lisa 7). Kuni siiani olid kõik närvivõrgud treenitud suvaline arv iteratsioone. Peagi selgus, et iteratsioonide arvu on võimalik lihtsalt optimeerida. Selle kõige tavalisem variant on kasutada valideerimisandmeid. Valideerimisandmed on sarnaselt treeningandmetele komplektid teadaolevatest sisenditest ja väljundites. Ainukene erinevus on see, et treenimise käigus võrreldakse iga iteratsiooni järel valideerimisandmete põhjal arvutatud närvivõrgu väljundi ja etaloni vahelist keskmist ruuthälvet. Kui keskmine ruuthälve peale mõnda iteratsiooni ei ole rohkem langenud, siis treenimine peatub. Selline optimeerimismeetodit tuntakse ristvalideerimise nime all ja on

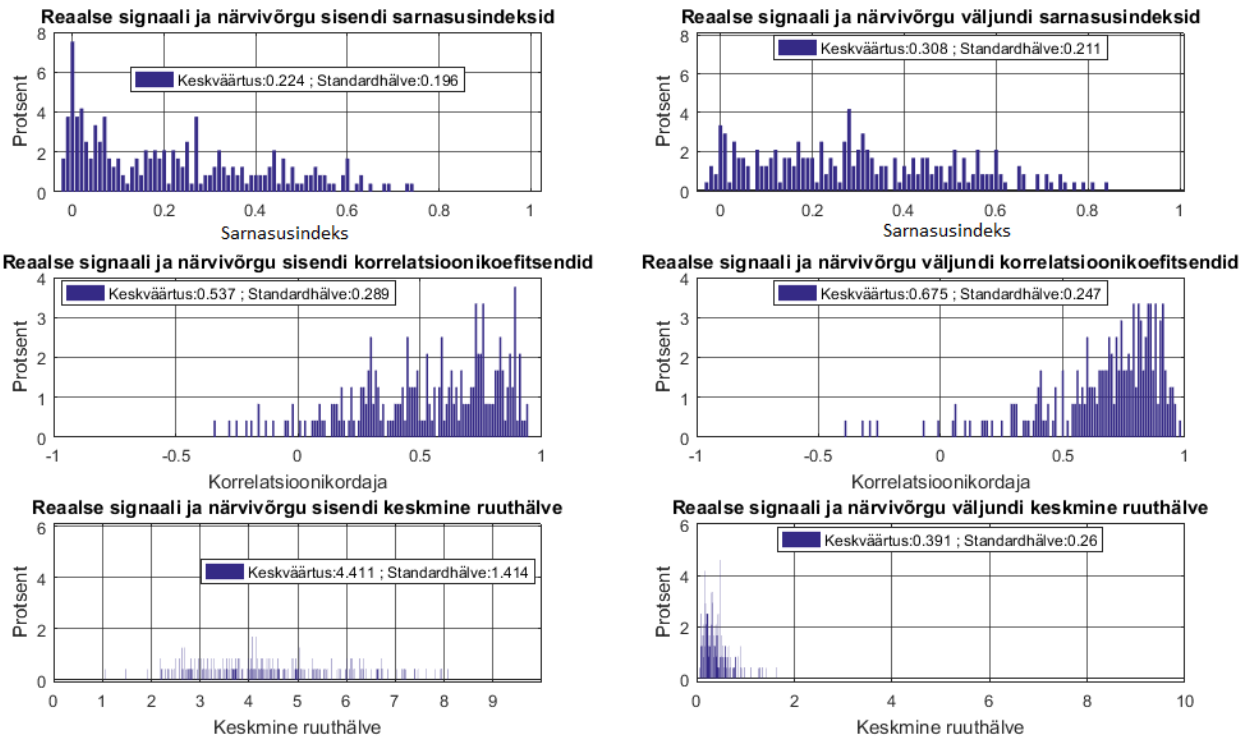
oluline, et vältida närvivõrgu ületreenimist. Vastasel juhul tulemuste kvaliteet langeb. Uurides võimalusi, kuidas optimeerimist koodis implementeerida, leiti, et sellised funktsioonid on MATLAB *Neural Networking Toolbox*-i juba sisse ehitatud. Seetõttu kasutati ka edaspidi ainult MATLAB-i (vt Lisa 8).

Lõpptulemusena oli andmetena kokku 15943 signaalilõiku, igas lõigus 50 andmepunkti. Üks lõik oli seega kestvusega 200 ms ning signaali kogukestvus 53 minutit. Kogusignaalist 90% kasutati närvivõrgu treenimiseks ning viimased 10% testimiseks. Närvivõrk oli ühe varjatud kihiga, milles oli 10 neuronit, aktiveerimisfunktsiooniks sigmoidfunktsioon (vt Joonis 15).



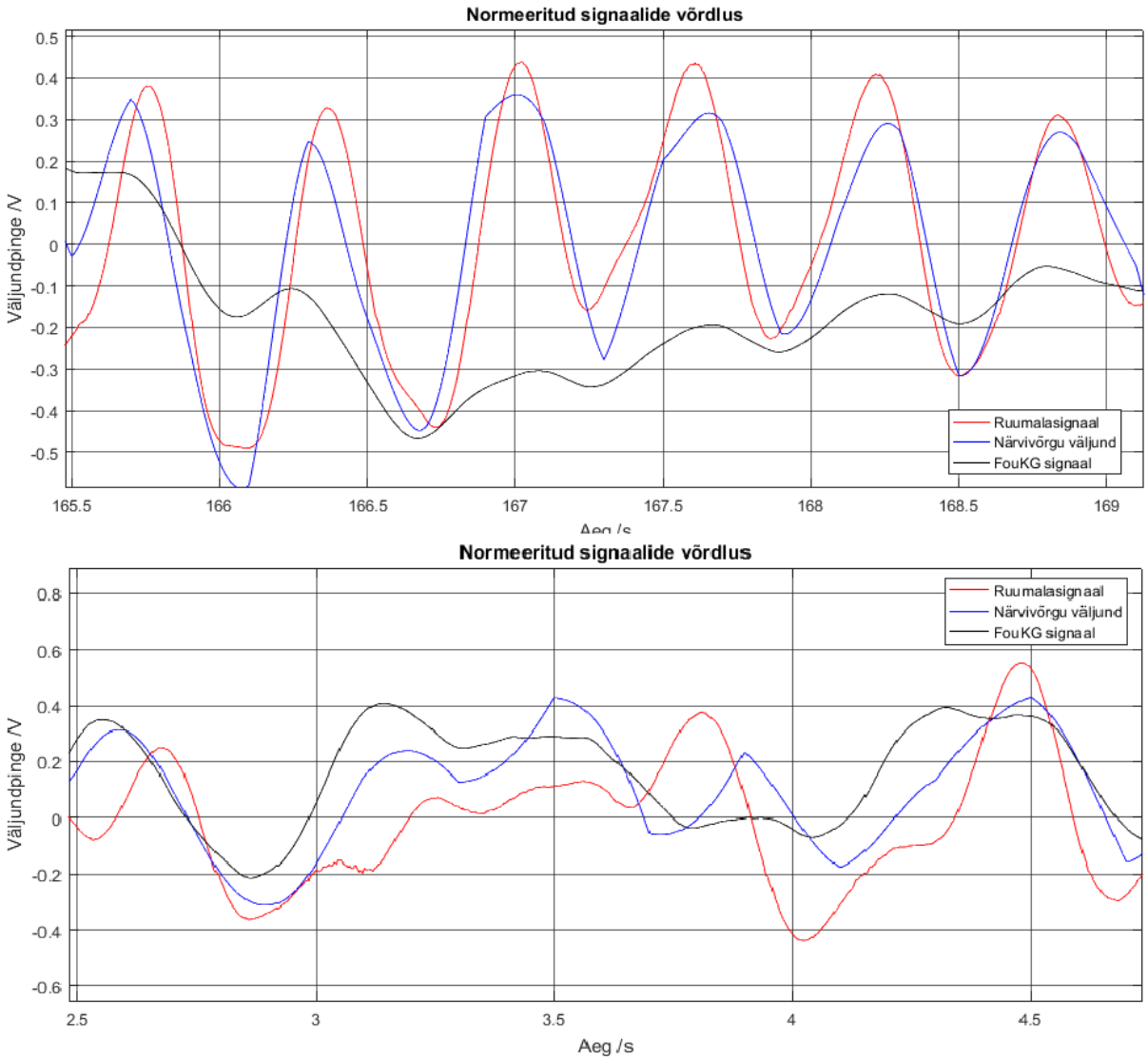
Joonis 15. Kasutatud närvivõrgu plokkskeem.

Närvivõrgu väljund-lõikude kokku panemisel saadakse väljundsignaal (vt Joonis 17). Sarnasuste hindamiseks oli jaotuse tekkimise eesmärgil väljundsignaal lõigatud 2 sekundilisteks lõikudeks (ligikaudu 2 südametsükli perioodi). Joonis 16 kuni Joonis 19 kujutatavad närvivõrgu test-andmete sisendite (FouKG signaal) ja väljundite sarnasuste võrdlusi etaloniga (ruumalakõver).

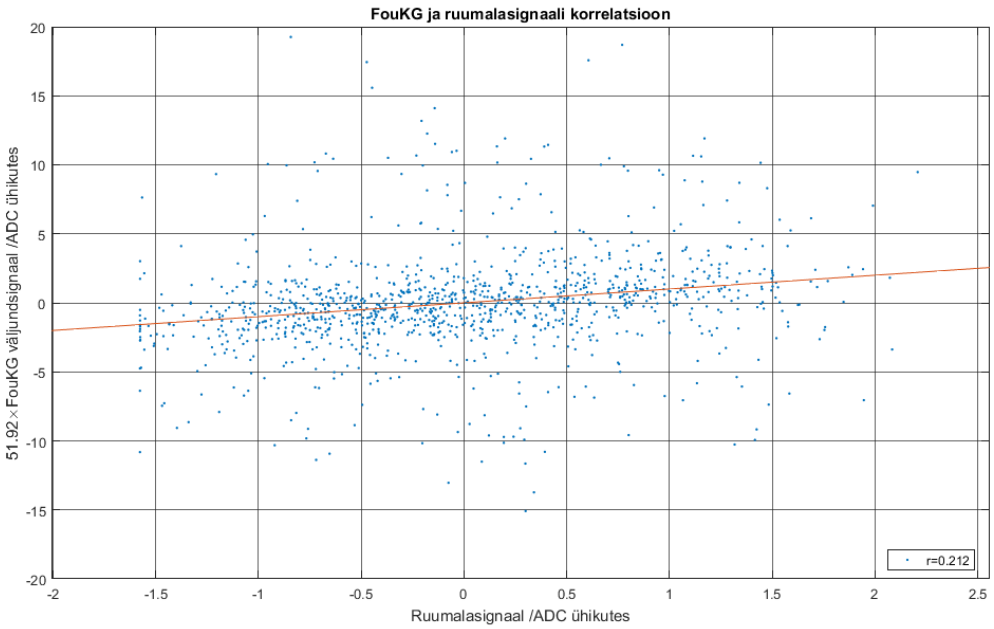


Joonis 16. 239 2-sekundi pikkuste signaalilõikude sarnasusindeksite, korrelatsioonikordajate ja keskmise ruuthälbe protsentuaalsed jaotused (iga sammas vastab vahemikule 0,01) enne ja pärast närvivõrgu rakendamist.

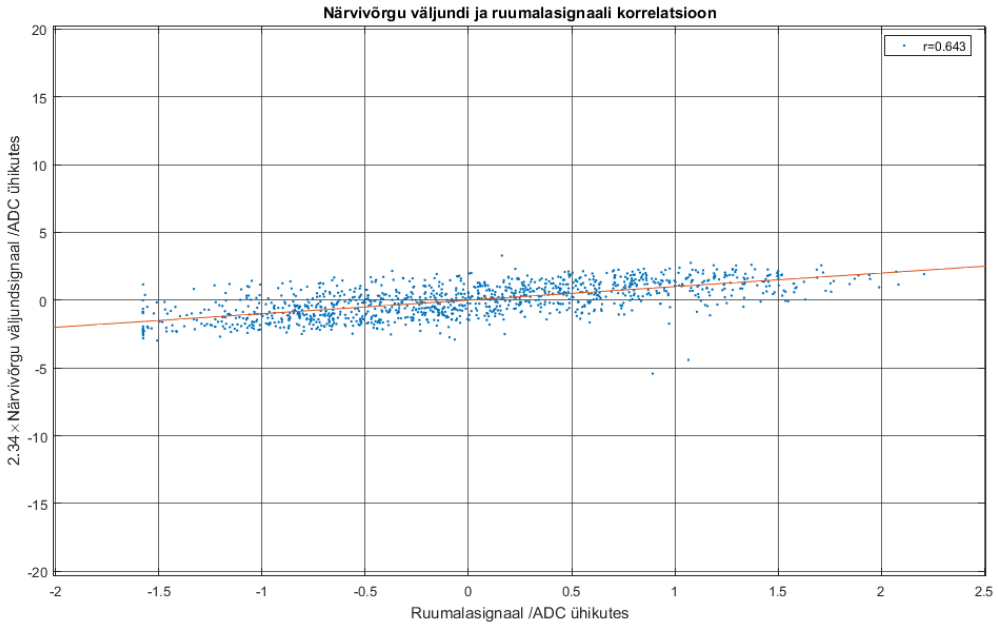
Sarnasusindeksi keskväärtus tõusis 0,224-lt 0,308-le, mis tähendab tulemuse 0,084 võrra paranemist. Standardhälve see-eest suurenes 0,015 - 0,196-lt 0,211-le. Korrelatsioonikordajate keskväärtus kasvas 0,537-lt 0,675-le, suurenedes seega 0,138. Samuti kahanes korrelatsioonikordajate standardhälve 0,042 - 0,289-lt 0,247-le. Suuremad muutused tekkisid lõikude keskmine ruuthälve, mille keskväärtus langes 4,411-lt 0,391-le ja standardhälve 1, 414-lt 0,26-le (Joonis 16). Ruumalasignaali korrelatsioon FouKG signaaliga oli 0,212 (Joonis 18) ja närvivõrgu väljundiga 0,643 (vt Joonis 19). Närvivõrgu väljundlõigud kokku komplekteerides saadi väljundsignaal, mida võrreldi FouKG- ja ruumalasignaaliga (Joonis 17). FouKG ja närvivõrgu väljundsignaali spektreid võrreldi ruumalasignaali spektriga (Joonis 20).



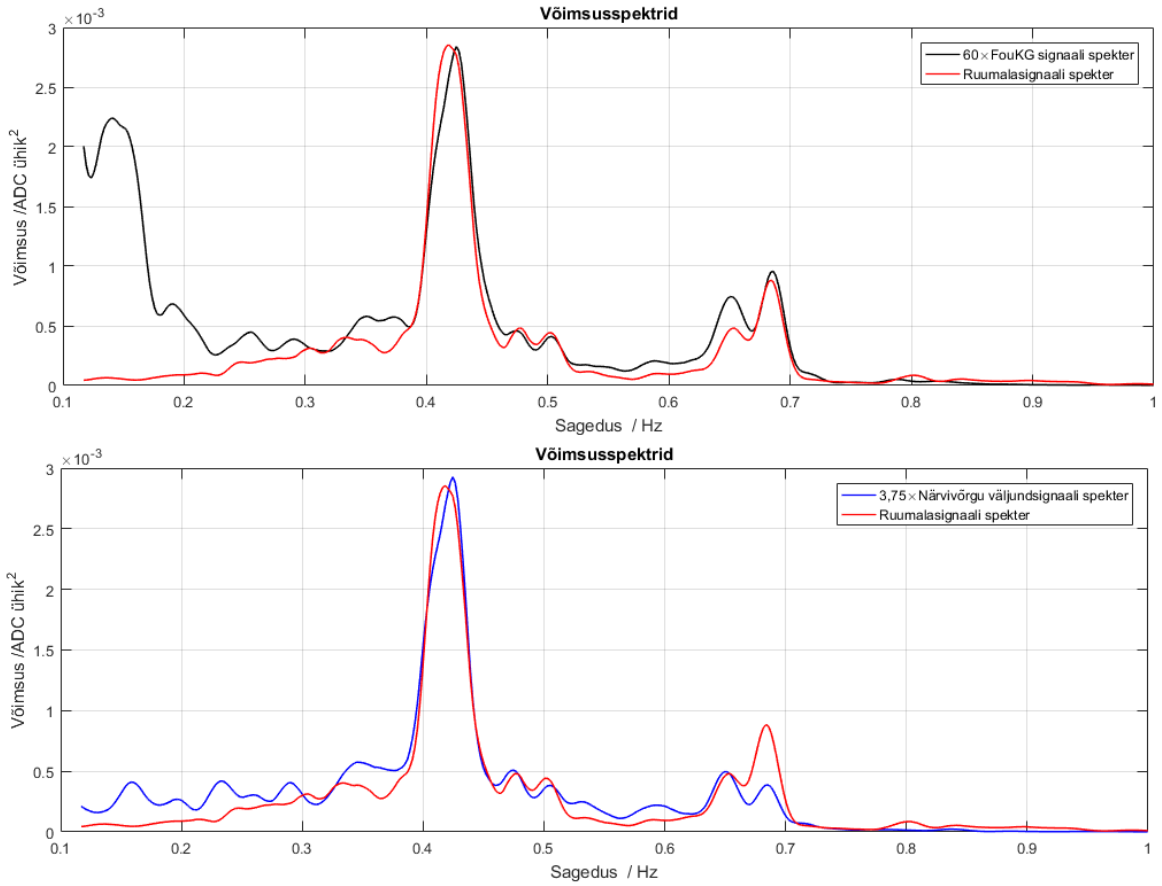
Joonis 17. Näited samaaegsetest ruumala-, FouKG ja närvivõrgu väljund signaalidest. Üleval näide tüüpilisest signaalist, all ebakorrapärane signaalilõik.



Joonis 18. Testsignaali regressioon enne närvivõrgu kasutamist. Punane joon tähistab ideaalset kokkulangevust ($y = x$). Andmepunktide suure arvu tõttu on neist kuvatud iga sajas, kokku 1195 tk. Kõikide andmepunktide pealt arvutatud $r = 0{,}212$. Detailsema pildi jaoks vt Lisa 3.



Joonis 19. Testsignaali regressioon pärast närvivõrgu kasutamist. Punane joon tähistab ideaalset kokkulangevust ($y = x$). Andmepunktide suure arvu tõttu on neist kuvatud iga sajas, kokku 1195 tk. Kõikide andmepunktide pealt arvutatud $r = 0{,}643$. Detailsema pildi jaoks vt Lisa 4.



Joonis 20. Joonise ülemisel poolel FouKG ja ruumalasignaali spektrite võrdlus. Alumisel poolel närvivõrgu väljundsignaali ja ruumalasignaali spekter

4. Analüüs

Joonis 10 kujutab katses kasutatud vedelike eritakistuse sõltuvus sagedusest. On näha, et sageduse edasisel tõusmisel kahaneks eritakistus vähe, mis tähendab, et ka sondeerival sagedusel (8 MHz) on eritakistused lähedased graafikul juba nähtavatele asümptootidele. Seda kinnitab ka asjaolu, et inimeselt ja fantoomilt registreeritud FouKG signaalid on sarnases suurusjärgus (vt Joonis 11).

Jooniselt 16 on näha, et närvivõrgu kasutamine parandas signaalilõikude sarnasustegurite ja korrelatsioonikordajate keskväärtust ning viimasel juhul vähendas ka standardhälvet. Testsignaali korrelatsioon ruumalasignaaliga oli pärast närvivõrguga töötlemist märkimisväärselt suurem (vt Joonis 18 ja Joonis 19). Nende sarnasushinnangute põhjal võib järeldada, et närvivõrgu kasutamine oli põhjendatud ning tulemusele positiivne.

Tehisnärvivõrgu positiivne mõju avaldub lainekujude sarnasuse suurendamises, faasinihke vähendamises ja signaalide keskväärtuste ühildamises (vt Joonis 17 ülemine signaalilõik). Ebakorrapäraste signaalilõikude puhul ei ole kokkulangevus märkimisväärselt parem esialgsest FouKG signaalist (vt Joonis 17 alumine signaalilõik). Võimsusspektrite (Joonis 20) võrdlusest järeldub, et närvivõrk on FouKG signaalist märkimisväärselt vähendanud madalate sageduste osakaalu ning suurendanud sarnasust ruumalasignaali spektriga. See tähendab, et järgnevates uurimustes peab eelnevalt närvivõrgu treenimisele FouKG signaali töötlema kõrgpääs filtriga. Eelduste kohaselt suudab närvivõrk peale madalate sageduste välja filtreerimist sisend- ja etalonsignaali spektrite sarnasusi suurendada ka kõrgemas sagedusdiapasoonis.

Tõenäoliselt on võimalik tulemusi andmemahtu ja närvivõrgu optimeerimise teel tugevasti parandada. Kui meetod töötab selliste, ebakorrapäraste (suur lainekujude varieeruvus) signaalide sarnasushinnangute parandamisel, võib eeldada, see töötab ka inimeselt registreeritud FouKG ja südameruumala signaalide sarnasuse suurendamisel. Ainukeseks probleemiks on see, et pole olemas meetodit, millega võrdlussignaale registreerida. Sellest hoolimata on saadud tulemused kasulikud edaspidises Foucault' kardiograafia arendustöös [23].

Üks võimalik järgnev katse on modelleerida fantoomist arvutisimulatsioon, st jäljendada reaalset objekti arvutuslike meetoditega, ja kasutada arvutatud andmeid närvivõrgu

treenimiseks. Seejärel uurida, kas simulatsioonide peal treenitud närvivõrk suudab reaalseid signaale parandada. Edasises arenduses saab närvivõrkude treenimiseks kasutada inimese rindkere simulatsioone [23].

Üheks püstitatud ülesandeks oli hinnata FouKG suutlikust fantoomis tekitatud ruumala muutmise registreerimisel. Selgus, et töös valminud fantoomil on selle ülesande täitmiseks mõned puudused. Ruumalavõnkumiste tekitamine põhjustas südameballooni pendeldamist. Nimetatud liikumine avaldus küll FouKG, kuid mitte ruumalasignaalis. Ruumalasignaali tekkimisel oli oluline roll Marey kapslil. Liiga peenikeste ühenduste tõttu paistis torustiku takistus signaali liigselt siluvat, mistõttu puudusid ruumalakõverast kõrgema sagedusega komponendid. Kui kolb (vt Joonis 21) alla suruda ja seejärel fikseerida, jääb vedelik kapslis oleva membraani elastsuse tõttu membraani ja ületõusutoru vahele pendeldama. See tekitab omakorda ruumalasignaalis lisakomponente, mis ei ole südameballooniga seotud. Nimetatud probleemide tõttu ei olnud eelmainitud ülesande täitmiseks fantoomi käitumine piisavalt hästi kontrollitav. Sellest tulenevalt on FouKG ja ruumalasignaali sarnasusnäitajad suhteliselt madalad (vt Joonis 13 ja Joonis 18).

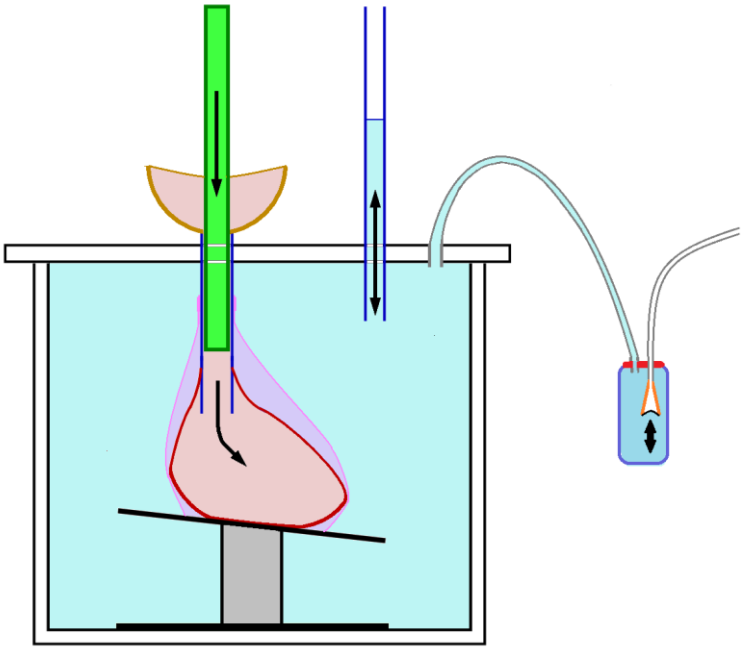
Üheks võimalikuks variandiks, kuidas Marey kapsli mõju signaalile vähendada, oleks kasutada jäigemat membraani ning paraja läbimõõduga torusid. Teine võimalus on kasutada signaalitöötlusest tuntud seoseid. Kui süsteemi väljund on $g(t)$, sisend $f(t)$ ja süsteemi impulsskoste $h(t)$, siis väljund on määratud kui

$$g(t) = h(t) * f(t), \tag{10}$$

millest konvolutsiooni teoreemi kaudu avaldub sisend

$$f(t) = \mathcal{F}^{-1}\left[\frac{\mathcal{F}[g(t)]}{\mathcal{F}[h(t)]}\right] \tag{11}$$

Registreerides süsteemi impulsskoste (reaktsioon südameballooni kiirele ja tugevale paisumisele ning sama kiirele ja tugevale taastumisele), saab väljundist arvutada südameballooni ruumalavõnkumise. Sellist võimalust ka korraks katsetati, kuid see ei paistnud esmapilgul andvat head tulemust. Lisaks muutusid süsteemi omadused igal südameballooni või kummikinnas-membraani vahetusel ning vedeliku koguse muutmisel. Nendel põhjustel sellesse meetodisse rohkem ei süvenetud. Käesoleva töö edasises arendamises tasuks kindlasti sellist parandusmeetodit rohkem katsetada – kas muudab FouKG- ja ruumalasignaali sarnasemaks. Kui muudab, võib olla sellest kasu ka närvivõrgu rakendustes.



Joonis 21. Fantoomi probleem. Kolvi liigutamisele lisandub kapsli membraani ja ületõusutoru vaheline vedeliku võnkumine.

Veel eksperimendi parandamise variante, mida võiks tulevikus FouKG- ja ruumalasignaalide võrdlemisel katsetada:

- Kasutada väiksema elastsusega südameballooni ja paremini tihendatud kolbi – vähendab eeldatavasti ballooni tuiklemist.
- Kontrollida lainekujude sarnasusi kasutades suurema perioodiga signaale. – kiired liikumised tekitavad tuiklemist ja muud müra.
- Kasutada kolvi liigutamiseks aktuaatorit - toru diameetrit teades saab ruumalamuutuse anduri vahenduseta välja arvutada.

Kokkuvõte

Närvivõrkude edu mitmetes mittelineaarse regressiooni rakendusvaldkondades on andnud alust arvata, et seda meetodit on võimalik kasutada ka Foucault' kardiograafia arendamises.

FouKG signaali on peetud kirjanduse ülevaatele tuginedes sarnaseks südame ruumalasignaalile, kuid siiani ei oldud seda katseliselt kontrollitud. Selle jaoks, et hinnata FouKG täpsust ruumalasignaali registreerimisel, valmistasime kontrollitava ruumalaga inimese südamepiirkonda imiteeriva süsteemi ehk fantoomi.

Töö käigus valmistatud fantoom täitis suuremas osas oma eesmärki ning on rakendatav ka järgmistes uurimustes. See-eest ilmnesid fantoomil ka mõned puudused, mistõttu ruumalasignaali õigsus on kaheldav. Sellest tulenevalt ei olnud võimalik FouKG- ja ruumalasignaali sarnasust adekvaatselt hinnata. Nimetatud asjaolule vaatamata sai töö alguses sõnastatud eesmärk positiivselt täidetud: närvivõrk on sobiv meetod FouKG- ja ruumalasignaali sarnasuse suurendamiseks. Saadud tulemused on paljulubavad ning neid on võimalik kasutada Foucault' kardiograafia edasisel arendamisel.

Järeldused:

- Ehitatud rindkere fantoom täitis oma eesmärki, kuid vajab täiustamist usaldusväärsemate tulemuste saamiseks.
- Ehitatud fantoom ei võimaldanud imiteerida südame ruumalalainele iseloomulike eksponentsiaalseid tõuse ja langusi (vt Lisa 5). Südame ruumalasignaalile iseloomulike detailide ja üldise lainekuju imiteerimine jääb keerulise ülesandena edaspidiseks lahendamiseks. Töös valminud fantoomilt mõõdetud FouKG ja ruumalasignaalides oli hästi korreleeritud piirkondi (Joonis 14), kuid esines ka lahknevusi (Joonis 17).
- Teadaolevad ruumala ja FouKG signaalide lahknemine oli tingitud fantoomi eripäradest. Registreeritud ruumalasignaal kujunes summana südameballooni ruumalavõnkumisest ning rõhu mõõtesüsteemi torustikus kujunenud pikivõnkumistest (pt 4, lk 31), mistõttu leidus ruumalasignaalis komponente, mida FouKG signaalis ei olnud. FouKG signaal kujunes ruumalavõnkumisest ning südameballooni pendeldamisest, mida ei tuvastatud ruumalasignaalis.

- Pilootkatsed tehisnärvivõrkudega andsid positiivse tulemuse. Närvivõrgu kasutamine suurendas oluliselt signaalide sarnasusindekseid, korrelatsioonikordajaid ning vähendas keskmist ruuthälvet (Joonis 13).
- Närvivõrgu treenimiseks vajalikud signaalid registreeriti infotehnoloogiliselt heal tasemel.
- Tehisnärvivõrk on võimekas meetod käesoleva probleemi lahendamiseks ning väärib kindlasti edaspidist arendamist.

Kasutatud kirjandus

1. W. G. Kubicek, J. N. Karnegis, R. P. Patterson, D. A. Wittsoe, and R. H. Mattson, „Development and evaluation of an impedance cardiac output system“, *Aerosp. Med*, 1966.

2. S. A. Smith, A. E. Russell, M. J. West, and J. Chalmers, „Automated non-invasive measurement of cardiac output: comparison of electrical bioimpedance and carbon dioxide rebreathing techniques“, *Br. Heart J.*, 1988.

3. G. Cybulski, A. Strasz, W. Niewiadomski, and A. Gąsiorowska, „Impedance cardiography: Recent advancements“, *Cardiol. J.*, 2012.

4. N. M. Albert, M. D. Hail, J. Li, J. B. Young, G. M. Kaufman, and L. H. Kaufman, „Equivalence of the bioimpedance and thermodilution methods in measuring cardiac output in hospitalized patients with advanced, decompensated chronic heart failure“, *Am. J. Crit. Care*, vol. 13, no. 6, Nov. 2004.

5. S. A. Kamath, M. H. Drazner, G. Tasissa, J. G. Rogers, L. W. Stevenson, and C. W. Yancy, „Correlation of Impedance Cardiography with Invasive Hemodynamic Measurements in Patients with Advanced Heart Failure: the BioImpedance CardioGraphy (BIG) Substudy of the ESCAPE Trial“.

6. R. P. Patterson, „Impedance cardiography: What is the source of the signal?“, *J. Phys.*, 2010.

7. J. Trolla, „Foucault’ kardiograafi katseeksemplari ehitamine ja uurimine“. Magistritöö, 2000.

8. V. Zadin, „Foucault’ kardiogrammi tekke uuringud südant sondeerivate pöörisvoolude arvutamise abil“. Magistritöö, 2008.

9. „Clinical Applications“, *NIMedical*. [Online]. Available: http://www.ni-medical.com/clinical-applications/. [Accessed: 14-May-2017].

10. *Jüri Vedru suuline konsultatsioon*. 2017.

11. K. McCaney, „Body sensors could improve future soldiers“ performance“, *Def. Syst.*

12. M. Ross, „Changes in Cardiac Output During Exercise“, *LIVESTRONG.COM*. [Online]. Available: http://www.livestrong.com/article/307554-changes-in-cardiac-output-during-exercise/. [Accessed: 03-May-2017].

13. J. Vedru and L.-H. Humal, „Physiological Measurement Based on Foucault Principle: Set-up of the Problem“, 1996.

14. M. H. Calvert and A. G. Heaton, „Stroke volume measurement in animals with pacemaker control of heart rate“, 1973.

15. V. Krasnopolsky, „Elements of Nonlinear Statistics and Neural Networks“, 25-Apr-2006.

16. N. Ganesan, K. Venkatesh, M. A. Rama, and A. Malathi Palani, „Application of Neural Networks in Diagnosing Cancer Disease Using Demographic Data“, *Int. J. Comput. Appl.*, 2010.

17. L. Bottaci *et al.*, „Artificial neural networks applied to outcome prediction for colorectal cancer patients in separate institutions“, *The Lancet*, 1997.

18. E. Petlenkov, „Tehisnärvivõrgud ja nende rakendused“. TTÜ Automaatikainstituut, 2004.

19. J. Vedru, Rauno Gordon, Leo Henn Humal, and Jaanus Trolla, „Modelling of an inductive sensor of the Foucault cardiogram“, *Est. J. Eng.*, 2011.

20. J. Vedru, O. Makarova, and V. Tina, „Averaged Waveforms of Electrical Bioimpedance: Construction and Comparion“, 2001.

21. G. Savustjan, „Comparison of Foucault' cardiogra with MRI venticular volume-time curve“. Master“s Thesis, 2010.

22. H. N. Koivo, „Neural Networks: Basics using Matlab Neural Network Toolbox“. 2008.

23. *Vahur Zadini suuline konsultatsioon*. 2016.

24. „resistivity_nacl.jpg (536×331)“. [Online]. Available: https://gasstationwithoutpumps.files.wordpress.com/2012/08/resistivity_nacl.jpg?w=536&h=331. [Accessed: 23-Apr-2017].

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Rando Avarmaa,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

   Tehisnärvivõrkude rakendatavuse hindamine FouKG signaali töötlemiseks,

   mille juhendajad on Jüri Vedru, bioloogia knd ja Vahur Zadin, PhD,

   (a) reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

   (b) üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace'i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

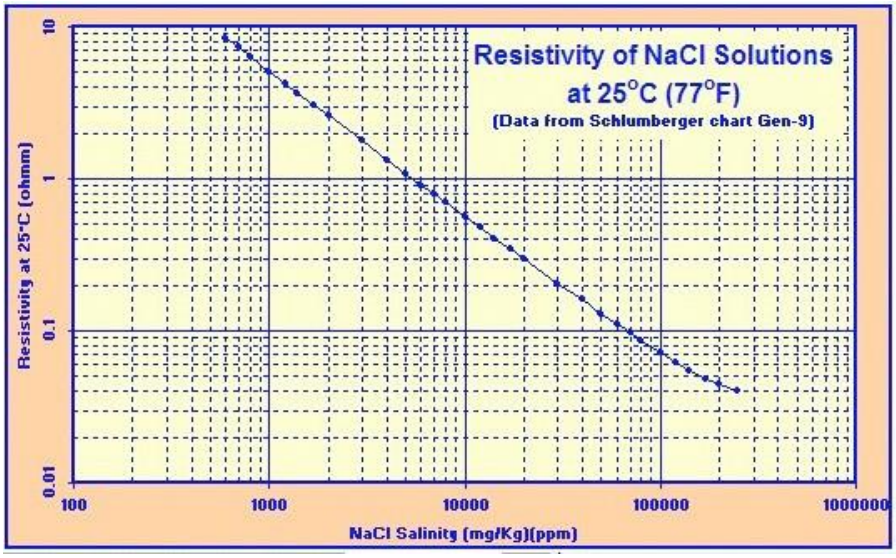
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartu, 26. mai 2017. a.

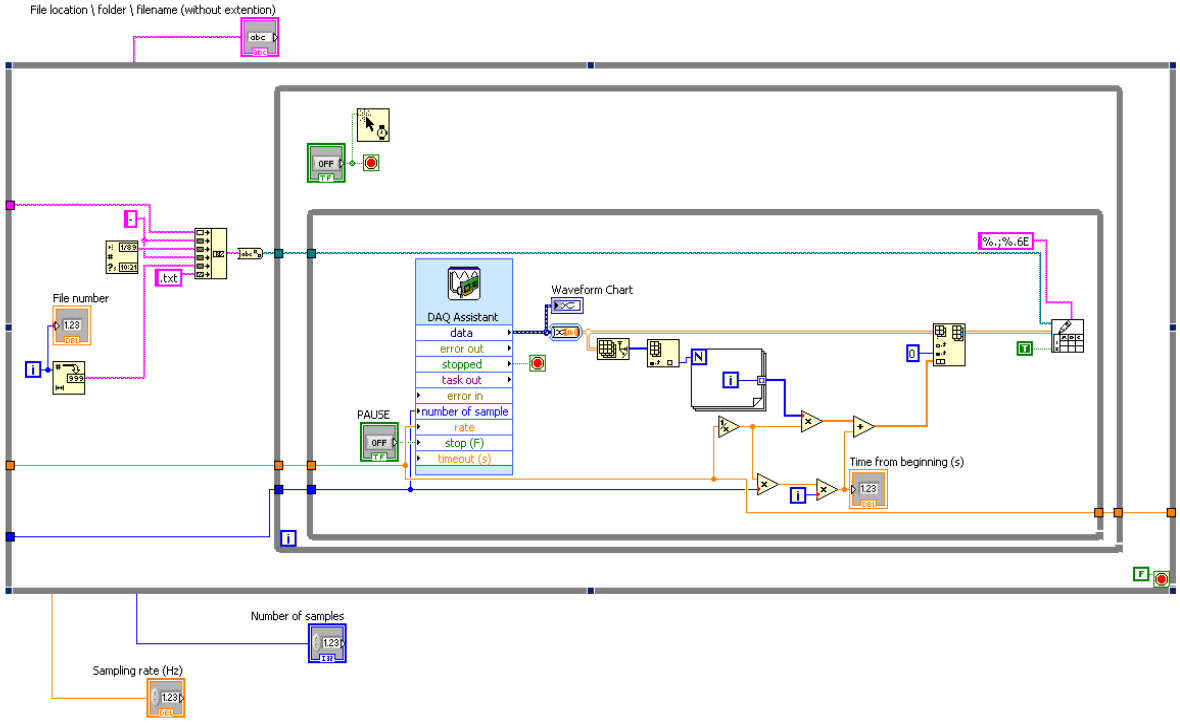
Lisad

Lisa 1



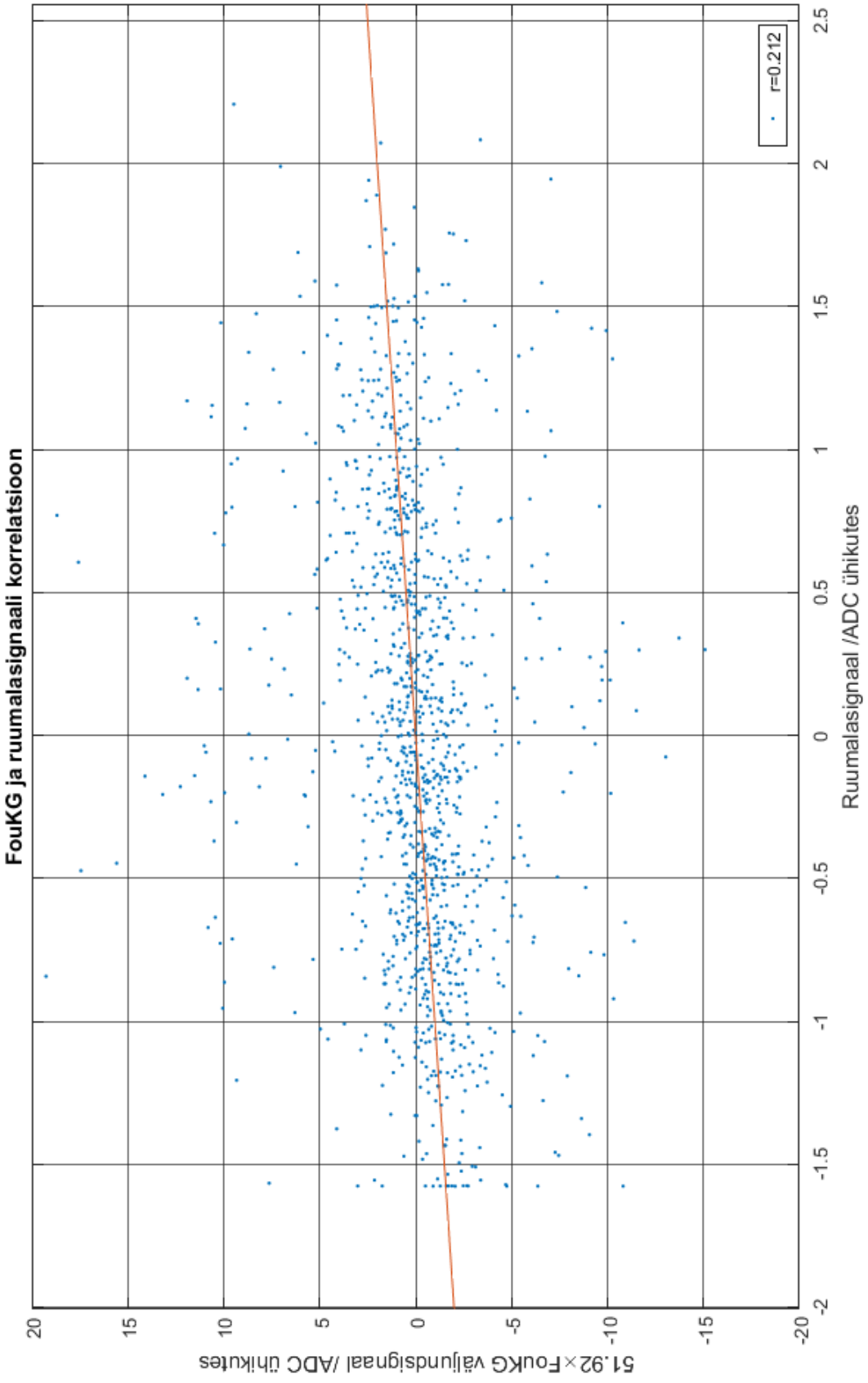
Lisa 1. NaCl lahuse juhtivuse sõltuvus konsentratsioonist. Joonis saadud veebist [24].

Lisa 2



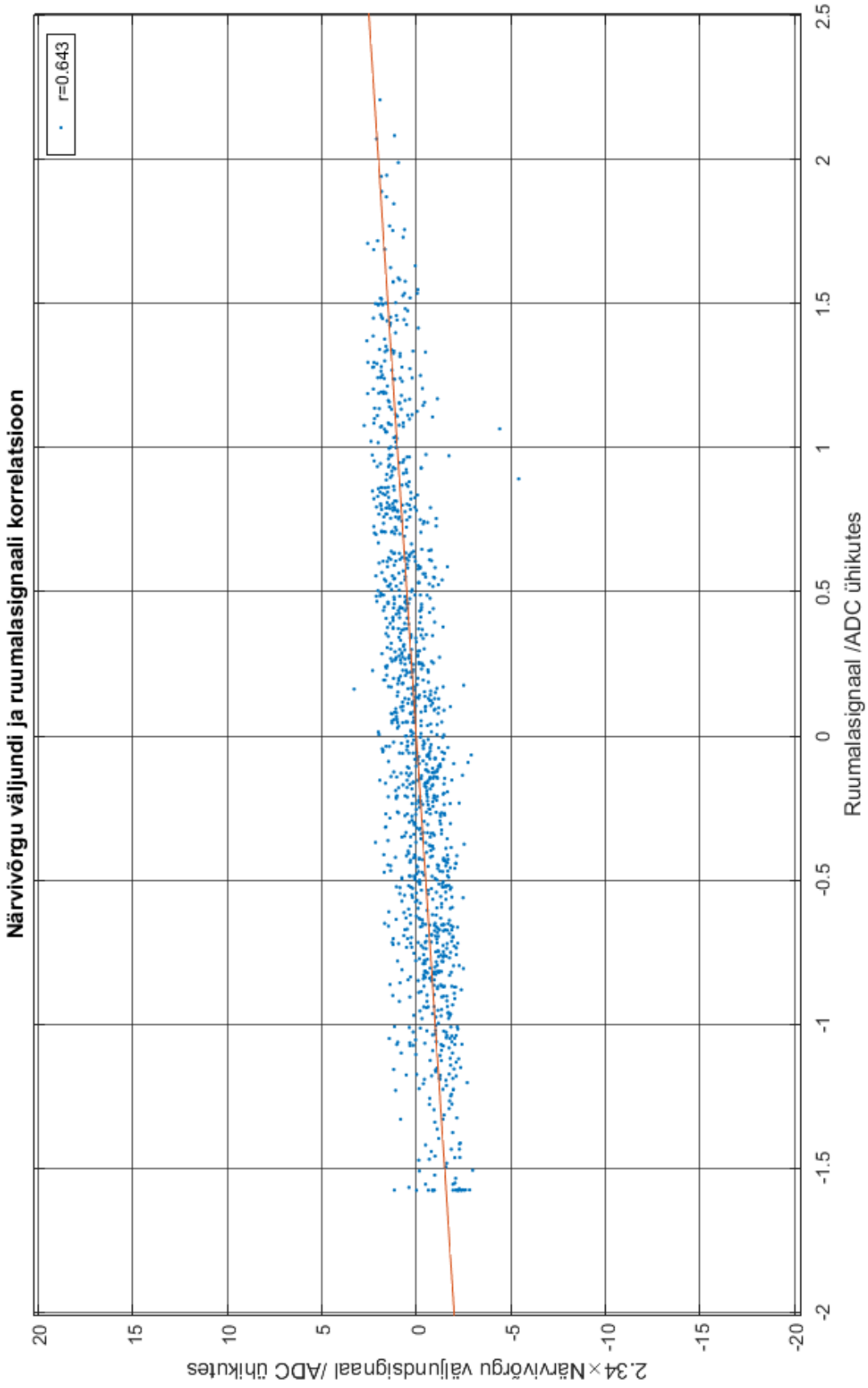
Lisa 2. Katseandmete kuvamiseks ning salvestamiseks kasutatud Labview skeem.

Lisa 3



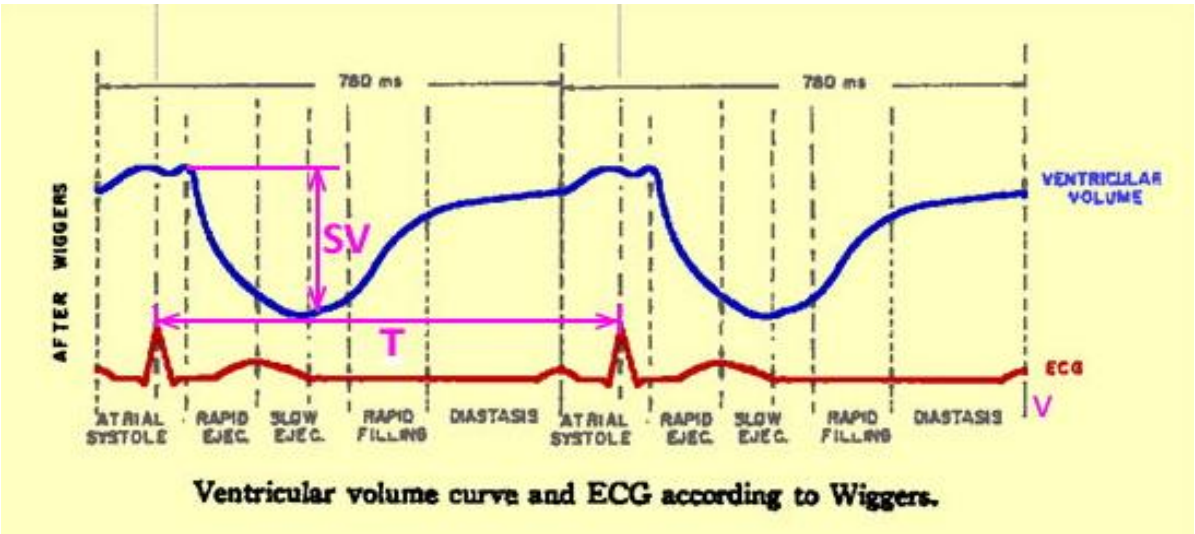
Lisa 3. Joonis 18 suurendatud kujul.

Lisa 4



Lisa 4. Joonis 19 suurendatud kujul.

Lisa 5



Lisa 5. Sinise joonega kujutatud inimese südame ruumalasignaali.

Lisa 6

```
################################################################################
#This code generates data for the network. Fixed training and testing data
#allow better network optimization, since I can compare success rates with
#different parameters for the same data.
import numpy as np
import random as rnd
import timeit
start_time = timeit.default_timer()

#################################################################
#Returns y values of damped cosine and sine function.
def f(t, f, A):
    return A * np.exp(-t) * np.cos(2*np.pi*f*t)

def f2(t, f, A): #Used for noise
    return A * np.exp(-t) * np.sin(2*np.pi*f*t)
```

```
def g(t, f, A): #Used as the modifying function
    return A*np.exp(-0.1*t) * np.sin(2*np.pi*np.sqrt(f)*t)

#####################################################################
#Generate the training data
#Define the datapoints of time axis.
t = np.arange(0.0, 1.0, 0.01)

data = open("Training_data.txt", "w")
answ = open("Training_answ.txt", "w")
#Create 50000 training data.
x = 0.1 #Max noise amplitude(0 to 1 from signal amplitude, meaning 0.1 is 10%)
for i in range(500000):
    freq = round(4*rnd.random(),2) + 1 #frequency between 1 and 5
    amp = round(5*rnd.random(),2) + 5 #amplitude between 5 and 10
    dist = round(50*rnd.random(),2) #distance between 0 and 50
    temp_ran = []
    temp = []
    #Random frequencies for the noise.
    sine_freq = 50*rnd.random()
    cosine_freq = 50*rnd.random()

    #Define random noise amplitudes, max 10% of amplitude.
    if rnd.random() > 0.5:
        A = -x*amp*rnd.random() #Noise will be subtracted.
    else:
        A = x*amp*rnd.random() #Noise will be added.

    if rnd.random() > 0.5:
```

```
        B = -x*amp*rnd.random() #Noise will be subtracted.
    else:
        B = x*amp*rnd.random() #Noise will be added.

    for j in range(len(t)):
        #Noise is a sum of sine and cosine
        sine = f(t[j], sine_freq, A)
        cosine = f2(t[j], cosine_freq, B)
        noise = sine + cosine
        #Random noise commented out.
        #noise = sign*(amp / 10)* rnd.random()#max 10% of amplitude
        temp_ran.append(round(f(t[j], freq, amp)+ g(t[j], dist/10, dist/10) + noise, 2))
        temp.append(round(f(t[j], freq, amp), 2))
    for i in range(len(temp_ran)):
        if i == len(temp_ran)-1:
            data.write(str(temp_ran[i]))
            answ.write(str(temp[i]))
        else:
            data.write(str(temp_ran[i])+",")
            answ.write(str(temp[i])+",")
    data.write("\n")
    answ.write("\n")

data.close()
answ.close()

######################################################################
#Generate the test data.
test = open("Test_data.txt", "w")
test_answ = open("Test_answ.txt", "w")
```

```
#Create 10000 test data.
for i in range(10000):
    freq = round(4*rnd.random(),2) + 1 #frequency between 1 and 5
    amp = round(5*rnd.random(),2) + 5 #amplitude between 5 and 10
    dist = round(50*rnd.random(),2) #distance between 0 and 50

    temp_ran = []
    temp = []
    #Random frequencies for the noise.
    sine_freq = 50*rnd.random()
    cosine_freq = 50*rnd.random()

    #Define random noise amplitudes, max 10% of amplitude.
    if rnd.random() > 0.5:
        A = -x*amp*rnd.random() #Noise will be subtracted.
    else:
        A = x*amp*rnd.random() #Noise will be added.

    if rnd.random() > 0.5:
        B = -x*amp*rnd.random() #Noise will be subtracted.
    else:
        B = x*amp*rnd.random() #Noise will be added.
    for j in range(len(t)):
        #Noise is a sum of sine and cosine
        sine = f(t[j], sine_freq, A)
        cosine = f2(t[j], cosine_freq, B)
        noise = sine + cosine
        #White noise commented out.
        #noise = sign*(amp / 10)* rnd.random()#max 10% of amplitude
        temp_ran.append(round(f(t[j], freq, amp)+ g(t[j], dist/10, dist/10) + noise, 2))
```

```
        temp.append(round(f(t[j], freq, amp), 2))
    for i in range(len(temp_ran)):
        if i == len(temp_ran)-1:
            test.write(str(temp_ran[i]))
            test_answ.write(str(temp[i]))
        else:
            test.write(str(temp_ran[i])+",")
            test_answ.write(str(temp[i])+",")
    test.write("\n")
    test_answ.write("\n")

test.close()
test_answ.close()
########################################################################

elapsed = timeit.default_timer() - start_time
print("Elapsed time: "+str(round(elapsed,2))+"s")

#Defining and training the network
########################################################################
from pybrain.tools.shortcuts import buildNetwork
from pybrain.datasets import SupervisedDataSet
from pybrain.supervised.trainers import BackpropTrainer
from pybrain.tools.customxml import NetworkWriter
import timeit
start_time = timeit.default_timer()
########################################################################
#Create a network with 100 input neurons, 75 hidden neurons
#and 100 output neurons.
net = buildNetwork(100, 75, 100)
```

```
#Define the dimentsions of the dataset.
ds = SupervisedDataSet(100, 100)
#########################################################################
#Read in generated training data. Amplitude is between 5 and 10, frequency between 1
#and 5.
inpt = open("Training_data.txt")
answ = open("Training_answ.txt")
data_list = inpt.readlines()
answ_list = answ.readlines()

for i in range(len(data_list)):
    data = data_list[i].strip('\n').split(',')
    answer = answ_list[i].strip('\n').split(',')
    ds.addSample(data, answer)
#########################################################################
#Link the dataset with the created network, and train it.
trainer = BackpropTrainer(net, ds)
trainer.trainEpochs(40)

#Save the trained network for future use.
NetworkWriter.writeToFile(net, 'network.xml')
#########################################################################

elapsed = timeit.default_timer() - start_time
print("Elapsed time: "+str(round(elapsed,2))+"s")

#Testing the network
#########################################################################
from pybrain.tools.customxml import NetworkReader
import matplotlib.pyplot as plt
```

```
import numpy as np
import timeit

data = []
start_time = timeit.default_timer()
#####################################################################
#Load in pretrained network
net = NetworkReader.readFrom('network.xml')
#####################################################################
#Net testing
inpt = open("Test_data.txt")
answ = open("Test_answ.txt")
netw = open("Net_answ.txt",'w')

data_list = inpt.readlines()
answ_list = answ.readlines()
answ = []
netansw = []

for i in range(len(data_list)):
    test = data_list[i].strip('\n').split(',')
    test_answ = answ_list[i].strip('\n').split(',')
    data.append(test)
    answ.append(test_answ)
    a = net.activate(test)
    netansw.append(a)
    for j in range(len(a)):
        if j == len(a)-1:
            netw.write(str(round(a[j],2)))
        else:
```

```
        netw.write(str(str(round(a[j],2))+','))
    netw.write('\n')

'''elapsed = timeit.default_timer() - start_time
print("Elapsed time: "+str(round(elapsed,2))+"s")

#######################################################################
#######################################################################
#Plot testing
t = np.arange(0.0, 1.0, 0.01)
i = 2212
plt.figure(2)
plt.subplot(211)
plt.plot( t, data[i], "ro", t, net.activate(data[i]), "b", t, answ[i],"g")
plt.show()'''
```

Lisa 6.Simulatsioonis kasutatud Python skriptid.

Lisa 7

```
#Takes all the .txt files from a location, given by 'path' and splits them into
#pieces with length N. Pieces are written into two separate files, from which
#the NN will take its data.

import os
import numpy as np

def norm(lst):
    lst2 = []
    lst3 = []
    for i in lst:
        lst2.append(i/abs(max(lst)-min(lst)))
```

```
    for j in lst2:
        lst3.append(round(j-np.mean(lst2),3))
    return lst3

path = 'C:\Users\Rando\Desktop\Net'

#rohk_data = open(path+'\NNtraining\Training_answ.csv','wb')
#foukg_data = open(path+'\NNtraining\Training_data.csv','wb')

rohk_data = open(path+'\NNtesting\Test_answ.csv','wb')
foukg_data = open(path+'\NNtesting\Test_data.csv','wb')
rohk = []
foukg = []

loik = 1250
jupp = 25
#for f in os.listdir(path +'\Rawdata'):

for f in os.listdir(path +'\Rawtestdata'):
    if f.endswith('.txt') == True:
#        txt = open('Rawdata' + '/' + f)

        txt = open('Rawtestdata' + '/' + f)
        fl = txt.readlines()
        k = 0
        for line in fl:
            data_trios = line.strip('\n').split('\t')
            rohk.append(round(float(data_trios[2]),3))
            foukg.append(round(float(data_trios[1]),3))
            k += 1
            if k >= loik:
                k = 0
                rohk = norm(rohk)
                foukg = norm(foukg)
```

```
            for i in range(loik//jupp -1):
                rohk_data.write(','.join([str(rohk[j]) for j in range(i*jupp,i*jupp+jupp)]) + '\r\n')
                foukg_data.write(','.join([str(-1*foukg[j]) for j in range(i*jupp,i*jupp+jupp)]) +
'\r\n')

            rohk = []
            foukg = []

        k = 0
        rohk = []
        foukg = []
        txt.close()

rohk_data.close()
foukg_data.close()

#Defining and training the network
######################################################################
from pybrain.tools.shortcuts import buildNetwork
from pybrain.datasets import SupervisedDataSet
from pybrain.supervised.trainers import BackpropTrainer
from pybrain.tools.customxml import NetworkWriter
import timeit
start_time = timeit.default_timer()
######################################################################
#Create a network with 50 input neurons, 75 hidden neurons
#and 50 output neurons.
net = buildNetwork(50, 95, 50)
#Define the dimentsions of the dataset.
ds = SupervisedDataSet(50, 50)
######################################################################
#Read in training data.
rohk_data = open('Mtrain\Training_answ.csv','r')
```

```
foukg_data = open('Mtrain\Training_data.csv','r')

data_list = foukg_data.readlines()
answ_list = rohk_data.readlines()
for i in range(len(data_list)):
    data = data_list[i].strip('\n').split(',')
    answer = answ_list[i].strip('\n').split(',')
    ds.addSample(data, answer)
#########################################################################
#Link the dataset with the created network, and train it.
trainer = BackpropTrainer(net, ds)
trainer.trainEpochs(40)

#Save the trained network for future use.
NetworkWriter.writeToFile(net, 'network.xml')
#########################################################################

elapsed = timeit.default_timer() - start_time
print("Elapsed time: "+str(round(elapsed,2))+"s")

#Testing the network
#########################################################################
from pybrain.tools.customxml import NetworkReader
import matplotlib.pyplot as plt
import numpy as np
import timeit

start_time = timeit.default_timer()
#########################################################################
#Load in pretrained network
net = NetworkReader.readFrom('network.xml')
#########################################################################
#Net testing
inpt = open("Mtest/Test_data.csv",'r')
```

```
answr = open("Mtest/Test_answ.csv",'r')
netw = open("Mtest/Net_answ.csv",'wb')

data_list = inpt.readlines()
answ_list = answr.readlines()
answ = []
data = []
netansw = []

for i in range(len(data_list)):
    test = data_list[i].strip('\n').split(',')
    test_answ = answ_list[i].strip('\n').split(',')
    data.append(test)
    answ.append(test_answ)
    a = net.activate(test)
    netansw.append(a)
    for j in range(len(a)):
        if j == len(a)-1:
            netw.write(str(round(a[j],3)))
        else:
            netw.write(str(str(round(a[j],3))+','))
    netw.write('\n')

inpt.close()
answr.close()
netw.close()

'''elapsed = timeit.default_timer() - start_time
print("Elapsed time: "+str(round(elapsed,2))+"s")

#####################################################################
#####################################################################
#Plot testing
t = np.arange(0.0, 1.0, 0.01)
```

```
i = 2212
plt.figure(2)
plt.subplot(211)
plt.plot( t, data[i], "ro", t, net.activate(data[i]), "b", t, answ[i],"g")
plt.show()'"

#####################################################################
```

Lisa 7. FouKG katses kasutatud Pythoni skriptid.

Lisa 8

```
%This code is used to take all .txt files from a folder and convert them
%into a single file, that contains all nessecary data for
%neural network
dirName = 'C:\Users\Rando\Desktop\Net\Rawdata';            %# folder path
files = dir( fullfile(dirName,'*.txt') );   %# list all *.txt files
files = {files.name}';                    %'# file names
N = 50;                                 %length
data = cell(numel(files),1);              %# store file contents
foukg=[];
rohk=[];
for i=1:numel(files)
   display(round(i/numel(files)*100,2),'%')
   fname = fullfile(dirName,files{i});    %# full path to file
   data{i} = importdata(fname);           %# load file

   foukg=[foukg,data{i}(:,2).'];
   rohk=[rohk,data{i}(:,3).'];
end

coeff = ones(1, 50)/50;                       %Filter details
fDelay = (length(coeff)-1)/2;
```

```
foukg = filter(coeff, 1,foukg);
foukg = foukg(fDelay:end);
A = floor(length(foukg)/N)*N;                    %Smoothing FouKG
foukg = foukg(1:A);

foukg=-1*reshape(foukg,N,round(length(foukg)/N));

rohk = rohk(1:length(rohk)-fDelay+1);            %Cut data into pieces
B=floor(length(rohk)/N)*N;
rohk=rohk(1:B);

rohk=reshape(rohk,N,round(length(rohk)/N));
save('test.mat','foukg','rohk');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This code is used to train the neural network and save the outputs.

load('test.mat')

net = fitnet(10,'trainbr');                      %Define the network
%net.layers{1}.transferFcn = 'tansig';
%net.layers{2}.transferFcn = 'purelin';
s = size(rohk);
net.divideFcn = 'divideblock'                    %Divide data successively
[trainInd,valInd,testInd] = divideblock(s(1),0.7,0.2,0.1); % 70% for training, 20% to validate
and 10% to test.

[net, tr] = train(net, foukg, rohk);             %Train the network

%my_weights = getx(net) %get weights             %In case I want to load the weights from
file

%net = setx(net,my_weights)                      %if I want to set new weights

save('network','net')                            %Save the network
```

```
outputs = net(foukg);
testout = outputs(:,tr.testInd);
testin = foukg(:,tr.testInd);
answ = rohk(:,tr.testInd);

save('testcompare.mat','testin','testout','answ'); %save outputs
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This code calculates all the similarity parameters and saves them into a file

clear all
close all
clc

coeff = ones(1, 50)/50;
fDelay = (length(coeff)-1)/2;            %NN output filtering parameters
N= 500                                   %Length of the pieces (2sec)

load('testcompare.mat')

A = answ;
A = reshape(A,[],1);                     %A=volume curve
A = A(1:length(A)-fDelay+1);             %Reshaping the vectors accordingly
a=floor(length(A)/N)*N;
A=A(1:a);
A=reshape(A,N,round(length(A)/N)).';

B = testout;
B = reshape(B,[],1);                     %B=NN output
B = filter(coeff, 1,B);
B=B(fDelay:end);
B=smooth(B,0.00005);
b=floor(length(B)/N)*N;                  %Taking account the delay from the filter
B=B(1:b);
B=reshape(B,N,round(length(B)/N)).';
```

```
C=testin;
C = reshape(C,[],1);
C = C(1:length(C)-fDelay+1);
c=floor(length(C)/N)*N;
C=C(1:c);
C=reshape(C,N,round(length(C)/N)).';

D=[];
D2=[];
d=[];
d2=[];
e=[];
e2=[];
K=size(A);
for i = 1:K(1);
   R=simi2004_3(A(i,:),C(i,:),0); %,'PLOT','FINAL','SAVE','OFF','TEXTA','TEXTB';
   R2=simi2004_3(A(i,:),B(i,:),0);          %Compute index of similaritY
   r=corrcoef(A(i,:),C(i,:));               %Compute correlation
   r=r(1,2);
   r2=corrcoef(A(i,:),B(i,:));
   r2=r2(1,2);
   err = immse(A(i,:),C(i,:));          %Compute mean-squared error
   err2 = immse(A(i,:),B(i,:));
   D=[D,R];
   D2=[D2,R2];
   d=[d,r];
   d2=[d2,r2];
   e=[e,err];
   e2=[e2,err2];
end
R_real_n_input=sort(D);
R_real_n_output=sort(D2);
r_real_n_input=sort(d);
```

```
r_real_n_output=sort(d2);
e_real_n_input=sort(e);
e_real_n_output=sort(e2);
save('Rvordlused.mat','R_real_n_input','R_real_n_output','r_real_n_input','r_real_n_output','e
_real_n_input','e_real_n_output');
```

Lisa 8. FouKG katses kasutatud MATLAB-i skriptid.