3-1-2013

Automated sleep classification using the new sleep stage standards

Elizabeth Fehrmann

Automated Sleep Classification Using the New Sleep Stage Standards

by

Elizabeth Ann Fehrmann

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science
in Computer Engineering

Supervised by

Associate Professor Dr. Juan Carlos Cockburn
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
March 2013

Approved by:

Dr. Juan Carlos Cockburn, Associate Professor
Thesis Advisor, Department of Computer Engineering

Dr. Daniel B. Phillips, Associate Professor
Committee Member, Department of Biomedical Engineering

Dr. Roy Melton, Senior Lecturer
Committee Member, Department of Computer Engineering

Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title:

I, Elizabeth Ann Fehrmann, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

Elizabeth Ann Fehrmann

Date

Acknowledgments

I would like to thank my committee members—Dr. Cockburn, Dr. Phillips, and Dr. Melton—for their generous gifts of time, advice, and assistance. A special thank you to Dr. Cockburn for his patient and persistent guidance. I would also like to extend my appreciation to all the coworkers, friends, and family who all have been so supportive of me over the years. In particular, thanks to my mom and dad; I couldn't have done it without you.

Abstract

Automated Sleep Classification Using the New Sleep Stage Standards

Elizabeth Ann Fehrmann

Supervising Professor: Dr. Juan Carlos Cockburn

Sleep is fundamental for physical health and good quality of life, and clinicians and researchers have long debated how best to understand it. Manual approaches to sleep classification have been in use for over 40 years, and in 2007, the American Academy of Sleep Medicine (AASM) published a new sleep scoring manual. Over the years, many attempts have been made to introduce and validate machine learning and automated classification techniques in the sleep research field, with the goals of improving consistency and reliability.

This thesis explored and assessed the use of automated classification systems with the updated sleep stage definitions and scoring rules using neuro-fuzzy system (NFS) and support vector machine (SVM) methodology.

For both the NFS and SVM classification techniques, the overall percent correct was approximately 65%, with sensitivity and specificity rates around 80% and 95%, respectively. The overall Kappa scores, one means for evaluating system reliability, were approximately 0.57 for both the NFS and SVM, indicating moderate agreement that is not accidental. Stage 3 sleep was detected with an 87–89% success rate.

The results presented in this thesis show that the use of NFS and SVM methods for classifying sleep stages is possible using the new AASM guidelines. While the current work supports and confirms the use of these classification techniques within the research community, the results did not indicate a significant difference in the accuracy of either approach—nor a difference in one over the other. The results suggest that the important clinical stage 3 (slow wave sleep) can be accurately scored with these classifiers; however, the techniques used here would need more investigation and optimization prior to serious use in clinical applications.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Problem Statement

Sleep is fundamental for physical health and good quality of life, and clinicians and researchers have long debated how best to understand it. Sleep is a complex phenomenon [11], and research on sleep quality and quantification is ongoing.

One prevalent method used to support sleep research involves "sleep stage classification". For decades, the scoring of sleep stages has been considered essential for the understanding of sleep architecture, and this method continues to be used as a standard quantifier in sleep research [7, 12, 13]. Classic approaches to sleep stage classification involve sleep specialists (or sleep technicians) utilizing a manual technique of scoring. Over the years, however—and in response to the issues that result from using standard approaches—researchers have been looking for advances that might improve the efficiency, reliability, and accuracy of the classification process [14, 1].

To that end, the recording of physiological phenomena that occur during sleep made the transition from paper recordings to computer recordings, and it was only a matter of time before automated computer classification schemes were developed with the intention of assisting with the scoring and research of sleep staging [15]. In addition to the potential time savings, the possibility of reducing the variability in scoring outcomes led to multiple attempts to advance automated sleep stage classifiers.

The range of attempts at sleep scoring automation have exhibited some success [16]. For example, automated classification methods such as artificial neural networks, neuro-fuzzy systems, and support vector machines have been applied to the sleep scoring problem. Even with that success, though, recent reviews of these attempts have reaffirmed the need for continued work to improve

these automated classification schemes [17, 18].

This thesis explores the use of automated classification systems with the updated sleep stage definitions and scoring rules. In 2007, after years of research and discussion, new sleep guidelines were published by the American Academy of Sleep Medicine (AASM). The AASM guidelines were intended to replace the Rechtschaffen and Kales scoring system and are now required for AASM certification in sleep clinics [13]. Professional opinions of sleep staging are varied: some have challenged the value of the AASM approach [19], some have questioned sleep staging itself [20, 21], and some view the five-year old approach as a positive development that invites more research [22].

So, given the ongoing research in sleep classification, the following questions can be asked:

1. Can the new AASM staging definitions be used with an automated process to identify the stages of sleep correctly?

2. Do implementations that have proven effective in the past show improvements in classification when using the new AASM guidelines?

3. Is there a difference in performance between classifier implementations using the new AASM guidelines with respect to sleep features and scoring?

1.2 Thesis Objectives

For years, researchers have been attempting to create automated sleep stage classifiers that can perform as well as sleep technicians. These efforts have seen some success—using methods such as neuro-fuzzy systems (NFS) and support vector machines (SVM)—which suggests the value of the pursuit of automated sleep systems [23, 24]. However, as noted earlier, the classification standards have changed, and there have been few, if any, explorations of NFS and SVM methods in conjunction with the AASM guidelines.

Therefore, the first thesis objective was to automate sleep classification using the new sleep manual's stage definitions, using NFS methods, and to evaluate the performance of that automation. A second objective involved implementing and evaluating a second classifier using SVM methodology. The performance of the NFS and SVM classifiers was evaluated as follows.

First, in keeping with traditional approaches to the evaluation of automated sleep stage classification, the accuracy of these classifiers was assessed by calculating the percentage agreement between the automated classification results and the classification performed manually. Because the manual scoring of sleep stages is still used as a "golden standard," this calculated agreement provides a metric indicating whether classifiers using AASM guidelines can be developed to identify sleep stages correctly.

To offer a perspective of this type of performance evaluation, a recent study reported the percent agreement between manual and automated scoring (based on 12 studies of automation attempts) ranging from 50.9% to 89.1% [25]. The results for an automated classification system could be compared to the manual results with the same sleep data that is equal to or greater than the agreement result range of 50–89%. This kind of evaluation provides a metric to evaluate whether the AASM guidelines can be developed to identify sleep stages correctly.

In addition, on top of the basic percent agreement metric, recent discussion of the assessment of classifier performance emphasizes the use of Cohen's kappa [26], so kappa scores were also used as a metric for assessment of the SVM and NFS classifiers. The range of kappa (κ) scores associated with the agreement scores noted above is $\kappa = 0.55$ to 0.81, which indicates moderate to "almost perfect" agreement, respectively [27]. Unfortunately, the two top agreement scores above, 81% and 89% did not include kappa scores. The classification results of this thesis, though, were evaluated using kappa. Moreover, other metrics recommended for measuring classification include "specificity" and "sensitivity" [26], and these are included and discussed in the results section.

The preceding discussion indicates the central approach to evaluating the classifiers developed for this thesis. However, the accuracy of the NFS and SVM classifiers was also compared to the accuracy of other NFS and SVM sleep classifiers that had been created with the old R&K sleep stage standards to see if there were any improvements in classification.

Finally, in addition to the above, the classifiers developed in this thesis were also compared to each other. Researchers have noted that comparing the relative performance of classifiers using the same data set is rare [28]. Comparing different classifiers using a single data set can mitigate differences in the performance results due to differences in data sources, as opposed to differences in the effectiveness of the classifiers themselves. Therefore, with the use of a single data set for training and testing both classifiers, this work evaluates the "true" relative performance of the classifiers.

Chapter 2

Background Theory

2.1 Technical Background

The discussion in this section provides an overview of the technical language that forms the basis for addressing the thesis objectives stated in section 1.2.

2.1.1 Electroencephalogram and Polysomnography

An electroencephalogram (EEG) is a recording of the voltages generated by brain activity. Electrodes placed at predefined locations on a patient's scalp (see Figure 2.1) allow clinicians to measure brain activity by registering the potential differences between certain pairs of electrodes and plotting the signals—called electroencephalogram, or EEG signals—over time [29].

In the context of sleep staging, these EEG signal recordings are gathered, along with electrooculogram (EOG) and electromyogram (EMG) signals (in addition to electrocardiogram and respiration traces, which will not be used in this analysis), into a single record called a polysomnogram (PSG), shown in Figure 2.2 [30, 31, 32, 33]. EEG signals recorded for the purpose of sleep staging are a combination of different frequencies, ranging from 0.5 hertz (Hz) up to approximately 35 Hz, and typically have an amplitude on the order of microvolts (μV). This broad-spectrum signal is then broken into sub-bands, called delta (δ), theta (θ), alpha (α), and beta (β), and their frequency bands range from low to high, respectively. For further discussion of the frequency bands, refer to section 3.2. Similarly, EOG signals are recordings of the voltages generated by eye movements, and EMG signals are recordings of muscle movements. All of these recordings are instrumental in helping clinical sleep specialists (or sleep technicians) with sleep staging—identifying a patient's sleep stage based on a set of characteristics observed in the PSG signals.

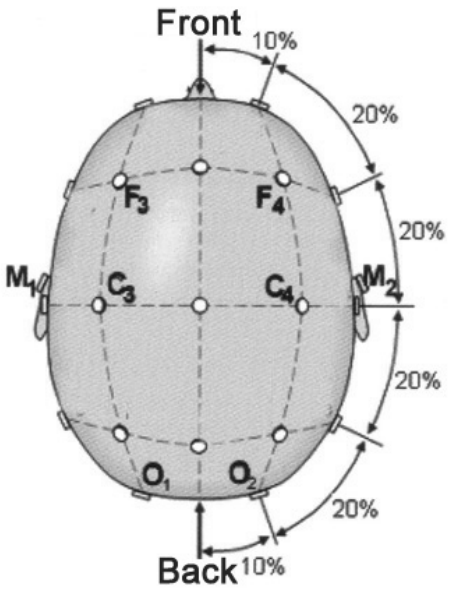

Figure 2.1: A diagram showing a top-down view of the 19 standard locations used for the 10-20 EEG electrode placement scheme. The labeled electrodes are as follows: F3 and F4 in the frontal lobe region, C3 and C4 along the centerline, and O1 and O2 in the occipital region, with electrode placements indicated by the white circles. Adapted from [1].

2.1.2 Sleep Staging

Sleep staging is very important, as it is typically used to assess and diagnose patients with sleeping disorders such as sleep apnea or hypersomnia, as well as in more generalized studies to learn more about the physiology of human sleep. The sleep stages of "normal" (non-disordered) adults typically cycle through a basic sequence over the course of the night, and that sequence is often referred to as "sleep architecture" [34, 35]. It has been shown that disruptions of one's sleep architecture can have a detrimental effect on cognitive functioning and that certain sleep disorders show distinct patterns that differ from the sleep architecture of the "normal" population [36, 37]. Therefore, to both aid in identification and diagnosis of disordered sleep as well as expand the understanding of regular sleep and its function, it is important that the outcome of these sleep studies be based on an accurate view of a patient's sleeping patterns.

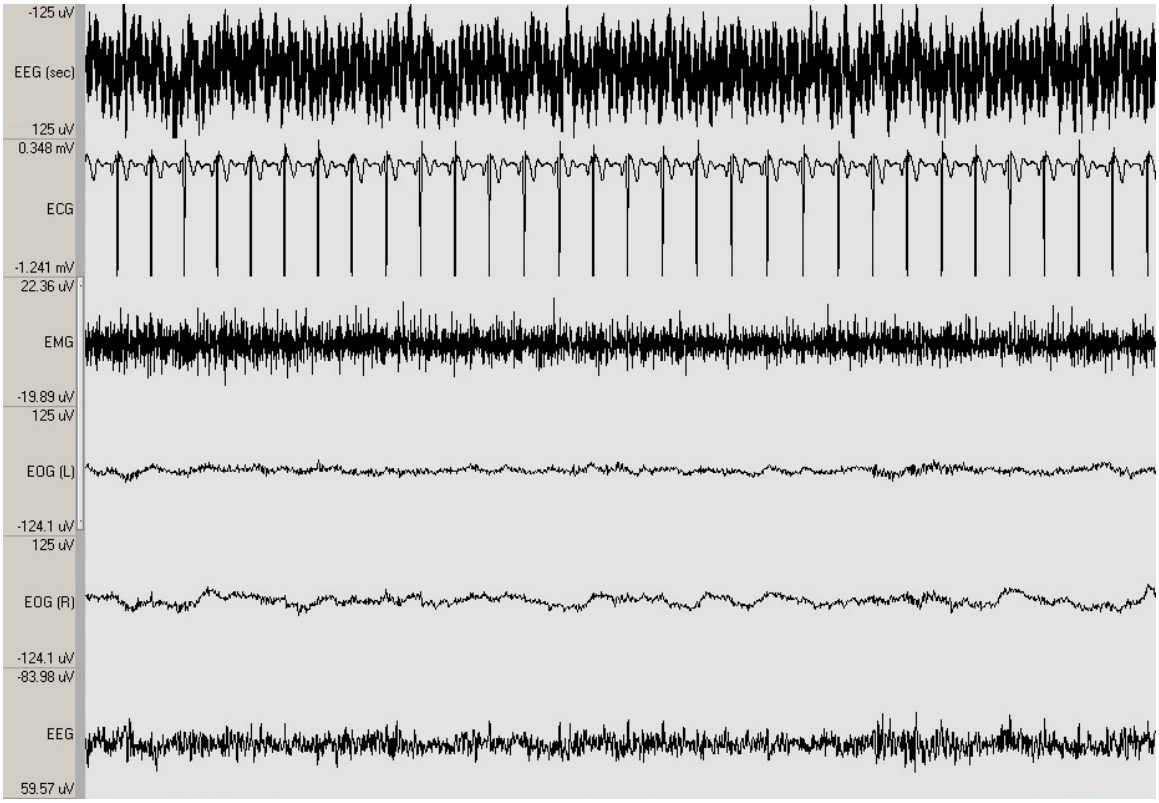

Figure 2.2: A sample PSG file with raw EEG, EOG, EMG, and ECG signals displayed over 30 seconds; channel labels and voltage ranges are shown to the left of the screen. This graph was generated by the freeware program "Polyman," which was designed as an European Data Format (EDF) viewer. The Polyman viewer program developed by Bob Kemp and Marco Roessen was downloaded from the "Download" section of the following website: http://www.edfplus.info/index.html.

R&K Methods

The dominant internationally accepted theory on sleep is that different levels and combinations of EEG wave patterns characterize the different stages of sleep. According to the standard sleep manual by editors Rechtschaffen and Kales (usually denoted "R&K"), *A Manual of Standardized Terminology, Techniques and Scoring System for Sleep Stages of Human Subjects*, there are six possible sleep stages: Wake (for periods when the subject is awake), Stage 1 (light sleep), Stage 2 (the most prevelant stage in normals), Stage 3, Stage 4 (both stage 3 and 4 are known as deep or slow-wave sleep stages), and rapid eye movement (abbreviated REM, this stage is also known as dream sleep or paralysis sleep) [7]. Each stage has its own set of identifiable traits (or features) that can be observed in an EEG recording.

The standard method by which sleep stages are classified in clinics is simply by hand. This manual approach is a very tedious process and requires the sleep lab technicians to stay awake all night to annotate the EEG recordings and write down the observed sleep stage every 30 seconds—or every "epoch." Understandably, this method has been critiqued as error-prone and achieving good inter-rater reliability has been challenging. One study of inter-rater reliability of technicians from different sleep centers reported an overall agreement rate of 76.8% [14]. This means that the same sleep study scored by two different sleep lab technicians would likely have—at least—a 20% difference in the scored stages. Even so, sleep researchers and clinicians have been using the R&K sleep manual as the "gold standard" for sleep staging since 1968—a little over forty years—without adapting the sleep stage definitions to technological advances and recent results in sleep research [12, 17].

AASM Methods

In June of 2007, after years of research and discussion, a new sleep stage scoring manual was released by the AASM, *The AASM Manual for the Scoring of Sleep and Associated Events*. Compiled by authors at the American Academy of Sleep Medicine, this manual is meant to replace the traditional 1968 manual by R&K and includes revised definitions of the sleep stages and new rules for sleep scoring [12]. These guidelines are now required for AASM certification in sleep clinics [13].

R&K Versus AASM

The first major difference between the old R&K method and the new method is the total number of sleep stages. Using the new method, there are only five sleep stages as opposed to six: Wake, Stage 1, Stage 2, Stage 3, and REM.

The sleep scoring is still to be performed on a 30-second interval epoch. The Wake stage is assigned when the EEG patterns indicate that greater than 50% of each epoch contains alpha activity—activity in the 8–13 Hz region—or when the existence of slow eye blinks or other irregular eye movements is detected. Stage 1 sleep is characterized by the disappearance of alpha rhythm and an increase in low-voltage, mixed-frequency activity (usually comprised predominantly of theta, or 4–7 Hz, activity, but can also include beta activity, or activity greater than 16 Hz) to occupy greater than 50% of the epoch. The low-voltage, mixed-frequency continues in Stage 2, but it is joined by

the regular appearance of sleep spindles and K-complexes (two distinctive EEG waveform features, see Figure 2.3). The new Stage 3 encompasses the definitions of the old Stage 3 and 4 sleep, which were defined by delta activity—activity in the 0–4 Hz region—in proportions of greater than 20% but less than 50% and greater than 50%, respectively. In accordance with the new manual, a subject has entered Stage 3 when more than 20% of an epoch contains slow-wave activity, thereby eliminating the need to look for a specific 20%-to-50% level of slow-wave content. The last possible stage for scoring is REM. The definition for this stage, like stage 1, is identical to its definition in the old R&K manual. The stage is labeled REM sleep if it is observed that more than 50% of each epoch is devoted to low-voltage, mixed-frequency activity—much like Stage 1 sleep—with the addition of detectable rapid movements of the eyes, thereby defining this sleep stage and giving rise to its name [7, 1].

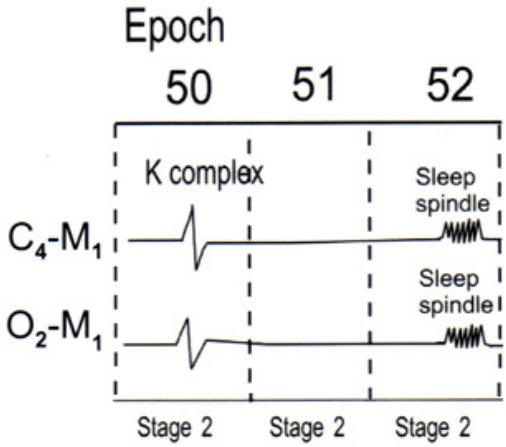

Figure 2.3: A graphical illustration—not to scale—of two defining characteristics of Stage 2 sleep: K-complexes and sleep spindles. Adapted from [1].

2.2 Classifiers and Automatic Sleep Staging

2.2.1 General Classification

Classification is the process of categorizing data into relevant groups. The first step in the classification process is identifying features or characteristics that will enable distinction between the different groups of data. Good feature selection is key to a good classifier; if the wrong features are

used for decisions, it is likely that the classifier's error rate—the rate of mis-classification—will be very high. To avoid this, the distinctive features of the data must be chosen with care to be easy to extract, insensitive to noise, as well as invariant with respect to transformations such as scaling and translation. Next, a classification model is developed that provides the structure for how the classification processes' actions will be realized. Ideally, this model should be chosen such that the performance of the system is optimized, although it may need to be revised as the classifier design progresses. A classifier is then implemented and "trained" to recognize the chosen features in the data, or to determine the best input-to-output mapping. There are many different ways to train classifiers, but all of them fall into one of two categories: supervised learning and unsupervised learning. A particular system is said to utilize supervised learning if the system uses a "knowledge expert" (or a teacher) to create an optimal response for the system, which is used as feedback to the learning system to increase accuracy; this method is the one that will be used in this work. In contrast, unsupervised learning occurs when the system does not use a teacher to modify its output. Once the system has "learned" to recognize and correctly classify specific inputs, the system is ready to be tested and evaluated with such metrics as speed of computation and accuracy of classification [3, 5]. See Figure 2.4 for a block diagram of the classification flow process.

One of the methods used for automating sleep stage classification is based on artificial neural networks (or ANN) [38, 28, 39, 40, 41]. These classifiers are occasionally combined with fuzzy logic decision rules to increase accuracy of classification [42, 43], and the resulting classifiers are called neuro-fuzzy systems (NFS). Yet another approach to sleep stage classification is the use of support vector machines (SVM) [44, 45, 24, 46]. Representative classification techniques are briefly described in sections 2.2.2 through 2.2.5.

2.2.2 Artificial Neural Networks

Artificial neural networks are classifiers modeled on the actual recognition and learning functionality of biological neural networks [3, 4]. A typical neural network consists of one input layer, one or more hidden processing layers, and one output layer. Figure 2.5 shows an example of an ANN.

The network can be regarded as an input-to-output mapping system: an input is applied to the input layer, then each hidden node combines the inputs using a transfer (or propagation) function and node weight, passing the input to other nodes for further processing (if the next layer is another

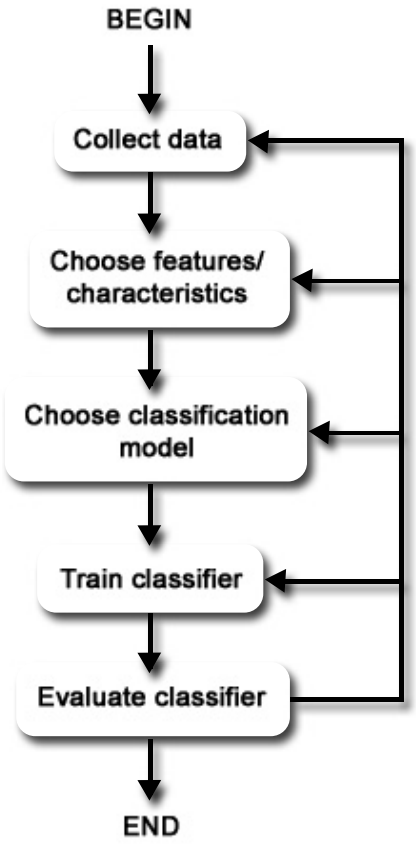

Figure 2.4: Action flow of a generic classifier design.

hidden layer) or to the output function for output realization. Figure 2.6 illustrates three popular propagation functions.

These ANNs are then trained—using either supervised or unsupervised learning processes—to produce a desired response upon recognition of a certain stimulus. Figure 2.7 illustrates a block diagram of a supervised learning network structure. To produce the desired result, a known training set—a set of data for which the classification is already known—is applied to the system and the weights are adjusted to minimize the error on the output side [3].

2.2.3 Fuzzy Logic

Fuzzy logic, in turn, is a way to describe a mathematical model that attempts to classify data correctly whose classification characteristics are vague or imprecise [5], or a way to deterministically

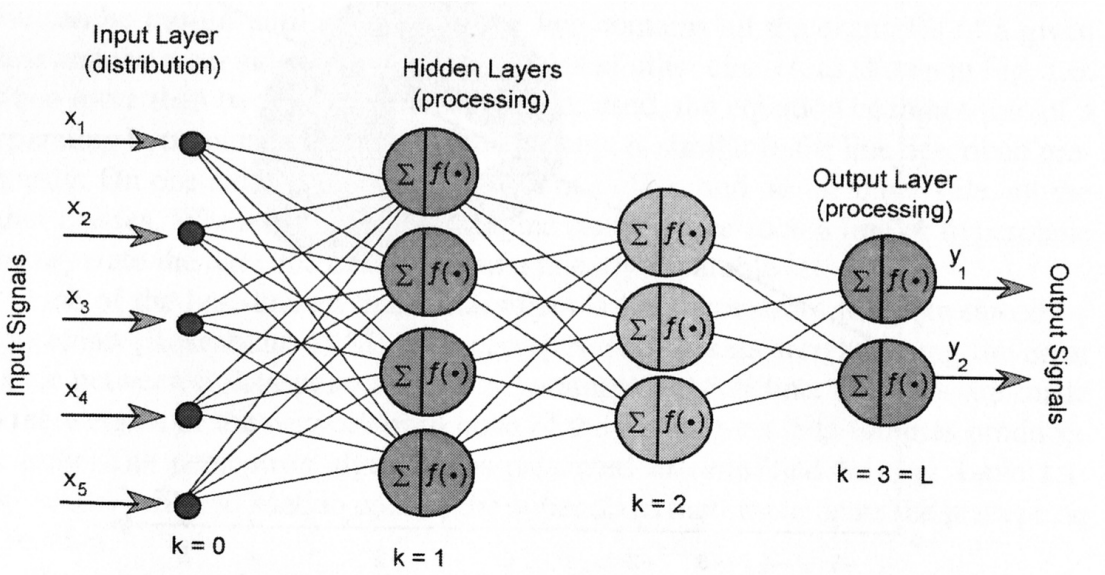

Figure 2.5: A sample artificial neural network, with inputs signals x_n and output signals y_n. Adapted from [3].

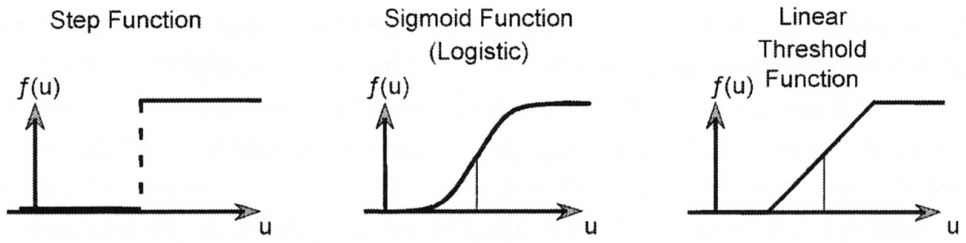

Figure 2.6: A selection of ANN propagation functions, from left to right: step, sigmoid, and linear transfer functions. Adapted from [3]. The sigmoid function is the most popular because of its general robustness and efficiency [4].

infer the outcome of a particular decision using approximate reasoning [47]. The function that defines a particular data point's inclusion in a set is called a membership function, and many such functions exist to approximate fuzzy sets as accurately as possible [48]. Figure 2.8 illustrates the difference between a binary, or "crisp" set and a fuzzy set. Given the varying degrees of membership of features in one group or another, fuzzy logic allows a feature to be defined as belonging to more than one class. The combination of this fuzzy logic with neural nets leads to neuro-fuzzy systems (NFS).

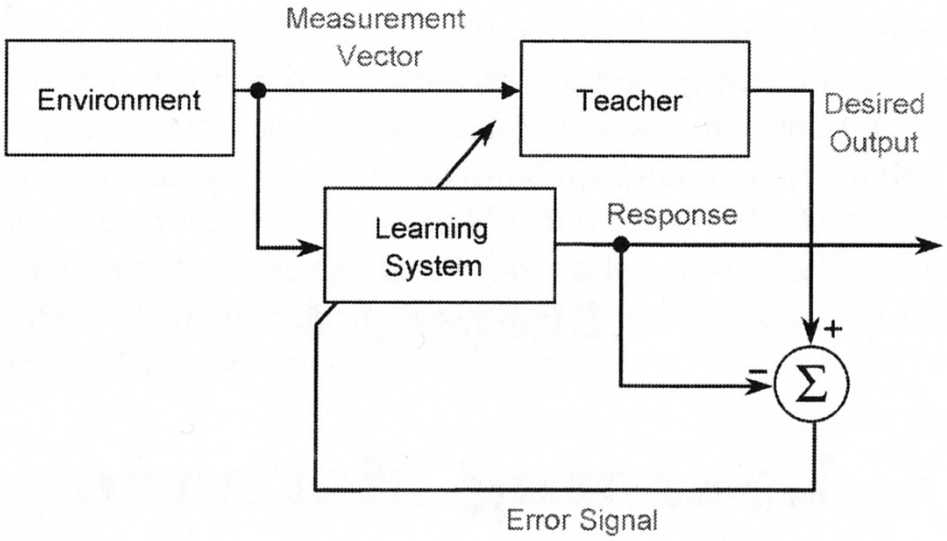

Figure 2.7: A block diagram of a supervised learning system; a neural network structure could be put in place as the "learning system", while a set of data with known outcomes could be used as the "teacher" to minimize the error in the network's response. Adapted from [3].

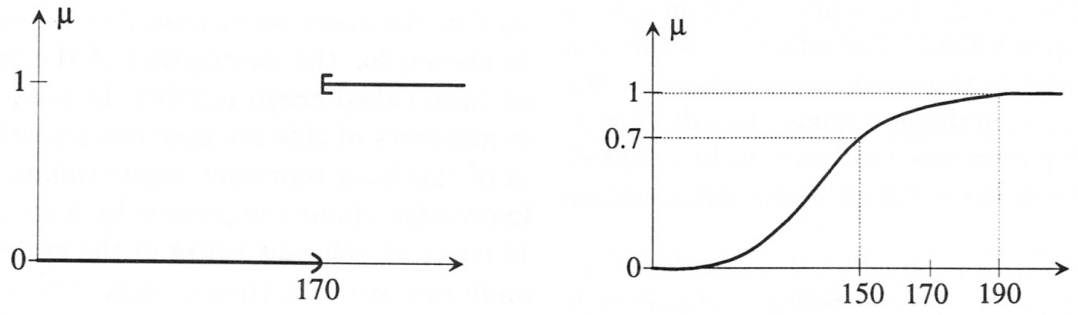

Figure 2.8: A comparison of a binary membership function (left) to a fuzzy membership function (right). If the characteristic under investigation is "tall", the function on the left will include only those data points greater than or equal to 170 centimeters, when 169 centimeters should perhaps be included. The function on the right, however, dictates degrees of membership, and therefore allows varying degrees of how "tall" the sample is. Adapted from [5].

2.2.4 Neuro-Fuzzy Systems

Fuzzy logic, by itself, is a good way to approximate a reasoning process, but it lacks the ability to optimize its membership functions and rules based on a given data set. Neural networks have

the ability to learn, but their decision capabilities are coded in the node weights and hard to interpret. Neuro-fuzzy systems combine the strengths of both, the fault-tolerance and adaptability of the ANN and the ability of fuzzy logic to use prior knowledge of the problem. This way, the NFS has increased learning functionality, able to "learn" (or optimize) fuzzy rules and membership functions that are specific to the data set of interest [47, 5]. That is, neuro-fuzzy systems do not combine ANNs and fuzzy logic, but instead use the ANN to create a fuzzy system via a heuristic learning method that leverages the learning procedures of the neural networks [49].

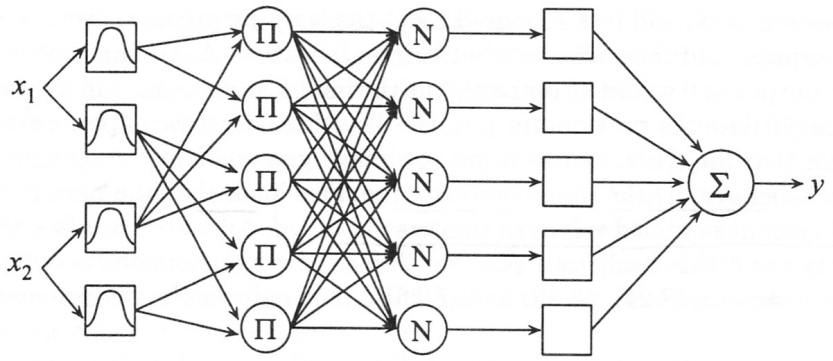

Figure 2.9: A generic diagrammatic implementation of one neuro-fuzzy-type system structure—called an Adaptive Network-based Fuzzy Inference System (ANFIS)—that has two inputs (x_1 and x_2), five intermediate processing layers, and one output (y). Adapted from [5].

The structure of an NFS is similar to a neural network structure: inputs are passed along and processed by one or more hidden layers that apply fuzzy logic-based transfer functions and weights (which can be modifying by the network's learning capabilities), and then combine the results into a single output decision. A sample diagram of the structure of these systems is provided in Figure 2.9.

Neuro-fuzzy methods can be applied if a fuzzy system is used for classification problems can be supported or replaced by an automatic learning process. This is meant to reduce the time and errors associated with manual modifications of the fuzzy decision logic, and to aid in the introduction of prior knowledge to and interpretation of the results produced by the neural networks. Systems called "Takagi-Sugeno" fuzzy systems are often used as a base for NFS solutions in function approximation problems [49]. Details of the system are described further in section 3.4.1.

In sleep-related studies that used NFS for sleep classification, researchers have been able to

achieve about 76–84% agreement rates [42, 50] when compared to sleep stage classification performed by human experts.

2.2.5 Support Vector Machines

There is a type of learning machine called a support vector machine (SVM) that classifies elements according to a scheme based on statistical learning theory in an effort to minimize the error and maximize the class separation interval [51, 6].

A support vector machine maps input vectors to a high-dimensional, linear feature space through a non-linear mapping, and then constructs an optimal separating hyperplane between the classes. With a particular data set made of features (known as the input space), it increases the dimensionality of the problem until it can find a hyperplane separation between the classes in the feature space [6]. A "maximum margin" hyperplane is created with all points corresponding to one class on one side of the plane and all points corresponding to the other class on the other side, such that the distance from the plane to the closest data points of both classes is at a maximum. Those points close to the plane become the support vectors (see the large "O's" and large "X's" in Figure 2.10) used to define the plane.

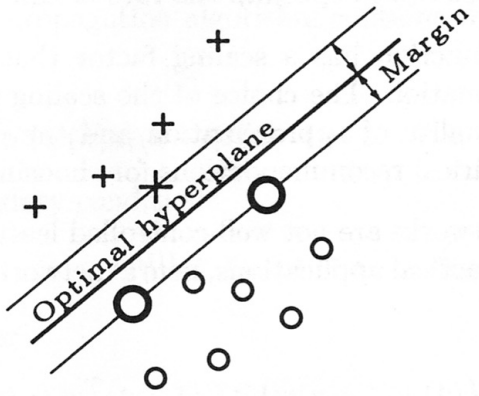

Figure 2.10: A sample hyperplane separating two data sets: the X's and the O's. The "maximum margin" hyperplane is defined using the data points closest to the plane, called the "support vectors" (indicated by the large X's and O's in the diagram), that maintain the maximum possible distance between those supporting points—or vectors—and the plane. Adapted from [6].

The maximal margin hyperplane is the simplest SVM, but works only on linearly separable

data, and therefore does not generalize well to non-linear or non-separable cases. If the data are not linearly separable, the optimization is modified to account for noise and outliers and allow erroneous classifications with a penalty. This type of SVM is known as a soft-margin SVM classifier [44, 52], and details of the system are described further in section 3.4.2.

In studies that looked at using SVM classifiers to score sleep automatically in humans, researchers have been able to achieve approximately 80–90% agreement rates [44, 45] when compared to a sleep lab technician's stage classification; 86–96% agreement has been seen in studies that looked at classifying sleep in rats [24, 46].

Clearly, with the updated sleep manual, the necessity arises for all of the previous automatic staging methods to be revised to conform to the new standards set forth by AASM, and this suggests the value of further research on staging efficacy utilizing NFS and SVM classifiers as automatic sleep staging systems.

Chapter 3

Data and Methods

In this section, the data collected and the methods used to process that data will be discussed. Construction of the classifiers will also be addressed.

3.1 Data Acquisition

This thesis used data gathered from the Sleep Heart Health Study Database[1]. The Sleep Heart Health Study (SHHS) was a prospective cohort study designed to assess the relationship between sleep-disordered breathing (SDB), such as obstructive sleep apnea (OSA), and the development of cardiovascular disease. Approximately 6,440 participants (aged 40 years and older with no history of OSA treatment) were enrolled and completed home-based sleep monitoring periods [53]. The resulting files were consolidated into the SHHS database and made available to the public for research purposes. Research criteria that were used to narrow the number of records for use in this work were the following:

- The overall study quality grade was equal to or greater than a score of "Excellent."
- The total time in bed was equal to or greater than 7 hours (420 minutes).

Of the resulting 571 available records that matched the given criteria, twenty-five subject data files were randomly chosen from the restricted pool for inclusion. The status of the subjects' disease states were ignored—even though sleep disorders can alter sleep architecture [37]—since the main

[1] "The findings in this report were based on publicly available data made available through the Sleep Heart Health Study (SHHS). However, the analyses and interpretation were not reviewed by members of the SHHS and does not reflect their approval for the accuracy of its contents or appropriateness of analyses or interpretation." (SHHS Data Distribution Agreement)

Table 3.1: A breakdown of the number of samples (and subsequent percentage) per sleep stage for the data used in this work. A "sample" indicates one epoch of time, described earlier in section 2.1.2.

Sleep Stage	*Number of Samples*	*Percent of Data*
Wake	9967	32.17%
Stage 1	1348	4.35%
Stage 2	12418	40.09%
Stage 3	2744	8.86%
REM	4496	14.51%

purpose of this work is identification of sleep patterns and stages in and of themselves and not the relation of disease states to sleep staging or sleep architecture. Table 3.1 provides a breakdown of the proportions of the 25 data files used by sleep stage.

3.2 Data Pre-Processing

Before the data could be classified by sleep stage, they required some pre-processing to clean the signals and extract relevant features for classification. The matrix manipulation software Matlab[2] was used for all the processing and classification development in this work.

3.2.1 AASM Recommendations

The guidelines that follow are taken from the 2007 AASM manual [1].

Data Sampling and Filtering

The AASM manual calls for the data to be sampled at a minimum rate of 200 Hz, with an ideal sampling rate of 500 Hz for the EEG, EOG, and EMG channels. The manual also recommends "routinely recorded filter settings" to provide some hardware filtering of the electrode signals to provide technicians with the ability to select important sleep characteristics visually. Those settings require that the EEG and EOG channels be bandpass filtered with cutoff frequencies of 0.3 Hz and 35 Hz, while the EMG channels should be filtered such that frequencies in the 10–100 Hz range are passed.

[2]Matlab® Student Version 7.7.0 (R2008b) was the version used and will be noted as "Matlab" in this document.

EEG

To support the new visual scoring rules presented in the AASM manual, it is recommended that the EEG electrode derivations used for staging be the following: F4-M1, the electrode on the right frontal lobe minus the reference electrode on the left ear (this reference electrode was traditionally referred to as A1 in the old R&K scoring manual, but it is now referred to as M1); C4-M1, the electrode to the right of the center ridge minus the left ear reference electrode; and O2-M1, the electrode on the right occipital lobe minus the left ear reference electrode. This recommended placement of electrodes ensures that the scoring algorithm will have the most overall coverage of the EEG activity in the frontal, center, and occipital regions. Refer to Figure 2.1 for the complete electrode layout.

EOG

The EOG channel derivations that are used are given as well, and they are as follows: E1-M2, the electrode placed one centimeter below the outer corner of the left eye minus the reference electrode on the right ear; and E2-M2, the electrode placed one centimeter above the outer corner of the right eye minus the right ear reference electrode. See Figure 3.1 for an EOG electrode placement diagram (labeled E1 and E2).

EMG

To record the muscle movements, a set of three electrodes is typically placed on and around the jawbone, one above and two below the inferior edge of the mandible. A single output is derived by taking the derivation between the electrode above and referencing it to one of the two electrodes below (one electrode is a backup in case the other malfunctions). Figure 3.1 illustrates the placement locations of these electrodes (labeled EMG), along with the reference electrode locations (M1 and M2).

3.2.2 SHHS Data Recording Procedures

The SHHS utilized data that were collected using an acquisition tool that had the capability of recording a full polysomnographic electrode montage at 500 Hz. The channels that were recorded

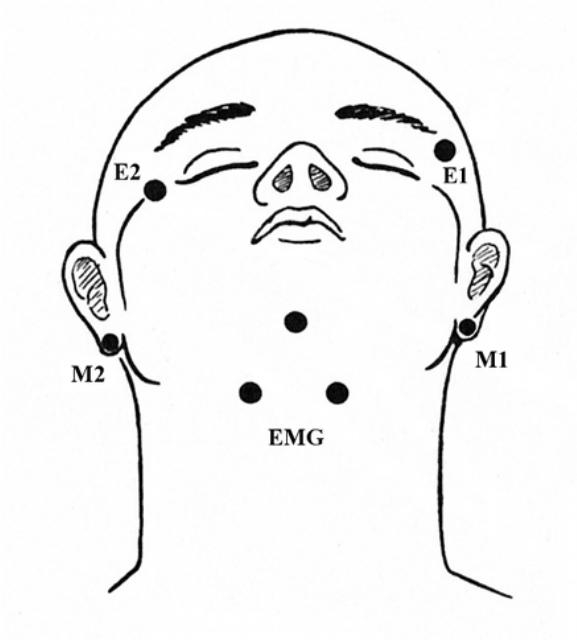

Figure 3.1: EMG, EOG, and reference electrode placement diagram; black dots indicate recommended electrode locations, which are the same for both the R&K and AASM guidelines. Adapted from [7] based on guidelines given by [1].

were C4-M1 and C3-M2 (EEG), standard EOG, and chin EMG, among others. The sleep data were then reviewed for quality and scored manually according to R&K standards by a central reading center in an attempt to provide consistency in sleep stage classification across all data sets [53].

The SHHS data used sampling rates of 125 Hz to collect the EEG and EMG signals and 50 Hz to collect the EOG signals. Additionally, only one of the three AASM-recommended EEG channel derivations (C4-M1) was collected. While these are not the recommended frequencies, nor the complete complement of recommended channels, a case can be made for the sufficiency of the information in the signals as recorded for the SHHS:

- Freely available, adequate data sets that follow all of the AASM recommendations for channel derivations and that contain sleep stages scorings according to AASM guidelines are not available at this time. Others who have published work on automation development during this transitional period have also cited the need for data [54].

- Recent studies have noted the explicit use of AASM scoring guidelines for their studies involving sleep using only the central (C4-M1) channel—or the central and occipital, with no

frontal channel—for recording EEG [28, 55, 56, 57].

- The AASM guidelines recommend the frontal (F4), central (C4), and occipital (O2) set of channels to maximize the chances for obtaining the necessary EEG information for certain stages. It does not state that there is no chance of seeing the same information for all stages if one uses only the central derivation [1].

- Other researchers have used the central channel to develop approaches to automated sleep stage classifications that they believe will be easily translated for use with the AASM guidelines when data collected according to the AASM standards are more readily available [54].

3.2.3 EDF to ASCII Data Conversion

Since the data processing and classification was performed in Matlab, the data needed to be formatted in such a way that it could be imported into Matlab. The SHHS data files are all in European Data Format (EDF) [58, 59]. Matlab does not have built-in functions to handle EDF data, so the files were converted to ASCII-formatted data files using a freeware program called the "EDF-to-ASCII Converter"[3]. The result of the conversion is two files: a ".ascii" file containing all the raw data samples for a particular electrode channel, and a ".txt" file containing information on the signal's source file, recording time, label, sampling frequency, and physical value conversion. All of the generated ".ascii" and ".txt" files belonging to a subject were assembled in a subject folder with a number designation (e.g. "01").

The resulting ASCII files, each containing a different channel of data, were then loaded into Matlab for processing. The ASCII file is imported into Matlab as "raw" data, which then must be converted into the physical values of micro-volts (μV). The ASCII-to-microvolt conversion factor was taken from the channel-specific ".txt" files, and the formulas for conversion are shown in equations (3.1), (3.2), and (3.3):

$$EEG(\mu V) = (\text{ASCII} + 32768) \times 0.00381475547417411 - 125 \tag{3.1}$$

$$EMG(\mu V) = (\text{ASCII} + 128) \times 0.247058823529412 - 31.5 \tag{3.2}$$

[3]The EDF-to-ASCII converter program developed by Bob Kemp and Marco Roessen was downloaded from the "Download" section of the following website: http://www.edfplus.info/index.html.

$$EOG(\mu V) = (\text{ASCII} + 128) \times 0.980392156862745 - 125 \tag{3.3}$$

3.2.4 Data Filtering

Once the data have been converted to real-world values, the next step in the process is to filter the data according to the AASM requirements—0.3–35 Hz for both the EEG and EOG signals, and 10–100 Hz for the EMG signal. These filter settings are given so that labs will have them in place while recording to capture the signal characteristics of interest, but it is also important that any noise introduced post-recording is filtered out using the same standards. Eighth-order elliptic filters with 0.1 dB pass-band ripple and an 80-dB stop-band were chosen for their relatively pass-band and quick roll-off at the cutoff frequencies, and implemented using the Matlab function `ellip`. Choosing the cutoff frequencies to limit the signals' frequency content, however, had to be performed with care, given the fact that the actual data acquisition frequencies (50 and 125 Hz) do not follow the recommended minimum 200 Hz collection rate.

The Matlab filter design tool adheres to rules that will not allow a filter to be built that is more than half of the collection frequency due to the following: generally, to have sufficient information at that highest necessary frequency, the sampling rate (f_s) of the acquisition must be at least two times that high cutoff due to the Nyquist frequency (f_N) rule (3.4):

$$f_s \text{ (Hz)} > f_N \tag{3.4}$$

The minimum sampling rate must satisfy the Nyquist Rate, so the sampling frequency must be greater than twice the highest frequency in the signal, as shown in (3.5) below:

$$f_N = 2 \times B, \tag{3.5}$$

where B, in this case, is the high-end cutoff of the bandpass filter. With that, then, the minimum required sampling rate for the EEG data channel would be $2 \times 35\,(\text{Hz}) = 70\,(\text{Hz})$, which the available 125 Hz fulfills, allowing the EEG band-pass filter to be designed as requried by the AASM manual. See Figure 3.2 for the frequency response of this filter.

The EMG and EOG channels, however, needed to be handled separately. Neither of them satisfy

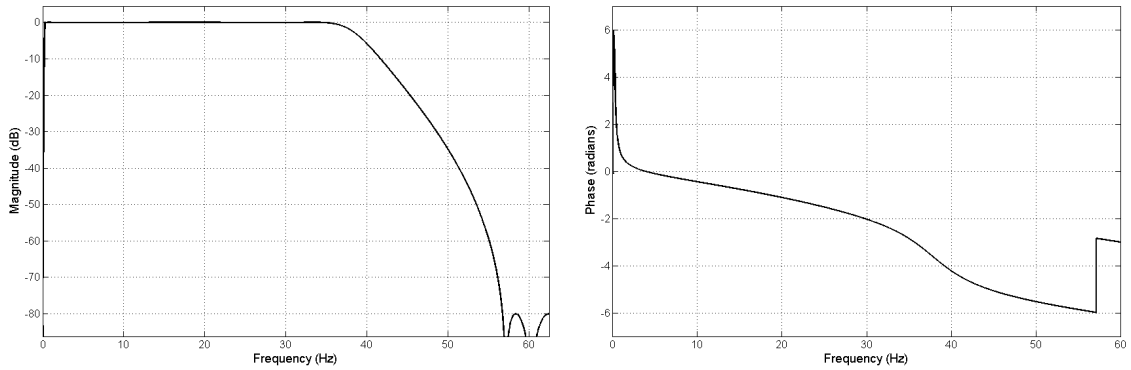

Figure 3.2: Filter used to extract the desired 0.3–35 Hz frequencies from the EEG signal as dictated by the AASM manual; the magnitude response is in the plot on the left, and the phase response in the plot on the right.

the Nyquist rule, as we would like to see a minimum sampling rate of 70 Hz on the EOG channel—where there is only a maximum f_s of 50 Hz—and a minimum sampling rate of 200 Hz on the EMG channel—where there is only a maximum f_s of 125 Hz. However, with careful inspection, the signals do still provide adequate signal information. The signal range needed for the EOG channel is 0.3–35 Hz (same frequency range as the EEG). Therefore, the 50 Hz sampling rate used to record the EOG is more than adequate to capture the signals of interest, and to achieve the recommended 0.3–35 Hz range, the signal was high-pass filtered at 0.3 Hz, making the final range 0.3–50 Hz. Similarly, the EMG signal, sampled initially at 125 Hz, is passed through a 10 Hz high-pass filter to achieve a signal range of 10–125 Hz, which is close to the recommended 10–100 Hz output.

The alternative to using only high-pass filters was to design band-pass filters using the maximum allowable filter cutoff frequencies, which for the EMG signal would have been 125 (Hz)$/2 = 62.5$ (Hz) and for the EOG signal would have been 50 (Hz)$/2 = 25$ (Hz). It was thought that this alternate course would take away too much signal information at the higher frequencies, and therefore the decision to proceed with only high-pass filters for the EMG and EOG signals was made. The filters are shown in Figure 3.3.

These filters were then applied to their respective matrices using the function `filter`, which creates a one-dimensional digital filter (implemented as a direct form II transposed structure). The

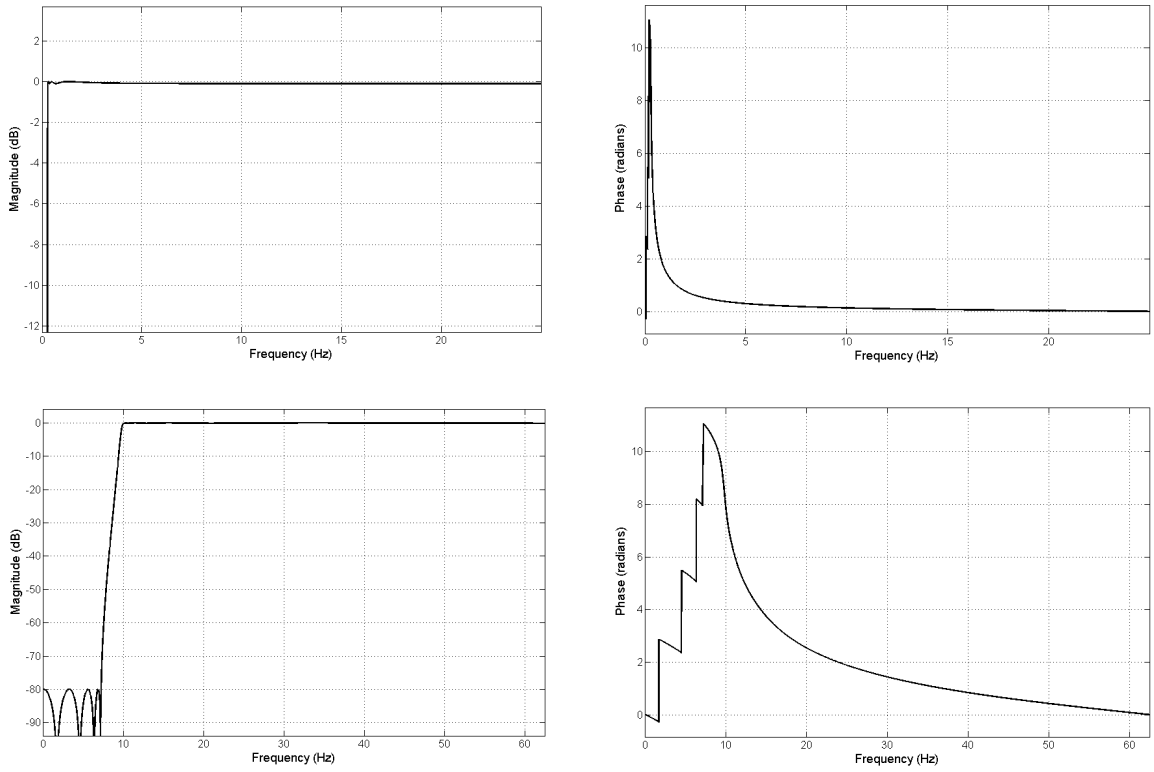

Figure 3.3: Filters designed for pre-processing the EOG and EMG data, with magnitude (on left) and phase response (on right). A 0.3 Hz high-pass filter was used for EOG and a 10 Hz high-pass filter was used for EMG.

difference equation implemented by the filter function is:

$$\begin{aligned} y(n) = b(1)x(n) \quad &+ \quad b(2)x(n-1) + \ldots + b(nb+1)x(n-nb) \\ &- \quad a(2)y(n-1) - \ldots - a(na+1)y(n-na), \end{aligned} \tag{3.6}$$

where $b(n)$ and $a(n)$ are the coefficients of the filter, $n-1$ is the filter order, x is the input vector to be filtered, and y is the output of the filter. The transfer function corresponding to that equation is:

$$Y(z) = \frac{b(1) + b(2)z^{-1} + ... + b(nb+1)z^{-nb}}{1 + a(2)z^{-1} + ... + a(na+a)z^{-na}} X(z) \tag{3.7}$$

The `filter` function takes three parameters: a vector of the filter coefficients from the numerator (the $b(n)$ coefficients), a vector of the filter coefficients from the denominator (the $a(n)$

coefficients), and the input vector to be filtered (the EEG, EOG, and EMG data).

3.2.5 Epoch Breakdown

After filtering, or "pre-conditioning," the data channels were segmented into epochs—independent, consecutive, 30-second-long sections of data containing each of the four channels—for scoring purposes, as dictated by the AASM manual. The number of samples in each epoch was determined by the sampling rate. Since the sampling rate was 125 Hz for the EEG and EMG channels, and 50 Hz for the EOG channels, the number of samples per epoch was $30 \text{ seconds} \times 125 \text{ samples/second} = 3750$ samples and $30 \text{ seconds} \times 50 \text{ samples/second} = 1500$ samples for the channels, respectively. Data structures were created with the following fields (and number of samples):

- "eegc4": the C4-M1 EEG channel data (3750 samples)
- "eogl": the EOG channel data from the left eye (1500 samples)
- "eogr": the EOG channel data from the right eye (1500 samples)
- "emg": the EMG channel data (3750 samples)
- "subj_id": an identifier for the data set, labeled by the folder from which the raw data came

The number of epochs per subject (N) varied depending on how long the subject's recording lasted, and the first and last epochs of data for each set were thrown out due to the minor distortions caused by the filtering process used per the discussion in section 3.2.4.

3.2.6 Sleep Staging

In addition to the EEG, EOG, and EMG channel information, each subject's data set had a file containing a professional's sleep stage scoring annotations associated with the recording. These text files contained the sleep stage codes that are used to identify the stages: "0" for Wake, "1" for Stage 1 sleep, "2" for Stage 2 sleep, "3" for Stage 3 sleep, and "5" for REM. There were other codes present in these vectors, such as "4" and "9", with the inclusion of stage 4 due to the fact that the SHHS study used the R&K rules for scoring. To create sleep stage annotations that focus on scoring only genuine sleep according to the AASM standards, all of the Stage 4 codes found in the sleep

stage code vectors were changed to "3" since there is no distinction of Stage 4 from Stage 3 in the AASM manual [1]. All other codes found were assumed to indicate events like movement time, were given a value of "-1" as a marker, and were ignored in later processing steps. This modified vector, then, became a sixth field in the subject's data structure (as described in the previous section, 3.2.5) with the label "sleep_stages."

3.2.7 Preprocessing Summary

Finally, the preprocessing routine steps are summarized in the following pseudo-code:

Algorithm 1 Raw sleep data preprocessing

for all patients **do**
  importdata(EEG, EMG, EOG)
  $f_s \leftarrow$ sampling frequency
  convert ASCII data $\rightarrow \mu$V
  filtered data $\leftarrow$ filter(EEG @ 0.3–35 Hz, EMG @ 10 Hz, EOG @ 0.3 Hz)
  $sleep\_stages \leftarrow$ sleep technician scores
  number of epochs $= study\_length$ (secs) $\div$ 30 seconds/epoch
  samples per epoch $= f_s \times 30$
  for $i = 1$ to number of epochs **do**
    get current epoch using samples per epoch and i
    [EEG_{C3}, EEG_{C4}, EMG, EOG_L, EOG_R, $sleep\_stages$] $\leftarrow$ filtered data(current epoch)
  end for
end for

3.3 Feature Selection and Extraction

Once all of the patients' data files had been processed as detailed in the previous section (3.2), specific sleep stage characteristics—or features—needed to be extracted from the epochs. The features that were extracted were chosen to reflect the visual staging rules that are detailed by the AASM sleep manual as much as possible, at least to the extent that an automated system is able to identify certain characteristics. For example, stage "Wake" has the following rule: "more than 50% of the epoch must display alpha rhythm in the occipital region for the epoch to be scored as 'W.'" Since there exist quite a few methods to extract the alpha-band frequency activity successfully (8–13 Hz) from the background EEG signal (such as spectral analysis or digital filtering [2]), it is

Table 3.2: Stage characteristics as defined by AASM and features extracted for this program [1].

Sleep Stage	*Visual Rule*	*Features Extracted Relevant to Stage*
Wake	50% or more of epoch dedicated to alpha rhythm	Energy in alpha (8–16 Hz) frequency band
Stage 1	Attenuated alpha rhythm and 50% or more of epoch dedicated to low amplitude, mixed frequency activity	Energy in theta (4–8 Hz) and beta (>16 Hz) frequency bands
Stage 2	The presence of K-complexes (unassociated with arousals) and sleep spindles with a continuation of low amplitude, mixed frequency activity	K-complexes and sleep spindles
Stage 3	20% or more of epoch dedicated to slow wave activity	Energy in delta (0.3–4 Hz) frequency band
REM	All of the following present: low amplitude, mixed frequency EEG, low chin EMG tone, and rapid eye movements	Eye movements, tonic and phasic EMG

acceptable for automated systems to evaluate the alpha-band frequency based on characteristics such as mean amplitude, mean frequency, and energy ratios [2, 60].

Table 3.2 summarizes the stages and stage characteristics (or visual rules) as defined by the AASM sleep manual, and it also shows the characteristics that were chosen to be extracted for each stage. It should be noted that the rules given are generalized rules and will not necessarily apply to all sleep samples, so only the extraction of the most generalized features that should be present for most all subjects (regardless of varying factors such as age) and features that are unique for each stage (such as extracting sleep spindles to help define Stage 2) are pursued.

The means by which researchers go about handling the extraction process varies greatly. Notable stage features and their extraction methods are shown in Table 3.3. Of the features chosen for extraction, four of them deal with energy in the various frequency bands. To get an accurate picture of this, then, wavelet decomposition was determined to offer the best separation mechanism between bands of different frequencies in biological signals [61, 43, 4, 41].

Table 3.3: Listing of possible methods for sleep feature extraction, first two columns are adapted from [2].

Sleep Pattern	*Possible Methods*	*Chosen Method*
Alpha (α) waves	Period-amplitude analysis, spectral analysis, digital band-pass filter	Wavelet decomposition
Delta (δ) waves	Period-amplitude analysis, spectral analysis, digital band-pass filter, model-based detector	Wavelet decomposition
Sleep Spindles	Digital filters, neural networks, matched filtering, wavelets	Match filter
K-complexes	Neural networks, matched filtering, wavelets, model-based detector	Match filter
Eye movements	Matched filtering, rapidly adapting neuronal fuzzy system, discrete wavelet transform, period-amplitude analysis	Match filter
Tonic muscle movements	Trimmed root-mean-squared amplitude	RMS amplitude
Phasic muscle movements	Maximal peak-to-peak amplitude	Max peak-to-peak amplitude
Artifact (removal)	Digital filtering, autoregressive-modeled filter smoothers, adaptive noise cancellers, independent component analysis	Digital band-pass filter

3.3.1 Wavelet Decomposition for Feature Extraction

For effective, adaptive analysis of the frequency components of biological signals, wavelet decomposition is often used. Signals such as EEG are non-stationary, dynamic, and often noisy, which can make them difficult to analyze using methods like Fourier transforms. The discrete wavelet transform (DWT) is a type of decomposition in which the input signals are split in two and the lower frequencies are filtered and decimated repeatedly as needed until the desired frequency ranges are extracted [41, 4]. This is illustrated in the diagram in Figure 3.4 below.

The wavelet decomposition function in Matlab, called `wavedec`, takes information about the number of desired levels of decomposition and the wavelet type and outputs the wavelet decomposition vector containing the sub-band signals. These sub-bands can then be analyzed separately as needed. Matlab has many wavelet types available, but researchers have suggested that the Daubechies second-order wavelet is a good choice [41, 40]. The scaling function—$\phi(t)$—and wavelet—$\psi(t)$—for second-order Daubechies wavelets are shown in Figure 3.5.

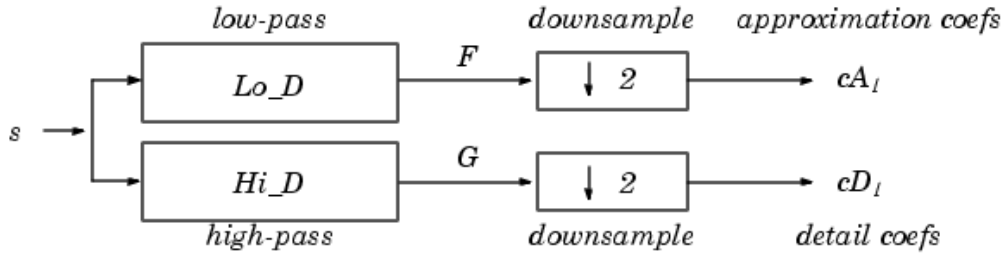

Figure 3.4: An illustration of how wavelet decomposition splits, filters, and decimates the signal into two sub-bands: the lower half of the frequency band in one output, and the upper frequency band in the other. Adapted from [8].

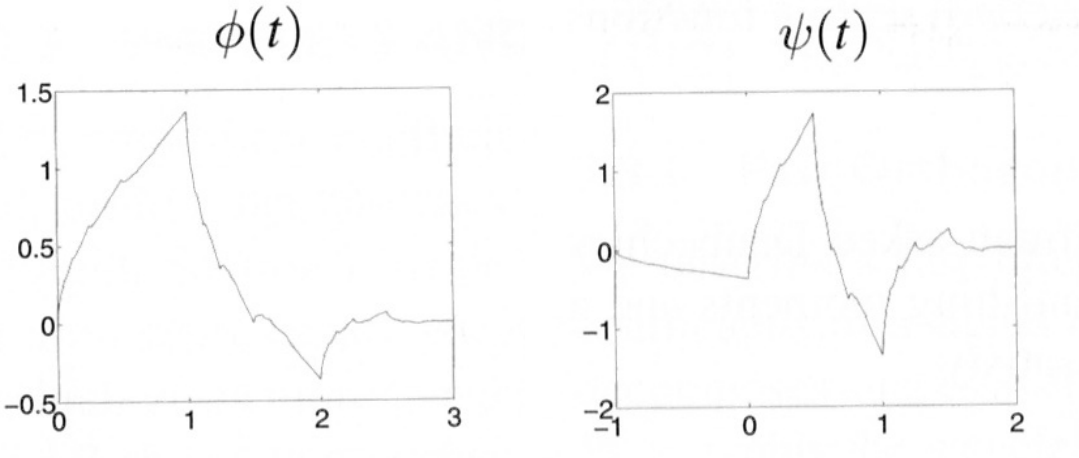

Figure 3.5: The scaling (left) and wavelet (right) functions associated with the second-order Daubechies wavelets. Adapted from [9].

3.3.2 Wake Stage Features

For the first stage, Wake, the manual indicates that one should look for more than 50% of the epoch to be alpha activity, so the feature chosen for extraction to represent the Wake stage was the energy in the alpha (8-16 Hz) band. The alpha frequency band was extracted from the rest of the background EEG using wavelet decomposition.

The "energy" of the signal is then computed using the Matlab function `wenergy`, which computes the percentage of energy in the signal sub-band of interest. To do this, it first computes the 2-norm, or Euclidean norm, of the wavelet coefficients and of the entire signal. Equation (3.8)

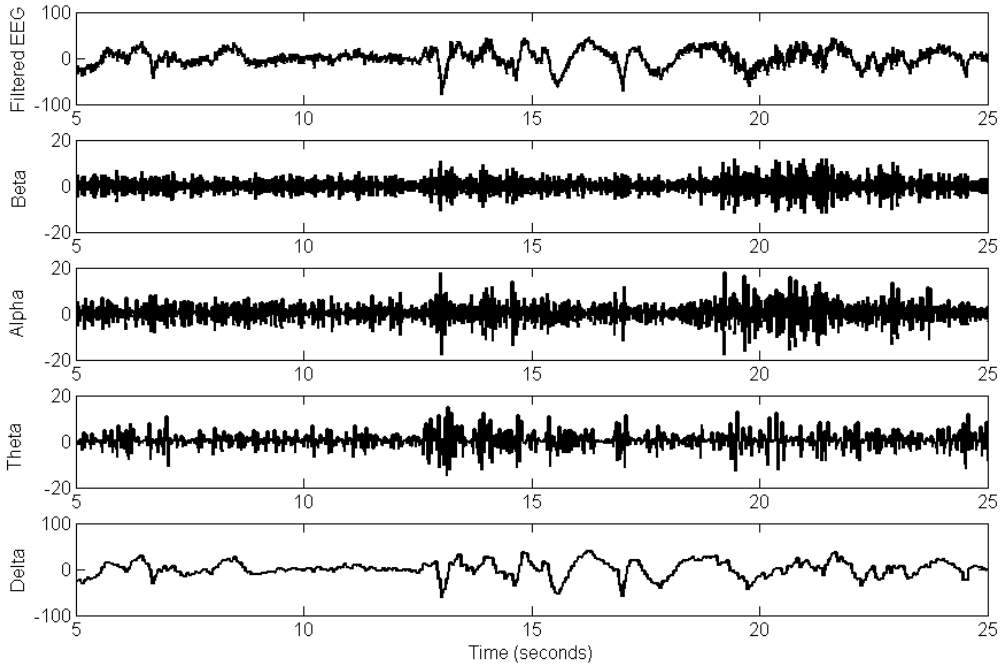

Figure 3.6: A sample of the output of wavelet decomposition performed on an epoch of sleep marked "Wake". The original signal is shown in the top plot, and then the decomposed signals in each of the four sub-bands of interest: β, α, θ, and δ (from the second plot down). The units on the y-axis are μV. Note the relatively large amplitude of α content and the presence of sharp, irregular eye movements that appear on the δ band.

illustrates the calculation for the 2-norm.

$$\|x\|_2 = \sqrt{x_1^2 + \ldots + x_n^2} = \sqrt{\sum_{n=1}^{N} x_n^2} \tag{3.8}$$

Then, the "relative energy"—or the ratio of energy in particular frequency band to the overall energy of the signal—is computed using the following equation (3.9):

$$\text{Relative Energy} = 100 \times \frac{\|x_{wave\_coeffs}\|_2^2}{\|x_{signal}\|_2^2} \tag{3.9}$$

3.3.3 Stage 1 Features

For the identification of Stage 1, the AASM definition states that "alpha rhythm is attenuated and replaced by low amplitude, mixed frequency activity for more than 50% of the epoch" [1], and

from that, it was determined that the unique stage feature to be extracted was the presence of "low amplitude, mixed frequency" activity, and typically includes activity in the theta (4–8 Hz) and beta (>16 Hz) frequency bands; therefore, the feature selected for Stage 1 was the energy in the theta and beta frequency bands. First, those bands were extracted from the background signal using wavelet decomposition (see Figure 3.7 for a sample Stage 1 decomposition), and then the `wenergy` function was used to compute the relative energy in those portions of the signal.

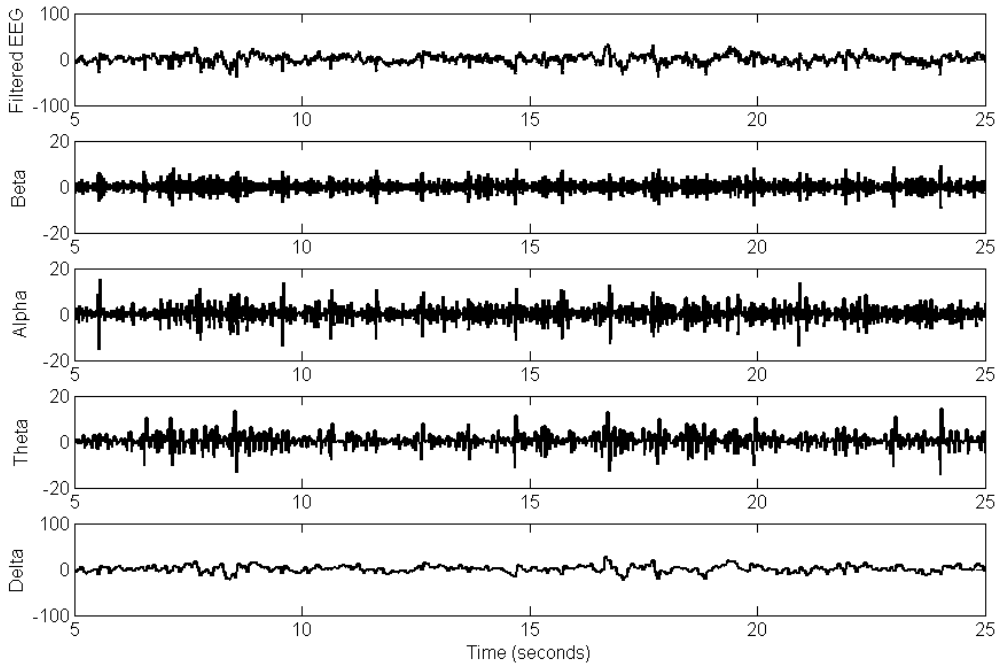

Figure 3.7: A sample of the output of wavelet decomposition performed on an epoch of Stage 1 sleep. The original signal is shown in the top plot, and then the decomposed signals in each of the four sub-bands of interest: β, α, θ, and δ (from the second plot down). The units on the y-axis are μV. Note the disappearance of the peaked eye movements and the reduction in α band activity (compared to 3.6).

3.3.4 Stage 2 Features

Stage 2 has what are probably the two most visually discernible characteristics of sleep: K-complexes and sleep spindles. Predictably, the AASM sleep manual states that one should begin scoring an epoch as stage two when either of these characteristics appear, unassociated with an arousal (an abrupt shift in frequency). The features that were extracted then for identification of Stage 2 sleep

were both K-complexes and sleep spindles. Since both of these characteristics have a well-defined shape and approximate standard frequency they can be effectively extracted using match filters. The concept of match filtering is as follows: a template pattern is defined to "match" the signal pattern hidden in the background EEG, then the template is compared to the EEG signal using convolution or correlation, and "matches" between the signal and the template can be quantified. Due to the non-symmetric nature of the K-complex, the template was compared to the EEG signal using correlation. Sleep spindles, however, are symmetric, and could be located using either method of comparison. For this work, correlation was the chosen method. Figure 3.8 illustrates the match filters used in this step.

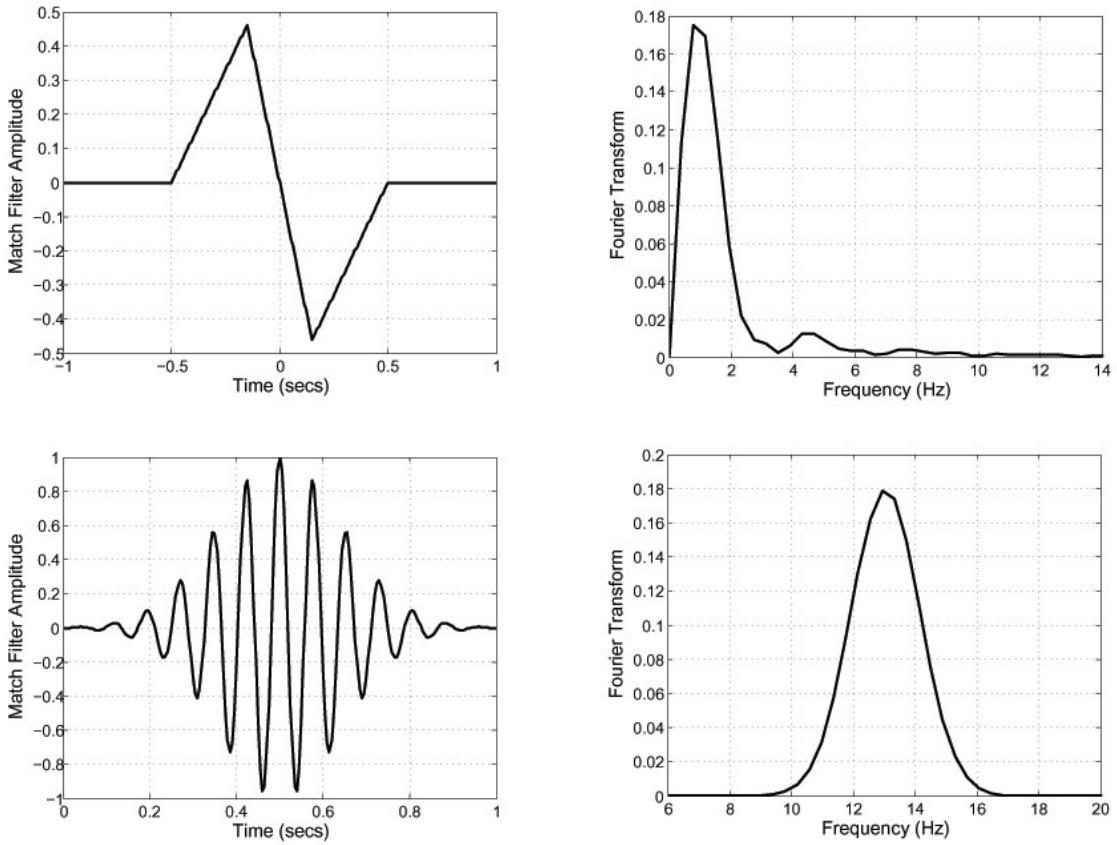

Figure 3.8: Match filters used to extract K-complexes (top) and sleep spindles (bottom) for idendification of Stage 2 sleep. Filter magnitudes are shown on the left, with their corresponding frequency responses on the right.

Since the duration of sleep spindles—to be considered spindles—must be at least 0.5 seconds,

the time-scale length of the template was fixed at approximately 0.5 seconds. Spindles can last longer than 0.5 seconds, but require a minimum of 0.5 seconds to meet the requirements of a sleep spindle; therefore as long as the template detects at least a 0.5-second run of spindle-frequency activity, the spindle train can continue after the 0.5-second template and not adversely affect the operation of the template's correlation. The dominant spindle frequencies are in the 11–16 Hz band, with most occurring between 12 and 14 Hz. The center frequency of the match filter was set at 13 Hz with a fractional bandwidth of 0.08 (or 8%, meaning the bandwidth is 13$\pm$8%, which is approximately 12–14 Hz).

The frequency of K-complexes is approximately 1 Hz (or the wave shape normally occurs over the course of one second) therefore the template was designed to have a time-domain duration of one second.

3.3.5 Stage 3 Features

The AASM manual defines Stage 3 sleep as slow-wave sleep, and an epoch should be scored as Stage 3 when more than 20% of the epoch is dedicated to slow-wave, or slow delta, activity (0.3–4 Hz). Since that is the only defining characteristic for Stage 3, it becomes the extracted feature by default. Similar to the way that alpha and theta frequencies were extracted previously, this delta frequency band is evaluated by wavelet decomposition and percent energy. See Figure 3.9 for an EEG sample of Stage 3 sleep decomposed into its sub-bands.

3.3.6 REM Stage Features

Stage REM (or R) contains two unique characteristics that are not seen in other stages: very low chin EMG "tone" (or low levels of muscle movements) and rapid eye movements, the latter of which gives this stage its name. These two characteristics, then, were chosen to be the extracted features. The eye movements, located in the EOG signal, are sharp peaked waves and their semi-regular shape allows them to be extracted using a match filter, as shown in Figure 3.10.

The chin EMG signal needs to be evaluated for low amplitude, so RMS amplitude and maximal peak-to-peak amplitude were chosen to represent this feature [2]. The RMS amplitude was computed in Matlab by taking the 2-norm of the signal and dividing by the square root of the length of

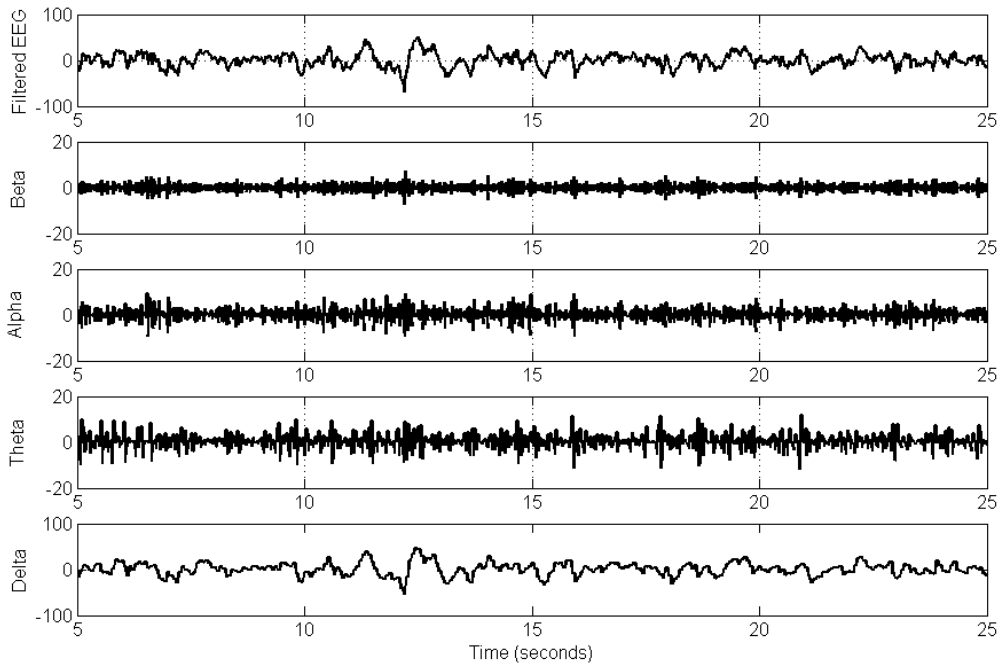

Figure 3.9: A sample of the output of wavelet decomposition performed on an epoch of Stage 3 sleep. The original signal is shown in the top plot, and then the decomposed signals in each of the four sub-bands of interest: β, α, θ, and δ (from the second plot down). The units on the y-axis are μV. Note the relatively low amplitude of β, α, and θ band activity and the large amplitude of the δ band activity.

the signal as shown below in equation (3.10):

$$x_{RMS} = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x_n^2} \tag{3.10}$$

The peak-to-peak amplitude was computed simply as the maximum value minus the minimum value, as shown in equation (3.11).

$$Amp_{peak-to-peak} = \max(\boldsymbol{x}) - \min(\boldsymbol{x}) \tag{3.11}$$

3.3.7 Linear Scaling and Normalization

Problems with feature bias in classification can arise when the numerical range of the features have large variability with respect to one another [46], as when the magnitude of the range of one feature

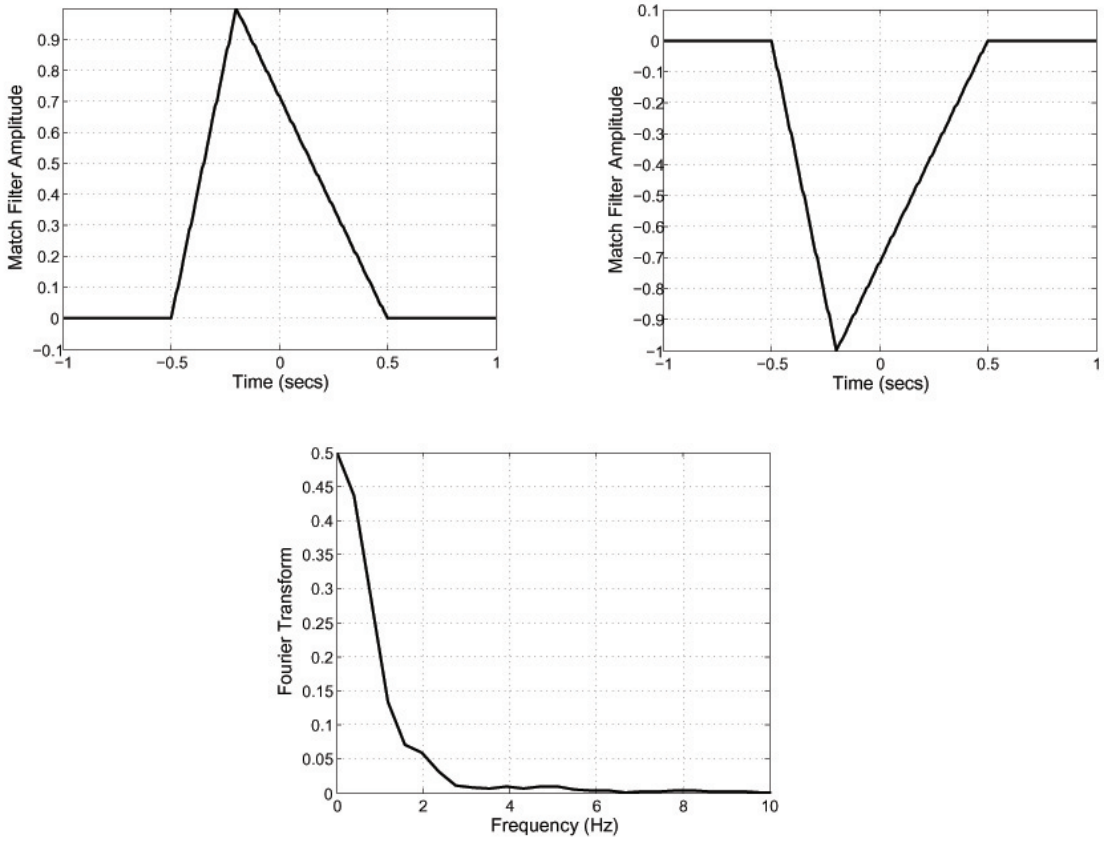

Figure 3.10: Match filters used to extract random eye movements (REMs). The filter magnitudes are shown at the top (left eye channel filter on the left and right eye channel on the right) and the frequency response is shown below.

is much greater than that of another. For example, the energy percentages calculated as a means of extracting information about different frequency bands will fall in the range of zero to one only, while the phasic EMG calculations can range from approximately 7 μV up to 200 μV (empirically determined from the data set), and therefore result in a classifier that would be much more sensitive to changes in the EMG signal than changes in the energy computations. One method of scaling to eliminate bias includes scaling the data to have zero mean and unit variance, but for this work the feature values were linearly scaled to the range of [-1 1] [62, 44].

The scaling was accomplished by taking the maximum and minimum values for each of the

Table 3.4: Features extracted for use in classification along with their abbreviations (right column).

Feature Number	*Description*	*Abbreviation*
1	Delta (δ) band energy	δ
2	Theta (θ) band energy	θ
3	Alpha (α) band energy	α
4	Beta (β) band energy	β
5	Sleep spindles	*SS*
6	K-complexes	*KC*
7	Left eye movements	EOG_L
8	Right eye movements	EOG_R
9	RMS muscle movement	EMG_{RMS}
10	Max amplitude muscle movement	EMG_{MAX}

features. Then, midpoints were determined using the calculation shown in equation (3.12):

$$mid\_val = \frac{max\_val - min\_val}{2} + min\_val \tag{3.12}$$

Once the minimum, maximum, and midpoint values were determined, each feature value was recomputed using equation (3.13) to scale each feature according to its individual range (thereby resulting in numerical features in the range $[-1\ 1]$):

$$feature\_val_{new} = \frac{feature\_val_{old} - mid\_val}{max\_val - mid\_val} \tag{3.13}$$

3.3.8 Feature Summary

In total, ten features were extracted from the EEG, EOG, and EMG data signals (summarized in Table 3.4): energy in the delta, theta, alpha, and beta frequency bands, sleep spindles, K-complexes, eye movements in the left and right eyes, and two different calculations of muscle movements. Each set of features (one for each epoch) then formed a 10-element vector, which was appended to an N-by-10 feature matrix (where N is the number of epochs recorded for a particular patient) to be passed on to the classifiers.

There are other rules and characteristics of sleep that are routinely observed and recorded but whose features are not strictly part of the definitions for one of the five sleep stages, including heart rate (electrocardiogram or ECG), arousals, limb movements, and respiratory rate or airflow.

Some of these may be counted as their own type of "stage" (such as movement time) or used to help conclusively diagnose sleep disorders (such as monitoring nasal airflow in patients with sleep apnea), but these other features were ignored in this work in favor of focusing on classifying only the five sleep stages as defined by AASM.

3.4 Design of Classifiers

For the purpose of this work, it was determined that designing both a neuro-fuzzy system and a support vector machine from scratch would not be necessary. Matlab toolboxes were used which provided the functions needed to create NFS and SVM classifiers.

The portions of the two systems that could be modified were the parameters used to affect the design of both the NFS and the SVM. These parameters are discussed in the following paragraphs.

3.4.1 Neuro-Fuzzy System Design

There are many different types of neuro-fuzzy system implementations, but one commonly used is the Adaptive Network-based Fuzzy Inference System (ANFIS), which is an NFS architecture developed by J.R. Jang. The structure provides an NFS solution that implements a Sugeno-type fuzzy system. The system has few constraints—it must be a feed-forward network with node functions that are continuous or piecewise differentiable—making it applicable to a variety of classification problems [5, 63].

For inputs $[x_1, x_2, \ldots, x_n]$, the fuzzy if-then rules take the following form [5, 63, 64]:

$$\textit{If } x_1 \textit{ is } A_1^1 \wedge x_2 \textit{ is } A_2^1 \wedge \ldots \wedge x_n \textit{ is } A_n^1 \textit{ then } y_1 = f^1(x_1, x_2, \ldots, x_n) = a_0^1 + a_1^1 x_1 + \ldots + a_n^1 x_n$$

$$\ldots$$

$$\textit{If } x_1 \textit{ is } A_1^r \wedge x_2 \textit{ is } A_2^r \wedge \ldots \wedge x_n \textit{ is } A_n^r \textit{ then } y_r = f^r(x_1, x_2, \ldots, x_n) = a_0^r + a_1^r x_1 + \ldots + a_n^r x_n$$

The rules that make up the system must be known in advance, as ANFIS only adjusts the premise and consequent parameter values. For a sample two-input (x_1 and x_2), the rules would be:

$$\textit{If } x_1 \textit{ is } A_1^1 \wedge x_2 \textit{ is } A_2^1 \textit{ then } y_1 = f^1(x_1, x_2) = a_0^1 + a_1^1 x_1 + a_2^1 x_2$$
$$\textit{If } x_1 \textit{ is } A_1^2 \wedge x_2 \textit{ is } A_2^2 \textit{ then } y_2 = f^2(x_1, x_2) = a_0^2 + a_1^2 x_1 + a_2^2 x_2$$

The network operation using those two rules, then, is as follows [5, 63]:

1. The first layer—the inputs are not counted as a layer—contains nodes that encompass the "premise parameters", or input membership function parameters. The membership functions are the interpretations of the quantifiers A_n, and follow the form:

$$O_n^1 = \mu_{A_n}(x), \tag{3.14}$$

where x is the input to the n^{th} node and μ_{A_n} is the function that defines the input's degree of membership. Typically, a bell-shaped curve is used, whose function is given by equation (3.15):

$$\mu_{A_n}(x) = \frac{1}{1 + \left(\frac{x - c_n}{a_n}\right)^{2b_n}}, \tag{3.15}$$

where $[a_n, b_n, c_n]$ are the parameters of the curve: c_n defines the center and $\frac{\pm b_n}{2a_n}$ is the slope of the line at $c_n \pm a_n$. See Figure 3.11 for an illustration of a sample bell curve and how it is defined by the given parameters.

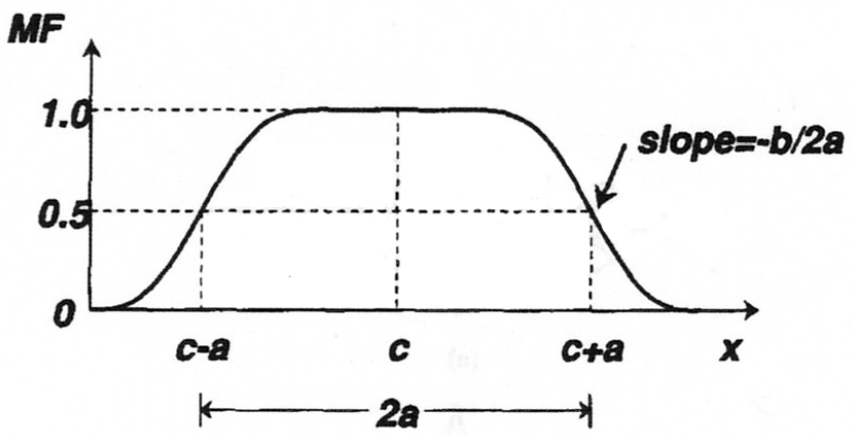

Figure 3.11: A generic sample bell curve membership function depicting the definition of ANFIS parameters *a*, *b*, and *c*. Adapted from [10].

2. The second layer is a multiplication of the outputs from layer 1. The products represent the "firing strength" of a particular rule, and are calculated using the equation shown in (3.16):

$$w_n = \mu_{A_n^r}(x) \times \mu_{A_n^r}(y), \tag{3.16}$$

where r and $n = 1, 2$.

3. The third layer takes in all the product results from layer 2 and computes a ratio of each individual rule's firing strength to a sum of all the firing strengths calculated in layer 2. This ratio can be viewed as a "degree of fulfillment" of each of the rules, and is given by equation (3.17):

$$\bar{w}_n = \frac{w_n}{w_1 + w_2} \tag{3.17}$$

   These outputs are also known as "normalized firing strengths."

4. The fourth layer is made up of one node for each of the outputs of layer 3. Each node takes as input one of the normalized firing strengths and one of each of the initial input values (x_n), for a total of $n + 1$ inputs. The output of these nodes can be calculated using the formula shown in equation (3.18):

$$O_n^4 = \bar{w}_n f_n = \bar{w}_n (a_0^r + a_1^r x_1 + a_2^r x_2) \tag{3.18}$$

   The coefficients a_n^r are known as the "consequent parameters."

5. The final layer (layer 5) computes the sum of all the signals output from layer 4 to give a final weighted sum. The summation is given by equation (3.19):

$$O_n^5 = \sum_n \bar{w}_n f_n = \frac{\sum_n w_n f_n}{\sum_n w_n} \tag{3.19}$$

Matlab's Fuzzy Logic toolbox provides an implementation of this neuro-fuzzy architecture, called "`anfis`." First, a fuzzy inference system (FIS), that implements the fuzzy logic portion of the NFS, must be generated. Next, that system is placed within a neural network framework for learning.

To create the FIS that will be input into the `anfis` function, Matlab's "`genfis`" functions can be used. While both `genfis1` and `genfis2` generate Sugeno-type FIS structures [8], `genfis1` was selected due to the nature of the definable parameters. The structure is generated with the following parameters: number of input membership functions per input, input membership function type (default value is for a bell-shaped membership function, "`bellmf`"), and output membership function type.

The default number of input membership functions is two, and this number was deemed sufficient for the purpose of this work. Classification of epochs into sleep stages can be stated as a binary problem: either the current epoch is in the current class, or not in the current class, and therefore lends itself to the use of two input functions that determine the degree of membership of the features for the current epoch.

Because triangular membership functions have already been used with automated classifiers [23], the decision was made to use a different type of function—for example, the Gaussian membership function—to explore potential improvement of the classification [65]. Therefore, the input membership function type was set to Gaussian ("`gaussmf`").

The ANFIS learning process proposed by Jang proceeds using a combination of backpropagation and least squares estimation (LSE). The former is used to modify the premise parameters (parameters defining the input membership functions in layer 1) and the latter is used to modify the coefficients associated with the consequents (weights assigned in layer 4). The network's learning, then, proceeds as follows [5]:

1. Fix the premise parameters, propagate the training inputs and determine the consequent parameters by LSE.

2. Fix the consequent parameters found in the first step, propagate the training inputs again and determine the premise parameters by backpropagation.

3. If the error was reduced in four consecutive steps, adjust the learning rate (either up or down depending on how the error is calculated).

4. If the error is less than the maximum allowable error, learning is done, otherwise repeat the process from step 1.

To tell the ANFIS network that this "hybrid learning" is the desired training method, the optimization code ("`optMethod`") for the `anfis` function was set to "1."

The output membership function parameter values offered a choice of either linear or constant. Due to empirically-achieved results, the constant output membership function was chosen because of its superior performance.

The number of training epochs (or, the number of times the training inputs would be propagated through the structure) was increased from the default setting of 10 to 1000. This gave the NFS more

iterations through its learning process and therefore a greater chance of finding a solution with a low error rate.

The output of an ANFIS classifier is a weighted average of the inputs [5, 63]. Since the sleep stages are coded using integers, a conversion method was needed to translate the averages to output that could be compared directly to the sleep stage classes. So, the weighted average output was rounded to the nearest integer, allowing the classifier to select one of the integer classes definitively. This practice was deemed sufficient because the weights in the network had, at that point in the process, already impacted the decision to result in a number that "lands" closer to one integer than another. Therefore, it was plausible that rounding the output to get the "closest" class was decided to be a sufficient way to achieve integer output.

A diagram of the ANFIS classifier built with the aforementioned parameters is shown in Figure 3.12.

3.4.2 Support Vector Machine Design

Support vector machines operate on the principle that there exists some training sample set which is separable via a hyperplane. The optimal separating hyperplane selection is a definable parameter in Matlab, and the method chosen was the quadratic programming (QP) method, which results in a two-norm (2-norm), soft-margin SVM. This classifier has been shown to produce a more accurate generalized classifier than the maximum margin SVM, which typically has a more specific, complex decision boundary.

A 2-norm soft-margin separating hyperplane is defined as follows [6, 66, 52]:

1. A training sample S is defined as:

$$S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)) \subseteq (X \times Y)^m, \tag{3.20}$$

   where m is the total number of data points, $X \subseteq \mathbb{R}^n$ represents the input space, and $Y = \{-1, +1\}$ represents the output domain for a two-class (binary) classifier.

2. Support vector machines are based on a class of hyperplanes defined as:

$$\langle \mathbf{w}, \mathbf{x} \rangle + b = 0 \text{ for } \mathbf{w}, \mathbf{x} \in \mathbb{R}^n, b \in \mathbb{R}, \tag{3.21}$$

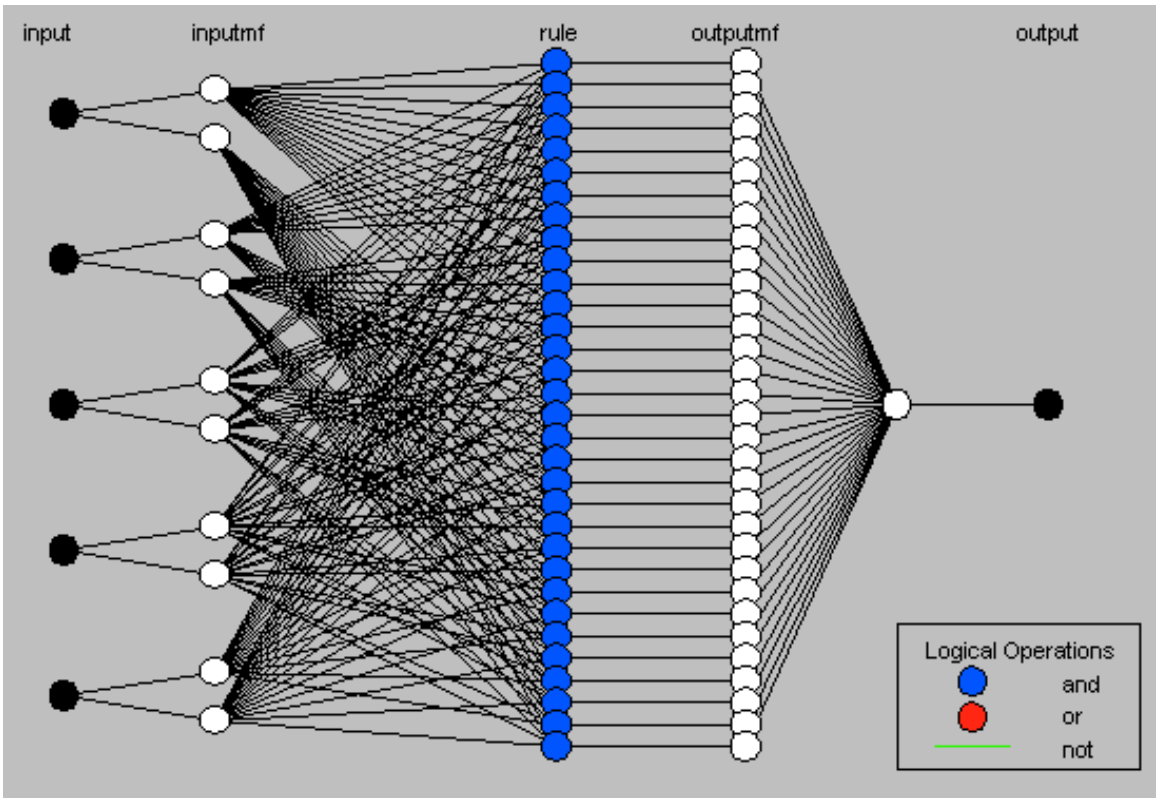

Figure 3.12: The Sugeno-type ANFIS used for each of the ten binary classifiers. Each classifier has five input features ("input" layer), which are then passed to the two membership functions that determine the degree of membership of each of the features ("inputmf" layer). The results are then multiplied (or combined using "AND") and degree of fulfillment determined in the third layer ("rule" layer); there were a total of 32 fuzzy rules generated for this design. Finally, the weighted functions are evaluated ("outputmf" layer) and summed to create a single decision output ("output" layer).

where the points, $\mathbf{x}$, lie on the hyperplane and satisfy Equation 3.21, while the weight vector $\mathbf{w}$ defines the direction perpendicular to the hyperplane, and b adjusts the parallel position of the hyperplane with respect to itself.

3. Then, instead of a maximum margin hyperplane calculation, we want to construct a soft-margin hyperplane. To do this, the "slack variable" ξ is introduced into the optimization problem:

$$\text{minimize}_{\mathbf{w},\, b} \langle \mathbf{w}, \mathbf{w} \rangle \tag{3.22}$$

$$\text{subject to } y_i(\langle \mathbf{w}_i, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \text{ for } \xi_i \geq 0, i = 1, \ldots, m. \tag{3.23}$$

4. The slack variables ξ_i, along with the parameter C—which controls the trade-off between classification errors and margin—modify the 2-norm minimization problem as follows:

$$minimize_{\xi,\,\mathbf{w},\,b}\langle \mathbf{w},\,\mathbf{w}\rangle + C\sum_{i=1}^{m}\xi_i^2, \tag{3.24}$$

subject to Equation 3.23.

5. The primal Lagrangian for the minimization problem in 3.24 can be given as:

$$L(\mathbf{w},\,b,\,\xi,\,\alpha) = \frac{1}{2}\langle \mathbf{w},\,\mathbf{w}\rangle + \frac{C}{2}\sum_{i=1}^{m}\xi_i^2 - \sum_{i=1}^{m}\alpha_i[y_i(\langle \mathbf{w}_i,\,\mathbf{x}_i\rangle + b) - 1 + \xi_i], \tag{3.25}$$

where $\alpha_i \geq 0$ are the Lagrange multipliers.

6. Next, the corresponding dual is found by taking partial derivatives of $L(\mathbf{w},\,b,\,\xi,\,\alpha)$ with respect to $\mathbf{w}$, ξ, and b and imposing stationarity, then resubstituting the resulting relations into the primal in Equation 3.25 and simplifying to obtain the objective function below:

$$L(\mathbf{w},\,b,\,\xi,\,\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}y_iy_j\alpha_i\alpha_j\langle \mathbf{x}_i,\,\mathbf{x}_j\rangle - \frac{1}{2C}\langle \alpha,\,\alpha\rangle. \tag{3.26}$$

7. Hence, maximizing the objective function 3.26 over α is equivalent to maximizing the following:

$$W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}y_iy_j\alpha_i\alpha_j\left(\langle \mathbf{x}_i,\,\mathbf{x}_j\rangle + \frac{1}{C}\delta_{ij}\right), \tag{3.27}$$

where δ_{ij} is the Kronecker δ, defined as 1 if $i = j$, and 0 otherwise. Taking into the Kuhn–Tucker conditions:

$$\alpha_i[y_i(\langle \mathbf{x}_i,\,\mathbf{w} + b) - 1 + \xi_i] = 0, \text{ for } i = 1,\ldots,m, \tag{3.28}$$

the resulting quadratic optimization problem is:

$$W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}y_iy_j\alpha_i\alpha_j\left(\langle \mathbf{x}_i,\,\mathbf{x}_j\rangle + \frac{1}{C}\delta_{ij}\right), \tag{3.29}$$

$$\text{subject to } \sum_{i=1}^{m} y_i \alpha_i = 0 \text{ for } \alpha_i \geq 0, i = 1, \ldots, m. \tag{3.30}$$

The implementation of the SVM, then, can be changed by varying the parameters that define the learning kernel used and how the algorithm divides the feature space (etc.). In the design of the SVM classifier in Matlab, a number of definable parameters are available to tailor the classifier to suit the problem at hand.

First, an appropriate kernel function must be chosen. Matlab offers five different built-in kernels; given the suggestions offered by others in the field, a radial basis function (RBF) kernel was selected for its general robustness and efficiency [62, 67].

The output of the SVM classifer is determined by equation (3.31), which is the equation for the separating rule of an RBF kernel:

$$F(x) = \text{sign}(f(\mathbf{x})) = \text{sign}\left(\sum_{i=1}^{N} a_i K_\sigma(|\mathbf{x} - \mathbf{x}_i|) - b\right), \tag{3.31}$$

where $K_\sigma(|\mathbf{x} - \mathbf{x}_i|)$ is given by equation (3.32):

$$K_\sigma(|\mathbf{x} - \mathbf{x}_i|) = e^{-\sigma|\mathbf{x}-\mathbf{x}_i|^2} \tag{3.32}$$

Often, when discussing SVM design, researchers using RBF kernels frequently mention the importance of parameters C and σ (Note: sigma, or σ, is also referred to as gamma—γ—in certain texts; σ is used here because Matlab defines it as such). The variable σ is the "width", or variance, of the RBF function. This parameter can take on any one of a number of values, with the smaller values indicating a tighter decision boundary. The parameter C, which defines the penalty for allowing misclassifications, can also take on any number of values. Lower values of C drive the classifier toward a soft-margin classification scheme, meaning more samples are allowed to be misclassified, while higher C values drive the classifier toward a "hard" margin, which attempts to find a more absolute separation at the expense of creating a more complex and potentially over-fitted boundary [6, 68, 61].

To find the optimal σ value, a range of values (from 0.1 up to 2 in varying increments) were assessed in an attempt to find the solution empirically that would result in the least amount of error.

The C parameter was also varied over a range of values (from 0.01 up to 1000 in increments of one order of magnitude). To find the best combination of these two parameters, a grid search iterating over the 72 possible [C,σ] combinations was performed because it has been shown that varying only one at a time can yield sub-optimal results [69].

The Matlab SVM training function, `svmtrain`, uses a function called `quadprog` to solve the quadratic programming problem. This function, in turn, uses the function `qpsub`, which is an implementation of the Newton method algorithm as described in [70], a means of minimizing the quadratic function, and each iteration through the algorithm is the equivalent of one Newton step. The number of iterations is the first of two stopping criteria for the training function; the other is the termination tolerance for constraint violation: the termination constraint violation is dictated by the parameter epsilon ("`eps`")—the distance from 1.0 to the next largest double-precision number—with a default value of 2^{-52} [8]. The number of iterations for this SVM design was set to 1000 to match the number of training epochs used in the NFS, and to allow the classifier an adequate amount of time to find suitable hyperplane definitions, while epsilon remained at its default value.

3.5 Training and Testing

Once the training vectors—feature matrices—were completed and the classifier's designs had been determined, the training and testing sets for the classifiers needed to be chosen. There are different theories about the best way to choose training and testing sets, but generally accepted practice dictates that sets are chosen from a larger data set randomly and that different classes have nearly equal representation so as not to under- or over-train the classifiers.

3.5.1 Data and Feature Subset Selection

First, a version of stratification was employed. Stratification is typically used to ensure that equal portions of all classes are represented in the training and testing groups as a basic preventative measure against over- or under-training one class over the others (which introduces bias) [71]. Because of the large number of samples (total of 30,973 epochs) unequally distributed over the five classes, an initial stratification and sample number equalization was performed to bring the total number of training and testing points to a reasonable, uniformly-distributed amount in a first step toward

mitigating potential class-bias problems [72, 73]. This was done by first determining the class that contained the fewest number of samples; in this case, Stage 1 had the least number of epochs represented at 1348 (see Table 3.1). Then, a percentage of those Stage 1 epochs—25%—were selected to be included in training and testing, while the same number of epochs ($1348 \times 25\% = 337$ epochs) were randomly selected from the remaining classes to be represented for a total of 1685 samples available to use for training and testing. The sleep stage labels corresponding to those samples were then pulled out and put into a separate vector.

Note that the variability of sleep patterns between subjects is higher than the variability of the sleep patterns from within one subject [74, 75]. However, training or testing with only one subject at a time does not support system robustness (the ability of the classifier to perform well in other situations). Developing the system with all available patient data is necessary to provide a better prediction of how the system might perform for generalized applications, so the aforementioned randomization of sample selection was employed.

One constraint Matlab places on its NFS classifier is that the total number of rules created can not exceed 250. The number of rules created for any given problem is given by the number of input membership functions to the power of the number of features. Therefore, with two input membership functions, the highest number of features the NFS could handle is $2^7 = 128$, since using 8 features would put the number of rules at 256 and over the Matlab-imposed limit of 250. So, out of the ten available features, feature subsets needed to be picked that included—at most—seven features. Due to processing capabilities, however, it was determined empirically that feature subsets of five features gave sufficient AASM stage information (allows exactly one unique feature per class, see Table 3.2 for the extracted features for each stage) while cutting down on the time and possible feature space to be searched. The number of all possible combinations ($C_{\text{n-choose-m}}$) consisting of five of the ten extracted features with no repeats and without regard to order can be found using equation (3.33):

$$C_{\text{n-choose-m}} = \frac{n!}{m!(n-m)!} \tag{3.33}$$

With a selection of five features out of ten (or ten, choose five), the total number of potential feature subsets was 252.

Table 3.5: A table detailing which binary class-pairs were assigned to which classifiers. The stages' respective codes, used for labeling the stages for use with the classifiers, are shown in parentheses.

Classifier #	*Class #1:*	*Class #2:*
1	Wake (0)	Stage 1 (1)
2	Wake (0)	Stage 2 (2)
3	Wake (0)	Stage 3 (3)
4	Wake (0)	REM (5)
5	Stage 1 (1)	Stage 2 (2)
6	Stage 1 (1)	Stage 3 (3)
7	Stage 1 (1)	REM (5)
8	Stage 2 (2)	Stage 3 (3)
9	Stage 2 (2)	REM (5)
10	Stage 3 (3)	REM (5)

3.5.2 Training and Testing Methodology

From there, a one-against-one (OAO) method of training and testing was used. The OAO method is generally understood to have better results than other methods, like one-against-all (OAA), but can be computationally expensive, as the number of classifiers that need to be built is given by (3.34), where n is equal to the number of classes:

$$N = \frac{n \times (n-1)}{2} \tag{3.34}$$

However, with only five classes (and therefore an N equal to ten), it was determined that building, training, and testing ten binary classifiers would still be a reasonable number. Table 3.5 details the class-pair comparisons that were made for each of the ten binary classifiers.

To give the classifiers the best possible chance of discriminating between classes (and since it is known that some features will appear in some classes and not at all in others), each class-pair was tested with each of the 252 feature subsets (described in the previous section, section 3.5.1) to find the combination of five features that best separates each class-pair. Out of the 2,250 possible combinations, the top ten performing feature subsets—to be used with both the NFS and SVM classifiers—were chosen and assigned to each corresponding class-pair. Table 3.6 indicates the set of features (from Table 3.4) used for class separation by each of the ten binary classifiers.

Once an appropriate subset of data was extracted and the classifier design finalized, the k-fold cross-validation technique was used to generate training and testing sets, with $k = 10$. K-fold

Table 3.6: A matrix illustrating which features were sub-selected for use with each of the 10 classifiers (binary class comparisons). The extracted features are shown in the column headers, the binary classifiers are shown in the left-hand column, and the features that were used for a particular classifier are indicated by dots in the appropriate columns.

Classifier	δ	θ	α	β	*SS*	*KC*	EOG_L	EOG_R	EMG_{RMS}	EMG_{MAX}
Wake/Stg. 1	•			•	•	•			•	
Wake/Stg. 2	•			•	•	•			•	
Wake/Stg. 3			•		•	•	•		•	
Wake/REM		•			•	•	•		•	
Stg. 1/Stg. 2		•	•		•			•		•
Stg. 1/Stg. 3			•				•	•	•	•
Stg. 1/REM		•		•		•	•			•
Stg. 2/Stg. 3		•	•		•			•		•
Stg. 2/REM		•	•		•				•	•
Stg. 3/REM		•		•		•			•	•

cross-validation is a technique which splits the available data into k partitions, using $k-1$ partitions for training the classifier and the remaining partition for "testing," or validation. This process is repeated k times, until all partitions have each been used once for testing, and the error for all of those k runs can be averaged to give an overall error rate. This technique is used when relatively few samples are available and it can not be guaranteed that all the samples in a group would be considered good examples of particular classes—for example, a Stage 2 epoch might be labeled as such, but not have any discernible K-complexes or sleep spindles. The lack of "good" samples will decrease the likelihood that the classifier will be able to see the separation between classes during training, since some of the epochs would not have clearly discernible features, and also decrease the probability that the classifier will correctly identify stages upon testing. However, even if a stage does not have all the defining characteristics present, it still should be classified as that particular stage. That being the case, k-fold cross-validation—with $k = 10$ for the best results using the smallest possible number of folds—for training and testing set selection was implemented in an effort to minimize this good sample/bad sample effect [71].

For each class-pair, starting with Wake (code 0) and Stage 1 (code 1), the proper C and σ values (needed for the SVM classifier) were assigned to temporary variables and all samples of the two current stages of interest were assigned to temporary matrix structures. The class labels were then assigned logical values of 0 and 1, with the "current" class—or the class in which the samples

are looking for membership—receiving the value of 1. The "comparison" class—the class that is outside the current class of interest—is assigned the value of 0. The appropriate five-feature subset was then extracted from the two classes' feature matrices and re-assigned to the temporary matrix of features.

Each of the ten binary classifiers (ten SVM classifiers and ten NFS classifiers) were then trained on these temporary data structures using the appropriate class groupings from the first nine k-fold cross-validation indices designated for training. Once training was performed using the first k-fold training group, the binary classifiers were then tested on the one held-out k-fold testing group. Training and testing continued until all ten groups had each been tested once.

3.5.3 Class Voting and Decision Methodology

Implementation of ten separate binary classifiers necessitates combination of the results by some method. Class voting systems have been used frequently for this sort of decision making with good results, so a simple voting system was created to consolidate the output into a single vector of classes.

To keep track of how many times a particular sample is classified as a particular class, a voting matrix (of size 1685 samples by 5 classes) was created for each of the two classifiers. As each sample was selected for membership as either belonging to the current class of interest or not (and therefore making the decision that it belonged to the comparison class), the corresponding index in the voting matrix was incremented by one.

After all of the samples had been passed through the classifiers and voted on, each samples' votes were assessed and the sample was assigned to the class that received the most votes. That final vector of class assignments—one per classifier—became the output that was compared to the target classes for performance evaluation.

Breaking Voting Ties

In some cases, the votes ended in a tie, with multiple class labels having acquired equal numbers of votes. Typically, ties are broken arbitrarily, with a random class picked out of the tied classes , but in an effort to introduce some "smart" decision making, a primitive sort of confidence measure was developed. A "distance matrix" (labeled as such because it tracked the distance of sample points to

their respective classes), in the same dimensions as the voting matrix, was created for each of the two classifiers.

NFS Decision Method

Since the output of the NFS is a weighted average, and therefore likely not an integer value, the decimal portion of the number can be used as a sort of confidence measure to break voting ties. Earlier in the process, the class labels were changed from the sleep stage codes ("0," "1," "2," etc.) to a simple binary one or zero (either the sample is in the class or out of the class respectively). In this case, then, the closer the weighted average output is to one, it could be said that that sample has more "confidence" with respect to membership in the current class, and the closer the output is to zero (or to two or higher), the less confident we are about that sample's membership in the class. Therefore, a basic "confidence metric" in (3.35) was implemented to evaluate how close the weighted average output is to one:

$$\text{Confidence Metric} = |1 - x| \tag{3.35}$$

For each vote a class received in the voting matrix, the NFS distance matrix added the result of this confidence metric computation to the appropriate index. So, in the case of a tie, all the possible class values are extracted (all the classes that received votes), along with their corresponding confidence measures from the distance matrix. Since a value close to one means more confidence is class membership, the confidence metric that was found to have the smallest result was picked as the "winning" class.

SVM Decision Method

The SVM tie-breaking method was more straightforward to implement, because Matlab's SVM function already computes a metric corresponding to the evaluation of the class determination function, which gives the distance from the current point to the established separating hyperplane. This variable is calculated in a private function, and the calling function does not save this number as an output, but the function result can be made visible simply by adding it to the list of outputs in the decision function. Therefore, as each class gets a vote in the voting matrix, the absolute value

of the distance measure associated with that sample gets added to its corresponding location in the distance matrix.

In the event of a tied vote in the SVM voting matrix—similar to the NFS tie-breaker—all possible class values are extracted along with their corresponding distance measures. In this case, however, it is reasonable that the sample that lies furthest from the separating hyperplane is the sample that would have the most confidence in belonging to one or the other class. So, then, the class associated with the largest distance metric is picked as the "winning" class.

Table 4.1: Confusion matrix for the NFS results. This indicates the percentage of samples per stage (”actual,” right most column) that were identified as each of the other stages (”predicted,” header row).

Actual/Predicted	*Wake*	*Stage 1*	*Stage 2*	*Stage 3*	*REM*
Wake	81.01%	13.95%	1.48%	1.93%	1.63%
Stage 1	12.46%	60.98%	11.13%	1.93%	13.50%
Stage 2	4.75%	21.07%	49.70%	10.83%	13.65%
Stage 3	1.78%	0.15%	9.94%	87.83%	0.30%
REM	3.26%	34.87%	9.79%	4.01%	48.07%

Chapter 4

Classification Results

The following section contains the output results of the NFS and SVM classifiers and provides a comparison of the performance of the two methods.

4.1 NFS Results

A confusion matrix of the output of the NFS is presented in Table 4.1, and a heatmap of the same information is presented in Figure 4.1, while a graph of the actual stage classification versus the classifier output is shown in Figure 4.2.

4.2 SVM Results

A confusion matrix of the output of the SVM is presented in Table 4.2 and a heatmap of the same information is presented in Figure 4.3, while a graph of the actual stage classification versus the classifier output is shown in Figure 4.4.

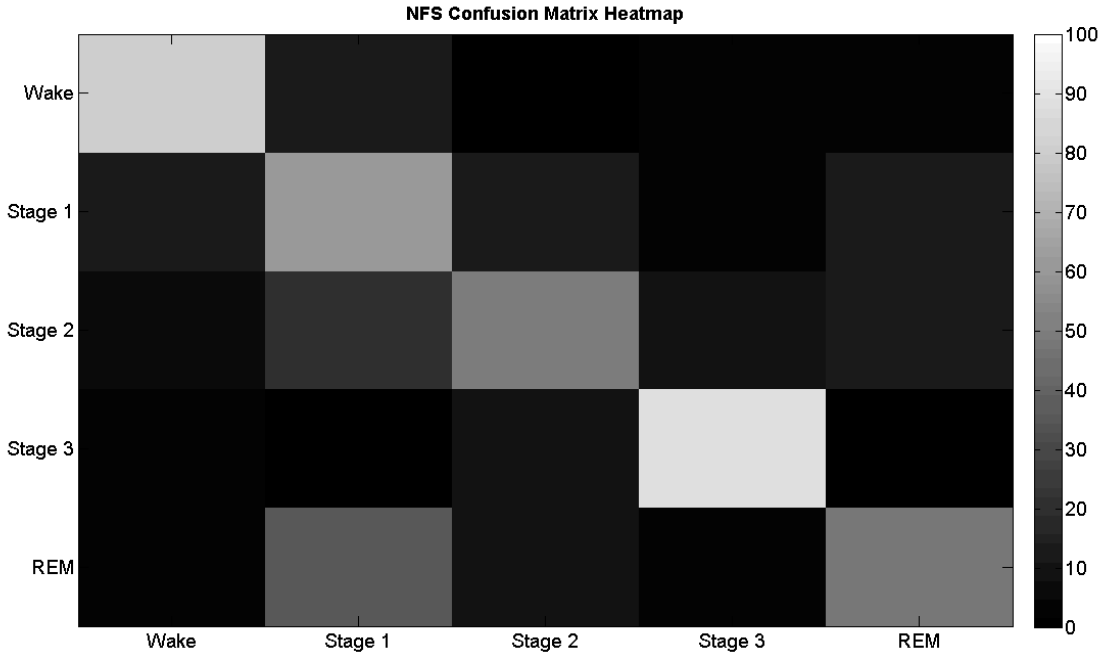

Figure 4.1: A heatmap representation of the NFS confusion matrix. The gradient indicates percentage agreement, with white indicating a 100% agreement rate and black at 0%. The gradient legend to the right of the figure shows the percentages that correspond to each grayscale level.

Table 4.2: Confusion matrix for the SVM results. This indicates the percentage of samples per stage (”actual,” right most column) that were identified as each of the other stages (”predicted,” header row).

Actual/Predicted	*Wake*	*Stage 1*	*Stage 2*	*Stage 3*	*REM*
Wake	78.78%	16.02%	1.63%	1.34%	2.23%
Stage 1	9.79%	63.80%	12.61%	1.63%	12.16%
Stage 2	3.56%	22.55%	44.81%	14.10%	14.99%
Stage 3	0.74%	0.15%	9.64%	89.17%	0.30%
REM	2.23%	33.98%	7.86%	4.75%	51.19%

4.3 Comparison of NFS and SVM Classification

The overall correct rate (percent correct), sensitivity, specificity, and Cohen’s kappa for both classifiers are presented in Table 4.3. In both cases, the Cohen’s kappa calculation returned results indicating both classification techniques have moderate agreement and that the “observed agreement is not accidental.”

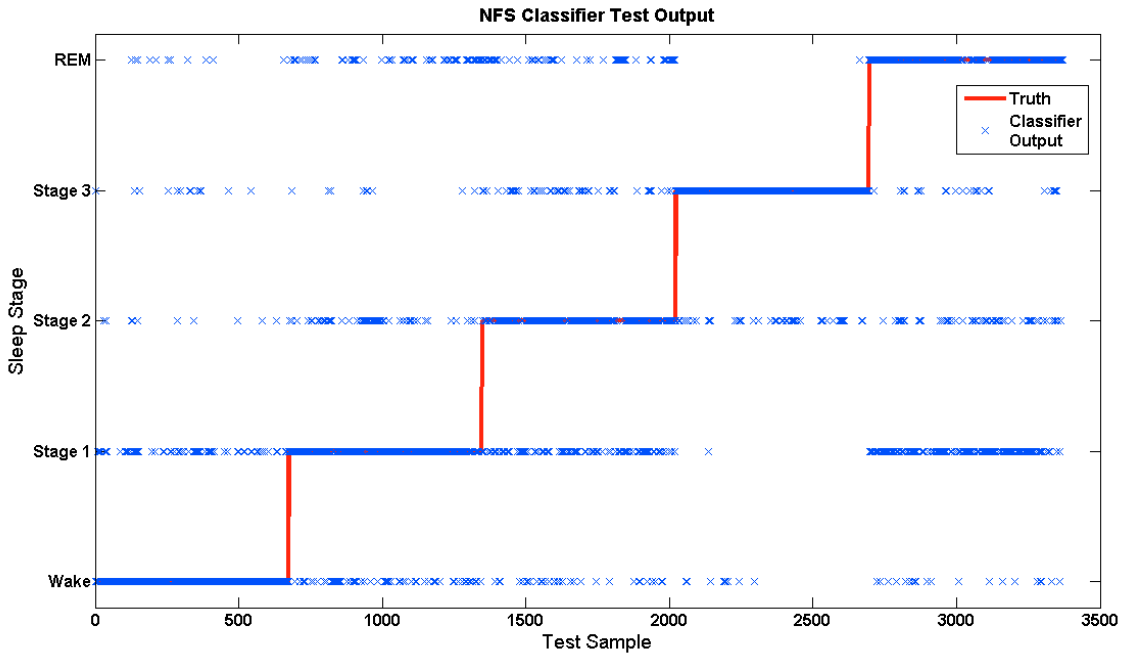

Figure 4.2: A graphical representation of the NFS classifier's test output. The red line indicates the known true stage of each of the test samples (the "truth"), while the blue x's indicate the stage classification assigned by the NFS classifier for each of the test samples. X's which do not fall on the red line are misclassifications.

Table 4.3: Comparison of results between the NFS and SVM. The percent correctly classified overall and by stage, as well as overall sensitivity, specificity, and Cohen's kappa are shown.

Results Summary	*NFS*	*SVM*
Wake	81.01%	78.78%
Stage 1	60.98%	63.80%
Stage 2	49.70%	44.81%
Stage 3	87.83%	89.17%
REM	48.07%	51.19%
Overall accuracy	65.52%	65.55%
Sensitivity	81.01%	78.78%
Specificity	94.44%	95.92%
Cohen's kappa	0.5690	0.5694

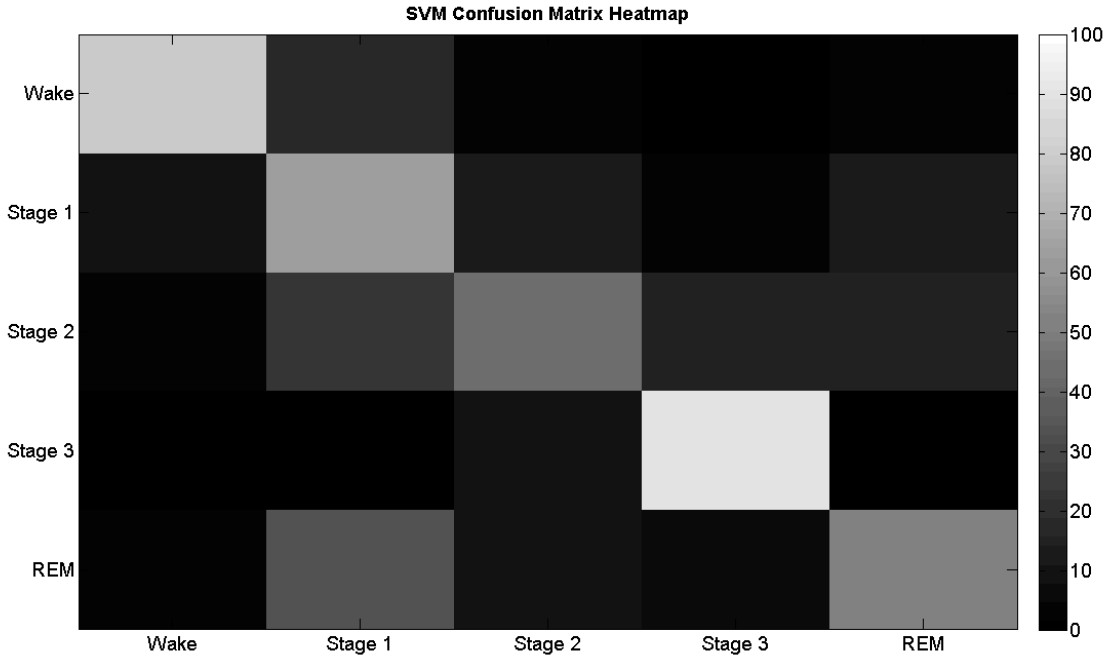

Figure 4.3: A heatmap representation of the SVM confusion matrix. The gradient indicates percentage agreement, with white indicating a 100% agreement rate and black at 0%. The gradient legend to the right of the figure shows the percentages that correspond to each grayscale level.

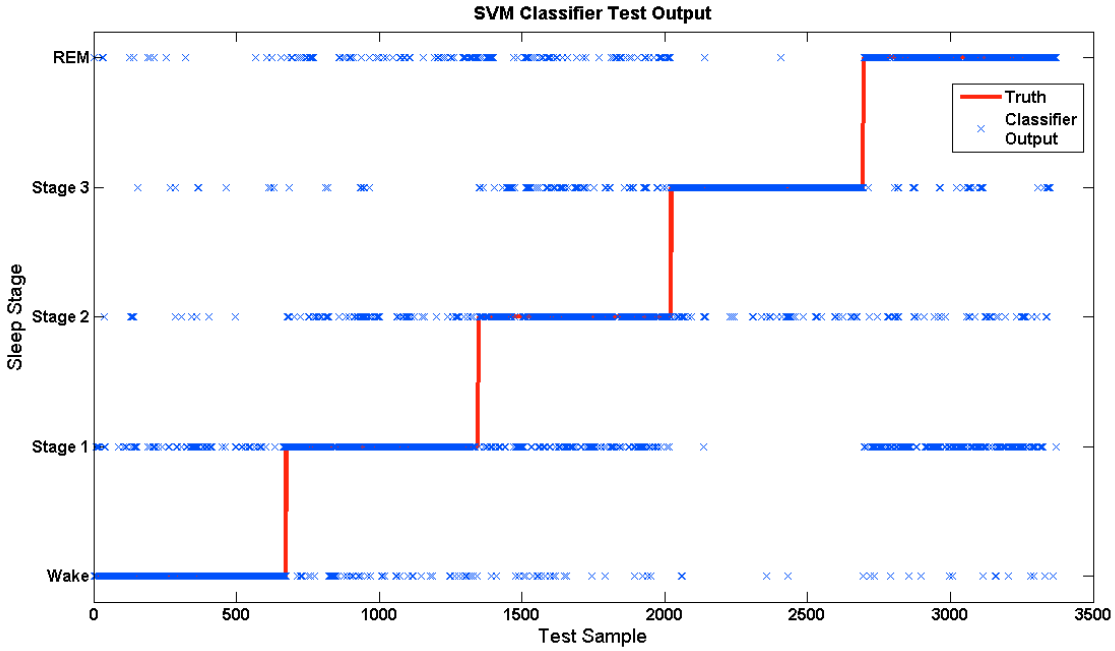

Figure 4.4: A graphical representation of the SVM classifier's test output. The red line indicates the known true stage of each of the test samples (the "truth"), while the blue x's indicate the stage classification assigned by the SVM classifier for each of the test samples. X's which do not fall on the red line are misclassifications.

Chapter 5

Discussion

This section contains the discussions regarding the methods and subsequent observed results, as well as some suggestions for future pursuits in this area.

5.1 Analysis of Methods

The methods used in this thesis were drawn from a number of sources, and choices made regarding the parameters and implementation of these methods certainly had an impact on the results. Discussion of this impact follows.

The EEG channel available was C4-M1, only one of three AASM-recommended channels used for sleep scoring (recall that the other two are F4-M1 and O2-M1). Due to this, it is possible that information that would have been evident on one or both of the two missing channels might have increased the probability of certain pattern detection. For example, some researchers have stated that it is "easier" to see K-complexes in the frontal lobe derivations because that is where that pattern is likely to be maximal in amplitude across all patients [1]. With the pattern-dampening effect due to this "detection at a distance" (or, a greater distance than is deemed ideal), there is the possibility that some stage characteristics were missed during feature extraction, thereby having a detrimental impact on the results.

With regard to the feature subset selection, recall that each class-pair was tested with each of the 252 feature subsets to find the combination of five features that best separated each class-pair. This set of empirically-selected features was an attempt to mitigate bias related to the selection of the feature subset. While the AASM manual dictates the type of features that should be used to identify the stages, those features may not be the same ones that are ideal for providing the best distinction

between stages.

The idea that the features that provide the best identification are not necessarily those features that are best for providing class separation is illustrated in Table 3.4. One feature that is often used to separate wake (0) from stage 3 sleep (3) is the δ band energy. However, in this work the features empirically chosen to separate those two classes, which are subject to the effectiveness of the feature extraction methods, were the α band energy, the eye movements (both channels), and the muscle movements (both calculations). There is the possibility that this seeming incongruent result comes from the methods used to extract the features—potentially sub-optimal for the application—but these results suggest that the feature separability of the stages should be defined by the actual data and not confined to only exact implementations and comparisons of the stage definitions provided by AASM [21, 76]. Additionally, while these subsets were selected using a brute–force search of all possible five–feature set combinations, the selection of the highest performing set combinations was based on a single training and testing run per set, and therefore was not necessarily truly exhaustive.

Misclassification rates are also impacted by the large variability of sleep patterns between subjects [74, 75]. Because the data sets for training and testing were selected randomly from a combination of all 25 available subject files, there is a high possibility that the variability of the samples within classes is too high for the classifiers to definitively separate. For example, the wake stage features could be sufficiently different from the first subject to the second subject, and therefore training on both of those feature samples could be detrimental when attempting to classify the wake stage features of the third and fourth subjects. Attempting to develop a classifier that works across many patients requires a design suited to generalization (given inter-subject feature variability), particularly when the data do not contain all the recommended channels and the channels that are available may not be ideal for detecting certain features.

One common problem with machine learning and classification is overfitting, which refers to instances for which the classifier learns the training data too well, resulting in poor generalization [77, 78]. Any learning algorithm or classification problem can suffer from overfitting, and the ANFIS is particularly susceptible to overfitting issues [79]. Two things that can be used to mitigate the overfitting problem are: setting a training error goal (or constraint) and defining the maximum number of training iterations. There is, for every problem, some optimal number of iterations that will minimize the amount of error a validation set will generate. The error constraint indicates the

maximum amount of error the model is allowed, and each iteration through the system attempts to bring the error down toward zero. However, at some point, the amount of error generated will stop decreasing and start increasing; that is the point at which the model is overfitting to the data and any further increase in the number of iterations past that point is detrimental to the learning algorithm. If there is the potential that the number of iterations could cause this overfitting to occur, the error constraint can act as a limiter by stopping the training when the error reaches some predefined acceptable level. Then, the training is halted either when the error constraint condition or the maximum number of iterations is satisfied. Generally, the number of iterations is defined empirically; that is, training and validation is performed over a range of iterations, and the error rates are monitored for the minimum [79, 80]. The ANFIS in this work was not tested for the optimal iteration count, but was instead given both an iteration maximum—identical to that of the SVM in an attempt to make the classifiers as comparable as possible—and an error constraint, and evaluated only with the NFS output results.

While certain amounts of subject variability, and the subsequent classification difficulty, were expected, steps were taken to mitigate any effect the difference in classifiers would have on the output. In an effort to make the results as comparable as possible, all steps of classification were performed the same way—the same training and testing data was input to both classifiers—so that the only difference between the two was the classification decision process, such as how it was decided whether a class would be included or not, inherent in the classifier design.

5.2 Analysis of Results

The objectives of this thesis were as follows:

- Develop an automated sleep scoring algorithm using the 2007 AASM sleep manual's stage definitions by implementing and evaluating a neuro-fuzzy system classifier.
- Develop an automated sleep scoring algorithm using the 2007 AASM sleep manual's stage definitions by implementing and evaluating a support vector machine classifier.

Both of these objectives were met: two classifiers were developed, one using NFS and one using SVM, which were trained and tested on labeled features as defined by the 2007 AASM manual. The

results were presented in chapter 4, and using those results, some answers can now be given to the questions that were posed in section 1.1:

1. Can the new AASM staging definitions be used with an automated process to correctly identify the stages of sleep?

   Yes, with variable success. For both the NFS and SVM classification techniques, the overall percent correct was approximately 65%, with sensitivity and specificity rates around 80% and 95%, respectively. These results (which use AASM for the classifier stage definitions) fall within the recently reported agreement range of 50–89% between manual and automated scoring using the R&K staging definitions [25]. The overall Kappa scores, one means for evaluating system reliability, were approximately 0.57 for both the NFS and SVM, indicating moderate agreement that was not "accidental" [27]. These Kappa results also fall within a recently reported range of κ = 0.55 to 0.81, a range indicating moderate to "almost perfect" agreement [27, 25].

   Although the agreement rates and Kappa scores alone are not yet high enough for confidence in independent use of the classifiers for automated scoring in clinics or research, detection accuracy for some of the individual stages was favorable. Both the NFS and SVM classifiers detected some of the stages with higher accuracy than others; for example, stage 3 was detected with an 87–89% success rate, while REM sleep was only detected correctly half of the time. That stage 3 accuracy, however, is greater than the 78% classification rate for stage 3 achieved by a recent implementation of a neuro-fuzzy system [50], and the authors of that study concluded that their classifier (which had an overall accuracy of 74.7%) was an improvement over other previous attempts, suggesting that the methods developed for stage 3 detection in this thesis are an additional improvement. These results for the automatic classification of stage 3, or slow wave sleep (SWS), using NFS and SVM could be interesting for those who are involved in medical or memory research and SWS. For example, lower SWS levels have been associated with an increased risk for type 2 diabetes [81], while "good" levels of SWS have been associated with improved cognition and memory consolidation [82, 36].

2. Do implementations that have proven effective in the past show improvements in classification when using the new AASM guidelines?

The results of this work suggest "no," except a number of other factors must be taken into account. Unfortunately, with a lack of available data that conform to the AASM recommendations, only partial derivations of the AASM guidelines were used. For example, the full 10-20 complement of surface electrodes is rarely used outside of sleep clinics, so obtaining all recommended EEG scoring channels (F4, C4, and O2) for this investigation proved challenging. Additionally, the field of sleep is in an evolutionary period, in which opinions about the direction of sleep assessment are vastly different; some researchers question the value of the AASM approach [19], some question the value of sleep staging [20, 21], and some applaud and encourage still more work using AASM [22]. With the "moving target" that is sleep research, there are many efforts whose value may not be apparent until more universal, consistent approaches to sleep definitions and assessment have been developed.

3. Is there a difference in performance between classifier implementations using the new AASM guidelines with respect to sleep features and scoring?

   No. Two different classification techniques, neuro-fuzzy systems and support vector machines, when evaluated using the same data set and same input features, perform very much the same (65% accuracy with 80% sensitivity and 95% specificity). This suggests that the quality, separability, and clarity of definition of the input features are the more important factors in achieving a highly capable automated sleep classifier, and not the classification method itself [83]. One must also consider implementing optimization techniques to achieve the best possible results for any given classification method.

   While future implementations might find differences between NFS and SVM approaches, it is interesting to note that here there was no discernible difference between the classifiers in the overall accuracy of scoring. Additionally, thus far it seems like this thesis is one among few (if any) studies that has attempted to compare two different classification schemes using the same data set scored using AASM guidelines. This is significant because unless the same data is used, the superiority of one classification method over another—or lack thereof—can not definitively be determined.

5.3 Future Work

Should the pursuit of this subject be sufficiently extensive, it might be beneficial for the classifiers to be developed from scratch so as to provide the maximum amount of flexibility in design and control over functionality. What follows, though, are only suggestions for improvements or further developments given the methods in this thesis.

The SHHS sleep database was used for this study, but the data in the SHHS database were scored using the R&K rules. While there are other databases such as PhysioNet and Siesta that could be considered for automated sleep classifier research, these databases were also developed using the R&K scoring guidelines. At this time, there do not seem to be any free, public sleep databases available that are sufficiently large and scored according to AASM standards. Having such a database available for ongoing work would benefit the sleep research community and promote research outcomes that are compliant with the new AASM standards.

One possible improvement to the results would include an exhaustive search of all possible classifier parameters with all possible combinations of feature subsets per class. Due to the processing capabilities of the computer and software used in this work, performing an exhaustive search for optimal parameter and feature subset combinations would not have been feasible, but may offer some additional benefit. Other methods that could be used for feature space and subset searching might include techniques such as genetic algorithms. Genetic algorithms are stochastic iterative search methods which use coding, scoring, ranking, mutating, and iterating steps to mimic the "survival of the fittest" rule often cited in biological evolution, and has been shown to be successful in identifying data set-specific significant features sets [77, 83].

In addition to performing an exhaustive search, a search for the optimal number of features to be included for the best class separation should also be analyzed. Again, due to limited processing capabilities, not all feature subset sizes were evaluated. It would perhaps increase the accuracy of the classification if the feature subset used to characterize each class or class pair was tailored to each of the binary classifiers using a different numbers of feature subsets (e.g. perhaps only one feature—instead of five—may be needed to distinguish stage 3 sleep from any other stage). Similarly, the possibility exists that the performance of the classifiers was hurt by attempting to implement a large number of visual scoring rules, and perhaps better performance could be achieved if it were possible to determine only one distinctive feature per class. Some of the features do have overlap between

classes (like eye movements, which occur in both REM and W) and may be causing some confusion. Eliminating features that have overlap, or adding only distinctly unique class features, would likely increase the overall performance. Furthermore, a detailed analysis of the efficacy of the feature extraction techniques would provide a better indication as to which methods are sufficiently specific for identification of particular feature patterns.

This work used a series of 10 binary classifiers and a voting system to produce a single consensus output for both the NFS and SVM classification methods. Improvements to the SVM method used might mean implementation of a single multi-class SVM. Multi-class SVMs generally perform well, and can be constructed by combining multiple binary classifiers in a cascade, either in a serial fashion using one-against-one and voting methodology, or in a decision tree fashion using directed, acyclic graphs [84]. Improvements to the NFS classifier, then, could take the form of tuning through more careful selection and evaluation (via thorough training and validation) of the training parameters, such as the number of iterations and the error constraint, in an attempt to further mitigate potential overfitting of the ANFIS.

Also, there is the possibility that the tie-breaking methods used to break voting ties in class decisions ultimately resulted in suboptimal performance of the methods. Due to the difference in the information available from each classifier output, the tie-breaking methods were implemented accordingly, and the effect of those differences should be evaluated, both within and between classifiers.

After classification has been completed, there is another potential processing step that could be used to improve the outcome. From the technologist's handbook, there is a section that reads as follows: "This is an epoch that must be scored *in context*. This means that you have to know what came before and what comes after to be able to provide a score" [85]. The book is referring to epochs that were previously considered "movement time"—such as when a patient rolls over. Epochs that contain major body movements may add enough artifact to the EEG signals to render the stage unclassifiable by the features defined in the AASM manual. This means that there are some stages that are not able to be classified based on their distinctive features, but should be classified based on and in accordance with the stages that come before and after. Therefore, probabilistically and practically, an unidentified stage that comes between two similarly classified stages would also be that particular stage. The implication of this goes beyond merely reflecting the rigid criteria

of visual scoring when automated classification comes into the picture: if some of the stages do not contain sufficient information to be classified properly in an automated environment, it may be due to this requirement that the stage be put "in context" to attain the correct classification. The suggestion, then, would be to incorporate some post-classification "cleaning" of the detection output to attempt to review the data for misclassification due to "out of context errors." After classification is complete, the output could be run through some probability routine that looks for one-off (or single-occurance) detections that do not "fit" in a particular string of detections, and then change the final decision to disallow these rapid shifts to and from a stage that differs from those around it.

Lastly, comparison of classifiers using the same data sets provides a backdrop against which the classification methods can be on more equal footing when comparing and contrasting results. It is suggested that this type of "internal benchmarking" be continued in the pursuit of the development of a system that can adequately and reliably aid physicians and sleep lab technicians in the scoring of sleep.

Chapter 6

Summary and Conclusions

This section presents a summary of the thesis topic as well as conclusions and comments based on the results of this study.

For decades, the scoring of sleep stages has been considered essential for the understanding of sleep architecture, and this method continues to be used as a standard quantifier in sleep research [7, 12, 13]. Classic approaches to sleep stage classification involve sleep technicians using a manual scoring technique. Over the years, however—and in response to the issues that result from using standard approaches—researchers have been looking for advances that might improve the efficiency, reliability, and accuracy of the classification process [14, 1].

Computers have been used to develop automated sleep scoring techniques, which, in addition to the potential time savings, have the possibility of reducing the variability in scoring outcomes, and this has led to multiple attempts to advance automated sleep stage classifiers. The range of attempts at automation have exhibited some success [16], but continued work is needed to improve upon these classification schemes [17, 18]. This thesis explored the use of automated classification systems with the sleep stage definitions and scoring rules published by the American Academy of Sleep Medicine in 2007.

Specifically, the results presented in this thesis show that the use of NFS and SVM methods for classifying sleep stages is possible using the AASM guidelines published in 2007. While the current work does support and confirm the use of these classification techniques within the research community, the results did not indicate a significant difference in the accuracy of either approach—nor a difference in one over the other. The results did suggest that the important clinical stage 3 (slow wave sleep) can be accurately scored with classifiers such as those developed for this thesis; however, the techniques used here would need more investigation and optimization prior to serious

use in clinical applications.

Finally, the work in this thesis sought to contribute to the continued discussion and evaluation of automated sleep classification. Questions such as those asked here require still more attention due to the fact that the field of sleep research is still evolving and the use of computer-assisted sleep analysis will continue to gain momentum in the coming months and years.

Bibliography

[1] C. Iber, S. Ancoli-Israel, A. L. Chesson Jr., and S. F. Quan. *The AASM Manual for the Scoring of Sleep and Associated Events : Rules, Terminology and Technical Specifications*. American Academy of Sleep Medicine, Westchester, IL, 2007.

[2] P. Anderer, G. Gruber, S. Parapatics, M. Woertz, T. Miazhynskaia, G. Kloesch, B. Saletu, J. Zeitlhofer, M. J. Barbanoj, H. Danker-Hopfe, S. L. Himanen, B. Kemp, T. Penzel, M. Grozinger, D. Kunz, P. Rappelsberger, A. Schlogl, and G. Dorffner. An E-health solution for automatic sleep classification according to Rechtschaffen and Kales: validation study of the Somnolyzer 24 x 7 utilizing the Siesta database. *Neuropsychobiology*, 51(3):115–133, 2005.

[3] K. L. Priddy and P. E. Keller. *Artificial Neural Networks : An Introduction*. SPIE Press, Bellingham, WA, 2005.

[4] N. Sezgin and M. E. Tagluk. Energy based feature extraction for classification of sleep apnea syndrome. *Computers in Biology and Medicine*, 39(11):1043–1050, 2009.

[5] D. Nauck, F. Klawonn, and R. Kruse. *Foundations of Neuro-Fuzzy Systems*. John Wiley, Chichester, England ; New York, NY, 1997.

[6] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[7] A. Rechtschaffen and A. Kales, editors. *A Manual of Standardized Terminology, Techniques and Scoring System for Sleep Stages of Human Subjects*. U.S. Dept. of Health, Education, and Welfare, Bethesda, MD, 1968.

[8] MATLAB. *Student Version 7.7.0 (R2008b)*. The MathWorks Inc., Natick, Massachusetts, 2008.

[9] S. G. Mallat. *A wavelet tour of signal processing*. Academic Press, San Diego, 2nd edition, 1999.

[10] J-S. R. Jang. ANFIS: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man and Cybernetics*, 23(3):665–685, 1993.

[11] N. A. Collop, R. E. Salas, M. Delayo, and C. Gamaldo. Normal sleep and circadian processes. *Critical Care Clinics*, 24(3):449–60, 2008.

[12] M. H. Silber, S. Ancoli-Israel, M. H. Bonnet, S. Chokroverty, M. M. Grigg-Damberger, M. Hirshkowitz, S. Kapen, S. A. Keenan, M. H. Kryger, T. Penzel, M. R. Pressman, and C. Iber. The visual scoring of sleep in adults. *Journal Of Clinical Sleep Medicine*, 3(2): 121–131, 2007.

[13] M. H. Silber. Staging sleep. *Sleep Medicine Clinics*, 4(3):343–352, 2009.

[14] H. Danker-Hopfe, D. Kunz, G. Gruber, G. Kloesch, J. L. Lorenzo, S. L. Himanen, B. Kemp, T. Penzel, G. Kloschke, H. Dorn, A. Schlogl, E. Trenker, and G. Dorffner. Interrater reliability between scorers from eight European sleep laboratories in subjects with different sleep disorders. *Journal of Sleep Research*, 13(1):63–69, 2004.

[15] T. Penzel and R. Conradt. Computer based sleep recording and analysis. *Sleep Medicine Reviews*, 4(2):131–148, 2000.

[16] B. Ahmed and R. Tafreshi. Advances in automatic sleep analysis. In *IFMBE Proceedings: 13th International Conference on Biomedical Engineering*, pages 422–426. Springer-Verlag, Berlin, 2009.

[17] T. Penzel, M. Glos, C. Schobel, and I. Fietze. Sleep recording and sleep analysis methods. In *World Congress on Medical Physics and Biomedical Engineering*, pages 431–433. Springer, Berlin, 2009.

[18] G. Stege, P.J. Vos, P.N. Dekhuijzen, P.H. Hilkens, M. J. van de Ven, Y.F. Heijdra, and F.J. van den Elshout. Manual vs. automated analysis of polysomnographic recordings in patients with chronic obstructive pulmonary disease. *Sleep and Breathing*, pages 1–7, 2012.

[19] L. Parrino, R. Ferri, M. Zucconi, and F. Fanfulla. Commentary from the Italian Association of Sleep Medicine on the AASM manual for the scoring of sleep and associated events: for debate and discussion. *Sleep Medicine*, 10(7):799–808, 2009.

[20] H. Schulz. Rethinking sleep analysis: Comment on the AASM manual for the scoring of sleep and associated events. *Journal Of Clinical Sleep Medicine*, 4:99–103, 2008.

[21] D. Álvarez-Estévez, J. M. Fernández-Pastoriza, E. Hernández-Pereira, and V. Moret-Bonillo. A method for the automatic analysis of the sleep macrostructure in continuum. *Expert Systems with Applications*, 40(5):1796–1803, 2013.

[22] M. M. Grigg-Damberger. The AASM scoring manual four years later. *Journal of Clinical Sleep Medicine*, 8(3):323–332, 2012.

[23] C. C. Varghese. *Sleep stage classification using neuro-fuzzy intelligence*. PhD thesis, University of Texas at El Paso, 2005.

[24] S. Crisler, M. A. Anch, M. Morrissey, and D. Barnett. Automated sleep-stage scoring using a support vector machine. *Sleep*, 27(Abstract Supplement):A368, 2004.

[25] D. Popovic. *Automatic staging of sleep using only two electrodes on the forehead*. PhD thesis, University of Southern California, United States – California, 2011.

[26] C. Berthomier and M. Brandewinder. Sleep scoring: man vs. machine? *Sleep and Breathing*, pages 1–2, May 2012.

[27] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[28] D. Alvarez-Estevez and V. Moret-Bonillo. Model comparison for the detection of EEG arousals in sleep apnea patients. In M. J. Cabestany, F. Sandoval, A. Prieto, and J.M. Corchado, editors, *Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part I: Bio-Inspired Systems: Computational and Ambient Intelligence*, pages 997–1004. Springer-Verlag, Salamanca, Spain, 2009.

[29] B. J. Fisch and R. Spehlmann. *Fisch and Spehlmann's EEG Primer : Basic Principles of Digital and Analog EEG*. Elsevier, Amsterdam ; New York, 1999.

[30] K. E. Bloch. Polysomnography: a systematic review. *Technology and Health Care*, 5:285–305, 1997.

[31] T. Penzel, M. Hirshkowitz, J. Harsh, R. D. Chervin, N. Butkov, M. Kryger, B. Malow, M. V. Vitiello, M. H. Silber, C. A. Kushida, and A. L. Chesson Jr. Digital analysis and technical specifications. *Journal Of Clinical Sleep Medicine*, 3(2):109–120, 2007.

[32] Ian G. Campbell. EEG recording and analysis for sleep research. *Current Protocols in Neuroscience*, 49(10.2):1–19, 2009.

[33] K. A. Carden. Recording sleep: The electrodes, 10/20 recording system, and sleep system specifications. *Sleep Medicine Clinics*, 4(3):333–341, 2009.

[34] I. M. Colrain. Sleep and the brain. *Neuropsychology Review*, 21(1):1–4, 2011.

[35] J. Matheson, R. Singh, and A. Packard. Polysomnography and sleep disorders. In A. S. Blum and S. V. Rutkove, editors, *The Clinical Neurophysiology Primer*, pages 393–445. Humana Press Inc., Totowa, NJ, 2007.

[36] S. C. Mednick and W. A. Alaynick. Comparing models of sleep-dependent memory consolidation. *Journal of Experimental & Clinical Medicine*, 2(4):156–164, 2010.

[37] S. Redline, H.L. Kirchner, S.F. Quan, D.J. Gottlieb, V. Kapur, and A. Newman. The effects of age, sex, ethnicity, and sleep-disordered breathing on sleep architecture. *Archives of Internal Medicine*, 164:406–418, 2004.

[38] M. E. Tagluk, N. Sezgin, and M. Akin. Estimation of sleep stages by an artificial neural network employing EEG, EMG and EOG. *Journal of Medical Systems*, 34(4):717–725, 2010.

[39] J. Caffarel, G. J. Gibson, J. P. Harrison, C. J. Griffiths, and M. J. Drinna. Comparison of manual sleep staging with automated neural network-based analysis in clinical practice. *Medical and Biological Engineering and Computing*, 44(1-2):105–110, 2006.

[40] R. K. Sinha. Artificial neural network and wavelet based automated detection of sleep spindles, rem sleep and wake states. *Journal of Medical Systems*, 32(4):291–299, 2008.

[41] F. Ebrahimi, M. Mikaeili, E. Estrada, and H. Nazeran. Automatic sleep stage classification based on EEG signals by using neural networks and wavelet packet coefficients. In *Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1151–1154. IEEE, Vancouver, British Columbia, Canada, 2008.

[42] C.M. Held, J.E. Heiss, P.A. Estévez, C.A. Perez, M. Garrido, C. Algarín, and P. Peirano. Extracting fuzzy rules from polysomnographic recordings for infant sleep classification. *IEEE Transactions on Bio-Medical Engineering*, 53(10):1954–62, 2006.

[43] C. F. Chao, J. A. Jiang, M. J. Chiu, and R. G. Lee. Automated long-term polysomnography analysis with wavelet processing and adaptive fuzzy clustering. *Biomedical Engineering - Applications, Basis and Communications*, 18(3):119–123, 2006.

[44] S. Gudmundsson, T. P. Runarsson, and S. Sigurdsson. Automatic sleep staging using support vector machines with posterior probability estimates. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, volume 2, pages 366–372. IEEE Computer Society, Washington, DC, USA, 2005.

[45] Y. H. Lee, Y. S. Chen, and L. F. Chen. Automated sleep staging using single EEG channel for REM sleep deprivation. In *Ninth IEEE International Conference on Bioinformatics and Bioengineering, 2009*, pages 439–442, 2009.

[46] S. Crisler, M. J. Morrissey, A. M. Anch, and D. W. Barnett. Sleep-stage scoring in the rat using a support vector machine. *Journal of Neuroscience Methods*, 168:524–534, 2008.

[47] R. Cruz-Cano. *Creation And Optimization Of Fuzzy Inference Neural Net-Works*. PhD thesis, University of Texas at El Paso, El Paso, Texas, 2005.

[48] H. T. Nguyen and E. Walker. *A First Course in Fuzzy Logic*. Chapman and Hall, Boca Raton, FL, 2000.

[49] D. D. Nauck and A. Nürnberger. Neuro-fuzzy systems: A short historical review. In Christian Moewes and Andreas Nürnberger, editors, *Computational Intelligence in Intelligent Data Analysis*, number 445 in Studies in Computational Intelligence, pages 91–109. Springer Berlin Heidelberg, 2013.

[50] N. Khasawneh, M. A. K. Jaradat, L. Fraiwan, and M. Al-Fandi. Adaptive neuro-fuzzy inference system for automatic sleep multistage level scoring employing EEG, EOG, and EMG extracted features. *Applied Artificial Intelligence*, 25(2):163–179, 2011.

[51] V. Kecman. *Learning and Soft Computing : Support Vector Machines, Neural Networks, and Fuzzy Logic Models*. MIT Press, Cambridge, MA, 2001.

[52] A. Mammone, M. Turchi, and N. Cristianini. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):283–289, 2009.

[53] S. F. Quan, B. V. Howard, C. Iber, J. P. Kiley, F. J. Nieto, G. T. O'Connor, D. M. Rapoport, S. Redline, J. Robbins, J. M. Samet, and P. W. Wahl. The Sleep Heart Health Study: design, rationale, and methods. *Sleep*, 20(12):1077–1085, 1997.

[54] P. S. Jensen, H. B. D. Sorensen, H. L. Leonthin, and P. Jennum. Automatic sleep scoring in normals and in individuals with neurodegenerative disorders according to new international sleep scoring criteria. *Journal of Clinical Neurophysiology*, 27(4):296–302, 2010.

[55] M. Bresler, K. Sheffy, G. Pillar, M. Preiszler, and S. Herscovici. Differentiating between light and deep sleep stages using an ambulatory device based on peripheral arterial tonometry. *Physiological Measurement*, 29(5):571–584, 2008.

[56] J. H. Choi, E. J. Kim, Y. S. Kim, J. Choi, T. H. Kim, S. Y. Kwon, H. M. Lee, S. H. Lee, C. Shin, and S. H. Lee. Validation study of portable device for the diagnosis of obstructive sleep apnea according to the new AASM scoring criteria: Watch-PAT 100. *Acta Oto-Laryngologica*, 130 (7):838–843, 2010.

[57] F. Chung, P. Liao, Y. Sun, B. Amirshahi, H. Fazel, C. Shapiro, and H. Elsaid. Perioperative practical experiences in using a level 2 portable polysomnography. *Sleep and Breathing*, 15 (3):367–375, 2011.

[58] A. Varri, B. Kemp, T. Penzel, and A. Schlogl. Standards for biomedical signal databases. *IEEE Engineering in Medicine and Biology Magazine*, 20(3):33–37, 2001.

[59] B. Kemp, A. Varri, A. C. Rosa, K. D. Nielsen, and J. Gade. A simple format for exchange of digitized polygraphic recordings. *Electroencephalography and Clinical Neurophysiology*, 82 (5):391–393, 1992.

[60] E. Oropesa, H. L. Cycon, and M. Jobert. Sleep stage classification using wavelet transform and neural network. Technical Report TR-99-008, International Computer Science Institute (ICSI), 1999.

[61] A.H. Khandoker, J. Gubbi, and M. Palaniswami. Automated scoring of obstructive sleep apnea and hypopnea events using short-term electrocardiogram recordings. *IEEE Transactions on Information Technology in Biomedicine*, 13(6):1057–1067, 2009.

[62] C. Hsu, C. Chang, and C. Lin. *A practical guide to support vector classification*, 2010. URL `http://citeseerx.ist.psu.edu`.

[63] M. Y. V. Min. *An EEG based study of unintentional sleep onset*. PhD thesis, National University of Singapore, 2008.

[64] M. Sugeno and T. Takagi. Multi-dimensional fuzzy reasoning. *Fuzzy Sets and Systems*, 9: 313–325, 1983.

[65] S. H. Huang and H. Xing. Extract intelligible and concise fuzzy rules from neural networks. *Fuzzy Sets and Systems*, 132(2):233–243, 2002.

[66] N. Cristianini and J. Shawe-Taylor. Support vector and kernel methods. In Michael Berthold and David J. Hand, editors, *Intelligent data analysis - An Introduction*, pages 169–197. Springer-Verlag New York, Inc., New York, NY, USA, 2003.

[67] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi. A review of classification algorithms for EEG-based brain–computer interfaces. *Journal of Neural Engineering*, 4(2): 1–24, 2007.

[68] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Books24x7.com, Norwood, MA, 2000.

[69] J. Acevedo, S. Maldonado, P. Siegmann, S. Lafuente, and P. Gil. Multi-class support vector machines based on arranged decision graphs and particle swarm optimization for model selection. In Bartlomiej Beliczynski, Andrzej Dzielinski, Marcin Iwanowski, and Bernardete Ribeiro, editors, *Adaptive and Natural Computing Algorithms*, volume 4432 of *Lecture Notes in Computer Science*, pages 238–245. Springer Berlin / Heidelberg, 2007.

[70] T. Coleman and Y. Li. A reflective Newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058, 1996.

[71] I. H. Witten. *Data mining*. Morgan Kaufmann, Burlington, MA, 2011.

[72] N. V. Chawla, N. Japkowicz, and A. Kotcz. Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explorations*, 6(1):1–6, 2004.

[73] A. Lewicke, E. Sazonov, M. J. Corwin, M. Neuman, and S. Schuckers. Sleep versus wake classification from heart rate variability using computational intelligence: Consideration of rejection in classification models. *IEEE Transactions on Biomedical Engineering*, 55(1):108–118, 2008.

[74] H. P. Van Dongen, K. M. Vitellaro, and D. F. Dinges. Individual differences in adult human sleep and wakefulness: Leitmotif for a research agenda. *Sleep*, 28(4):479–496, 2005.

[75] A. M. Tucker, D. F. Dinges, and H. P. A.Van Dongen. Trait interindividual differences in the sleep physiology of healthy young adults. *Journal of Sleep Research*, 16(2):170–180, 2007.

[76] A. Lewandowski, R. Rosipal, and G. Dorffner. Extracting more information from EEG recordings for a better description of sleep. *Computer Methods and Programs in Biomedicine*, 108 (3):961–972, 2012.

[77] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. Wiley, New York, 2nd ed edition, 2001.

[78] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H-T. Lin. *Learning from data: a short course*. AMLBook.com, United States, 2012.

[79] A. Al-Hmouz, J. Shen, J. Yan, and R. Al-Hmouz. Modeling mobile learning system using ANFIS. In *2011 11th IEEE International Conference on Advanced Learning Technologies (ICALT)*, pages 378–380, 2011.

[80] M. Ronzhina, O. Janousek, J. Kolárová, M. Nováková, P. Honzík, and I. Provazník. Sleep scoring using artificial neural networks. *Sleep Medicine Reviews*, 16(3):251–263, 2012.

[81] E. Tasali, R. Leproult, D. A. Ehrmann, and E. V. Cauter. Slow-wave sleep and the risk of type 2 diabetes in humans. *Proceedings of the National Academy of Sciences*, 105(3):1044–1049, 2008.

[82] J. Yordanova, V. Kolev, U. Wagner, J. Born, and R. Verleger. Increased alpha (8-12 hz) activity during slow wave sleep as a marker for the transition from implicit knowledge to explicit insight. *Journal of Cognitive Neuroscience*, 24(1):119–132, 2012.

[83] D. L. Tong and R. Mintram. Genetic algorithm-neural network (GANN): a study of neural network activation functions and depth of genetic algorithm search applied to feature selection. *International Journal of Machine Learning and Cybernetics*, 1(1-4):75–87, 2010.

[84] E. Tagliazucchi, F. V. Wegner, A. Morzelewski, S. Borisov, K. Jahnke, and H. Laufs. Automatic sleep staging using fMRI functional connectivity data. *NeuroImage*, 63(1):63–72, 2012.

[85] American Academy of Sleep Medicine. *A technologist's handbook for understanding and implementing the AASM manual for the scoring of sleep and associated events : rules, terminology and technical specifications.* American Academy of Sleep Medicine, Westchester, IL, 2009.