Graduate Thesis for Technical Artificial Intelligence

Using an Artificial Neural Network for Predicting Reaction Time Based on Physiological Data

Author:
Shoshannah Tekofsky

Supervisors:
Dr. Henry Prakken
M.Sc. Nanja Smets

May 31, 2012

Abstract

In this thesis findings are described in relation to the accuracy of predicting an individual's reaction times based on his physiological variables with the use of an Artificial Neural Network. It also includes a reflection on the feasibility of this method. In the course of this exploratory research 5 participants were tested on a custom reaction time task as their Electrocardiographs (ECG), Galvanic Skin Response, and the activity of the masseter (jaw) and corrugator (eyebrows) muscles in the face were monitored. Using cross-validation, it was found that mean error rates are around the 0.3 mark, and minimum errors are often near-zero. This is a very promising result and merits further exploration, as such a system of reaction time prediction can function as a safe-guard for people working in highly-demanding situations. This technique might be able to timely advise them about a lower expected performance from themselves before they decide to take up any dangerous or demanding tasks. This can be relevant for professions like astronaut, firefighter or surgeon. Throughout the paper, astronauts are the focus group, as this research was inspired upon the premise to enrich the MECA tool suite that aims to assist astronauts on long-duration missions.

Contents

Preface

Welcome, reader. Before we dive into the depths of my first major academic foray, I would like to take a moment to thank all the people that supported me on this journey. First off all, my supervisors: Henry Prakken for being so flexible to allow me to pick up this research opportunity at TNO, Jurriaan van Diggelen for teaching me so much about large-scale research as part of the MECA project, and Nanja Smets for supporting me on my own research track and always suggesting ways to make things work out even better. I would also like to thank Mark Neerincx, Olaf Binsche, Anne-Marie Brouwer, Pjotr van Amerongen, Martin van Schaik, and Paul Miller for taking the time to support my research, each in their own way, even though they were under no obligation to do so. Thank you.

A different kind of thanks goes out to the following people. First of all, thank you, Gwen Ferdinandus and Armon Toubman, for the camaraderie at TNO and the sanity-preserving chocolate milk breaks. Another heartfelt thank you to my family, for their forbearance and support through the tough times, and their cheerleading through the good times. And lastly, a simple and sincere "thank you" to my boyfriend, Andreas Lundgren, for being my "rock in the surf", during this research project, and in life :)

1 Introduction

Astronauts are under a tremendous amount of stress during a mission. Such intense and/or prolonged stress has been known to lower performance and overall health [19]. Astronauts are highly trained individuals who are in great part selected on their ability to deal with complex and demanding situations [12] [16]. However, as space missions become more and more demanding, natural resilience will meet its limits. Additionally, more ambitious missions will last longer and go farther out from Earth, hampering communications and support from home. Computerized solutions are the perfect candidate to fill in this growing gap in astronaut support. However, in an environment where the smallest malfunction can mean the death of the whole crew and the failure of the mission, it is logical that the space industry has always been wary of empowering electronic solutions.

For this reason there is currently not much research or progress in the field of electronic psychological support for astronauts. However, two very notable exceptions have inspired our research. Firstly, the Psychomotor Vigilance Task (PVT) [27] is a tool used by NASA to test the alertness of the astronaut before he tries to perform a demanding task. It is a 3-10 minute reaction time performance test that gives a reasonably accurate indication of the astronaut's alertness. The second relevant research was done for the US Air Force. Wilson & Russell [28] looked at the possiblity of having artificial neural networks (ANN) classify the work load an individual is experiencing based on his physiological data. They found a classification accuracy of more than 80% over three categories of work load intensity.

This research falls within the Mission Executive Crew Assistant (MECA) project. This is a tool designed to help alleviate and mediate the increasing demands on the astronaut while conforming to the Human in the Loop philosophy of design. Taking the wisdom of the two previously mentioned research lines together, we came up with the concept of predicting performance with an ANN based on physiological data as an addition to the MECA tool suite. We hope that this approach will prove to be more accurate and less time-consuming than previous initiatives, while fully keeping "the human in the loop" by allowing only a prediction to be made. The operator can decide himself what to do with this information.

We will first expand on the problem we are facing, and then carefully indicate the issues and considerations that we have taken as our focus in our search for a viable and innovative solution. Lastly, we will wrap up this introduction with an overview of what the rest of this paper has to offer.

1.1 Problem Description

Astronauts on extraterrestrial missions are under an immense amount of pressure, both physically and psychologically. More ambitious missions will increase these demands on them, while the price of failure is very high both in terms of financial investments and human life. On any manned space mission, the astronauts will have to deal with such challenges as staying fit in a confined, zero-gravity environment, staying mentally balanced while in profound isolation with a handful of other people for extended periods of time, performing repetitious check-ups and procedures while staying sharp enough to deal with unexpected circumstances. Therefore, it is important to develop ways to ensure that these individuals stay at peak performance in a profoundly challenging environment for extended periods of time.

1.2 MECA

MECA was developed for ESA by a collaboration of TNO, S&T, EADS-Astrium, and OK-systems, and aims to be a software tool that will help astronauts weather the stresses and challenges of interplanetary space travel. Within this development, the focus lies on discovering what shape such a tool should take to give maximum support to the astronauts. MECA should assist astronauts in dealing effectively with the challenges mentioned above, with a particular focus on psychological support. Neerinckx [13] stresses the role that such support plays in keeping astronauts at peak performance during their missions, as small errors due to mental stressors or otherwise, can lead to catastrophic consequences for the mission and the human lives involved. He states that

> "[...] crew support needs are increasing, starting from usability, via cognitive support to partnership"

and goes on to explain how MECA can fulfill this need as it is being iteratively tailored to this use based on empirical measures. However, testing MECA in the actual target situation (space travel) is unfeasible. As such, analogous environments are used for all the different aspects of the tool. Previously, tests have been run in the volcanic area of the Eifel as well as during the Mars520 [21] project in Moscow where a whole trip back and forth to Mars was simulated.

The research described in this paper aims to test the viability of adding an ANN-based performance prediction feature to the MECA tool suite. The next section will outline our focus in more detail.

1.3 Our Focus

In this paper we will constrain ourselves to one specific but crucial aspect of the spectrum of challenges an astronaut faces in outerspace: stress and its effect on performance. This is of course a very general and complex issue that we will more deeply delve into in Section 2. The reason for this focus is because stress unifies and pervades all the challenges the astronaut faces. As will be farther explained in the aforementioned section, negative stress is known to lower work performance, both in terms of quantity and quality. Minimizing this form of stress will prevent possible catastrophic losses of life or material, and increase the chances of successfully performing a mission.

Looking at the problem of assisting the astronaut in dealing with the stress of extraterrestrial life while keeping performance up, we found that the domain limits us in two crucial ways. Firstly, astronauts have a lot of demands on their time, so any assistance should aim to have a high payoff per time spent. Secondly, as space missions become more ambitious, contact with Earth will become more and more difficult. As such, the type of assistance should also not depend on contact with Earth. Taking these two factors together, time-limited computer-based assistance would provide the most benefits compared to methods depending on contact with Earth or time-consuming personal training. To fill this gap we envision an ANN learning to make predictions on the performance of an astronaut based on his physiological data. We have chosen reaction time as a quantifiable, relevant and feasible measure of performance.

Thus the question we will seek to answer is: How well can an ANN predict an individual's reaction time based on the physiological data from that person? We expect that an ANN will be able to gain a high enough accuracy of prediction to become a reliable safe-guard system for astronauts on duty.

1.4 Overview of this Paper

Section 2 will cover all the background information relevant to tying this research to the grander scale of astronaut needs, the nature of stress, and the (AI) algorithms used in similar areas of research. Section 3 will outline the methods of this research, outlining the experimental setup we used to gain our data, how we processed the data and why we decided on that approach. Section 4 describes the results we obtained from applying the ANN to the experimental data. Section 5 will discuss points of improvement and options for future research. Lastly, Section 6 sums up the conclusions we have been able to draw from our research.

2 Background

This research aims to improve stress management for astronauts during long-duration extraterrestrial missions by evaluating the potential of ANNs to make predictions about performance based on the physiological data of the astronaut. To put this endeavor into perspective, we will look at what characterizes astronauts; what causes them stress during a mission; what we know about stress in general; how stress relates to performance and how it is expressed physically; what techniques have already been used to make predictions based on physiological data; and what the basic concept is behind the ANN approach.

2.1 Astronauts

The user base we are targeting in this research is that of astronauts. Astronauts are a very particular breed of people that diverge from the general populace on a number of important points. Musson, Sandal & Helmreich [12] performed an extensive research that gives a clear overview of the personality type of

an astronaut. Their research looks at 259 NASA astronauts from the period of 1989 to 1995. These individuals completed the Personality Characteristic Inventory (PCI) that showed them all to score very high on Instrumentality (goal-directed, motivated behavior), and Expressivity (interpersonal orientated, socially engaged behavior). 147 participants also completed a Big Five measure that showed them to rank very highly on all scales except Neuroticism. The researchers performed an analysis on the results that showed that these traits clustered to the traditional "Right Stuff", "Wrong Stuff", and "No Stuff" clusters. Here the "Right Stuff" is characterized by positive instrumentality and expressivity, the "Wrong stuff" is defined by negative instrumentality (ie. domineering) and/or expressivity (ie. complaining), and "No Stuff" is simply the absence of any strong instrumentality and/or expressivity values. Astronauts were consistently selected on having the "Right Stuff".

These results parallel those on other high-performance jobs [12] as well as the average profile for a productive employee within the general populace. This profile is based on the Big Five inventory and shows that high Conscientiousness is most strongly correlated with work performance [17], followed by low Neuroticism. Jobs that demand regular social contact also have a better fit with people who score high on Agreeableness. All this makes an intuitive type of sense when you see that this personality profile describes a motivated, organized, emotionally stable, and social individual as performing well both in general work situations as well as in the position of an astronaut. However, there is also a strong indication that Conscientiousness, the strongest factor in all the above results, might actually be detrimental to astronaut performance. Palinkas [14] found that in polar over-winter crews Conscientiousness was negatively correlated with job performance and general ability to cope. He suggests that this is caused by the fact that people need to be flexible in their approach to their tasks when surviving in dangerous and challenging environments like the polar regions and space. People ranking high on Conscientiousness have too much need for organization, control and achievement, all of which become frustrated in these dynamic and difficult environments. This contradicts the previously mentioned finding that Conscientiousness is actually a helpfull characteristic and for that reason remains a contentious point in the field.

All in all, it is clear that astronauts excel on all the dimensions that any productive employee would, with the notable enigma of Conscientiousness. On the one hand this trait embodies the motivation and organizational skills of an individual, but it also results in a level of inflexibility that might become overly frustrating in highly irregular environments like those found in space missions. Apart from this one trait all sources agree that astronauts are strongly selected on possessing one uniform optimal personality profile that makes them most likely to succeed on their mission.

A relevant observation for our line of research is that these individuals already excel at stress management, but that our suggested addition of physiologically based predictions would be a promising avenue to further increase their resilience. We have to keep in mind that no matter how high someone's natural resilience is, demanding long-term missions will create stress in even the most hardened astronauts. The following is a short overview of the main psychological stressors astronauts normally encounter.

2.2 Stress in Space

There are three stress factors that are relevant to consider when looking at the work environment of an astronaut: work, physical, and social.

Work-related stress is the stress that astronauts experience from performing their mission tasks and reaching their assigned goals. This type of stress is most comparable to regular work stress as experienced by the general populace, and ties in most strongly with the Conscientiousness dimension of the Big Five, mentioned in the previous section. It stems from organizing tasks, reaching goals and dealing with setbacks that occur during the core work activities of the astronaut.

On the level of personality, we already indicated that it is not entirely clear if high or low Conscientiousness benefits the astronaut more. However, all sources do agree that astronauts are very intelligent people that are fully capable and willing to deal with any and all work complications [12]. Additionally they strongly prefer to be able to solve their problems and manage their time in their own fashion because it affords them a sense of freedom in an otherwise extremely regulated and confined work environment [18]. As such we can conclude that work itself does not afford the astronauts much stress, and any stress it does cause, they would rather manage themselves.

A more salient stress factor is the physical environment of the astronauts. During a mission in space, they must deal with zero-gravity, extreme confinement, absence of a day-night cycle, and a very hostile surrounding atmosphere. Data on the psychological and performance repercussions of these

environmental factors is very scarce because such data has seldomly been collected during space missions. However, from experiments and data collection in analogous environments we do know that these factors endanger performance and psychological well-being [18].

The final core source of stress for astronauts is of a social kind and consists of two important components. The first is the extreme isolation from friends, family, and humanity at large. The second is dealing with the extended and intense interaction with a small group of people that is often from a different cultural/professional background. The social stress factor can be observed and tested very well in analogous environments, especially polar over-winter expeditions. Sandal et al. [18] did a meta-analysis of the research on the relevant group dynamics within such over-winter crews and distinguished four key components: Crew tension and cohesion, leadership, heterogeneity, and inter-group relationships. Each of these factors are complex constructs that interweave in to a pattern of stress relieving and reinforcing influences.

As we will be focusing on dealing with all types and levels of stress, the details of these factors fall outside of the scope of this research. The core lesson to take away is that combating social stress for astronauts is a very important factor in increasing job performance and productivity. Palinkas points out that

> "Russian cosmonaut Valentine Lebedev, who spent 211 days aboard the Mir Space Station in 1982, estimated that 30 percent of the time spent in space involved crew conflict." [14]

Clearly astronauts must still deal with immense psychological pressure despite their excellent coping techniques and resilient personality make-up. To better understand the impact this can have on an astronaut and the mission at large, we will now take a closer look at stress as a general phenomenon.

2.3 Stress in General

Hans Selye [19], arguably one of the most prominent stress experts in history, defined stress as follows:

> "Stress is the nonspecific response of the body to any demand. A stressor is an agent that produces stress at any time. The general adaptation syndrome (GAS) represents the chronologic development of the response to stressors when their action is prolonged. It consists of three phases: the alarm reaction, the stage of resistance and the stage of exhaustion."

It is in this stage of exhaustion that performance, health and general well-being go down. In 2010/2011 British governmental institution HSE [1] concluded that 400,000 out of the 1,152,000 cases of occupational illness were due to stress. In the US it is estimated that employers run into costs of almost 120 billion dollars annually as a direct or indirect result of occupational stress [22]. At the same time the individual under pressure suffers from increased chances of a variety of diseases like heart disease, stroke, arthritis, and others. Psychologically prolonged stress most often results in a burn-out characterized by depression and a general decrease in life satisfaction.

All in all, it is clear that prolonged negative stress takes its toll on the individual and society at large. Most importantly for the purpose of our research, prolonged negative stress lowers productivity, motivation and health. Stress looms very closely over the astronaut during his missions in to space (see previous section), and with the development of increasingly longer duration missions, prolonged negative stress becomes a crucial problem to face. The negative effects are such that missions and even lives can be endangered if astronauts are pushed to the limit of their ability. Yet there is more to this issue than simply minimizing all forms of stress.

"Eustress" indicates the forms of mild and/or short-lived stress that make us feel alive and challenged to excel. The flip-side of lowering all stress is that boredom and lethargy threaten there. Eustress is about the middle way we all try to walk, where we feel enough excitement and pressure to be motivated to move forward, while not being pushed so hard we collapse under the weight of it all. Selye [19] was the first to put forward this theory of positive and negative stress, phrasing it thus:

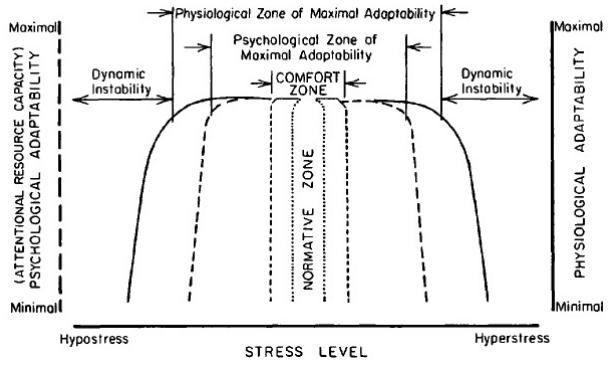
> [There is a distinction between] "eustress" and "distress" - the former being agreeable or healthy, and the latter, disagreeable or pathogenic. The way a certain stimulus will be received depends upon its intensity and the particular receptiveness of the affected person.

In the case of our astronauts, their receptiveness to stress will be particularly low, but the intensity of the pressure on them will always remain a factor, especially on future long-duration missions. In the end it is this overload we seek to help them deal with by exploring the viability of an ANN predicting an astronaut's performance based on his physiological data. The relation of stress, performance and physiological activation is more complex than laymen might expect, and will be briefly explained in the following two sections.

2.4 Stress & Performance

The concept of eustress and distress ties in directly to the performance of individuals under varying levels of stress. In 1908 Yerkes and Dodson [29] formulated a theory that is now known as the Yerkes-Dodson Law. It states that there is a certain level of stress that will result in optimum performance, but that anything above or below that will result in a decrease in performance. This law was formulated based on their experiments with rats and habitual learning, and so was never intended to grasp the deeper psychological and physiological layers of stress in humans. Hancock and Warm [8] tried to do just that with their expansion of the Yerkes-Dodson Law: The Inverted U Curve. As shown in Figure 1, it gives more detail and depth to the concept of optimal stress. Applying the concepts of eustress and distress to the Inverted U Curve, we can conclude that eustress corresponds with the stress necessary to remain near the comfort zone, while distress is the stress that sends us over the optimum and into the realm of hyperstress. The concept of hypostress is not accounted for in the eustress/distress paradigm. Also, the vertical axis of the stress model can be roughly equated with performance, as attentional resource capacity is a defining factor for this variable.

Figure 1: Inverted U-Curve



2.5 Physiological Measures of Arousal

Stress is physiologically embodied by increased activation of the autonomic nervous system (ANS). This activity is generally referred to as "arousal". Speaking in terms of "stress" and "arousal" gives the impression that the relevant physiological measures of ANS activation correspond linearly to them, but this is not the case. Prominent stress research Thayer [25] proposed that arousal is at least a bi-dimensional construct, if not even multi-dimensional. He posits that arousal is made up of an *energy* and a *tension* factor. "Energy" in this sense refers to the level of fatigue versus alertness that we feel and is related to our biological rhythm. "Tension" refers to the calmness or high alert states we experience due to our environment or internal workings. This model of arousal makes an intuitive kind of sense and was supported by Thayer's own research [25].

It also explains the lack of linearity in the arousal-performance relationship. The energy dimension would basically describe a "positive" component of arousal, that presumably increases performance, while the tension dimension seems to be more of a "negative" component of arousal that is plausibly negatively correlated with performance. Combined with the Inverted U Curve model of stress-performance, this brings us to a complex problem: Can we distinguish between alertness and tension? Might the energy and tension dimensions be basic manifestations of eustress and distress? The first question will have to

be partly answered in our research, but the second question can already receive a firm "no". We are all familiar with the sense of boredom that can arise due to inactivity or lack of stimulation, and this type of negative stress (if prolonged) is clearly a state of low tension where the level of energetic arousal does not significantly mediate the effect.

To further complicate the issue there are big differences between and within subjects when looking at the ANS activation caused by different stressors [2]. The established approach is to try and discern common patterns in the data. Such patterns are either characterized by *stimulus-response specificity* or *individual-response specificity* [23]. The last category is what will be of interest to us. This means we will be looking at what characterizes an individual's response patterns, while passing over the commonalities of responses to a certain stressor between subjects.

To measure the ANS activation we have to be familiar with its physiological components. We will limit ourselves to an overview of the most commonly measured variables when we look at ANS activation.

- **GSR (Galvanic Skin Response)** - This is the most known and straightforward measure of physiological arousal. It measures how well the skin conducts an electrical current. This conduction is influenced by how moist the surface of the skin is. Sweating increases the moisture on the skin and thus increases skin conductance. The sweat glands are solely innervated by the ANS and activate when the individual is either too hot or when he becomes more alert or excited. GSR is one of the clearest factors in determining emotion and related arousal from physiology [11] [20]. Additionally, Collet et al. [6] found that GSR and performance were so directly related that people with a low skin conductance level (SCL) were much more likely to crash a car in unforeseen circumstances than those people with a high SCL. At the same time, GSR and heart rate correlate strongly within subjects, but not between subjects, indicating very different patterns per individual [10]. Both these findings highlight the relevance of the GSR as a measure of arousal and its predictive value for performance.

- **ECG (Electrocardiogram)** - The ECG shows the electrical activity of the heart. From it we can deduce the heart rate (HR), heart rate variability (HRV) and more. Feldman et al. [7] found that these cardiovascular variables are influenced by vocalization, type of exercise (static versus dynamic) and coping style (active versus passive). On the other hand, Kreibig [11] compiled a meta-analysis that showed cardiovascular variables to be one of the most consistent indicators of strong emotion. As emotions are all embodied in part by particular arousal profiles, this goes to show the importance of the ECG measure within this research.

- **EMG (Electromyogram)** - This measure shows the activity of certain muscles. In the realm of stress and arousal, the muscles right above the eyes and on the side of cheeks are most relevant. The first is called the corrugator supercilli and is used to pull down the eyebrows in a frown. The second is called the zygomatic major and is used to pull up the corners of the mouth as when a person smiles. These muscles are generally monitored when looking at emotion, but can also be used to determine arousal [3]. Additionally, Picard et al. [15] also included the masseter muscle used to clamp the jaws shut. To keep the data set rich and not exclude any potential sources of information that are easy to monitor, we will include all these measures in our research.

- **EOG (Electrooculogram)** - This is a measure of eye-movement, startle response and blink rate. Especially the last two are considered an indication of increased arousal [28]. At the same time it is not a common measure to use because it is very sensitive to conscious control and the nature of the stressor.

- **Respiratory Measures** - There are a number of relevant respiratory measures of which respiratory rate is the most informative when looking at arousal. There is a deep interaction between the cardiovascular system and the respiratory system, partly orchestrated by the ANS. The phenomenon of respiratory sinus arythmia (RSA) is a clear example of this. When breathing is synchronized with heartbeats in a particular way, nutrients are more efficiently absorbed and pumped around the body [11]. However, respiratory activity is easily influenced by conscious action, or as an unintended side-effect of other actions that are not related to arousal levels. A clear example of this is when someone speaks, their respiratory activity is adjusted to allow for the relevant vocalization. For this reason, respiratory measures were not included in our experiment.

These five measures are the most prominent ways of collecting data on ANS activation. We will now take a closer look at how researchers have processed this data in the past and with what results.

2.6 Current Classifying Techniques

In the past researchers have used a number of classifying techniques to make sense of the physiological data they gathered. The majority of this research focuses on discerning the overall patterns of emotions along the valence-arousal spectrum. For our research, valence will not be relevant, and we instead limit ourselves to the arousal dimension, which is complex enough in itself (see previous section). At the same time there has been very little research focused only on arousal, or relating arousal or emotion to performance. As such, we have grouped the most relevant research by classifying techniques, including some more peripheral work that still overlaps in some areas with our own line of research. This scarcity of reference material was further increased by the fact that many researchers did not explain or even mention the methods they used to classify or otherwise manipulate their data. Presumably they only used standard statistical analysis techniques.

- **Artificial Neural Networks** - The US Air Force had research done into how physiological data could be used to control optimum work load. Wilson & Russell [28] used neural networks to give real-time assessments of mental workload. They used EEG, EOG, ECG and respiratory measures to give workload assessments that were over 80% accurate on average.
- **Probability-Based Classifiers** - A number of these have been used in determining emotions from physiological data, most notably the Fisher Discriminant Analysis [5] and Fisher Projection [15], but also Naive Bayes Classifiers [5]. It was found that only the Fisher Projection offered emotion recognition performance significantly above chance level, and only when combined with feature selection algorithms like Sequential Floating Forward Search. Wagner et al. [26] used the same technique to classify 4 emotions based on their arousal-valence levels. He found that classifying arousal (95%) was easier than valence (87%), but only distinguished between two levels of arousal (high/low). No link was made with performance.
- **Clustering** - Allen et al. [2] used ECG and GSR measures of participants under various forms of stress, and found that a Ward's method clustering algorithm created meaningful clusters of participant responses for each stressor but not between stressors. In other words, clusters did not correspond to each other in characteristics or membership between stressors. Someone who responded with a high heart rate and RSA under one stressor, might show a completely different pattern for the next stressor. The stressors in question were purposefully of a different nature: reaction time test, mental arithmetic, and a cold pressor task. No link was made with performance.

The greatest difference between these methods used in the past and our intent is that we will look at the viability of creating reasonably accurate predictions of performance based on the user's physiological data. Only the classification work done for the US Air Force [28] really comes close in aim and setup. Table 1 gives a more complete overview of the machine learning techniques that were considered for this research. Pros and cons are only reviewed for relevant aspects (ie. that a method is strong/weak with huge data sets is not relevant for our current research).

2.7 Artificial Neural Networks

The following is a short review of the basic principles of an ANN. For a full in-depth introduction to this topic, we refer the reader to Basheer & Hajmeer's paper published in the Journal of Microbiological Methods [4].

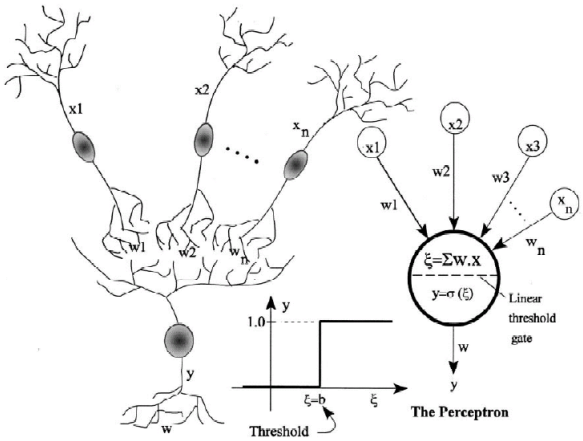
First off, we not only picked ANN as our method of prediction generation because of Wilson & Russell's [28] prior research, but also because of a number of benefits inherent to the method: nonlinearity, robustness, fault and failure tolerance, learning, ability to handle imprecise and fuzzy information, and capability to generalize [9]. As mentioned before, the physiological data we will acquire is notoriously noisy and imprecise, while on the other hand the relation between arousal and performance are different per individual and nonlinear in nature. Therefore, ANN is the most plausible choice for our research.

Table 1: Overview of Machine Learning Techniques Considered for this Research

Technique	**Main Advantage**	**Main Disadvantage**
Decision Tree Learning	Creates a model of the data that is easy to understand	Finding a model is NP-complete problem
Support Vector Machines	Creates classification of data that is retraceable by researchers	Results are categorical and not continuous
Inductive Logic Programming	Closely approximates human (rational) reasoning	Does not handle unknown relations well
Bayesian Learning	Everything that is learned can be understood by the researchers	Relations between variables must be known
Clustering	Used a lot in the strongly-related field of emotion research	Categorical, so not a good choice for learning to predict continuous performance scores
Reinforcement Learning	Simple and effective	Patterns must be known in advance
Genetic Programming	No domain knowledge necessary	Cannot run simulation to test fitness
Artificial Neural Networks	No domain knowledge necessary & can deal with continuous data	Hard to extract what has actually been learned (black box)

The qualities, structure and mechanics of the ANN are broadly based on natural neuronal systems. Figure 2 shows the parallels between biological and artificial neurons. The basic idea is that every neuron gets input and output from a number of other neurons. The signal each neuron passes on is either on or off (0 or 1). These signals are denoted by x_1 to x_n. Each neuron receives n such signals and each signal is weighted by a factor w_1 to w_n that can be any value between -1 and 1. Negative weights create so-called *inhibitory* signals, while positive weights create *excitory* signals. All incoming signals are summed and compared to a threshold. This sum is denoted $\xi = \sum w.x$. If this sum is greater than the threshold value b, then the output signal y will be 1, otherwise it will be 0.

Figure 2: Overview of a Perceptron



This is the basic principle of ANNs. The complexity of this system comes to bear in how the neurons are structured in relation to each other and what the form of the learning algorithm should be. We have used a Backward Propagation ANN (BPANN) that runs in two phases. First it will generate an output based on the input. Then it will sweep backward to adjust the weights of the neurons based on the error size of the prediction (supervised learning). The exact structure of the network and the learning algorithms used will be discussed in the methods section, as this is highly dependent on the shape of the data we collect.

3 Method

Participants were all repeatedly tested on their reaction time (RT) on an adjusted version of the Stroop Test. Each test took 1 minute, and was proceeded by 1 minute of white noise (randomly fluctuating sound that is equally distributed over a certain bandwidth). During this rest period, their ECG, GSR and masseter and corrugator activity were measured. This combined into 40 data points consisting of 1 minute of rest followed by 1 minute of RT task. The physiological data in the rest period is used as the input of the ANN to predict the reaction time of the participant in the following minute of the reaction time task. The following is a more in-depth description of how the experiment was executed.

3.1 Task

A custom reaction time task was created to test the reaction times of the participants. The challenge was to create a task that was short and easily repeatable so that sufficient data points could be gathered in a reasonable time, while the task had to be complex enough to show some variability in performance based on how stressed or aroused the participant was. A repeatable 1-minute reaction time task would be ideal, but no such thing was on offer. Therefore, we created a task ourselves that merged the strengths of the PVT [27] and the Stroop Test [24]. The PVT is the validated reaction time test used by NASA that was previously mentioned. The Stroop test is a cognitive dissonance task popular in psychology. Normally it involves a list of words that all indicate a specific color, but are also printed in different colored fonts that do not match their meaning. For instance, the word "GREEN" will be printed in a red font.

To combine the strengths of these two approaches, we created a custom stand-alone reaction time experiment (Figure 3) that worked as follows: a black window was displayed as well as a counter for recent and average reaction time and mistakes. Once a session started, a word was shown at a random position on the screen. The meaning of the word was a color (ie, GREEN), but the font of the word also had a distinctive color. If the meaning and the font color were the same, the user was expected to press "y". If they were different they were expected to press "n". The task was to press the correct button as quickly as possible, while keeping mistakes per session at zero. A session took 1 minute, in which around 21 words were shown with a random between-word-interval of 2-5 seconds. Each 1-minute session would create one reaction time data point: the average reaction time for that session. We also tracked the number of mistakes per session. This application was created in Java using the Eclipse IDE. The code can be viewed in Appendix C.

In between sessions, the participant was subjected to a continuous blast of white noise to increase mental fatigue and/or irritation. Sessions were repeated 40 times with 1-minute intervals of white noise between each session.

3.2 Design

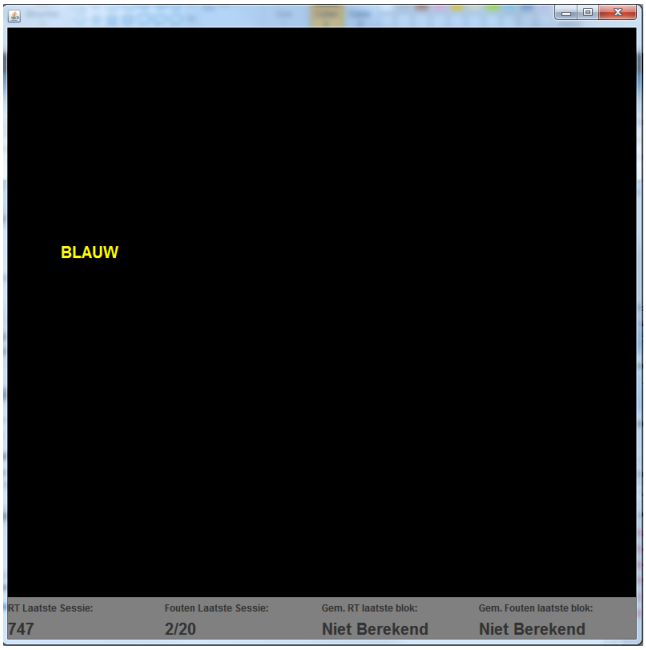
The experiment is a within-subject design where each participant is first trained on the reaction time task till they reach a stable plateau of performance. The next day they are extensively tested on their reaction time under mild stress (white noise, fatigue and boredom). The neural networks are trained on a person-by-person basis and are thus custom per individual.

3.3 Materials

The following materials were used for the experiment:

- The custom reaction time task as described in the Task section above.
- A sound-proof room encased in a Faraday mesh to ward off electro-magnetic disturbances in the sensor data
- The "G-Tec USBamp: USB BioSignal Amplifier", "G-Zcheck: Impedance Check", Tenzo Conductive EEG paste, and 8 basic cup electrodes were used for the physiological measurements. The Matlab Simulink module was used to receive and store the signals.

Figure 3: Screenshot of Reaction Time Task



- A 52 inch HD television was used to present the first 10 minutes of the series Planet Earth, a popular nature documentary.
- The data was preprocessed using Matlab.
- The ANN was created using the Encog library for Java.

5 participants, 3 male and 2 female, took part in this experiment. Each participant was screened for relevant health issues and color blindness.

3.4 Measures

The experiment yielded 5 data sets consisting of the 4 physiological channel measurements (ECG, GSR, masseter, corrugator), as well as all the temporal data of the reaction time task such as the reaction times, mistakes, and what color meaning-font combination was shown. All physiological variables were determined for the intervals **before** a reaction time session. From the ECG channel the heart rate, AVNN, SDNN and RMSSD were determined. The GSR was taken as a variable as-is. For both the masseter and the corrugator the muscle activity was determined. For the ECG and GSR related variables the average per rest session was calculated. For the masseter and corrugator the sum of the activity over the rest session was determined. In this manner we generated one value per variable per session to generalize and simplify the data processing for the ANN. The average reaction time per reaction time task session was calculated to serve as target output for the training and testing of the ANN.

Each participant also completed a small questionnaire at the end of the experiment, asking them basic questions about the general experience they had with the experiment.

3.5 Procedure

Before we tested the participants for the final test setup, we first executed a pilot run with one participant to test all the elements of the experimental design. The data from the pilot experiment is not included in any of the results. After the data collection of the experiment, a neural network was designed to process all the data and yield an answer to our research question.

Figure 4: Masseter & Corrugator Sensors on a Participant

3.5.1 Pilot Experiment

During this pilot study one participant was used under fully informed consent, who gave continuous feedback on the procedure. From this pilot study it became clear that the measurements of the zygomatic muscle and the masseter muscle were nearly identical, with the electrodes for the zygomatic being much harder to place. Therefore, the zygomatic measurements were scrapped from the program. This is also apparent in the Measures section above. Secondly, we tried out different measurement techniques and figured out the details of an effective procedure, including efficient skin preparation, optimal electrode configuration (bipolar measurement with ground), and correct filtering (notch filter to exclude noise). It also became clear during the experiment that applying the electrodes was an involved and slow process, and therefore it was better to restrict the experiment to one long session, instead of 2 shorter ones. That way the electrodes only needed to be placed once. Additionally, the white noise stressor was not found overly stressful by the pilot participant, but the length of the experiment was. The original idea was to split the experiment into two sessions of 20 cycles, but to increase the mental fatigue related stress, we changed that. Thus we increased the length of the experiment by making it one long session instead of two short ones, fully capitalized on this fatigue effect so the participants would experience some form of stress to influence their arousal levels, leading to a more diversified data set.

3.5.2 Core Experiment

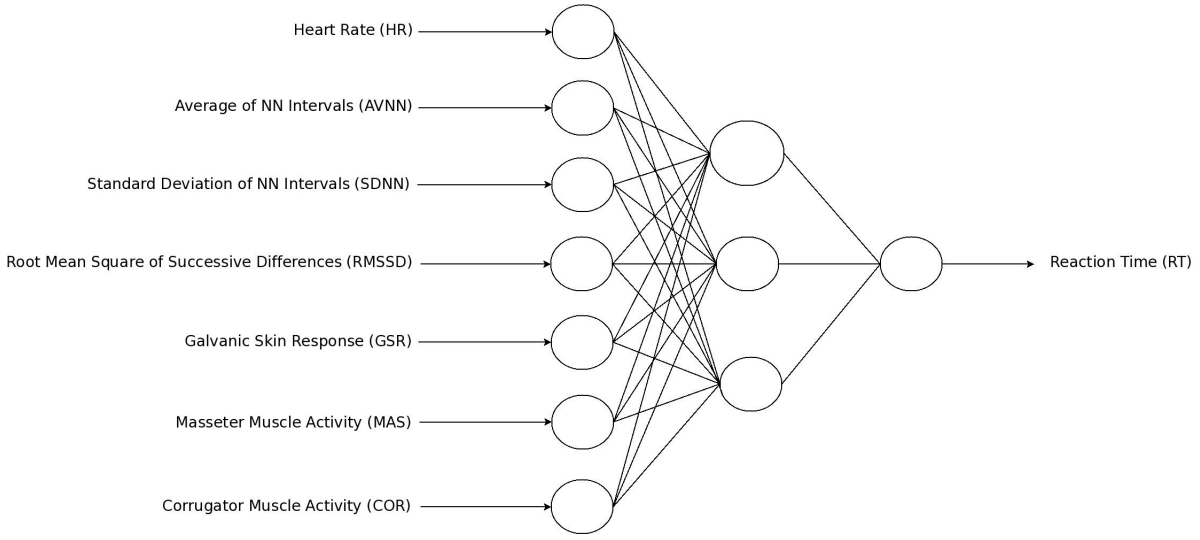
The actual experiment was split into two phases: the training phase and the test phase. In the training phase the participant was introduced to the test environment and the task. The experiment took place in a sound-proof room encased in a Faraday-mesh to minimize ambient noise and electrical signals. The participant was seated behind a computer running the reaction time task described previously. He was presented with a consent form that he was asked to sign, as well as instruction sheet for the training phase. For this training part of the experiment, the participants could initiate the next session of the task described above themselves with a button press ("F1"). The intention was that the participants would become familiar enough with the task to cancel out any type of training effect during the test phase, so that fluctuations in reaction time could be reasonably attributed to physiological states of arousal. The participant would continue training until his reaction time did not improve anymore, and his average mistakes per five sessions was below 1. This took between 20 and 50 minutes for the various participants.

Each participant came back the next day for the test phase of the experiment. The one day interval was chosen to minimize the time strain of the experiment per day, while being so close together that the training effect would still be minimized as much as possible. After receiving instructions on the details of the procedure for that day, a number of electrodes were placed on the participant's body. Three electrodes were placed on the torso according to the 3-channel ECG measurement scheme. Two electrodes were placed on the corrugator muscle (eyebrow), and two were placed on the masseter muscle (jaw), so that bipolar measurements of the relevant muscle activity could be made. A ground electrode for this facial EMG was placed on the forehead of the participant. Lastly, the GSR sensor was placed around the ring finger and the little finger of the right hand of the participant, leaving the left hand free for the reaction time task. The experiment started with a 10 minute session of watching a calming clip of a nature documentary. This was intended to allow the participant to reach a baseline rest state and to become used to the feeling of the electrodes on his/her body. Once this clip was over, the reaction time task was initiated by the researcher. The task was identical to the one used in the training phase except that the participant could not start the next session themselves. Instead, after each 1-minute session, they were exposed to one minute of white noise through a pair of headphones. This process was repeated 40 times, where each pair of 1-minute rest with white noise, and 1 minute reaction time task, was one measurement point in the data set. Only the very first reaction time task session was not proceeded by the white noise as the documentary viewing immediately rolled over in to the reaction time task. Throughout the experiment, the researcher kept an eye on the measurements and the participant. The participant had been instructed to sit as still as possible throughout the experiment to minimize the contamination of the physiological measurements by body movements.

3.5.3 Neural Network

There were 7 input variables defined for the ANN: heart rate (HR), AVNN, SDNN, RMSSD, GSR, masseter activity (MASS), corrugator activity (COR). One output variable was defined: reaction time (RT). All variables were normalized to the interval [0,1]. The Encog Java library was used to create, train and test the neural network. We employed the basic Resilient Propagation training method with a 0.01 error rate. As only positive number output was expected, a sigmoid activation function was used in the neural network. For each participant the training was executed 50 times to a limit of 10,000 epochs. This was done for both the complete data set per participant, and the data set with obvious outliers excluded. An overview of our ANN can be seen in Figure 5. We refer the reader to Basheer & Hajmeer's paper published in the Journal of Microbiological Methods [4] for more in-depth information about the the ANN procedures we are using. Figure 5 shows an overview of the ANN structure for this experiment.

Figure 5



Ideally, an ANN is trained on a large subset of the total data set, and tested on a reasonable remainder set of about 20%. However, due to our small sample size, we wanted to maximize the size of the training

set, without skewing our results too much by only have a tiny test set left. To this end, we used a k-1 cross validation technique to test our hypothesis. Our implementation of this was to take 39 of the 40 data points as a training set, and then to use the last point as a test set. Then we repeated this procedure taking a different 39 points as training set, and then predicting another single data point. We performed this procedure for every possible permutation, meaning that we tried to predict every single point in the data set while using the rest of the data set as training material. For each try to predict a particular point, we used the iterations mentioned above (50 iterations with each ANN being trained to a max of 10,000 epochs). The result is that we created 40x50 ANNs to a grand total of 2000 per participant. The procedure was suitably adjusted if outliers were taken out of the data set, and then run again for another 2000 ANNs.

3.6 Statistical Analysis

Before the data for each participant was used to train an ANN, it was first subjected to some basic statisical analyses. First, the correlations between all 8 variables were determined so it would be clear what kind of patterns might already be discerned in the data. It is informative to see if an ANN can obtain similar or better prediction accuracy for participants with low input-to-output variable correlations in comparison to participants with high input-to-output variable correlations. Additionally, it can inform the Discussion section as it can help find anomalies in the data.

In the statistical analysis in the Raw Data section the correlation coefficients among all variables were calculated. The r-value is the size of the correlation between two variables. The p-value is the chance that such a correlation would exist randomly when the true correlation is zero. P-values equal to or smaller than 0.05 (α-value) are considered significant and printed in **bold font**.

4 Results

First we will look at the raw data of each of the five participants, both with and without any possible outliers. Next, we will investigate the results of applying the ANN to the data to see how well it can predict reaction times.

4.1 Raw Data

For each participant we will present noticeable details of the measurement procedure and the participant's experience (questionnaire results, see Appendix A), followed by the p and r values of the correlations between all the calculated variables outlined in the Methods section. First all the data is presented in its entirety, and then with the most obvious outliers explained and removed. See Appendix B for the Matlab code used for preprocessing the data.

4.1.1 Participant 1

During the experiment with the first participant, the reaction time task was accidentally set to 20 cycles instead of 40. For this reason, after 20 cycles, the researcher had to interrupt the participant and reset the task for another 20 cycles. The end result is a full set of 40 data points without any noticeable artifacts due to the interruption.

As can be seen in Table 2, this participant had a lot of correlations among the physiological variables. Some of these make sense by nature, like the various ECG-related measures, while others show some interesting patterns. Most notably, increased activity in the muscles of the face (MASS & COR) had a strong negative correlation with reaction time. This means the participant frowned and clenched jaws when reacting faster.

Furthermore, this participant made a total of 8 mistakes in the reaction time task, spread out over 7 sessions. (S)He judged the experiment to be extremely tiring, and hardly considered it to become easier with time. The movie was rated extremely relaxing, and the sensors were not found unpleasant at all.

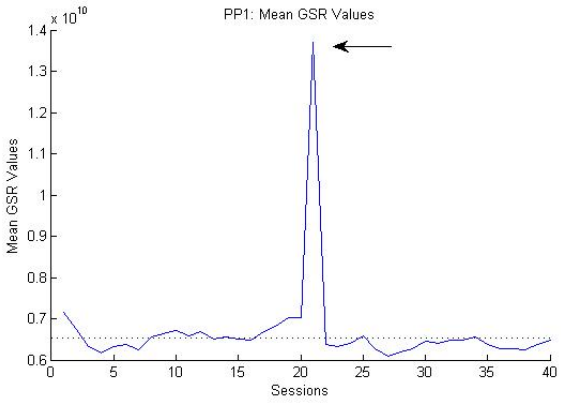
Table 2: Participant 1 — p & r values for complete data set

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	0.2862	**0.0000**	**0.0000**	**0.0000**	**0.0001**	**0.0008**	0.7405
	r	-	-0.1728	**0.6146**	**0.9870**	**0.9585**	**0.5950**	**0.5100**	-0.0540
AVNN	p	*	-	0.4914	0.9096	0.9801	**0.0408**	**0.0201**	**0.0159**
	r	*	-	0.1120	-0.0185	-0.0041	**-0.3248**	**-0.3663**	**0.3789**
SDNN	p	*	*	-	**0.0000**	**0.0000**	**0.0493**	0.2039	0.8897
	r	*	*	-	**0.6563**	**0.6172**	**0.3129**	0.2052	-0.0227
RMSSD	p	*	*	*	-	**0.0000**	**0.0002**	**0.0025**	0.9944
	r	*	*	*	-	**0.9724**	**0.5577**	**0.4645**	0.0012
GSR	p	*	*	*	*	-	**0.0011**	**0.0094**	0.5195
	r	*	*	*	*	-	**0.4976**	**0.4058**	0.1049
MASS	p	*	*	*	*	*	-	**0.0000**	**0.0011**
	r	*	*	*	*	*	-	**0.9745**	**-0.4981**
COR	p	*	*	*	*	*	*	-	**0.0003**
	r	*	*	*	*	*	*	-	**-0.5407**
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

There was one clear outlier in the GSR data, as visible in Figure 6. After removal of the entire session related to that data point (21), the correlations were as seen in Table 3. When we compare the two tables before and after the removal of the outlier, it becomes clear that no correlations were added or lost with the outlier removal. Some correlations became stronger, such as those related to reaction time, while others became weaker such as those among the ECG measures.

Figure 6



4.1.2 Participant 2

Training for participant 2 had to be repeatead due to a scheduling conflict, where the participant was not available for testing the day after the initial training. As training only served to familiarize the participant with the task and avoid a training effect during testing, this repeated training did not impact the results in any meaningful way.

As can be seen from the correlations in Table 4, there were no physiological variables in the full data set that significantly correlated with reaction time (RT). All physiological variables apart from the GSR

Table 3: Participant 1 — p values for data set without outliers (point 21)

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	0.0000	**0.6898**	**0.0011**	**0.5624**	**0.0404**	**0.0289**	0.0079
	r	-	-0.8248	**-0.0660**	**0.5024**	**0.9585**	**0.3296**	**0.3501**	-0.4193
AVNN	p	*	-	0.3251	0.9202	0.6357	**0.0181**	**0.0112**	**0.0172**
	r	*	-	0.1618	0.0166	0.0783	**-0.3766**	**-0.4021**	**0.3796**
SDNN	p	*	*	-	**0.0613**	**0.8618**	**0.7154**	0.4408	0.7666
	r	*	*	-	**0.3025**	**-0.0288**	**-0.0603**	-0.1271	-0.0491
RMSSD	p	*	*	*	-	**0.7570**	**0.6280**	**0.6684**	0.1918
	r	*	*	*	-	**-0.0512**	**0.0801**	**0.0708**	-0.2136
GSR	p	*	*	*	*	-	**0.1056**	**0.1266**	0.0116
	r	*	*	*	*	-	**-0.2631**	**-0.2488**	0.4000
MASS	p	*	*	*	*	*	-	**0.0000**	**0.0000**
	r	*	*	*	*	*	-	**0.9734**	**-0.6143**
COR	p	*	*	*	*	*	*	-	**0.0000**
	r	*	*	*	*	*	*	-	**-0.6217**
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

correlated at least somewhat with each other. The lack of correlations with the GSR measure, which is generally considered a good predictor of arousal, is quite notable.

From the questionnaire results it became clear that this participant was strongly displeased with the experiment. He did finish the experiment while he was aware that he could quit at any time. He found the experiment extremely tiring, was bothered quite a bit by the sensors, would not want to (hypothetically) repeat the experiment at all, and did not find the task to become easier with time.

Table 4: Participant 2 — p & r values for complete data set

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	**0.0000**	**0.0001**	**0.0006**	0.2086	**0.0000**	**0.0000**	0.4841
	r	-	**-0.9640**	**0.5922**	**0.5200**	-0.2032	**0.6805**	**0.6407**	-0.1139
AVNN	p	*	-	**0.0001**	0.0580	0.2601	**0.0000**	**0.0000**	0.4827
	r	*	-	**-0.5947**	-0.3023	0.1823	**-0.6466**	**-0.6286**	0.1142
SDNN	p	*	*	-	**0.0183**	0.1159	**0.0304**	0.0748	0.4327
	r	*	*	-	**0.3715**	-0.2525	**0.3428**	0.2849	-0.1276
RMSSD	p	*	*	*	-	0.0591	**0.0335**	0.0976	0.9979
	r	*	*	*	-	-0.3010	**0.3370**	0.2656	0.0004
GSR	p	*	*	*	*	-	0.2005	0.2812	0.5159
	r	*	*	*	*	-	-0.2068	-0.1746	0.1058
MASS	p	*	*	*	*	*	-	**0.0000**	0.1279
	r	*	*	*	*	*	-	**0.9111**	-0.2448
COR	p	*	*	*	*	*	*	-	0.2425
	r	*	*	*	*	*	*	-	-0.1891
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

After analyzing the different input variables and how they are distributed, one outlier was found for GSR, as can be seen in Figure 7. The line indicates the average of the GSR values, and this outlier was the point farthest removed from there. The other low dip after that was not removed as there is an equal

positive amplitude spike in the data. Removing session 16 created the following correlational data (Table 5). Removal of the outlier did not create any more correlations between any of the input variables and the GSR. It also did not create any significant correlations between RT and any of the input variables. It did however improve correlational significance and strength of most variables all across the board by a small margin.

Figure 7

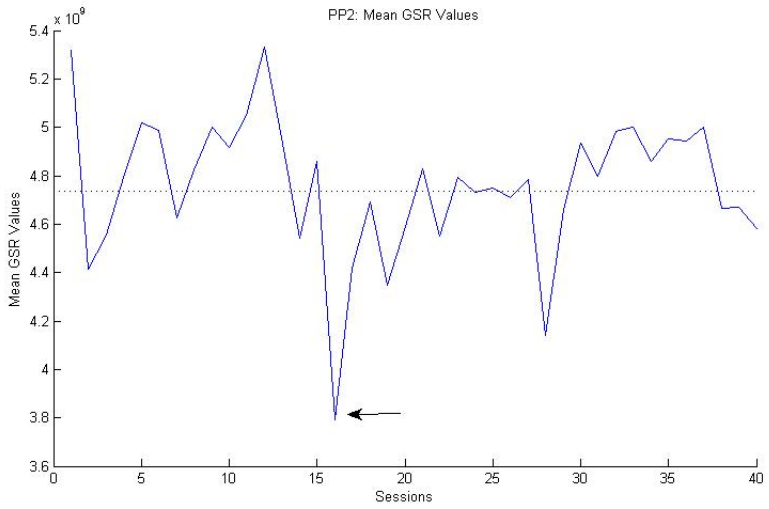


Table 5: Participant 2 — p & r values for data set without outliers (point 16)

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	**0.0000**	**0.0001**	**0.0004**	0.1658	**0.0000**	**0.0000**	0.4785
	r	-	**-0.9653**	**0.5919**	**0.5417**	-0.2264	**0.6896**	**0.6467**	-0.1169
AVNN	p	*	-	**0.0001**	**0.0408**	0.1499	**0.0000**	**0.0000**	0.4965
	r	*	-	**-0.5970**	**-0.3291**	0.2350	**-0.6488**	**-0.6294**	0.1122
SDNN	p	*	*	-	**0.0182**	0.0974	**0.0277**	0.0715	0.4186
	r	*	*	-	**0.3764**	-0.2693	**0.3526**	0.2917	-0.1333
RMSSD	p	*	*	*	-	0.3448	**0.0107**	**0.0490**	0.8389
	r	*	*	*	-	-0.1554	**0.4043**	**0.3174**	-0.0336
GSR	p	*	*	*	*	-	**0.0382**	0.0898	0.2439
	r	*	*	*	*	-	**-0.3331**	-0.2753	0.1911
MASS	p	*	*	*	*	*	-	**0.0000**	0.1494
	r	*	*	*	*	*	-	**0.9103**	-0.2352
COR	p	*	*	*	*	*	*	-	0.2701
	r	*	*	*	*	*	*	-	-0.1810
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

4.1.3 Participant 3

The experiment proceeded without any notable irregularities for this participant. His results show a reasonable amount of correlations among the physiological variables themselves, with again the notable exception of the GSR (see Table 6). However, corrugator activity (COR) did correlate significantly with reaction time. This correlation is a weak/medium correlation of 0.3581. That means that longer reaction times for this participant were generally proceeded by a mild increase in frowning movement.

This participant found the experiment very tiring, was not really bothered by the sensors, thoroughly enjoyed the relaxing movie at the start, would not hesitate to repeat the experiment if asked, and found the experience to become reasonably easier with time. He added a note himself about how he found his mind wandering off at times.

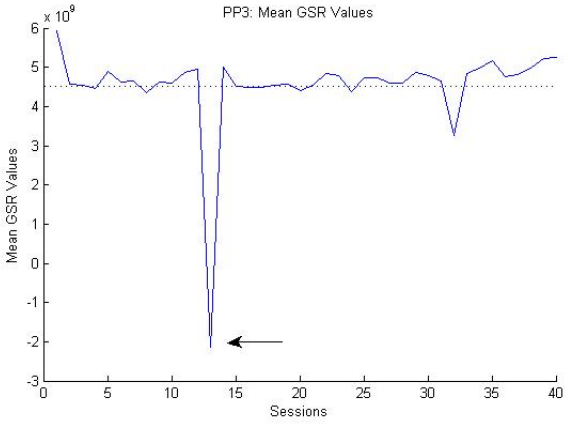
Table 6: Participant 3 — p & r values for complete data set

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	**0.0000**	**0.0118**	**0.0223**	0.1990	**0.0000**	**0.0001**	0.7907
	r	-	**-0.9163**	**0.3942**	**0.3606**	-0.2075	**0.7533**	**0.5827**	-0.0433
AVNN	p	*	-	**0.0002**	0.8405	0.1448	**0.0000**	**0.0000**	0.8420
	r	*	-	**-0.5549**	0.0328	0.2347	**-0.7698**	**-0.5988**	-0.0325
SDNN	p	*	*	-	**0.0302**	0.1928	**0.0086**	0.2323	0.9135
	r	*	*	-	**-0.3431**	-0.2103	**0.4099**	0.1932	0.0177
RMSSD	p	*	*	*	-	0.7746	0.6348	0.6155	0.4110
	r	*	*	*	-	0.0467	0.0774	0.0819	-0.1336
GSR	p	*	*	*	*	-	0.1976	0.1375	0.5425
	r	*	*	*	*	-	-0.2081	-0.2390	-0.0992
MASS	p	*	*	*	*	*	-	**0.0000**	0.5738
	r	*	*	*	*	*	-	**0.8058**	0.0916
COR	p	*	*	*	*	*	*	-	**0.0233**
	r	*	*	*	*	*	*	-	**0.3581**
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

As can be seen in Figure 8, data point 13 deviates quite far from the mean in comparison to the other points. For that reason this point was labelled an outlier for the second iteration of the experiment. The second dip after that is not considered an outlier because the amplitude is of a similar size as that of the very first point. The correlational data for the data set without outlier can be seen in Table 7. The removal of the outlier gained us one significant correlation between corrugator activity and the GSR measure. Apart from that it created barely any noticeable differences in correlational significance or strength.

Figure 8



4.1.4 Participant 4

There were no irregularities in the experiment with this participant. If we look at Table 8 we can see the variables deduced from this participant's data did not contain many correlations. Notably the ECG-

Table 7: Participant 3 — p & r values for data set without outliers (point 13)

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	**0.0000**	**0.0163**	**0.0212**	0.1212	**0.0000**	**0.0001**	0.7269
	r	-	**-0.9147**	**0.3824**	**0.3679**	-0.2524	**0.7505**	**0.5727**	-0.0577
AVNN	p	*	-	**0.0003**	0.8624	0.0883	**0.0000**	**0.0001**	0.9177
	r	*	-	**-0.5449**	0.0287	0.2766	**-0.7675**	**-0.5868**	-0.0171
SDNN	p	*	*	-	**0.0328**	0.1590	**0.0112**	0.3018	0.9852
	r	*	*	-	**-0.3426**	-0.2300	**0.4020**	0.1697	0.0031
RMSSD	p	*	*	*	-	0.7251	0.6252	0.5877	0.4253
	r	*	*	*	-	0.0582	0.0807	0.0896	-0.1314
GSR	p	*	*	*	*	-	**0.0260**	0.3365	0.9622
	r	*	*	*	*	-	**-0.3563**	-0.1581	-0.0078
MASS	p	*	*	*	*	*	-	**0.0000**	0.6176
	r	*	*	*	*	*	-	**0.8063**	0.0825
COR	p	*	*	*	*	*	*	-	**0.0307**
	r	*	*	*	*	*	*	-	**0.3465**
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

derived variables contained only two correlations. On the other hand, the two ECG-derived variables did correlate with the GSR. Such a correlation between heart activity and GSR was missing in participants 2 and 3 previously. Facial activity correlated significantly with SDNN, but also notably with itself. That is to say, the measurements of the masseter and corrugator muscles were almost identical (p = 0.0000, r = 0.9999). This could either be a measurement error or a peculiarity of the facial expressions and muscle structure of this participant. There were no significant correlations between reaction time and the physiological variables.

This participant found the experiment extremely tiring, was a little bothered by the sensors, appreciated the relaxation movie, would surely participate again in similar research, while he did not find the experiment to become easier with time in the slightest. He added the comment that he found the whole procedure sleep-inducing.

As can be seen in Figure 9, this participant had one major outlier in his masseter readings. These readings were practically identical to his corrugator data, but for succinctness, only the masseter graph is shown here. This one was removed for the outlier-free data set. Unlike previous participants, he did not have a clear outlier in his GSR data. After removing the MAS/COR outlier, the correlations among his derived variables were as shown in Table 9. The impact of the outlier removal is not very great, as we have exchanged one correlation for another, while neither relate to reaction time.

Figure 9

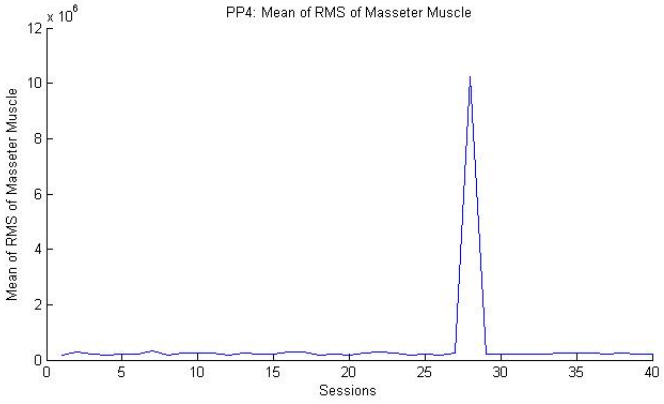


Table 8: Participant 4 — p & r values for complete data set

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	**0.0000**	0.6551	**0.0006**	**0.0340**	0.6037	0.6158	0.9090
	r	-	**-0.9670**	0.0729	**0.5167**	**0.3361**	0.0846	0.0818	0.0187
AVNN	p	*	-	0.2689	0.0657	**0.0236**	0.2623	0.2699	0.8116
	r	*	-	-0.1791	-0.2939	**-0.3573**	-0.1815	-0.1787	0.0389
SDNN	p	*	*	-	0.0691	0.2458	**0.0000**	**0.0000**	0.8711
	r	*	*	-	-0.2904	-0.1878	**0.7229**	**0.7184**	-0.0265
RMSSD	p	*	*	*	-	0.6745	0.3597	0.3589	0.2223
	r	*	*	*	-	0.0685	-0.1487	-0.1490	0.1973
GSR	p	*	*	*	*	-	0.8511	0.8695	0.9270
	r	*	*	*	*	-	-0.0306	-0.0268	-0.0150
MASS	p	*	*	*	*	*	-	**0.0000**	0.6248
	r	*	*	*	*	*	-	**0.9999**	-0.0797
COR	p	*	*	*	*	*	*	-	0.6303
	r	*	*	*	*	*	*	-	-0.0785
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

Table 9: Participant 4 — p & r values for data set without outliers (point 28)

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	**0.0000**	0.8944	**0.0004**	**0.0346**	0.2910	0.3247	0.8797
	r	-	**-0.9713**	0.0220	**0.5367**	**0.3392**	0.1734	0.1619	0.0250
AVNN	p	*	-	0.6483	**0.0404**	**0.0213**	0.3306	0.4561	0.8769
	r	*	-	-0.0754	**-0.3297**	**-0.3676**	-0.1600	-0.1229	0.0256
SDNN	p	*	*	-	0.1039	0.1340	**0.0353**	0.2486	0.8002
	r	*	*	-	-0.2644	-0.2443	**0.3380**	0.1892	0.0419
RMSSD	p	*	*	*	-	0.6918	0.5879	0.1795	0.2509
	r	*	*	*	-	0.0655	0.0895	0.2194	0.1883
GSR	p	*	*	*	*	-	0.1423	0.1509	0.9185
	r	*	*	*	*	-	-0.2393	-0.2344	-0.0169
MASS	p	*	*	*	*	*	-	**0.0000**	0.5855
	r	*	*	*	*	*	-	**0.9183**	-0.0901
COR	p	*	*	*	*	*	*	-	0.5291
	r	*	*	*	*	*	*	-	-0.1039
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

4.1.5 Participant 5

This participant did not sit still during the experiment. All participants were instructed to remain seated in as calm a manner as possible throughout the experiment. No procedure was devised to enforce this instruction, as it was initially considered to be simple to follow up on. However, this participant was often bopping her legs up and down, and shifting in her seat. Additionally, this participant made 46 mistakes spread out over 29 of the 40 sessions. She found the experiment very tiring, was a bit bothered by the sensors, found the movie relaxing, would not want to repeat the experiment and did not find the experiment to become easier with time.

Looking at the correlations between physiological variables and reaction time, it is noticable that all

her ECG-derived variables correlate very strongly with each other, and that her masseter and corrugator readings were basically identical. Interestingly, her GSR ratings did correlate significantly with those masseter and corrugator readings, showing a negative trend. That means that increased activity in the muscles of the face were associated with lower GSR values. There were no significant correlations between her reaction time (RT) and any of the physiological variables.

When searching for outliers, it became clear that the data for this participant was strongly contaminated, quite possibly by her restless movements. The contamination is shown in the highly erratic and unlikely values for the the physiological variables. This is most clearly illustrated by her heart rate values (HR), as shown in Figure 10. The arrow points to the most impossible value: a heart rate of 15 beats per minute. However, the ratings overall are rather unlikely in general, with extremely low values being prevalent, while the 15 beats/minute is followed by a 70 beat/minute interval.

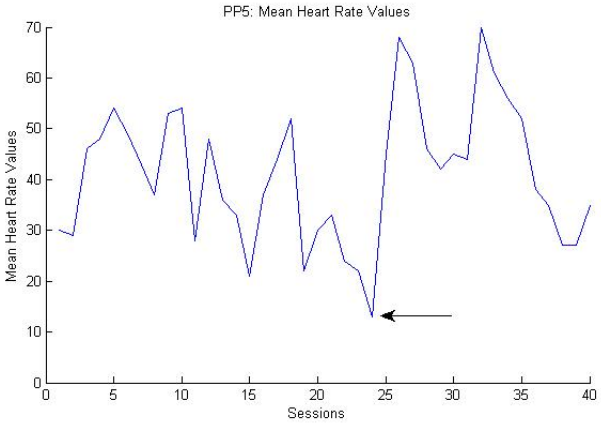
For this reason, there was no way to determine relevant outliers in the data set. For completeness sake, an ANN was trained on the full data set to see what results could be gleaned.

Table 10: Participant 5 — p & r values for complete data set

		HR	AVNN	SDNN	RMSSD	GSR	MASS	COR	RT
HR	p	-	**0.0000**	**0.0000**	**0.0102**	0.5489	0.9770	0.9770	0.1475
	r	-	**-0.8660**	**-0.7594**	**0.4016**	-0.0977	-0.0047	-0.0047	-0.2332
AVNN	p	*	-	**0.0000**	**0.0019**	0.2153	0.9821	0.9840	0.2881
	r	*	-	**0.9606**	**-0.4765**	0.2003	-0.0037	-0.0033	0.1721
SDNN	p	*	*	-	**0.0050**	0.1630	0.9223	0.9247	0.2104
	r	*	*	-	**-0.4356**	0.2249	-0.0159	-0.0154	0.2024
RMSSD	p	*	*	*	-	0.8555	0.6824	0.6817	0.1440
	r	*	*	*	-	-0.0297	-0.0668	-0.0669	0.2352
GSR	p	*	*	*	*	-	**0.0242**	**0.0241**	0.1364
	r	*	*	*	*	-	**-0.3559**	**-0.3561**	0.2396
MASS	p	*	*	*	*	*	-	**0.0000**	0.4045
	r	*	*	*	*	*	-	**1.0000**	-0.1355
COR	p	*	*	*	*	*	*	-	0.4044
	r	*	*	*	*	*	*	-	-0.1355
RT	p	*	*	*	*	*	*	*	-
	r	*	*	*	*	*	*	*	-

Legend: * = duplicate value; - = self-correlation; **bold text** = significant value at $\alpha = 0.05$

Figure 10



4.2 Neural Network

For each participant there will be 4 graphs presented: one graph with the mean prediction error per session and one graph with the minimum prediction error per session, both for the data set with and without the outliers determined in the participant raw data review above. The exception to this format is participant 5. Her data was too contaminated to determine any outliers in a meaningful manner, as stated in the previous paragraph. The means and minimums are of each of the 50 repetitions of training a network on all data points except the relevant one, and then letting it predict the relevant data point. In the graphs that are presented, the dotted lines for mean errors indicate the 0.1 and 0.4 marks, while the minimum error graphs have a dotted line for the 0.2 mark. These are consistent references across the graphs of all the participants and indicate the general range of the values.

4.2.1 Participant 1

As can be seen from the graphs in Figures 11 and 12, the removal of the outlier did not create any clear improvement of the accuracy of the predictions. Also, it is clear from these graphs that while some values can be extremely accurately predicted, others can have a sizeable error, up to the 0.3 mark for the best networks (minimum error). Such a 0.3 deviation in prediction comes down to about a 60 millisecond deviation in predicted reaction time over a maximum range difference of around 195 milliseconds.

Figure 11

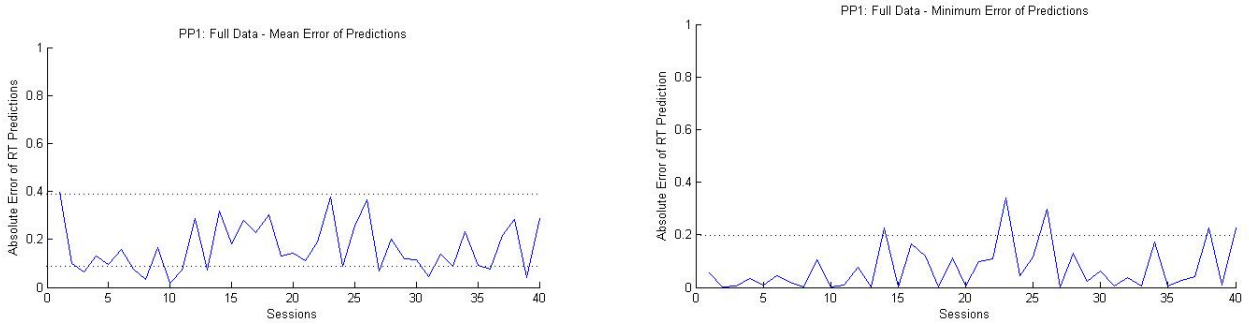
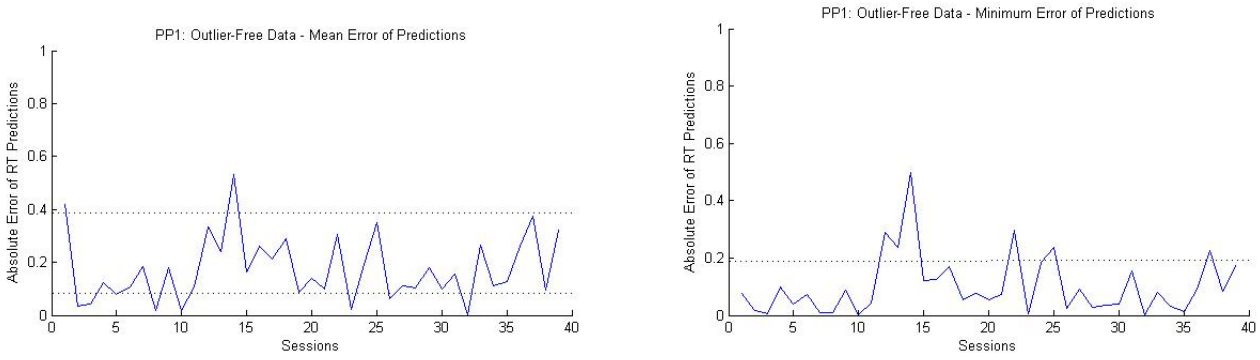


Figure 12

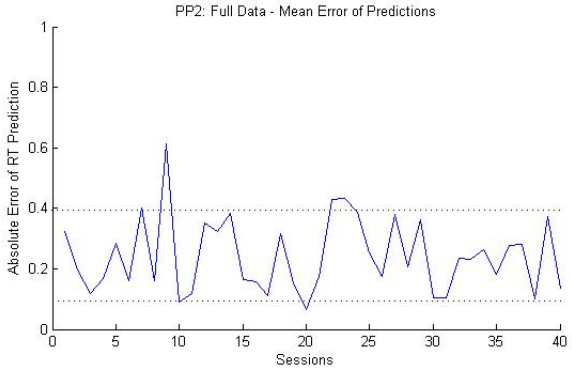


4.2.2 Participant 2

Looking at the results in figures 13 and 14, the first thing to notice is that the removal of the outlier actually worsens the performance of the ANN by noticably increasing minimum error rates, while not having any clear impact on mean error rates. Looking at the full data set, and the respective mean and minimum error rates, we can see that for most data points, the ANN reaches a near perfect prediction

at some point. This is indicated by the near-zero minimum error rate. The maximum deviation of the minimum error is around 0.2 and indicates a reaction time prediction error of about 29 milliseconds. The full range of reaction time was 145 milliseconds from minimum to maximum speed. The mean error of the ANN prediction per point was around the 0.3 mark, which indicates an error of about 43 milliseconds, with one spike shooting clear out to the 0.6 error of about 87 milliseconds.

Figure 13



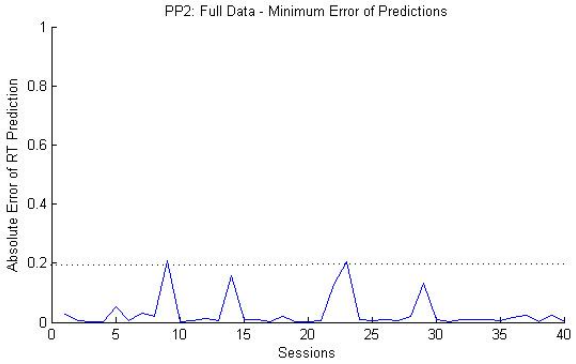
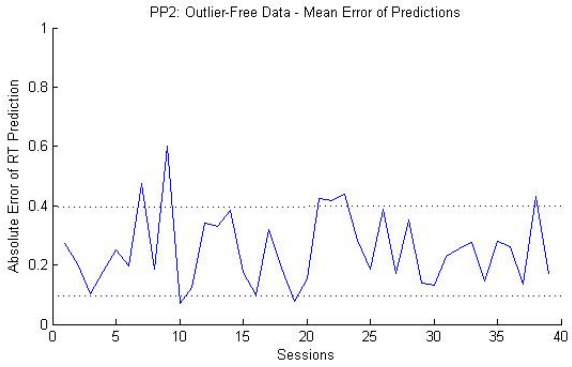
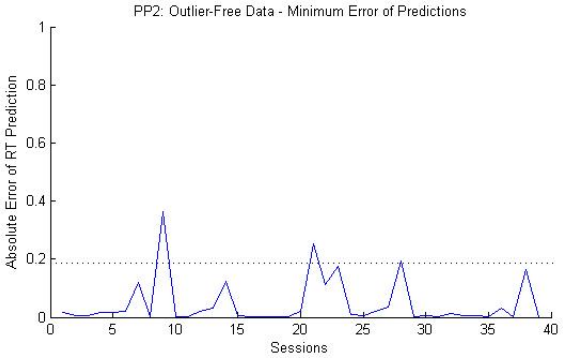


Figure 14





4.2.3 Participant 3

The results in Figure 15 and 16 show that outlier removal for this participant had no noticable impact on mean error prediction, but did "flatten" some of the minimum error prediction spikes. Especially the strong error prediction of session 9 was largely mitigated by removing session 13 from the data set. Overall, mean prediction errors are again between the 0.1 and 0.4 mark, centering around 0.3. For this participant that comes down to a mean error rate between 18 and 72 milliseconds with most errors lying around the 54 millisecond mark. Minimum errors were near 0 milliseconds and up to around the 0.2 mark of about 36 milliseconds. All this was over a variability range of 180 milliseconds between minimum and maximum reaction time over all 40 sessions.

4.2.4 Participant 4

Looking at the results in Figures 17 and 18, it is clear that the removal of the outlier did not create a noticable improvement in performance. While mean error rate gravitates around the 0.3 mark again, the minimum error rate frequently approaches 0.0. This comes down to a mean prediction error of about 66 milliseconds over a 220 millisecond range for this participant, and a minimum error rate approaching zero. It is noticable that the minimum and average error rate increased with the sessions, while the participant

Figure 15

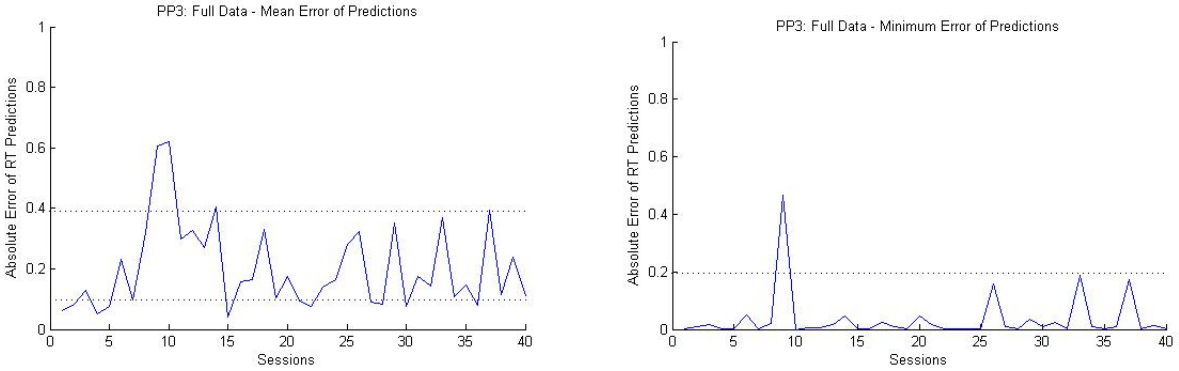
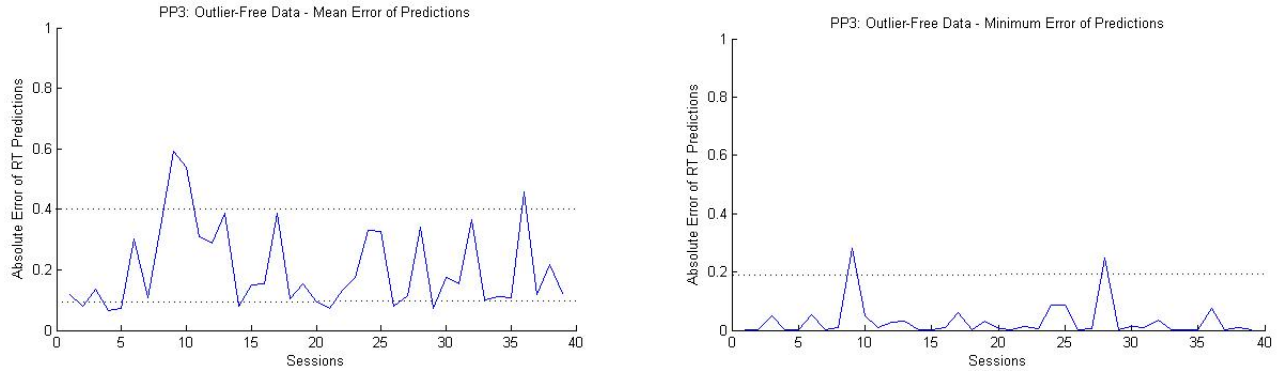


Figure 16



indicated that the experiment was "sleep-inducing" and extremely disagreed with the statement that "it got easier with time". The resulting spikes were in the 0.4 to 0.5 range for mean error rate, and in the 0.2 range for the minimum error rate. This is equal to an 88-110 millisecond error for the spikes in mean error predictions, and a 44 millisecond error for the extremes in minimum error predictions.

Figure 17

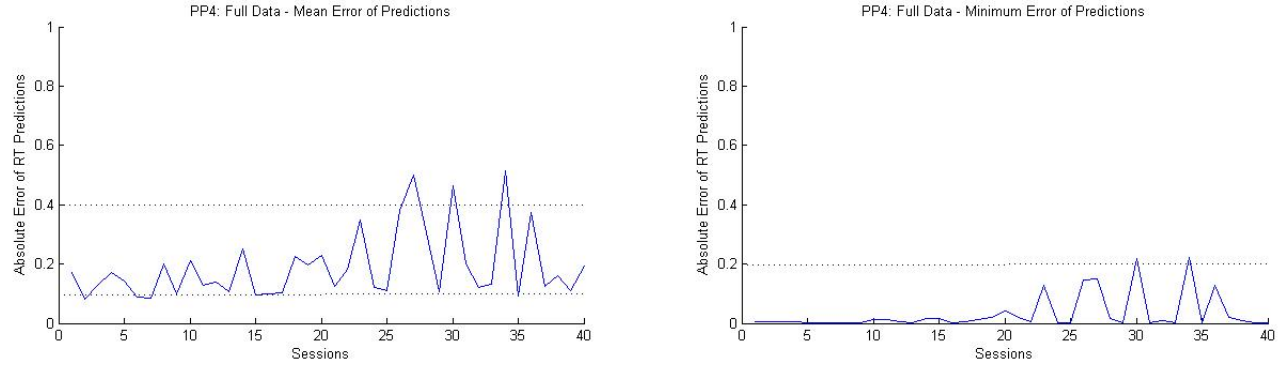
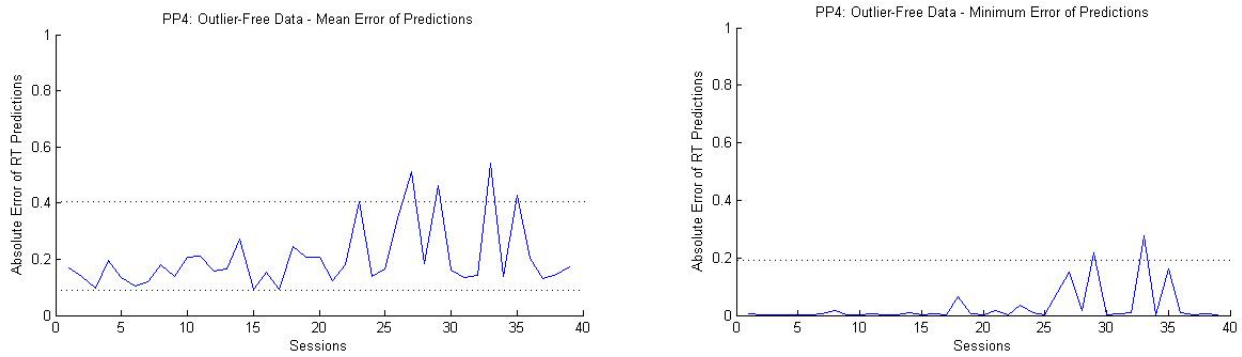


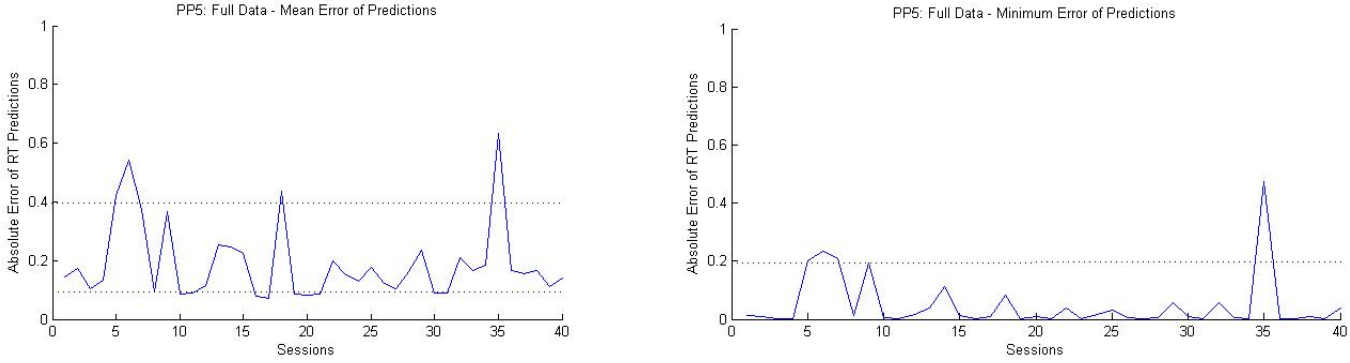
Figure 18



4.2.5 Participant 5

The results in Figure 19 show mean and minimum error rates in the same range as previous participants. That is to say, mean error rates were between 0.1 (18ms) to 0.4 (74ms) and minimum error rates mostly approximated 0.0, with a handful of spikes going in the direction of the 0.2 (37ms) mark. There is one notable exception for the mean and minimum prediction of session 35. The mean error there was around 0.7 (130ms) and the minimum almost reaches the 0.6 (111ms) mark. The range of minimum to maximum reaction times for this participant was 186 milliseconds.

Figure 19



5 Discussion

The nature of this research was intentionally exploratory, and specifically intended to determine the potential and feasibility of using physiological measures in determining performance, in this case embodied by reaction time. Time, resources, and even preceding research to model our current project on was scarce. Given these circumstances, there are a number of factors that could lead to a vast improvement in the results found. We will review these in turn and also highlight peculiarities in the current research that might be avoided or exploited.

5.1 Participants

The current research was done with 5 participants, which is clearly a very limited number that was decided upon because of time restraints as well as a limited pool of participants to draw from. Participants were not compensated for their efforts, while the experiment was time-consuming and purposefully strenuous. This did not only limit the number of people available and willing to sign up, but might also have

impacted motivation during the experiment. It might be the case that some form of compensation would have increased compliance with the experiment instructions. For instance, if participant 5 had been told that she would not receive compensation if she did not sit still for the majority of the experiment, then she might have been more able to remain calmly seated. In the same way, a wider participant pool would have made it easier to avoid working with people with extremely low motivation like participant 2. In hindsight there is no way to tell what impact motivation and possible compensation or wider participant offer could have made on the results, but it is a reasonable factor to keep in mind if anyone wishes to reproduce this research.

5.2 Measurements

For this experiment, it was decided to use the ECG, GSR, masseter and corrugator activity as physiological measures, for the reasons explained in the background and methods sections. However, including EEG readings into the mix is also a promising avenue of exploration. We decided against this option because of the inherent complexity of this measure as well as the increase in the already considerable experimental strain on the participants. The lack of feasibility of this measure in this experiment does not detract from the fact that it would be very interesting to explore what improvements might be made in the ANN with the addition of this measure.

Secondly, in our background section we indicated that the zygomatic muscle of the face could offer useful input for the ANN to use in its predictions. However, during the pilot study it became clear that with the available measurement instruments, it was not possible to accurately measure activity in this muscle. The signals from the zygomatic and masseter were identical on each person this was tested on, and additionally the zygomatic sensors were very hard to place correctly. As such, we dropped this measure. However, it would be interesting to see if it still shows promise when more robust measurement techniques are used.

Lastly, there is the consideration of the fallibility of the measurement instruments themselves. The clearest example is participant 5, who's body movement was probably measured more than her ECG activity. This might have been avoided with more reliable sensors than the cupelectrodes that we used. Either way, the accuracy of the predictions of the ANN are per definition limited by the quality of the physiological measurements. As technology offers us increasingly accurate and reliable measurement techniques, this method of predicting reaction times might become more and more reliable as well. It is conceivable that the current prediction errors are largely due to measurement errors, but we cannot know this for sure until we gain access to sufficiently robust and accurate measurement techniques.

5.3 Reaction Time Task

We measured reaction time using our own custom made reaction time task. The reasoning behind the design was solid in combining the compactness of the short sessions [27] with the cognitive load of the Stroop task [24]. However, it was underestimated how many errors would be made in the task. Due to the repetitive and mentally exhausting nature of the experiment, mistakes were much more frequent than anticipated. Before the experiment, we estimated that participants would learn to make near-zero mistakes at all times after the training phase. However, during the experiment it became clear that there was a great variability in how error-prone the participants were. Additionally, none of them reached the near-zero error performance we had expected. While we have mentioned the error rates, we have not integrated them in our predictions because we did not want to stray from our central research question. Also, we could not grant the ANN access to error rates per session as an input, as the intention of the whole project was to *predict* performance in advance, and so the ANN should never have access to information that can only be gleaned *after* the participant has performed the task. The question is if we can retain the current cognitive load of the task while removing the ability to make mistakes at all.

Right now, the ability to make an error in judgement constitutes most of the cognitive load of the task. Additionally, reaction times in such a complex error-risk environment might allow for much more relevant predictions of reaction time as most real life tasks indeed always contain a component of error-risk. It might be that when a person has no chance of making a mistake and simply needs to respond, that a different mental circuitry is accessed than when erros are possible. For that reason, we would like to argue for retaining this component of the task, but always keeping an eye on it and trying to minimize it. Ways of minimizing it might be more extensive training and including more motivated participants.

Secondly, due to the strain of repetition inherent in the reaction time task, as well as the limited time available for the experiment, it was decided to only gather 40 data points per participant. More promising results might be obtained with a larger data set per participant. However, to achieve this it might be necessary to employ more motivated participants to begin with.

5.4 Stressor

White noise was chosen as a mild stressor because it was feasible, non-invasive and had precedents for successful use. However, during the experiment, it seemed that none of the participants found the noise overly stressful. Instead they indicated that it was the tiring and repetitive nature of the experiment that created stress. In a way this muddles the distinction between hypo- and hyperstress, as the participants struggled to perform well in an understimulating environment. Other more salient stressors might produce much stronger effects and offer a wider sprectrum of physiological profiles and associated reaction times.

5.5 Stress & Performance

In this experiment we only directly measured performance by means of reaction times. Stress was indirectly measured by the physiological readings, and the participants were directly asked about their experience once the experiment was over. From these results it became clear that the main source of stress for the participants was the prolonged low arousal state that the experiment created. They all indicated that they had trouble staying alert, which means they were experiencing hypostress. Paradoxically, the experience of hypostress and the realization that their performance might suffer, actually increased alertness as they kept self-correcting their lagging attention. This goes to show again how complex a matter stress and its relation to performance truly is. Conceivably we can get the most complete stress-arousal-performance profile from a person by subjecting them to a wide spectrum of varying forms and intensities of stress. This is of course an unattainable ideal with the current level of technology and ethical considerations being what they are. However, it should be possible to at least sample a wider spectrum of states and responses by more effectively generating hyperstress. The above paragraph about stressors is relevant here, as it all comes down to creating a sufficiently stressful environment for the participant.

5.6 Analysis

Due to the limited size of the data set, we employed the cross validation technique to extract as many meaningful results from this as we could. For this cross validition we used a k-1 approach, training the network on all points but one, and then using the one left out point as test set. A larger data set would both allow a more rich training set as well as a more representitive test set. Thus a straightforward increase in the sample size per participant would already strengthen the reliability of the results tremendously.

5.7 Outliers

For all participants but one, we found at least one extreme outlier. However, after removal of this outlier the predictions of the ANN did not become more accurate across the board. Either the ANN is so robust that it was not influenced much by the outlier to begin with, or the outlier actually contained meaningful data that lined up with the rest of the data set. In that last case, the outlier would follow the same overall trend the ANN is picking up, so it's removal would not change anything. Either way, what is clear is that considering the noiseness of the data and the near-absent effect of outlier removal, we can consider meriting the inherent robustness of the ANN method. This in turn strengthens the argument for our choice for this AI method, which was more closely considered in the background section.

5.8 Error Rate

For this experiment, each ANN was trained within a 1% error rate of the correct answer, before it was exposed to the test set. All ANNs reached this error rate before hitting the maximum of 10,000 training

epochs. For determining the accuracy of prediction on the test cases we only had the error rate measure at hand. As seen in the results section, minimum errors were often near zero, with a handful of spikes per participant reaching to the 0.2 mark and beyond. Average error rates hovered around the 0.3 mark. Though these measures give an intuitive and absolute view of the accuracy of the predictions, there is no significance test we can perform to determine if the results are likely due to chance or not. With this in mind, we can still determine a few basic benchmarks. The full range of error values is from 0 to 1. As such, an error value between 0 and 0.1 can reasonably be considered a very accurate prediction. An error value between 0.1 and 0.3 can be considered low. An error value of 0.3 to 0.5 medium, and an error value above 0.5 will be considered high. These determinations are subjective, but we consider them to be reasonable given the scale of possible error values. In this light, mean errors were low to medium mostly, while minimum errors were mostly near-perfect to low. Although we are not currently aware of a more objective criteria to judge the performance of the ANN, it might still be possible to find one.

5.9 Iterations

Per data point that we wanted to predict, we ran 50 iterations of the neural network training. Due to the random initial weights used in the Encog library, we cannot be sure we find the best network, but we can increase our chances of doing so by simply repeating the process. Our initial choice for iterations was 20, but during the recalculation of one participant's results, we found that we had reached noticably different minimum error rates. For that reason, we increased the iterations for all participants to 50. Objectively, more iterations are better, but practically speaking we have to draw the line somewhere due to calculation time. In this case that became 50 iterations, but anyone aiming to reproduce this type of research should be cautioned to explore the relative payoff for increased iterations.

5.10 Practical Application

We set out on this research project with the MECA tool suite in mind, and the intention to provide a meaningful addition to it that would help an astronaut perform and remain in peak condition. The potential of this research, as discussed above, not only lies in finding more accurate results, but also in making the measurement process more practical in nature. In our experiment, we had to apply eight electrodes plus a GSR sensor to gather all our data. Once an ANN is optimally trained, an astronaut or other individual in a high-performance job, would have to hook himself up to all these sensors and wait for the signals to level out, before the ANN can give a prediction about reaction times. This is not very time-efficient. Especially if we consider that the results from the ANN are most useful when the individual is not in peak condition, because then he can be warned about that. However, that also means the individual is more likely to make mistakes in applying the sensors or other possible parts of the procedure. Thus making it more likely to offer up contaminated and less reliable data when solid data would be most important.

However, this research was exploratory in nature, and other more feasible ways of testing might be available. It is conceivable that the measurements necessary become integrated into one packaged sensor machine that can be more easily applied. It is also conceivable that some measures that we used now do not substantially add to the accuracy of the predictions, and so they can be left out, simplifying the procedure even further. Additionally, some measures like EEG might show more promise, and have fairly simple sensor application methods that are less error-prone and time-consuming. This is only true of some EEG measuring techniques.

Lastly, the usefulness of this reaction time prediction system is limited to situations where the individual knows that a highly-demanding task is coming up, but he has time to gauge his reaction times with this system. Arguably it is more efficient in such a situation to simply let the person perform a reaction time task so you have the direct measure of their performance. However, this would put an unnecessary cognitive load on the individual, using up mental resources that could otherwise be employed for the demanding task ahead. Additionally, it is conceivable that at some point in the future, physiological readings can be made continuously as sensors become less and less obtrusive. In such a situation, an ANN could give real time predictions of expected reaction times. For the core focus group of astronauts on long duration missions, it is reasonable to think that such advancements in measurement techniques have been made by the time that long duration missions, like a Mars expedition, become feasible.

All in all, there is the potential to adapt this physiology-based safety valve technique in high-performance situations. Measurement techniques would have to be adapted and optimized, and predictions would only be useful in situations where the individual has a choice about doing a certain demanding procedure right now or not. Still, when a lot is at stake, it is important to know that people are up to the job. Even if they have to do the job anyway, it might help them to know that they are not in peak condition so they do not overextend themselves.

6 Conclusion

We set out to answer the question "How well can an ANN predict an individual's reaction time based on the physiological data from that person?". We found that even though the individual differences in the data set from each participant were noticeable, the eventual accuracy of the ANN's predictions ended up in the same range for both mean error and minimum error rate. While mean error rate values were spread along the low to medium range of 0.2 to 0.4, minimum error rates were very accurate at near-zero values with a few spikes that mostly reached into the low error range of 0.2. Given the small sample size, and less-than-perfect compliance of at least two of the five participants, this is a very promising result. Additionally, it's the *best* case results (minimum error rate per case), that shows the true potential of the ANN method to accurately predict reaction times based on physiological variables. In the majority of the cases, for all participants, it is true that an ANN can be trained to a near-zero prediction error. This shows promise for experiments set up on a grander scale and taking into account all the points mentioned in the discussion.

References

[1] Stress and Psychological Disorders, 2011.

[2] M. T. Allen, A. J. Boquet, and K. S. Shelley. Cluster analyses of cardiovascular responsivity to three laboratory stressors. *Psychosomatic Medicine*, 53(3):272–288, May 1991.

[3] Omar Alzoubi, Md Hussain, Sidney D'Mello, and Rafael Calvo. Affective Modeling from Multichannel Physiology: Analysis of Day Differences Affective Computing and Intelligent Interaction. volume 6974 of *Lecture Notes in Computer Science*, chapter 4, pages 4–13. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2011.

[4] I. A. Basheer and M. Hajmeer. Artificial neural networks: fundamentals, computing, design, and application. *Journal of Microbiological Methods*, 43(1):3–31, December 2000.

[5] Guillaume Chanel, Julien Kronegg, Didier Grandjean, Thierry Pun, Guillaume Chanel, Julien Kronegg, Didier Grandjean, and Thierry Pun. Emotion Assessment: Arousal Evaluation Using EEG's and Peripheral Physiological Signals Multimedia Content Representation, Classification and Security. In Bilge Gunsel, Anil Jain, A. Tekalp, Bülent Sankur, Bilge Gunsel, Anil K. Jain, A. Murat Tekalp, and Bülent Sankur, editors, *Multimedia Content Representation, Classification and Security*, volume 4105 of *Lecture Notes in Computer Science*, chapter 70, pages 530–537. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2006.

[6] Christian Collet, Claire Petit, Alain Priez, and André Dittmar. Stroop colorword test, arousal, electrodermal activity and performance in a critical driving situation. *Biological Psychology*, 69(2):195–203, May 2005.

[7] Pamela J. Feldman, Sheldon Cohen, Natalie Hamrick, and Stephen J. Lepore. Psychological stress, appraisal, emotion and Cardiovascular response in a public speaking task. *Psychology & Health*, 19(3):353–368, June 2004.

[8] P. A. Hancock and J. S. Warm. A Dynamic Model of Stress and Sustained Attention. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 31(5):519–537, October 1989.

[9] A. K. Jain, Jianchang Mao, and K. M. Mohiuddin. Artificial neural networks: a tutorial. *Computer*, 29(3):31–44, March 1996.

[10] Joni Kettunen, Niklas Ravaja, Petri Näätänen, Pertti Keskivaara, and Liisa Keltikangas-Järvinen. The synchronization of electrodermal activity and heart rate and its relationship to energetic arousal: a time series approach. *Biological Psychology*, 48(3):209–225, August 1998.

[11] Sylvia D. Kreibig. Autonomic nervous system activity in emotion: A review. *Biological Psychology*, 84(3):394–421, July 2010.

[12] David M. Musson, Gro Sandal, and Robert L. Helmreich. Personality Characteristics and Trait Clusters in Final Stage Astronaut Selection. *Aviation, Space, and Environmental Medicine*, pages 342–349, April 2004.

[13] M. A. Neerinckx. Situated Cognitive Engineering for Crew Support in Space. *Pers Ubiquit Comput*, 15:445–456, 2011.

[14] L. A. Palinkas. Psychosocial issues in long-term space flight: overview. *Gravitational and space biology bulletin : publication of the American Society for Gravitational and Space Biology*, 14(2):25–33, June 2001.

[15] R. W. Picard, E. Vyzas, and J. Healey. Toward machine emotional intelligence: analysis of affective physiological state. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(10):1175–1191, October 2001.

[16] R. M. Rose, L. F. Fogg, R. L. Helmreich, and T. J. McFadden. Psychological predictors of astronaut effectiveness. *Aviation, space, and environmental medicine*, 65(10 Pt 1):910–915, October 1994.

[17] Jesús F. Salgado. Predicting job performance using FFM and non-FFM personality measures. *Journal of Occupational and Organizational Psychology*, 76(3):323–346, 2003.

[18] G. M. Sandal, G. R. Leon, and L. Palinkas. Human challenges in polar and space environments Life in Extreme Environments. In Ricardo Amils, Cynan Ellis-Evans, and Helmut Hinghofer-Szalkay, editors, *Life in Extreme Environments*, chapter 25, pages 399–414. Springer Netherlands, Dordrecht, 2007.

[19] H. Selye. Forty years of stress research: principal remaining problems and misconceptions. *Canadian Medical Association journal*, 115(1):53–56, July 1976.

[20] Henrique Sequeira, Pascal Hot, Laetitia Silvert, and Sylvain Delplanque. Electrical autonomic correlates of emotion. *International Journal of Psychophysiology*, 71(1):50–56, January 2009.

[21] N. Smets. Improving Crew Support Methods in Human-Machine Teams for Long-Duration Missions. *63rd International Astronautical Congress*, 2012.

[22] Franklin Stein. Occupational stress, relaxation therapies, exercise and biofeedback. *Work: A Journal of Prevention, Assessment and Rehabilitation*, 17(3):235–245, January 2001.

[23] Chad L. Stephens, Israel C. Christie, and Bruce H. Friedman. Autonomic specificity of basic emotions: Evidence from pattern classification and cluster analysis. *Biological Psychology*, 84(3):463–473, July 2010.

[24] J. R. Stroop. Studies of interference in serial verbal reactions. *Journal of Experimental Psychology*, 18:643–662, 1935.

[25] Robert E. Thayer. Toward a psychological theory of multidimensional activation (arousal). *Motivation and Emotion*, 2(1):1–34, March 1978.

[26] J. Wagner, J. Kim, and E. Andre. From Physiological Signals to Emotions: Implementing and Comparing Selected Methods for Feature Extraction and Classification. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 940–943. IEEE, July 2005.

[27] R. T. Wilkinson and D. Houghton. Field Test of Arousal: A Portable Reaction Timer with Data Storage. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 24(4):487–493, August 1982.

[28] Glenn F. Wilson and Christopher A. Russell. Real-Time Assessment of Mental Workload Using Psychophysiological Measures and Artificial Neural Networks. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 45(4):635–644, December 2003.

[29] Robert M. Yerkes and John D. Dodson. The relation of strength of stimulus to rapidity of habit-formation. *Journal of Comparative Neurology and Psychology*, 18:459–482, 1908.

Table 11: Questionnaire Results

	A	B	C	D	E	Full Text Feedback
pilot	7	2	6	4	3	Koud, Vermoeiend, Geestdodend, Film was wel leuk ondanks al gezien
1	7	1	7	4	2	Leuk maar toch vermoeiender dan gedacht. Moeite om mijn ogen open te houden in de pauzes, koud en een beetje ongemakkelijk qua houding omdat ik wat geforceerd stil zat. Film was super leuk jammer dat die uit moest!
2	7	6	2	1	1	Belachelijk, verschrikkelijk en onethisch. Ik zou graag willen eisen dat de begeleiders van dit project de volledige test zelf eens zouden ondergaan. Koude voeten, oncomfortabele stoel, irritatie van de huid, pijnlijke huid na behandeling met de alcohol, uitzichtloos, gebrek aan informatie en geen beloning voor de gedane moeite. In mijn ogen rond uit onacceptabel.
3	7	2	7	6	4	De taak is weinig uitdagend en cognitief vermoeiend. Mijn aandacht werd soms afgeleid van de taak (afdwalende gedachten). Het voelde soms of ik automatisch in veel gevallen 'N' drukte omdat dat relatief meer voorkwam. Stilzitten is ook niet mijn sterkste punt.
4	7	3	7	7	1	Slaapverwekkend
5	6	5	6	2	2	Het experiment was erg vermoeiend en saai. De sensoren wond ik wel interessant, maar daar merk je al gauw weinig meer van. Ik vond het voor mijn gevoel erg lang duren. Was in het begin wel leuk, later vooral saai en vervelend.

Legenda: A) Experiment Vermoeiend; B) Sensoren Storend; C) Film ontspannend; D) Weer meedoen; E) Werd Makke-lijker

A Questionnaire Results

Participants were native Dutch, and the questionnaire was conducted in that language. The results are shown in Table 11.

B MatLab Code: Data Processing

```
%"data" is a matrix containig all data collected by the Matlab Simulink
%"javadata" is a matrix containing all data collected by the custom RT task


%Correct data if necessary (only for pp1)

temp = size(data(10,:));    %count number of data points
datalengte = temp(1,2);     %store number of datapoints
sessiondata = data(10,:);   %take session data
for i = 800000:datalengte
    %Corrects measurement error in the data of pp1.
    %Not present in other pp data.
    if data(10,i) >= 0
       sessiondata(1,i) = sessiondata(1,i)+20;
    end
end

%Cut out only the relevant data from the entire stream

delen = 40;     %number of sessions
eindpunt = max(find(sessiondata(1,:) == 39)); %Last measurement point of the last session
firstm1 = find(sessiondata(1,:) == -1, 1, 'first'); %start of first session
first = find(sessiondata(1, firstm1:eindpunt) == 0, 1, 'first'); %interval from first session to end
beginpunt = (firstm1 + first - (256*60)); %Point one minute before the start of the first session
session = sessiondata(1,beginpunt:eindpunt);  %all session data without dead areas before/after measurements

%Determine lengths of the sessions and intervals so each session and
%interval can be directly accessed

sessielengte = zeros(delen,1);              %matrix of session lengths
intervallengte = zeros(delen,1);            %matrix of interval lengths
intervallengte(1) = 256*60;                 %first interval is exactly 1 minute
sessielengte(1) = find(session == 0, 1, 'last') - (256*60); %first session length

for n = 2:delen
    %Determine length of the n-th session
    l = size(find(session==n-1));
    sessielengte(n) = l(1,2);

    %move up the window of what session/interval we're looking at
    begin = find(session == n-2,1,'last');
    ender = find(session == n-1, 1, 'first');

    %Determine length of the n-th interval
    intervallengte(n) = ender - begin-1;
end

%Splitting off the relevant datastreams for the relevant interval from the
%entire data set.

ecg = data(2, beginpunt:eindpunt);
gsr = data(3, beginpunt:eindpunt);
cor = data(4, beginpunt:eindpunt);
mas = data(6, beginpunt:eindpunt);
rt = data(12,beginpunt:eindpunt);
c1 = data(13,beginpunt:eindpunt);
c2 = data(14,beginpunt:eindpunt);
```

```
%ECG - Determining the heart rate during each interval

ecgbeats = zeros(delen,max(intervallengte));    %empty matrix for beat moments
beatthreshold = 2000;                           %value above noise and below heart beat peak
vorigpunt = -9999;                              %previous point
vorigepiek = -9999;                             %previous peak
state = 1;                                      %state 1 is climbing, state -1 is descending
tempstartindex = 0;                             %initialize indexing

for sessie=1:delen

    %intialize values per session
    templengte = intervallengte(sessie);
    vorigpunt = -9999;
    vorigepiek = -9999;
    state = 1;

    %determine the moments of the heart beat peaks
    for i=1:templengte
        punt = ecg(1,tempstartindex+i);         %current point
        if(punt > vorigpunt)
            state = 1;                          %if current point is higher than the previous point,
                                                %then the state is climbing.
        elseif(punt == vorigpunt)
            % Empty case: current point equal to previous point
        elseif(punt < vorigpunt)
            if(state == 1)
                if(vorigpunt > beatthreshold)
                    %if current is lower than previous point, and we were
                    %climbing, and we are above the beatthreshold, then
                    %that was the peak of a heartbeat.
                    ecgbeats(sessie,i-1) = 1;
                end
            end
            state = -1;                         %state is now descending
        end
        vorigpunt = punt;                       %previous point is current point
    end
    %update startindex
    tempstartindex = tempstartindex + intervallengte(sessie) + sessielengte(sessie);
end

%heartrate is equal to the number of peaks found
heartrate = sum(ecgbeats,2);


%Determining HRV

AVNN = zeros(delen,1);                          %empty array for average interval between heartbeats.
SDNN = zeros(delen,1);                          %empty array for st. dev. of the interval between heartbeats.
RMSSD= zeros(delen,max(heartrate));             %empty array for Root Mean Square of Successive Differences
                                                %of the interval between heartbeats.

for sessie=1:delen
    %intialize values per session
    iterator = 0;
    temp = zeros(heartrate(sessie)-1,1);
    templengte = intervallengte(sessie);

    %determine the intervals between heartbeats.
    for j=1:templengte
```

```
        if (ecgbeats(sessie, j)==0 && iterator == 0)
           %Do nothing, don't count first zeros
        elseif (ecgbeats(sessie, j)==0 && iterator < heartrate(sessie))
           temp(iterator) = temp(iterator) + (1/256);
        elseif (ecgbeats(sessie, j)==1 && iterator < heartrate(sessie))
            iterator = iterator + 1;
        end
    end

    AVNN(sessie) = mean(temp);  %determine average interval between heartbeats.
    SDNN(sessie) = std(temp);   %determine standard deviation of the interval between heartbeats.

    %determine RMSSD
    for l=1:(heartrate(sessie)-1)
       temp(l) = ((temp(l)*256)/1000)^2; %Convert to millisenconds and then square it
    end

    for q=3:(heartrate(sessie)-4)
       RMSSD(sessie,q-2) = sqrt(mean(temp(q-2:q+3)));
    end
end

meanRMSSD = mean(RMSSD,2);  %determine average RMSSD per interval

%Determine GSR

gsrsessies = zeros(delen, max(intervallengte)); %empty array of gsr values
tempstartindex = 1;                             %initialize start index

for sessie=1:delen
    %for each interval, take the relevant GSR data
    templengte = intervallengte(sessie);
    gsrsessies(sessie,1:templengte) = gsr(1, tempstartindex:(tempstartindex + templengte-1));
    tempstartindex = tempstartindex + intervallengte(sessie) + sessielengte(sessie);
end

gsrmean = sum(gsrsessies, 2); %GSR values per session are taken as the sum total
                              %of the activity during that interval

%Masseter & Corrugator bepalen

masspos = abs(mas);     %absolute values for the masster
corpos = abs(cor);      %absolute values for the corrugator

massrms = zeros(delen, max(intervallengte));    %empty array for masseter root mean squared values
corrms = zeros(delen, max(intervallengte));     %empty array for corrugator root mean squared values

tempstartindex = 1;     %initialize index

for sessie=1:delen
    %initialize length variable
    templengte = intervallengte(sessie);

    %RMS values are calcluated for a window of 14 data points.
    for k=(7):(templengte-7)
        tempmass = masspos(1,(tempstartindex+k-6):(tempstartindex+k+6));
        tempcor = corpos(1,(tempstartindex+k-6):(tempstartindex+k+6));
        massrms(sessie,k) = sqrt(mean(tempmass.^2));
        corrms(sessie,k) = sqrt(mean(tempcor.^2));
    end
```

```
    %update start index
    tempstartindex = tempstartindex + intervallengte(sessie) + sessielengte(sessie);
end

masssum = sum(massrms, 2);  %sum of all rms values are used as the masseter session value
corsum = sum(corrms, 2);    %sum of all rms values are used as the corrugator session value


%Reaction Time

lossewaardes = zeros(delen, 30);    %empty array for individual reaction time values per word
meanrt = zeros(delen,1);            %empty array for average reaction times
tempstartindex = 0;                 %initialize start index



for sessie=1:delen
    %initialize variables
    tempstartindex = tempstartindex + intervallengte(sessie);
    counter = 1;

    %determine individual reaction times
    for s=tempstartindex:(tempstartindex+sessielengte(sessie))
        if rt(s) == 0 | rt(s) == -1
            %do nothing
        elseif lossewaardes(sessie, 1) == 0
           lossewaardes(sessie,counter) = rt(s);
           counter = counter +1;
        elseif rt(s) ~= lossewaardes(sessie,counter-1)
           lossewaardes(sessie,counter) = rt(s);
           counter = counter +1;
        end
    end

    %update start index
    tempstartindex = tempstartindex + sessielengte(sessie);

    %determine mean reaction time
    ttemp = find(lossewaardes(sessie, :) ~= 0);
    rtsession = lossewaardes(sessie, ttemp);
    meanrt(sessie) = mean(rtsession);
end


mistakessessies = zeros(delen, 1);  %empty matrix for mistakes per session
temp = size(javadata);              %determine number of entries per session
inputslength = temp(1,1);           %set entries per session as length of input array

for i = 1:inputslength
    %load data for the relevant session
    sessie = javadata(i, 1);

    %count mistakes per session
    if javadata(i, 6) == 1 && javadata(i, 5) ~= javadata(i, 4)
        mistakessessies(sessie+1) =  mistakessessies(sessie+1) +1;
    elseif javadata(i, 6) == 0 && javadata(i, 5) == javadata(i, 4)
        mistakessessies(sessie+1) =  mistakessessies(sessie+1) +1;
    end
end

%Remove Extreme Outliers
```

```
outliers = 0;

%Remove commenting tags from the following code, to run the code without
%outliers in the data. Outliers are different per participant and the
%relevant code shuold be added for each outlier according to the template
%below.

%Remove session 21
%{
heartrate = [heartrate(1:20,:); heartrate(22:40,:);];
AVNN = [AVNN(1:20,:); AVNN(22:40,:);];
SDNN = [SDNN(1:20,:); SDNN(22:40,:);];
meanRMSSD = [meanRMSSD(1:20,:); meanRMSSD(22:40,:);];
gsrmean = [gsrmean(1:20,:); gsrmean(22:40,:);];
masssum = [masssum(1:20,:); masssum(22:40,:);];
corsum = [corsum(1:20,:); corsum(22:40,:);];
meanrt = [meanrt(1:20,:); meanrt(22:40,:);];

outliers = outliers + 1;
%}


%Scaling

goodsessions = delen-outliers; %number of sessions we will be using for this analysis

%Scaled Values
%All Values per session are scaled to the interval [0,1]

scaledhr = zeros(goodsessions,1);
scaledAVNN = zeros(goodsessions,1);
scaledSDNN = zeros(goodsessions,1);
scaledRMSSD = zeros(goodsessions,1);
scaledGSR = zeros(goodsessions,1);
scaledmass = zeros(goodsessions,1);
scaledcor = zeros(goodsessions,1);
scaledrt = zeros(goodsessions,1);

for i = 1:goodsessions
    scaledhr(i,1) = (heartrate(i,1) - min(heartrate))/(max(heartrate)-min(heartrate));
    scaledAVNN(i,1) = (AVNN(i,1) - min(AVNN))/(max(AVNN)-min(AVNN));
    scaledSDNN(i,1) = (SDNN(i,1) - min(SDNN))/(max(SDNN)-min(SDNN));
    scaledRMSSD(i,1) = (meanRMSSD(i,1) - min(meanRMSSD))/(max(meanRMSSD)-min(meanRMSSD));
    scaledGSR(i,1) = (gsrmean(i,1) - min(gsrmean))/(max(gsrmean)-min(gsrmean));
    scaledmass(i,1) = (masssum(i,1) - min(masssum))/(max(masssum)-min(masssum));
    scaledcor(i,1) = (corsum(i,1) - min(corsum))/(max(corsum)-min(corsum));
    scaledrt(i,1) = (meanrt(i,1) - min(meanrt))/(max(meanrt)-min(meanrt));
end

%Create a matrix with all relevant scaled values per session
ProcessedData = [scaledhr scaledAVNN scaledSDNN scaledRMSSD scaledGSR scaledmass scaledcor scaledrt]

[r,p] = corrcoef(ProcessedData) %Correlation of variables among each other

dlmwrite('processeddata.csv', ProcessedData); %Write the full data set to a csv file

%Remove commenting tags from the following code if the predicted values for
%a cross validated k-1 test have been imported as a matrix "predictions"
%for the full data set, and as "PredOutliers" for the data without
%outliers. Update the relevant values of the outliers.
```

```
%This will return mean and minimum differences between predictions and true
%values for both data sets.

%{
differences = zeros(40,20);

for i=1:40
    for j=1:20
        differences(i,j) = abs(predictions(i,j)-scaledrt(i));
    end
end

figure;plot(mean(differences,2));
figure;plot(min(differences2,[],2));

%outlier 21
differences2 = zeros(39,20);
oCounter = 0;

for i=1:39
        for j=1:20
            differences2(i,j) = abs(PredOutliers(i,j)-scaledrt(i));
        end
end

figure;plot(mean(differences2,2));
figure;plot(min(differences2,[],2));
%}
```

C Java Code: Reaction Time Test

CLASS ActionPane.java

```
import java.awt.Color;
import java.awt.Font;
import java.util.Random;

import javax.swing.JPanel;
import javax.swing.JLabel;

//Class describing the display area of the RT task, and its relevant functions.

public class ActionPane extends JPanel
{
  private JLabel    activeLabel;
  private Color[]   colorArray = {Color.RED, Color.BLUE, Color.GREEN, Color.YELLOW};
  private String[]  stringArray = {"ROOD", "BLAUW", "GROEN", "GEEL"};
  private int     numColors = 4;
  private int      x,y;
  private int[]    latestScores, latestMistakes;
  private int      scoreIt;
  private int      averageScore;
  private GUI      gui;


  //Constructor

  ActionPane(GUI gui)
  {
    this.gui = gui;

    this.setLayout(null);
    this.setBackground(Color.BLACK);
    this.setVisible(true);

    Font big = new Font("SansSerif", Font.BOLD, 20);
    activeLabel = new JLabel("Klaar voor de start ...");
    activeLabel.setFont(big);
    activeLabel.setForeground(Color.WHITE);

    this.add(activeLabel);
    activeLabel.setBounds(x/2, y/2, 200, 30);
    activeLabel.setVisible(true);

    latestScores = new int[5];
    latestMistakes = new int[5];
    scoreIt = 0;
    averageScore = Integer.MAX_VALUE;
  }


  //Cycles to the next word-color pair after a random interval of 0-5 seconds.
  //Returns true if color and word match, otherwise false.
  //Color-word pair are randomly selected.
  //Color word pair is returned.

  public int[] cycle()
  {
    activeLabel.setVisible(false);
```

```
  Random r = new Random();
  int randomIndex1   = r.nextInt(numColors);
  int randomIndex2   = r.nextInt(numColors);
  int randomX      = r.nextInt(x);
  int randomY      = r.nextInt(y);


  display(colorArray[randomIndex1], stringArray[randomIndex2], randomX, randomY);

  int[] pair = new int[2];
  pair[0] = randomIndex1;
  pair[1] = randomIndex2;

  return pair;
}


//Displays string s, in color c, at location [x,y].

private void display(Color c, String s, int x, int y)
{
  activeLabel = new JLabel(s);
  Font big = new Font("SansSerif", Font.BOLD, 20);
  activeLabel.setFont(big);
  activeLabel.setForeground(c);
  activeLabel.setBounds(x, y, 100, 20);

  this.add(activeLabel);
  activeLabel.setVisible(true);
  this.repaint();
}

//Takes care of redisplaying the label with new text

public void setLabel(String text)
{
  activeLabel.setVisible(false);

  activeLabel = new JLabel(text);
  Font big = new Font("SansSerif", Font.BOLD, 20);
  activeLabel.setFont(big);
  activeLabel.setForeground(Color.WHITE);
  activeLabel.setBounds(x/2, y/2, 500, 30);

  this.add(activeLabel);
  activeLabel.setVisible(true);
  this.repaint();
}

//Adds scores to the set of five latest scores (for calculating latest average performance)

public boolean addScore(int s, int m)
{
  boolean r = true;

  if(latestScores[4] == 0) //the array is not full
  {
    latestScores[scoreIt] = s;
    latestMistakes[scoreIt] = m;
  }
  else //if the array is full
```

```
  {
    for(int i = 0; i < 5; i++)
    {
      latestScores[i] = 0;
    }

    latestScores[scoreIt] = s;
    latestMistakes[scoreIt] = m;
  }

  if(scoreIt == 4)
  {
    scoreIt = 0;
    r = progress();
  }
  else
  {
    scoreIt++;
  }

  return r;
}

//Check if the user has made any "progress" and updates the GUI accordingly
//Progress is defined as shorter reaction times over the last 5 sessions
//than the last best reaction time over a block of five session.

private boolean progress()
{
  int total = 0;
  float mistakes = 0;

  if(latestScores[4] != 0)
  {
    for(int i = 0; i < 5; i++)
    {
      total = total + latestScores[i];
      mistakes = mistakes + latestMistakes[i];
      System.out.println(i + ": " + latestScores[i]);
    }

    System.out.println("Your previous average score was: " + averageScore);
    System.out.println("Your latest average score is: " + (total/5) + "; Mistakes: " + (mistakes/5));

    //update labels
    gui.interactionPane.setAverageRT(averageScore + "->" + (int)(total/5));
    gui.interactionPane.setAverageMistakes("" + (float)(mistakes/5));

    if(total/5 <= averageScore)
    {
      System.out.println("You have improved!");
      averageScore = total/5;
      setLabel("Je bent beter geworden. Ga verder (F1)");
      return true;
    }
    else
    {
      System.out.println("You have not improved. You have probably reached your optimal performance");
      setLabel("Optimale prestatie bereikt. Een moment aub");
    }
  }
```

```
    return false;
  }


  //Checks the size of the actionpane.

  public void checkSize()
  {
    x = this.getWidth()-100;
    y = this.getHeight()-80;
  }

}
```

CLASS CycleColor.java

```
import java.util.TimerTask;

//Task that is scheduled to show one Stroop word.

public class CycleColor extends TimerTask
{
  private ActionPane ap;
  private int[] pair;

  //Constructor

  CycleColor(ActionPane ap)
  {
    this.ap = ap;
  }

  //Initializes what font color word color pair to show

  public void run()
  {
    pair = ap.cycle();

  }

  //Returns font color word color pair

  public int[] getPair()
  {
    return pair;
  }

}
```

CLASS EndTask.java

```
import java.awt.event.KeyEvent;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.TimerTask;

//Task that is scheduled to calculate all the results of the reaction time task on the Java side.
//RT is also calculated straight from Matlab data.
```

```
public class endTask extends TimerTask
{
  int nrColorTasks, mistakes;
  TimerTask[] tasks;
  KeyEvent[] keyEvents;
  long[] rtArray;
  long rtTotal;
  long rtAverage;

  PrintWriter printer;
  ActionPane actionPane;
  GUI gui;

  //Constructor

  public endTask(int nrColorTasks, GUI gui)
  {
    this.nrColorTasks = nrColorTasks;
    this.tasks = gui.timers;
    this.keyEvents = gui.keyEvents;
    this.printer = gui.printer;
    this.actionPane = gui.actionPane;
    this.gui = gui;
    rtArray = new long[nrColorTasks+1]; //reaction time in milliseconds
    rtTotal = 0;
    mistakes = 0;

  }

  //Calculates reaction times and mistakes for teh last session and print these to the console and a file.
  //Also updates the actionPane to indicate if the user should continue or not (relevant in training mode)

  public void run()
  {

    while(gui.keyCounter <= nrColorTasks)
    {
      gui.sendData(false);
      gui.keyCounter++;
    }

    gui.uc.msgSend("-2 -2 -2 -2 -2 -2");

    for(int i = 0; i <= nrColorTasks; i++)
    {
     //if a button has been pressed and it was pressed AFTER the relevant task
     if(keyEvents[i] != null && rtArray[i] == 0 && keyEvents[i].getWhen() > tasks[i].scheduledExecutionTime())
      {
        //if the button is Y and should have been Y
        if(keyEvents[i].getKeyCode() == KeyEvent.VK_Y &&
            ((CycleColor) tasks[i]).getPair()[0] == ((CycleColor) tasks[i]).getPair()[1])
        {
          rtArray[i] = keyEvents[i].getWhen()- tasks[i].scheduledExecutionTime();

          System.out.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
                ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 1");
          printer.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
                ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 1");

        }
        //if the button is N and should have been N
```

```
    else if(keyEvents[i].getKeyCode() == KeyEvent.VK_N &&
        ((CycleColor) tasks[i]).getPair()[0] != ((CycleColor) tasks[i]).getPair()[1])
    {
      rtArray[i] = keyEvents[i].getWhen()- tasks[i].scheduledExecutionTime();

      System.out.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 0");
      printer.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 0");
    }
    else if(keyEvents[i].getKeyCode() == KeyEvent.VK_N &&
        ((CycleColor) tasks[i]).getPair()[0] == ((CycleColor) tasks[i]).getPair()[1])
    {
      rtArray[i] = keyEvents[i].getWhen()- tasks[i].scheduledExecutionTime();

      System.out.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 0");
      printer.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 0");
      mistakes++;
    }
    else
    {
      rtArray[i] = keyEvents[i].getWhen()- tasks[i].scheduledExecutionTime();

      System.out.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 1");
      printer.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 1");
      mistakes++;
    }


  }
  else if(keyEvents[i] == null) //if no button was pressed, then RT is the time till the next stimulus
  {
    rtArray[i] = tasks[i+1].scheduledExecutionTime() - tasks[i].scheduledExecutionTime();
    System.out.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 9");
    printer.println(gui.sessionCounter + " " + i + " " + rtArray[i] + " " +
        ((CycleColor) tasks[i]).getPair()[0] + " " + ((CycleColor) tasks[i]).getPair()[1] + " 9");

    mistakes++;
  }

  rtTotal = rtTotal + rtArray[i];
}

rtAverage = rtTotal / (nrColorTasks +1);
gui.interactionPane.setRT("" + rtAverage);
gui.interactionPane.setMistakes("" + mistakes + "/" + (nrColorTasks +1));

System.out.println("Average Reaction Time: " + rtAverage + " and mistakes: "+ mistakes+" over " +
    (nrColorTasks+1) + " iterations");

boolean progress = true;

if(gui.training == true)
{
  progress = actionPane.addScore((int)rtAverage, mistakes);
}
```

```
    if(gui.training == true && progress == true)
    {
      actionPane.setLabel("Klaar. Druk op F1 om verder te gaan");
    }
    else if(gui.training == true)
    {
      actionPane.setLabel("Wacht op instructies");
    }
    else if(gui.training == false)
    {
      actionPane.setLabel("Klaar. Een moment geduld ...");
    }


    printer.flush();
  }

}
```

CLASS GUI.java

```
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.text.DateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.Random;
import java.util.Timer;
import java.util.TimerTask;

import javax.swing.JButton;
import javax.swing.JFrame;

//GUI class processes user inputs and manages the interaction and action panes that are displayed.

public class GUI extends JFrame implements KeyListener
{
  ActionPane actionPane;
  InteractionPane interactionPane;

  int keyCounter, runTime, sessionCounter, maxSession, cycles;
  KeyEvent[] keyEvents;
  TimerTask[] timers;
  PrintWriter printer;
  Timer t;
  boolean training;
  UDPClient uc;


  //Constructor

  GUI(int runTime, int x, int y, int maxSession, boolean training, String logName)
  {
    this.runTime = runTime;
```

```
  this.maxSession = maxSession;
  this.training = training;

  actionPane = new ActionPane(this);
  this.getContentPane().add(actionPane, BorderLayout.CENTER);

  interactionPane = new InteractionPane(this);
  this.getContentPane().add(interactionPane, BorderLayout.SOUTH);

  this.setTitle("");
  this.setSize(x, y);
  this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  this.setVisible(true);
  this.addKeyListener(this);
  this.setFocusable(true);

  actionPane.checkSize();

  timers = new TimerTask[runTime];
  sessionCounter = -1;
  keyCounter = 0;
  t = new Timer();
  try {
    uc = new UDPClient(logName);
  } catch (IOException e2) {
    System.out.println("Could not create UDP Client");
    e2.printStackTrace();
  }

  //Create Printer

  File text = new File("text.txt");
  try {
    text.createNewFile();
  } catch (IOException e1) {
    System.out.println("Error creating new file");
    e1.printStackTrace();
  }
  FileWriter dataFile;
  try {
    dataFile = new FileWriter(text);
    printer = new PrintWriter(dataFile, true);
  } catch (IOException e) {
    System.out.println("Error creating filewriter");
    e.printStackTrace();
    printer = null;
  }
}


//Plans all tasks to be executed for one cycle, including white noise

public void run(int seconds)
{
  int counter = 0;
  int remaining = seconds;
  int interval;

  keyCounter = 0;
  sessionCounter++;
```

```
  timers = new TimerTask[seconds];
  keyEvents = new KeyEvent[runTime];

  Random r = new Random();

  actionPane.setLabel("Klaar voor de start");

  uc.msgSend("-1 -1 -1 -1 -1 -1");


  while(remaining >= 0)
  {
    interval = (r.nextInt(3) +2);
    timers[counter] = new CycleColor(actionPane);

    remaining = remaining - interval;
    t.schedule(timers[counter], (seconds-remaining)*1000);
    counter++;
  }

  timers[counter] = new endTask(counter-1, this);
  t.schedule(timers[counter], (seconds+5)*1000);

  if(training == false)
  {
    t.schedule((TimerTask) (new WNTask(runTime/10)), (seconds+5)*1000);
  }

  cycles = counter;

}


//Sends data to the UDPServer.

public void sendData(boolean buttonPressedForStimulus)
{
  String data = sessionCounter + " " + keyCounter;

  if(buttonPressedForStimulus == true)
  {
    data = data + " " + (keyEvents[keyCounter].getWhen() - timers[keyCounter].scheduledExecutionTime()) +
      " " + ((CycleColor) timers[keyCounter]).getPair()[0] + " " +
          ((CycleColor) timers[keyCounter]).getPair()[1];

    if(keyEvents[keyCounter].getKeyCode() == KeyEvent.VK_Y)
    {
      data = data + " " + "1";
    }
    else
    {
      data = data + " " + "0";
    }
  }
  else
  {
    data = data + " " +
      (timers[keyCounter+1].scheduledExecutionTime() - timers[keyCounter].scheduledExecutionTime()) +
      " " + ((CycleColor) timers[keyCounter]).getPair()[0] + " " +
      ((CycleColor) timers[keyCounter]).getPair()[1];
    data = data + " 9";
```

```
  }

  uc.msgSend(data);
}


//Processes Key Pressed Events.
//All Y and N presses are saved.
//F1 starts a cycle

public void keyPressed(KeyEvent e)
{
  //if there was an event and all relevant keypresses haven't been recorded yet
  if(timers[cycles] != null  && keyCounter != cycles && keyEvents[cycles-1] == null &&
      (e.getKeyCode() == KeyEvent.VK_Y || e.getKeyCode() == KeyEvent.VK_N))
  {
    //if the key was pressed after the relevant event and before the next
    if(e.getWhen() > timers[keyCounter].scheduledExecutionTime() &&
      e.getWhen() < timers[keyCounter+1].scheduledExecutionTime())
    {
      keyEvents[keyCounter] = e;
      sendData(true);
      keyCounter++;
    }
    else if(keyEvents[keyCounter] == null &&
      e.getWhen()>timers[keyCounter].scheduledExecutionTime())
    {
      while(keyEvents[keyCounter] == null)
      {
        sendData(false);
        keyCounter++;

        if(e.getWhen() >= timers[keyCounter].scheduledExecutionTime() &&
          e.getWhen() <= timers[keyCounter+1].scheduledExecutionTime())
        {
          keyEvents[keyCounter] = e;
        }
      }
      sendData(true);
      keyCounter++;
    }
  }

  //Press F1 to schedule preindicated number of sessions
  if(e.getKeyCode() == KeyEvent.VK_F1)
  {
    int sessionsToPlan;

    if(training == true)
    {
      sessionsToPlan = 1;
    }
    else
    {
      sessionsToPlan = maxSession;
    }

    int i = 0;

    while(i < sessionsToPlan)
    {
```

```
        t.schedule((TimerTask) new SessionTask(this), (runTime*2*i + 4*i)*1000);
        i++;
      }
    }

  }

  //Empty - Irrelevant

  public void keyReleased(KeyEvent arg0) {

  }

  //Empty - Irrelevant

  public void keyTyped(KeyEvent arg0) {


  }

}
```

CLASS InteractionPane.java

```
import java.awt.Color;
import java.awt.Font;
import java.awt.GridLayout;

import javax.swing.JLabel;
import javax.swing.JPanel;

//Displays the reaction times and mistakes to the user.

public class InteractionPane extends JPanel
{
  private JLabel rt, averageRT, mistakes, averageMistakes,
          rtNr, averageRTNr, mistakesNr, averageMistakesNr;
  private GUI gui;

  //Constructor

  InteractionPane(GUI gui)
  {
    this.setLayout(new GridLayout(2,6, 0, 0));
    this.setBackground(Color.GRAY);

    this.gui = gui;

    Font big = new Font("SansSerif", Font.BOLD, 20);

    rt = new JLabel("RT Laatste Sessie: ");
    this.add(rt);

    mistakes = new JLabel("Fouten Laatste Sessie: ");
    this.add(mistakes);

    averageRT = new JLabel("Gem. RT laatste blok: ");
    this.add(averageRT);

    averageMistakes = new JLabel("Gem. Fouten laatste blok: ");
    this.add(averageMistakes);
```

```
    rtNr = new JLabel("0");
    rtNr.setFont(big);
    this.add(rtNr);

    mistakesNr = new JLabel("0");
    mistakesNr.setFont(big);
    this.add(mistakesNr);

    averageRTNr = new JLabel("Niet Berekend");
    averageRTNr.setFont(big);
    this.add(averageRTNr);

    averageMistakesNr = new JLabel("Niet Berekend");
    averageMistakesNr.setFont(big);
    this.add(averageMistakesNr);
  }

  //Sets the reaction time and displays it.

  public void setRT(String i)
  {
    rtNr.setText(i);
    rtNr.repaint();
  }

  //Sets the mistakes counter and displays it.

  public void setMistakes(String i)
  {
    mistakesNr.setText(i);
    mistakesNr.repaint();
  }

  //Sets the average reaction time and displays it.

  public void setAverageRT(String i)
  {
    averageRTNr.setText(i);
    averageRTNr.repaint();
  }

  //Sets the average mistakes counter and displays it.

  public void setAverageMistakes(String i)
  {
    averageMistakesNr.setText(i);
    averageMistakesNr.repaint();
  }
}
```

CLASS ReactionTimeTest

```
import java.io.IOException;


public class ReactionTimeTest
{
  //F1 to run
  //F5 to save
  //Press Y when color and content match
  //Press N when color and content do not match
  //Comment in or out to use the relevant code line.
```

```
  //This example is set to training mode.

  public static void main(String[] args)
  {

    //Experiment
    //GUI gui = new GUI(60, 800, 800, 20, false, "test");

    //Run from commandline
    //GUI gui = new GUI(Integer.parseInt(args[0]), Integer.parseInt(args[1]),
            Integer.parseInt(args[2]),
    //Integer.parseInt(args[3]), Boolean.parseBoolean(args[4]), args[5]);

    //Training
    GUI gui = new GUI(60, 800, 800, 5, true, "test");

  }
}
```

CLASS SessionTask

```
import java.awt.event.KeyEvent;
import java.util.TimerTask;

//Schedules a new session. This is used in test mode,
//where sessions are scheduled by the application.

public class SessionTask extends TimerTask
{
  GUI gui;

  //Constructor

  SessionTask(GUI gui)
  {
    this.gui = gui;
  }

  //Runs another session

  public void run()
  {

    gui.run(gui.runTime);
  }

}
```

CLASS UDPClient.java

```
import java.io.*;
import java.net.*;

//This class takes care of sending the relevant data to the UDP server

public class UDPClient
{
  int serverPort, clientPort, msgSize;
  byte msg[];

  DatagramSocket ds;
  DatagramPacket dp;
```

```
//Constructor

UDPClient(String logName) throws IOException
{
  serverPort=5011;
  clientPort=5012;
  msgSize=20;
  msg = new byte[msgSize];

  ds=new DatagramSocket(clientPort);

  setLog(logName);
}

//Sends a message containing the string "input" to the UDP server.

public void msgSend(String input)
{
  String temp = "data =" + input;
  try {
    msg = temp.getBytes("ASCII");
  } catch (UnsupportedEncodingException e1) {
    System.out.println("Could not convert message to ASCII Byte Array");
    e1.printStackTrace();
  }


  try {
    dp=new DatagramPacket(msg,msg.length,InetAddress.getByName("255.255.255.255"),serverPort);
    ds.send(dp);
  } catch (IOException e) {
    System.out.println("Failed to create Datagram Packet.");
    e.printStackTrace();
  }

}

//Sets the log name. A requirement of the UDP server of Matlab Simulink module.

private void setLog(String logName)
{
  String temp = "log =" + logName;

  try {
    msg = temp.getBytes("ASCII");
  } catch (UnsupportedEncodingException e1) {
    System.out.println("Could not convert message to ASCII Byte Array");
    e1.printStackTrace();
  }


  try {
    dp=new DatagramPacket(msg,msg.length,InetAddress.getLocalHost(),serverPort);
    ds.send(dp);
  } catch (IOException e) {
    System.out.println("Failed to create Datagram Packet.");
    e.printStackTrace();
  }
}
```

```
}
```

CLASS WhiteNoise.java

```
import java.io.File;
import java.io.IOException;

import javax.sound.sampled.*;

//Creates a sound clip of 10 seconds of white noise.

public class WhiteNoise
{

  Clip clip;

  //Constructor

  WhiteNoise()
  {
    try {
        // From file
        AudioInputStream stream = AudioSystem.getAudioInputStream(new File("White_Noise.wav"));


        // At present, ALAW and ULAW encodings must be converted
        // to PCM_SIGNED before it can be played
        AudioFormat format = stream.getFormat();
        if (format.getEncoding() != AudioFormat.Encoding.PCM_SIGNED) {
            format = new AudioFormat(
                    AudioFormat.Encoding.PCM_SIGNED,
                    format.getSampleRate(),
                    format.getSampleSizeInBits()*2,
                    format.getChannels(),
                    format.getFrameSize()*2,
                    format.getFrameRate(),
                    true);
            stream = AudioSystem.getAudioInputStream(format, stream);
        }

        // Create the clip
        DataLine.Info info = new DataLine.Info(
            Clip.class, stream.getFormat(), ((int)stream.getFrameLength()*format.getFrameSize()));
        clip = (Clip) AudioSystem.getLine(info);

        // This method does not return until the audio file is completely loaded
        clip.open(stream);

    }

    catch (IOException e) {}
    catch (LineUnavailableException e) {}
    catch (UnsupportedAudioFileException e) {}
  }

  //Repeats the sound clip as often as "numberOfPlays" input.
  //The White noise clip used in this experiment was 10 seconds long but was played for 60 seconds.
  //So it was on a 6 times loop.

  public void play(int numberOfPlays)
  {
     clip.loop(numberOfPlays-1);
```

```
  }

}
```

CLASS WNTask

```
import java.util.TimerTask;

//Creates a task that loops a 10 second clip of white noise for the indicated number of repetitions.

public class WNTask extends TimerTask
{
  WhiteNoise wn;
  int numberOfPlays;

  //Constructor

  WNTask(int numberOfPlays)
  {
    wn = new WhiteNoise();
    this.numberOfPlays = numberOfPlays;
  }

  //Runs the task

  public void run()
  {
    wn.play(numberOfPlays);

  }}
```

D Java Code: ANN

```
import org.encog.Encog;
import org.encog.engine.network.activation.ActivationSigmoid;
import org.encog.ml.data.MLData;
import org.encog.ml.data.MLDataPair;
import org.encog.ml.data.MLDataSet;
import org.encog.ml.data.basic.BasicMLDataSet;
import org.encog.neural.error.ATanErrorFunction;
import org.encog.neural.networks.BasicNetwork;
import org.encog.neural.networks.layers.BasicLayer;
import org.encog.neural.networks.training.propagation.back.Backpropagation;
import org.encog.neural.networks.training.propagation.resilient.ResilientPropagation;
import org.encog.util.csv.CSVFormat;
import org.encog.util.simple.TrainingSetUtil;


public class ANN {

  public static double PP1_FULL_INPUT[][] =
  {{0,0.6495,0.1753, 0,0.1396, 0, 0}
  ,{0.0857,0.5343,0.0639,0.0668,0.0851,0.0256,0.0615}
  ,{0.0143,1.0000,0.4725,0.0479,0.0280,0.1146,0.1350}
  ,{0.0571,0.5592,0.1155,0.0416,0.0096,0.0815,0.0890}
  ,{0.0857,0.4788,0,0.0631,0.0306,0.0088,0.0993}
  ,{0.0714,0.4225,0.5158,0.0279,0.0362,0.0666,0.0877}
  ,{0.0714,0.3256,0.0614,0.0335,0.0188,0.0755,0.1120}
  ,{0.0714,0.5809,0.0688,0.0575,0.0606,0.1470,0.1326}
  ,{0.0286,0.9783,0.4360,0.0569,0.0706,0.2563,0.2376}
  ,{0.0714,0.4631,0.0419,0.0491,0.0799,0.1253,0.1859}
  ,{0.0429,0.8392,0.2746,0.0579,0.0633,0.1575,0.1968}
  ,{0.0857,0.2884,0.2698,0.0362,0.0780,0.1411,0.1671}
  ,{0.0714,0.7252,0.0756,0.0683,0.0520,0.2648,0.3993}
  ,{0.0714,0.3876,0.1836,0.0384,0.0616,0.0957,0.2201}
  ,{0.0571,0.7504,0.3132,0.0603,0.0529,0.3529,0.3652}
  ,{0.0286,0.7471,0.0876,0.0306,0.0503,0.2262,0.3505}
  ,{0.0571,0.4979,0.0574,0.0365,0.0778,0.0749,0.2197}
  ,{0.0571,0.3164,0.0583,0.0207,0.0945,0.3461,0.4286}
  ,{0.0857,0.4394,0.2606,0.0596,0.1216,0.4885,0.6102}
  ,{0.1000,0.3715,0.1173,0.0645,0.1202,0.2568,0.3930}
  ,{1.0000,0.4391,1.0000,1.0000,1.0000,1.0000,1.0000}
  ,{0.1000,0.3120,0.1261,0.0536,0.0365,0.4054,0.4468}
  ,{0.0429,0.6996,0.0407,0.0388,0.0277,0.4847,0.6096}
  ,{0.0714,0.4828,0.0936,0.0517,0.0395,0.5135,0.5460}
  ,{0.0857,0.4771,0.0066,0.0598,0.0628,0.2831,0.3546}
  ,{0.0429,0.6758,0.1623,0.0396,0.0212,0.3026,0.4121}
  ,{0.0286,0.6472,0.0407,0.0227,0,0.2785,0.3968}
  ,{0.1000,0.2256,0.2132,0.0488,0.0126,0.3729,0.4840}
  ,{0.0286,0.8648,0.0720,0.0372,0.0210,0.3998,0.4978}
  ,{0.0714,0.4152,0.0330,0.0389,0.0447,0.4845,0.5779}
  ,{0.0857,0.1270,0.0594,0.0277,0.0387,0.4353,0.5072}
  ,{0.0714,0.2608,0.0463,0.0254,0.0500,0.4370,0.4892}
  ,{0.0571,0.4000,0.1198,0.0304,0.0507,0.3694,0.4929}
  ,{0.1000,0.2043,0.0433,0.0503,0.0614,0.3982,0.5375}
  ,{0.0857,0.2080,0.1505,0.0371,0.0374,0.6158,0.7644}
  ,{0.0857,0.1841,0.0253,0.0363,0.0219,0.6117,0.7246}
  ,{0.1000,0.2338,0.7966,0.0860,0.0220,0.4597,0.5391}
  ,{0.1286,0,0,0.0666,0.0559,0.0195,0.4561,0.5169}
  ,{0.1143,0.1625,0.2132,0.0596,0.0347,0.3463,0.4686}
  ,{0.0857,0.3828,0.2027,0.0450,0.0511,0.2891,0.3944}};
```

```
public static double PP2_FULL_INPUT[][] =
{{0.1429,0.6491,0.1355,0,0.9901,0.0959,0.2965}
,{0.4762,0.3937,0.3678,0.4342,0.4046,0.9418,1.0000}
,{0.3333,0.5946,0.6387,0.7696,0.4944,0.2419,0.1757}
,{ 0,0.9355,0.0421,0.1768,0.6508,0.1292,0.0975}
,{0.0952,0.7923,0.0462,0.1742,0.7941,0.0400,0.0586}
,{0.0476,1.0000,0.1002,0.6063,0.7759,0,0}
,{0.3333,0.5443,0.2098,0.3816,0.5405,0.2241,0.2281}
,{0.1905,0.6313,0.1426,0.0388,0.6697,0.2235,0.2254}
,{0.1905,0.7427,0.1300,0.4491,0.7845,0.1293,0.2275}
,{0.5714,0.3166,0.2593,0.4245,0.7289,0.7532,0.6479}
,{0.1429,0.7364,0.0124,0.1602,0.8141,0.4970,0.2979}
,{0.1429,0.8817,0.1371,0.6348,1.0000,0.1861,0.2223}
,{0.3333,0.4960,0.1237,0.2027,0.7553,0.1406,0.0749}
,{0.1429,0.7182,0.0263,0.1225,0.4868,0.2756,0.3020}
,{0.7619,0.1940,0.3101,0.7952,0.6925,0.5953,0.6275}
,{0.3810,0.6129,0.2985,1.0000,0,0.1450,0.1891}
,{0.4286,0.5472,0.2165,0.7550,0.4065,0.5829,0.6599}
,{0.1429,0.7260,0.1816,0.2003,0.5843,0.0486,0.1446}
,{0.0952,0.7738,0.2715,0.0591,0.3603,0.4006,0.4014}
,{0.5714,0.3142,0.1647,0.5728,0.5191,1.0000,0.8769}
,{0.4286,0.5126,0.2270,0.5965,0.6731,0.2494,0.1759}
,{0.3810,0.4480,0.1512,0.1984,0.4913,0.1161,0.2097}
,{0.2381,0.7625,0.1804,0.6770,0.6504,0.1819,0.2754}
,{0.2381,0.6005,0.3512,0.0743,0.6089,0.1830,0.1427}
,{0.3333,0.5375,0.1786,0.3774,0.6221,0.2247,0.2554}
,{0.7619,0.2265,0.2321,0.9342,0.5963,0.4937,0.2850}
,{0.3810,0.5833,0.2900,0.6388,0.6428,0.5886,0.3959}
,{0.2857,0.6803,0.2913,0.7368,0.2273,0.2936,0.1920}
,{0.1905,0.8522,0.3038,0.8245,0.5643,0.4607,0.1834}
,{0.6190,0.3016,0.4772,0.6061,0.7399,0.5791,0.4795}
,{0.7619,0.0805,1.0000,0.5555,0.6536,0.5480,0.4287}
,{0.2857,0.6357,0,0.5091,0.7714,0.1942,0.2694}
,{0.1429,0.8053,0.1277,0.3660,0.7840,0.1655,0.2381}
,{0.2857,0.6282,0.2948,0.4091,0.6938,0.2099,0.2604}
,{0.0952,0.8163,0.0613,0.1126,0.7512,0.2806,0.1838}
,{0.4762,0.4560,0.3763,0.5885,0.7465,0.2302,0.3146}
,{0.2857,0.7105,0.2375,0.7044,0.7830,0.4327,0.5723}
,{1.0000,0,0.2954,0.7452,0.5675,0.8841,0.8088}
,{0.2857,0.6349,0.1367,0.3133,0.5690,0.2809,0.2690}
,{0.8095,0.0586,0.6044,0.4854,0.5128,0.4953,0.5113}};

public static double PP3_FULL_INPUT[][] =
{{0,0.8228,0.1289,0,1.0000,0,0}
,{0.1333,1.0000,0.2141,0.7018,0.8329,0.1458,0.1275}
,{0.0667,0.8731,0.2632,0.2653,0.8286,0.0322,0.0278}
,{0.3333,0.5257,0.3603,0.2893,0.8173,0.5571,0.4177}
,{0.4000,0.6170,0.1770,0.7029,0.8723,0.1768,0.1731}
,{0.6000,0.2390,0.5770,0.3625,0.8395,0.5840,0.6353}
,{0.2667,0.6697,0.1271,0.4055,0.8413,0.3503,0.4948}
,{0.4000,0.4628,0.4171,0.2793,0.8067,0.6555,0.7441}
,{0.3333,0.7359,0.0034,0.7747,0.8402,0.4293,0.6055}
,{0.4667,0.3434,0.3763,0.2249,0.8363,0.5232,0.7588}
,{0.5333,0.3905,0.1981,0.6069,0.8705,0.7693,1.0000}
,{0.4667,0.4108,0.2444,0.4904,0.8805,0.3812,0.7856}
,{0.6000,0.2618,0.5175,0.4293,0,0.6290,0.7817}
,{0.3333,0.5512,0.3320,0.3792,0.8875,0.4361,0.6904}
,{0.6667,0.1530,0.8406,0.2995,0.8253,0.4733,0.3805}
```

```
,{0.3333,0.5572,0.1350,0.3959,0.8208,0.2081,0.3259}
,{0.5333,0.4627,0.2598,0.7508,0.8215,0.6137,0.5158}
,{0.9333,0.0201,0.3521,0.7606,0.8297,1.0000,0.9219}
,{0.4667,0.5844,0,0.8577,0.8307,0.2292,0.2521}
,{0.4000,0.3638,0.2328,0.1573,0.8107,0.4980,0.4425}
,{0.4000,0.6026,0.2317,0.6372,0.8305,0.3950,0.3319}
,{0.3333,0.7690,0.0142,0.8167,0.8659,0.1299,0.1545}
,{0.6000,0.3379,0.2999,0.6686,0.8599,0.5667,0.4451}
,{0.4000,0.3910,0.3363,0.1560,0.8087,0.7090,0.5735}
,{0.2000,0.9572,0.0820,0.8014,0.8534,0.4881,0.4952}
,{0.4000,0.5801,0.6053,0.6135,0.8524,0.7358,0.5840}
,{0.5333,0.2786,1.0000,0.2967,0.8352,0.4920,0.3353}
,{0.1333,0.8106,0.1795,0.3010,0.8365,0.2315,0.1206}
,{0.8000,0.2264,0.3218,1.0000,0.8706,0.5963,0.4053}
,{0.2667,0.6757,0.4515,0.4766,0.8577,0.2337,0.1313}
,{0.4000,0.3832,0.7723,0.1471,0.8424,0.6270,0.4794}
,{0.7333,0.0819,0.5422,0.3296,0.6701,0.9797,0.7631}
,{0.2667,0.6242,0.2118,0.4248,0.8646,0.5890,0.5003}
,{0.4000,0.4566,0.3038,0.3279,0.8817,0.4671,0.4466}
,{0.2667,0.6534,0.4685,0.4031,0.9055,0.3948,0.5429}
,{0.6000,0.2528,0.3401,0.3924,0.8572,0.5901,0.3754}
,{0.4667,0.3530,0.2943,0.2779,0.8617,0.7965,0.6431}
,{0.2667,0.5963,0.1672,0.2575,0.8836,0.3904,0.4788}
,{0.2667,0.7180,0.1945,0.5869,0.9127,0.3607,0.6232}
,{1.0000,0,0.4119,0.8605,0.9166,0.9136,0.8088}};

public static double PP4_FULL_INPUT[][] =
{{0,1.0000,0.5702,0,0.6673,0.0021,0.0006}
,{0.5294,0.5629,0.3411,0.8385,0,0.0132,0.0062}
,{0.4118,0.6578,0.2406,0.6624,0.4369,0.0070,0.0030}
,{0.3529,0.6810,0.0195,0.4748,0.5123,0,0}
,{0.4118,0.6745,0.2147,0.7052,0.5663,0.0066,0.0029}
,{0.4706,0.5931,0.0602,0.7119,0.6067,0.0041,0.0021}
,{0.4706,0.5681,0.2230,0.6103,0.6131,0.0164,0.0063}
,{0.5882,0.4210,0.0599,0.6781,0.6611,0.0020,0.0009}
,{0.5294,0.4939,0.1420,0.6679,0.6677,0.0110,0.0048}
,{0.4706,0.5363,0.3276,0.5184,0.2071,0.0085,0.0029}
,{0.4118,0.7861,0.0758,1.0000,0.6444,0.0093,0.0043}
,{0.4118,0.6393,0.0409,0.6309,0.6678,0.0020,0.0021}
,{0.7059,0.2980,0.3843,0.7345,0.4290,0.0095,0.0046}
,{0.8235,0.2153,0.0763,0.9162,0.9310,0.0045,0.0025}
,{0.6471,0.3123,0.1481,0.5992,0.9619,0.0042,0.0031}
,{0.7647,0.1335,0.1844,0.4634,0.9501,0.0143,0.0052}
,{0.7647,0.1793,0.3649,0.5633,0.9442,0.0136,0.0040}
,{0.8235,0.1620,0.2206,0.7640,0.9412,0.0011,0.0005}
,{0.9412,0.0878,0.2822,0.9204,0.9553,0.0048,0.0019}
,{0.9412,0.0499,0.2122,0.7967,0.9574,0.0022,0.0008}
,{0.9412,0.0628,0.1702,0.8180,0.5881,0.0078,0.0035}
,{0.7059,0.2727,0.2696,0.6099,0.6698,0.0137,0.0046}
,{1.0000,0.0218,0.3714,0.9434,0.6826,0.0096,0.0043}
,{0.7059,0.2742,0,0.7376,0.6734,0.0022,0.0017}
,{0.5882,0.4099,0.1920,0.6733,0.6943,0.0051,0.0025}
,{0.4706,0.4878,0.2101,0.5091,0.6863,0.0031,0.0023}
,{0.9412,0,0.1825,0.6128,0.5141,0.0082,0.0038}
,{0.7059,0.1414,1.0000,0.4661,0.6343,1.0000,1.0000}
,{0.5882,0.4342,0.1290,0.7191,0.3922,0.0070,0.0037}
,{0.3529,0.7756,0.1567,0.7975,0.6790,0.0056,0.0033}
,{0.5882,0.3019,0.4058,0.3300,0.4602,0.0057,0.0027}
,{0.5882,0.4614,0.1086,0.7717,0.9115,0.0048,0.0035}
,{0.6471,0.4311,0.1616,0.9169,0.9935,0.0041,0.0028}
,{0.7647,0.2772,0.1356,0.8659,1.0000,0.0086,0.0046}
```

```
,{0.5882,0.3046,0.2015,0.3218,0.9728,0.0081,0.0039}
,{0.7647,0.2125,0.2386,0.6633,0.5157,0.0108,0.0048}
,{0.5882,0.4560,0.0409,0.7655,0.6698,0.0067,0.0035}
,{0.3529,0.7696,0.2902,0.7408,0.2777,0.0100,0.0040}
,{0.2941,0.7214,0.0855,0.3793,0.7117,0.0049,0.0027}
,{0.2941,0.7027,0.0129,0.3266,0.7220,0.0051,0.0023}};

public static double PP5_FULL_INPUT[][] =
    {{0.2982,0.3005,0.1449,0.4195,0.1436,0.0044,0.0046}
    ,{0.2807,0.2992,0.1798,0.3625,0.0719,0.4685,0.4676}
    ,{0.5789,0.1255,0.1510,0.7902,0.0317,0.0195,0.0193}
    ,{0.6140,0.0985,0.0838,0.5022,0.0395,0.3331,0.3328}
    ,{0.7193,0.0589,0.0590,0.5434,0.0823,0.3125,0.3125}
    ,{0.6316,0.1004,0.0762,0.6541,0.1765,0.0143,0.0149}
    ,{0.5263,0.1490,0.0972,0.6305,0.1478,0.3480,0.3478}
    ,{0.4211,0.2054,0.1335,0.6012,0.1197,0.0806,0.0804}
    ,{0.7018,0.0646,0.1088,0.6201,0.0952,0.1133,0.1131}
    ,{0.7193,0.0662,0.0541,0.6639,0.1364,0.0065,0.0067}
    ,{0.2632,0.3164,0.2003,0.4337,0.1426,0.4751,0.4752}
    ,{0.6140,0.1024,0.0763,0.5315,0.1153,0.3587,0.3582}
    ,{0.4035,0.2002,0.2593,0.3355,0.1294,0.1526,0.1527}
    ,{0.3509,0.2605,0.1741,0.6812,0.1486,0.1288,0.1294}
    ,{0.1404,0.5238,0.4578,0.3034,0.5613,0.0005,0.0007}
    ,{0.4211,0.1837,0.1195,0.5837,0.1719,0.2438,0.2434}
    ,{0.5439,0.1344,0.0996,0.5541,0.1139,0,0.0008}
    ,{0.6842,0.0702,0.0957,0.7358,0.0604,0.3899,0.3905}
    ,{0.1579,0.4959,0.3584,0.6467,0.0265,0.6625,0.6624}
    ,{0.2982,0.3356,0.2214,0.5706,0.0470,0.2399,0.2399}
    ,{0.3509,0.2306,0.1823,0.5870,0.1357,0.5062,0.5056}
    ,{0.1930,0.4626,0.3452,0.5300,0.0993,0.0420,0.0418}
    ,{0.1579,0.2668,0.1998,0.0998,0.0038,0.7052,0.7039}
    ,{0,1.0000,1.0000,0,0.4186,0.3070,0.3077}
    ,{0.5614,0.1305,0.1174,0.7746,0.0395,0.1824,0.1813}
    ,{0.9649,0,0,0.5293,0.0532,0.3799,0.3799}
    ,{0.8772,0.0225,0.0501,0.7137,0.0094,0.2312,0.2302}
    ,{0.5789,0.1191,0.0767,0.5431,0.0267,0.7823,0.7818}
    ,{0.5088,0.1650,0.1309,0.6779,0.0166,0.1886,0.1874}
    ,{0.5614,0.1412,0.1674,0.8232,0.0258,0.1744,0.1747}
    ,{0.5439,0.1404,0.1607,0.8182,0.0838,1.0000,1.0000}
    ,{1.0000,0.0006,0.0032,0.6157,0.0284,0.7653,0.7653}
    ,{0.8421,0.0291,0.0217,0.5752,0.9669,0.0002,0.0002}
    ,{0.7544,0.0718,0.0851,0.8079,0.3781,0.2854,0.2850}
    ,{0.6842,0.0770,0.0774,0.6307,0.1066,0.0136,0.0129}
    ,{0.4386,0.1943,0.1417,0.6495,1.0000,0.0030,0.0015}
    ,{0.3860,0.2284,0.2133,0.7311,0.1382,0.2616,0.2618}
    ,{0.2456,0.3773,0.3495,0.7681,0.3309,0.0000,0}
    ,{0.2456,0.4037,0.3799,1.0000,0.4553,0.4078,0.4074}
    ,{0.3860,0.2444,0.1640,0.6059,0,0.4076,0.4062}};

public static double PP1_FULL_OUTPUT[][] = {{1.0000}, {0.5798},{0.5589},
      {0.6763},
      {0.6394},
      {0.4034},
      {0.5076},
      {0.5724},
      {0.4052},
      {0.5706},
      {0.4835},
      {0.6580},
      {0.5642},
      {0.8484},
```

```
      {0.4919},
      {0.3307},
      {0.3804},
      {0.6278},
      {0.4081},
      {0.4759},
      {0.4343},
      {0.4042},
      {0.6244},
      {0.3409},
      {0.1501},
      {0},
      {0.3069},
      {0.4246},
      {0.4324},
      {0.1484},
      {0.2297},
      {0.1860},
      {0.3384},
      {0.0647},
      {0.2272},
      {0.2052},
      {0.0425},
      {0.0061},
      {0.2466},
      {0.0164}};

public static double PP2_FULL_OUTPUT[][] =
  {{0.7050},
  {0.1277},
  {0.3625},
  {0.5466},
  {0.2543},
  {0.5525},
  {0.8412},
  {0.4827},
  {1.0000},
  {0.2797},
  {0.3534},
  {0.1751},
  {0.1705},
  {0.6686},
  {0.4973},
  {0.5445},
  {0.3175},
  {0.1580},
  {0.4092},
  {0.2685},
  {0.3499},
  {0.0666},
  {0.1336},
  {0.0145},
  {0.7200},
  {0.4871},
  {     0},
  {0.2291},
  {0.6172},
  {0.2570},
  {0.3600},
  {0.7040},
  {0.2475},
```

```
  {0.7070},
  {0.5322},
  {0.7109},
  {0.2640},
  {0.3222},
  {0.0817},
  {0.4300}};


public static double PP3_FULL_OUTPUT[][] =
      {{0.3552},
      {0.4028},
      {0.2142},
      {0.3751},
      {0.3455},
      {0.2099},
      {0.4477},
      {0.3025},
      {0.8811},
      {1.0000},
      {0.8601},
      {0.8731},
      {0.5350},
      {0.2632},
      {0.3560},
      {0.2598},
      {0.2147},
      {0.1688},
      {0.2607},
      {0.5052},
      {0.4097},
      {0.2762},
      {0.4072},
      {0.2807},
      {0.1542},
      {0.6566},
      {0.3816},
      {0.2716},
      {     0},
      {0.2801},
      {0.5435},
      {0.4434},
      {0.6837},
      {0.4534},
      {0.3468},
      {0.3812},
      {0.0736},
      {0.3363},
      {0.3018},
      {0.3631}};

public static double PP4_FULL_OUTPUT[][] =
      {{0.5022},
      {0.4766},
      {0.6544},
      {0.3196},
      {0.7208},
      {0.5128},
      {0.5204},
      {0.3407},
      {0.4210},
```

```
      {0.3198},
      {0.4693},
      {0.3688},
      {0.5078},
      {0.6734},
      {0.4049},
      {0.4423},
      {0.4786},
      {0.4313},
      {0.7118},
      {0.7449},
      {0.5468},
      {0.5996},
      {0.2305},
      {0.5646},
      {0.5090},
      {0.8403},
      {0.8499},
      {0.4010},
      {0.5291},
      {1.0000},
      {0.2911},
      {0.3902},
      {0.6215},
      {     0},
      {0.4407},
      {0.1665},
      {0.5611},
      {0.6270},
      {0.3501},
      {0.2248}};

public static double PP5_FULL_OUTPUT[][] =
      {{0.4984},
      {0.2659},
      {0.5046},
      {0.4400},
      {     0},
      {0.0234},
      {0.1558},
      {0.4742},
      {0.2212},
      {0.4612},
      {0.5313},
      {0.4334},
      {0.7195},
      {0.8013},
      {0.7746},
      {0.5743},
      {0.5184},
      {0.9062},
      {0.5999},
      {0.4700},
      {0.6074},
      {0.3627},
      {0.3098},
      {0.4868},
      {0.7295},
      {0.2082},
      {0.3055},
      {0.2755},
```

```
      {0.3052},
      {0.6414},
      {0.5686},
      {0.4233},
      {0.4011},
      {0.7314},
      {1.0000},
      {0.8007},
      {0.7810},
      {0.5567},
      {0.7089},
      {0.6008}};


/**
 * The main method.
 * @param args No arguments are used.
 */
public static void main(final String args[]) {

  int outliers[] = {20};
  //int outliers[] = {};
  int oNumber = outliers.length;
  int oCounter = 0;

  double pp1TrainingInput[][] = new double[39-oNumber][7];
  double pp1TrainingOutput[][] = new double[39-oNumber][1];

  double pp1TestInput[][] = new double[1][7];
  double pp1TestOutput[][] = new double[1][1];

  double predictions[][] = new double[40-oNumber][50];

  for(int j = 0; j<(40); j++)
    {

    if(oCounter<oNumber && j == outliers[oCounter])
    {
      oCounter++;
    }
    else
    {
      int oCounter2 = 0;

      for(int i = 0; i<(40);i++)
      {
        boolean outlierI = false;


        for(int s = 0; s<oNumber; s++)
        {
          if(i == outliers[s])
          {
            outlierI = true;
            oCounter2++;
          }
        }

        if(outlierI)
        {
          //do nothing
```

```
      }
      else if(i < j)
      {
        pp1TrainingInput[i-oCounter2] = PP1_FULL_INPUT[i];
        pp1TrainingOutput[i-oCounter2] = PP1_FULL_OUTPUT[i];
      }
      else if(i==j)
      {
        //do nothing
      }
      else if(i >j)
      {
        pp1TrainingInput[i-1-oCounter2] = PP1_FULL_INPUT[i];
        pp1TrainingOutput[i-1-oCounter2] = PP1_FULL_OUTPUT[i];
      }

    }

    pp1TestInput[0] = PP1_FULL_INPUT[j];
    pp1TestOutput[0] = PP1_FULL_OUTPUT[j];


    // create a neural network, without using a factory
    BasicNetwork network = new BasicNetwork();
    network.addLayer(new BasicLayer(null,true,7));
    network.addLayer(new BasicLayer(new ActivationSigmoid(),true,4));
    network.addLayer(new BasicLayer(new ActivationSigmoid(),false,1));
    network.getStructure().finalizeStructure();
    network.reset();

    BasicNetwork networkArray[] = new BasicNetwork[50];

    // create training data
    MLDataSet trainingSet = new BasicMLDataSet(pp1TrainingInput, pp1TrainingOutput);
    MLDataSet testSet = new BasicMLDataSet(pp1TestInput, pp1TestOutput);

    // train the neural network
    final ResilientPropagation train = new ResilientPropagation(network, trainingSet);


    for(int i = 0; i < 50; i++)
    {
      int epoch = 1;

      do {
        train.iteration();
          epoch++;
      } while(train.getError() > 0.01 && epoch <= 10000);


      // test the neural network
      for(MLDataPair pair: testSet ) {
        final MLData output = network.compute(pair.getInput());
        predictions[j-oCounter][i] = output.getData(0);
      }

      networkArray[i] = network;
      network.reset();
    }
  }
}
```

```
    // Print results

    for(int j = 0; j<(40-oNumber);j++)
    {
      for(int i=0;i<50;i++)
      {
        System.out.print(predictions[j][i] + ", ");
      }

      System.out.print("\n ");
    }

    Encog.getInstance().shutdown();
  }
}
```