BRAININFO 2017

The Second International Conference on Neuroscience and Cognitive Brain Information

ISBN: 978-1-61208-579-1

July 23 - 27, 2017

Nice, France

BRAININFO 2017 Editors

Birgit Gersbeck-Schierholz, Leibniz Universität Hannover, Germany

Lei Zhang, University of Regina, Canada

BRAININFO 2017

Foreword

The Second International Conference on Neuroscience and Cognitive Brain Information (BRAININFO 2017), held between July 23 - 27, 2017 - Nice, France was dedicated to evaluate current achievements and identify potential ways of making use of the acquired knowledge, covering, the neuroscience, brain connectivity, brain intelligence paradigms, cognitive information, and specific applications.

Complexity of the human brain and its cognitive actions stimulated many researches for decades. Most of the findings were adapted in virtual/artificial systems in the idea of brain-like modeling them and used in human-centered medical cures, especially for neurotechnologies. Information representation, retrieval, and internal data connections still constitutes a domain where solutions are either missing or in a very early stage.

We take here the opportunity to warmly thank all the members of the BRAININFO 2017 Technical Program Committee, as well as the numerous reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to BRAININFO 2017. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the BRAININFO 2017 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that BRAININFO 2017 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the area of neuroscience and cognitive brain information.

We are convinced that the participants found the event useful and communications very open. We also hope that Nice provided a pleasant environment during the conference and everyone saved some time for exploring this beautiful city.

BRAININFO 2017 Chairs:

BRAININFO Steering Committee

Marius George Linguraru, Children's National Medical Center | George Washington University School of Medicine and Health Sciences, Washington D.C., USA
Konrad Ciecierski, Institute of Computer Science - Warsaw University of Technology / Clinic of Neurosurgery - Warsaw Institute of Psychiatry and Neurology, Poland
Erwin Lemche, Institute of Psychiatry, Psychology & Neuroscience | King's College School of Medicine and Dentistry, UK
Nikola Kasabov, Auckland University of Technology, New Zealand

BRAININFO 2017

Committee

BRAININFO Steering Committee

Marius George Linguraru, Children's National Medical Center | George Washington University School of Medicine and Health Sciences, Washington D.C., USA
Konrad Ciecierski, Institute of Computer Science - Warsaw University of Technology / Clinic of Neurosurgery - Warsaw Institute of Psychiatry and Neurology, Poland
Erwin Lemche, Institute of Psychiatry, Psychology & Neuroscience | King's College School of Medicine and Dentistry, UK
Nikola Kasabov, Auckland University of Technology, New Zealand
Yudong Zhang, Nanjing Normal University, China
Domenico Ursino, University "Mediterranea" of Reggio Calabria, Italy
Mariano Alcañiz Raya, Institute of Research and Innovation in Bioengineering (I3B) - Universidad Politécnica Valencia, Spain
Jayfus Tucker Doswell, The Juxtopia Group, Inc., USA

BRAININFO 2017 Technical Program Committee

Mariano Alcañiz Raya, Institute of Research and Innovation in Bioengineering (I3B) - Universidad Politécnica Valencia, Spain
Mehdi Ammi, LIMSI | University of Paris-Saclay, France
Gian Carlo Cardarilli, University of Rome Tor Vergata, Italy
Konrad Ciecierski, Institute of Computer Science - Warsaw University of Technology / Clinic of Neurosurgery - Warsaw Institute of Psychiatry and Neurology, Poland
Liezel Cilliers, University of Fort Hare, South Africa
Jayfus Tucker Doswell, The Juxtopia Group, Inc., USA
Deniz Erdogmus, Northeastern University, USA
Frank Hsu, Fordham University, New York, USA
Xiaoping Hu, Emory University & Georgia Institute of Technology, USA
Christof Karmonik, Houston Methodist Research Institute, USA
Asoka Karunananda, University of Moratuwa, Sri Lanka
Nikola Kasabov, Auckland University of Technology, New Zealand
Peter Kieseberg, SBA Research, Austria
Erwin Lemche, Institute of Psychiatry, Psychology & Neuroscience | King's College School of Medicine / Dentistry Department of Psychosis Studies | Section of Cognitive Neuropsychiatry, UK
Marius George Linguraru, Children's National Medical Center | George Washington University School of Medicine and Health Sciences, Washington D.C., USA
Eirini Mavritsaki, Birmingham City University, UK
Vasilis Megalooikonomou, University of Patras, Greece
Jantima Polpinij, Mahasrakham University, Thailand
Pinaki Sarder, University at Buffalo - The State University of New York, USA
Tiziano Squartini, IMT School for Advanced Studies Lucca, Italy

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

N-Back Training and Transfer Effects in Healthy Young Subjects Using EEG

Valentina Pergher, Benjamin Wittevrongel
KU Leuven - University of Leuven, Department of Neurosciences,
Laboratory for Neuro- & Psychophysiology,
Leuven, Belgium
email: valentina.pergher@kuleuven.be,
benjamin.wittevrongel@kuleuven.be

Birgitte Schoenmakers
Academic Centre of General Practice,
KU Leuven - University of Leuven,
Leuven, Belgium
email: birgitte.schoenmakers@med.kuleuven.be

Jos Tournoy
Faculty of Medicine, Department of Experimental Medicine,
Division of Gerontology and Geriatrics,
KU Leuven - University of Leuven ,
Leuven, Belgium
email: jos.tournoy@uzleuven.be

John Arsenault, Marc M. Van Hulle
KU Leuven - University of Leuven, Department of Neurosciences,
Laboratory for Neuro- & Psychophysiology,
Leuven, Belgium
email: john.arsenault@kuleuven.be,
marc.vanhulle@med.kuleuven.be

***Abstract*- We investigate whether N-Back working memory (WM) training improves both trained WM- and untrained cognitive function performance (transfer effects). Previous studies showed that EEG responses, in particular Event Related Potentials (ERPs), can be used as a measure of working memory load during cognitive task performance. Here, we used three groups of young healthy participants to assess the effect of N-Back training: cognitive training group (CTG), active control group (ACG) and passive control group (PCG). The cognitive training group performed an N-Back task with 3 difficulty levels (1, 2, 3-Back), the active control group used the same task but with lower difficulty levels (0, 1, 2-Back), and the control group no N-Back training at all. Pre- and post-tests were administered to all three groups to gauge any transfer effects (partial memory, attention, reasoning and intelligence). Our results showed that training improved N-Back task performance for CTG participants compared to ACG and PCG participants. In contrast, transfer effects were not so clear across cognitive tasks but transfer effects were present and stronger in CTG compared to ACG for attention (TOVA test).**

Keywords-EEG; working memory training; transfer effects; P300 ERP.

I. INTRODUCTION

Working memory (WM), as defined by Baddeley [1], refers to a brain system that provides temporary storage and manipulation of information necessary to execute complex cognitive tasks. WM training was originally used to enhance WM in neuropsychiatric subjects with a WM deficit, such as attention deficit hyperactivity disorder (ADHD) [2] and several authors studied the mechanisms behind and the effect of WM training [3][4].

The N-back task is a working memory task introduced by Wayne Kirchner in 1958 [24] as a visuo-spatial task with four load factors (‘‘0-Back’’ to ‘‘3-Back’’), and by Mackworth [23] as a visual letter task with up to six load factors. Gevins et al. [5] introduced it to the field of neuroscience by using it as a ‘‘visuomotor memory task’’ with one load factor (3-Back). The task involves multiple processes and is considered a dual task: working memory updating, which involves the encoding of incoming stimuli, the monitoring, maintenance, and updating the sequence, and stimulus matching (matching the current stimulus to the one that occurred N positions back in the sequence). It reflects a number of core Executive Functions (EFs) besides working memory, such as inhibitory control and cognitive flexibility, as well as other higher-order EFs, such as problem solving, decision making, selective attention, among others [6]. The N-Back task requires participants to maintain simultaneously stimulus information necessary for successful task performance in working memory across multiple trials [6]. It has been shown that the N-Back task consistently activates dorsolateral prefrontal cortex (DLPFC), as well as parietal regions in the adult brain [7]. Schneiders et al. [8] have shown that, using a N-Back training, it is possible to achieve an improvement in task performance and an alteration in brain activity, such as a decreased activation in the right superior middle frontal gyrus (BA 6) and posterior parietal regions (BA 40).

Following a series of studies, Jaeggi et al. [9][10] reported that by performing an N-Back task, the effects of WM training transfer to untrained tasks requiring WM (transfer effects) and improve upon a complex human ability known as fluid intelligence. Jaeggi et al.’s [9] findings support the hypothesis that transfer effects to general cognitive functions can be achieved after single and dual N-Back training for tasks that conceptually overlap, albeit only slightly, with the N-Back. Training of the general fronto-parietal WM network should lead to improvements in cognitive functions that rely on the same network [2]. This general overlap hypothesis predicts that if training considerably engages the fronto-parietal WM network and the transfer task generates a similar activation pattern, an extensive training of this network will yield a general boosting of cognitive functions. An alternative hypothesis predicts that WM training effects transfer only if training improves specific cognitive processes required in both training and transfer tasks. Dahlin et al. [11] found transfer, after WM updating training, to an N-Back task that resembled the original trained task in also relying on updating processes, but not to a Stroop task that involved inhibition but no updating.

 ISBN: 978-1-61208-579-1

The aim of our study was to verify whether N-Back task performance improves and whether transfer effects to other (untrained) cognitive functions are obtained, such as spatial memory, attention and reasoning, in three different groups of healthy young subjects: cognitive training group (CTG), active control group (ACG) and passive control group (PCG).

The paper is organized as follows. In Section 2, we describe the material and methods (subjects, procedure, EEG recording). In Section 3, we focus on the behavioral and ERPs results using a WM training and on the transfer effects pre and post-training. Finally, in Section 4, we discuss our results and propose a number of technical and conceptual goals for future studies.

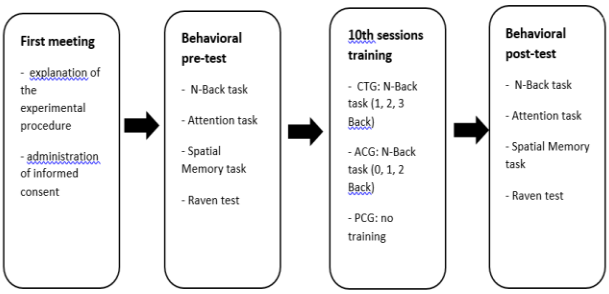


Figure 1. Study design

II. MATERIALS AND METHODS

In this section we describe the participants, procedure and EEG recording.

A. *Subjects*

We recruited 16 healthy young subjects (6 females, mean age 29 years, range 24-34 years), undergraduate or graduate students from KU Leuven and non- students. Participants were healthy, reported normal or corrected vision, no history of psychiatric or neurological diseases, they were not taking any medications and never participated in working memory training. Participants were assigned to three sub-groups, cognitive training (N=6), active control (N=5) and passive control group (N=5), to evaluate improvements in task performance after the WM training and to record any transfer effects to other cognitive tasks (see further for their definition). During all training sessions, EEG was recorded (see also further). In the cognitive training group, 3 subjects performed WM training with visual feedback on the correctness of their behavioral response and other 3 subjects with monetary reward (with a maximum of 10 € if all responses are correct), however, the sample turned out to be too small to reveal any significant differences. The active control group performed the same training task, but the difficulty level was lower (0, 1, 2-Back task) and with monetary reward (max. 10 €/session). The control group did not undergo any training. A battery of cognitive tests were administered before and after training (pre and post-tests, note that for the control group there was no training between these tests) to see if there were transfer effects in attention, spatial memory, reasoning and intelligence. The study was approved by our university's ethical committee and informed consent was obtained from our subjects prior to their participation in the experiment.

B. *Procedure*

Subjects participated in an N-back task in which, see Figure 2, a sequence of stimuli were shown and the task was to decide whether the current stimulus matched the one presented N items earlier.

The stimuli were presented for 1000ms followed by a 2000ms Inter-stimulus interval (ISI), adding jitter of ± 100 ms, during which the picture is replaced by a fixation cross. This is the moment where the participants needed to press the button if the stimulus was a target; 33% of our pictures were targets.

Sequences with identical difficulty levels (all 0-back, 1-back, 2-back, 3-back) were grouped into 2 min. blocks across four sessions. Each session included two repetitions of 3 sequences. In total there were 8 blocks. For each sequence, there were 60 stimuli presented in pseudorandom order. Before starting with the first three sequences, a training session consisting of ten stimuli for each difficulty level was administered to explain the N-Back task.

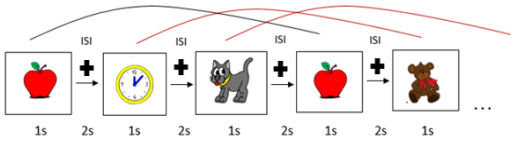


Figure 2. Graphical rendition of 3(N)-back task

Subjects performed an N-Back training during 10 sessions, 3 times per week (30 minutes each time), as shown in Figure 1. This is in line with literature reports on significant training and transfer effects obtained after 3 weeks of training [9][12]-[15].

TABLE 1. COMPARISON (MEAN) OF PRETEST AND POSTTEST PERFORMANCE (ACCURACY) BETWEEN TRAINING GROUP (N=6), ACTIVE (N=3) AND PASSIVE CONTROL GROUPS (N=5) IN THE TRAINED (N-BACK) AND IN UNTRAINED TASKS.

Task	Training group		Active control group		Passive control group	
	Pretest	Posttest	Pretest	Posttest	Pretest	Posttest
N-Back task	21*	5*	19*	6.3*	16*	13*
TOVA task	7.8*	3.3*	9.3*	3*	7.4*	5*
Corsi task	8.8**	10.4**	8.7**	9.7**	9.2**	9.4**
Raven task	3.5*	1.8*	6*	3.3*	4.6*	4.2*

*Incorrect responses **Correct responses

 ISBN: 978-1-61208-579-1

All participants were administered a battery of pre- and post-tests to evaluate whether there were transfer effects to other cognitive functions (attention, spatial memory, reasoning and intelligence). We used Test of Variables of Attention (TOVA) [16], Spatial Working Memory Test (CORSI) [17] and Raven test [19]. The behavioral pre- and post-tests were administered to compare task performance between groups (cognitive training, active control and passive control groups) in the trained (N-Back task) and untrained tasks (TOVA, CORSI and Raven test).

N-back task and transfer tasks had similarities and differences [9][19][20]. The spatial memory task (Corsi test) engaged WM updating processes just as the N-Back task, but differed in stimuli (squares in Corsi task vs pictures in the N-Back task) and task rules (recognition of previously presented items in the N-Back tasks vs. recollection of items in the updating transfer tasks). Given these similarities and differences, we are using near transfer tasks according to Karbach and Kray [21].

In the first experimental session (pre-test), each participant was informed about the experimental procedure and invited to sign the informed consent form. The day after the first meeting, the participants performed the behavioral pre-test session, and from the third meeting, the two training groups (CTG, ACG) started their training procedure of CTG and ATG participants were not informed about the group to which they were assigned or its purpose. At the beginning of each training session, an EOG calibration session was performed to capture eye movements and blinks using the method described in Croft & Barry [22].

C. EEG recording

EEG was recorded continuously from 32 Ag/AgCl electrodes at a sampling rate of 2 kHz using a SynampsRT device (Neuroscan, Australia). The electrodes were placed at O1,Oz, O2, PO3, PO4, P8, P4, Pz, P3, P7, TP9, CP5, CP1, CP2, CP6, TP10, T7, C3, Cz, C4, T8, FC6, FC2, FC1, FC5, F3, Fz, F4, AF3, AF4, Fp1, Fp2. The reference was placed at AFz and the ground at CPz. Additionally, four electrodes were placed around the eyes, on the upper and lower side of the left eye (vertical) and near the external canthus of each eye (horizontal), for electro-oculogram recording (EOG, bi-polar recording).

The recorded EEG signal was re-referenced offline from the original reference to the average of two mastoid electrodes (TP9 & TP10), corrected for eye movement and blinking artifacts [22], band-pass filtered in the range of 0.1–315Hz, and cut into epochs starting 200 ms pre- till 1000 ms post-stimulus onset. Baseline correction is performed by subtracting the average of the 200 ms pre-stimulus onset activity from the 1000 ms post-stimulus onset activity. Finally, the epochs are down sampled to 64 Hz and stored for ERP detection.

Recorded epochs with incorrect responses were excluded from further analysis. In addition, epochs with EEG signals greater than 100mV were excluded from analysis. A two-way ANOVA (factors: n-back X target) was performed on all sampled EEG time points between -300 ms to 700 ms. Bonferroni correction for multiple comparisons was used across all samples within this time window.

III. RESULTS

In this section we describe working memory training (behavioral and ERPs results) and transfer effects pre and post-tests.

A. Working memory training (behavioral)

In Figures 3 and 4, we analyzed changes due to cognitive training by examining behavioral data (accuracy, reaction time (RT)) of CTG and ACG during N-Back training (10 sessions). The purpose is to test our second hypothesis: training can improve related cognitive function performance, and also transfer to other cognitive functions, in terms of RT and response accuracy revealed significant effects.

For the CTG, we observed a reduction in RT with an increased number of training sessions. To test this, we performed a three-way ANOVA across factors (N-back level, subject and session). We found a significant effect of session ($F_{(9)}$=4.9, p<0.001) confirming that RT indeed decreases with more training. Importantly, the N-Back x session interaction was significant ($F_{(18)}$=3.01, p<0.001), which indicates that the N-back levels are differentially affected by training. In contrast, when we looked at accuracy, the main effect of session was not significant (p=0.56) indicating that accuracy did not substantially increase as a result of training although there was a main effect of N-back level confirming that task difficulty affected performance ($F_{(2)}$=7.97, p<0.05).

For the active control group (ACG), RT decreases. It is significant for N-Back x session ($F_{(18)}$=1.95, p<0.05), and for subject x session interactions ($F_{(18)}$=4.84, p<0.001). This indicates that the number of training sessions is subject and task-specific. Accuracy differences were significant for N-Back x subject interaction ($F_{(4)}$=6.8, p<0.001), N-Back x session interaction ($F_{(18)}$=2.31, p<0.05), and for subject x session interaction ($F_{(18)}$=2.54, p<0.05), which means that N-Back and training session are subject-specific, and N-Back is affected by the number of training sessions.

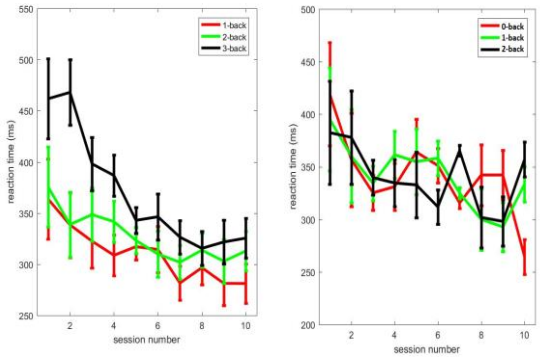


Figure 3. Left, RT during 10 sessions of cognitive training in CTG; right, RT during 10 sessions of cognitive training in ACG. Error bars indicate SEM.

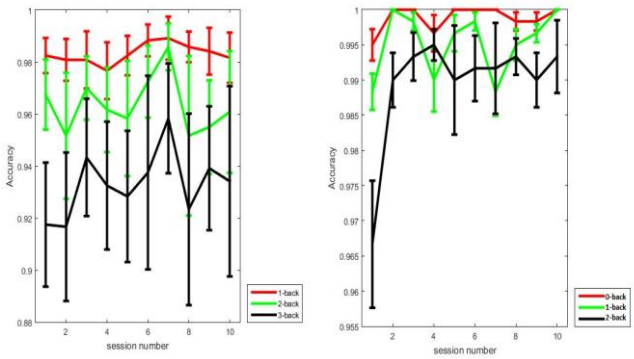


Figure 4. Left, accuracy during 10 sessions of cognitive training in CTG; right, accuracy during 10 sessions of cognitive training IN ACG. Error bars indicate SEM.

We observed significant effects between the two groups (CTG, ACG): the accuracy between CTG and ACG was significant for N-Back ($F_{(1)}$=8.26, p<0.05), and group ($F_{(1)}$=18.39, p<0.001). The RT in the two groups was significant for session ($F_{(9)}$=3.44, p<0.001) and group ($F_{(1)}$=7.02, p<0.05).

B. *Working memory training (ERPs results)*

As neuroimaging studies have shown that during N-Back task performance the most activated brain regions are the lateral premotor cortex, dorsal cingulate and medial premotor cortex, dorsolateral and ventrolateral prefrontal cortex, frontal poles, and medial and lateral posterior parietal cortex [5], and several studies showed that the midline electrodes are the most significant [25][26], we decided to analyze ERPs using electrodes located over these areas: Fz, Pz, and Cz. Figure 5 has shown a peak in P300 amplitude in three different moments (3 sessions/each moment) during training (first-, middle- and last sessions).

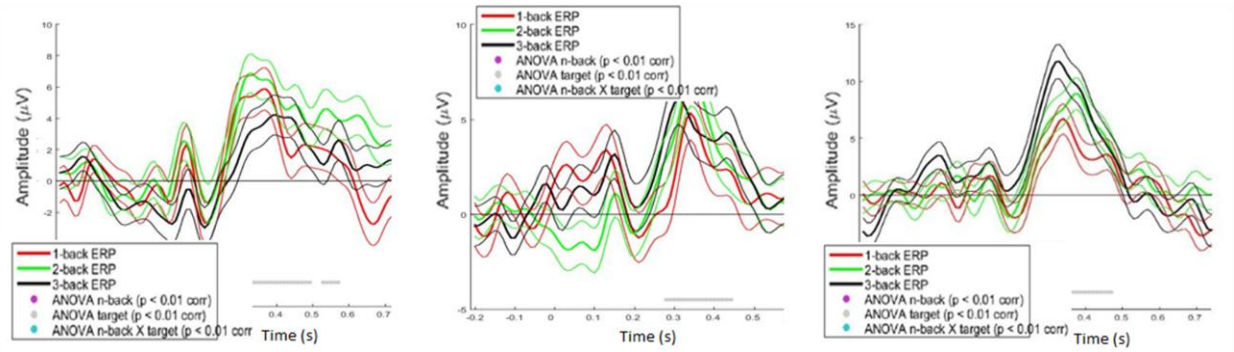
Data from mean P300 peak amplitude is presented in Figures 5 and 6. P300 peak amplitude data from midline electrodes (Fz, Cz, Pz) were analyzed with a three-way ANOVA (N-Back, target, and N-Back x target). P300 peak amplitude (target minus no-target) was higher for the N-Back difficulty levels that were easier (1 and 2-Back), and was lower for the more difficult one (3-Back). P300 peak amplitude (difference between target and no-target) was largest for the frontal electrode (Fz) and decreased for the central (Cz) and posterior electrodes (Pz). Furthermore, the P300 peak amplitude decreased progressively from the easiest task (0 or 1-Back) to the most difficult one (3-Back).

As a result of working memory training, the P300 peak became higher also for the most difficult task (3-Back). All together, these data support the observation that the P300 peak amplitude decreases with increased task load/difficulty, and with WM training it is possible to increase it also for the more difficult task.

C. *Transfer effects (Pre- and Post-tests)*

Means for each task are presented in Table 1 for the pre- and post-tests. In Figures 7 and 8, a multivariate ANOVA (MANOVA) was conducted between groups (CTG, ACG and PCG) and between sessions (pre- and post-tests). Significant effects for accuracy in N-Back task between CTG and PCG ($F_{(1)}$=6.21, p<0.05), and between CTG and ACG ($F_{(1)}$=14.21, p<0.05) for pre- and post-testing, were observed as well as significant effects in pre- and post-testing for accuracy in TOVA between CTG and ACG ($F_{(1)}$=8.18, p<0.05) and between ACG and PCG ($F_{(1)}$=5.24, p<0.05). No significant differences in CORSI and RAVEN test accuracies between groups were found.

For the N-Back task, significant effects were found for RT between CTG and PCG, for pre- and post-tests ($F_{(1)}$=40.9, p<0.001), for task difficulty level ($F_{(2)}$=4.92, p<0.05), for group x pre- and post-test interaction ($F_{(1)}$=9.14, p<0.05), and for pre- and post-test x N-Back level interaction ($F_{(2)}$=3.54,



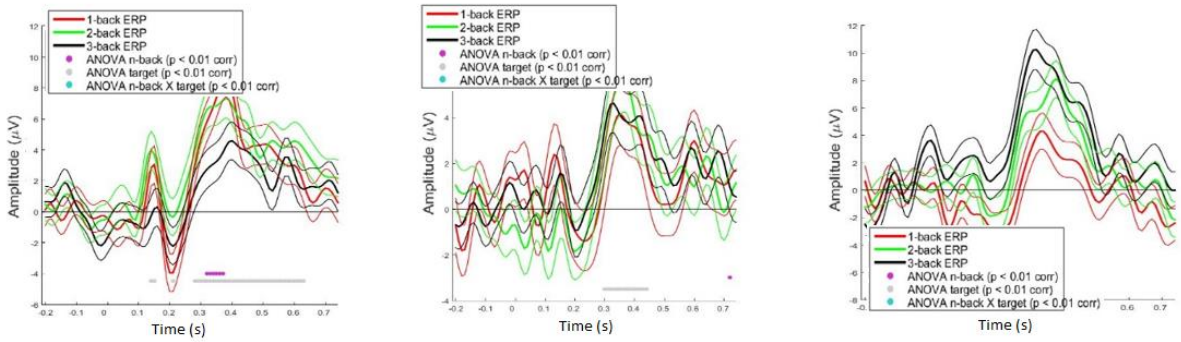


Figure 5. Pick of **P300**-ERPs (Fz and Cz, target minus non-target) in 6 subjects (CTG) in the **first** sessions of training (left), the **middle** (center) and the **last** ones (right). Significance measured using two-way ANOVA ($p < 0.01$, Bonferroni corrected for multiple comparisons. Error bars indicate SEM.

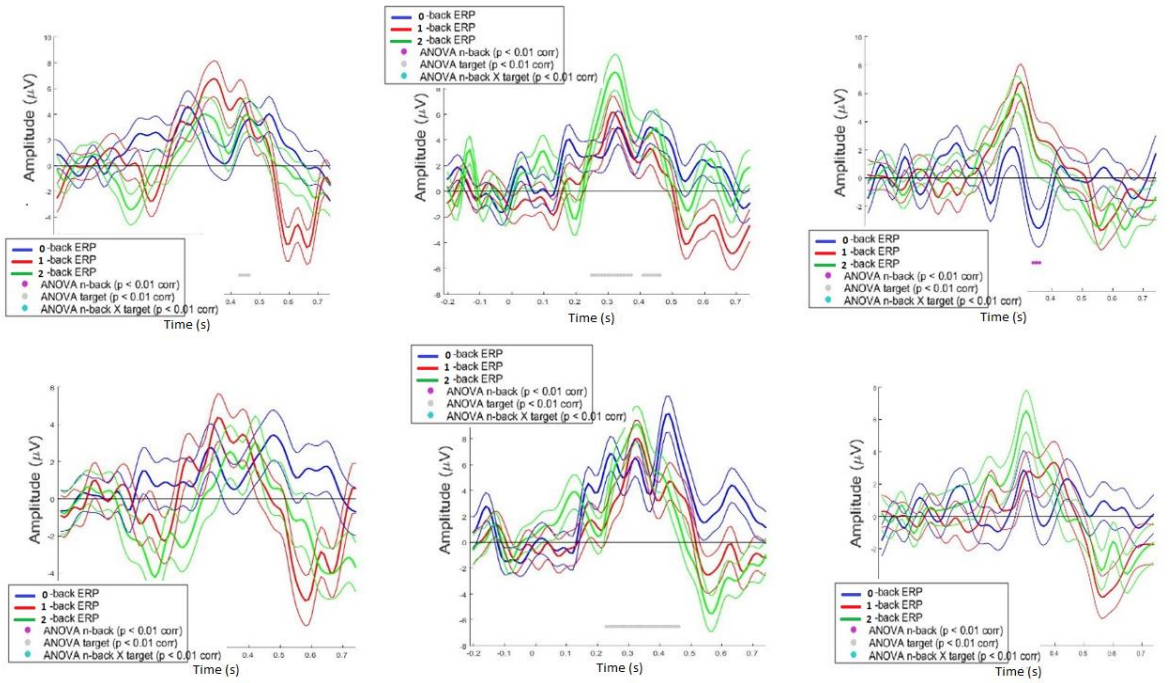


Figure 6. Peak of **P300**-ERPs (Fz and Cz target minus non-target) in 6 subjects (ACG) in the **first** sessions of training (left), the **middle** (center) and the **last** ones (right). Significance measured using two-way ANOVA ($p < 0.01$, Bonferroni corrected for multiple comparisons. Error bars indicate SEM.

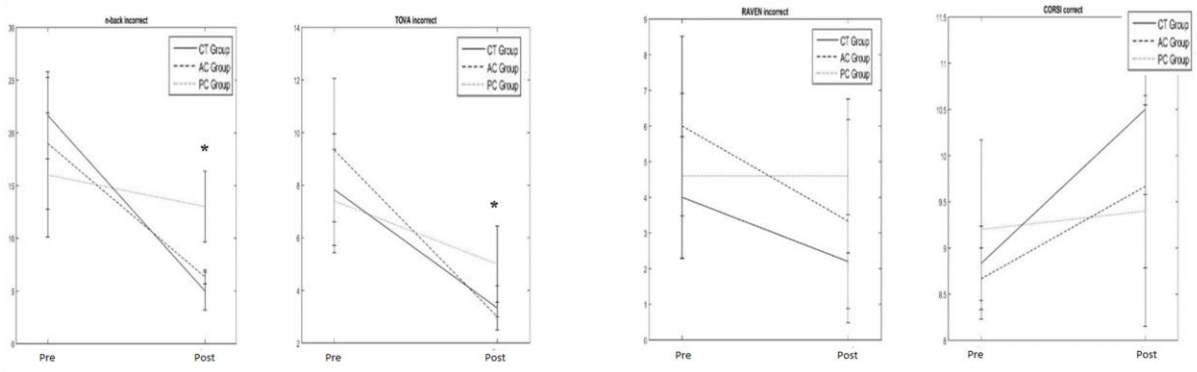


Figure 7. % incorrect performance of 3 groups for pre- to post-test in N-back task (1st figure, left), TOVA test (2nd figure), RAVEN test (3rd figure) and % correct performance in CORSI test (4th figure, right). Error bars indicate SEM. An asterisk indicates a significant difference between pre and post-tests

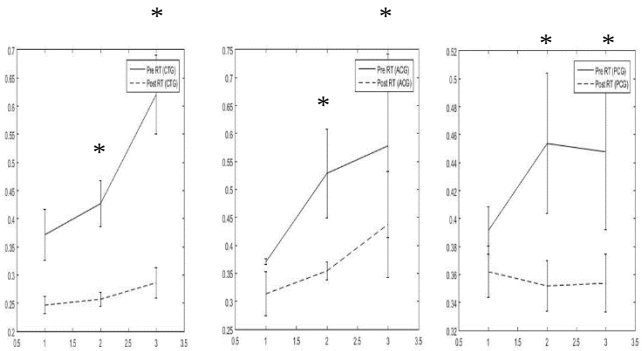


Figure 8. RT for correct responses for pre- and post-testing in the N-Back task of CTG (left), ACG (right), and PCG (bottom). Error bars indicate SEM. An asterisk indicates a significant difference between pre and post-tests.

$p<0.05$); between CTG and ACG for pre- and post-test ($F_{(1)}=25.6$, $p<0.001$), and for task difficulty level ($F_{(2)}=7.45$, $p<0.001$), and between ACG and PCG for pre- and post-test ($F_{(1)}=10.48$, $p<0.05$).

In summary, with our pre- and post-training tests, we wanted to verify whether any transfer effects could be obtained after N-Back training. Our results show clear improvements in attention.

IV. DISCUSSION

We investigated whether cognitive training using an N-Back task improves only N-Back task performance or does it transfer to other tasks. To assess this, we performed 10 N-Back training sessions in one group of participants (CTG) and assessed their cognitive performance for a battery of cognitive tasks (N-Back, TOVA, CORSI and Raven test) before and after training. During training, CTG participants performed the 1-,2-3-Back version of the N-Back task. To assess whether the level of difficulty affected training outcome, a second group of participants (ACG) performed the same experiment but with the 0-,1-2-Back versions of the N-Back task. Finally, a third group of participants (PCG) performed no training but was subjected to the same battery of cognitive tests. We found that training indeed improves performance for the CTG group compared to both the ACG and the PCG groups. Therefore, there is a clear improvement for the trained group on the task they were trained on. In contrast, the transfer of training effects into other tasks is more nuanced and although there was a trend for training effects in CTG to be stronger than for ACG this was only significant for the TOVA tests. These results are in contrast with the conclusions of Jaeggi at al. (2008) [9] who showed that a working memory task improves working memory and also fluid intelligence, and the study of Dahlin et al. [11] found that working memory training improves another working memory task but not other cognitive functions.

An issue that deserves consideration is why N-Back training in our study did not produce transfer effects in CORSI test (spatial memory) while in Dahlin et al. [11] they observed transfer effects to another memory task. In our view, this difference could be related to the size of the sample. Furthermore, as the EEG results from our study suggest a change in the P300 during the cognitive training, future study will consider not only the behavioral data (accuracy and RT), but also P300 component to change in real time the difficulty level of the task, avoiding too much fatigue or boredom for the subject.

In conclusion, we showed that N-Back training not only improves WM but also transfers improvement to another cognitive function (attention). The results provide evidence that it is possible to improve not only performance of tasks that include the same cognitive function (working memory), but also other cognitive tasks, as attention in our case.

ACKNOWLEDGMENTS

VP is supported by research grant G088314N, MMVH by research grants PFV/10/008, IDO/12/007, IOF/HB/12/021, G088314N, G0A0914N, IUAP P7/11, GOA 10/019, and the Hercules Foundation (AKUL 043). BW is supported by a Strategic Basic Research (SBO) grant, funded by VLAIO (Flemish Agency for Innovation and Entrepreneurship).

REFERENCES

[1] A. Baddeley, “Working Memory and Conscious Awareness”, Theories of memory, pp.11-20, 1992.

[2] T. Klingberg, “Training and plasticity of working memory”, Trends Cogn Sci 14, pp.317–324, 2010.

[3] C. C. von Bastian, and K. Oberauer, “Effects and mechanisms of working memory training: a review”, Psychol Res 78, pp.803–820, 2014.

[4] J. Au, et al., “Improving fluid intelligence with training on working memory: a metaanalysis”, Psychon Bull Rev 22, pp.366 –377, 2015.

[5] A. S. Gevins, et al., “Effects of prolonged mental work on functional brain topography”, Electroencephalography and clinical neurophysiology, 76(4), pp.339-350, 1990.

[6] M. J. Kane, and R. W. Engle, “The role of prefrontal cortex in working-memory capacity, executive attention, and general fluid intelligence: An individual-differences perspective”, Psychonomic bulletin and review, 9(4), pp.637-671, 2002.

[7] A. M. Owen, K. M. McMillan, A. R. Laird, and E. Bullmore, “N-back working memory paradigm: A meta-analysis of normative functional neuroimaging studies”, Human brain mapping, 25(1), pp.46-59, 2005.

[8] J. A. Schneiders, B. Opitz, C. M. Krick, and A. Mecklinger, “Separating intra-modal and across-modal training effects in visual working memory: an fMRI investigation”, Cerebral Cortex, 21(11), pp.2555-64, 2011.

[9] S. M. Jaeggi, “Improving fluid intelligence with training on working memory”, Proc. Natl. Acad. Sci. 105, pp.6829– 6834, 2008.

[10] S. M. Jaeggi, “The relationship between n-back performance and matrix reasoning implications for training and transfer”, Intelligence 38, pp.625–635, 2010.

[11] E. Dahlin, A. S. Neely, A. Larsson, L. Backman, and L. Nyberg, “Transfer of learning after updating training mediated by the striatum”, Science 320, pp.1510–1512, 2008.

[12] J. Chein, and A. Morrison, “Expanding the mind's workspace: Training and transfer effects with a complex working memory span task”, Psychonomic Bulletin and Review, 17 (2), pp.93–199, 2010.

[13] E. Dahlin, L. Backman, A. S. Neely, and L. Nyberg, “Training of the executive component of working memory: Subcortical

areas mediate transfer effects", Restorative Neurology and Neuroscience, 27 (5), pp.405–419, 2009.

[14] Y. Brehmer, H. Westerberg, and L. Bäckman, "Working-memory training in younger and older adults: training gains, transfer, and maintenance", Training-induced cognitive and neural plasticity, p.72, 2012.

[15] L. L. Richmond, A. B. Morrison, J. M. Chein, and I. R. Olson, "Working memory training and transfer in older adults. Psychology and aging", 26(4), p.813, 2011.

[16] L. M. Greenberg, and I. D. Waldmant. "Developmental normative data on the test of variables of attention (TOVA™)." Journal of Child Psychology and Psychiatry 34 (6), pp.1019-1030, 1993.

[17] R. P. Kessels, M. J. Van Zandvoort, A. Postma, L. J. Kappelle, and E. H. De Haan, "The Corsi block-tapping task: standardization and normative data", Applied neuropsychology, 7(4), pp.252-258, 2000.

[18] J. C. Raven, and J. H. Court, "Raven's progressive matrices and vocabulary scales", Oxford, UK: Oxford Psych Press, 1998.

[19] J. Persson, and P. A. Reuter-Lorenz, "Gaining Control: Training Executive Function and Far Transfer of the Ability to Resolve Interference [retracted]", Psychological Science, 19(9), pp.881-888, 2008.

[20] X. Zhao, Y. Wang, D. Liu, and R. Zhou, "Effect of updating training on fluid intelligence in children", Chinese Science Bulletin, 56(21), pp.2202-2205, 2011.

[21] J. Karbach, and J. Kray, "How useful is executive control training? Age differences in near and far transfer of task-switching training', Dev Sci 12, pp.978–990, 2009.

[22] R. J. Croft, and R. J. Barry, "Removal of ocular artifact from the EEG: a review", Neurophysiologie Clinique/Clinical Neurophysiology, 30(1), pp.5-19, 2000.

[23] J. F. Mackworth, "Paced memorizing in a continuous task", Journal of Experimental Psychology, 58(3), p.206, 1959.

[24] W. K. Kirchner, "Age differences in short-term retention of rapidly changing information", J Exp Psych, pp.1-17, 1958.

[25] S. Watter, G. M Geffen, and L. B. Geffen, "The n-back as a dual-task: P300 morphology under divided attention", Psychophysiology, 38(06), pp.998-1003, 2001.

[26] A. M. Brouwer,, et al., "Estimating workload using EEG spectral power and ERPs in the n-back task", Journal of neural engineering, 9(4), p.045008, 2012.

 ISBN: 978-1-61208-579-1

A GPU-accelerated Framework for Fast Mapping of Dense Functional Connectomes

Kang Zhao, Haixiao Du and Yu Wang
Department of Electronic Engineering, Tsinghua University
Beijing, China
Email: {zhaok14,duhx11}@mails.tsinghua.edu.cn,yu-wang@tsinghua.edu.cn

***Abstract*—In the context of voxel-based modalities like functional magnetic resonance imaging (fMRI), a dense connectome can be treated as a large-scale network where single voxels are directly used to define brain network nodes. Contrary to parcellated connectomes, dense connectomes have higher spatial resolution and are immune from the parcellation quality. However, the analysis of dense connectomes basically requires more powerful computing and storage capacities. Here, we proposed a graphics processing unit(GPU)-accelerated framework to perform fast mapping of dense functional connectomes. Specifically, the framework is scalable to high voxel-resolution imaging data($<$2mm) and can construct large-scale functional brain networks with lower time and memory overheads. Based on the proposed framework, three functional connectivity measures (Pearson's, Spearman's and Kendall's) were accelerated on the GPU for fast detection of possible functional links in dense connectomes. Experimental results demonstrated that our GPU acceleration for the Kendall's measure delivered a $>$50x speedup against both multi-core CPUs implementations and GPU-based related works.**

Keywords–neuroinformatics; dense connectomes; functional connectivity measures; GPU; voxel resolution.

I. INTRODUCTION

Recent advances in resting-state functional magnetic resonance imaging (rs-fMRI) technologies have provided a non-invasive way to depict spontaneous fluctuations in brain activity and thus facilitates the mapping of functional connectomes [1]. Thanks to the constant increase of imaging resolution, researchers nowadays are able to analyze functional connectivity patterns of human brain at a finer spatial resolution, which triggers the rise of 'dense connectome' study[2][3].

A dense functional connectome is generally modeled as a large-scale network whose nodes can be defined directly by voxels in fMRI imaging data [4]. The investigation of voxel-wise functional networks allows to uncover more detailed connectivity information but is typically coupled with considerable computation and storage demands [5]. Specifically, the total amount of voxels grows cubically as a function of voxel-resolution, leading to a sharp increase in computations when measuring the functional connectivity between all pairs of voxels. Moreover, formally represented by a connectivity matrix, a voxel-wise network requires quadratic complexity of storage with the growth of voxel amounts, which implies a considerable memory footprint for the construction of large-scale functional networks. As shown in Figure 1, at the 1mm resolution of approximate 1,600,000 voxels, more than 8 TB memory is required for the storage of the voxel-wise whole-brain connectivity matrix. Taken together, the computation and storage requirements are the most pressing problems in the study of dense functional connetomes.

Given the limited computational power, extensive research has attempted to scale down brain networks by either down-sampling the imaging data towards a coarser level, or aggregating network nodes to several large parcels in the light of anatomically or functionally-defined brain atlases, i.e., parcellated connectomes [6]. However, it is quite obvious that these solutions may lead to the loss of potentially significant connectivity information, not to mention that the analysis of parcellated connectomes are highly sensitive to the parcellation selection [7].

In recent years, the advent of general-purpose graphics processing units (GPGPUs) opens a new door to gigantic data processing [8]. Benefiting from many-core architectures, GPUs exhibit a high bandwidth and tremendous computational horsepower, and the collaboration of CPU-GPU can achieve remarkable performance boosts for many applications [9]. In the field of imaging connectomics, several attempts have been made to accelerate the mapping of dense connectomes using GPUs [10][11]. Nonetheless, these studies either are powerless in the treatment of high resolution data (e.g., 2mm or higher resolutions), or present a rapid deterioration of performance as the growth of voxel aggregates. Scalable GPU-based algorithms used to map dense functional connectomes are currently lacking.

In this paper, we proposed a GPU-accelerated framework aimed for fast mapping of dense functional connectomes. Specifically, the proposed framework enables fast construction of large-scale functional network based on three distinct functional connectivity (FC) measures: Pearson's, Spearman's and Kendall's measures [12]. Moreover, attributed to a novel memory optimization strategy, our framework is scalable to the high-resolution imaging data ($<$2mm). Experimental results showed that running on a single-GPU system, our framework can extract large-scale functional networks (10^6 nodes, 1% edge sparsity) within 1000 seconds.

The remaining part of this paper proceeds as follows. In section II, we begin with an overview on the general flow of functional network construction, focusing on the possible challenges for mapping dense connectomes. After that, a scalable GPU-accelerated framework and the corresponding accelerated methods of FC measures are described in order to tackle these challenges. The performances of these algorithms are analyzed in Section III. Section IV discusses the major contributions of our study along with some future expectations.

II. METHODS AND MATERIALS

This section details three aspects: the basic steps and issues of voxel-wise brain network construction, our solutions under different thresholding and FC measuring approaches as well as the design of experiments including data generation and the selection of benchmarks.

 ISBN: 978-1-61208-579-1

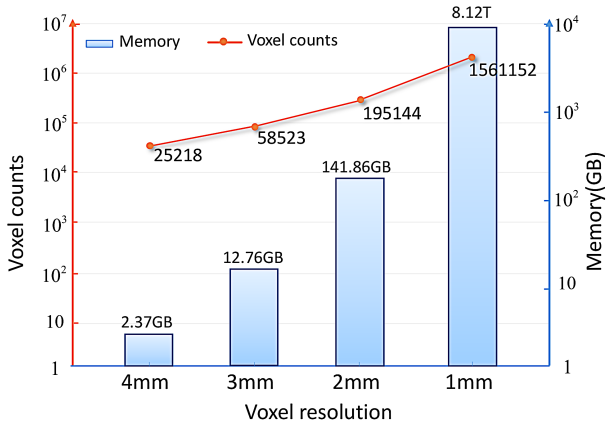


Figure 1. For different imaging resolutions, the corresponding number of voxels, and memory requirements for generating connectivity matrices. Source of statistical data: http://fcon_1000.projects.nitrc.org/indi/CoRR/html/bnu_1.html.

Algorithm 1 Constructing networks given the connectivity strength (CS) threshold

Input:data matrix **D**, the CS threshold **CSthreshold**;
Output:the resultant network **ResultNet_host**;
Variables defined on CPU main memory: **D_host**, **ResultNet_host**;
Variables defined on GPU memory: **D_dev**, **Batch_dev**;

```
1:  Transfer D_host to D_dev;
2:  Partition D_dev into m blocks, numbered from D_1 to D_m;
3:  for row← 1 to m do
4:      for column←row to m do
5:          Batch_device← GPU_f (D_row, D_column);
6:          GPU_thresholding(Batch_dev, CSthreshold);
7:          GPU_compressing(Batch_dev);
8:          Transfer Batch_dev to Batch_host;
9:          CPU_assemble(Batch_host,ResultNet_host);
10:         Batch_dev.clear();
```

A. An Overview on Functional Network Construction

In fMRI-based functional connectomics, the imaging data of a certain subject is acquired by successively recording blood oxygenation level-dependent signals at each imaging voxel site [4]. Then, the imaging data is preprocessed by some conventional means (e.g., slice timing correction, spatial and temporal filtering) before it is finally represented by a data matrix $\mathbf{D}^{N\times L}$, where N is the number of voxels and L is the length of time series [13]. After that, the construction flow of a functional network can be typically summarized into two main steps: generating connectivity matrix and thresholding. Firstly, the functional connectivity strength between any two voxels is calculated via diverse FC measures to describe how N distinct voxels functionally interact with each other, which generates an $N \times N$ connectivity matrix. Once a connectivity matrix is generated, given the consensus that human brain functional networks organize as an economical small-world topology tending to minimize wiring costs [14], a subsequent thresholding procedure should be applied to remove spurious connections to ensure the sparsity nature of the brain networks [15]. At present, there is no gold standard for the set of threhold values. Generally a sensitive analysis across diverse thresholds is recommendatory for researchers to seek appropriate thresholding parameters[16]. Finally, the sparse networks established though thresholding procedures can be compressed into a sparse format with lower memory footprints, e.g., the compressed sparse row (CSR) format [17].

For the construction of voxel-level networks, it is noteworthy that despite underlying huge memory demands of intermediate connectivity matrix (Figure 1), the established networks after thresholding are normally less memory-hungry. For example, under 1mm isotropic resolution, the established sparse network with 0.1% edge density requires only ∼16 GB memory, much lower than that of the entire connectivity matrix. Hence, the basic idea of our proposed framework is to avoid maintaining the entire connectivity matrix by employing a GPU-based block-wise thresholding strategy.

B. Scalable Solutions for Voxelwise Network Construction

As the network size grows dramatically with the increasing voxel aggregates, a scalable method is required for the voxel-wise network construction. Here, we proposed a GPU-based block-wise thresholding strategy under two different thresholding modes: given the connectivity strength threshold or the sparsity threshold [18].

1) Specify Connectivity Strength Thresholds: Assigning the connectivity strength threshold means that a fixed threshold value is set as the baseline when thresholding the connectivity matrix so that matrix elements greater than the given threshold are reserved while others are set to 0s.

In this case, the steps of generating connectivity matrix and thresholding can be easily merged. Specifically, the aforementioned data matrix $\mathbf{D}^{N\times L}$ can be divided into multiple (m) blocks, i.e., $\mathbf{D} = \{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_m\}$, where the block size is adjustable. Then, the corresponding connectivity matrix $\mathbf{R}^{N\times N} = f\left(\mathbf{D}^T, \mathbf{D}\right)$ can be derived by:

$$\mathbf{R} = \begin{bmatrix} f\left(\mathbf{D}_1^T, \mathbf{D}_1\right) & f\left(\mathbf{D}_1^T, \mathbf{D}_2\right) & \ldots & f\left(\mathbf{D}_1^T, \mathbf{D}_m\right) \\ f\left(\mathbf{D}_2^T, \mathbf{D}_1\right) & f\left(\mathbf{D}_2^T, \mathbf{D}_2\right) & \ldots & f\left(\mathbf{D}_2^T, \mathbf{D}_m\right) \\ \vdots & \vdots & \ddots & \vdots \\ f\left(\mathbf{D}_m^T, \mathbf{D}_1\right) & f\left(\mathbf{D}_m^T, \mathbf{D}_2\right) & \ldots & f\left(\mathbf{D}_m^T, \mathbf{D}_m\right) \end{bmatrix}, \quad (1)$$

where f represents distinct FC measures, e.g., the Pearson's measure. In this way, $\mathbf{R}^{N\times N}$ is generated block by block. Once a block is obtained, a subsequent thresholding and compressing procedure is performed immediately instead of doing this after the generation of the entire $\mathbf{R}^{N\times N}$. All the computation, thresholding and compression procedures can be efficiently completed by a GPU device, while the CPU only serves as an assembly line for receiving compressed data from the GPU in series and continuously jointing them together into a complete network that stays in main memory with a sparse format. The execution procedure is shown in Algorithm 1.

It should be noted that during the process, only a sparsely stored matrix is maintained in CPU main memory. Thus, the algorithm has a linear spatial complexity $O(N + E)$ for storing voxel-level functional networks, where E is the number of valid edges after thresholding. As the network can be quite sparse, the CPU memory usage is largely decreased in this way.

2) Specify Sparsity Thresholds: Another commonly used thresholding strategy is to fix the network sparsity. The sparsity

 ISBN: 978-1-61208-579-1

is defined as the proportion of the quantity of existing edges to the maximum possible number of edges in a network. Comparatively, the sparsity-based thresholding approach is more suitable for group-level comparisons on network topology [19] but is more complex to be applied in the construction of large-scale networks because the sparsity threshold should be firstly transferred to a corresponding connectivity strength threshold, which is in need of a time-consuming statistic for all entries in the connectivity matrix $\mathbf{R}^{N\times N}$.

In response, a GPU-accelerated algorithm characteristic of calculating the connectivity matrix $\mathbf{R}^{N\times N}$ for two times was designed. During the first generation, the GPU analyzes the distribution of element values in $\mathbf{R}^{N\times N}$ to derive the connectivity strength threshold corresponding to the given sparsity. Considering that a sparsity threshold restricts the number of actual connections k in a network, our basic idea is to find the k-th maximal element r_k in $\mathbf{R}^{N\times N}$ via GPU statistics, to serve as the connectivity strength threshold. Once r_k is obtained, the algorithm described above (Algorithm 1) can be reused to establish networks, which will also satisfy the constraint of the specified sparsity threshold. Then, the running of Algorithm 1 actually requires computing $\mathbf{R}^{N\times N}$ one more time.

Specifically in the first round, the range of the connectivity strength from 0 to 1 is segmented into multiple bins, with the bin quantity N_{bin}, and the bin width $\varepsilon = 1/N_{bin}$. Each time a block of $\mathbf{R}^{N\times N}$ is generated, a statistical histogram is maintained and updated by counting the quantity of elements in the block falling into different bins. A GPU-based sort-search histogram algorithm is applied to attain a fast and stable performance for large number of bins [20]. After finishing statistics of all blocks, the very bin where r_k is located can be find from the statistical histogram and the eventual outcome $\hat{r}$ is set as the median of this bin. The process is summarized in Algorithm 2. Notably, the error between $\hat{r}$ and r_k can be estimated by :

$$|\hat{r} - r_k| \leq \varepsilon/2 = 1/(2N_{bin}), \tag{2}$$

where the precision can be simply improved by increasing the number of bins, i.e., N_{bin}. In practice, we set $N_{bin} = 10^6$, rendering $\hat{r}$ an extreme approximation to r_k. Taken together, by adopting the block-wise statistical and approximate strategy we avoid maintaining the entire connectivity matrix as a whole, thereby reducing the algorithmic demand for CPU/GPU memory.

Algorithm 2 Derive the corresponding connectivity strength threshold from the given sparsity threshold

Input:data matrix **D**, the sparsity threshold **Sparsity**;
Output:the connectivity strength threshold **CSthreshold**;
Variables defined on CPU main memory: **D_host**;
Variables defined on GPU memory: **D_dev**, **Batch_dev**, **histogram_dev**;

```
Define k ← N × N × Sparsity;
Transfer D_host to D_dev;
Partition D_dev into m blocks, from D_1 to D_m;
for row← 1 to m do
    for column←row to m do
        Batch_dev← GPU_f (D_row, D_column);
        GPU_histogram(Batch_dev, histogram_dev);
        Batch_dev.clear();
Define Position ← GPU_upperBound(histogram_dev, k);
CSthreshold ← binWidth × Position + binWidth/2.0;
```

C. GPU implementation of three FC Measures

FC measures are used to quantify the strength of functional connections between network nodes. In this section, we will detail our GPU-accelerated algorithms for three commonly used FC measures: Pearson's, Spearman's and Kendall's measures, of which the latter two are considered more robust to outlying observations [21]. To accelerate the calculation of FC measures on a GPU, the basic principle of our proposed algorithms is to transform the computation of these measures into normative operations that GPUs excel in, e.g., vectors or matrices multiplications, both of which that possess high parallelism can be executed very quickly on GPUs.

As mentioned above, an fMRI data set of a single subject can be represented by a data matrix $\mathbf{D}^{N\times L} = (d_i)$, where $1 \leq i \leq N$, and $d_i = (d_{i1}, d_{i2} \ldots, d_{iL})$, i.e., the *i*-th row of $\mathbf{D}^{N\times L}$, denoting the time series of the *i*-th voxel, with the sequence length *L*. Thus $r_{i,j}$, the FC measurements between voxel *i* and voxel *j*, can be described as follows:

$$r_{i,j} = f(d_{i,}d_j), \tag{3}$$

where $f \in \{f_p, f_s, f_k\}$, representing Pearson's, Spearman's and Kendall's measures, respectively, whose definitions will be specified below.

1) Vectorization for Pearson's and Kendall's Measures: The Pearson measure of temporal correlation between two time-series d_i, d_j is defined by:

$$f_p = \frac{\sum_{k=1}^{L}\left(d_{ik} - \bar{d}_i\right)\left(d_{jk} - \bar{d}_j\right)}{S_{d_i} \cdot S_{d_j}}, \tag{4}$$

where $\bar{d}_i = \left(\sum_{k=1}^{L} d_{ik}\right)\Big/ L$ and $S_{d_i} = \sqrt{\sum_{k=1}^{L}\left(d_{ik} - \bar{d}_i\right)^2}$ are the mean and standard deviation of the time series of the *i*-th voxel respectively. Several studies have suggested that f_p be derived as the product of two normalized vectors [22]:

$$f_p = \sum\nolimits_{k=1}^{L}\left(\frac{d_{ik} - \bar{d}_i}{S_{d_i}}\right)\left(\frac{d_{jk} - \bar{d}_j}{S_{d_j}}\right) = \vec{p}_i \cdot \vec{p}_j, \tag{5}$$

where $\vec{p}_i$, $\vec{p}_j$ are vectors with the length L and $\vec{p}_{ik} = \left(d_{ik} - \vec{d}_i\right)\Big/ S_{d_i}$. Likewise, the formula of the Kendall's measure can be vectorized to:

$$f_k = \sum_{k=1}^{L-1}\sum_{q=k+1}^{L}\left(\frac{sign\left(d_{ik} - d_{iq}\right)}{\sqrt{m - m_i}}\right)\left(\frac{sign\left(d_{jk} - d_{jq}\right)}{\sqrt{m - m_j}}\right) = \vec{z}_i \cdot \vec{z}_j \tag{6}$$

where $\vec{z}_{ik} = sign\left(d_{ik} - d_{iq}\right)/\sqrt{m - m_i}$ and m, m_i are scalars [23]. Note that vectors $\vec{z}_i$ and $\vec{z}_j$ have the length $L\left(L-1\right)/2$.

To obtain all-pairs FC measurements, i.e., the connectivity matrix $\mathbf{R}^{N\times N} = (r_{i,j})$, the $\vec{p}_i\left(\vec{z}_i\right)$ of every voxel should be firstly derived. Then, the subsequent operations of all-pairs $r_{i,j} = \vec{p}_i \cdot \vec{p}_j$ (Pearson's measure), or $r_{i,j} = \vec{z}_i \cdot \vec{z}_j$ (Kendall's measure), can be unified as the matrix-matrix multiplication, which is efficient on GPUs.

2) Accelerating Rank Assignments for Spearman's Measure: The Spearman correlation between two time-series can be calculated by measuring the Pearson correlation of their ranked values of each samples [24], i.e.,

$$f_s(d_i, d_j) = f_p(n_i, n_j), \quad (7)$$

where n_i, n_j are the ranked sequences of d_i, d_j, respectively. That is, by replacing every element in a time-series with its rank, our proposed GPU-based procedure of Pearson's measures described above can be applied directly to accelerate the computation of the Spearman's measure. Hence, the key issue is how to calculate element ranks efficiently on a GPU.

A GPU-based sort-detection algorithm was introduced by Kim et al. (2012) [25] for assigning ranks, yet this approach is powerless in process of tied elements, i.e., multiple identical values in one time-series. Here, we propose a new rank assigning strategy calculating the rank of d_{ik} in the *i*-th time series as follows:

$$rank(d_{ik}) = LessNumber + (1 + EqualNumber)/2, \quad (8)$$

where $LessNumber$ is the amount of elements less than d_{ik} and $EqualNumber$ is the amount of tied elements equal to d_{ik} (including itself). To obtain the rank of an element d_{ik}, a GPU only needs to traverse all elements of a time-series and count the number of elements less than or equal to itself, instead of sorting all temporal samples. This strategy avoids possible branch operations and irregular memory access among multiple threads, thereby easily parallelized by a GPU with single instruction, multiple threads (SIMT) model [26]. Performances of the sort-detection algorithm and our own implementation of the Spearman's measure will be compared later.

D. Application and Example Datasets

Several experiments were conducted to illuminate the advantages of our proposed framework in two aspects: the run time efficiency and the scalability. To comprehensively assess the run time efficiency, two sets of data with respective number of nodes *N*=25218 (approximate to the voxel aggregates of a 4mm isotropic resolution imaging data) and *N*=58523 (approximate to the voxel aggregates of a 3mm isotropic resolution imaging data) were randomly generated and stored using a floating point format of 4 bytes per element. In addition, each data set has four variants with varied number of temporal samples (*L*=128, 256, 512, 1024), to evaluate the performance of our GPU-accelerated algorithms under different length of time series. In contrast, to investigate the scalability of the proposed framework, another input data set was produced with constant length of time series (*L*=128) but spanning a broad range of voxel quantities (*N*= 200,000, 400,000, 600,000, 800,000, 1000,000, respectively).

E. Benchmarks and Programs

All experiments were conducted on a workstation with an Intel(R) Core(TM) i7-6700K CPU (4G Hz, 8 cores, hyper-threading disabled), 64GB main memory, and an NVDIA GeForce GTX TITAN Black GPU with 6 GB device memory. The workstation supports dual operation systems (Windows 8.1 and Linux Ubuntu 16.10). MATLAB(R2016a) and CUDA(v8.0) are available on both systems.

So far, several parallel-processing approaches have been put forward to accelerate the calculation of FC measures. Here we consider two class of typical related works as comparisons: multi-core CPUs based implementations, and GPU based implementations by others.

The contrastive programs of Pearson's and Spearman's measures on multi-core CPUs are parallelized by invoking the Intel Math Kernel library(MKL), a widely used math library featuring highly optimized and easily parallelizable functions on multi-core systems [27]. Besides, the parallel implementation of the Kendall's measure on multi-core CPUs is based on a classical algorithm built upon merge sort [28]. On the current workbench, all these CPU-based programs are parallelized using 8 threads. As for GPU-based related works, the programs from *gputools*, a prevalent toolbox enabling efficient GPU computing in R [29], are picked up for the performance comparison on Pearson's and Kendall's measures, and the aforementioned sort-detection algorithm proposed by Kim et al. (2012) [25] is re-implemented and tested against our own implementation of the Spearman's measure.

III. RESULTS

We first assessed the performance of our GPU-accelerated algorithms for three FC measures (Figure 2). In this case, all programs were required to generate full-stored connectivity matrices without thresholding and compression operations. Then, the elapsed time of constructing sparse networks among different network scales and sparsity thresholds was presented in Table 1 to highlight the scalability of our framework.

A. Performance

Experimental results in Figure 2 showed that our proposed GPU-accelerated algorithms for Pearson's, Spearman's and Kendall's measures were more time-efficient against both multi-core CPUs and GPU based related works. In particular, our own implementation of the Kendall's measure exhibited a over 50x speedup than the other two ways.

Specifically, our GPU implementation of Pearson's measures only cost 1.7 seconds on average for generating the connectivity matrix at a 4mm isotropic resolution (*N*=25218) across different lengths of time series, and 4.3 seconds at a 3mm isotropic resolution (*N*=58523). Similarly, for the Spearman's measure, the average time costs were 2.0 and 4.4 seconds, respectively, at the two network scales. Actually, our GPU-accelerated procedure of the Spearman's measure performed only slightly slower than that of the Pearson's measure. Considering that the execution of Spearman's measure internally called the procedure of the Pearson's measure upon finishing rank assignments, the minor variance in computing time between the two measures implied the high efficiency of our proposed strategy for rank assignments, which led to the little time occupancy of this step during the calculation of the Spearman's measure. As for the Kendall's measure, our procedure spent averagely 63.5 and 340.5 seconds generating the connectivity matrix at the resolution of 4mm and 3mm, respectively, which was much less than that of the multi-core CPUs implementation using MKL parallel computing library [27] or the GPU implementation using *gputools* [29]. The latter two ways both consumed $>$1 hour dealing with 4mm resolution data and $>$5 hours with 3mm data. Notably, the calculation of the Kendall's measure that has a quadratic time

 ISBN: 978-1-61208-579-1

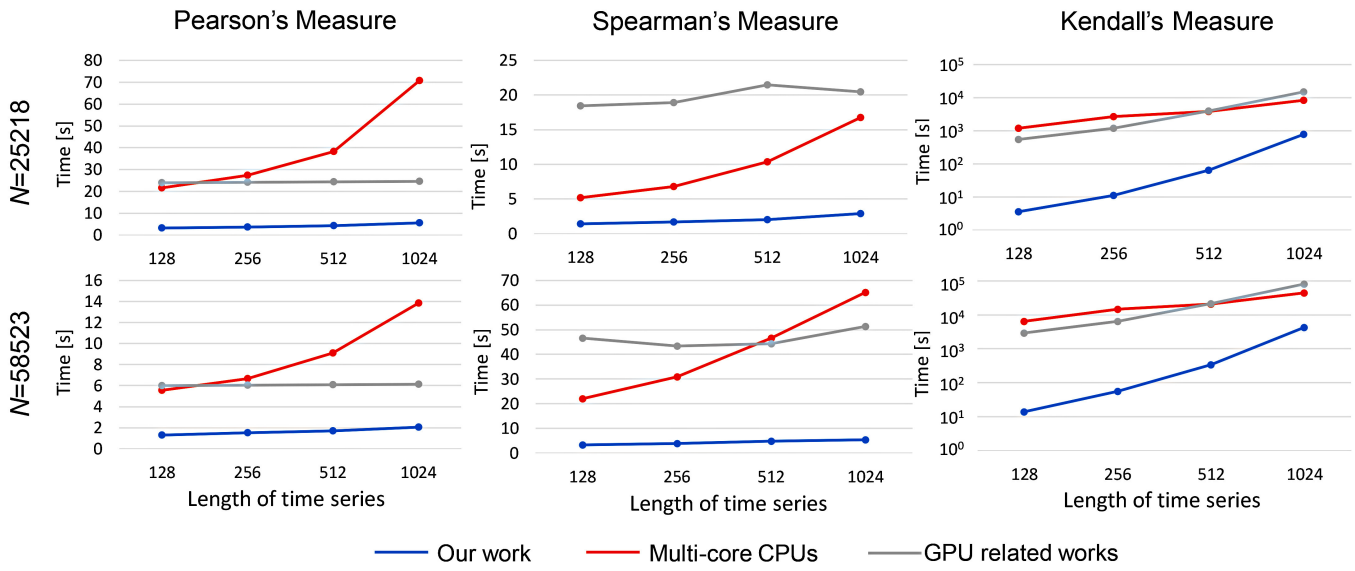


Figure 2. Performance comparisons of different implementations of three FC measures (Pearson's, Spearman's and Kendall's).

complexity as the increase of lengths of time series is generally more time-consuming than that of Pearson's and Spearman's measures with linear span.

Additionally, it was observed that multi-core CPUs implementations of Pearson's and Spearman's measures were adversely sensitive to the length of time series. Moreover, GPU-based related works regarding three measures, due to their lack of specialized treatments to efficiently adapt these measures on the GPU architecture, showed relatively inferior performance to ours, and even sometimes to multi-core CPUs implementations. E.g., for the computation of the Spearman's measure, it took averagely 64 seconds for GPU-based sort-detection algorithm [25] to handle 3mm isotropic resolution (N=58523, L=512) while <40 seconds were needed for the multi-core CPUs implementation accordingly.

B. Scalability

To assess the scalability of the proposed framework, the elapsed time for constructing large-scale functional networks with varied numbers of networks nodes were demonstrated in Table 1. Sparsity thresholds were specified at 0.1%, 1%, 2%, respectively. In particular, for Kendall's network construction that usually takes longer time than Pearson's and Spearman's, the table only listed the time records within 1000s and corresponding track of N while the framework was certainly applicable to larger scale data sets. The results in Table 1 illustrated that our proposed framework were scalable to high resolution data and could establish voxel-wise functional networks with a large amount of nodes in a short period of time. Specifically, the proposed framework could construct Pearson's or Spearman's networks with 10^6 nodes and 1% edge sparsity using less than 1000 seconds. Considering that only about 200,000 nodes need to be maintained in dense connectomes at an isometric resolution of the 2mm level (Figure 1), our framework is capable of handling high resolution data whose voxel size is far lower than 2mm. Moreover, in actual measurements, it was tested that the framework could handle 1mm voxel-resolution data (N=1561152, L=1200, 0.1% sparsity threshold) and finish the Pearson's network construction within an hour. To the best of our knowledge, this is the first work enabling the processing of such magnitude data in an acceptable amount of time. Notably, the elapsed time of our network construction algorithms is affected by both node amounts N and sparsity thresholds of networks. The time complexity is $O(N^2)$ approximately, and the distinct network sparsity thresholds mainly affect the time expenditure of CPU-GPU data transfers.

Finally, it was observed that our Pearson's and Spearman's network construction procedures failed when $N = 10^6$ given the 2% sparsity threshold, as a result of inadequate CPU main memory. Actually, the relation between the sparsity threshold and the corresponding quantity of network nodes that our framework could handle is constrained by:

$$N^2 \cdot Sparsity \leq MainMemory/B, \quad (9)$$

where B represented the number of bytes used to store an element (node or edge) in established networks, and $0 < Sparsity \leq 1$. For example, given B = 4 bytes and $Sparsity$ = 2%, our framework supports up to 92×10^4 nodes in the network construction under the current computing environment with 64GB CPU main memory. By contrast, for related works which did not employ the block-wise thresholding strategy, the CPU main memory is required to be large enough to at least hold the entire connectivity matrix, i.e.,

$$N^2 \leq MainMemory/B, \quad (10)$$

in which case the maximum value of N is far less than that in (9). Under the same condition (64GB main memory), at most 16×10^4 nodes are supportable for those works, no matter how sparse the network is. Comparatively, our framework has the better scalability for the increasing number of network nodes.

IV. CONCLUSION AND FUTURE EXPECTATIONS

In summary, this paper provides a scalable GPU-accelerated framework for the fast mapping of dense functional

 ISBN: 978-1-61208-579-1

TABLE I. THE SCALABILITY DEMONSTRATION OF THE PROPOSED FRAMEWORK.

L=128	Sparsity	N=$20 \cdot 10^4$	$40 \cdot 10^4$	$60 \cdot 10^4$	$80 \cdot 10^4$	$100 \cdot 10^4$
Pearson's	0.1%	31.90	126.57	278.07	505.76	777.21
Measure	1%	34.47	135.82	307.56	546.13	914.09
time:(s)	2%	37.36	146.61	324.51	587.20	-
Spearman's	0.1%	31.66	124.90	280.03	497.58	782.02
Measure	1%	32.03	133.76	308.60	541.77	917.25
time:(s)	2%	36.41	145.01	322.00	591.03	-
L=128	Sparsity	N=$10 \cdot 10^4$	$15 \cdot 10^4$	$20 \cdot 10^4$	$25 \cdot 10^4$	$30 \cdot 10^4$
Kendall's	0.1%	91.01	210.43	386.50	602.77	861.64
Measure	1%	94.31	217.03	395.61	634.54	914.17
time:(s)	2%	95.79	223.40	404.97	640.80	951.33

connectomes based on three commonly used FC measures. The proposed framework significantly accelerated the voxel-wise network construction (speedup>50, Kendall's measure) and could scale up to higher voxel-resolution data (<2mm) against related works. We hope that the present study will serve as a building block to facilitate dense connectome studies. In the future, we expect to implement more FC measures under the current framework, especially those measures that enable the detection of the nonlinear, multivariate and frequency-domain connectivity among brain network nodes [30].

ACKNOWLEDGMENT

This work was supported by the Natural Science Foundation of China (Grant Nos. 61373026 and 61622403).

REFERENCES

[1] O. Sporns, G. Tononi, and R. Ktter, "The human connectome: A structural description of the human brain." Plos Computational Biology, vol. 1, no. 4, 2005, p. 42.

[2] D. C. Van Essen and K. Ugurbil, "The future of the human connectome," Neuroimage, vol. 62, no. 2, 2012, pp. 1299–1310.

[3] K. Loewe, S. E. Donohue, M. A. Schoenfeld, R. Kruse, and C. Borgelt, "Memory-efficient analysis of dense functional connectomes," Frontiers in Neuroinformatics, vol. 10, 2016.

[4] Smith et al., "Resting-state fmri in the human connectome project," Neuroimage, vol. 80, 2013, pp. 144–168.

[5] S. Hayasaka and P. J. Laurienti, "Comparison of characteristics between region-and voxel-based network analyses in resting-state fmri data," Neuroimage, vol. 50, no. 2, 2010, pp. 499–508.

[6] V. Essen et al., "The human connectome project: a data acquisition perspective," Neuroimage, vol. 62, no. 4, 2012, pp. 2222–2231.

[7] M. A. de Reus and M. P. Van den Heuvel, "The parcellation-based connectome: limitations and extensions," Neuroimage, vol. 80, 2013, pp. 397–404.

[8] E. Wu and Y. Liu, "Emerging technology about gpgpu," in Circuits and Systems, 2008. APCCAS 2008. IEEE Asia Pacific Conference on. IEEE, 2008, pp. 618–622.

[9] J. Ghorpade, J. Parande, M. Kulkarni, and A. Bawaskar, "Gpgpu processing in cuda architecture," arXiv preprint arXiv:1202.4347, 2012.

[10] A. Eklund, P. Dufort, D. Forsberg, and S. M. LaConte, "Medical image processing on the gpu–past, present and future," Medical image analysis, vol. 17, no. 8, 2013, pp. 1073–1094.

[11] D. Wu et al., "Making human connectome faster: Gpu acceleration of brain network analysis," in Parallel and Distributed Systems (ICPADS), 2010 IEEE 16th International Conference on. IEEE, 2010, pp. 593–600.

[12] N. S. Chok, "Pearson's versus spearman's and kendall's correlation coefficients for continuous data," Ph.D. dissertation, University of Pittsburgh, 2010.

[13] S. C. Strother, "Evaluating fmri preprocessing pipelines," IEEE Engineering in Medicine and Biology Magazine, vol. 25, no. 2, 2006, pp. 27–41.

[14] E. Bullmore and O. Sporns, "The economy of brain network organization," Nature Reviews Neuroscience, vol. 13, no. 5, 2012, pp. 336–349.

[15] M. Rubinov and O. Sporns, "Complex network measures of brain connectivity: uses and interpretations," Neuroimage, vol. 52, no. 3, 2010, pp. 1059–1069.

[16] S. L. Simpson, F. D. Bowman, and P. J. Laurienti, "Analyzing complex functional brain networks: fusing statistics and network science to understand the brain," Statistics surveys, vol. 7, 2013, p. 1.

[17] N. Bell and M. Garland, "Implementing sparse matrix-vector multiplication on throughput-oriented processors," in Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis. ACM, 2009, p. 18.

[18] K. A. Garrison, D. Scheinost, E. S. Finn, X. Shen, and R. T. Constable, "The (in) stability of functional brain network measures across thresholds," Neuroimage, vol. 118, 2015, pp. 651–661.

[19] B. C. Van Wijk, C. J. Stam, and A. Daffertshofer, "Comparing brain networks of different size and connectivity density using graph theory," PloS one, vol. 5, no. 10, 2010, p. e13701.

[20] U. Milic, I. Gelado, N. Puzovic, A. Ramirez, and M. Tomasevic, "Parallelizing general histogram application for cuda architectures," in Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIII), 2013 International Conference on. IEEE, 2013, pp. 11–18.

[21] C. Croux and C. Dehon, "Influence functions of the spearman and kendall correlation measures," Statistical methods & applications, vol. 19, no. 4, 2010, pp. 497–515.

[22] Y. Wang et al., "A hybrid cpu-gpu accelerated framework for fast mapping of high-resolution human brain connectome," PloS one, vol. 8, no. 5, 2013, p. e62789.

[23] R. Nelsen, "Kendall tau metric," Encyclopaedia of Mathematics, vol. 3, 2001, pp. 226–227.

[24] J. L. Myers, A. Well, and R. F. Lorch, Research design and statistical analysis. Routledge, 2010.

[25] S. Kim, M. Ouyang, and X. Zhang, "Compute spearman correlation coefficient with matlab/cuda," in Signal Processing and Information Technology (ISSPIT), 2012 IEEE International Symposium on. IEEE, 2012, pp. 000 055–000 060.

[26] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with cuda," Queue, vol. 6, no. 2, 2008, pp. 40–53.

[27] E. Wang, Q. Zhang, B. Shen, G. Zhang, X. Lu, Q. Wu, and Y. Wang, "Intel math kernel library," in High-Performance Computing on the Intel® Xeon Phi. Springer, 2014, pp. 167–188.

[28] W. R. Knight, "A computer method for calculating kendall's tau with ungrouped data," Journal of the American Statistical Association, vol. 61, no. 314, 1966, pp. 436–439.

[29] J. Buckner, J. Wilson, M. Seligman, B. Athey, S. Watson, and F. Meng, "The gputools package enables gpu computing in r," Bioinformatics, vol. 26, no. 1, 2010, pp. 134–135.

[30] F. D. V. Fallani, J. Richiardi, M. Chavez, and S. Achard, "Graph analysis of functional brain networks: practical issues in translational neuroscience," Phil. Trans. R. Soc. B, vol. 369, no. 1653, 2014, p. 20130521.

 ISBN: 978-1-61208-579-1

Design and Implementation of Neural Network Based Chaotic System Model for the Dynamical Control of Brain Stimulation

Lei Zhang

Faculty of Engineering and Applied Science
University of Regina
Regina, Canada S4S 0A2
Email: lei.zhang@uregina.ca

***Abstract*—Brain stimulation has been used in practice to treat neurological diseases, such as Parkinson's Disease and Epilepsy. However, the stimulation signals are generated based on trail and error; and the underpinning theory of this treatment is still unclear. Artificial neural network (ANN) resembles biological neural network in the brain and has been used for many artificial intelligence applications such as classification and pattern recognition. In order to generate accurate stimulation signals in brain stimulation treatment, it is beneficial to establish an ANN model to simulate the brain dynamics and study the effects of various stimulation signals. Previous research shows that brain activities captured by Electroencephalogram (EEG) demonstrate chaotic patterns. Chaotic systems, such as Hénon map can be represented by a set of mathematical equations, and therefore are predictable and controllable. The aim of this research is to implement an optimal ANN architecture model to generate the output pattern of a chaotic system, which can be used to simulate the brain dynamics under stimulation. This paper presented the preliminary work of an ANN architecture design and optimization for generating the outputs of Hénon map chaotic system, and the simulation results for controlling the chaotic system with periodic stimulation signals. The ANN design method and chaotic control method can be extended for other chaotic systems in general.**

Keywords–Brain stimulation; Chaotic systems; Artificial Neural networks; Dynamic control; Hénon map.

I. INTRODUCTION

The growing interest in brain stimulation as a form of neuromodulation has led to increased empirical data from clinical practice for further theoretical research. Targeted brain stimulation has been employed to treat neurological diseases, with reported success in controlling shaking in Parkinson's Disease and Epilepsy seizures. However, theoretical and analytical research is urgently in need to discover the impact, especially the potential long-term side effects of these focal perturbations. Therefore, it is critically important to develop theoretical models in order to design high performance dynamical control systems for brain stimulation.

Brain stimulation has been employed clinically to diagnose, monitor and treat neurological disorders, such as epilepsy seizures [1] [2] and Parkinson's disease [3]–[6]. Commonly used non-invasive stimulation methods include transcranial magnetic stimulation (TMS), transcranial direct current stimulation (tDCS) [7] [8] and transcranial focused ultrasound [9]. Various data recording methods and devices, such as positron emission computed tomography (PET) [6], electroencephalograph (EEG) [10], functional magnetic resonance imaging (fMRI) [11] [12], and recently some single devices [13]–[16] have been used to monitor the brain stimulation effects on brain activities. This provides valuable research data for further study to optimize stimulation protocols, which include identifying accurate target stimulation area and applying effective stimuli, in order to improve the performance of the treatment and meanwhile minimize or eliminate the potential side-effects.

Previous research reported that brain waves demonstrate chaotic behaviors [17] [18]. A chaotic system is a bound system which obtains the existence of an attractor. Chaotic time series are dynamic systems that are extremely sensitive to initial conditions and can exhibit complex external behavior. A chaotic system can be stable, periodic or chaotic depending on the system parameters as well as its initial conditions. A known chaotic system can be analyzed and controlled based on its system equations using conventional dynamic control methods. However, when the equations of a chaotic system are unknown, such as the EEG time series signals, the pattern recognition of such a system and the discovery of its system parameters become a challenging task, which is important for the dynamic control of the system.

Artificial Neural Network (ANN) is a leading machine learning method inspired by biological neural network structure. In recent years, ANN has been widely used for pattern recognition and classification based on a number of pre-defined features. An ANN model with a feedback loop can be designed to generate chaotic outputs by training the ANN using the output values of a chaotic system with selected parameters and initial conditions [19]. Since a chaotic system with specified initial values and system parameters can be represented by an ANN model, the system can be controlled by varying the weight and bias values of the ANN. The training process is carried out on a computer and the weights are generated for all neurons in an ANN architecture. These weights are then used in constructing an ANN model to generate the expected outputs for the target chaotic system. The implementation cost and speed of an ANN architecture is determined by its complexity, therefore it is beneficial to use less number of hidden neurons to achieve the target training performance. A simple ANN design generally has a 3-layer architecture, including one input, one hidden and one output layer. The MATLAB Neural Network Toolbox is used to train the ANN with three MATLAB training algorithms: Levenberg-Marquardt, Scaled Conjugate Gradient algorithm and Bayesian Regulation. The optimization of the ANN architecture is important for improving the performance of hardware implementation on a Field Programmable Gates Array (FPGA) device.

 ISBN: 978-1-61208-579-1

Network control theory has been used to study the effects of stimulation on brain networks [20]. Network control refers to the possibility of manipulating local interaction of dynamic components to steer the global system along a chosen trajectory. The neural network based chaotic model can aid the understanding of the underlying structural connectivity that modulates the system dynamics. A novel approach is presented to combine the ANN design and the dynamical control theory for the control of brain dynamics. In this approach, an ANN model is designed with optimized architecture based on Hénon map chaotic system. Hénon map [21] has its significance in studying chaotic systems with a simple 2-dimensional structure and is used initially as the study subject of the research project.

In previous related work, a model-based hardware implementation of the Hénon map is presented in [22] and the dynamic analysis of the system stability at critical points with varying system parameters is provided by [23]. The chaotic system can be controlled to change from chaotic mode to stable or periodic mode by adding a period stimulation pulse signals. The design approach and control method can be easily extended for other chaotic systems in general, such as 3-dimensional Lorenz attractor [24].

Section II describes the ANN design and training procedure; section III explains the chaos control of the Hénon map; section IV discusses the conclusion and future work.

II. ANN Design and Training

The ANN is a network of interconnected neurons arranged in multiple layers, including one input layer, one output layer and one or multiple hidden layers. A general mathematic representation of an individual neuron within an ANN architecture is shown by equation(1).

$$a_j^l = \sum_{i=1}^{N_{l-1}} w_{j,i}^l x_i + b_{j,0}^l \qquad j = 1, 2, ...N_l$$
$$y_j^l = f_l(a_j^l) \tag{1}$$

where N_l is the number of neurons at l-layer. Each hidden neuron j receives the output of each input neuron i from the input layer multiplied with a weight of $w_{j,i}^l$. The sum of all weighted inputs is used by an activation function f_l to produce the output of the hidden layer neuron and feed it forward to the output layer. A similar weighted sum is generated for each output neuron. $b_{j,0}^l$ is the bias of the jth neuron at the lth layer, which are added as noise to randomize the initial condition in order to get better chance to converge. The weight matrix connecting between the $(l-1)^{th}$ layer to the l^{th} layer is represented by equation (2).

$$\mathbf{W}^l = \begin{bmatrix} w_{1,1} & w_{1,2} & ... & w_{1,N_{l-1}} \\ w_{2,1} & w_{2,2} & ... & w_{2,N_{l-1}} \\ ... & ... & ... & ... \\ w_{N_l,1} & w_{N_l,2} & ... & w_{N_l,N_{l-1}} \end{bmatrix}_{N_l \times N_{l-1}} \tag{2}$$

Let $\mathbf{y}^l = (y_1, y_2, ..., y_{N_l})$ be the output vector from the l^{th} layer, the weighted sum vector to the $l+1^{th}$ layer is represented by (3), the outputs are calculated using (4), where f is the activation function.

$$\mathbf{a}_k^{l+1} = \sum_{j=1}^{N_l} \mathbf{y}_j * w_{j,k}, \quad (k = 1, 2, ..., N_{l+1})$$
$$\mathbf{a}^{l+1} = \mathbf{y}^l \mathbf{W}^l y_j^{l+1} = f(a_j^{l+1}), \quad j = 1, 2, ..., N_{l+1} \tag{3}$$

$$\mathbf{y}^{l+1} = f(\mathbf{a}^{l+1}) = \begin{bmatrix} y_1 * w_{1,1} & y_1 * w_{1,2} & ... & y_1 * w_{1,N_{l+1}} \\ y_2 * w_{2,1} & y_2 * w_{2,2} & ... & y_2 * w_{2,N_{l+1}} \\ ... & ... & ... & ... \\ y_{N_l} * w_{N_l,1} & ... & ... & y_{N_l} * w_{N_l,N_{l+1}} \end{bmatrix}_{N_l \times N_{l+1}} \tag{4}$$

A. ANN Architecture Design

The ANN model design process includes identifying the correct topology of the network. An optimal ANN architecture should contain minimal number of hidden layers and hidden neurons, and yet sufficient enough for representing the variability of the training data. A network with insufficient complexity will fail to learn the underlying function, while a network with more neurons and layers than required will cause overfitting of the model and fail to generalize. An ANN model design approach for chaotic systems based on model topology analysis [19] is used. After the construction of the ANN, its predictive ability needs to be measured. The accuracy is a quantification of the proximity between the outputs of the ANN and the target output values, measured by mean square error (MSE).

The implementation performance of the chaotic system depends on the ANN topology. For the Hénon map ANN design, both input and output layer has two neurons, one hidden layer is employed. In order to optimize the ANN architecture with a minimal number of the hidden neurons to improve implementation performance, 16 different ANN topologies with one to sixteen hidden neurons are trained using three training algorithms respectively to find the optimal ANN topology. Sigmoid function is used as the activation function for the neurons in the hidden layer. Ramp activation function is used for the output layer [25].

B. Training Data and Training Algorithms

The 6,000 training samples for Hénon map are generated in MATLAB using the ode23 method. During the training process, the first pair of samples (x_1, y_1) is provided as the 2 inputs of ANN, the second pair of samples (x_2, y_2) is provided as the target outputs. Then the second pair is provided as the inputs and the third pair as the target outputs, and so on. The training samples are divided into three subsets: training(70%), validation(15%) and testings(15%). The training set is used for computing the gradient and updating the network weight and bias values. The validation set is used to monitor the error during training process in order to avoid overfitting. The test set is used to test the training performance. The ANN training is carried out using three network training functions provided by the MATLAB Neuron Network Toolbox: $trainlm, trainbr$ and $trainscg$. $trainlm$ function updates weight and bias values based on Levenberg-Marquardt (LM) optimization [26]. $trainbr$ function also updates weights and biases based on LM optimization. It uses Bayesian Regulation (BR) process [27] to minimize and determine a combination of squared errors and weights, in order to produce a network

TABLE I. ANN TRAINING FUNCTION PARAMETERS

Training Fuctions	LM	BR	SCG
Learning rate	0.01	0.01	0.01
Momentum Constant	0.9	0.9	0.9
Maximum Epochs	1000	1000	1000
Maximum Training Time	inf	inf	inf
Performance Goal	0	0	0
Minimum Gradient	1.00E-07	1.00E-07	1.00E-06
Maximum Validation Checks	6	0	6
Mu	0.001	0.005	N/A
Mu Decrease Ratio	0.1	0.1	N/A
Mu Increase Ratio	10	10	N/A
Maximum mu	1.00E+10	1.00E+10	N/A
Sigma	N/A	N/A	5.00E-05
Lambda	N/A	N/A	5.00E-07

with good generalization. $trainscg$ function updates weight and bias values based on Scaled Conjugate Gradient (SCG) method [28]. The training parameters of these three MATLAB functions are listed in Table. I.

Each algorithm is used for 16 different ANN topologies, with the number of hidden neurons increasing from 1 to 16 in the architecture. Three training iterations are carried out per architecture per algorithm. Each training iteration runs for 1000 epochs. An epoch is a measure of the number of times all of the training vectors are used once to update the weights. The learning rate is 0.01. It is a constant used by the training algorithm to update the weight and bias values at each step. The momentum is 0.9. This is another constant value for adjusting the learning rate by adding a proportion of the weight value in the previous step. As the training time is infinity (inf) and the training goal is 0, the training stops when the performance gradient falls below the Minimum Gradient, or the Maximum Epoch is reached, or the validation performance has increased more than the Maximum Validation Checks since the last time it decreases.

C. ANN Performance Evaluation

The ANN training result is measured by the error between the calculated ANN output y and the target training output $\hat{y}$. The training target is a threshold error value small enough for the output to be considered as correct. The performance of the ANN training process is evaluated by how fast and well the error converge to the target threshold. The most common method for measuring the output error is MSE, as illustrated respectively by (5).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{5}$$

where N is the number of outputs, which is 2 in the case of Hénon map: y_1 and y_2. For 16 architectures and three training algorithms: LM, BR and SCG, the ANN training is carried out three repeated iterations per architecture per algorithm. The training performance (MSE) are listed in Table. II.

The training performance of the Levenberg-Marguardt algorithm is shown in Figure 2(a). It can be observed that the MSE values for all three iterations over the number of hidden neurons have non-monotonicity. Nevertheless, the MSE is decreasing in general. The MSE values for all three iterations decrease below 1.7E-07 $(= 1.7 \times 10^{-7})$ with only 2 hidden neurons. The MSE of iteration I increases when the number of hidden neurons increases from 2 to 3; and the MSE of iteration III has big increase from 2.3E-8 to 8.6E-6 (by 2 logarithmic scales) when the number of hidden neurons increase from 3 to 4. The best performance is achieved with 15 hidden neuron in iteration II, when MSE=1.1397E-10. Although the overall performance is improving with increasing number of hidden neurons, it is hard to predict accurately whether the performance can be improved by adding one more neuron for an individual iteration. The result is random.

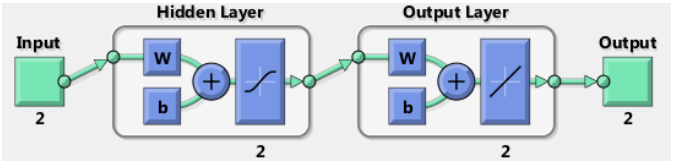


Figure 1. ANN Architecture for Hénon Map Chaotic System

The training performance of the Bayesian Regulation algorithm is shown in Figure 2(b). Similar to the LM algorithm, the MSE values for all three iterations decrease below 1E-07 with only 2 hidden neurons, but increase as the number increases from 2 to 3 for iterations I and III. The overall trend is for the MSE to gradually decrease while the number of hidden neurons increases, but the effect for adding or removing one neuron for each training iteration is unpredictable. The smallest MSE (3.1178E-12) is achieved by iteration III with 15 hidden neurons.

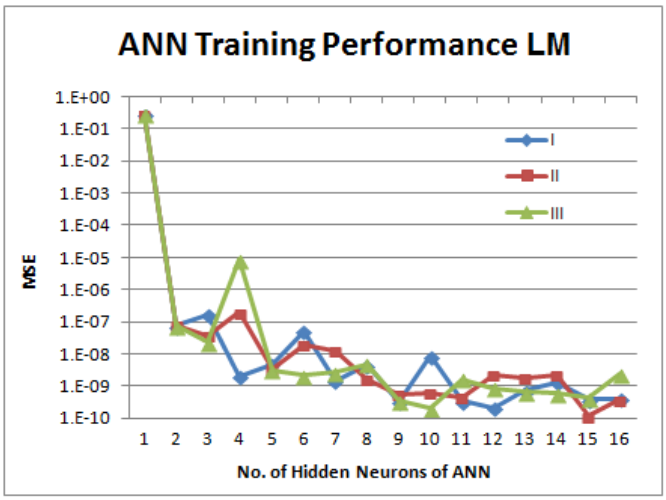
The training performance of the Scaled Conjugate Gradient algorithm is shown in Figure 2(c). The smallest performance (MSE=5.9783E-05) is achieved when n=4. The MSEs increases generally while n increases after n=4. The LM and BR algorithms have better training performance than the SCG algorithm.

The average MSE values of all three iterations for each algorithm is compared in Figure 2(d). The training performance can be only improved slightly once the number of hidden neurons is greater than 2. The increased number of hidden neurons will increase hardware resource utilization for the system implementation. The ANN model is generated using double precision floating point data format during the training process in MATLAB simulation environment. The fixed-point data format for the FPGA implementation does not require the target MSE to be smaller than the quantization error. For instance, the 32-bit fixed-point data format with 18 fractional bits can have $2^{-18}(\approx 3.8147e{-}06)$ resolution, which is bigger than the smallest MSE achieved by the ANN model with 2 hidden neurons using the LM and BR training algorithms. Therefore, the ANN model is designed using 2 hidden neurons. The ANN architecture is illustrated by Figure 1, including an input layer with two inputs, a hidden layer with two hidden neuron, and an output layer with two outputs. A Simulink Model is created using the ANN with delayed feed-back loop as shown in Figure 3; and the Simulink simulation output of the ANN-based Hénon map chaotic system is shown in Figure 4. The training performances and training states for the three training functions of the selected architecture with 2 hidden neurons are plotted in Figure 5. For LM and BR, the training stops at epoch 1000. For the SCG, the training stops at epoch 255 as the maximum number of validation fails reaches 6.

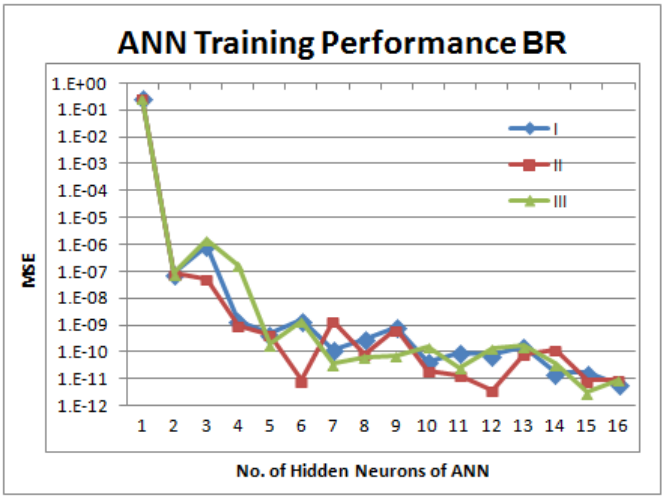
 ISBN: 978-1-61208-579-1

TABLE II. TRAINING PERFORMANCE (MSE) OF 3 TRAINING ALGORITHMS WITH 16 ARCHITECTURES – EPOCH 1000

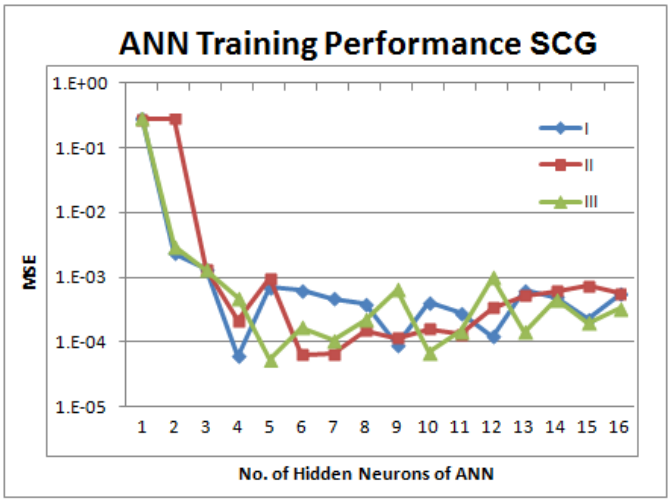
MSE	Levenberg-Marguardt (LM)			Bayesian Regulation (BR)			Scaled Conjugate Gradient (SCG)		
Neurons	I	II	III	I	II	III	I	II	III
1	0.2810	0.2810	0.2810	0.2810	0.2810	0.2810	0.2810	0.2810	0.2828
2	7.5920E-08	7.5411E-08	7.5841E-08	8.1986E-08	8.1497E-08	8.1474E-08	2.3649E-03	2.8101E-01	2.9964E-03
3	1.7042E-07	3.7541E-08	2.3484E-08	9.1948E-07	5.4859E-08	1.5598E-06	1.3356E-03	1.3299E-03	1.3042E-03
4	2.0187E-09	1.9369E-07	8.5939E-06	1.4455E-09	9.3581E-10	1.7478E-07	5.9783E-05	2.1042E-04	4.8217E-04
5	4.7927E-09	3.5006E-09	3.1200E-09	4.7033E-10	4.6482E-10	1.9332E-10	6.9701E-04	9.5137E-04	5.3533E-05
6	5.2199E-08	1.8775E-08	2.1202E-09	1.5030E-09	8.8459E-12	1.4452E-09	6.2341E-04	6.2949E-05	1.6708E-04
7	1.4853E-09	1.2016E-08	2.7629E-09	1.2308E-10	1.4387E-09	3.5900E-11	4.6338E-04	6.5173E-05	1.0628E-04
8	4.5645E-09	1.6143E-09	4.8532E-09	2.9488E-10	8.4649E-11	6.2597E-11	3.8860E-04	1.4984E-04	2.1843E-04
9	3.2232E-10	5.3065E-10	3.3891E-10	8.5851E-10	7.0971E-10	7.0933E-11	8.9483E-05	1.1520E-04	6.4021E-04
10	8.5592E-09	6.0050E-10	1.9888E-10	4.2823E-11	1.9704E-11	1.6047E-10	4.0709E-04	1.5928E-04	6.7822E-05
11	3.4364E-10	4.5139E-10	1.6145E-09	9.6083E-11	1.3954E-11	2.6842E-11	2.7726E-04	1.2903E-04	1.4429E-04
12	2.0994E-10	2.1447E-09	8.7262E-10	7.7306E-11	3.9343E-12	1.4594E-10	1.2205E-04	3.3368E-04	9.9476E-04
13	7.4649E-10	1.8049E-09	6.6516E-10	1.5392E-10	8.2397E-11	1.6851E-10	6.2968E-04	5.2315E-04	1.4134E-04
14	1.3254E-09	2.2304E-09	5.6996E-10	1.6843E-11	1.1866E-10	3.6662E-11	4.8872E-04	6.1157E-04	4.5022E-04
15	3.8671E-10	1.1397E-10	4.1528E-10	1.6680E-11	8.8259E-12	3.1178E-12	2.2542E-04	7.3086E-04	1.9543E-04
16	4.1175E-10	3.7105E-10	2.3944E-09	6.6127E-12	9.4176E-12	8.9851E-12	5.4813E-04	5.5410E-04	3.2486E-04



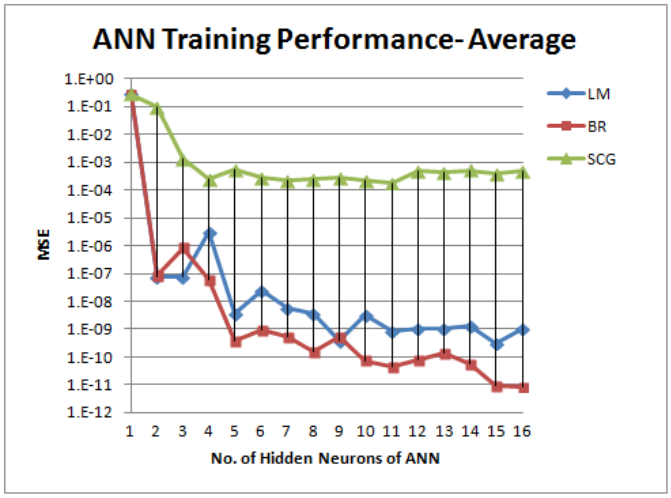
(a) Levenberg-Marquardt



(b) Bayesian Regulation



(c) Scaled Conjugate Gradient



(d) Average of 3 Iterations

Figure 2. ANN Training Performance for Hénon Map

 ISBN: 978-1-61208-579-1

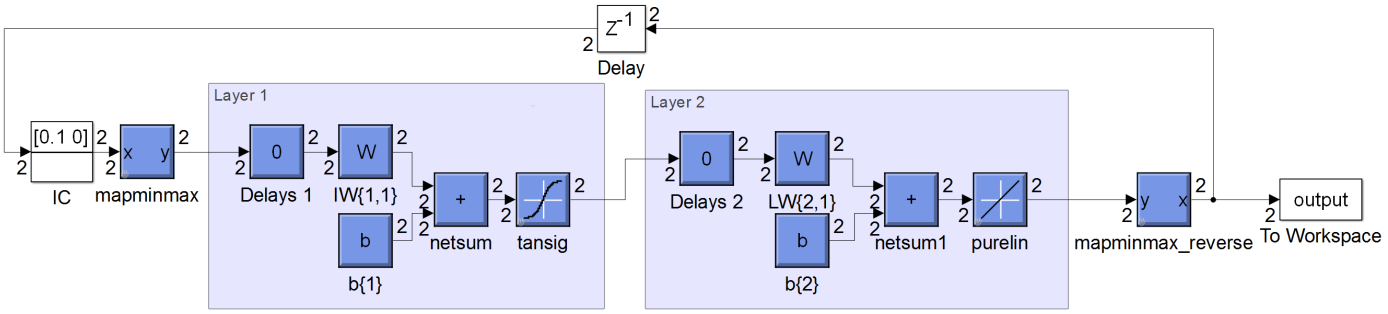


Figure 3. Simulink Model for ANN-based Hénon Map Chaotic System

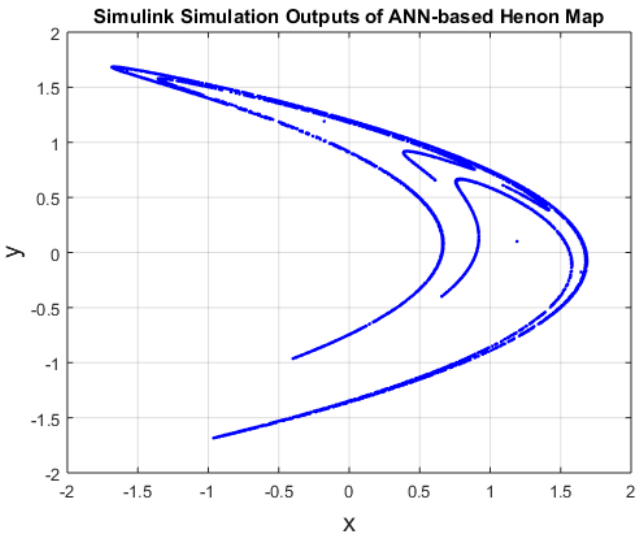


Figure 4. Simulink Simulation Outputs of ANN Model (6000 samples)

III. Chaos Control of the Hénon Map

In order to control chaotic systems for brain stimulation, periodic pulses can be generated as stimuli. The stability of a chaotic system at its critical points can be analyzed by calculating the eigenvalues of the Jacobian matrix of its system equations. Based on the analysis, stimulation signals can be generated to alter the state of the chaotic system.

A. Periodic Orbits and Critical Points

Given a 2-dimensional discrete system $X_{n+1} = F(X_n)$, which can be represented by (6).

$$x_{n+1} = P(x_n, y_n), \qquad y_{n+1} = Q(x_n, y_n) \tag{6}$$

where X is a vector in $\mathbf{R^2}$; F is a map of a domain D of $\mathbf{R^2}$ onto itself; P and Q are scalar valued functions. A critical point of a system is defined as a point at which $X_{n+1} = F(X_n) = X_n$ for all n [29]. The term 'critical point' is often referred to as 'fixed point' of dynamic system. The type of critical point is determined from the eigenvalues of the Jacobian matrix of the function $F(X)$ at the critical point. A critical point of period N is a point at which $X_{n+N} = F^N(X_n) = X_n$, for all n. Given that a discrete nonlinear system represented by (6) has a critical point at (x_s, y_s), the Jacobian matrix at the critical point can be represented by (7).

$$J(x_s, y_s) = \begin{pmatrix} \frac{\partial P}{\partial x} & \frac{\partial P}{\partial y} \\ \frac{\partial Q}{\partial x} & \frac{\partial Q}{\partial y} \end{pmatrix}\Bigg|_{(x_s, y_s)} \tag{7}$$

Suppose that the Jacobian matrix has eigenvalues λ_1 and λ_2. In the discrete case, the critical point is stable as long as $|\lambda_1| < 1$ and $|\lambda_2| < 1$, otherwise the critical point is unstable. The critical points of period one X_s satisfies $X_s = F(X_s)$.

The discrete Hénon map equations by definition are lised in (8).

$$x_{n+1} = 1 + y_n - ax_n^2, \qquad y_{n+1} = bx_n \tag{8}$$

where x_n and y_n are system variables, α and β are system parameters. A reformed equivalent set of equations are obtained by taking a transformation $x'_n = \frac{1}{\alpha}x_n, y'_n = \frac{\beta}{\alpha}y_n$, as listed in (9). It is easier to use the reformed equations for stability analysis and control [23].

$$x_{n+1} = \alpha + \beta y_n - x_n^2, \qquad y_{n+1} = x_n \tag{9}$$

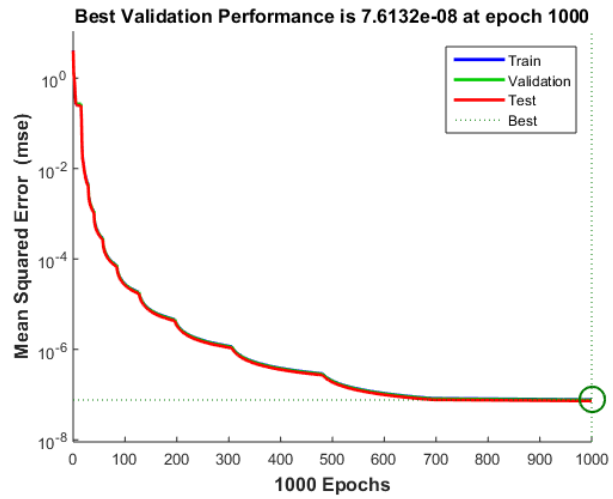
The critical points of period one and the Jacobian matrix for the original Hénon Map are listed in (10).

$$\begin{aligned} x_s =& \frac{(\beta - 1) \pm \sqrt{(1-\beta)^2 + 4\alpha}}{2\alpha} \\ y_s =& \beta \left(\frac{(\beta - 1) \pm \sqrt{(1-\beta)^2 + 4\alpha}}{2\alpha} \right) \\ J =& \begin{pmatrix} -2ax & 1 \\ \beta & 0 \end{pmatrix} \end{aligned} \tag{10}$$

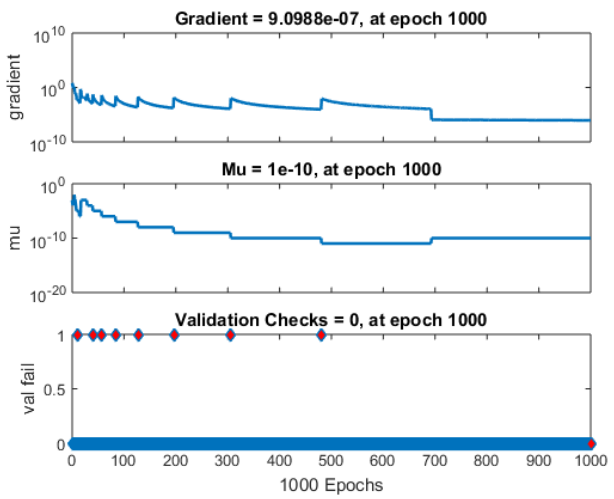
The critical points, Jacobian matrix and its two eigenvalues of the reformed Hénon map equations are listed in (11).

$$\begin{aligned} x_1 =& y_1 = \frac{\beta - 1 - \sqrt{(\beta-1)^2 + 4\alpha}}{2} \\ x_2 =& y_2 = \frac{\beta - 1 + \sqrt{(\beta-1)^2 + 4\alpha}}{2} \\ J =& \begin{pmatrix} -2x & \beta \\ 1 & 0 \end{pmatrix} \\ \lambda_{1,2} =& \begin{pmatrix} \sqrt{x^2 + \beta} - x & 0 \\ 0 & -\sqrt{x^2 + \beta} - x \end{pmatrix} \end{aligned} \tag{11}$$

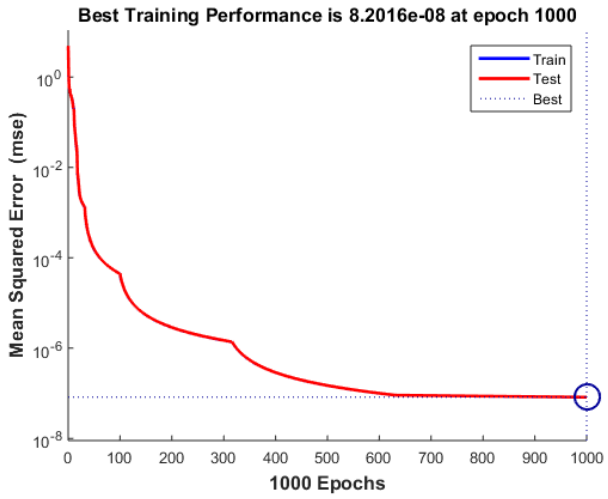
 ISBN: 978-1-61208-579-1



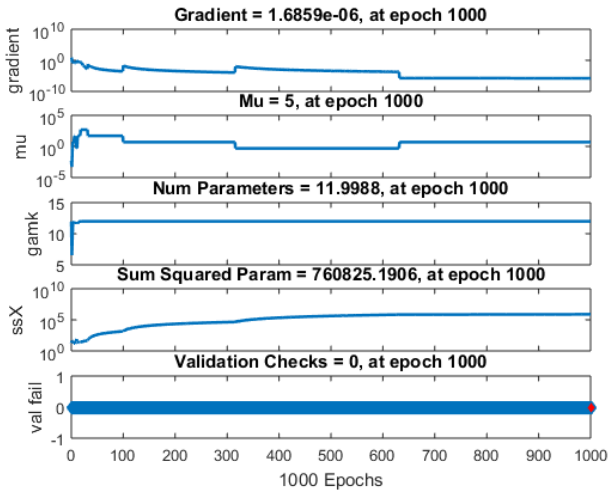
(a) Levenberg-Marquardt Training Performance



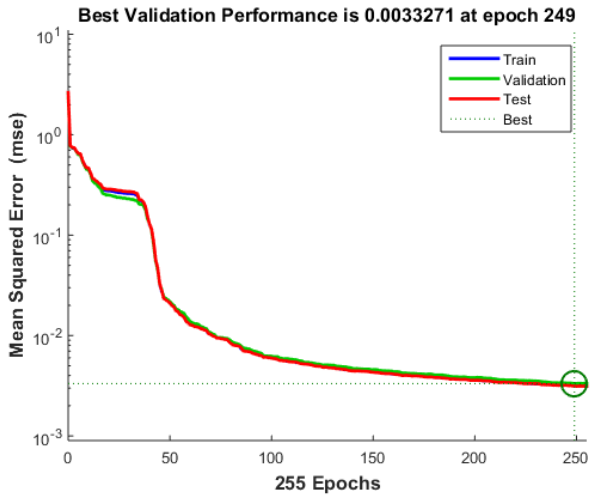
(b) Levenberg-Marquardt Training States



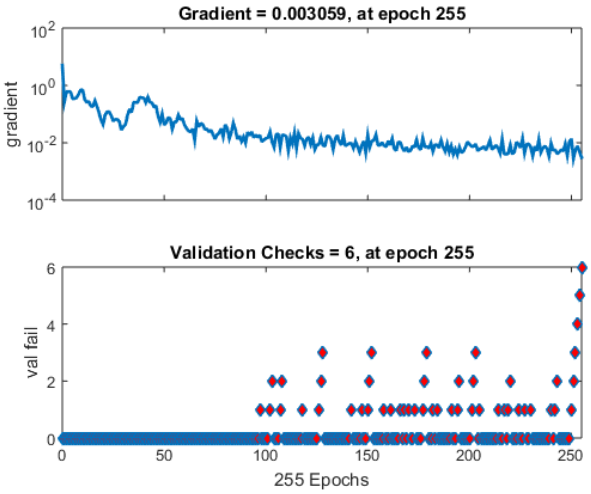
(c) Bayesian Regulation Training Performance



(d) LBayesian Regulatio Training States



(e) Scaled Conjugate Gradient Training Performance

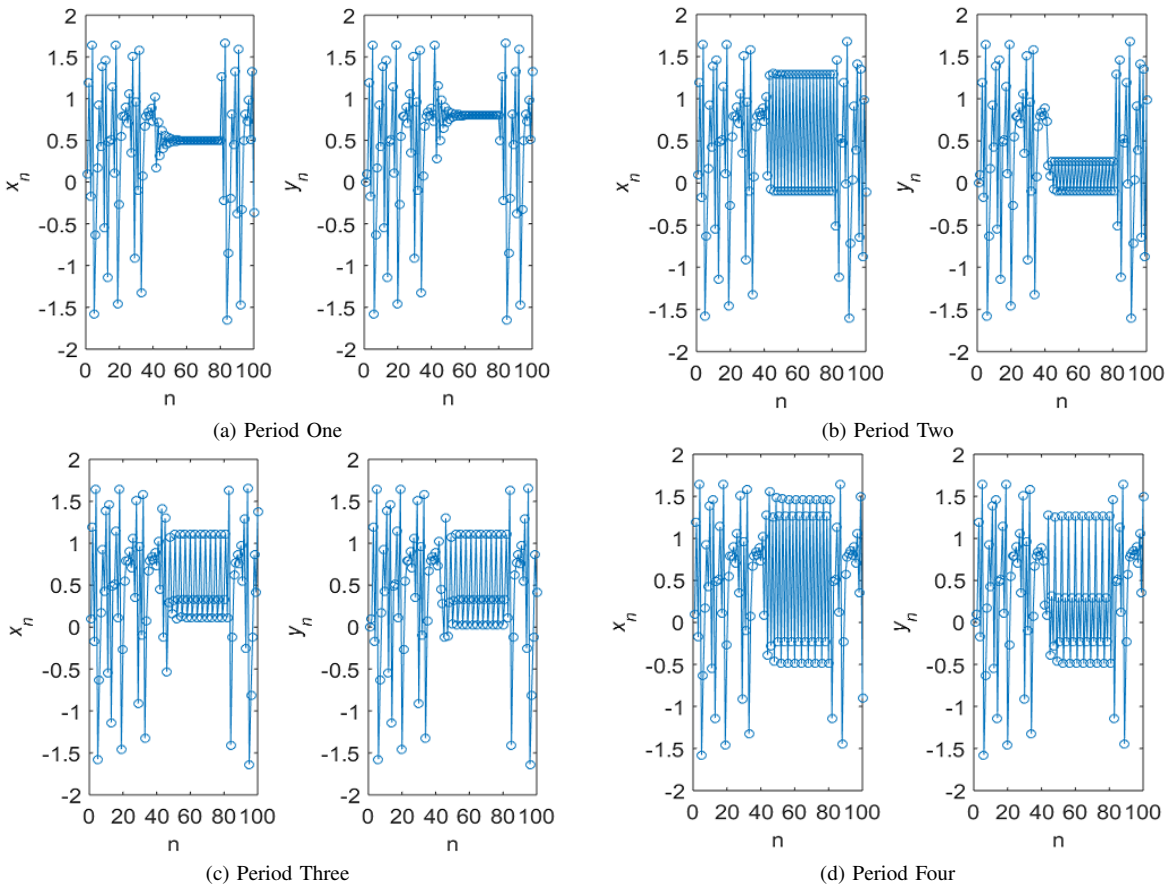


(f) Scaled Conjugate Gradient Training States

Figure 5. ANN Training Performance for ANN with 2 Hidden Neurons

The Hénon map has two real critical points of period one if and only if $(1-\beta)^2 + 4\alpha > 0$. Therefore $\alpha > 0, |\beta| < 1$. The determinate of the Jacobian matrix is $|\beta|$, which is less than 1.

 ISBN: 978-1-61208-579-1



(a) Period One (b) Period Two

(c) Period Three (d) Period Four

Figure 6. MATLAB Simulation for Hénon Map Chaos Control. $\alpha = 1.2, \beta = 0.4, k = 0.2$; Control period: $40 < n < 80$; Initial values: $x_0 = 0.1, y_0 = 0$

B. Chaos Control Using Periodic Proportional Pulses Method

Control and synchronization of chaotic system using the *periodic proportional pulses* method is evaluated in [29] [30] and can be adapted for the control of brain stimulation. Instantaneous pulses can be applied to the system variables X_n every N iterations. Define the composite function $C = KF^N$. K is the diagonal matrix with diagonal elements k_1 and k_2, which can be derived for a given period N and a given critical point X_s. A critical point $X_s = (x_s, y_s)$ of the function F satisfies $KF^N(X_s) = (X_s)$. The Jacobian(J) of C has two eigenvalues. The critical point is locally stable if both the modulus of eigenvalues are less than one. The absolute of the determinate of $J : |det(J)| = |\beta| < 1$ indicates that the Hénon chaotic system is a attractor.

Practically, this method can be easily applied when dealing with chaos control with periodic orbits of low periods. Figure 6 shows time series signals with period-one, period two, period three and period-four stimulation respectively.

IV. Conclusion and Future Works

The paper presented an ANN-based Hénon map chaotic system model design and the optimization of the ANN architecture with the consideration for FPGA hardware implementation. The ANN-based chaotic system model is designed for simulating chaotic brain activities and the dynamic control of the chaotic system by varying the weight and bias values of the ANN. The ANN has a 3-layer architecture and is designed using 16 different topologies with 1 to 16 hidden neurons respectively. It is shown that the simple topology of 2 hidden neurons can be implemented to generate the desired chaotic outputs, which is highly beneficial to improve performance and reduce resource utilization for hardware implementation. The ANN model was trained by 3 MATLAB training functions: $trainlm$, $trainbr$ and $trainscg$, which can be used to generate Simulink model for simulation, and for the further FPGA hardware model development using the Xilinx System Generator in the MATLAB software environment. The ANN design approach with topology optimization can be extended to other chaotic systems and will be used for the design and development of a dynamic control system for non-invasive brain stimulation in order to treat neurological disease. The chaotic system analysis and control is also discussed based on Hénon map, which can be extended to other chaotic systems. The control simulation results are presented as preliminary research work and will be applied to the ANN-based chaotic system control for brain stimulation. Future work includes the hardware implementation of the designed ANN on FPGA device. The ANN design and optimization approach will be used with other state-of-the-art training algorithms to compare the training speed and training performance in order to develop an optimal online training algorithm for hardware implementation.

 ISBN: 978-1-61208-579-1

REFERENCES

[1] V. C. Terra et al., "Vagus nerve stimulator in patients with epilepsy: indications and recommendations for use," Arquivos de Neuro-Psiquiatria, vol. 71, no. 11, nov 2013, pp. 902–906.

[2] C. M. DeGiorgio and S. E. Krahl, "Neurostimulation for drug-resistant epilepsy," CONTINUUM: Lifelong Learning in Neurology, vol. 19, jun 2013, pp. 743–755.

[3] T. D.-B. S. for Parkinson's Disease Study Group, "Deep-brain stimulation of the subthalamic nucleus or the pars interna of the globus pallidus in parkinson's disease," New England Journal of Medicine, vol. 345, no. 13, sep 2001, pp. 956–963.

[4] M. C. Rodriguez-Oroz, "Bilateral deep brain stimulation in parkinson's disease: a multicentre study with 4 years follow-up," Brain, vol. 128, no. 10, jul 2005, pp. 2240–2249.

[5] G. Deuschl et al., "A randomized trial of deep-brain stimulation for parkinson's disease," New England Journal of Medicine, vol. 355, no. 9, aug 2006, pp. 896–908.

[6] S. T. Grafton et al., "Normalizing motor-related brain activity: Subthalamic nucleus stimulation in parkinson disease," Neurology, vol. 66, no. 8, apr 2006, pp. 1192–1199.

[7] S. R. Soekadar, M. Witkowski, E. G. Cossio, N. Birbaumer, S. E. Robinson, and L. G. Cohen, "In vivo assessment of human brain oscillations during application of transcranial electric currents," Nature Communications, vol. 4, jun 2013.

[8] L. D. J. Fiederer et al., "Electrical stimulation of the human cerebral cortex by extracranial muscle activity: Effect quantification with intracranial EEG and FEM simulations," IEEE Transactions on Biomedical Engineering, vol. 63, no. 12, dec 2016, pp. 2552–2563.

[9] K. Yu, A. Sohrabpour, and B. He, "Electrophysiological source imaging of brain networks perturbed by low-intensity transcranial focused ultrasound," IEEE Transactions on Biomedical Engineering, vol. 63, no. 9, sep 2016, pp. 1787–1794.

[10] F. Ferreri and P. M. Rossini, "TMS and TMS-EEG techniques in the study of the excitability, connectivity, and plasticity of the human motor cortex," Reviews in the Neurosciences, vol. 24, no. 4, jan 2013.

[11] A. Pascual-Leone et al., "Characterizing brain cortical plasticity and network dynamics across the age-span in health and disease with TMS-EEG and TMS-fMRI," Brain Topography, vol. 24, no. 3-4, aug 2011, pp. 302–315.

[12] J. Leitao, A. Thielscher, S. Werner, R. Pohmann, and U. Noppeney, "Effects of parietal TMS on visual and auditory processing at the primary cortical level - a concurrent TMS-fMRI study," Cerebral Cortex, vol. 23, no. 4, apr 2012, pp. 873–884.

[13] H. Kim, P. Yves, and K. Lee, "Development of chip-less and wireless neural probe functioning stimulation and reading in a single device," Microelectronic Engineering, vol. 158, jun 2016, pp. 118–125.

[14] "Device and circuitry for controlling delivery of stimulation signals," US Patent 9 289 608, 2016.

[15] M. Parastarfeizabadi, A. Z. Kouzani, I. Gibson, and S. J. Tye, "A miniature closed-loop deep brain stimulation device," in 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). Institute of Electrical and Electronics Engineers (IEEE), aug 2016.

[16] Y. Su, S. Routhu, K. Moon, S. Lee, W. Youm, and Y. Ozturk, "A wireless 32-channel implantable bidirectional brain machine interface," Sensors, vol. 16, no. 10, sep 2016, p. 1582.

[17] E. Pereda, R. Q. Quiroga, and J. Bhattacharya, "Nonlinear multivariate analysis of neurophysiological signals," Progress in Neurobiology, vol. 77, no. 1-2, sep 2005, pp. 1–37.

[18] R. Falahian, M. Mehdizadeh Dastjerdi, M. Molaie, S. Jafari, and S. Gharibzadeh, "Artificial neural network-based modeling of brain response to flicker light," Nonlinear Dynamics, vol. 81, no. 4, 2015, pp. 1951–1967.

[19] L. Zhang, "Artificial Neural Network Model Design and Topology Analysis for FPGA Implementation of Lorenz Chaotic Generator," in 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), Apr. 2017, pp. 216–219.

[20] S. Gu et al., "Controllability of structural brain networks," Nature Communications, vol. 6, oct 2015, p. 8414.

[21] M. Hénon, "A two-dimensional mapping with a strange attractor," Communications in Mathematical Physics, vol. 50, no. 1, Feb 1976, pp. 69–77. [Online]. Available: http://dx.doi.org/10.1007/BF01608556

[22] L. Zhang, "Fixed-point FPGA Model-Based Design and Optimization for Hénon Map Chaotic Generator," in Latin American Symposium of Circuits and Systems (LASCAS), Feb. 2017, pp. 105–108.

[23] L. Zhang, "Hénon Map Chaotic System Analysis and VHDL-based Fixed-point FPGA Implementation for Brain Stimulation," in 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE), Apr. 2017, pp. 808–811.

[24] L. Zhang, "System Generator Model-based FPGA Design Optimization and Hardware Co-simulation for Lorenz Chaotic Generator," in 2017 2nd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS 2017), Wuhan, Jun. 2017, pp. 170–174.

[25] L. Zhang, "FPGA Implementation of Fixed-point Neuron Models with Threshold, Ramp and Sigmoid Activation Functions," in 2017 4th International Conference on Mechanics and Mechatronics Research (ICMMR 2017), Xian, Jun. 2017.

[26] M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the marquardt algorithm," IEEE Transactions on Neural Networks, vol. 5, no. 6, Nov 1994, pp. 989–993.

[27] F. D. Foresee and M. T. Hagan, "Gauss-newton approximation to bayesian learning," in Neural Networks,1997., International Conference on, vol. 3, Jun 1997, pp. 1930–1935 vol.3.

[28] M. F. Moller, "A scaled conjugate gradient algorithm for fast supervised learning," NEURAL NETWORKS, vol. 6, no. 4, 1993, pp. 525–533.

[29] S. Lynch, Dynamical Systems with Applications using MATLAB. Springer International Publishing, 2014.

[30] N. Chau, "Controlling chaos by periodic proportional pulses," Phys. Lett., vol. A 234, 1997, p. 193197.

 ISBN: 978-1-61208-579-1