FACULTY OF ENGINEERING OF THE UNIVERSITY OF PORTO

Development of High-Density Surface Electromyography Signal Processing and Machine Learning Algorithms for Clinical Use

Francisco Samuel Neves Fidalgo

A dissertation submitted for the degree of
Master in Biomedical Engineering

Supervisor: Miguel Velhote Correia, PhD

October 2, 2024

© Francisco Samuel Neves Fidalgo, 2024

Development of High-Density Surface Electromyography Signal Processing and Machine Learning Algorithms for Clinical Use

Francisco Samuel Neves Fidalgo

A dissertation submitted for the degree of
Master in Biomedical Engineering

October 2, 2024

Abstract

This dissertation presents the development of advanced High-Density Surface Electromyography signal processing techniques and machine learning models, designed to enhance clinical applications. The research addresses multiple aspects of muscle dynamics analysis, including the implementation of novel algorithms for signal pre-processing, amplitude and spectral analysis, and timing metrics. Other key contributions include principal component-based innervation zone estimation, cross-correlation-based and spike-triggered-averaging-based conduction velocity estimations, as well motor unit decomposition through blind source separation methods. To validate these approaches, beyond synthetic signals – which, when simpler, showed high robustness of the algorithms, and for decomposition showed agreement rates of over 90% –, real data from patients with Parkinson's Disease and patients with Duchenne muscular dystrophy was adopted, which successfully showed the algorithms' ability to distinguish between healthy individuals and those with the neuromuscular conditions. Beyond signal analysis, this was also pursed through machine learning, following a pipeline that facilitates in the detection of different events during gait. Particularly in determining when freezing of gait occurred, the models were able to reach weighted accuracies of 98% on unseen individuals. A digital platform was also created to simplify HDsEMG data processing and model training, using convolutional neural networks and spectral images to be able to classify between seven-gesture and dystrophy-or-healthy, to accuracies as high as 96% on test data. Overall, these results hold significant implications for clinical applications, providing insights into muscle behavior and potential for better therapeutic interventions

Acknowledgments

I'd like to thank Professor Miguel Correia, who, having accepted me as his supervisee, proved to be an essential accompanying factor, always able to swiftly extinguish my doubts and concerns, whether by explaining some concepts more intricately yet concisely, whether by guiding me toward a more structured and well-adjusted dissertation.

My sincerest appreciation to INESC TEC, for allowing me to pursue a project of undeniable relevance, and for the resources and logistical support provided throughout the completion of this work.

To my MBE Colleagues, who were there to show that I was not alone in this – except for when I was alone, which was not very often – except for when it was very often. Their willingness to discuss about our works ignited ideas in my mind that perhaps would not have been herein explored otherwise.

To my Family and Friends who, being my family and my friends, are people who have led me to where I am at this very, exact, precise, particular, definite, specific moment in my life.

And lastly, I must thank the most prevalent of players: the cats I often see outside my bedroom window (a black-and-white piebald, whom I've come to call MegaTux, can be observed right as I'm typing this) — their playfulness and *insouciant* attitude coupled with their unabashed sense of serenity proved to be ataractic to me, and one *could* argue that these critters were the driving force behind this work.

"The only man who never makes a mistake is the man who never does anything."

Theodore Roosevelt

Paraphrased by Frank Heffley

Contents

List of Figures ix

List of Tables xi

List of Abbreviations xii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Objectives . . . 2
- 1.3 Expected Results . . . 2
- 1.4 Document Structure . . . 3

2 Electromyography Analysis 4
- 2.1 Electromyography . . . 4
  - 2.1.1 Brief History of EMG . . . 4
  - 2.1.2 Physiology and Generation of EMG . . . 5
  - 2.1.3 Characteristics of Motor Units . . . 6
  - 2.1.4 Surface EMG . . . 8
  - 2.1.5 Clinical Cases and Pathologies . . . 8
- 2.2 Amplitude Analysis . . . 10
  - 2.2.1 Root Mean Square . . . 11
  - 2.2.2 Average Rectified Value . . . 11
  - 2.2.3 Topographical Mapping . . . 13
  - 2.2.4 Timing Analysis and Onset Detection . . . 14
- 2.3 Spectral Analysis . . . 16
  - 2.3.1 Fourier Transform . . . 16
  - 2.3.2 Wavelet Transform . . . 17
- 2.4 Conduction Velocity . . . 17
- 2.5 Motor Unit Decomposition . . . 20

3 Review of the Literature 23
- 3.1 Machine Learning Algorithms used with HDsEMG . . . 24
  - 3.1.1 Advancements in Gesture and Other Human Activity Recognition . . . 24
  - 3.1.2 Detection, diagnosis, monitoring, prognosis . . . 29
  - 3.1.3 Metric Extraction . . . 30
- 3.2 Commercially and Non-Commercially Available Software Options . . . 30

4 Methodology **32**
4.1 Validation and Testing Approaches . . . 32
4.1.1 Simulated Signals . . . 32
4.1.2 Benchmark Dataset of Real Signals . . . 35
4.2 Pre-Processing Tools . . . 38
4.3 Amplitude Analysis . . . 43
4.3.1 Features . . . 43
4.3.2 Topographical Mapping . . . 44
4.3.3 Center of Gravity . . . 45
4.4 Timing Analysis . . . 46
4.5 Spectral Analysis . . . 49
4.5.1 Frequency Domain . . . 49
4.5.2 Time-Frequency Domain . . . 49
4.6 Conduction Velocity Estimation . . . 51
4.7 Innervation Zone Estimation . . . 54
4.8 Motor Unit Decomposition . . . 56
4.9 Gait Monitoring . . . 59
4.9.1 Dataset: Freezing of Gait in Parkinson's Disease . . . 59
4.9.2 Pre-Processing and Filtering . . . 61
4.9.3 Segmentation . . . 61
4.9.4 Resampling . . . 62
4.9.5 Feature Extraction . . . 62
4.9.6 Feature Selection . . . 64
4.9.7 Classification Model Evaluation . . . 65
4.10 Deep Learning-Based Gesture and Health Condition Classification . . . 66
4.10.1 Database Platform . . . 68
4.10.2 Dataset: Healthy and Duchenne Muscular Dystrophy Patients . . . 69
4.10.3 Pre-Processing and Filtering . . . 70
4.10.4 Downsampling . . . 70
4.10.5 Segmentation . . . 70
4.10.6 Label Distribution Adjustment . . . 72
4.10.7 Spectral-Based Processing . . . 72
4.10.8 Neural Network Architectures . . . 74
4.10.9 Classification Metrics and Evaluation . . . 74

5 Results and Discussion **77**
5.1 Pre-Processing Tools . . . 77
5.2 Amplitude Analysis . . . 84
5.3 Spectral Analysis . . . 92
5.4 Timing Analysis . . . 96
5.5 Conduction Velocity Estimation . . . 99
5.5.1 XC Algorithm . . . 99
5.5.2 STA Algorithm . . . 102
5.5.3 Comparison between Algorithms . . . 105
5.6 Innervation Zone Estimation . . . 108
5.7 Motor Unit Decomposition . . . 111
5.8 Gait Monitoring . . . 118
5.8.1 Feature Selection by Channel . . . 118
5.8.2 Feature Selection by Domain . . . 119

5.8.3 Feature Selection by Score 120
5.8.4 Final Assessment of Model Evaluation 121
5.9 Deep Learning-Based Gesture and Health Condition Classification 125
5.9.1 HDsEMG Images as Input Data 125
5.9.2 Spectral Images as Input Data 126
5.9.3 Final Assessment of Model Evaluation 128

6 Conclusions **131**

List of Figures

2.1 Diagram of the spinal cord. . . . 5
2.2 Representation EMG signal of a patient upon a targeted muscle reinnervation procedure and Motor Unit Spike Trains. . . . 7
2.3 RMS envelopes from patients with neuropathic, myopathic, and healthy conditions. 11
2.4 Diagram of the ARV EMG and Peak EMG processing methods. . . . 12
2.5 RMS maps of a stroke subject for five force levels, during an isometric non-fatiguing activity. . . . 13
2.6 Activity maps and CoG distance by time graphs, upon moderate isokinetic knee extension. . . . 14
2.7 Activation intervals of gait cycles, detected with EMG signals captured at the gastrocnemius lateralis. . . . 15
2.8 Block diagram of the HHT process. . . . 18
2.9 Single differential multichannel sEMG signals detected from the vastus medialis muscle upon a cycling exercise, with dashed lines indicating the propagation of APs. 18
2.10 Diagram displaying the concept of EMG MU decomposition. . . . 20
2.11 Block diagram of the FastICA MU decomposition technique for HDsEMG. . . . 21
2.12 Discharge times for MUs across 4-week strength-training intervention. . . . 22

3.1 HDsEMG activity maps for different gestures upon each of Tam et al.'s proposed pre-processing steps. . . . 27

4.1 Example of the signal's characteristics describing a simulated action potential. . . . 33
4.2 Process describing approach for synthetic signal generation and visualization, and acquisition of MU firing rates, signal intensity, and AP shapes. . . . 34
4.3 Display of differently simulated 2D signals. . . . 35
4.4 Display of one simulated action potential propagating through a column grid, and its effects upon three different (a) IZs, (b) SNRs, and (c) decay rates. . . . 36
4.5 Images of the FOG dataset's (a) leg muscle anatomy and (b) correct electrode placement on the surface of skin and other equipment placement. . . . 37
4.6 Entire timeline of the FOG dataset's signals (EMG1, EMG2, EMG4) acquired from subject 10 during task 2. . . . 37
4.7 Entire timeline of the FOG dataset's signal (EMG1) acquired from subjects 10 and 11 during all 4 tasks. . . . 38
4.8 Signal acquisition setup for the DMD gesture dataset. . . . 39
4.9 Display of HDsEMG signals acquired by the DMD and healthy patients during wrist flexion gestures. . . . 39
4.10 Pipeline for standard pre-processing of sEMG signals with the tools implemented in this work. . . . 41

4.11 Comparison of original EMG signal and its TKEO result. 48
4.12 Thresholding methods with and without TKEO or max TKEO. 48
4.13 STA Algorithm approach. 52
4.14 Diagram of the application of PCA on a 2D EMG signal array. 55
4.15 Diagram of the pipeline adopted for developing the gait monitoring's FOG classification model. 59
4.16 Process adopted for creating and managing a HDsEMG signal database. 67
4.17 Screen-capture of the database platform's user interface developed in HTML, from which to upload one's sEMG signals. 68
4.18 Diagram displaying the periods of each gesture from the DMD dataset. 71
4.19 Step-by-step process adopted for extracting spectral information from HDsEMG segments in time, and resulting DL input shapes. 73

5.1 Detrending of FOG signal. 78
5.2 Butterworth 15-499Hz bandpass filter applied to FOG signal, and its frequency response and FFT spectrum. 79
5.3 Notch filter at 50Hz applied to FOG signal, and its frequency response and FFT spectrum. 79
5.4 Minmax Normalization, from -1 to 1, applied to FOG signal, and its full-wave rectification. 80
5.5 All three FOG EMG signals and their FFT spectra, after applying two bandpass filters. 81
5.6 Power Spectral Density, through Welch's Method, of the original FOG signal, and of the signal after applying two different bandpass filters. 82
5.7 Comparison of entire signals' timeline after two different bandpass filters. 83
5.8 Pre-processing pipeline as applied to channel 0 of two patients (dystrophic and healthy) during wrist extension and flexion, from the DMD dataset. 85
5.9 Amplitude features on a specific interval, on 1000-sample epoch window, of subject 1 of FOG dataset, and bar plots of average values. 86
5.10 Differentiation features on a specific interval, on 1000-sample epoch window, of subject 1 of FOG dataset, and bar plots of average values. 87
5.11 Amplitude and differentiation features on entire timeline, with 3000-sample epochs, for two different bandpass intervals. 88
5.12 Bar plot summarizing resulting FOG dataset's amplitude averages. 89
5.13 Topographical mapping on DMD dataset, at three different gesture repetitions. . . 90
5.14 Center of Gravity in x and y axis on entire timeline. 91
5.15 Spectral coherence, on FOG signals of patient 1, task 1, for bandpass 15-499Hz. . 93
5.16 Spectral coherence, on FOG signals of patient 1, task 1, for bandpass 100-499Hz. 93
5.17 Time-frequency spectra for FOG patient 1 task 1, on signal EMG1, given the two bandpasses. 94
5.18 Power spectral density of raw signal for DMD and Healthy patients, during flexion. 95
5.19 Time-Frequency Analysis On All Repetitions, healthy patient, wrist flexion, with CWT, STFT, HHT. 96
5.20 Time-Frequency Analysis through time, of 2D FFT, of CWT and STFT spectra from a 500-sample window, at three different repetitions. 97
5.21 Hilbert-Huang Transform's Intrinsic Mode Functions through time, from a 500-sample window, at three different repetitions. 98
5.22 XC algorithm results on different configurations of the simulated HDsEMG signals. 100
5.23 XC algorithm results on DMD dataset 101

5.24 Bar plots of the percent error results of STA-based algorithm upon different timewindows for estimation, and different simulated signals of MUSTs with different CVs. 103
5.25 Bar plots of the percent error results of STA-based algorithm, on 2m/s-simulated signals. . . . 104
5.26 STA-based CV estimation results on DMD dataset. . . . 106
5.27 Bar plot of the averages of the STA CVs, on DMD dataset; error bars corresponding to STD of each average. . . . 107
5.28 Bland-Altman plots from the comparison of CV results. . . . 107
5.29 Duration of XC algorithm on different sample sizes. . . . 108
5.30 Duration of STA-based algorithm on different time windows, at 2048Hz. . . . 108
5.31 Interpolated plot of the 2nd PC coefficients on simulated signals with IZ at different rows, on simpler signals. . . . 109
5.32 Plots of the 2nd PC coefficients on more complex signals. . . . 110
5.33 Average IZ accuracy per closeness of IZ to center row. . . . 110
5.34 MU Decomposition algorithm duration. . . . 111
5.35 Comparison of synthetic signals (GT) with its corresponding decomposition (algo), for 5 MU-generated signals of lower firing rates. . . . 112
5.36 Comparisons of spike trains between Algo and GT, for three different conditions. 112
5.37 Comparisons of firing rate, signal strengths and MUAP shapes on three different conditions. . . . 116
5.38 Results of the MU decomposition algorithm on DMD dataset. . . . 117
5.39 Bar chart of the accuracy of the Random Forest classifier, during score-based feature selection. . . . 120
5.40 Predictions throughout entire timelines of all 4 tasks, on both subject 10 and 11. . 122
5.41 Normalized confusion matrices of FOG state prediction, on all tasks and the two test subjects, for different step sizes. . . . 122
5.42 Correction of FOG detection mismatches through tolerance adjustment. . . . 124
5.43 Training and validation results for different architectures, given HDsEMG as input, upon DMD gesture and condition classification. . . . 125
5.44 Training and validation results for branched CNN+LSTM architecture, given different inputs, upon DMD gesture and condition classification. . . . 127

List of Tables

3.1 Some of the most commonly used "manually"-extracted features, in the time and frequency domains, for traditional machine learning-based EMG classifications. . 25

4.1 Summary of amplitude features computed at every new time epoch. 44
4.2 Implemented BioSPPy's EMG onset detection methods. 47
4.3 Summary of frequency domain analysis functions. 49
4.4 Summary of time-frequency domain analysis functions. 50
4.5 Features considered for training the machine learning model, with quantities obtained under the time domain, frequency domain and fractional Fourier transform domain. 62
4.6 Distribution of the FOG dataset's labels after segmentation; (FOGn, Tm) corresponds to the combination of FOG state n and task m. 65
4.7 Summary of Neural Network Architectures used for pre-processed HDsEMG inputs. 75
4.8 Summary of Neural Network Architectures used for spectral-based HDsEMG inputs. 75

5.1 Summary of CoG during Wrist Flexion and Extension, in Healthy (patient 2) and DMD (patient 3) Conditions. 91
5.2 Resulting average PSD before and after pre-processing (two different bandpasses), as computed by two different PSD methods (traditional and Welch's method) . . 92
5.3 Time deviations, and mean error, of each onset/offset detection method. 96
5.4 Overall averages of CV results of Extension and Flexion for Healthy and DMD conditions, during performing and at rest, and ratio between the two moments, for XC-based algorithm. 102
5.5 Summarized rate of agreement and matching firings results of the algorithm, averaged across the 1-MU, 5-MU (high FR and low FR), 10-MU and 20-MU simulated signals, over 5 distinct tolerances. 115
5.6 Rate of agreement and matching firings results of the algorithm, for two sets of 5-MU simulated signals; one of higher FR, the other of lower FR. 115
5.7 Impact on model accuracy upon using different window lengths (from 1 second to 4 seconds), with step size of 300 samples, with task feature known ("+1"). 118
5.8 Results of model accuracy upon using different combinations of the four channels, with task feature known ("+1"). 119
5.9 Results of model accuracy upon using different combinations of the three domains, with task feature known ("+1"). 119
5.10 Results of model accuracy upon using different models and numbers of features, with features selected by the Chi-squared, ANOVA F-value and Mutual Information scoring techniques, with task feature known. 120

5.11 Results of model accuracy upon classification of different aspects of PD patients' gait. 121
5.12 Summary of results on unseen test subjects (10 and 11) over different step sizes, and corresponding distribution of observation numbers. 123
5.13 Summary of final accuracy and loss results over 500 epochs for different architectures, given HDsEMG as input. 127
5.14 Summary of final accuracy and loss results for branched CNN+LSTM architecture, given different inputs (HDsEMG, Spectral, HDsEMG+Spectral). 128

List of Abbreviations

ALS	Amyotrophic lateral sclerosis
AP	Action potential
ARV	Average Rectified Value
BSS	Blind source separation
CKC	Convolution Kernel Compensation
CNN	Convolutional Neural Network
CNS	Central nervous system
CoG	Center of gravity
CV	Conduction velocity
CWT	Continuous Wavelet Transform
DL	Deep Learning
DMD	Duchenne Muscular Dystrophy
DWT	Discrete Wavelet Transform
EMG	Electromyography
FFT	Fast Fourier Transform
FOG	Freezing of Gait
HDsEMG	High-density surface electromyography
ICA	Independent Component Analysis
IED	Inter-electrode distance
iEMG	Intramuscular electromyography
LDA	Linear discriminant analysis
LSTM	Long Short-Term Memory
MAV	Mean absolute value
MDF	Median frequency
MFCV	Muscle fiber conduction velocity
ML	Machine Learning
MLE	Maximum-likelihood estimation
MNF	Mean frequency
MSE	Mean squared error
MU	Motor unit
MUAP	Motor unit action potential
MUCV	Motor Unit Conduction Velocity

MUST	Motor unit spike train
MVC	Maximum voluntary contraction
nEMG	Needle electromyography
PCA	Principal Component Analysis
PD	Parkinson's Disease
PNS	Peripheral nervous system
sEMG	Surface electromyography
SNR	Signal-to-noise ratio
STA	Spike-triggered averaging
STFT	Short-time Fourier Transform
SVM	Support vector machines
TF	Time-frequency
TMS	Transcranial magnetic stimulation

Chapter 1

Introduction

1.1 Motivation

In the last few decades, with the emergence of wearable technology, so too came the rising use of high-density microelectrode arrays, built into textile interfaces, capable of capturing electromyographic signals at the surface of the skin, without the need for invasive intervention.

Despite its clear advantages in non-invasiveness, muscle activity mapping ability in time and space, means for estimating muscle fiber properties, and the array of decomposition techniques at its disposal for motor unit function assessment, high-density surface electromyography (HDsEMG) is still a subject whose research has proven to involve a lot of unexplored potential; and said potential has risen even more so with the progress found in the artificial intelligence scope, wherein, through machine learning algorithms, researchers can extract valuable insights from the vast amount of data gathered (from classifying motions, to detecting, monitoring and even diagnosing muscle anomalies and disease progression).

Because all these metrics often require complex software tools in order to be extracted and interpreted, and seeking to maximize the impact of HDsEMG technology, collaboration between software developers and hardware manufacturers is imperative. By working closely with hardware producers, within the context of the *Health from Portugal* project[1], this work aims to create a cohesive framework whereby a software platform—within which the algorithms can be run—seamlessly integrates with the produced HDsEMG hardware. This collaboration ensures optimal performance, compatibility, and user-friendly experiences for practitioners relying on this integrated solution, which can sometimes be hard to find in many HDsEMG hardware-software products.

The integration of HDsEMG technology, alongside its intricate analytical tools and the novel machine learning techniques, in clinical settings has the potential to revolutionize diagnostics and treatment planning in fields such as rehabilitation, neurology, and neuromuscular and sports medicine. A set of robust algorithms that can facilitate the study of HDsEMG and aid in analyses will empower healthcare professionals to leverage the full diagnostic capabilities of this technology, ultimately improving patient outcomes.

1.2 Objectives

To aid a major portuguese biosignals-monitoring tech company in the production of a HDsEMG product, this work seeks to build and develop a reliable set of algorithms, which, within a digital platform, are capable of receiving, processing and displaying the HDsEMG signal acquired, providing tools for amplitude and timing analyses, spectral analysis, mapping activity and visualization, conduction velocity and innervation zone estimations, motor unit decomposition, and extraction of many other metrics.

Additionally, following a streamlined framework, and using machine learning-based approaches, one should be able to detect and monitor a person's gait or even any muscle anomaly and be reassured by the accuracy of these medical assessments, all in a rapid, non-invasive manner – such is the high-density surface electrode array's purpose, when placed on a specific region of interest.

In order to attain the general objective, the following specific objectives ought to be accomplished:

- Identification of the most relevant EMG metrics and prominent processing techniques for their extraction
- Implementation of these identified processing techniques
- Analysis of accuracy and reliability of the methods employed – validated and tested with simulated and real data –, upon acquisition of relevant datasets
- Building and optimization of machine learning models for practical applications in clinical assessments

1.3 Expected Results

The work explored within this dissertation, having been made in the context of developing a software capable of analyzing surface EMG signals and providing metrics for clinical assessment, is expected to aid researchers and clinical professionals in diagnosing and monitoring individuals who may suffer from neuromuscular disorders; by providing metrics such as conduction velocity, innervation zone, motor unit spike trains, without much margin for quantitative error when compared to other available software. Ideally, a subject, upon acquiring signals from their electrode-array, should also be able to upload them and be given relevant metrics for assessment, and, within the context of ML applications, even a classification of their pathology.

Beyond the software, however, this very dissertation is expected to provide explanations behind each resulting metric, while also explaining how to pursue the methodologies leading to the implementation of its tools – accurate tools which should prove to be robust, reliable and applicable in real-life situations.

1.4 Document Structure

This document is structured in five parts, or chapters, beyond the current introductory one, and the final 'Conclusions' chapter:

In the second chapter of 'Electromyography Analysis', an explanation of electromyography is given, regarding its history and generation. Afterward, a discussion is had about the signal processing techniques used for extracting metrics, with each section clarifying why the metric may be important in the scope of HDsEMG and the more prevalent methods for computing them.

The third chapter is the review of the literature, wherein recent advancements regarding, and most research done involving, both HDsEMG and machine learning – to better assess its limitations and potential – are presented, and finally the software options (commercial or not) that currently exist for either extracting metrics or training machine learning models.

The fourth chapter pertains to the methodologies adopted for the development and implementation of the algorithms – with a section for each type of analysis –, including validation approaches, and the synthetic signals and datasets used for testing; the processing done for two machine-learning-based applications ("ML Applications") is also explored within the last two sections of this chapter, serving as a proof of concept to its capabilities and inclusion in-software. The first application relates to gait monitoring of a patient with Parkinson's Disease, and how feature engineering can automate detection of Freezing of Gait. For the purpose of validating the creation of a database platform onto which one can upload their own HDsEMG signals and have trained models either learn or classify them, the second application relates to simultaneous gesture and health condition classification (in this work, Duchenne Muscular Dystrophy), and how different neural networks behave when learning from different EMG spectra.

A fifth chapter presents the results from each methodology of the previous chapter, and a discussion about how reliable each processing technique may be in a clinical context.

Chapter 2

Electromyography Analysis

2.1 Electromyography

Electromyography (EMG) is a fundamental diagnostic tool used in both clinical and research settings to assess the electrical activity of skeletal muscles. EMG provides valuable insights into muscle function, motor control, and neuromuscular disorders. This section will delve into the history and generation of EMG signals and the various aspects associated with high-density surface electromyography (HDsEMG).

2.1.1 Brief History of EMG

Starting in 1666 with the biologist Francesco Redi's work on electricity as generated by a fish's highly-specialized muscles [2], and progressing to 1792 when Luigi Galvani, professor of anatomy at the University of Bologna, showed that muscle contractions could be initiated by electricity [3] and to 1849 with Dubios-Raymond finding that voluntary muscle contractions' electrical activity can be recorded, the study of this myoelectric signal has been of increasing interest [4]. The term "electromyography", brought upon in 1890 by Étienne-Jules Marey who first recorded EMG [5], much like the hardware and acquired signal quality, has evolved, addressing different approaches to this study.

In 1929, Adrian and Bronk developed a method that allowed to record a single motor unit action potential (MUAP) in a human, through the connection of concentric and bipolar needle electrodes to an amplifier and a loud speaker [6]; eventually, a decade later, this led to Denny-Brown and Pennybacker capturing and analyzing fasciculations, from spontaneously motor unit firing, and fibrillation potentials (*i.e.*, action potentials from recently denervated muscle fibers), evaluating normal and abnormal MUAPs [7]. This opened the gateway toward the use of EMG for the pathological study of neuromuscular disorders (such as amyotrophic lateral sclerosis (ALS)), albeit in an invasive manner.

Non-invasive alternatives for clinical use were later introduced (in the 1960s), as electrode technology improved in the early 1900s. These consisted of surface EMG (sEMG), first adopted for treatments by Hardyck's research team [8]; whose eventual advancements have since included

the use of high-density EMG arrays, particularly at the surface of the skin, for capturing more muscle information and at a higher resolution.

2.1.2 Physiology and Generation of EMG

As is understood today, the generation of EMG signal is as follows:

The nervous system serves as the body's central control system, responsible for generating and transmitting neural signals. It consists of two primary components: the central nervous system (CNS) and the peripheral nervous system (PNS). The CNS is composed of the brain and spinal cord, while the PNS includes a network of nerves that connect the CNS to various parts of the body, including muscles.

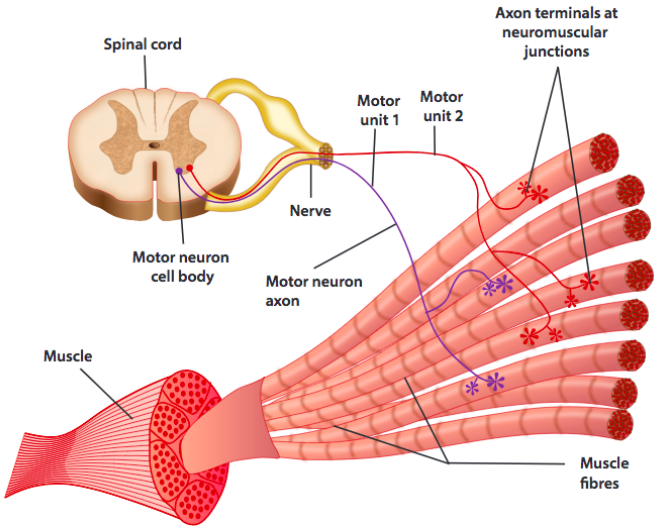


Figure 2.1: Diagram of the spinal cord composed of motor neurons whose nerve cells extend to multiple muscle fibers. Each neuron and the fibers it innervates known as a motor unit; from [9].

At the core of the nervous system are neurons, which are the nerve cells responsible for producing and transmitting electrical signals known as action potentials (AP) throughout the body. Among these neurons, motor neurons play a crucial role in controlling muscle movements: when a motor neuron fires an action potential, it triggers the contraction of all the muscle fibers within its associated motor unit (MU).

The alpha motor neurons are situated at the spinal cord and have axons that, as nerves, extend down to the skeletal muscle, branching out to connect to multiple muscle fibers on their individual motor endplate (found near the middle of its fiber). This means that, while each fiber is innervated by only one motor neuron, each motor neuron can simultaneously innervate multiple fibers (comprising a MU), as illustrated in Figure 2.1

Innervation occurs when, upon the arrival of the electrical impulse to the motor endplate of the muscle fiber, there is the release of neurotransmitters which create depolarization waves that

propagate both ways along the fiber, depolarizing it and generating an electrical signal that travels along the surrounding, conductive tissue, possibly reaching a nearby electrode. The signal this electrode captures is thus a result of a motor unit action potential (MUAP), or of the superposition (sum) of many individual MUs.

This depolarization wave also goes into the inside of the fiber, releasing calcium ions and leading to the mechanical contraction, which, in the case of a set of neurons innervating the same muscle (motor neuron pool), will result in greater force of contraction with the increase of the number of recruited active motor units in that muscle. This also results in more MUAPs and, from the sum, stronger EMG signal.

2.1.3 Characteristics of Motor Units

As the activity of EMG is heavily reliant on the behavior of the motor unit, understanding how each of its properties influences muscle activity and resulting signal can provide an insight into how the neuromuscular system works and eventually to identifying abnormalities and diseases.

The regulation of muscle force within the CNS, ability known as neural drive, involves motor unit recruitment and the rate at which it happens; for said recruitment, motor units discharge (or activate, firing electrical signal), resulting in a coordinated contraction of its fibers. Motor unit recruitment refers to the progressive and successive activation of muscle fibers, which results in an increase in the intensity of muscle contraction. This process is guided by Henneman's Size Principle [10], which dictates that, during a muscle contraction, smaller motor units with slow conduction velocity and fewer muscle fibers are activated first. As the need for greater muscle force arises (activities of higher voluntary contraction), larger motor units with fast conduction velocity and more muscle fibers are recruited last. Consequently, the number of motor units recruited (along with the rate of discharge) increases, allowing for a gradual increase in muscle force during weak contractions and more significant changes when greater force is required [11].

This principle is significant because it enables fine control for most activities through the use of fatigue-resistant small motor units. In contrast, larger motor units, which fatigue easily, are only called into action occasionally, such as during activities like running, where fine control is not a primary concern. How many muscle fibers are present within a motor unit will vary and will often correlate with the latter's size. Smaller motor units contain fewer muscle fibers (about 10 per MU), while larger motor units contain more (up to 2000 per MU).

The spinal cord drives different motor units asynchronously, causing muscle contraction to alternate among motor units one after another. This asynchronous activation ensures a smooth and continuous contraction even at low frequencies. The firing of motor units produces short mechanical twitches lasting about 4 milliseconds, and sustained contraction requires continuous, subsequent firing.

When fatigued, or suffering from a pathological muscle condition, however, a synchronization can occur, meaning that the action potentials being propagated are synchronized, possibly leading to force tremor [12; 13].

The MUAP's bidirectional propagation along the muscle fiber happens at a given conduction speed – being the distance travelled divided by time taken to travel –, known as the motor unit conduction velocity (MUCV) (or muscle fiber conduction velocity (MFCV), when averaging various action potentials propagating along fiber). Conduction velocity (CV) is related to aspects such as the diameter, size and type of muscle fibers, ion concentration, and other membrane properties [14; 15; 16], and the MU's firing rate (often detected as patterns of electrical impulses — the spike trains), being often used for evaluating contractile properties of MUs [17] and muscle fatigue [18; 19; 20], and for assessing motor function of patients with motor neuron disease; research by van der Hoven et al. [21], for instance, showed that ALS patients had an increased CV along with a decrease in median frequency; another study by Gerdle et al. [22] showed that fibromyalgia patients also had increased CV compared to healthy control, possibly due to alterations in histopathology and microcirculation.

Since the motor unit spike trains (MUSTs), often represented as shown in Figure 2.2B, are essentially the moments in which action potential are fired, they represent a summarized idea of most of the MUs' properties (factors such as the rate of firing, synchronization among motor neurons, and recruitment of additional motor units, all crucial determinants of muscle force and control), conveying *encoded* information about the timing and intensity of muscle activation; such as the force and precision of the muscle contraction.

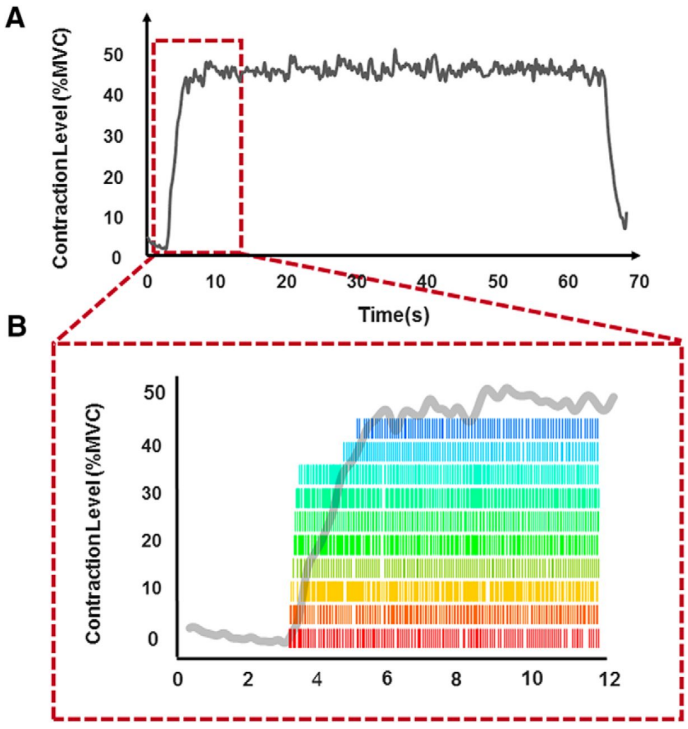


Figure 2.2: Representation of (A) EMG signal of a patient upon a targeted muscle reinnervation procedure and (B) Motor Unit Spike Trains (MUST) during the recruitment stage of the associated contraction, from [23].

2.1.4 Surface EMG

Surface electromyography (sEMG) is a widely used technique in the field of electromyography, serving as a non-invasive method for monitoring and recording the electrical activity of muscles. What distinguishes surface electromyography from other types of electromyography is the placement of electrodes directly on the surface of the skin, as opposed to invasive methods that involve inserting needles or wires into the muscle tissue, like with needle EMG (nEMG).

The non-invasiveness of sEMG makes it a more practical and comfortable option for both researchers and patients. The electrodes detect the electrical signals generated by muscle fibers during contraction and relaxation, providing valuable information about muscle activity, coordination, and function, albeit with noisier signal results, hence the need for more intricate signal processing techniques.

One notable advancement in surface electromyography is high-density surface electromyography. HDsEMG involves the use of a higher number of electrodes, distributed over a more extensive area, compared to traditional, "sparse"-electrodes sEMG systems. This increased spatial resolution allows for a more detailed and comprehensive analysis of muscle activity patterns. With HDsEMG, researchers and clinicians can obtain a finer representation of the electrical activity within a muscle or across multiple muscles simultaneously.

HDsEMG stands out in the scope of electromyography due to its ability to capture intricate details of muscle activation, enabling a more precise understanding of muscle function and movement. This advanced technology has applications in various fields, including motor control research, biomechanics, and the development of prosthetics and orthotics. The enhanced spatial resolution of HDsEMG makes it a valuable tool for detecting subtle changes in muscle activity, providing insights into muscle coordination and contributing to more targeted and effective interventions in clinical and research settings.

2.1.5 Clinical Cases and Pathologies

As EMG generation may be affected by not just anatomical features, but pathological ones, this work employs two datasets of EMG signals generated by individuals with one of two neuromuscular conditions: Parkinson's Disease (PD) and Duchenne Muscular Dystrophy (DMD) (as well as healthy control), for assessing their effects; understanding what they entail is, thus, relevant.

2.1.5.1 Parkinson's Disease and Freezing of Gait

Parkinson's disease (PD) — first described in 1817 by James Parkinson, who referred to it as the "shaking palsy" — is a progressive neurodegenerative disorder that primarily affects motor control, although non-motor symptoms are also prevalent. PD is the second most common neurodegenerative disease after Alzheimer's disease, with an incidence that increases with age. The hallmark of Parkinson's is the degeneration of dopaminergic neurons in the substantia nigra, a region of the brain that plays a pivotal role in regulating movement. This degeneration results in a

significant reduction in dopamine levels, particularly in the striatum, which is essential for smooth, coordinated muscle movements.

Clinically, PD is classically defined by the cardinal motor symptoms: tremor at rest, bradykinesia (slowness of movement), rigidity, and postural instability. Tremor, which often starts asymmetrically, is perhaps the most visible symptom, but bradykinesia and rigidity tend to be more disabling. Bradykinesia is characterized by a reduction in the amplitude and speed of voluntary movements, leading to difficulties with everyday tasks such as writing, buttoning clothes, and walking. Rigidity, on the other hand, refers to the stiffness of muscles, often presenting as a "lead-pipe" resistance to passive movement. Postural instability, which tends to develop in later stages, increases the risk of falls and fractures.

Freezing of gait (FOG) is a complex and debilitating motor symptom frequently observed in advanced PD, characterized by the sudden and episodic inability to initiate or maintain walking. This phenomenon manifests as a brief, transient episode where the feet appear to *stick* to the ground, making forward movement impossible despite an individual's intention to walk. It often occurs during specific situations, such as when turning, walking through narrow spaces (like doorways), or approaching obstacles. FOG is not exclusive to Parkinson's but is also seen in other conditions, such as progressive supranuclear palsy and multiple system atrophy. However, in Parkinson's disease, it is one of the most disabling symptoms as it significantly increases the risk of falls and contributes to loss of independence. In FOG, the muscles may be in a state of co-contraction, where opposing muscle groups (agonists and antagonists) contract simultaneously, leading to a rigid, immobile state. this form of isometric contractions, shows that, even if active, opposing muscles activate simultaneously, canceling each other out, preventing any movement.

In PD, due to degeneration of dopaminergic neurons in the basal ganglia, disrupted motor output can manifest as abnormal EMG patterns. During normal gait, EMG signals show a consistent and organized activation of muscles in a pattern that reflects the coordination between different muscle groups. In contrast, during FOG episodes, this coordination breaks down, resulting in a lack of appropriate muscle activation. This also reflects a reduced recruitment of motor units, reduced firing rates or more irregular firing patterns, all of which are, outside of this FOG state, already more prevalent or inefficient than that of a healthy individual's.

2.1.5.2 Duchenne Muscular Dystrophy

Duchenne Muscular Dystrophy (DMD) is a genetic disorder characterized by progressive muscle degeneration and weakness. It is one of the most common and severe forms of muscular dystrophy, typically affecting young boys and manifesting in early childhood. The disease is caused by mutations in the dystrophin gene, which is located on the X chromosome. Dystrophin is a crucial protein that helps maintain the structural integrity of muscle cells by linking the internal cytoskeleton of muscle fibers to the surrounding extracellular matrix.

The absence or dysfunction of dystrophin in DMD leads to the breakdown of muscle cell membranes, causing increased permeability and subsequent muscle fiber damage. Over time, this damage results in inflammation, fibrosis, and progressive loss of muscle function. Clinically,

patients with DMD experience difficulty with motor skills such as walking and climbing stairs, and the condition eventually leads to wheelchair dependence and respiratory complications as the disease progresses.

In DMD-affected muscles, EMG may reveal signs of ongoing muscle fiber degeneration and regeneration. This is often seen as increased spontaneous activity, such as fibrillations and positive sharp waves, alongside myopathic motor unit potentials. These abnormalities are indicative of the muscle's attempt to cope with the constant damage and the resulting disruption of normal muscle function.

As muscle fibers are destroyed and replaced by connective tissue and fat, surviving motor units may have to compensate for the lost muscle fibers. Such compensation often results in altered motor unit recruitment patterns. The remaining motor units may become larger as they innervate more muscle fibers in an attempt to maintain muscle function. This adaptation can lead to decreased precision and coordination in muscle contractions. Because of that, EMG of DMD subjects may often show increased motor unit potentials and recruitment thresholds, reflecting the increased effort required by the surviving motor units to generate the same amount of force. Altered firing patterns, possible increase in firing rates and motor unit numbers being recruited, to compensate for the reduced efficiency and force production of affected muscle fibers, and reduced conduction velocity due to the presence of fibrosis, may all occur, all reflecting, and resulting in, prolonged and less efficient muscle activations.

2.2 Amplitude Analysis

The relevance of EMG as a monitoring tool comes from the many metrics it provides. To that end, many EMG signal processing and analysis techniques are adopted. Among them, amplitude analysis.

An electromyogram graph represents the voltage (often between 50 μV and 30 mV) of the acquired signal as it varies throughout time. Said signal displays changes in amplitude that come from muscle contraction which, in turn, can be affected by motor impairment, fatigue, etc. From this, one can quantify muscle effort, compare muscle activation among different subjects, muscle groups and physical interventions. In performing an isometric exercise of the quadriceps femoris muscle, and when compared to healthy subjects, patients suffering from congenital myopathy with type I fiber predominance exhibited six times greater EMG voltages per unit of force [24], which not only meant that the latter group would suffer from fatigue more easily, but that their neuromuscular efficiency was worse [25].

This type of assessment, however, is often better understood by obtaining time-domain metrics such as peak signal amplitude or the mean amplitude, which often come accompanied by normalization strategies that seek to decrease signal variability: using peak-to-peak M-wave amplitude or using peak electromyography signal during a maximum voluntary contraction (MVC) [26], for example.

Similarly, by calculating EMG envelopes which smoothen the raw EMG signal and, therefore, filter the signal, while providing a representation of the overall muscle activity trends: these are amplitude estimators, often offering insights into the muscle's sustained activity levels, aiding in the assessment of muscle fatigue and overall performance during various physical activities.

2.2.1 Root Mean Square

From the signal, one can determine the root mean square (RMS), which provides a measure that reflects both the amplitude and duration of muscle contractions. This makes it a valuable tool for evaluating overall muscle activation levels and detecting changes in muscle activity patterns,

$$\mathrm{RMS} = \sqrt{\frac{1}{L}\sum_{i=1}^{L} x_i^2} \tag{2.1}$$

As is shown in Figure 2.3, the envelopes obtained from RMS values can help in detecting altered myoelectric activity – a clear alteration in EMG onset is visible from healthy to neuropathic to myopathic –, whose analyses may aid in the diagnosis of gait disorders.

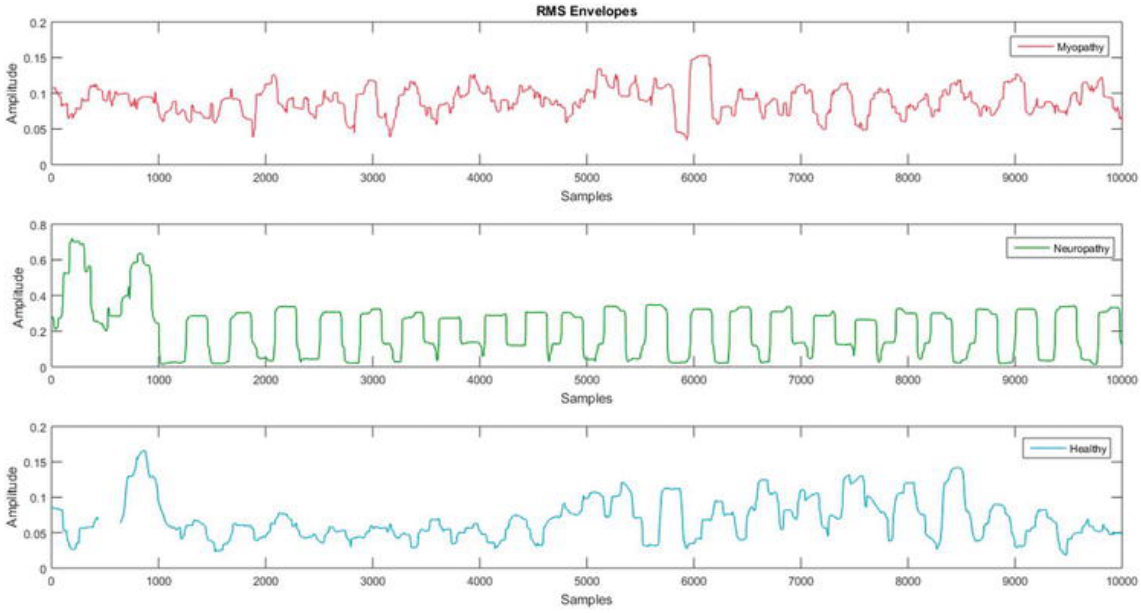


Figure 2.3: RMS envelopes from patients with neuropathic, myopathic, and healthy conditions. Using a non-overlapping window of 200 samples, a paired student t-test showed that there was a statistically significant difference ($p < 0.05$) between the conditions of healthy and neuropathic and healthy and myopathic [27].

2.2.2 Average Rectified Value

Also, the mean absolute value (MAV), which provides a representative measure of the overall signal magnitude, computed as follows,

$$\text{MAV} = \frac{1}{L}\sum_{i=1}^{L}|x_i| \tag{2.2}$$

wherein L is the size of the time window used and x_i is the i-th sample of the signal.

This metric can be represented through time, often also known as an average rectified value (ARV) representation of the EMG signal. A work by Hibbs et al. [28], from which Figure 2.4 was obtained, showed that this measure is the least variable when it came to measuring muscle activity (produced during core training exercises), serving as an appropriate indicator for submaximal activities that may occur during the stabilization of the body [29]. The Figure also shows peak EMG as a relevant variable that shows maximal activity of a given muscle.

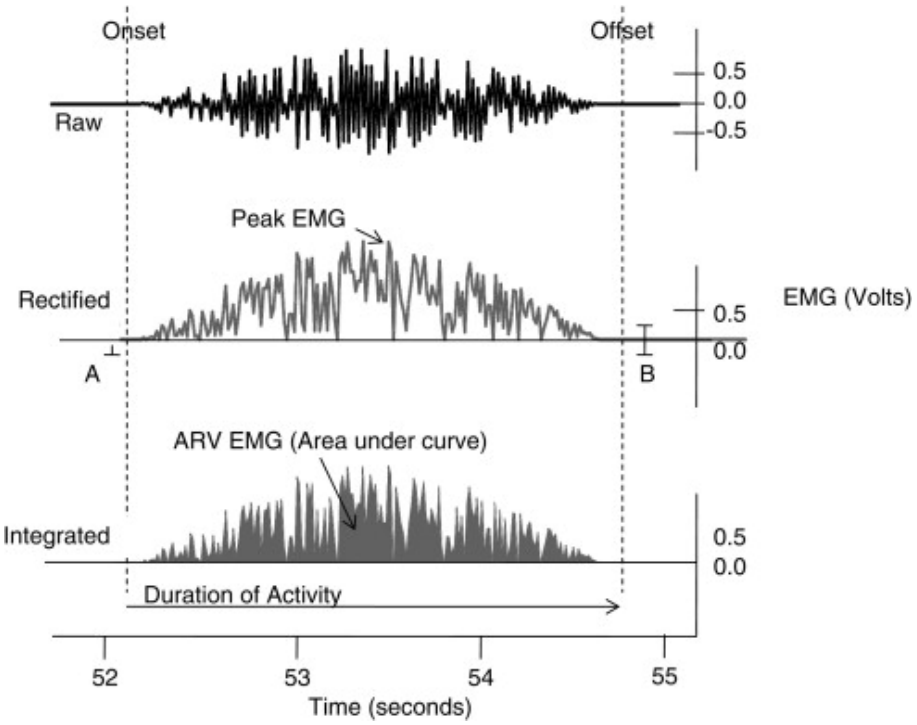


Figure 2.4: Diagram of the ARV EMG and Peak EMG processing methods, from [28].

Other approaches that use the average EMG value over a moving window that traverses the length of the EMG signal (moving average) or that use the integrated EMG (the area under the curve of a rectified EMG signal) computed from time window to time window, can be used as a neuromuscular efficiency analysis tools [30].

Statistical temporal correlations between samples can often lead to higher variances (compared to uncorrelated) in estimates of values such as the MAV or the RMS [31]. In attempts to fix this issue, prior to computing estimates, six steps for decorrelating samples are often employed – (1) baseline noise and line interference rejection and attenuation (from preparing one's skin, to adopting digital and analog filters); (2) whitening (decorrelation) of the signal [32]; (3) multiple-channel combination; (4) amplitude demodulation (rectifying whitened signal and raising MAV or

RMS to the power of 1 or 2, respectively); (5) smoothing (by way of moving averages or low-pass filters); (6) relinearization (inversion of the demodulation process) [33].

2.2.3 Topographical Mapping

From the 2D array of electrodes it becomes possible to represent the signal in space (the area of the skin over which the array is placed) for each instant or interval of time – this can by done by way of colormapping the HDsEMG signals' amplitude, where each pixel corresponds to a different channel (for greater image resolution, it is often also applied methods of interpolation for estimating values between pixel locations). Besides the raw EMG voltage amplitude itself, this amplitude may be the ARV, MDF, MNF, RMS of each of the matrix's channels, meaning that topographical mapping works as an extension of amplitude and spectral analyses: on top of comparing activation levels and muscle effort, it also allows for a representation of its spacial distribution and identification of areas of greater or lesser muscle activity. When observed throughout time, it shows how the electrical potential at the surface of the skin evolves, sometimes leading to the identification of motor unit discharge and the MUAP's propagation along the mapped area [34].

Using a $16cm \times 4cm$-size, 128-monopolar channel grid, covering the biceps brachii muscle, Rasool et al.'s [35] study on muscular activations in stroke survivors showed that, from a stroke, a shrinkage of the active sEMG region occurred on the affected arm, along with a decrease in intensity, for all force levels, as evident in the activation maps of Figure 2.5, where the darker reds correspond to the highest RMS values, and the darker blues the lowest; allowing for the assessment of muscle atrophy upon a stroke, and of the potential loss of motor unit numbers from denervation or of how the CNS pathways may have been disrupted.

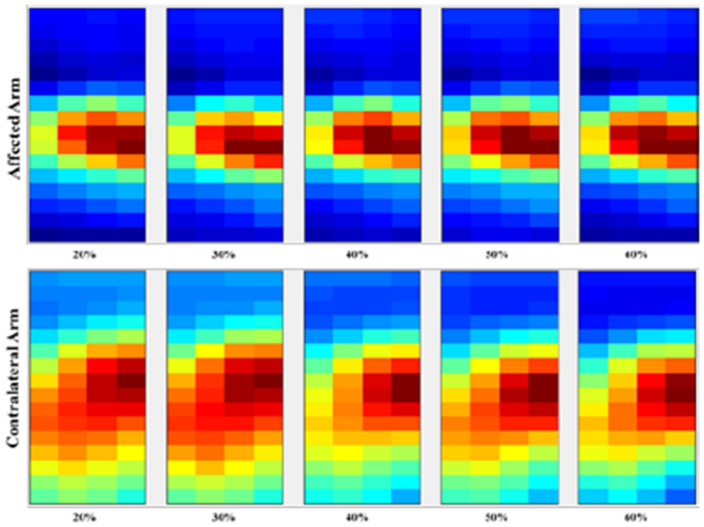


Figure 2.5: RMS maps of a stroke subject for five force levels, from 20% to 60% of MVC, during an isometric non-fatiguing activity, captured from the long head of the biceps brachii of both arms. Each map normalized to its maximum value, with dark red = 1 and dark blue = 0; from [35].

By quantifying the shift of the HDsEMG map's center of gravity (CoG), or centroid – often computed for the entire grid area,

$$\text{CoG} = \frac{1}{\sum_{n=1}^{N} H_n} \sum_{n=1}^{N} H_n \cdot \mathbf{u}_n \tag{2.3}$$

where H_n is the intensity (RMS, for instance) in channel n whose location is described by vector $\mathbf{u}_n$; but which can also be done for specific regions ("isolated masses") within the area [36] – one can quantitatively evaluate the change in distribution of muscle activity during contractions [37]: by comparing the CoGs of different activities with and without transcranial magnetic stimulation (TMS), van Elswijk et al.[38] showed that the voluntary wrist extensions' CoG was more radial and more distal than that of voluntary finger extensions and than that of TMS-evoked responses during rest, and that the ulnar-radial CoG position of TMS responses had a strong dependence on stimulus intensity. By tracking its trajectory during the entirety of an isokinetic knee extension exercise, Kuruganti et al.'s [37] work showed that a patient with a transtibial amputation's CoG had more variability when compared to a healthy control (as well as reduced intensity), as seen in Figure 2.6. Aspects like these may hint at changes in motor-unit recruitment [39; 40].

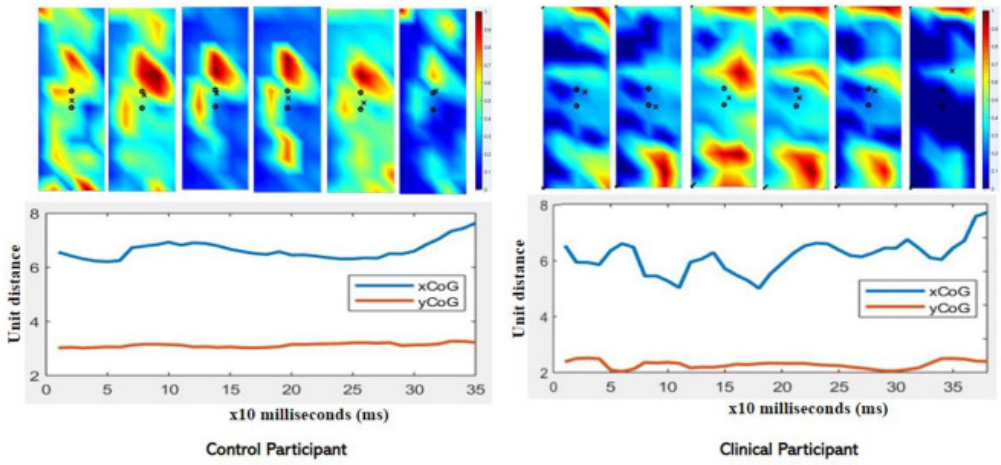


Figure 2.6: Activity maps and distance by time graphs for comparison of CoGx_{mean} and CoGy_{mean}, the mean values of the CoGs in the x and y coordinates respectively, for a control and clinical patient upon moderate isokinetic knee extension, from [37].

2.2.4 Timing Analysis and Onset Detection

When performing a repetitive activity, such as cycling or running, the active muscles tend to contract in a coordinated manner. Since the amplitude of the EMG signal increases with the contraction, one can detect muscle activation onset in the signal, be it through visual inspection, statistical methods, machine learning, or threshold-based techniques (among many others); to better understand this notion, Figure 2.7 shows an example by Agostini et al. [41] wherein activation intervals within gait cycles were detected from EMG electrodes at the gastrocnemius lateralis.

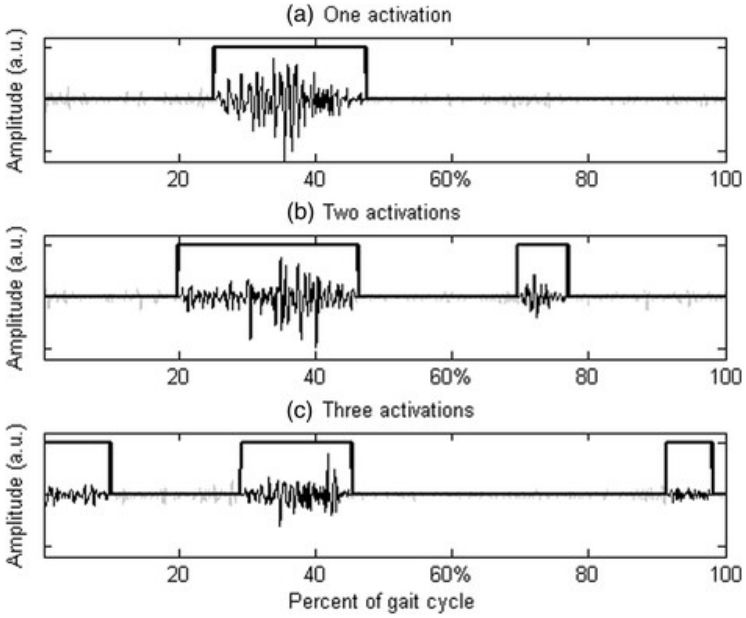


Figure 2.7: Activation intervals of gait cycles, detected with EMG signals captured at the gastrocnemius lateralis, from [41].

The thresholding method, for instance, is one that, starting from signal conditioning (rectification, to low-pass filtering (Butterworth or Chebyshev, predominantly), to whitening, etc.) – often the EMG envelope is adopted, though Teager-Kaiser Energy Operator (TKEO) is prevalent too – is followed by detection based on the use of a test function and a decision rule.

Various signal characteristics, such as Standard Deviation, Period of time, %Maximum Voluntary Contraction, and %Maximum EMG Amplitude, are considered for threshold selection. Three categories of threshold-based methods exist, with different rules: Single Threshold (ST), Double Threshold (DT), and Adaptive Threshold (AT). ST, the most prevalent, compares EMG signal amplitude with a predefined threshold. DT introduces a second threshold to enhance precision and avoid false positives by having an upper one that, once crossed, defines the beginning of muscle activity, and lower threshold that, once dropped below it, defines its end [42]. AT adapts the threshold using signal-to-noise or energy values, offering potential improvements in precision over time.

Some other approaches have used statistical techniques (like using Approximated Generalized Likelihood Ratio [43]) and mathematical techniques [44; 45; 46; 47; 48], entropy-based [49; 50; 51; 52; 53] or energy-based [54; 55; 56] methods, slope detectors [57; 58; 59], computer vision [60] and external stimulation [61; 62].

More recently, the use of machine learning for this same purpose has increased, and to comparable results: Ghislieri et al. [63] introduced and validated a deep-learning (DL) method, using long short-term memory (LSTM) recurrent neural networks. The performance of the LSTM-based muscle activity detector, using both simulated and real sEMG signals obtained from healthy individuals, orthopedic patients, and those with neurological conditions, was assessed by comparing it to the TKEO and DT methods; the findings indicated that the LSTM outperformed the first two,

while avoiding the necessity for background-noise and signal-to-noise ratio (SNR) estimation.

2.3 Spectral Analysis

Spectral decomposition provides information related to the frequency content of a signal, such as mean frequency (MNF) and median frequency (MDF, which is a parameter dividing the total power into two equal parts); knowing the power spectral density $S(f)$ and sampling rate f_s,

$$f_{mean} = \int_0^{f_s/2} f \cdot S(f) \cdot df \tag{2.4}$$

$$\int_0^{f_{median}} S(f) \cdot df = \frac{1}{2} \int_0^{f_s/2} S(f) \cdot df \tag{2.5}$$

By studying the spectrum of frequencies (spectral analysis) resulting from mathematical algorithms such as Fourier Transform, or Wavelet Transform, clinical research has shown that correlations can be found between, for instance, muscle fatigue and higher spectral densities at lower frequencies; or between tremors from someone with Parkinson's disease (PD) and lower spectral amplitude [64], as compared to the higher amplitudes seen for usual tremors.

2.3.1 Fourier Transform

By employing a Fourier Transform it becomes possible to project continuous time-domain data into frequency domain, and detect which regions of frequency have greater amplitudes. This conversion method can be described by a mathematical operation that transforms a domain $x(t)$ into a domain $X(f)$, where t is time and f is the frequency:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-2\pi i f t} dt \tag{2.6}$$

Or, for the discrete case (sequence $x[n]$), where $X[k]$ is the frequency content at discrete $f_k = \frac{k}{N} \cdot f_s$,

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-2\pi i k n / N} \tag{2.7}$$

The Fast Fourier Transform (FFT) provides a spectrum for a discrete signal, though the time evolution is not considered; this can be useful for cases of static contractions, where the signal is thought of as roughly stationary and the muscle force, speed and length don't change [65].

Conversely, in overcoming the non-applicability of FFT on dynamic tasks, the short-time Fourier Transform (STFT) is a related algorithm that divides the signal into segments (defined by 0.25-second to 1-second windows of partial to no overlap [34], often rectangular [34]) and obtains the Fourier transform of each one, thereby providing information about how the frequency content changes over time – allowing for Time-Frequency (TF) domain analysis, through spectrograms. This means that the STFT may perform adequately on non-stationary signal, being better for slow actions than rapid bursts: short fixed observation windows (as would be used for 500-ms

muscle activity bursts) have improved time resolution but worse frequency resolution, long fixed windows have the opposite [66].

2.3.2 Wavelet Transform

Another way to decompose the EMG signal is by employing a wavelet transform – often used in studies of non-stationary EMG [67; 68; 69; 70] –, which, as opposed to the Fourier Transform, does not assume signal stationarity. It uses windows of different sizes (the wavelets), thereby allowing for multiple resolutions.

The Continuous Wavelet Transform (CWT), given by

$$\mathrm{CWT}(a,b) = \int_{-\infty}^{\infty} x(t) \cdot \psi^* \left(\frac{t-b}{a} \right) dt \tag{2.8}$$

where a and b define scale and position of the wavelet, $x(t)$ is the signal, and ψ^* is the complex conjugate of the mother wavelet function ψ, which is often used to analyze EMG,

$$\psi(t) = \pi^{-1/4} e^{iw_0 t} e^{-t^2/2} \tag{2.9}$$

with w_0 the dimensionless frequency parameter (the Discrete Wavelet Transform (DWT) is analogous to CWT, but using discrete wavelets to decompose the signal into frequency bands), is a commonly used approach, since it balances time and frequency resolutions. This means that, much like with the STFT, this transform is used as a TF analysis technique. It also means that, be it in static or dynamic conditions, it ultimately leads to similar types of information about muscle fatigue as that of the Fourier Transform (similar patterns and tendencies during isometric and dynamic exercises [71; 67; 70]), just with better accuracies and levels of reliability and noise quality, according to reports by Karlsson et al. [70; 68].

Other transforms may include the Hilbert-Huang Transform (HHT), for analyzing non-linear and non-stationary time series data. It decomposes a signal into its intrinsic mode functions (IMFs) using empirical mode decomposition (EMD) and then applies the Hilbert transform to these IMFs to extract instantaneous frequency and amplitude information, as Figure 2.8 shows. This method is particularly useful in analyzing complex signals where traditional Fourier-based techniques may fall short.

2.4 Conduction Velocity

Among several aspects, estimation of the muscle fiber conduction velocity (MFCV, or CV) has proven to be reliable for muscle fatigue development monitoring [14; 73], since it has been shown to decrease with it [19; 74; 18]. Its use as a clinical assessment tool is, thus, of relevance.

When estimating the CV, several methods have been adopted, often requiring knowledge about the time it took for an action potential to travel a given distance within the muscle fiber. To that end, using a HDsEMG grid and knowing its configuration and dimensions has proven helpful in

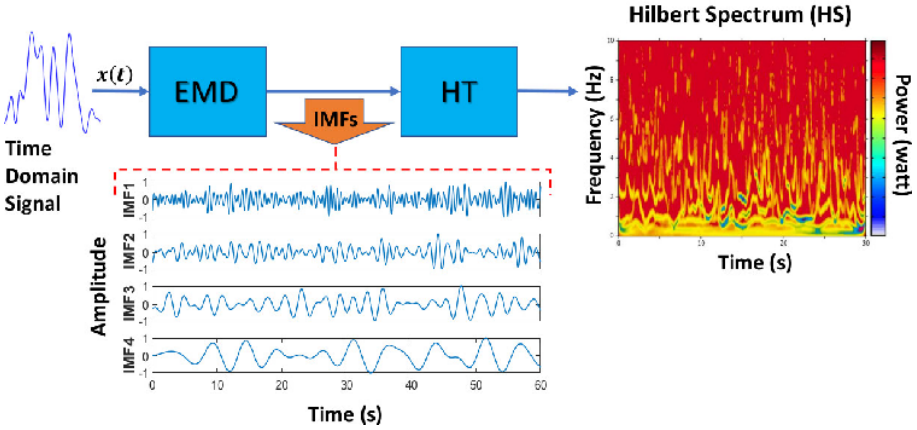


Figure 2.8: Block diagram of the HHT process, from [72].

accurately estimating the CV without the need of an invasive approach, and has, thus, led to the development of more intricate techniques, many of which specific to surface EMG signals [75; 76].

The multichannel nature of HDsEMG lends itself to the easy identification of propagating APs. Visually, as in Figure 2.9 – of the vastus medialis muscle's EMG signals resulting from a cycling exercise [77] – a shape similarity is evident between adjacent channels through time and along the proximal-distal direction, indicated by the dashed lines.

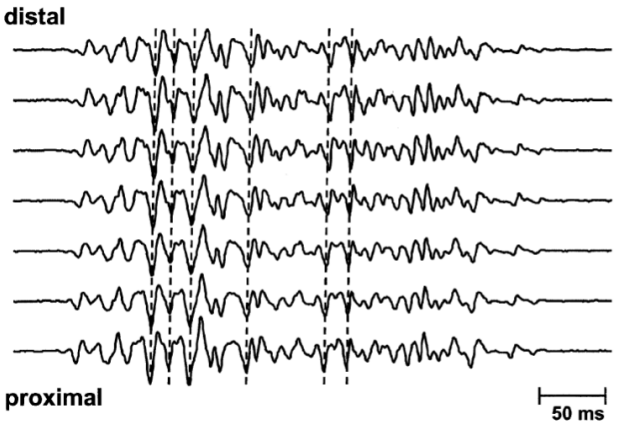


Figure 2.9: Single differential multichannel sEMG signals detected from the vastus medialis muscle upon a cycling exercise resulting in a burst of activity, with dashed lines indicating the propagation of APs, from [77].

Cross-Correlation and Maximum-Likelihood Estimation

Supposing that two electrodes are placed along the muscle fiber, the CV estimation can done by measuring the time delay Δt between the first peak or zero crossing seen in the first electrode and that same peak or zero crossing now in the second. Knowing the inter-electrode distance (IED),

thus distance between electrodes Δs, allows for the computation:

$$\mathrm{CV} = \frac{\Delta s}{\Delta t} \tag{2.10}$$

This type of approach is known as a cross-correlation measurement, which tracks signals and compares their similarities while knowing their relative displacement.

Time resolution problems often make the estimation of Δt less accurate, so looking to minimize the mean squared error (MSE) in the frequency domain [78; 79; 80] – as done in Farina et al.'s [81] algorithm which uses the iterative Newton method for finding the zero of the MSE's first derivative – or resorting to the Fourier transform of the signals to obtain a phase difference and using it to compute the delay [82] – known as spectral matching – have been implemented to address the issue.

Mathematically, and as proposed by Farina et al. [83] through the maximum-likelihood estimation (MLE) for the delay from bi-dimensional sEMG recordings (grid with K rows and N_c columns),

$$\mathrm{MSE}(\hat{\theta}) = \sum_{c=1}^{N_c} \sum_{k=1}^{K} \sum_{n=1}^{N} \left(x_{ck}(n) - \hat{s}_c(n)\right)^2 \tag{2.11}$$

where $\hat{\theta}$ is the estimated delay, n the n-th sample (N total), and x_{ck} is the signal obtained at column c and row k from what is assumed to be the same AP (creating signal $\hat{s}_c$, which, ideally, is the same along column c, provided the matrix is aligned with the muscle fiber, and is estimated as

$$\hat{s}_c(n) = \frac{1}{K} \sum_{m=1}^{K} x_{ck}\left(n + (m-1)\hat{\theta}\right) \tag{2.12}$$

which is the average of all K signals in that column), but whose difference in relation to it derives from different independent, white, zero mean, Gaussian noises $w_{ck}(t)$ of equal variance,

$$\begin{gathered} x_{ck}(n) = \hat{s}_c(n - (k-1)\hat{\theta}) + w_{ck}(n), \\ k = 1, \ldots, K; \qquad n = 1, \ldots, N; \qquad c = 1, \ldots, N_c \end{gathered} \tag{2.13}$$

This MLE-criterion-based approach to minimizing the MSE results in an estimated delay $\hat{\theta}$ that has a temporal resolution defined by the sampling rate (*i.e.* depends on n), which should thus be sufficiently high. The application of a Fourier Transform (spectral matching) avoids this problem, resulting in a $\mathrm{MSE}_f(\hat{\theta})$ equation that no longer depends on n; this is preferable to the computationally-demanding alternative of interpolating and re-sampling the signals to reach the desired sampling frequency.

When it comes to CV estimation, while the above are commonly employed, other methods may involve generating difference-power-spectra or an ensemble interference-pattern-data-point average cross-spectrum, known as dip analysis and power-modulating-component techniques respectively [84]; others may also involve only looking at the small interval of time where there's

a peak in the signal (unlike cross-correlation, which uses the entire signal period), known as interpeak latency, which can give information about the spread of the CVs [85]; among many others [86].

2.5 Motor Unit Decomposition

The complexity of the raw sEMG signal captured by an electrode over a skin region (which covers a muscle and, as such, several muscle fibers) is the result of superpositions of many MUAPs, as well as noise (and other convolutions that happen as the electrical signal travels to the skin) – it is, essentially, the algebraic summation [87]. A subject of clinical interest pertains to MU decomposition, *i.e.*, attempting to isolate each signal in that superposition and identify single motor units (as represented in Figure 2.10), so as to study their MUSTs' properties (*e.g.*, number of motor units, firing rate, recruitment and derecruitment thresholds of MUs during physical tasks), often as an insight to neuromuscular diseases affecting motor function [88; 89].

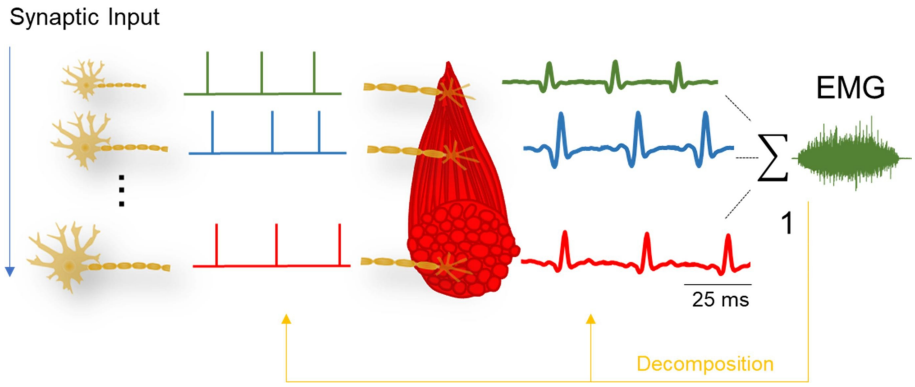


Figure 2.10: Diagram displaying the concept of EMG MU decomposition, given an EMG signal that is the superposition of several action potentials resulting from synaptic inputs, from [90].

While intramuscular (iEMG) electrodes allow for the recording from deep muscles, leading to stronger signals that are more local and specific to the fiber of study [88] and with higher signal-to-noise ratio (SNR) – thus, providing more accurate motor unit decomposition results [91; 92] –, when it comes to surface electromygraphy (which is non-invasive and easier to use), high level of superposition is inevitable, often times requiring the adoption of decomposition techniques that take advantage of the high number of channels in HDsEMG.

Blind Source Separation

Among the HDsEMG decomposition techniques stand out those based on blind source separation (BSS) methods, which can detect and track motor unit discharge times [93; 94; 95] upon the execution of isometric tasks [96; 97; 98] – thus providing estimates of the neural drive that stimulates muscle contraction [90] (overcoming some of the weaknesses found when estimating it through global EMG analysis [99]), information like the maximal rate of force development [98],

and correlations between aspects like aging and force steadiness and variability in the common synaptic input to motor neurons [100]. These are automatic procedures that separate motor units ("sources") with no *apriori* knowledge ("blind") and with only a few assumptions made about the source signals, though often followed by user-inspection of the motor unit spike trains identified herewith [101].

This can be done by way of independent component analysis (ICA), which focuses on statistical independence to separate mixed signals, assuming each component corresponds to a distinct motor unit; spike-triggered averaging (STA) has also been used, relying on averaging signal segments around known trigger events to reveal the common features of motor unit action potentials, facilitating their isolation and characterization.

Other proposed techniques, mostly based on special cases of ICA, are ones like the FastICA [102], which involves (1) channel selection, (2) signal extension, (3) whitening through eigenvalue decomposition, (4) FastICA-based deconvolution to obtain sources, (5) peak detection through clustering (*e.g.* k-means) and (6) removal of duplicated MUs, as shown in Figure 2.11, and is highly computationally efficient [103; 95]; from it there have also been derived new approaches (*e.g.*, Peel-off FastICA [95] for resolving superimposed waveforms and convergence of the ICA algorithm to the same MU; deflationary FastICA [94] for avoiding noise).

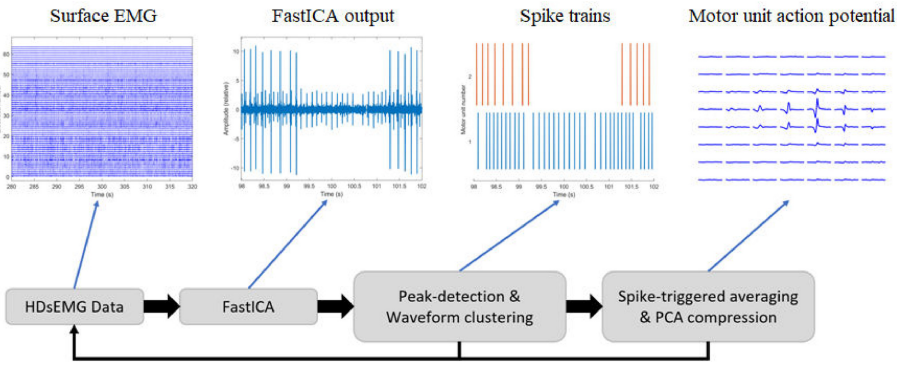


Figure 2.11: Block diagram of the FastICA MU decomposition technique for HDsEMG; this specific case was developed within a peel-off method framework, from [104].

There is also the prominently used Convolution Kernel Compensation (CKC) method as proposed by Holobar and Zazula [105], as well as its natural gradient extension [106], both identifying the source pulse trains in a sequential manner using a source deflation technique to prevent reidentifications, and bypassing the need of the ICA approach of estimating convolution mixing matrices by instead estimating cross-correlation vectors (the gradient extension optimizing it by gradient descent), while also being noise resistant; its output has also been used to train recurrent [107] and convolutional [108] neural networks, as deep learning approaches to MU identification. Another is the Convolutive Sphering as proposed by Negro et al. [94], which combines several of convolutive blind source separation approaches mentioned previously, and which, along with an iterative optimization process, maximizes the non-gaussianity and sparseness of the extracted sources, enabling the decomposition of a large number of motor units .

From the decomposition, motor unit analysis can ensue, often leading into the study of muscle fiber properties (*e.g.*, size and diameter [109]); or even MU conduction velocity estimation, through double differential derivations computed by differentiation of the averaged monopolar motor unit action potentials along the columns of the electrode grids [110]. Additionally, knowing the intervals of time between firings, for instance, can provide insight into discharge rate; similarly, one can determine which motor units are recruited and derecruited upon a physical task. Figure 2.12 displays results obtained by Del Vecchio et al. [111] that evidenced increases in discharge rates and decrease in recruitment-threshold of the same MU at plateau of force after 4 weeks of strength training at the tibialis anterior muscle; this information was extracted from the spike trains of MU decomposition.

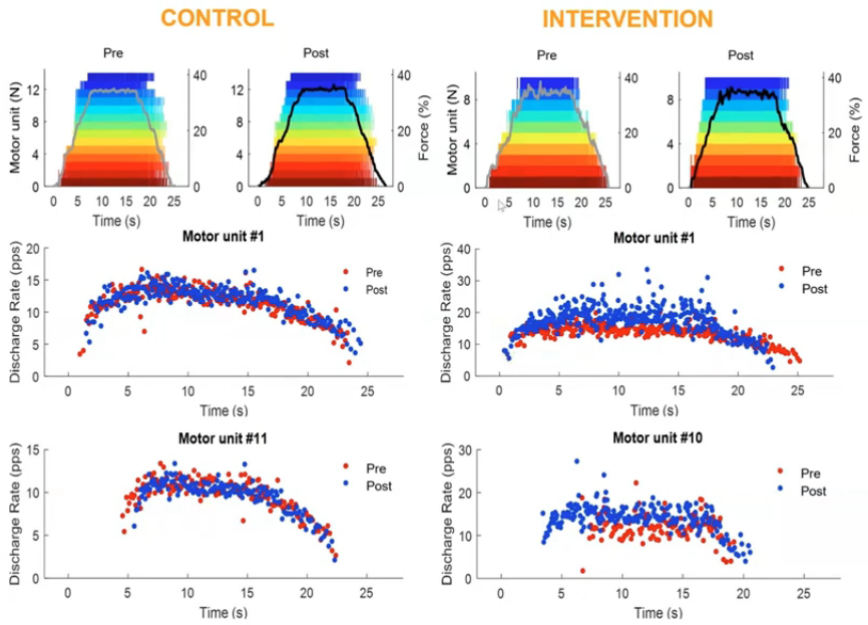


Figure 2.12: Discharge times for MUs across 4-week strength-training intervention, from [111].

Chapter 3

Review of the Literature

A comprehensive search of academic databases was conducted, including PubMed, IEEE Xplore, ScienceDirect, and Scopus. Four sets (from a to d) of keywords (and some variations compatible with each database) were used for the search, each with a specific topic in mind:

#1: ("high density surface electromyography" OR "high density surface electromyogram" OR "high density surface EMG" OR sHDEMG OR "HD sEMG" OR HD-sEMG OR hdsemg)
#2: (“machine learning” OR “deep learning” OR ML OR DL OR AI OR “Artificial Intelligence” OR “neural network” OR “*supervised learning”)
#3: (detect* OR prognos* OR diagnos* OR monitor*) AND (fatigue OR patholog* OR condition OR "musc* disorder" OR “neuromusc* disorder” OR “musc* dystrophy” OR “neuromusc* disease” OR “musc* disease” OR neuropath*)
#4: (track* OR “motor unit” OR MU OR recruitment OR “motor neuron” OR “motor-neuron” OR “motor task” OR fiber OR “firing rate” OR "motor control" OR "spike train*" OR discharge OR “conduction velocity” OR CV)
#5: (“processing technique” OR synergy OR activation OR “pattern recognition” OR “activity map” OR “mapping” OR decompos* OR “component* analysis” OR imaging OR “blind source separation” OR BSS OR “signal separation” OR “spatial pattern” OR “conduction velocity”)

a. #1 AND #2

b. #1 AND #3 NOT #2

c. #1 AND #4 NOT #2

d. #1 AND #5 NOT #2

Set a) was used for the collection of the known applications of artificial intelligence in HD-sEMG signal processing (be it for gesture classification, be it for activation pattern recognition in general).

Set b) sought to include the known cases of HDsEMG when used as a tool for diagnosing, detecting and monitoring muscular anomalies, from nerve damages to neuromuscular diseases. To

note that the express exclusion of any artificial intelligence-related content was done to prevent repetitions.

Set c) includes terms that are very prevalent in HDsEMG studies and which are or relate to metrics of focus within this work.

Set d) addresses the signal processing techniques that are known to have been used in HDsEMG studies; ranging from mapping to component analysis.

The search was limited to studies published from 2015 onwards, ensuring the inclusion of the more recent developments in the field.

.

3.1 Machine Learning Algorithms used with HDsEMG

To better understand the current state of machine learning (including deep learning) when used with HDsEMG, this section delves into the many applications found within this scope.

3.1.1 Advancements in Gesture and Other Human Activity Recognition

A common topic of study is that of using HDsEMG for gesture recognition (mostly hand gestures with the HDsEMG array placed on a limb) from anywhere between 2 to over 20 gestures (as done by Zhang and Zhou [112], who classified 20 different arm, hand, and finger movements with an 89-electrode array, to accuracies of over 96%; and by Amma et al. [113], who introduced a 27-gesture HDsEMG dataset (CSL-HDEMG) captured at the upper forearm with a 192-electrode, 8×24 array, reaching within-scenario accuracies of 90% though with inter-session accuracies of 59%; and as done by Sun et al. [114], classifying transient phases within 65 gestures with a heterogeneously dilated LSTM model for prioritizing the deep observations and hidden states of data segments), often offline with results from 95% to 99% [115; 116], but also for real-time integration [115; 117; 118; 119; 120; 121; 122; 123]. The results of these works show that there are benefits to the high spatio-temporal resolution and high-density activity mapping provided by HDsEMG, from which a lot of substantial information can be extracted. This is often done within the scope of rehabilitation engineering, seeking to provide patients with tools that may help them recover lost abilities. In the case of amputees, prosthetic limbs controlled via myoelectric commands are able to replace the missing body part, and more commands means more degrees of freedom that it can be controlled with. To that end, feature extraction able to differentiate between even the most similar of distinct contractions is often pursued.

Automatic extraction of features – which, as opposed to "manually"-extracted features (like ones found in Table 3.1 and, as can be seen, often involve metrics from amplitude or spectral analysis), don't require as much time and effort spent by the experimenter – is made possible by way of algorithms that make use of the specific amplitude or spectral informations of EMG, like temporal-spatial descriptors [124], and even more so by deep learning methods such as convolutional neural networks (CNN): Geng et al. [118] adopted a ConvNet architecture to classify

Table 3.1: Some of the most commonly used "manually"-extracted features, in the time and frequency domains, for traditional machine learning-based EMG classifications.

Time domain features	**Frequency domain features**
Integrated EMG	Mean frequency
Mean Absolute Value	Median frequency
Modified mean absolute value 1	Mean power frequency
Modified mean absolute value 2	Peak frequency
Root Mean Square	Total power
Variance	Frequency ratio
Waveform length	Power spectrum ratio
Zero crossing	The power spectrum deformation
Slope sign change	Variance of central frequency
Willison amplitude or Wilson amplitude	Signal-to-motion artifact ratio
Kurtosis	Signal-to-noise ratio
Skewness	Spectral moment
Moving Approximate Entropy	Energy
Fuzzy approximate entropy	Wavelet decomposition
Simple square integral	Wavelet decomposition difference
v-Order	Modified mean frequency
Log detector	Modified median frequencies
Average amplitude change	Short Time Fourier transform
Difference absolute standard deviation value	
Mean absolute value slope	
Multiple hamming windows	
Multiple trapezoidal windows	
Histogram of EMG	
Auto-regressive coefficients	
Cepstral coefficients	
Standard deviation	
Cepstral coefficients	
Sample entropy	
Integral absolute value	
Variance	
Maximum Amplitude	

HDsEMG images (as 2D arrays) of 8 gestures, to intrasubject accuracies from 89%, with only one image, to 99%, with windows (at a 1000-Hz sampling rate) of 40 images, proving the feasibility of employing HDsEMG images for CNN inputs. From their work, adaptations to the architecture have been pursued, based on Deep Domain Adaptation for getting better inter-session gesture recognition [125], on multi-stream CNN (with a "divide-and-conquer strategy") that learns the correlations between individual muscles and specific gestures [126], on 3D CNNs over 2D CNNs, for capturing temporal information simultaneously with spatial features [127], on hybrid models like an attention-based CNN-RNN architecture [128] and CNN/Bi-LSTM (bidirectional long short-term memory) architecture [129], and multiscale time-frequency information fusion representation used with a deep belief network classifier [130] (though done with sparse electrodes, not

high-density), both also able to extract temporal information, and on Light CNN+LSTM [131] adequate for being inserted in small-storage space chips embedded in prosthetic hand hardware; there have also been Deep CNNs [132]; shallow CNNs [133]; other lightweight CNNs [134] like the All-ConvNet by Islam et al. [135], composed only of convolutional layers, reducing learning parameters but still effective for learning discriminative features; CNNs that process three individual parts of previously-divided signals and concatenate the outputs after extracting features (as done with Peng et al.'s [136] termed multi-stream feature fusion network (MSFF-Net)); employment of genetic algorithms for getting the best hyperparameters of the used CNN [122], leading to increases in 8-gesture identification accuracies from 91.86% to 96.4% in real-time, and 99.6% offline;

Beside deep learning architectures, more traditional machine learning approaches using SVM classifiers (often requiring less computational efforts), for instance, have been used, but, instead of with manual feature extraction, with automatic extraction tools, like Histogram of Oriented Gradients (HOG), often used for object recognition, and used by Islam et al. [137] to extract feature vectors from the training HDsEMG images, in hand gesture recognition. The process begins with gradient estimation, where convolution with specific filters yields the magnitude and orientation of each pixel in the sEMG image. The key insight is that the cumulative statistics of the spatial distribution of gradient orientations and magnitudes over small regions provide robust descriptors of the instantaneous sEMG image. HOG feature extraction involves dividing the gradient information into non-overlapping rectangular cells and blocks. Each block contains an orientation histogram with a specified number of bins. The orientation histograms are calculated for each cell, and the resulting HOG feature vectors are formed by combining these histograms. Additionally, trilinear interpolation is applied to smooth the distribution of gradients across cells and reduce aliasing effects. Normalization is performed on overlapping blocks individually to handle variations in gradient strengths across the sEMG image. The final normalized HOG representation, a 588-dimensional feature vector, is used for the instantaneous sEMG image classification, where parameters such as cell size, block size, and the number of bins are critical for extracting discriminant features.

Other studies have involved proportional estimation of finger movements [138]; of thumb position [139]; of elbow torque [140], incorporating information from a biodex machine; of fingertip forces [141] like in Jian et al.'s [142] work, who in the process also showed that deep forest models are highly robust against high-density sensor noise; and of finer hand-muscle contractions like in the case of Tam et al. [115], who also proposed a novel frequency-time-space cross-domain pre-processing method reducing the variance that is observed in the data of a single gesture – the assumption that EMG signal is a zero-mean Gaussian process results in the observation of weak correlation between consecutive frames of the same gesture, which leads to high variance in the data. This method is such that, after band-pass filtering, analog-to-digital conversion and sampling, digital high-pass and notch filtering, there is EMG signal rectification of each channel (nullifying the zero-mean assumption and lowering background noise from inactive muscle areas), followed by a Teager-Kaiser Energy operator which works as an envelope filter to reduce baseline noise and

improve the SNR: in HDsEMG maps, this is reflected in starker contrast between inactive and active muscle areas; finally, a smoothing moving average for reducing fluctuation reflected in better active region consolidation – all steps and resultant effects on HDsEMG colormapping depicted in Figure 3.1. This enhances gesture-specific data homogeneity and leads to more reliable muscle activation maps which, after batch normalization and when fed onto a lightweight CNN architecture for low computing resource usage (work done within the scope of an embedded application), provided 98.15% accuracy by majority vote for 6 and 8 classes of two different gesture sets during offline testing, and 96.83% for real-time operation of the 32-electrode HDsEMG sensor.

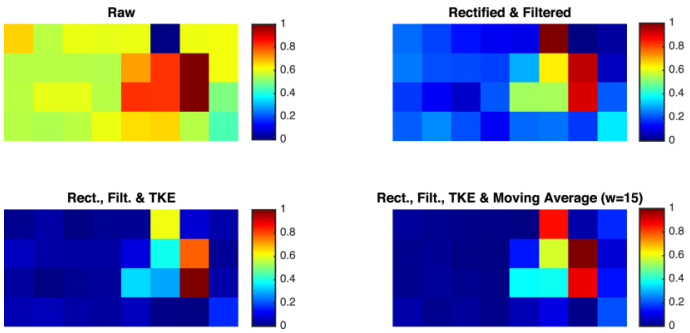


Figure 3.1: HDsEMG activity maps for different gestures upon each of Tam et al.'s proposed preprocessing steps, with each pixel corresponding to one channel. Muscle activity is such that dark blue = 0 (low) and dark red = 1 (high); from [115].

More pre-processing techniques have been studied in the scope of improving CNN-based classification accuracy: from Houston et al.'s [143] comparisons, it was concluded that accuracies from baseline monopolar signals outperformed ones from bipolar-converted signals, fully rectified signals, signals re-referenced to the common average, and signals subjected to Laplacian spatial filters.

To address artifacts induced by motion or electrocardiogram interference, ambient noise, cross talk, and baseline shifts, among other factors, various denoising techniques for sEMG signals have been proposed to enhance the overall recording quality. These methods include Gaussian filtering [144], Hampel filtering [145], sinusoidal modeling [145], adaptive and notch filtering [146], enhanced squirrel search algorithm [147], Median filtering [148], and Wiener Filtering [149].

For signal and electrode variability, Moin et al.'s [150] classification algorithm – processing EMG features in one-shot learning, based on brain-inspired high-dimension computing [151] – led to classifications tolerant to noise and electrode misplacement: whereas typically inter-session gesture classification accuracies may degrade by more than 30% when using other classifiers (like SVM [152]), Moin et al.'s only degraded 7% between consecutive sessions (going from 96.64% to 89.19% for five hand gestures). This means that training the model in session 1 would still be mostly adequate for use in session 2, though, in session 3, results were not satisfactory, meaning that the model may require remapping of the input features during preprocessing.

More works have sought to improve myoelectric pattern recognition by assessing signal quality [153] or improving it through denoising [149], and by reducing effects of electrode shift (and damage [154]) specific to HDsEMG arrays: using a dilated convolutional neural network [155], which enlarges global view of the network thereby becoming more robust against pixel shift (consequence of electrode shift) of the images; by adaptive electrode calibration [156]; by including new features like the variogram, which measures degree of spatial correlation [157]; by a CNN deep transfer learning approach by [158] that also has the advantage of not requiring that the electrode array be high-density; by adding synthetic electrode shift (*i.e.*, using only part of the array as input, and leveraging for augmentation the electrodes that were not included (representing shift)), thereby preventing a 20% loss of accuracy of the used 3D CNN classifier, work done by Sun et al. [159].

Also to tackle universal adversarial perturbation attacks found within the EMG signals and to which CNN are very vulnerable: by calculating the Chebyshev distance between adjacent channels of HDsEMG, using it as a correlation feature, and appending a binary classifier in the entire framework to detect these perturbations, a study by [160] added perturbations values to different channels, so as to decorrelate and make adjacent channels' signals less similar. Doing this allowed for detecting over 91% of attacks, and presumably may be dealt with before feeding the HDsEMG signals into the CNN, ensuring increased safety of the system.

Beyond hand gestures, many works have explored complex movements that are described by the superposition of several simpler ones – by giving multi-labeled classifications, as was the case with Olsson et al.'s [161] work with image sequences onto a deep CNN, using 16 degrees of freedom (16 different labels) to model the movements of the hand and forearm state (leading to a mean exact match rate of 78.7% across 14 healthy subjects), other body motions like in the shoulders [162], muscle activation onset detection – as done with horses by Andrei [163], who used CNNs and LSTMs –, and even silent speech decoding as done by Chen et al. [164], whereby, from a dataset of 15 subjects with two arrays placed on the buccinators, masseter and orbicularis oris muscle and another array on the laryngeal muscles (64 electrodes in total), signals captured upon silent reading of 33 phrases were fed into the proposed CNN-BiLSTM (bidirectional long short-term memory) neural network for extracting discriminative spatio-temporal feature representations and a connectionist temporal classification (CTC) decoder for syllable-level decoding, followed by a Language Model algorithm for further improvement – with a phrase classification accuracy of 95%, showing promise in tackling some of the shortcoming of the field of automatic speech recognition, which does not cater to people unable to speak due to laryngeal disease, for example. In the same vein, a work by Zhang et al. [165] seeking to develop stretchable HDsEMG electrodes for application on complex epidermal surfaces allowed for a CNN-based (with three convolutional layers, residual blocks, two pooling layers, and two fully connected layers) classification of eleven swallowing activities with the array placed on the swallowing muscles, leading to an average of 80% accuracy, showing its potential for use in diagnosing conditions like dysphagia.

While most deep-learning works use HDsEMG images to train the model, the neural drive information obtained from MU decomposition has also been used, particularly by Montazerin et al.

[166], whose Hybrid Transformer-based Architecture for Fusion of Macroscopic and Microscopic Neural Drive Information (HYDRA-HGR) uses attention mechanisms to learn from peak-to-peak values of extracted MUAPs; once fused with other temporal and spatial features (through a Fully Connected fusion layer), it was also used to classify hand movements with an average accuracy of 94.86%. Furthering their work, they also created Compact Transformer-based framework (CT-HGR) [167], which, while similar to the former, can extract the temporal and spatial features in parallel with the MUAPs. Zhao et al. [168] did similar work, using a relatively simple CNN.

Chen et al. [169], by reconstructing MUAP waveforms in space and employing an SVM to classify them, and Xie et al. [170], by using a spiking neural network (SNN) (whose ability to better process pulse signals allowed for better decoding of MUSTs) to process the decomposed MUAPs, also classified hand gestures (to 95% and 88% accuracies, respectively).

Gestures performed by healthy subjects, as opposed to subjects with degenerative conditions such as Duchenne muscular dystrophy (DMD), will impact the distribution results (as assessed by Nizamis et al. [171] (using a LDA classifier, upon consideration of features such as MAV, zero crossing, slope sign change and waveform length), who found that, despite similar patterns across gestures, fewer electrodes at the forearm were activated by DMD patients during gesturing tasks compared to healthy control). Therefore, signals acquired for a dataset and a trained model's validity will be heavily influenced by the participants' characteristics, anatomical and physiological.

This means that, in the context of rehabilitation, improvement of cross-user variability is important, having also been explored in works such as Li et al.'s [172].

3.1.2 Detection, diagnosis, monitoring, prognosis

As mentioned, gestures performed by healthy subjects, as opposed to subjects with degenerative conditions, will lead to different distributions, which makes distinguishing between people with certain pathologies possible with HDsEMG-integrated machine learning, given the adequate dataset – while not high-density, several sEMG pathology datasets have been developed using sparse electrodes – like the "N2001" found in the open-source database EMGLab [173], with a group of ten normal individuals, a group of seven patients with myopathy and a group of eight patients with ALS –, often with classification accuracies of above 95% [174; 175; 176; 177; 178] and suggesting that HDsEMG, as an extension of the former, can make pathology detection feasible.

A recent work by Bitar et al. [179] used a 64-monopolar electrode HDsEMG matrix to record different gestures as performed by healthy individuals and those with DMD; by extracting 22 features (*e.g.*, waveform length, MAV), it was possible to pre-process them and perform outlier removal and feature selection. This provided features that improved gesture recognition for DMD patients, also implying that, through a similar pre-processing pipeline, one could identify which individuals the features belonged to, thereby differentiating between healthy and DMD. This specific "differentiation" study, however, was not explored.

When it comes to monitoring, HDsEMG with machine learning has been used by Moctar et al. [180] to confirm active aging: from 16 discriminative time and frequency domain features, obtained after pre-processing based on Canonical Correlation Analysis (CCA) between sEMG

signals, a dataset composed of subjects of different age groups performing specific physical activities allowed for training a K-NN classifier capable of predicting individuals' motor functional age with a 72% accuracy. From this information, one is able to deduce whether an individual is maintaining an active lifestyle, for instance.

Currently, there appears to be no muscle disease diagnosis-related research directly using HDsEMG alongside machine learning. However, it can be expected that, in replicating some of the approaches found with other studies (often times using sparse sEMG instead) pathology detection results may be similar, if not improved. Additionally, metrics such as conduction velocity and motor unit recruitment rate have been used to help in diagnosis and monitoring; in that scope, machine learning has indeed been used with HDsEMG for metric extraction, as is discussed in 3.1.3.

3.1.3 Metric Extraction

In recent years, new architectures developed for HDsEMG signal decomposition and MU identification (often with the purpose of better assessment of neural control of muscle contraction) have shown promising results. An evaluation by Urh et al. [181] was able to detect MU firings of simulated HDsEMG signals (whose MU firing patterns were known, and used for training) where inputs were the samples of the signal and where binary outputs of the trained model would correspond to a discharge, while also concluding that, for this task, recurrent NNs (*e.g.*, LSTMs) may be more sensitive to noise than non-recurrent NNs. Another work by Clarke et al. [107] proposed a framework for decomposing HDsEMG into MUSTs by a gated recurrent unit (GRU) network (whose input was z-scoring based standardized HDsEMG data and whose output was then converted to time stamps) trained on the time series output of the gradient-version of the CKC decomposition algorithm (gCKC), to a median rate of agreement (RoA) between gCKC and GRU of 94.5% on unseen test data. Similar work by Wen et al. [108] was done, instead using a Deep CNN whose input was a window of the HDsEMG signals, having been able to identify as many MUs as the CKC algorithm (in this case, 17 MUs) with a precision of 95% and sensitivity of 97%.

Further research by Urh et al. [182] concluded that whitening as a pre-processing step in neural network-based MU decomposition is significant: for a dense NN, precision of MU identification went from 32.2$\pm$20.2% (non-whitened signal) to 93.1$\pm$8.7% (whitened signal); for a LSTM NN, it went from 59.7$\pm$19.7% to 99.4%$\pm$2.0%, evidencing the importance of spatial whitening, for the used neural networks.

3.2 Commercially and Non-Commercially Available Software Options

Currently, many programs have been developed for the express purpose of EMG amplitude analysis, as is the case of software such as NeuroKit2 or EMG Tools. In the market, OriginLab's EMG

Toolbar and Motion Lab System's EMG Analysis software are common adoptions for motion and timing analysis. An open-source option is that of GallVp's emgGO, available for Matlab [183].

Similarly, spectral analysis can often be pursued within Matlab using packages such as Wavelab 850 made for wavelet analysis, DiscreteTFDs for time-frequency computations, LastWave allowing for fractal analysis, all of which open-source, though developed within Matlab or Octave. This implicates licensing which The Large Time/Frequency Analysis Toolbox (LTFAT) and the Time-Frequency Toolbox (TFTB) are free ones that, being under the GNU General Public License, allow for users to freely use them, modify them and distribute them, though any change must be made open-source as well.

An additional Matlab-implemented open-source toolbox has been developed by Jiang et al. [184], provided alongside their datasets, known as "Hyser", made with the intention of processing their own HDsEMG datasets for pattern recognition and analyzing them by way of linear discriminant analysis (LDA) and CNNs, allowing for deep learning-based hand gesture classification with average accuracies of over 88% for dynamic and maintenance tasks, while also providing EMG-force regression, and performing MU decomposition of HDsEMG signals into spike trains, via ICA (specifically following Negro et al.'s [94] FastICA/CKC approach), validated by the 'silhouette distance values' indicators (SIL) (implying good decomposition performance [185]).

In dealing with conduction velocity estimation, the GNU GP-licensed "openhdemg" toolbox is one that does so by using MLE, STA-based cross-correlation[186], though commercial options such as BioSemi's MyoLab and OT Bioelettronica's BioLab+ are also capable of CV estimation, with the latter being often used for that purpose [187]. The two options, alongside DELSYS' EMGWorks and NeuroMap are the more prominent software available in the market, though most requiring the HDsEMG hardware developed by their respective companies. They are also capable of providing MU decomposition, though other commercial options like DEMUSE (which uses the CKC method), and other open-source options such as Neuromechanist's hdEMG Decomposition tool [188] (one that implements the FastICA method), software such as EMGTools [189] and Hyser are also capable, being often adopted for research work.

Made specifically for analyzing MUSTs, there are the MUedit, developed by Avrillon et al. [190], and tweedell's open-source motoRneuron package [] for the R programming language, both of which allow for identification of discharge timing, as well as manual editing of detected MUSTs.

Known to have used machine learning, there are tools such as Bio-SP, which does automatic extraction of features meant for being fed into ML models; BioSPPy, with an integrated tool for biometric recognition from EMG signals; and the ReSurfEMG python library made for respiratory EMG analysis, and within which there is a trained model capable of classifying different types of breathing. Common approaches for exploring ML within-software include employing known packages such as PyTorch, TensorFlow and Scikit-learn. In the previous cases, these can be used with the added benefit of being specifically EMG-focused.

Chapter 4

Methodology

This chapter begins by identifying the approaches adopted for validating algorithms: particularly, which synthetic signals were used and how they were simulated, and which datasets were used and their descriptions.

Afterward, each section discusses the methodology for implementing the algorithms developed for the HDsEMG software, from standard amplitude and spectral analyses, to more intricate CV estimation and MU decomposition.

Finally, two machine learning-based pipelines (ML applications) are explored, with the way to implement and deploy them described herein.

4.1 Validation and Testing Approaches

In software development, particularly signal processing and clinical evaluation software, validating each algorithm is essential to ensure that each one is both reliable in providing accurate results (be they processing a signal, or estimating a metric) and computationally efficient at doing so, and each section of Chapter 4 delineates how to best pursue their corresponding validations through functions whose ground-truth is known (from mathematical equations to simulated, synthetic signals, the latter further discussed in Subsection 4.1.1).

Alongside these approaches also come tests using real signals, obtained from what is herein called "benchmark datasets", discussed in 4.1.2 – while their ground-truths may not be known, they allow for better understanding of the effect of the processing tools on actual electrode-captured data, and also for comparison of results with other existing algorithms.

4.1.1 Simulated Signals

For generating signals, two approaches were taken: the first one generates pulses by defining an action potential shape and statically propagating it in time depending on current channel and distance to source channel (useful for creating simpler signals with fewer firings with known CVs and action potential duration).

The second approach also generates synthetic action potentials (as obtained in Figure 4.1), while also allowing for customizing many fiber properties (number, length, diameter, etc.) and motor unit properties (like density, depth, and size) to effectively simulate surface EMG, developed by J. Djärf and A. Alosta[191], publicly available, and made for use in investigation studies to allow for assessment of the performance of decomposition algorithms for sEMG (known spike trains, for instance, as shown in Figure 4.2) which are generated following the parameters given to the function; Figure 4.2 shows 5 steps from which one is able to obtain the synthesized signals' characteristics, including MUAP shapes on all channels (an estimation made from the signal found within a time window surrounding each firing), instantaneous firing rate and signal intensity, of a given MU.

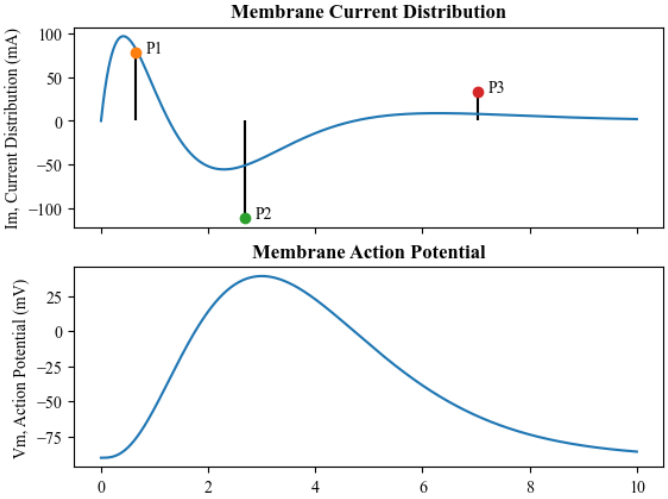


Figure 4.1: Example of the signal's characteristics describing a simulated action potential.

With that, these simulations allow for the definition of grid shape, the channel in which signal is generated (analogous to innervation zones), firing rate, MUAP duration, number of MUs, conduction velocity, sampling rate, IED, among others: changing signal complexity, as was done (some examples shown in Figure 4.3) for validating CV and IZ estimation algorithms.

For further assessing the effects of different signal behavior, different IZs were defined within the grid (as in Figure 4.4a), for various grid shapes. Signal-to-noise ratio (SNR), simulated by way of adding independent white Gaussian noise to each element of the input signal array, given an SNR value,

$$P_{\text{signal}} = \frac{1}{N}\sum_{i=1}^{N} X_{i,j,k}^2 \tag{4.1}$$

to calculate the mean power of the signal X; followed by conversion of SNR from dB to linear scale,

$$\text{SNR}_{\text{linear}} = 10^{\left(\frac{\text{SNR}_{\text{dB}}}{10}\right)} \tag{4.2}$$

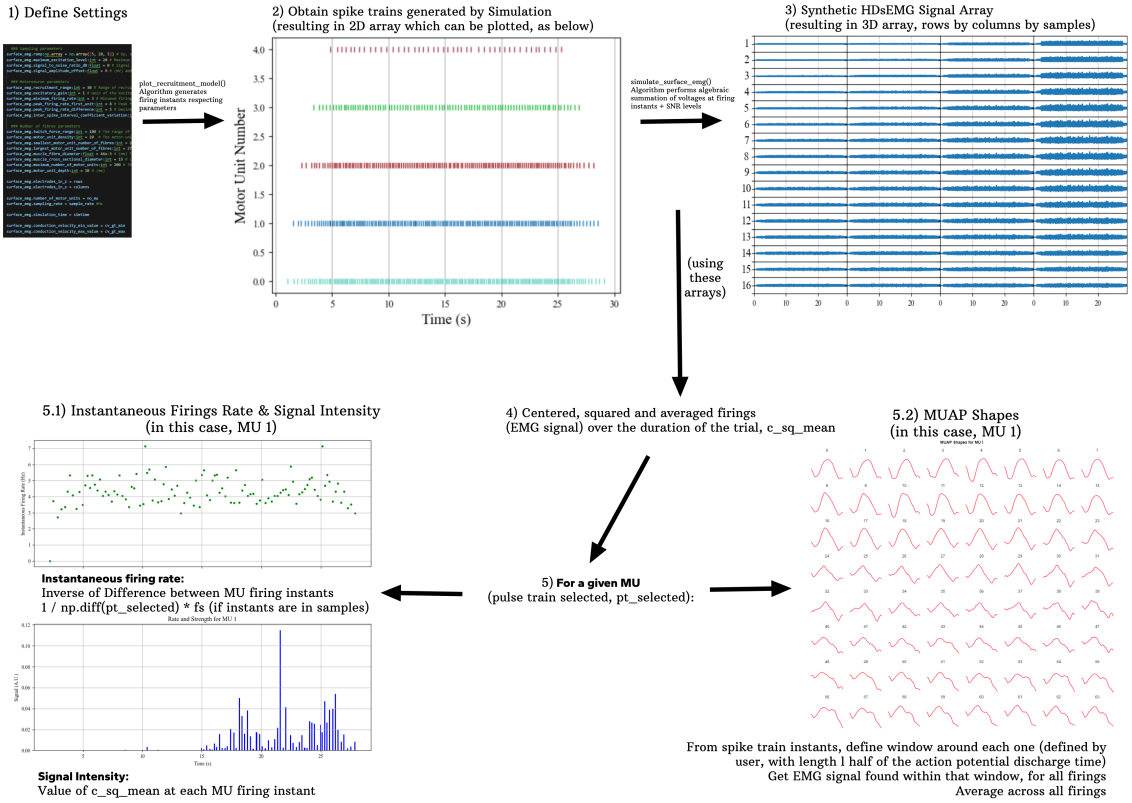


Figure 4.2: Process describing approach for synthetic signal generation and visualization, and acquisition of MU firing rates, signal intensity, and AP shapes. In this case, a 16 by 4 grid of signals from 5 MUs.

and then, computing the power of the noise and its standard deviation,

$$P_{\text{noise}} = \frac{P_{\text{signal}}}{\text{SNR}_{\text{linear}}} \tag{4.3}$$

$$\sigma_{\text{noise}} = \sqrt{P_{\text{noise}}} \tag{4.4}$$

and finally, adding the noise to the signal (Y the now-noisy signal),

$$\mathbf{Y} = \mathbf{X} + \mathbf{N}, \quad \mathbf{N} \sim \mathcal{N}(0, \sigma^2_{\text{noise}}) \tag{4.5}$$

And dampening of the signal, simulated by way of an exponential decay, described by

$$\text{distance} = \sqrt{(x - \text{source}_x)^2 + (y - \text{source}_y)^2} \tag{4.6}$$

$$\text{decay_factor} = e^{-\lambda \cdot \text{distance}} \tag{4.7}$$

$$\text{dissipated signal}[x, y, :] = \text{signal}[x, y, :] \cdot \text{decay_factor} \tag{4.8}$$

with (x, y) the coordinates in the HDsEMG grid; $(\text{source}_x, \text{source}_y)$ the coordinates of the source of decay; λ the decay rate.

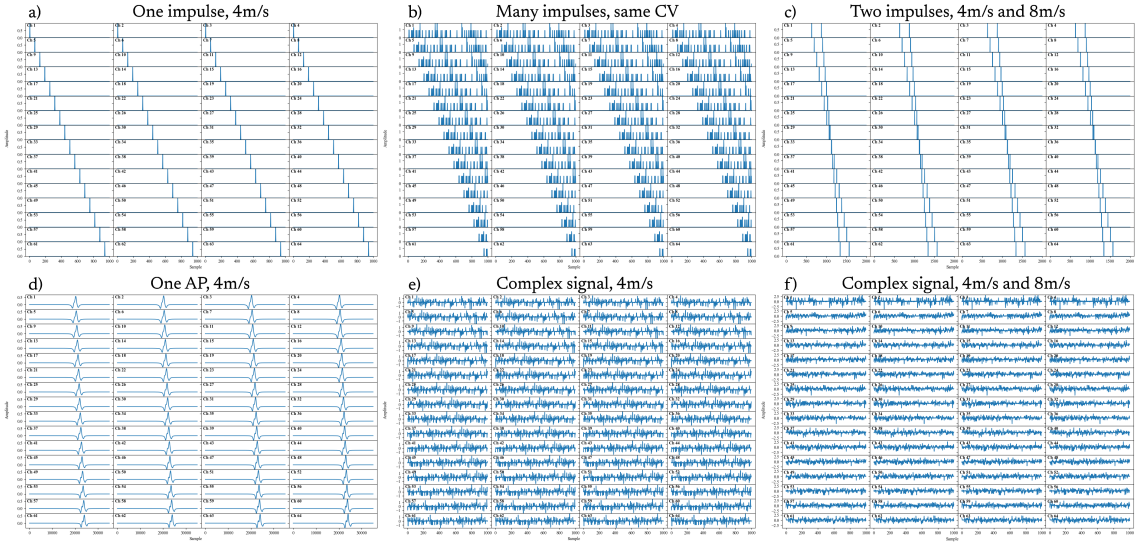


Figure 4.3: Display of differently simulated 2D signals, from (a) one impulse propagating row-wise through time, (b) many impulses with same CV, (c) two impulses at two different velocities, (d) the propagation of an more accurate AP, (e) static propagation of complex signal at 4m/s and (f) of two complex signals at 4m/s and 8m/s, respectively.

These result in signals akin to the ones present in Figure 4.4.

4.1.2 Benchmark Dataset of Real Signals

In order to analyze amplitude and spectral features, as well as obtain CV and MU trains from real-life scenarios of clinical relevance, two datasets were utilized: the first one, herein called 'FOG Dataset', due to its sparse electrodes (*i.e.*, not a high-density grid) was used solely for pre-processing, amplitude and spectral analyses, as well as for a traditional machine learning-based application.

The second one, herein called 'DMD Dataset', this one containing signals from a HDsEMG grid, was indeed used for all aspects, as well as for a deep-learning-based application.

With that, this subsection will describe each dataset more intricately.

4.1.2.1 FOG Dataset: Freezing of Gait in Parkinson's Disease

The FOG Dataset, created by W. Zhang, et al.[192]'s, was made for investigation studies and is publicly available. The data comprises 12 PD patients' signals (3 of EMG, 25 of EEG, among others, like ECG and acceleration values), to a length of 3 hours and 42 minutes of valid data: 2 hours and 14 minutes of normal gait and 1 hour and 28 minutes of freezing of gait (the latter labeled by two qualified physicians) across four tasks (including quarter turns, U-turns, obstacle bypassing). For this work, only the EMG signals were considered — recorded at a sampling rate of 1 kHz using a 32-channel wireless 'MOVE' system, with the electrodes sparsely placed at

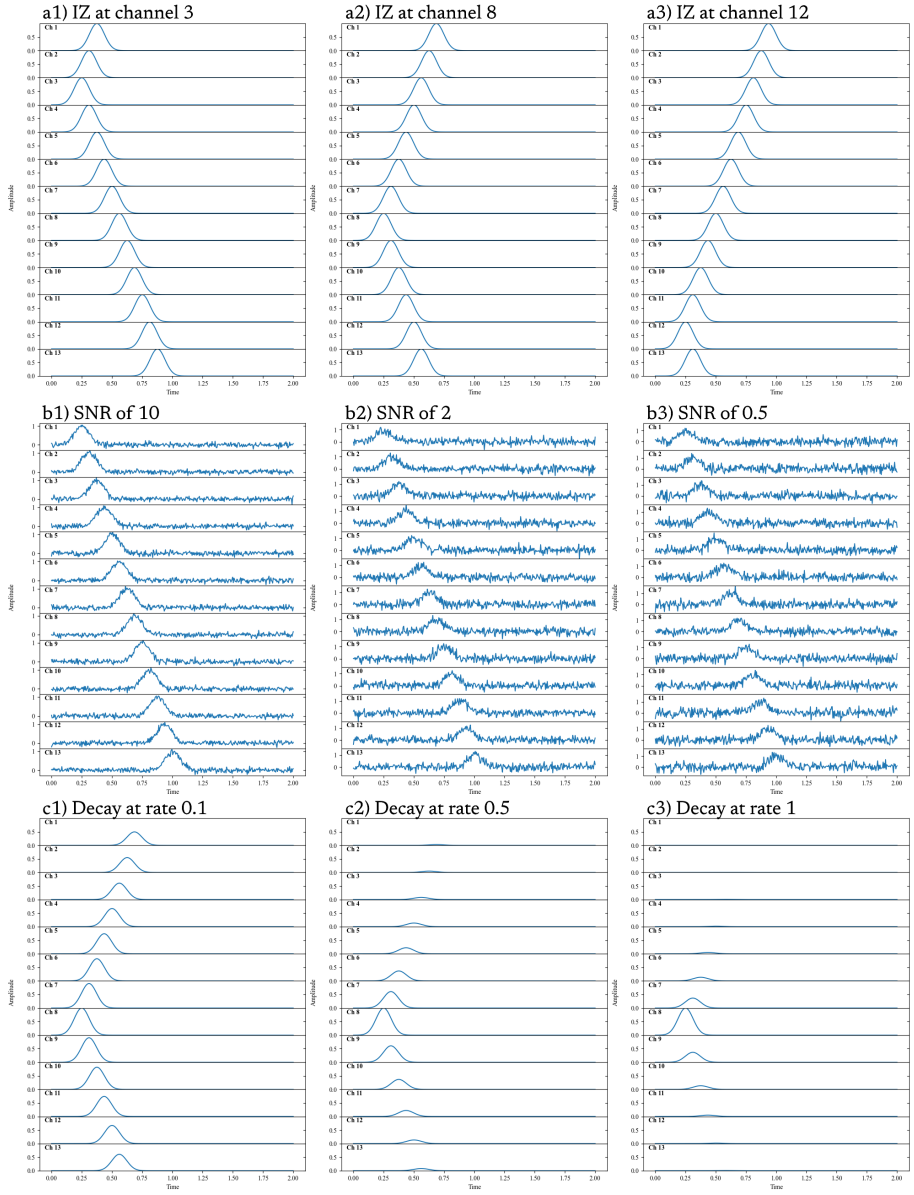


Figure 4.4: Display of one simulated action potential propagating through a column grid, and its effects upon three different (a) IZs, (b) SNRs, and (c) decay rates.

three surface leg muscles — the gastrocnemius of the right leg, and the tibia anterior of both legs; anatomy shown in Figure 4.5; the three unfiltered sEMG signals acquired during task 2, by one of the subjects (subject 10), are shown in Figure 4.6.

For testing analytical algorithms, only subject 1 performing task 1 was adopted (with the three EMG channels (EMG1, EMG2, EMG4)). On the other hand, for the ML application, all subjects and tasks were taken into consideration, with all taken as the train subjects, save for subjects 10 and 11, used as test.

The signals (EMG channel 1) acquired by subjects 10 and 11 are shown in Figure 4.7, across the four tasks, also highlighting the two FOG states.

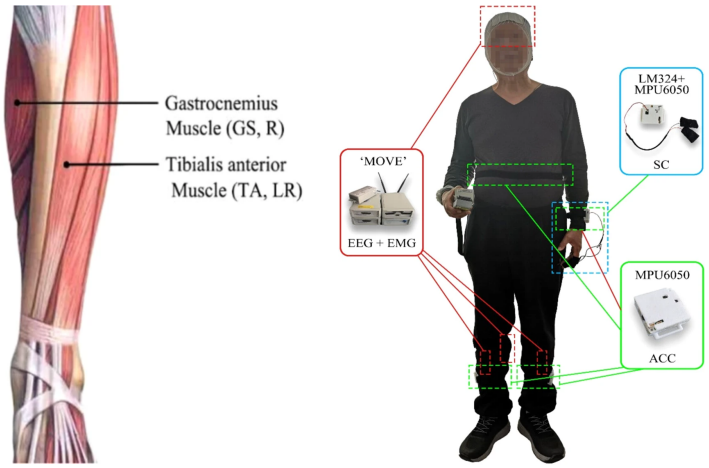


Figure 4.5: Images of the FOG dataset's (a) leg muscle anatomy and (b) correct electrode placement on the surface of skin and other equipment placement, from [192].

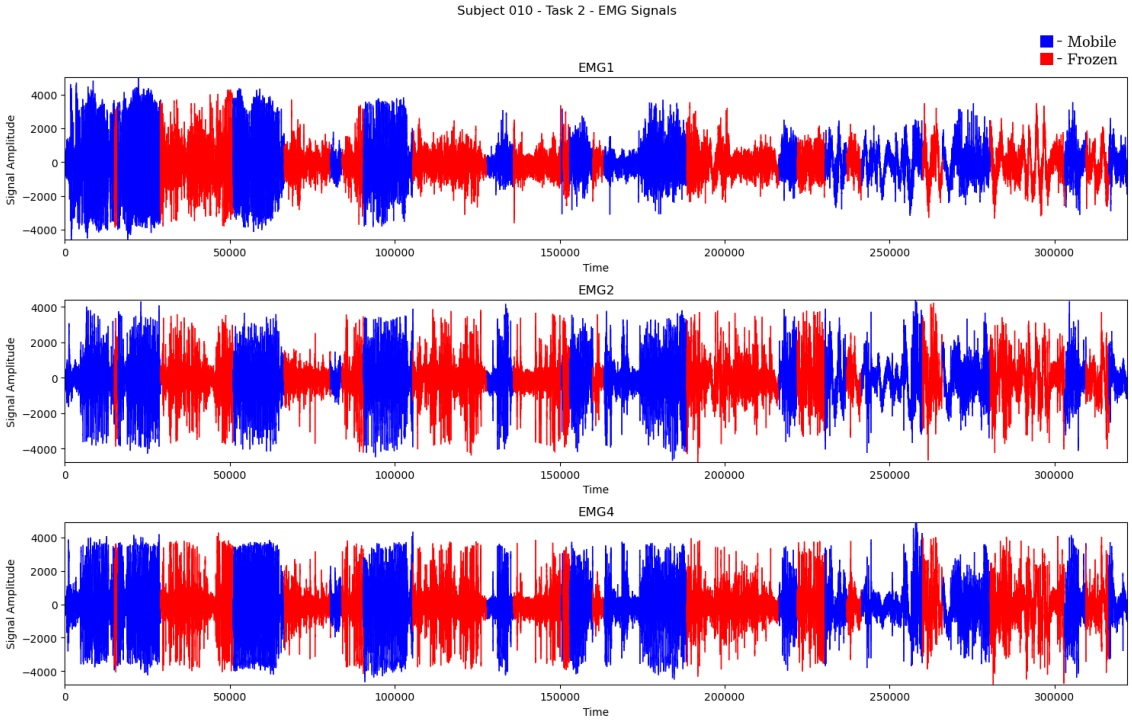


Figure 4.6: Entire timeline of the FOG dataset's signals (EMG1, EMG2, EMG4) acquired from subject 10 during task 2; x-axis representing time in number of samples (10000 samples corresponds to 10 seconds, for instance), y-axis representing unnormalized signal amplitude (original units unspecified).

Because during a freezing episode, a PD patient may experience an inability to initiate or continue movement, and due to the often abnormal muscle activation during these episodes – conditions further explained in Section 2.1.5.1 – it proves of relevance to study these EMG signals and assess if the tools employed detect what is to be expected.

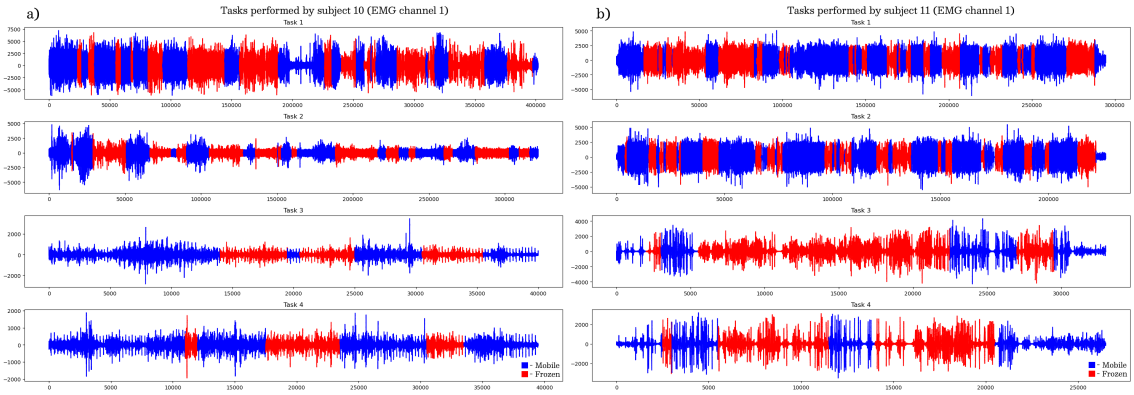


Figure 4.7: Entire timeline of the FOG dataset's signal (EMG1) acquired from subjects 10 and 11 during all 4 tasks; x-axis representing time in number of samples (10000 samples corresponds to 10 seconds, for instance), y-axis representing unnormalized signal amplitude (original units unspecified).

4.1.2.2 DMD Dataset: Healthy and Duchenne Muscular Dystrophy Patients

The DMD dataset, created by K. Nizamis, et al.[171] (for use in human-computer interaction studies and which is publicly available) involves sEMG signals acquired – at 2048 Hz, by way of 64 monopolar surface electrodes placed 8-by-8 around the forearm, shown in Figure 4.8 – upon the performance of 7 hand/wrist gestures (open hand, closed hand, pointing, thumb extension, thumb flexion, wrist extension and wrist flexion), both by 3 Duchenne muscular dystrophy patients and 8 healthy individuals. Each gesture was performed for 3 seconds, 10 times, with 3-second rests in-between.

Though the values from each file are in μV, one may also normalize them: for every subject there is also a MVC trial signal file, wherein all tasks are performed once more, at full contraction.

Furthermore, for the faulty electrodes indicated in each file, a replacement of its signals (or lack thereof) was done by interpolating with respect to surrounding channels in each recorded instant.

For evaluating algorithms, from topographical mapping, to CV estimation, to MU decomposition, DMD patient 3 and Healthy patient 2, performing wrist extension and wrist flexion, were considered – this way, one is able to compare activities not just between two different tasks, but between control and pathology; Figure 4.9a shows the raw, unfiltered signals produced by both patients during wrist flexion, with 4.9b closed in on a single activation, after pre-processing. On the other hand, all tasks and most subjects were considered for the DL application.

4.2 Pre-Processing Tools

In the development of the sEMG signal processing software, a variety of pre-processing tools were employed to enhance the quality of the raw sEMG signals and prepare them for subsequent analysis. These tools include the Butterworth filter, notch filter, detrending, rectification, normalization

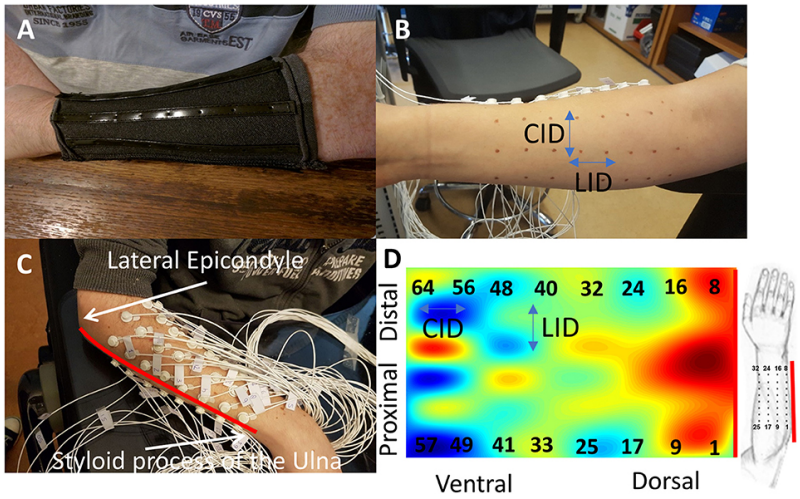


Figure 4.8: Signal acquisition setup for the DMD gesture dataset, showing (A) the custom sleeve, flexible around the arm but stiff along its length; (B) the skin, marked with a 2 cm gap between electrodes along the arm, and a variable gap around the arm based on its width; (C) the 64 electrodes on the participant's arm, with a red line marking the boundary between the top and bottom of the forearm. Electrode one is placed 20% down from the elbow, with the rest placed in a counterclockwise direction on the right arm; (D) the 1-32 electrodes on the top, and 33-64 on the bottom of the forearm, also showing the center of gravity for the gesture; from [171].

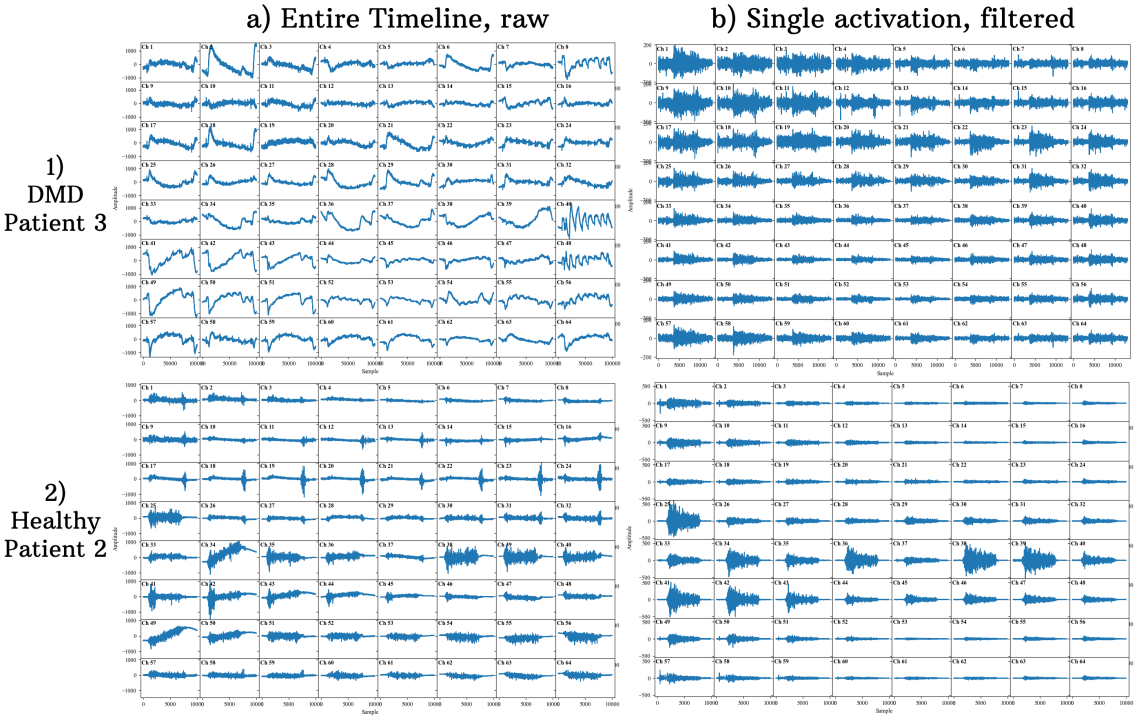


Figure 4.9: Display of HDsEMG signals acquired, in μV, by the (1) DMD and (2) healthy patients during wrist flexion gestures, with (a) entire timeline on raw signals, and (b) a close-in on a single activation (3-second flexion) on pre-processed signals, from DMD dataset; x-axis representing time in number of samples (10000 samples corresponds to roughly 5 seconds, for instance).

(including MVC-based calibration normalization), resampling, time shifting, and signal-to-noise ratio (SNR) calculation.

While, in-software, each tool can be used individually, a suggested order is displayed in the pipeline's flowchart found in Figure 4.10.

Hereafter is a detailed description of each pre-processing tool implemented, along with the pertinent packages and methods used.

The Butterworth filter was implemented to remove unwanted noise and interference from the raw sEMG signals. Using the *scipy.signal* library, the Butterworth filter was designed to have a maximally flat frequency response in the passband, minimizing signal distortion. The filter can be configured as low-pass, high-pass, band-pass, or band-stop, depending on the specific requirements. Key parameters, such as cutoff frequency and filter order, were carefully chosen based on the characteristics of the sEMG signals and the demands of subsequent analyses.

A notch filter was used to remove specific frequency components, primarily to eliminate power line interference at 50 or 60 Hz, thereby improving signal-tp-noise ratio. The implementation utilized the *iirnotch* function from the *scipy.signal* library, which allows for the design of narrow-band filters to target these specific frequencies. The quality factor of the notch filter can be selected manually, though the default of 6 was selected, known to balance the trade-off between attenuation of the interference and preservation of the signal integrity.

Detrending was implemented to remove any linear trend components from the sEMG signals, addressing baseline drift or slow fluctuations, and DC offsets. The *detrend* function from the *scipy.signal* library was employed for this purpose. This step ensures that the baseline of the sEMG signals remains stable and set at 0, facilitating more accurate analysis and interpretation where the signal has now both positive and negative values.

Rectification involves converting the sEMG signals to their absolute values (in the case of full-wave rectification) or excluding all negative values (in the case of half-wave rectification), making all remaining signal values positive. This process is crucial because sEMG signals are typically bipolar. The rectification was implemented using the *numpy* library, which provides efficient computation of the absolute values (*np.abs*) – the full-wave option is more often adopted[4]. This step is a prerequisite for further analyses, such as envelope detection and amplitude estimation.

Normalization is performed to scale the sEMG signals within a specific range, ensuring comparability across different subjects or recording sessions. Various normalization methods were implemented, including min-max normalization,

$$X = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{4.9}$$

with x a data point and X its normalized result, and $x_{\max}$ and $x_{\min}$ the maximum and minimum of the given signal(s), respectively; and its scaled version,

$$X = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \times (M - m) + m \tag{4.10}$$

where min-max is arbitrarily scaled to maximum value M and minimum value m; and z-score

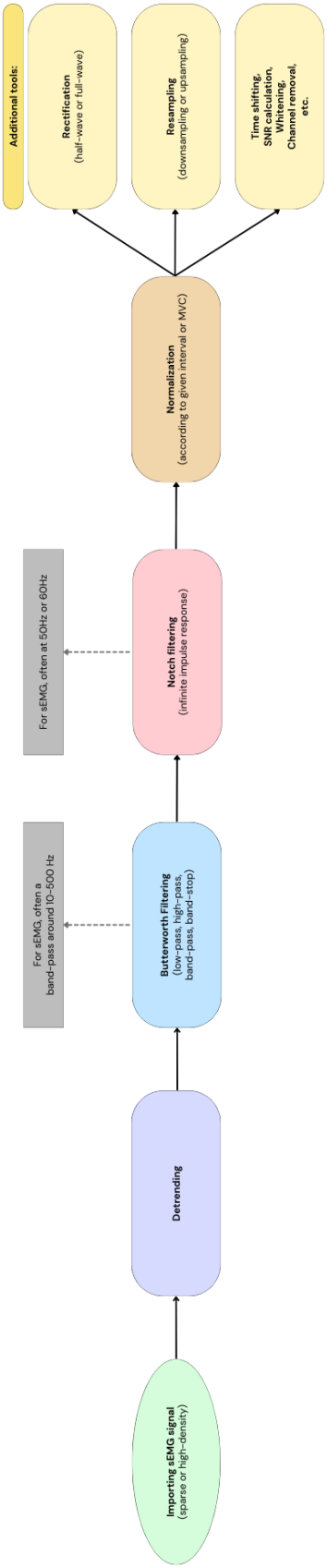


Figure 4.10: Pipeline for standard pre-processing of sEMG signals with the tools implemented in this work.

normalization (standardization to zero mean μ and unit variance σ),

$$X = \frac{x - \mu}{\sigma} \tag{4.11}$$

The *numpy* library was used for these calculations, leveraging its efficient array manipulation capabilities. The choice of normalization method is then dictated by the specific analysis requirements.

MVC calibration normalization is a specialized form of normalization used in sEMG analysis. It involves normalizing the sEMG signals scaled relative to the maximum voluntary contraction (MVC) level of the muscles being studied (whose signals can be acquired when performing a given isometric task at full contraction). This method provides a meaningful reference for comparing muscle activation levels across different tasks or conditions. The implementation relied on calculating the average MVC value (using absolute values of the signal) and normalizing the sEMG signals accordingly, *i.e.*,

$$X = \frac{x}{\text{MVC}} \tag{4.12}$$

Resampling involves changing the sampling rate of the sEMG signals, which can be achieved through downsampling or upsampling. Several resampling methods were implemented using the *scipy.signal*, *numpy*, *pandas*, and *neurokit2* libraries: interpolation – which utilizes scipy.ndimage.zoom() for fast resampling; numpy – which uses np.interp() for straightforward interpolation; pandas – employing time series resampling methods for flexible operations; polyphase – utilizing scipy.signal.resample_poly() for high-quality resampling; FFT – which uses scipy.signal.resample() for frequency domain resampling, offering high accuracy for periodic signals.

Time shifting is the process of shifting the sEMG signals in time to correct for delays or to align the signals with external events or markers. This step is essential for synchronizing the sEMG signals with other data sources. The implementation used basic array slicing techniques to achieve the desired time shift, based on the sampling rate and the amount of shift required,

$$\Delta t = \frac{n}{f} \Leftrightarrow n = \Delta t \cdot f \tag{4.13}$$

so the signal is shifted n samples (rounded to nearest integer), for a desired time shift Δt, at a sampling rate f.

The calculation of the signal-to-noise ratio (SNR) provides a measure of the quality of the sEMG signals. The SNR is defined as the ratio of the power of the signal (computed in this work as being the average of the squared values of the signal, $P_{\text{signal}} = \frac{\Sigma x^2}{n}$) to the power of the noise ($P_{\text{noise}} = \frac{\Sigma z^2}{n}$, with z being the noise estimation – either average of signal when muscle is inactive, or of random values from a Gaussian distribution $Z \sim \mathcal{N}(0, \sigma_Z^2)$ that approximates it), expressed in decibels (dB).

$$SNR_{dB} = 10 \log_{10} \frac{P_{\text{signal}}}{P_{\text{noise}}} \tag{4.14}$$

The numpy library was used to calculate the mean power of the signal and an estimated noise,

facilitating the computation of the SNR. Higher SNR values indicate better signal quality, which is crucial for reliable analysis and interpretation of the sEMG data. Its importance within the pre-processing scope arises upon the comparison of different filtering and artifact removal approaches: evaluation of their effectiveness can be done by comparing resulting SNRs.

These pre-processing tools collectively ensure that the sEMG signals are clean, stable, and ready for further analysis, and are thus a crucial part of any platform that offers the user the ability to work with these signals. By systematically applying these tools, the quality and reliability of the sEMG data are significantly enhanced, facilitating accurate and meaningful interpretations in subsequent stages of the signal processing pipeline.

Validation

In validating the reliability of the used packages and corresponding pre-processing tools, two approaches were adopted: use of test functions seen as ground-truths followed by its visual and numerical assessment after pre-processing step; and visual (quantitative for SNR) inspection of benchmark sEMG signal before and after a given pre-processing step.

For detrending, random values following a (linear) trend along x-axis were created, and a statistical test – the Augmented Dickey-Fuller (ADF) test – was done (used to check for stationarity), obtaining p-value along with it (a lower p-value after detrending indicates that the data is more stationary); residual analysis was also done, wherein it was checked whether the mean of the residuals (original data minus the trend) is found to be close to 0.

For filtering (band-pass and notch), sinusoidal functions with known angular frequency were used to show their attenuations upon each filter usage (essentially, a test signal composed of a combination of low-frequency and high-frequency components); this was done with the *freqs* function of the *scipy.signal* library which gives frequency response of a digital filter, and by inspecting the signals' frequency spectra (through fast Fourier transform) before and after filtering.

Remaining tools were analyzed by numeric and visual comparisons of the benchmark sEMG data; for instance, full-wave rectification tool should lead to all-positive values, normalization should set them to be in relation to a given value (such as MVC) and scaled to a given maximum and minimum; SNR should provide values comparable to those often found in research.

The set of sEMG signals used, from benchmark datasets, is further expounded upon in Subsection 4.1.2.

4.3 Amplitude Analysis

4.3.1 Features

In the development of the sEMG signal processing software, the `amplitude_features` function was implemented to extract various amplitude-related features from the raw sEMG signals. These features provide essential insights into muscle activity and signal characteristics. The function computes the amplitude-related features found in Table 4.1, all of which employ either *numpy*

or the open-source library *SciPy*. Among these features are ARV, RMS and epoch averages, whose relevance is discussed in Section 2.2; a distinction is also made between epochs and moving averages – while the first divides the signal into several windows of epoch-length (no overlap), and computes the features within each window, the moving average allows for the user to define step size (overlaps) as well, thereby further smoothing the resulting envelope.

Features such as minmax, derivative and integral and polynomial approximations can provide further information about the muscle activation patterns, signal characteristics, and underlying neuromuscular dynamics, helping to identify subtle variations in muscle performance, fatigue, and coordination, and are often found within most standard EMG-analysis software.

Table 4.1: Summary of amplitude features computed at every new time epoch.

Amplitude Feature	Explanation	Tools used	Equation
Epoch Average	Computes the mean value of the EMG signal within each epoch.	*numpy*	Epoch Average $= \frac{1}{N}\sum_{n=1}^{N}(x[n])$
ARV (Average Rectified Value)	Measures the average magnitude of rectified EMG signal over an epoch.	*numpy*	ARV $= \frac{1}{N}\sum_{n=1}^{N}\lvert x[n]\rvert$
Mean Square	Calculates the mean of the squared EMG signal values.	*numpy*	Mean Square $= \frac{1}{N}\sum_{n=1}^{N}x[n]^2$
RMS (Root Mean Square)	Determines the root mean square amplitude of the EMG signal.	*numpy*	RMS $= \sqrt{\frac{1}{N}\sum_{n=1}^{N}x[n]^2}$
Moving Average	Applies a moving average to smooth the absolute EMG signal amplitude over specified window sizes and overlaps.	*numpy*	Moving Average $= \frac{1}{W}\sum_{i=0}^{W-1}\lvert x[t-i]\rvert$
Integral	Integrates the EMG signal using Simpson's rule to compute the total area under the signal curve.	*scipy.integrate.simps*	$S_t(x) = \int_{t_1}^{t_2} x(t)\,dt \approx \frac{\Delta t}{3}\sum_{i=1}^{N/2}\left(x(t_{2i-2}) + 4x(t_{2i-1}) + x(t_{2i})\right)$
Derivative	Computes the derivative of the EMG signal with respect to time.	*np.gradient*	Derivative $= \frac{dx(t)}{dt} \approx \frac{x(t_{i+1})-x(t_{i-1})}{2h}$
MinMax and PTP	Identifies the maximum and minimum amplitudes of the EMG signal, along with their time occurrences, and peak-to-peak.	*np.argmax, np.argmin*	$\underset{i}{\text{argmin(max)}}\, x_i = \{i \mid x_i \leq (\geq) x_j \;\forall\; j \in [1,n]\}$
Zero Crossing	Determines the points in time where the EMG signal crosses zero.	*numpy*	$i : x(t_i)\cdot x(t_{i+1}) < 0$
Coefficient of Variation	Computes the coefficient of variation (standard deviation divided by the mean) of the EMG signal.	*numpy*, $\frac{\text{np.std(data)}}{\text{np.mean(data)}}$	CoV $= \left(\frac{\sigma}{\mu}\right) \times 100$
Linear Approximation	Fits a linear curve to the EMG signal data to approximate its trend.	*np.poly1d*	Fit to $y = mx + b$
Parabolic Approximation	Fits a parabolic curve to the EMG signal data to approximate its shape.	*np.polyfit*	Fit to $y = ax^2 + bx + c$ by minimizing $\sum_{i=1}^{n}(y_i - P(x_i))^2$

Some additional features, such as Hjorth parameters and fractional Fourier transform-related ones, can also be acquired, and their use is further explained and explored in Section 4.9, within the context of machine learning.

The function operates by segmenting the input EMG signal into epochs based on the specified epoch time. Within each epoch, the defined features are computed using standard signal processing techniques and mathematical calculations. Notably, the function leverages *numpy* and *scipy* libraries for efficient computation of these features.

4.3.2 Topographical Mapping

For mapping HDsEMG grid, the previous approach is applied to all rows of a 2D *numpy* array, of size (number of channels, number of samples), by using *np.apply_along_axis* on each row. Only then is a reshaping performed (*np.reshape*), such that the channels are distributed across rows and columns (shape of (number of rows, number of columns, number of samples)) – with this 3D array, and using the *Matplotlib* library within *Python*, one can obtain the meshgrid of the x and y

axes (*plt.meshgrid*), and display it with corresponding intensities (*plt.pcolormesh*), along with its color bar (*plt.colorbar*).

To make the transition between colors within the mesh grid gradual (as opposed to discrete), the function *interp2d* from *scipy.interpolate* is used, plotting it with *plt.contourf*.

Finally, the function `plot_sEMG_sample` created for this purpose also allows for parameters such as distance between electrodes in both x and y axes, index of the sample to be plotted, to be specified.

4.3.3 Center of Gravity

Computing the CoG was done by resorting to Equation 2.3 mentioned in Section 2.2.3; as such, at every sample of the 2D signal, a weighted averaging of the signal intensity (be it standard sEMG intensity, RMS, ARV, etc.) is done for each axis separately, by matricially multiplying the 1D-array resulting from the sum of the 2D sample along the axis with the transposed 1D-array of the distances along that axis in each entry; the scalar is then divided by the total, summed up intensity of all channels.

```
cog_x=((np.sum(np.abs(sample),axis=0))@(x.T))/(np.sum(np.abs(sample)))
cog_y=((np.sum(np.abs(sample),axis=1))@(y.T))/(np.sum(np.abs(sample)))
```

Validation

To validate the accuracy of the amplitude features, the function includes options to plot each computed feature alongside the original EMG signal – with the *matplotlib.pyplot* library. This visualization aids in assessing the effectiveness of each processing step and ensures the integrity of subsequent analyses. In the current context, the benchmark sEMG FOG dataset is used for visual and numerical inspection, alongside assessments on usefulness of the features used and what they may represent clinically. Likewise, for assessing 2D mapping and CoG, the DMD dataset serves as a way to understand changes in activation depending on task and patient (healthy or dystrophic). Finding commonalities or lack thereof, both in magnitudes and distributions, allows for validating these tools' ability for clinical assessment.

To note that the packages herein adopted are widely used, well-tested, and trusted in the scientific and data analysis community. *Matplotlib*, *numpy* and *scipy* libraries are open-source, developed and maintained by a large community of contributors, and have been extensively used in academia, research, and industry – because of this, these can generally be used without needing to validate their methods with specific functions; with that, the benchmark datasets are considered in this work as being sufficient for validating and testing.

4.4 Timing Analysis

Onset Detection Approaches

Muscle activation can be detected when muscles are contracted such that the electrodes receive the electrical impulses that derive from said contractions – thereby increasing signal intensity –, and, as mentioned in Subsection 2.2.4, such detection can be done algorithmically through different approaches. In this work, onset and offset detection was implemented through the BioSPPy library [193], which provides seven different approaches to onset detection, as summarized in Table 4.2. To implement these methods, this library employs the *numpy*, *bidict*, *h5py*, *matplotlib*, *scikit-learn*, *scipy*, *shortuuid*, *six*, and *joblib* dependencies, to process the signal and facilitate in the intensification of active states, when compared to inactive ones, though the thresholding parameter itself and the size (detection window size in time) are often defined manually. Additionally, many require a `rest` signal be given, *i.e.*, a part of the signal where only resting, or baseline noise, is occurring, no activity. These serve as reference signals from which one can determine what constitutes an active state, or even adequate thresholding ($2 \times$ STD, for instance).

Additionally, one other approach was developed, based upon the Teager-Kaiser Energy Operator, whereby a filtered signal goes through TKEO pre-processing: TKEO is a mathematical operator that estimates the energy of a signal in a way that tracks rapid changes better than traditional energy measures (like squaring the amplitude). This operator is defined as:

$$\psi[x(n)] = x(n)^2 - x(n-1) \cdot x(n+1) \tag{4.15}$$

where, $x(n)$ is the signal at time n; $\psi[x(n)]$ is the energy estimate at time n; all implemented through *numpy* (using *np.square* and *np.roll*, specifically). From this process results a new 'signal', as displayed in Figure 4.11; in it one can see its starker prevalence, and reduced baseline noise, when compared to the original EMG signal.

With the TKEO, a mean of its baseline noise is acquired, b_m, along with its standard deviation, b_s (as opposed to the original signal's baseline noise) – these are then used to define the onset/offset threshold: $\text{ThresholdTKEO} = b_m + \lambda b_s$, with λ an adjustable coefficient.

Figure 4.12 shows how this thresholding method, 4.12a, may compare to a standard approach (using original signal and its baseline noise for computing threshold), seen in 4.12b, and to an additional approach of defining the threshold as a percentage of maximum TKEO, 4.12c.

Validation

For validating onset detection, the adoption of the DMD dataset, specifically the healthy patients, focusing on a single channel, allowed for periodic muscle activations (upon each gesture) to be used.

With that, knowing the expected instants of onset and offset – in this case, after 3-second rest, 3-second instant (6144th sample, given the 2048-Hz sampling rate) and 6-second instant (12288th sample), respectively, for each rest-gesture-rest repetition –, the algorithm detection's deviation

Table 4.2: Implemented BioSPPy's EMG onset detection methods.

Method	**Description**	**Parameters**
Bonato[194]	Uses statistical analysis of muscle activation intervals by comparing the EMG signal variance during rest and active periods.	`signal`, `rest`, `sampling_rate`, `threshold`, `active_state_duration`, `samples_above_fail`, `fail_size`
Lidierth[195]	Employs a threshold-based method where the signal is segmented and analyzed for deviations from baseline rest levels to detect muscle activity. Temporary dips below the threshold are handled using a fail-size parameter.	`signal`, `rest`, `sampling_rate`, `size`, `threshold`, `active_state_duration`, `fail_size`
Abbink[196]	Uses a combination of thresholding and transition detection to identify the onset and termination of muscle activity, focusing on changes in the signal's amplitude.	`signal`, `rest`, `sampling_rate`, `size`, `alarm_size`, `threshold`, `transition_threshold`
Londral[197]	Analyzes windows of the signal using a specialized test function that calculates the variance of the signal over a given window size. It compares this variance to a threshold computed from the rest signal and uses the duration of detected active states to distinguish between true activations and random noise. This method also ensures that state changes meet a minimum duration requirement to avoid false activations.	`signal`, `rest`, `sampling_rate`, `size`, `threshold`, `active_state_duration`
Silva[198]	Starts by rectifying and smoothing the signal with a moving average to create a test function. It uses a second, adaptive moving average to adjust its detection threshold dynamically. Muscle activations are identified based on whether the smoothed signal exceeds both a fixed and adaptive threshold, ensuring that activations are robustly detected through a double criteria system.	`signal`, `sampling_rate`, `size`, `threshold`, `threshold_size`
Solnik[199]	Computes the Teager-Kaiser energy operator on the EMG signal after removing the baseline, rectifies the result, and then compares it to a threshold calculated based on the rest signal. This method tracks state durations, detecting active and inactive phases. It requires both an alarm condition and a minimum duration of activation or inactivation to confirm the presence of muscle activity.	`signal`, `rest`, `sampling_rate`, `threshold`, `active_state_duration`
HodgesBui[200]	Subtracts the baseline offset and applies full-wave rectification followed by a moving average to detect EMG pulse onsets, using statistical properties of the rest period for thresholding.	`signal`, `rest`, `sampling_rate`, `size`, `threshold`

from them was computed and averaged across 10 repetitions, 7 gestures, and 8 individuals, totaling 560 observations (activations) per method. Thus,

$$\text{Error}(\%) = \frac{|t_{\text{expected}} - t_{\text{obtained}}|}{\text{Active state duration}} \times 100(\%) \tag{4.16}$$

where activation time corresponds to how long the muscle is contracted and activated – in this case, 3 seconds. Alongside the average of the 560 observations' errors, one obtains the standard

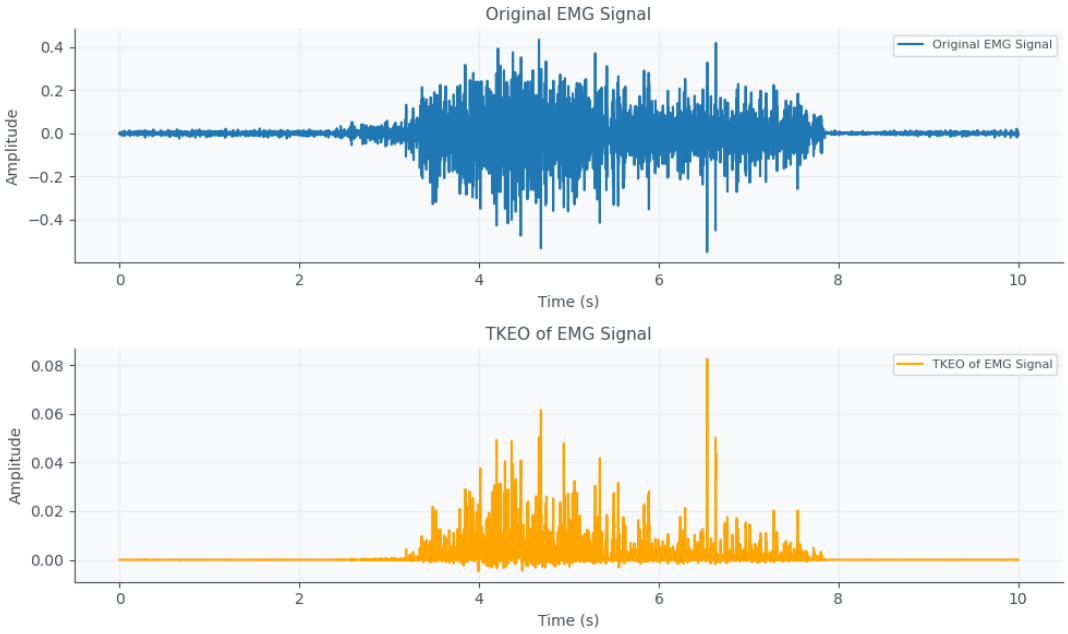


Figure 4.11: Comparison of original EMG signal and its TKEO result.

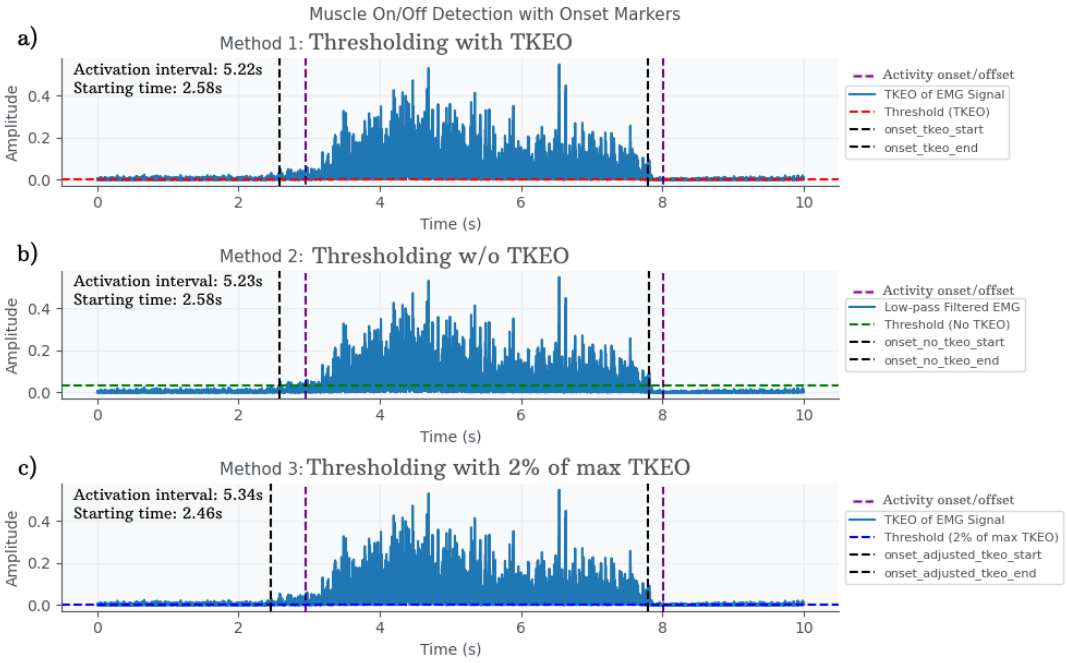


Figure 4.12: Thresholding methods with and without TKEO or max TKEO.

deviation.

To note that thresholds and other parameters of the detection functions were set manually for a singular activation (for example, the 'rest' signal that some algorithms require – representing the signals' baseline – was defined as the resting period preceding the gesture), per each individual, and were thereafter kept the same for the validation process. This way, the error is indicative of

the algorithms' ability for detection assuming and knowing that it works on at least one activation.

4.5 Spectral Analysis

When analyzing the spectral contents of sEMG signals – as first mentioned in Section 2.3 –, both the frequency domain and the time-frequency domain can be studied, useful in evaluating aspects such as fatigue. In this section of the work, the tools used for such analyses are described along with their importance.

4.5.1 Frequency Domain

For frequency domain features – specifically across an entire time window (without segmenting it and displaying metrics through time) –, standard functions were utilized, resorting to *numpy* and *scipy*, as well as *matplotlib* for visual representation. These functions, summarized in Table 4.3, are streamlined such that one needs only provide as arguments the signal (or signals, in case of computing cross Power Spectral density (CPSD), or coherence between signals), sampling frequency, and nperseg (length of each segment). To note that Welch's method was utilized for obtaining Power Spectral Density.

Table 4.3: Summary of frequency domain analysis functions.

Function	Purpose	Parameters	Returns	Packages Used
`FFT1D`	Computes the 1D FFT of a signal	`emg_sig`, `fs`, `plot`	`xf`, `yf`	`scipy.fft.fft`, `matplotlib.pyplot`
`FFT2D`	Computes the 2D FFT of a signal	`emg_sigs`, `fs`, `plot`	`yf`	`np.fft.fft2`, `matplotlib.pyplot`
`PSDwelch`	Computes the PSD of signal using Welch's method	`emg_sig`, `fs`, `nperseg`, `plot`	`frequencies`, `power_spectrum`	`scipy.signal.welch`, `matplotlib.pyplot`
`CrossPSDwelch`	Computes the CPSD between two signals using Welch's method	`emg_sig1`, `emg_sig2`, `fs`, `nperseg`, `plot`	`f`, `Pxy`	`scipy.signal.csd`, `matplotlib.pyplot`
`CoherenceWelch`	Computes the coherence between two signals using Welch's method	`emg_sig1`, `emg_sig2`, `fs`, `nperseg`, `plot`	`f`, `Cxy`	`scipy.signal.coherence`, `matplotlib.pyplot`

4.5.2 Time-Frequency Domain

For time-frequency domain, summary in Table 4.4, the signal is divided into epochs (whose time-lengths can be manually defined) – from whose transforms mean frequency, median frequencies, and total power can be computed – or given fully to a functions that allows for manual definition of nperseg (length of each segment) – as in the case of obtaining the STFT of a signal (using *scipy.signal.stft*).

A `WaveletTransform` function is defined such that one can choose which type of wavelet transform to obtain ('discrete' or 'continuous') and the respective function ('cmor', 'sym', etc.) – while both types of WT utilize the PyWavelet package *pywt*[201], in the case of DWT, the process taken is such that it first performs a wavelet decomposition of the input data, breaking it down into approximation and detail coefficients at different levels (`coeffs=pywt.wavedec(...)`), followed by the `pywt.upcoeff` function, which is then used to reconstruct the approximation

Table 4.4: Summary of time-frequency domain analysis functions.

Function	Purpose	Parameters	Returns	Packages Used
`mdfreq`	Compute mean and median frequency and total power of given spectrum, through time	1-D signal, spectrummatrix, `fs`: sampling frequency [Hz], epoch_size	1D arrays of mean and median frequency, Total power	scipy, numpy
`compute_stft`	Generate spectrogram using STFT	`data`: 1-D signal, `epoch_time`: duration, `fs`: sampling frequency [Hz], `plot`: boolean	Magnitude of the STFT, Frequencies, Times	scipy.signal.stft
`compute_hht`	Compute magnitude of Hilbert Transform of each IMF	signal, max_imfs	2D array, IMF per row	PyEMD.EMD, scipy.signal.hilbert
`WaveletTransform`	Perform (discrete or direct) wavelet transform and plot spectrogram	`data`: 1-D signal, `fs`: sampling frequency [Hz], `wavelet`: type, `level`: decomposition, `mode`: continuous/discrete, `plot`: boolean	Spectrogram matrix, Frequencies, Times	`pywt`, `matplotlib`
`SSmodule`	Perform synchrosqueezing transform	`x`: 1-D signal, `fs`: sampling frequency [Hz]	Synchrosqueezed transform, Frequencies, Times	`ssqueezepy.ssq_cwt`, `ssqueezepy.ssq_stft`

part of the signal from these coefficients. This function takes the approximation coefficients (`coeffs[0]`), the wavelet type, the level of decomposition, and the length of the original data, producing a time-domain signal called approx, which is the approximation of the original data at the given level. Finally, the `plt.specgram` function is used to create a spectrogram of this reconstructed approximation signal. On the other hand, CWT is done by applying the `pywt.cwt` function on the signal (given widths, wavelet function, sampling period), leading to a CWT matrix (2D array where each row represents the wavelet coefficients at a different scale (or frequency)) whose absolute values give the magnitude of these coefficients, done to visualize the intensity of the wavelet transform.

A synchrosqueezing approach to obtaining CWT and STFT is also provided – after the initial transform, synchrosqueezing is applied to refine the time-frequency representation, which essentially "squeezes" the time-frequency coefficients, making the representation sharper and more concentrated around the true frequency components –, through the `SSmodule` function which uses functions from the SSqueezepy library[202] (*ssq_cwt* and *ssq_stft*, specifically) to directly from the signal attain CWT and STFT spectra.

While most functions present are using common packages (*scipy.signal*, *pywt*), which streamline the computation of STFT, CWT, DWT, for some of the less common metrics, such as HHT's IMFs, the process is more involved: the function `compute_hht`, for instance, returns the Hilbert-Huang Transform (HHT) of a given signal, after Empirical Mode Decomposition (EMD) – the function starts by applying EMD to the input signal, `emd = PyEMD.EMD()`, imfs = emd(signal), decomposing it into a set of Intrinsic Mode Functions (IMFs) and a residual –, from the resulting number of IMFs, this number is limited to a 'max_imfs' parameter. If there are more IMFs than 'max_imfs', it truncates the excess IMFs. If there are fewer IMFs than 'max_imfs', it pads the result with zeros. Finally, after obtaining the IMFs, the function applies the Hilbert transform to each IMF (`hilbert_spectrum = np.abs(hilbert(imfs, axis=1))`) to obtain the analytic signal. It computes the magnitude of the Hilbert transform of each IMF, as a 2D array. The first dimension corresponds to the number of IMFs (adjusted to 'max_imfs'), and the second dimension corresponds to the length of the time array.

To obtain the Hilbert spectrum, one can also calculate the instantaneous frequency and then combine this with the amplitude to get a time-frequency-energy representation (not explored herein).

Validation

Similar to amplitude analysis, thanks to the commonly adopted and widely accepted packages being used herein (thus, theoretical validations having been already undertaken with trusted results), validation here is done by comparing results of two individuals, under two different tasks; for this work, only some types of transforms are explored: for wavelet transforms, for instance, CWT's cmor function and DWT's sym function are focused on; for HHT, only the IMF functions limited to 10 were assessed as their amplitude changes through time (disregarding the Hilbert spectrum itself).

4.6 Conduction Velocity Estimation

For estimating CV, the cross-correlation method described in Section 2.4 was used, adopted over two approaches: 2-signal correlation-based (here called "XC") and MLE through spike-triggered averaging (here called "STA"), based on the work by Farina et al.[203; 110], and Valli et al.[186], and resorting to openhdemg library's tools[186].

XC Algorithm

For the XC-based approach, the correlation coefficients between two signals are computed (using numpy's *correlate* for a direct-summation approach; and using scipy's *signal.correlate* for convolutions using FFT (better for bigger arrays)), with the index of maximum correlation used to get number of samples that distance signals' similarity – this is taken as the delay between signals, whose channels and physical distance between them are known: thus, CV = distance/delay.

STA Algorithm

For the STA-based approach, an estimation of the delay of a given MU is done by obtaining the STA of the double differential derivations of the signal upon MU pulses (known from MUAP decomposition), a process shown in Figure 4.13.

To do this, the algorithm starts by obtaining the double differential, *dd*, of the filtered, sorted array of signals (process often to reduce noise, extract features related to muscle activity, and emphasize local variations in the signal), through

$$\mathrm{dd} = -\mathrm{EMG}(n-2) + 2 \times \mathrm{EMG}(n-1) - \mathrm{EMG}(n) \tag{4.17}$$

where n is the current row (electrode). This formula is the central difference method for second-order derivatives, applied to the EMG data across rows.

This is followed by the `sta()` function that calculates the STA of the resulting data, dd. It takes as inputs the firing times of MUs, the number of MUs, the EMG sampling frequency, the EMG data, a range of firings to include, and a time window (these last two, user-defined parameters). The function extracts EMG signals in a specified time window around each MU

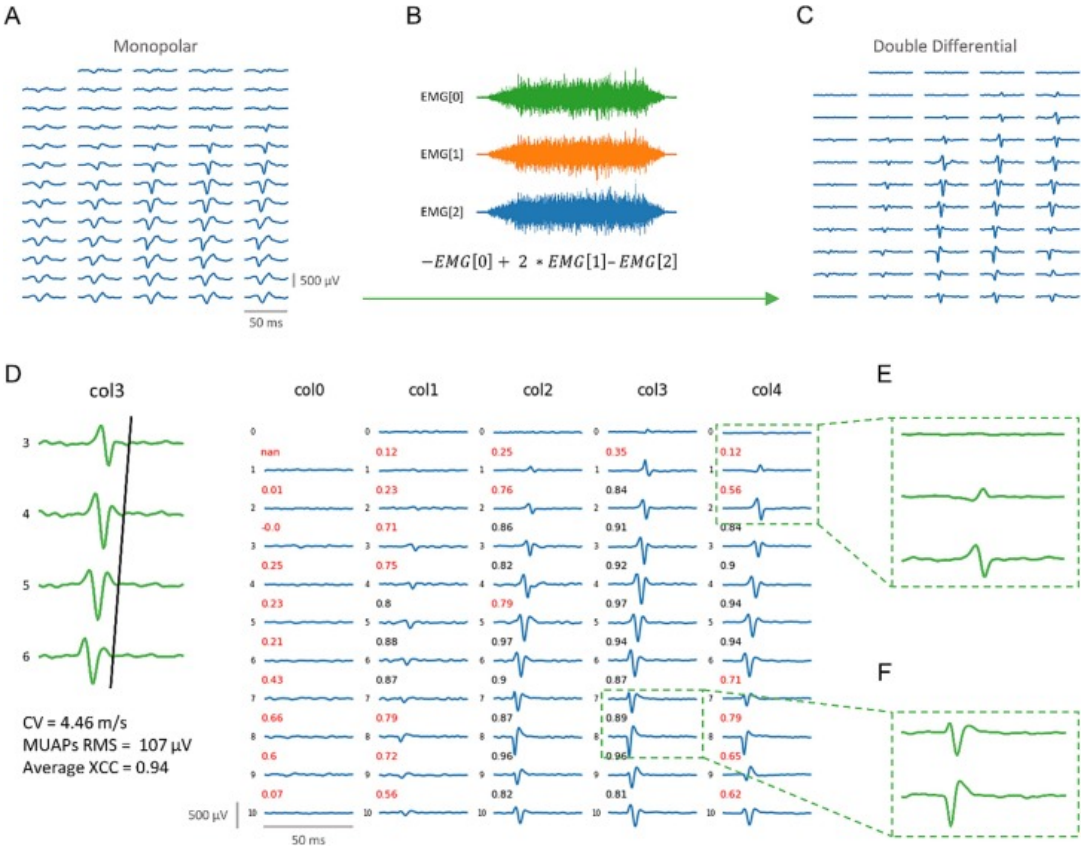


Figure 4.13: STA Algorithm approach by analysis of peripheral motor unit properties which involves estimating motor unit conduction velocity (MUCV) from motor unit action potentials (MUAPs) obtained via double differential spatial filtering of raw EMG signals (A, B, C), showing the speed of MUAP propagation along muscle fibers (D). To accurately estimate MUCV, selection of maximum number of adjacent channels aligned with muscle fibers, ensuring clear action potential propagation and high cross-correlation coefficients (XCC). while avoid channels showing end-of-fiber effects (E) or innervation zone inversions (F), from [186].

spike and averages them to isolate MUAP, which are then stored in a dictionary – this means not only that the algorithm requires that the signal be pre-decomposed (through MU Decomposition), but that the following CV computation can be done per MU.

Once chosen said MU, choosing from which column/row to compute MUCV needs also be done (by choosing two channels within the column/row); with them, and knowing inter-electrode distance and sampling rate, the `find_mle_teta()` function takes these four as inputs, so as to estimate a starting value for delay θ (close enough to the real one to allow for the CV algorithm to converge to an accurate value). The function calculates the optimal time delay (‘teta‘) between two 1D signals (‘sig1‘ and ‘sig2‘) by evaluating cross-correlation. It first defines a range of possible delays based on input parameters (‘ied‘ and ‘fsamp‘). It then computes the cross-correlation for each delay and identifies the maximum value to determine the initial delay. To refine this delay, it uses quadratic interpolation (via *poly.polyfit*), fitting a second-degree polynomial to the peak and its neighboring points to find a more precise delay value. The function returns the best delay,

accounting for possible boundary conditions and providing a more accurate estimate of the time shift between the signals.

Finally, with this θ approximation, the `mle_cv_est` function estimates the CV using maximum likelihood estimation (MLE)[77; 83]. It initializes 'teta' and iteratively refines it using Newton's method by calculating the first and second derivatives of the MSE – via the `derivatives_beamforming` function.

The `derivatives_beamforming` function calculates the first and second derivatives of the mean square error (MSE) for a beamforming technique with respect to a parameter 'teta', using a reference signal from a specified row. It starts by arranging the signals and computing their FFT (using *scipy.fft*) for each channel. The function then computes the first derivative by evaluating the difference between the cross-correlation terms of the FFTs, adjusting for phase shifts induced by 'teta'. It similarly calculates the second derivative by examining the curvature of the MSE with respect to 'teta'. These are computations, summarily shown in equations 4.18 and 4.19 (adopted from Farina et al.[83]'s work), that are efficiently employed using standard *numpy* operations.

$$\begin{aligned}
\frac{d\hat{e}_k^2}{d\theta} &= \frac{4}{N}\sum_{\alpha=1}^{N/2}\left[\mathrm{Re}\{\gamma(\alpha,\theta)\}\frac{d\mathrm{Re}\{\gamma(\alpha,\theta)\}}{d\theta}+\mathrm{Im}\{\gamma(\alpha,\theta)\}\frac{d\mathrm{Im}\{\gamma(\alpha,\theta)\}}{d\theta}\right]\\
\frac{d^2\hat{e}_k^2}{d\theta^2} &= \frac{4}{N}\sum_{\alpha=1}^{N/2}\left\{\left[\frac{d\mathrm{Re}\{\gamma(\alpha,\theta)\}}{d\theta}\right]^2+\mathrm{Re}\{\gamma(\alpha,\theta)\}\frac{d^2\mathrm{Re}\{\gamma(\alpha,\theta)\}}{d\theta^2}\right.\\
&\quad\left.+\left[\frac{d\mathrm{Im}\{\gamma(\alpha,\theta)\}}{d\theta}\right]^2+\mathrm{Im}\{\gamma(\alpha,\theta)\}\frac{d^2\mathrm{Im}\{\gamma(\alpha,\theta)\}}{d\theta^2}\right\}
\end{aligned} \tag{4.18}$$

$$\begin{aligned}
\mathrm{Re}\{\gamma(\alpha,\theta)\} &= \frac{1}{K-1}\sum_{m=1,m\neq k}^{K}\left[\mathrm{Re}\{X_k(\alpha)*Q(\alpha)\}-\mathrm{Re}\left\{\left(X_k(\alpha)e^{j2\pi\alpha(m-k)\theta/N}\right)*Q(\alpha)\right\}\right]\\
\frac{d\mathrm{Re}\{\gamma(\alpha,\theta)\}}{d\theta} &= \frac{1}{K-1}\sum_{m=1,m\neq k}^{K}\left[\mathrm{Im}\left\{\left(X_k(\alpha)\frac{2\pi\alpha(m-k)}{N}e^{j2\pi\alpha(m-k)\theta/N}\right)*Q(\alpha)\right\}\right]\\
\frac{d^2\mathrm{Re}\{\gamma(\alpha,\theta)\}}{d\theta^2} &= \frac{1}{K-1}\sum_{m=1,m\neq k}^{K}\left[\mathrm{Re}\left\{X_k(\alpha)\left(\frac{2\pi\alpha(m-k)}{N}\right)^2e^{j2\pi\alpha(m-k)\theta/N}*Q(\alpha)\right\}\right]\\
\mathrm{Im}\{\gamma(\alpha,\theta)\} &= \frac{1}{K-1}\sum_{m=1,m\neq k}^{K}\left[\mathrm{Im}\{X_k(\alpha)*Q(\alpha)\}-\mathrm{Re}\left\{\left(X_k(\alpha)e^{j2\pi\alpha(m-k)\theta/N}\right)*Q(\alpha)\right\}\right]\\
\frac{d\mathrm{Im}\{\gamma(\alpha,\theta)\}}{d\theta} &= \frac{1}{K-1}\sum_{m=1,m\neq k}^{K}\left[\mathrm{Re}\left\{\left(X_k(\alpha)\frac{2\pi\alpha(m-k)}{N}e^{j2\pi\alpha(m-k)\theta/N}\right)*Q(\alpha)\right\}\right]\\
\frac{d^2\mathrm{Im}\{\gamma(\alpha,\theta)\}}{d\theta^2} &= -\frac{1}{K-1}\sum_{m=1,m\neq k}^{K}\left[\mathrm{Im}\left\{X_k(\alpha)\left(\frac{2\pi\alpha(m-k)}{N}\right)^2e^{j2\pi\alpha(m-k)\theta/N}*Q(\alpha)\right\}\right]
\end{aligned} \tag{4.19}$$

where $*$ is the convolutive product, α is the discrete frequency, $Q(\alpha)$ is the Fourier transform of $q(n)$ (square root of the window applied to the MSE), and $X_i(\alpha)$ is the Fourier transform of the

signals $x_i(n)$.

The resulting derivatives (first and second) are fed into `mle_cv_est` in each iteration, for updating their values as well as 'teta', during Newton's method, as below

$$\theta_{i+1} = \theta_i - \frac{\left.\frac{de^2_{MLE}(\theta)}{d\theta}\right|_{\theta_i}}{\left.\frac{d^2e^2_{MLE}(\theta)}{d\theta^2}\right|_{\theta_i}} \quad i = 0, 1, \ldots \tag{4.20}$$

The loop continues until the change in 'teta' is minimal or a maximum number of iterations is reached. The estimated CV is then computed based on the final 'teta' and the interelectrode distance ('ied').

As mentioned, and unlike with XC, this STA method computes CV for individual motor units from a decomposed signal (MUCV). XC does not require decomposition, but, as a consequence, the result from using real EMG resulting from superpositions ought to be considered as an average of muscle fiber conduction velocity (MFCV).

Both approaches are applied column-wise (linear, 1D arrays, though more columns allow for additional computations and averaging within a HDsEMG grid).

Validation

To validate the ability of the cross-correlation approach depending on signal characteristics, simulated signals were utilized (with known ground-truths about the CV) – with different SNR, IZ, dampening, sampling rate, IED, window size and signal complexity (rate of firings, number of motor units, differing CVs). These aspects are further explained and shown in Subsection 4.1.1.

To assess accuracy, percent deviation was computed in relation to ground-truth CVs,

$$\text{Error}(\%) = \frac{|\text{CV}_{expected} - \text{CV}_{obtained}|}{\text{CV}_{expected}} \times 100(\%) \tag{4.21}$$

STA method's accuracy is dependent on the accuracy of the method used for MU decomposition. Thus, for validation of the CV estimation itself, the MU firings are known *apriori*, from the simulated signals' settings.

Additionally, from the DMD dataset, comparisons of both algorithms' results can be done, along with the assessment of their clinical feasibility.

4.7 Innervation Zone Estimation

To determine the zone in which an action potential is fired – the innervation zone (IZ) –, this work resorted to a method based on Principal Component Analysis (PCA), specifically, a method adapted from Huang et al.'s [204] study (in Figure 4.14, the process of obtaining the two components from a signal is shown). According to their work, PCA identifies key components that capture significant variance across multiple EMG channels, and using the 2nd principal component for IZ estimation was validated by a time-misaligned model showing its coefficients are linearly

related to channel time delays; more specifically, the local extrema displays a correlation to the IZ index (on simpler, single-IZ signals, they coincide). Additionally, Rodriguez-Falces [205] found that the IZ location may also correspond to the electrode which recorded the highest slope of the rising phase of the maximum MUAP. These two factors indicate that both extremum and sharpness (steepness) are of relevance.

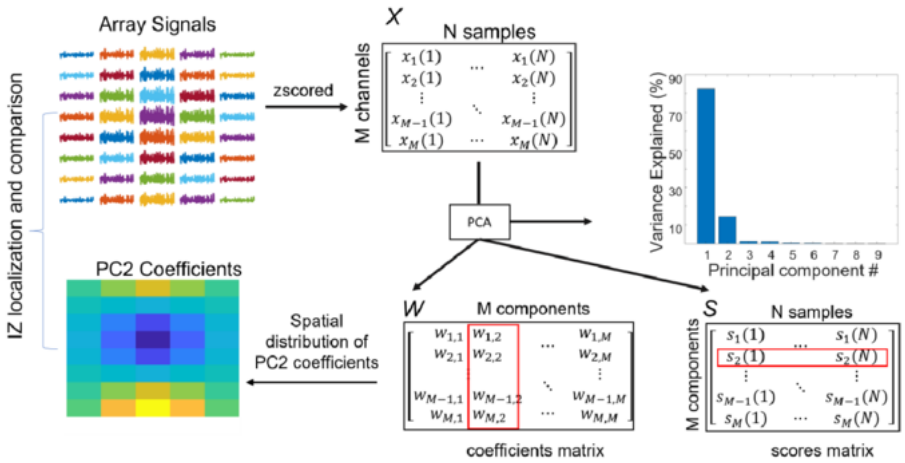


Figure 4.14: Diagram of the application of PCA on a 2D EMG signal array, and resulting components, displayed matricially, as well as their variance and spatial distribution, from [204].

With that, the developed algorithm follows the process of, initially, taking multiple EMG signals (belonging to the same column of the electrode grid), which undergo standardization. This standardization process involves transforming the data to zero mean and unit variance. This transformation is essential to ensure that the subsequent PCA treats all features equally, without being biased by differing scales or magnitudes among the channels. The use of the StandardScaler from scikit-learn achieves this normalization, providing a standardized signal ready for PCA.

Following standardization, PCA is applied to the standardized EMG signal to reduce its dimensionality and identify the principal components (through the *PCA* model of the *sklearn.decomposition* module, with two or more components, onto which the signal array is fitted, `standardscaler.fit(signal)`, and transformed, `standardscaler.transform(signal)`). PCA is a statistical technique that transforms the original variables into a new set of variables, called principal components, which are orthogonal and ordered by the amount of variance they capture from the data. In this case, the algorithm computes the first two principal components. The first principal component captures the maximum variance in the data, while the second principal component captures the next highest variance, orthogonal to the first. This transformation is pivotal in highlighting the underlying structure of the data, enabling a more straightforward interpretation of the signal.

From the PCA results, the coefficients (loadings) of the second principal component are extracted, as a 1D array. These coefficients indicate the contribution of each original variable (electrode signal) to the second principal component. The focus on the second principal component is

due to its potential to reveal significant patterns related to the innervation zone, which may not be as apparent in the first principal component.

To identify the innervation zone, the algorithm analyzes these coefficients to find the sharpest extremum, which can be either a local minimum or maximum. The detection of local minima and maxima within the coefficients is performed through a comparison with neighboring array values. For each extremum, the sharpness is calculated as the absolute difference between the extremum value and the average of its neighboring values. The extremum with the highest sharpness is identified as the sharpest extremum, representing a significant feature in the signal that likely corresponds to the innervation zone. This sharpness criterion ensures that the identified extremum is not only a local extreme point but also exhibits a pronounced change relative to its surroundings. All these operations, done with *numpy* or lists, as each local extremum is iterated through a *for loop* and has their sharpness calculated, `abs(value - (arr[index-1] + arr[index+1]) / 2)`.

Finally, the algorithm returns the index and value of the sharpest extremum point in the second principal component coefficients. This point corresponds to the estimated location of the innervation zone in the EMG signal column.

Validation

Similar to the method used for velocity conduction described in Subsection 4.6, in order to validate the algorithm, simulated signals (expanded upon in Subsection 4.1.1) were used to determine the effects of SNR, dampening, sampling rate, IED, window size, and signal complexity, with a focus on accuracy ACC depending on different IZ-originated channels and overall number of channels. To note that this algorithm is made for single-column grids, so only one column was considered.

To assess accuracy, the closeness of the estimated IZ in relation to the actual IZ known from the simulated signal's ground truth was investigated, such that ACC $= 1 -$ Error, where

$$\text{Error}(\%) = \frac{|IZ_{\text{expected}} - IZ_{\text{obtained}}|}{\text{Total number of channels}} \times 100(\%) \tag{4.22}$$

4.8 Motor Unit Decomposition

For decomposing HDsEMG signals into their motor units, a series of steps to extract individual motor unit activities from the raw signal is adopted.

1. Data Preprocessing: Initially, the EMG signal is preprocessed. This involves flattening the input if it has multiple dimensions (`if len(x[0][0].shape)==2: x = flatten_signal(x)`) and discarding specified channels (`x = np.delete(x, discard, axis=0)`), which refines the data to include only relevant channels. Following this, the signal is centered and extended (`x_ext = extend_all_channels(x, R)`). Centering adjusts the data to have zero mean, and extending increases the data matrix dimensions to aid in the separation process.

2. Whitening: The extended data matrix is then whitened to standardize the variance across all dimensions (`z = whiten(x_ext)`). Whitening transforms the data such that its covariance matrix is the identity matrix, facilitating more effective separation of sources. For that, covariance matrix is calculated (`np.cov`), followed by eigenvalues and eigenvectors of resulting matrix (`scipy.linalg.eig`), regularization factor, replacing eigenvalues smaller than it with the factor (`np.where`), obtaining the diagonal matrix inverse square root of eigenvalues, obtaining its whitened array using zero component analysis (`np.dot`).

3. Separation: The core of the decomposition involves iterating through the source separation and refinement processes. The function initializes the separation matrix 'B' and identifies high-activity columns in the whitened data (`z_peak_indices, z_peak_heights = initial_w_matrix(z)`). This matrix guides the initialization of separation vectors. In each iteration (up to 'M' times), the algorithm selects the column of 'z' with the highest activity and uses it to initialize the separation vector (`w_init = z_peaks[:, z_highest_peak]`). The separation process, handled by the 'separation' function, uses fixed-point algorithms to refine this vector based on contrast and orthogonalization functions. The separation matrix 'B' is updated with the resulting separation vector.

4. Refinement: Following separation, the 'refinement' function optimizes the separation vector. This step includes applying peak-finding algorithms to detect significant features in the estimated source (`s_i2 = np.square(s_i)`), and then uses K-Means clustering to differentiate between prominent and less significant peaks. The effectiveness of the refined source is evaluated using metrics such as Silhouette Score (SIL) and Pulse-to-Noise Ratio (PNR). The separation vector is accepted if its SIL or PNR score exceeds a predefined threshold ('thresh'). Refinement iterates until the coefficient of variation (CoV) of inter-spike intervals stabilizes or the maximum number of iterations is reached.

5. Results Compilation: At the end of the iterations, only those columns of 'B' that correspond to accepted separation vectors are retained (`decomp_results["B"] = B[:, B.any(0)]`). Additionally, the firing indices of the motor units ('MUPulses'), and their corresponding SIL and PNR scores are collected.

This was made possible by resorting to the *numpy*, *emgdecomp*[206; 207], and *EMGdecomPy*[208] libraries, the last two based heavily on Negro et al.'s[94] Convolutive Sphering BSS algorithm.

Obtaining the MUAP firing rates and signal intensities was done by centering the signal (subtracting mean of each row – resulting in the data being centered around $x = 0$), squaring and averaging the results and, once known the spike train indices, for a given MU the instantaneous firing rate was computed as the inverse of difference between MU firing instants; the signal intensity (or strength) as the value of the centered+squared+mean array at each MU firing instant (index). For MUAP shapes, an average across all firings within a time-window surrounding the discharge times of a given MU was done.

The three previous metrics are operations efficiently obtained with *numpy*, and have been summarized in Figure 4.2 from Section 4.1.1.

Validation

For evaluating spike trains, each obtained MU's firing instants were compared to the ground-truth's MU's firing instants, using rate of agreement to tolerances of 5ms, 10ms, 50ms, 100ms and 1000ms – often surrounding a tolerance of 1/10 the sampling rate (for 2048 Hz, for example, this would be of a time-distance of 205 samples, roughly 100ms) – on a one-to-one basis (for each obtained firing there can be *one and only one* corresponding ground-truth agreement), and vice-versa.

The rate of agreement (ROA), as in Equation 4.23, given a tolerance, refers to how often two sets of measurements or data points fall within a specified range of each other.

$$\text{Rate of Agreement} = \frac{\text{Number of Agreements within Tolerance}}{\text{Total Number of Comparisons}} \tag{4.23}$$

Mathematically, with two sets of data points $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$, and a tolerance T, the rate of agreement R can be expressed as:

$$R = \frac{\sum_{i=1}^{n} I(|a_i - b_i| \leq T)}{n} \tag{4.24}$$

where: $|a_i - b_i|$ is the absolute difference between the i-th pair of data points; $I(\cdot)$ is an indicator function that is 1 if the condition inside is true and 0 if it is false; n is the total number of pairs.

Visual comparisons were also done, for spike trains, instant firing rates – along with numeric comparisons of mean signal intensities and mean rate of firings found between ground-truth and obtained –, and MUAP shapes, focusing on amplitude, signal complexity and envelope similarities, proximity to expected channel, and initial and final firing moments.

Validation was done over several simulated signals (mostly 16 by 4, at 1000 Hz, to a simulation time of 30 seconds) with differing number of MUs, CVs, and firing rates, as also explained in Subsection 4.1.1. For instance, signals from 1 MU, 5 MUs, 10 MUs, 20 MUs – with roughly similar complexity – were evaluated, and so were signals from 5 MUs with lower complexity (lower CV, FR, noise) and signals from 5 MUs with higher complexity, for better understanding the effects that differing CV and firing rates may have upon the algorithm's ability for MU and firing identification.

Additionally, the algorithm was used on the DMD dataset, possibly allowing for clinical assessments.

To note that, for consistency, parameters were kept the same throughout the study, from SIL threshold for accepting a separation vector (equaled to 0.92) and tolerance for element-wise comparison in separation (equaled to 0.001), to maximum number of iterations on refinement, fixed point algorithm and decomposition.

ML Applications

4.9 Gait Monitoring

While timing analysis often involves activation onset detection, event detection may extend itself into determining when different tasks are being performed or even into monitoring one's gait. In the case of Parkinson's disease patients, monitoring their middle to late stages may be necessary in order to identify moments of gait freezing: this part of the work thus sought to demonstrate the software's ability to discern between two different events, *mobile* and *freezing of gait* (FOG) – of clinical relevance, as explained in Section 2.1.5.1.

In particular, the software makes use of traditional machine learning techniques to facilitate in the automated learning of each event's characteristics (through steps like feature engineering and feature selection); to do so, a pipeline – seen in Figure 4.15 – is followed such that, for other events of clinical relevance (given the pertinent dataset), it too can be used to train a model dedicated to event detection. This section of the methodology explores each of the pipeline's steps.

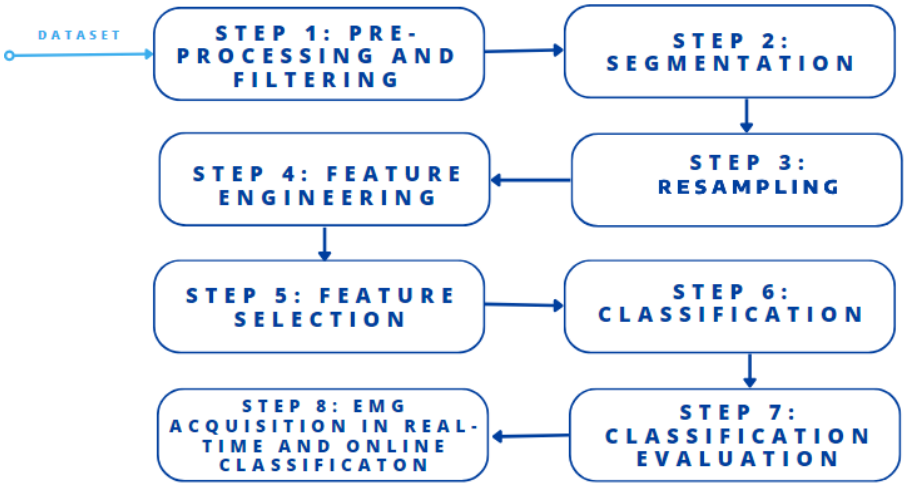


Figure 4.15: Diagram of the pipeline adopted for developing the gait monitoring's FOG classification model.

To note that, while step 8 of Figure 4.15 is not herein explored, the steps that precede it are implemented such that they account for its eventual inclusion, as explained in some of the following subsections.

4.9.1 Dataset: Freezing of Gait in Parkinson's Disease

To develop a model that can differentiate and classify different tasks and states, not every electrode placement is equally effective; when it comes to gait monitoring, one ought to focus on the leg muscles, for example. Additionally, as per machine learning techniques, particularly supervised learning — which is the method by which a model is able to forecast or classify inputs based on previous and present labeled data used to train it[209], and, thus, is the type of classifier adopted in

this part of the work —, resorting to a vast set of different subjects' gestures is essential, because of factors such as:

- Variability in Human Anatomy: People have different anatomical structures, muscle compositions, and limb proportions. These variations can affect the way EMG signals are generated and transmitted. By incorporating data from a diverse set of subjects, one can capture a wider range of anatomical differences. This helps the model generalize better to unseen individuals, making it more robust and adaptable.

- Interpersonal Variability: Different individuals may perform the same movement or gesture in slightly different ways due to factors like personal habits, physical conditioning, or motor control strategies. Incorporating a diverse set of subjects helps the model learn the commonalities and variations in how different people execute the same movement. This allows the model to recognize the underlying patterns and disregard individual idiosyncrasies.

- Environmental Factors: EMG signals can be influenced by environmental factors such as electrode placement, ambient noise, and muscle fatigue. By training the model on data collected from various subjects in different settings, it gets exposed to a wider range of real-world conditions. This helps the model learn to differentiate between genuine movement patterns and noise or artifacts introduced by the environment.

- Generalization and Adaptability: Machine learning models are designed to generalize from the training data to unseen examples. By training on a diverse set of subjects, the model learns to capture the essential features that are common across individuals. This allows it to generalize well to new individuals who were not part of the training set. Additionally, having a wide range of subjects helps the model adapt to new individuals by leveraging the knowledge acquired from similar subjects during training.

- Avoiding Bias and Overfitting: Incorporating data from different subjects helps prevent the model from becoming biased towards specific individuals or populations. If the model is trained on a limited set of subjects, it may become overly specialized to their unique characteristics, leading to poor generalization. By including a diverse range of subjects, one can reduce the risk of overfitting and promote a more balanced and representative model.

A vast set will, as such, allow the model to have a better ability to generalize, adapt, and perform well across different individuals and real-world conditions, *i.e.*, for there to be higher chances that anyone else outside of those subjects will be provided more accurate classifications of their gestures without the need for much tinkering.

With that in consideration, the dataset used was that of W. Zhang, et al.[192]'s, *i.e.*, the FOG Dataset, further described in Subsection 4.1.2.1.

4.9.2 Pre-Processing and Filtering

As established, EMG signals are often contaminated with various noise sources that can hinder accurate interpretation and analysis. To address this challenge, signal filtering techniques are commonly employed to enhance the quality of EMG signals. With that in mind, detrending and the application of a Butterworth bandpass filter and a notch filter to the EMG signals, specifically with cutoff frequencies set at 15-500 Hz and a notch frequency at 50 Hz, respectively, was done.

The Butterworth filter is a widely used digital filter that provides a maximally flat response in the passband and a smooth roll-off in the stopband. By applying a bandpass filter with appropriate cutoff frequencies, unwanted low-frequency drifts and high-frequency noise can be effectively attenuated, while preserving the frequency components of interest within the desired range[210]. For sEMG signals, a typical choice for the lower cutoff frequency is 15 Hz, which helps eliminate baseline wander and electrode motion artifacts[211], while the upper cutoff frequency of 499 Hz (just under 500 Hz, given the 1000-Hz sampling rate) allows for the retention of the more relevant muscle activity information[212].

In addition to the sixth-order Butterworth bandpass filter, a second-order infinite impulse response (IIR) notch filter is employed to attenuate specific interfering frequencies, such as the powerline interference at 50 Hz[213]. Powerline interference often contaminates EMG signals due to electromagnetic coupling between power lines and the measurement system. This interference can introduce artifacts that degrade the accuracy of EMG analysis. The notch filter selectively eliminates the powerline interference, leaving the desired EMG signal intact.

With the entire data imported into Python (three arrays [3 channels] per person), each one was put through Python's lfilt function, which is a causal forward-in-time filter — unlike the filtfilt function, which, while it doesn't shift the signals as it filters, it can't be used in "online" real-life applications and only for offline processing because it filters forward and backwards in time.

While the dataset's processing is indeed done offline, one ought to maintain consistency between how the data is filtered offline and online – for the purpose of this work, only offline data is analyzed, but monitoring a patient in real-time would be an expected course of action to eventually take.

Finally, minmax normalization of each signal, and scaling from -1 to 1, was done, to account for possible different gains.

4.9.3 Segmentation

Segmentation refers to the act of dividing each subject's dataset into the parts that present the same statistical characterizations[214], thus, leading to similar classification. This means that each segment would correspond to a different task (one of four) performed by the PD patient and a different gait state (active or frozen). Given these labels (provided within the dataset itself), all data (in groups of three sEMG signals) were divided in windows of 3000 samples, or 3 seconds (though other window lengths were experimented with, as explored in Chapter 5), each separated from the

next by step lengths of 300 samples, or 0.3 seconds. In the end, a total of 12846 observations was gathered, across all 12 patients.

4.9.4 Resampling

While for this dataset such did not prove to be necessary, reducing or increasing the number of samples from the dataset can be an important step, to ensure cohesion between all signals, trying to make them have characteristics that are as similar as possible. If ever this event detection process were to be adapted to signals obtained at a different sampling rate (for example, from equipment used for the real-time classification), the need for resampling should not be ignored. In this case, however, no resampling (up or downsampling) was done.

4.9.5 Feature Extraction

To make predictions or, in this case, classifications, the ML model requires input data that can inform it about which of the signal's characteristics are relevant and that can help it recognize patterns. The selection and transformation of raw data (as the EMG signals from the dataset) into features is given the name "feature engineering"[215].

Instead of applying the entire array of EMG sample values obtained during an interval of time (which can work for more complex deep-learning models), it was chosen to instead focus on the various domains under which EMG signals may prove most relevant: time domain; frequency domain (Fourier transform and Power Spectral Density); fractional fourier transform domain — from which one can extract values (or features) such as mean, root mean squared (RMS), variance, kurtosis, skewness, among others.

Present in Table 4.5 are all the features that were considered for this work.

Table 4.5: Features considered for training the machine learning model, with quantities obtained under the time domain, frequency domain and fractional Fourier transform domain.

Transform Domain	Features Considered
Time domain	Root mean squared (RMS); minimum; maximum; sum of absolute differences; SSI; mean absolute value (MAV); variance; differential absolute standard deviation value (DASDV); zero crossing; skewness; kurtosis; Willison amplitude; integrated EMG; logarithmic detector; standard deviation; mean absolute difference; myopulse **Hjorth**: Mobility; Complexity **Fractal**: Higuchi Fractal Dimension
Frequency domain	**Fourier domain**: Peak; total power; mean; median; max power; skewness; kurtosis **Power spectral density (PSD)**: Mean frequency; median; entropy; first moment (mean); second moment (variance); third moment (skewness)
Fractional Fourier domain (alpha = 1.5)	Mean; Difference between maximum and minimum; RMS; variance; standard deviation

Studying the **time domain** of EMG signals is important as it provides insights into muscle activation patterns, recruitment strategies, fatigue, and motor control. Examining features in the time domain enables the analysis of signal variations over time, capturing temporal characteristics

such as duration, onset, and termination of muscle contractions during different movements[216]. This information can aid in distinguishing between the different tasks and states.

Studying the **frequency domain** of EMG signals, through techniques like Fourier transform and power spectral density analysis (done through Welch's method[217]), is crucial for gaining insights into the frequency content and distribution of muscle activity. By converting EMG signals from the time domain to the frequency domain, several important aspects can be understood in a concise manner. Fourier transform allows for the decomposition of the EMG signal into its constituent frequency components, enabling the identification of dominant frequencies and their respective amplitudes. Power spectral density analysis further quantifies the power distribution across different frequency bands, revealing information about the intensity and distribution of muscle activity. This analysis aids in detecting specific frequency patterns associated with muscle activation, fatigue, and neuromuscular abnormalities. Overall, studying the frequency domain of EMG signals using Fourier transform and power spectral density analysis facilitates a comprehensive understanding of the spectral characteristics of muscle activity and its functional implications[218].

The **fractional Fourier transform domain** analysis is a mathematical technique that extends the traditional Fourier transform, allowing for a more flexible representation of signals between the time and frequency domains[219].

It provides a multidimensional representation of signals, capturing both time and frequency information simultaneously. By studying the fractional Fourier domain of EMG signals, one can gain insights into the complex time-frequency properties of muscle activity, uncovering subtle variations, temporal patterns, and non-stationarities that may not be apparent in other domains. For the purpose of this project, an alpha of 1.5 (as per conventions in S. Maddali[220]'s frft Python code) was chosen because it provides a better balance between time and frequency resolution; adequate, since what is being analyzed are signals with moderate time-varying frequency components (which is the case for the steady-state disposition that describes the involuntary muscle contractions initiated by these moments of gait freezing).

Minimum, maximum, averages and variances are all important features, seeing as they provide essential information about the range, magnitude, central tendency, and variability of muscle activation levels. Even so, more complex quantities ought to be considered, such as:

Root mean squared (RMS); adequate because it provides a representative measure of the overall muscle activity level, capturing both the amplitude and duration of muscle contractions[221].

Kurtosis; a statistical measure that describes the shape of a distribution by measuring the heaviness of its tails and the presence of outliers, making it a good feature for EMG as it can capture abnormal muscle activation patterns, such as sudden bursts or prolonged contractions[222].

Skewness; a statistical measure that quantifies the asymmetry of a distribution, and it can be a useful feature for EMG as it helps detect and differentiate between different muscle activation patterns, such as one-sided muscle dominance or imbalances in muscle recruitment during movement[222].

Zero crossing; a feature that counts the number of times an EMG signal crosses the zero-axis, possibly valuable for EMG analysis as it indicates changes in muscle activation patterns and can help identify the onset and offset of muscle contractions[223].

Willison amplitude; a feature that calculates the number of times the amplitude of an EMG signal crosses a specific threshold within a given time window, and it is valuable for EMG analysis as it provides a measure of rapid changes in muscle activity, helping identify quick contractions or bursts of muscle activation[224]. The threshold in this work being defined as the sum of the average of all acquired samples with the triple of its standard deviation.

Hjorth parameters, specifically Hjorth mobility and complexity; statistical measures used in EMG analysis that provide information about the signal's overall activity level and its irregularity or complexity, making them valuable features as they capture both the dynamic range of muscle contractions (mobility) and the variability and pattern of muscle activation (complexity)[225].

Higuchi Fractal Dimension; an algorithm for measuring the fractal dimension of discrete time series (as is the EMG signal), by quantifying its self-similarity at different scales, making it a valuable feature for EMG as it can characterize the intricate and fine-grained patterns of muscle activation, providing insights into muscle coordination and control[226].

Entropy; in the context of EMG analysis, referring to the amount of randomness or unpredictability in the distribution of muscle activation levels, and it is a valuable feature as it captures the complexity and information content of EMG signals, allowing the assessment of muscle coordination, fatigue, and changes in motor control strategies[227].

To note that, while only 37 features were added, they are multiplied by three ($37 \times 3 = 111$, in total), since these were computed for each channel.

While it is true that using the raw data directly can sometimes be done, especially when resorting to more complex deep learning models, particularly multilayered neural networks[228], such usually requires much more processing power and is only feasible when there's much larger dataset sizes being worked with[229]; this option, though not explored in this work, is adopted in the work of Section 4.10 (for gesture and health condition classification).

4.9.6 Feature Selection

Among the many features considered, some of them may be more impactful than others, and choosing to keep specific ones and excluding others can be beneficial for the overall precision of the ML model. Not only that, but having too few features won't allow for the detection of underlying patterns, leading to underfitting[230]; conversely, having too many features, or irrelevant ones, can cause overfitting[230].

In order to select the ones conducive to better results, feature selection was done by number of channels (where combinations of the three EMG channels were evaluated) and by domain (combinations between the three different domains considered).

Additionally, and once the above steps were done, three statistical methods of feature selection for ML classifications were adopted: Chi-square, ANOVA F-value, and Mutual Information

scoring techniques; these work through univariate selection, in which features are selected based on how strong their relationships with the output variable are[231].

From a total of 111 features (plus the task categorical feature, which is considered to be known), starting from only 4 and, 4 by 4 (chosen arbitrarily, with a minimum of 3 based on the fact that three EMG channels were used, plus the task), ending at 112 (with scikit-learn's SelectKBest), the classifiers selected the features each time, followed by an evaluation of their resulting precision (done with several of the ML models).

4.9.7 Classification Model Evaluation

With all features gathered (in Python, a *Pandas* DataFrame was used for arranging all features within an organized table that could then be used as the "dataset" X), the set was randomly permutated (to avoid biases) and then split (with *train_test_split*) into training (80%) and testing (remaining 20% – chosen based on the empirical assessments that an 80/20 split is best to avoid overfitting[232]) sets, along with the "target" (y) values (entries that correspond to the gestures) – splitting was stratified to ensure equal distribution of labels on train data and test data. Additionally, the splitting was done only for the observations acquired from 10 of the 12 subjects; with that, the resulting distribution of labelled data was as shown in Table 4.6 – though with relatively similar distributions when looking at the case where classifications are done based solely on FOG state (about 60% corresponding to mobile, and 40% to frozen), the same cannot be said about the other cases' balancing: further discussed in Section 5.8 of the 'Results and Discussion' chapter.

Table 4.6: Distribution of the FOG dataset's labels after segmentation; (FOGn, Tm) corresponds to the combination of FOG state n and task m.

FOG State Only	Distribution	Task Only	Distribution	FOG State and Task	Distribution
0 - Mobile	0.594	1	0.535	0 (FOG0, T1)	0.298
				1 (FOG1, T1)	0.237
		2	0.333	2 (FOG0, T2)	0.209
				3 (FOG1, T2)	0.124
1 - Frozen	0.406	4	0.072	4 (FOG0, T3)	0.044
				5 (FOG0, T4)	0.043
		3	0.060	7 (FOG1, T4)	0.029
				6 (FOG1, T3)	0.016

The two remaining subjects (chosen to be subjects 10 and 11), whose data was not used for any training, served as further model accuracy testing, and to validate its inter-subject reliability on unseen data.

Using the "scikit-learn" library in Python, the several ML models were imported — the ones considered in this work are the 200-estimators Random Forest (RFC), linear-kernel Support Vector Machine (SVM), Linear Discriminant Analysis (LDA), Gradient Boosting (GBC), Multi-Layer Perceptron (NNC), K-Nearest Neighbors (KNN) and Multinomial Naïve-Bayes (MNB) classifiers. After which, each model was fitted with the training set, thereby training the model. From that, determining how adequate the model is is done by checking how often it is able to correctly

classify the test set (the "accuracy"). For the heavily unbalanced class groups ("task only" and "FOG State and Task"), weighted averages of the F1-score metric were considered to account for it. The F1-score is a metric used to evaluate the accuracy of a classification model, particularly when the class distribution is imbalanced. It is the harmonic mean of precision and recall, defined as

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4.25}$$

Precision (P) is the ratio of true positive predictions to the total predicted positives ($P = \frac{TP}{TP+FP}$), and recall (R) is the ratio of true positive predictions to the actual positives ($R = \frac{TP}{TP+FN}$). For unbalanced data, the weighted average F1-score is particularly useful because it accounts for the varying sizes of different classes, ensuring that the performance metric is not dominated by the majority class. The weighted average F1-score is calculated as

$$F1_{weighted} = \sum_{i=1}^{n} \frac{N_i}{N} \cdot F1_i \tag{4.26}$$

where N_i is the number of instances in class i, N is the total number of instances, and $F1_i$ is the F1-score for class i. This approach provides a more balanced view of the model's performance across all classes, making it a robust metric for evaluating models on imbalanced datasets[233].

In this work, to evaluate a model and further train it, 5-fold cross validation was also performed on the data (with *cross_val_score*, which performs the aforementioned splitting process on 5 different sections of the entire dataset), which provided a resulting average accuracy and standard deviation (ACC $\pm$ STD); these two were used to compare results with one another.

4.10 Deep Learning-Based Gesture and Health Condition Classification

When seeking to identify different gestures performed by an individual, or even when trying to detect possible muscular conditions that said individual may suffer from, the use of deep learning for automated learning of the characteristics that differentiate each situation can be employed, if given a large and relevant enough dataset, and if following a processing pipeline that competently provides the DL's architecture with the relevant input. To automate the process of classification, regardless of specific gesture or pathology, this part of the work follows a pipeline that both facilitates the creation of a database and investigates which architectures are better suited for classification, as shown in Figure 4.16.

For the DL side, this work used TensorFlow 2.15.0, through the Google Colab platform, leveraging the NVIDIA T4 GPU for running the computational tasks – while adequate, it carries RAM limitations. Because of this, and also to make the pipeline implementable in-software or in-device (like chip-embedded ones[115]), lightweight architectures were explored, both in terms of their complexity and of the size of the training set used as input.

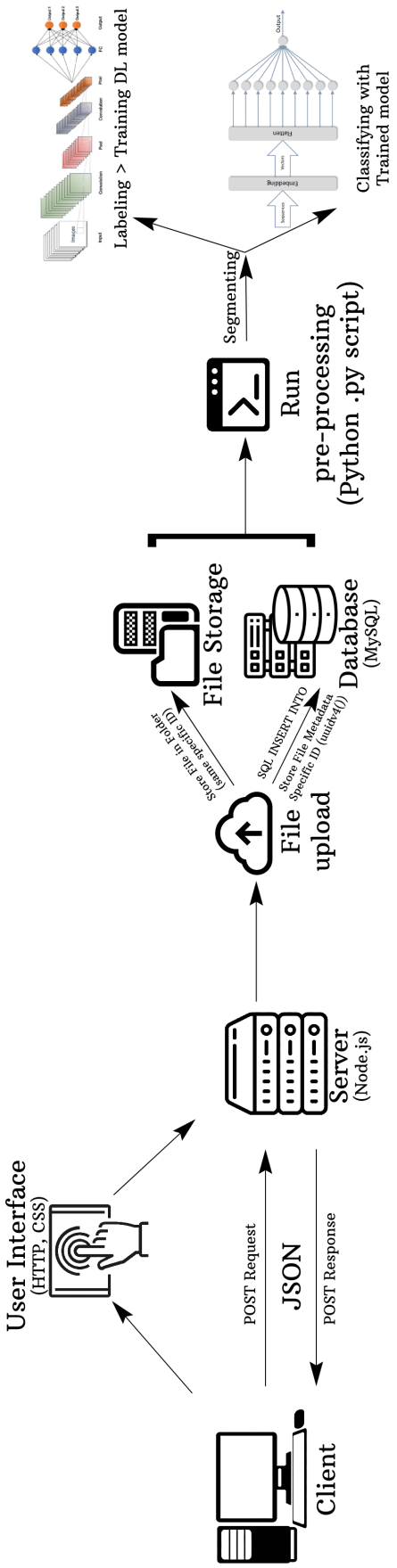


Figure 4.16: Process adopted for creating and managing a HDsEMG signal database, allowing for the user to interact with its user interface and upload their own (.CSV or .MAT) files, whose signals are then pre-processed for DL training or classification.

4.10.1 Database Platform

While distinct datasets are often found in distinct databases, for this work a common database platform was created that enables any user to upload their signals (multichannel or otherwise) in a .CSV or .MAT format, along with its related metadata, as is exemplified in Figure 4.17.

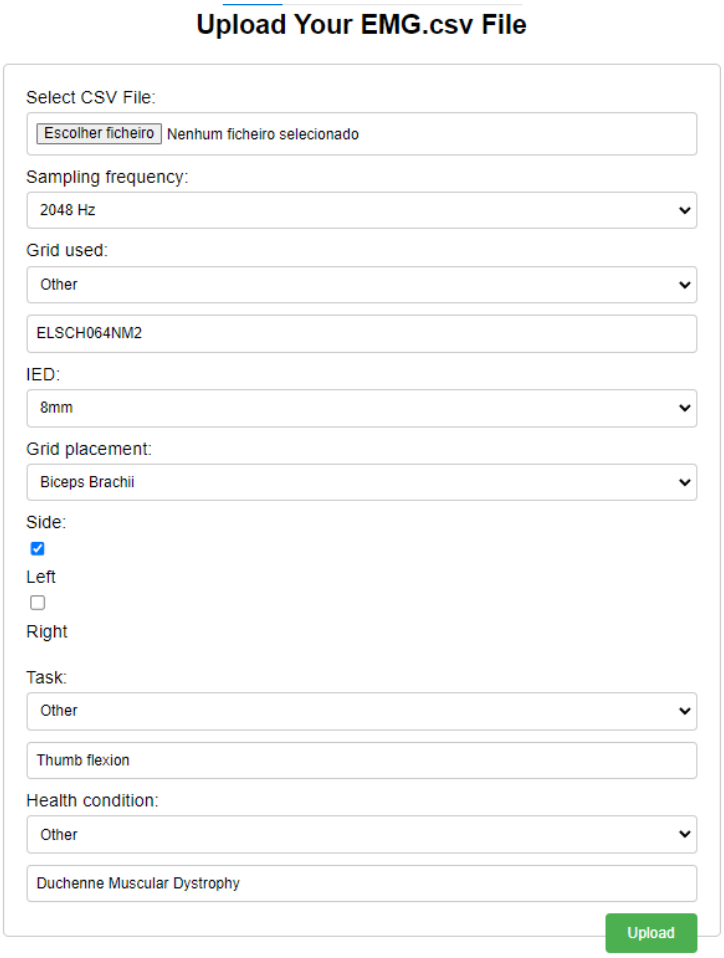


Figure 4.17: Screen-capture of the database platform's user interface developed in HTML, from which to upload one's sEMG signals.

To create the database platform, setting up a MySQL database was done, used as the context for all operations. A table structure within this database was defined, designed to store the metadata – various pieces of information such as file ID, sampling frequency, grid used, IED, grid placement, side of body on which the grid was placed, task performed, and health condition. Each column was carefully defined with appropriate data types and constraints. From the MySQL Workbench IDE, it is possible to ensure the table is correctly set up, by running queries to retrieve all records, allowing for the verification of the table's structure and contents.

Additional information such as task protocol, wherein one may include several tasks and specify time intervals between them, can also be implemented, facilitating in the signals' eventual segmentation and labeling.

With the database configured, a Node.js server was established, using Express.js to create a server capable of handling file uploads, leveraging middleware like ‘express-fileupload‘ to manage the upload process; likewise, comes the use of modules such as ‘path‘ and ‘fs‘ for file handling, and ‘uuid‘ for generating unique file IDs. Additionally, middleware functions were incorporated to validate the uploaded files, ensuring they met specific criteria for existence, file extension, and size.

The server is connected to the MySQL database using the mysql module, configured with appropriate credentials including host, user, password, and database name. Error handling is implemented to manage connection issues effectively. A POST endpoint /upload is set up to handle file uploads. Upon file upload, the server extracts metadata from the accompanying form fields such as grid used, IED, grid placement, side, task, health condition, and sampling frequency. Each upload generates a unique file ID using the uuidv4() function, thus never running the risk of overwriting previous entries

For each uploaded file, the server moves the file to a designated directory and creates a corresponding text file containing the metadata. This metadata along with the file ID is inserted into a MySQL table using a prepared SQL insert statement. Upon successful upload and data insertion, the server responds with a JSON containing the file ID and a success message. From this directory, it then becomes possible to run a Python script that reads the .CSV or .MAT file and pre-processes its data, eventually integrating it into the training or testing sets, as is explained in the following subsections by way of a “proof-of-concept” dataset, or even classified, assuming that its task and/or health condition are unknown and that a model compatible with the signal’s characteristics has already been trained.

To facilitate user interaction, an HTML form – with basic CSS styling to enhance the form’s appearance – was developed allowing users to upload .CSV and .MAT files and provide metadata, as previously mentioned, and as seen in Figure 4.17. The form includes fields for file selection and the metadata attributes. Additional input fields dynamically appear when "other" options are selected in dropdowns, information that is stored as well.

JavaScript handles the form submission process. Upon submission, a JavaScript function gathers form data and files, packages them into a FormData object, and sends them to the server via a POST request. This approach ensures accurate capture and processing of user inputs, enabling seamless interaction with the database platform.

Thus, once implemented, the pipeline supports dataset uploads, assuming proper segmentation (currently manual, but potentially automatable with onset detection algorithms) and accompanying labeling of segments.

4.10.2 Dataset: Healthy and Duchenne Muscular Dystrophy Patients

In order to classify gestures and conditions, the DMD dataset was used, being deemed adequate for the exploration of this part of the work; this dataset is further detailed in Subsection 4.1.2.2.

Additionally, since each individual, while performing 7 gestures, can either suffer from DMD or be healthy (2 conditions), this work will take each pair "(gesture, condition)" as a class: as such, there are 14 classes in total, from this dataset.

As explained in Section 4.9.1, in order to develop a model capable of classifying the gestures made by someone in motion, it is not enough to place electrodes on the body and to acquire the EMG signals produced therefrom; ensuring that the muscle on which the electrodes are placed are relevant for the analyzed task and that the dataset itself is varied enough to allow for generalizability is important. It was recognized by the creators of the dataset that, due to the low number of subjects with DMD, it does not cover the complete spectrum. Even so, this proof of concept was followed through, showing its potential.

4.10.3 Pre-Processing and Filtering

Each signal was firstly filtered through a 6-th order Butterworth bandpass filter, with cutoff frequencies at 20 Hz and 500 Hz; also, a notch filter was applied at 50 Hz – similar to the process found in Section 4.9.2, and for likewise reasons.

This filtering was also applied on the MVC signals, which were then used to normalize the data: 64 absolute maximums of the MVC signals – one for each channel of the HDsEMG grid used – were computed for each individual, and used as the normalizing factor for all other signals of said individual. This step can aid in deep learning, since it enhances comparability, reduces variability, and has been shown to improve model robustness by accelerating the learning process (limiting the range of input values such that the weight updates calculated through backpropagation become more reasonable)[108].

4.10.4 Downsampling

Though the original signals were captured at a 2048-Hz sampling rate, the process of downsampling to 1000 Hz was undergone: this was done, both because, following a consistent pipeline, it was chosen a constant sampling rate with which to always conduct the deep learning process (*i.e.*, if another dataset had been chosen, it too would have been resampled to 1000-Hz, thereby ensuring input sizes remain consistent too), and also due to hardware limitations: higher sampling rates imply bigger batches, occupying more RAM.

This reduction, or downsampling, can be done before or after segmenting, but it should (and was) done always after filtering, so as to avoid losing relevant frequency information and to prevent aliasing effects[234].

4.10.5 Segmentation

As shown in Figure 4.18, each person performed 3-second actions, separated by 3-second resting periods, a total of 10 times. Additionally, because the start times weren't always at the start of the protocol, the first few seconds of rest often went on longer than three seconds. Because of this, before segmentation, the signals were shortened so as to start at the moment of an initial burst,

considered to be an activation resulting from the first time the gesture is being performed. This was done manually (visually), on each individual.

Therefore, and taking into account the 1000 samples per second, one would get the second gesture, for instance, by extracting the samples from the 6000th to the 9000th (of each cycle) (corresponding to the interval starting at 6 seconds and ending at 9 seconds); an analogous process on all the other gestures is done, keeping in mind that the moments of "rest" between each one aren't used.

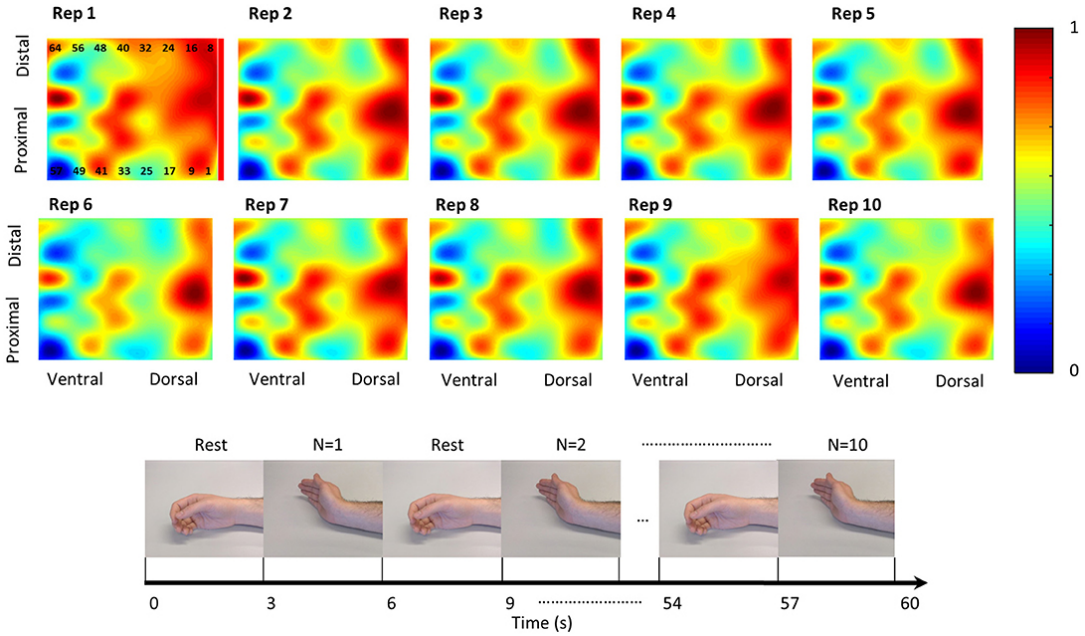


Figure 4.18: Diagram displaying the periods of each gesture from the DMD dataset, in this case, wrist extension; each cycle done once per subject, along with corresponding average normalized maps, from [171].

Once segmented, each 3-second interval (of static gesture) was also divided into windows of 0.5 seconds — ensuring that the classification model is able to determine gestures that are done for only 0.5 seconds with better accuracy, as opposed to having to stay in that position for 2 or 3 seconds —, and with step intervals of 0.2 seconds (200 samples).

Correct segmentation means that, in this case, there will be a total of 1050 segments per individual (150 per gesture), with each segment containing 500 samples (0.5-second duration) per channel (64 channels) and corresponding to a specific gesture. Overall, it could be matricially represented by an array of shape (1050,64,500), per person – for the sake of the "proof of concept", the number of observations was deemed adequate.

To note that filtering should happen before segmenting, so that it acts on a bigger window and upon the entire enveloped signal, thereby reducing artifacts and leakage upon convolutional signal processing methods[235].

Any big negative impact that filtering may lead to will mostly only happen at the ends of the signals, from tapering[236], which isn't very impactful, since the samples around those regions

aren't used.

4.10.6 Label Distribution Adjustment

Since the purpose of this work focused mostly on being able to not only identify the gesture, but also the condition of whoever is performing it (between healthy and DMD), it became important to ensure that the distributions of labels was not markedly biased or unbalanced; this would mean only 3 healthy subjects being used for training, since there are only 3 DMD subjects. However, since being healthy is a more common occurrence, 6 healthy subjects were used to represent that fact (discarding the remaining 3). This means that 33.3% of the labels corresponded to gestures made by DMD patients (4.8%, each gesture), and 66.6% by healthy patients (9.5%, each gesture).

4.10.7 Spectral-Based Processing

Though neural networks are able to "learn" from raw sEMG given as input (be they given as images, or as time-distributed vectors), additional pre-processing can be applied to discern more intricate characteristics of a signal and aid in the learning process, as discussed in Chapter 3. To explore this idea, this work obtained the time-frequency images of the Short-Time Fourier Transform (STFT), Continuous Wavelet Transform (CWT), and Hilbert-Huang Transform (HHT) of each segment. As explained in Chapter 2, these are techniques used to analyze the frequency content of signals over time. STFT provides a sequence of Fourier transforms over short, overlapping time segments, allowing visualization of how frequency components evolve; CWT, on the other hand, uses wavelets of varying scales to decompose the signal, providing a multi-resolution analysis that can capture both high and low-frequency details; HHT involves empirical mode decomposition followed by Hilbert spectral analysis, offering a time-frequency representation that can adaptively reflect the intrinsic mode functions of the signal; and have had their use explored in other sEMG-based gesture classification works, such as Ozdemir et al.'s[237], who, although only using 4 sparse electrodes (plus a grounding one), *i.e.*, only 4 channels (as opposed to this work's 64 channels), was able to classify 7 hand gestures to accuracies of above 94%.

To obtain these spectral images, it was adopted the methodology of Section 4.5, with the following specifications: instead of utilizing all 64 channels of the used 8x8 grid, the most prominent CoG found within all the 500-sample length-long signals was acquired and taken as the representative signal used for processing – thereby reducing computation time and resources, while still keeping some of the relevant spectral information that describes the full set of signals.

Additionally, a limit for the number of IMFs for obtaining the Hilbert spectrum was set and equalled to 10 (reduced, so as to also avoid large file sizes given the available resources), ensuring that its resulting image shape stays constant for all observations – this is done by either truncating (when an observation has more than 10), or by padding (when it has fewer than 10).

With that, from each 500-sample length, 1000-Hz, averaged signal resulted three images: STFT of shape (5, 129); CWT of shape (500, 127); and HHT of shape (500, 10). Visually exemplified in the images of Figure 4.19, acquired wrist flexion of DMD patient 3.

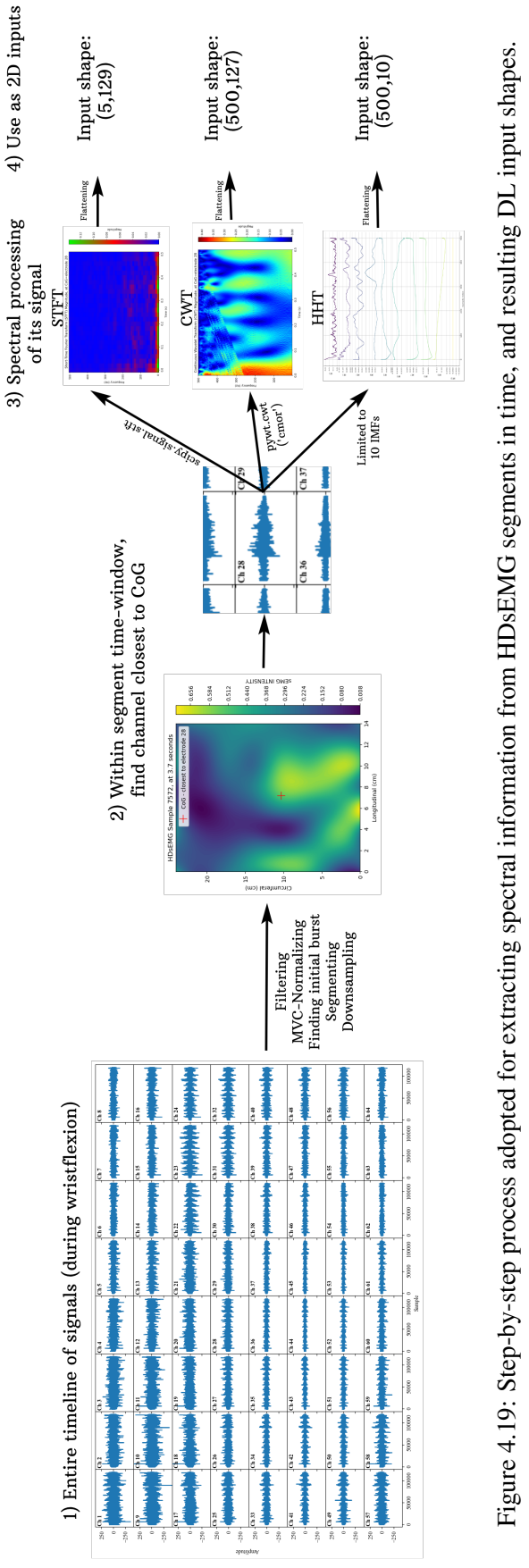


Figure 4.19: Step-by-step process adopted for extracting spectral information from HDsEMG segments in time, and resulting DL input shapes.

4.10.8 Neural Network Architectures

In the domain of gesture classification signals (in this case, also the pathology of the individual performing said gesture), choosing an appropriate neural network architecture is crucial for achieving accurate and robust classification performance. This subsection explores three distinct architectures tailored to handle the specific characteristics of the dataset provided; one can extend its results to new datasets, via either transfer learning from the trained models, or re-adoption of the architecture found to be most adequate and robust among the ones studied herein.

4.10.8.1 HDsEMG Images as Input Data

For studying the efficacy of HDsEMG signals (8 channels by 8 channels) as input data, each segment (blocks of 500-samples per signal) is fed into the networks considered hereafter. Furthermore, since this is a pipeline that is made to be adaptable and applicable to any HDsEMG grid, the relevance of the 8x8 shape is itself ignored, and the input shape is concatenated, going from (8,8,500) to (64,500).

For this work, added complexity was given to each new network – to assess effect on classification performance –, starting from an LSTM-only one, to a CNN+LSTM, to a CNN+Transformer+LSTM, hereafter detailed.

Summarily, these architectures are detailed in Table 4.7, with n_classes being equal to 14 (7 gestures $\times$ 2 conditions). In the Table are also descriptions that evidence why these may be viable architectures to use for HDsEMG gesture classification; similar networks have also been adopted in past works to high accuracies, discussed in Chapter 3.

4.10.8.2 Spectral Images as Input Data

To study the impact of STFT+CWT+HHT's on classification accuracy, its images were used as input – three separate sets, inserted into three separate parts of the network (branches); resulting outputs of each branch are then concatenated and go through a common 4-layer Dense process, with the first three followed by batch normalization and dropout; similarly, a second approach was done such that a new branch would be added, with a HDsEMG set. These details are found in Table 4.8.

To assess the impact of both approaches adopted simultaneously, both groups of inputs were also considered: these two are taken into parallel networks (branched as in the previous case – similarly, only CNN+LSTM was used, discarding the other two, as addressed in Chapter 5).

4.10.9 Classification Metrics and Evaluation

To assess each model's classification ability, and learning efficiency, some data was left out of the training: for testing with unseen data, the sets were randomly permutated (to avoid biases) and then split (with *train_test_split*) into training (80%) and testing (remaining 20%), stratified in relation to labels y, to ensure equal distributions of classes.

Table 4.7: Summary of Neural Network Architectures used for pre-processed HDsEMG inputs.

Architecture	Input Shape	Layers and Details	Description
LSTM Only	(64, 500)	- LSTM(64, return_sequences=True) - Dropout(0.5) - LSTM(64) - Dropout(0.5) - Dense(64, ReLU) - Dense(n_classes, softmax) Total params: 71118; Trainable params: 71118; Non-trainable params: 0	LSTMs are well-suited for sequential data as they can capture long-term dependencies and temporal patterns, making them effective for time-distributed signals[181].
CNN+LSTM	(64, 500)	- Conv1D(64, kernel_size=3, ReLU) - MaxPooling1D(pool_size=2) - Conv1D(64, kernel_size=3, ReLU) - MaxPooling1D(pool_size=2) - Dropout(0.5) - Reshape - LSTM(64, return_sequences=True) - Dropout(0.5) - LSTM(64) - Dropout(0.5) - Dense(64, ReLU) - Dense(n_classes, softmax) Total params: 95822; Trainable params: 95822; Non-trainable params: 0	Combining CNNs and LSTMs leverages the feature extraction capabilities of CNNs and the temporal modeling of LSTMs, making this architecture robust for spatial and temporal patterns in HDsEMG data[128].
CNN+Transformer+LSTM	(64, 500)	- Conv1D(64, kernel_size=3, ReLU) - MaxPooling1D(pool_size=2) - Conv1D(64, kernel_size=3, ReLU) - MaxPooling1D(pool_size=2) - Dropout(0.5) - Reshape - 2 Transformer Blocks (LayerNormalization, MultiHeadAttention, Add, Dense) - LSTM(64, return_sequences=True) - Dropout(0.5) - LSTM(64) - Dropout(0.5) - Dense(64, ReLU) - Dense(n_classes, softmax) Total params: 262222; Trainable params: 262222; Non-trainable params: 0	This hybrid model integrates CNNs for spatial feature extraction, Transformers for capturing long-range dependencies, and LSTMs for temporal processing, providing a comprehensive approach for complex HDsEMG signals[166].

Table 4.8: Summary of Neural Network Architectures used for spectral-based HDsEMG inputs.

Architecture	Input Shape	Layers and Details	Description
CNN+LSTM	X_STFT: (129, 5) X_CWT: (127,500) X_HHT: (10,500)	- Input layers for STFT, CWT, HHT data; three branches - Conv1D + BatchNormalization + MaxPooling1D - Conv1D + BatchNormalization + MaxPooling1D - LSTM(32) for STFT data - LSTM(32) for CWT data - LSTM(32) for HHT data - Concatenate LSTM outputs - Dense(256, ReLU) + BatchNormalization + Dropout(0.5) - Dense(128, ReLU) + BatchNormalization + Dropout(0.5) - Dense(64, ReLU) + Dropout(0.5) - Dense(n_classes, softmax) Total params: 99902; Trainable params: 98846; Non-trainable params: 1056	CWT provides localized frequency information with varying resolutions, STFT captures signal changes in both time and frequency domains with fixed resolution, and HHT adapts to the signal's intrinsic modes, offering a detailed decomposition. Together, they offer a rich set of features that the CNN can learn spatial patterns from, while LSTM handles temporal dependencies, improving classification accuracy[237; 238; 239].
CNN+LSTM	X_HDsEMG: (64,500) X_STFT: (129, 5) X_CWT: (127,500) X_HHT: (10,500)	- Input layer for HDsEMG data - CNN layers + LSTM for HDsEMG data (Table 4.7) - Input layers and branch functions for STFT, HHT, CWT data - Conv1D + MaxPooling1D + LSTM for each data type - Concatenate LSTM outputs - Dense(128, ReLU) + BatchNormalization + Dropout(0.5) - Dense(n_classes, softmax) Total params: 196302; Trainable params: 196046; Non-trainable params: 256	Adding raw HDsEMG as an input alongside transformed features (CWT, STFT, HHT) can enhance classification performance because it preserves the original signal information that might be lost or overlooked during transformations. While the transforms highlight time-frequency characteristics, the raw HDsEMG data retains the complete spatial and temporal details, which could capture additional patterns or nuances. By incorporating both raw and transformed inputs, the CNN+LSTM model can learn from a more comprehensive feature set, potentially improving its ability to recognize subtle differences in muscle activity.

In all cases, the model was compiled using categorical cross-entropy loss and the Adam optimizer; and a learning rate of 0.0001. The training process was always of roughly 500 epochs and it included a validation split (of the training set itself) of 0.2, a batch size of 64, and the usage of the ModelCheckpoint callback to save the best model based on validation loss.

To evaluate the models' performance, accuracy was considered, both for validation set and test set; likewise, the validation loss was examined – given the constant number of epochs, one can have a better assessment of a model's learning efficiency through not just accuracy but also loss, since a decreasing loss typically indicates that the model is improving in its predictions; whereas constant loss or loss that fluctuates without a decreasing trend might indicate issues such as insufficient learning rate, poor model architecture, or insufficient training data.

Chapter 5

Results and Discussion

When developing a software meant to be used either for research or especially for clinical assessments, it proves essential to ensure that the tools provided are precise and robust to intra or inter-subject variations. As such, in this chapter, the results of each processing and metric extraction technique are presented, upon being applied to synthetic signals whose ground-truths are known, or real signals from PD or DMD patients, followed by a discussion seeking to explain each algorithm's accuracy, reliability, repeatability, and eventual clinical applicability.

5.1 Pre-Processing Tools

FOG Dataset

Starting by detrending the FOG signals of subject 1, during task1 (of the FOG dataset), Figure 5.1 showed the implementation was adequate, leading to a reduction of the p-value and a more negative value of the ADF statistic: as established, this is compatible with what would be expected: ADF statistic should become more negative after detrending, because a more negative ADF statistic indicates stronger evidence against the null hypothesis that the "series" has a unit root (i.e., the series is non-stationary). Likewise, the p-value should decrease after detrending, indicating that the null hypothesis of non-stationarity can be rejected, suggesting that the detrended series is more likely to be stationary.

Visually speaking, one can see that, for a part of the raw signal that displays an approximation to a polynomial trending (though very minute), this approximation is nullified after detrending.

When applying a Butterworth filter, at a band-pass of 15-499Hz, to the signal, by inspecting the corresponding signal spectrum (Fourier Transform, shown in Figure 5.2), it can be seen that the intensity of the spectrum greatly reduces from 0 to 15 Hz; again, as expected; the signal too became smoother and less noise, evidencing that some of the removed frequencies may indeed have represented noise.

Once having applied a 50-Hz notch filter, with results shown in Figure 5.3, although not resulting in any stark contrast between original and filtered signal, one can indeed find that the frequency components found at that region reduced (indicated within a red box, in the Figure). To note that,

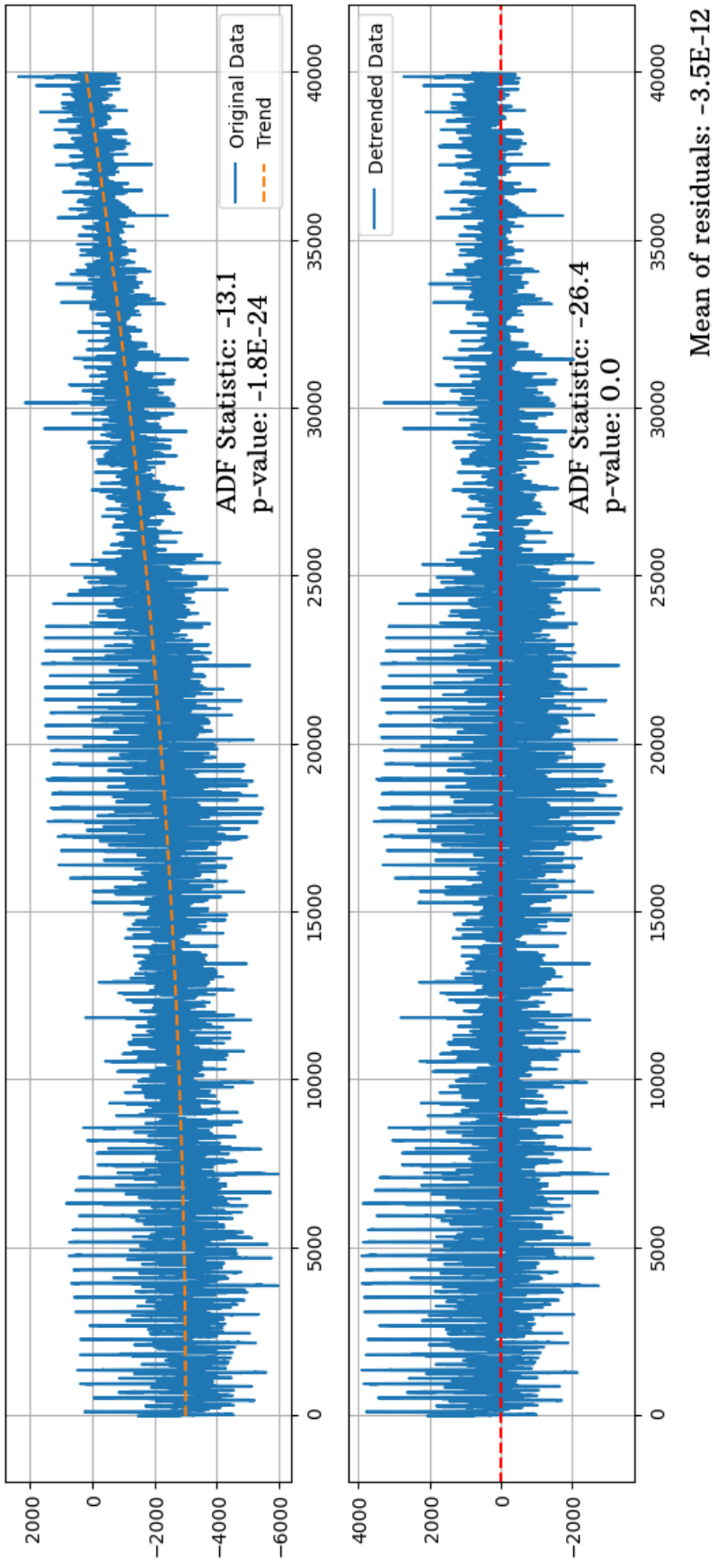


Figure 5.1: Detrending of FOG signal; x-axis representing time in samples (10000 samples corresponds to roughly 10 seconds, for instance), and y-axis representing unnormalized signal amplitude (original units unspecified).

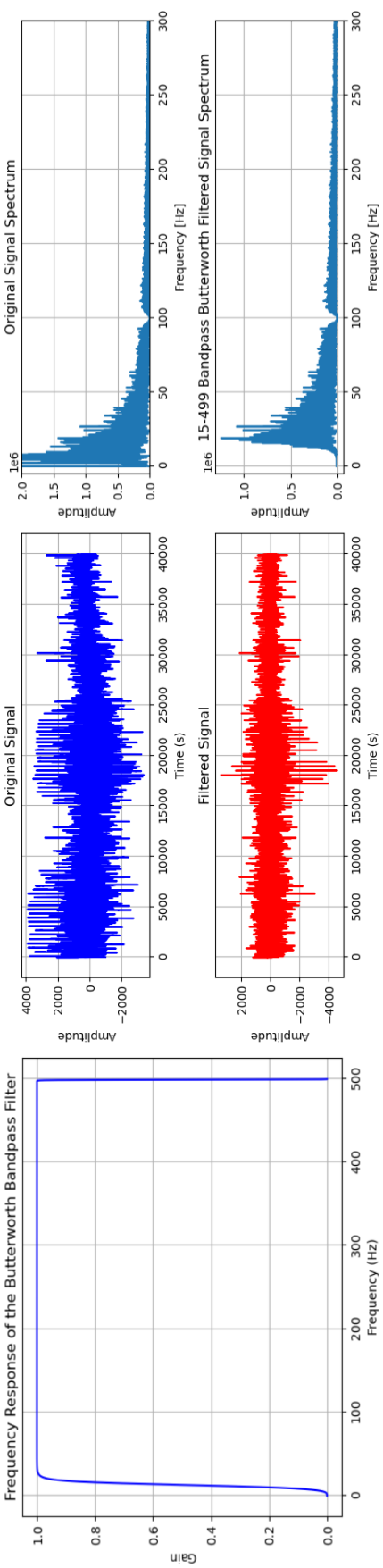


Figure 5.2: Butterworth 15-499Hz bandpass filter applied to unnormalized FOG signal, and its frequency response and FFT spectrum; time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

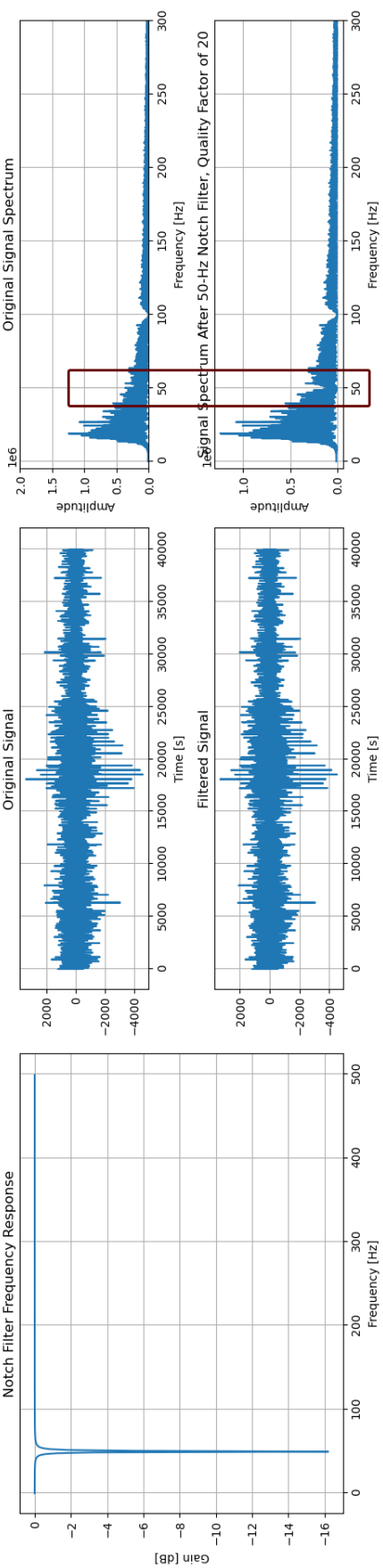


Figure 5.3: Notch filter at 50Hz applied to unnormalized FOG signal, and its frequency response and FFT spectrum; time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

while 50 Hz is often found to be unusually high, representing unwanted powerline noise, the original FFT spectrum did not appear to display this nature (possibly due to some pre-processing done previously by the dataset creators).

Other tools for minmax normalization and rectification lead to signals found in Figure 5.4. As evidenced, upon normalizing, the signals become contained to a [-1,1] interval (scaled in relation to its absolute maximum), while, upon full-wave rectification, all values become positive, and equal to their absolute. To note that, because this FOG dataset did not provide gain specification, nor voltage references, the remainder of this work will employ the normalized EMG1, EMG2, and EMG4 signals.

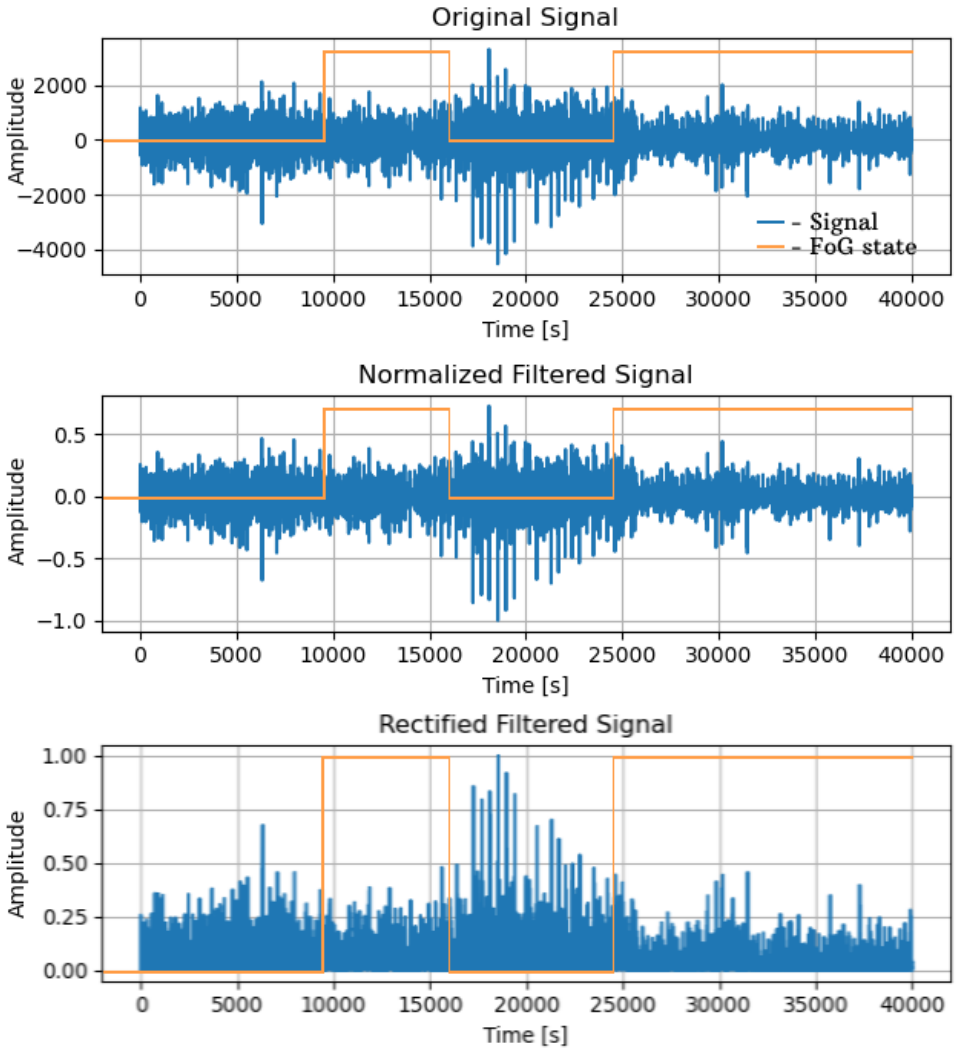


Figure 5.4: Minmax Normalization, from -1 to 1, applied to FOG signal, and its full-wave rectification. Additionally, an indication of FOG state, with 0 being "mobile" and 1 being "frozen". Time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

In assessing these tools, other bands for filtering were tried, with the 100-499Hz bandpass one displaying results shown in Figure 5.5. As shown, here too the FFT spectrum has lower density

at its bandpass interval (once again, showing the reliable implementation of these tools), on all 3 EMG signals; a dip at 100Hz is also noticed, for both bandpass cases, possibly indicating that a notch filter at 100-Hz at previously been applied to this dataset.

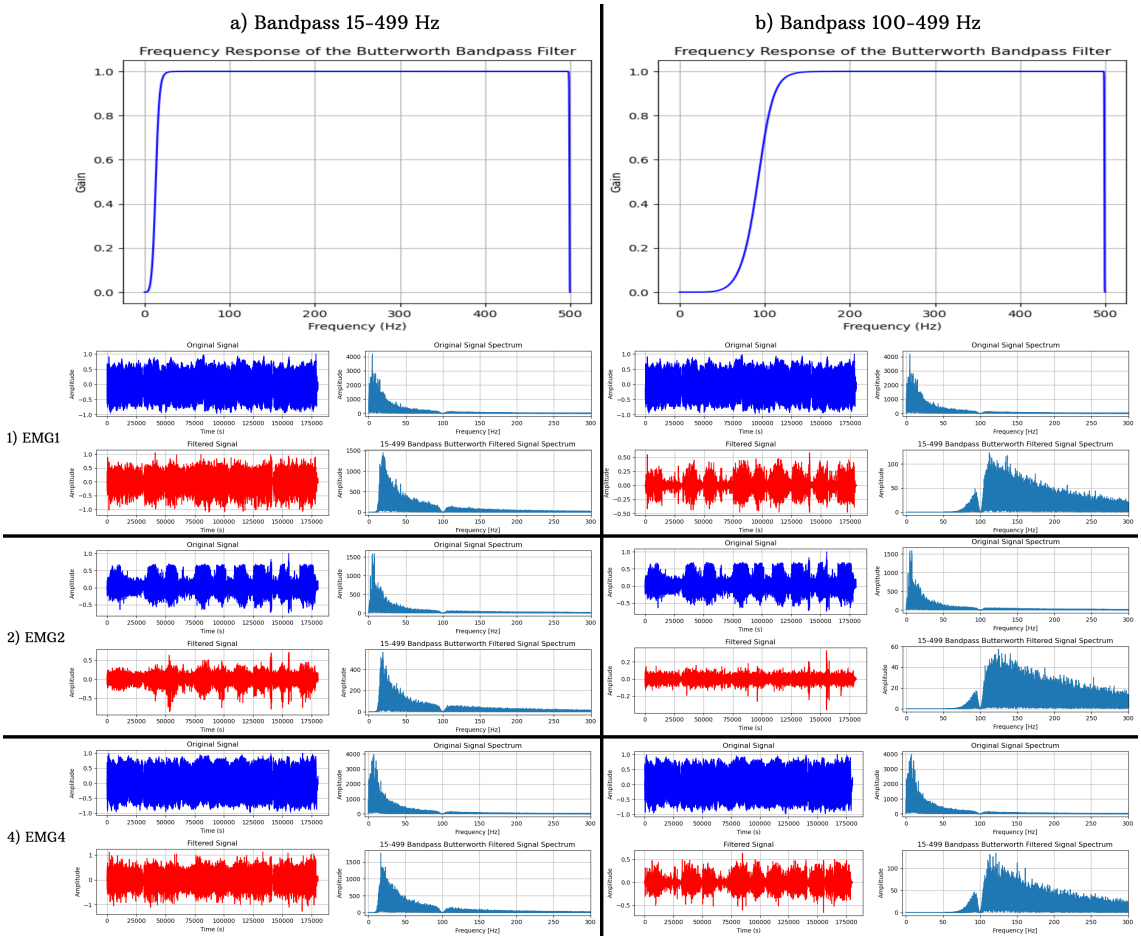


Figure 5.5: All three normalized FOG EMG signals and their FFT spectra, after applying a bandpass of (a) 15-499 Hz and (b) 100-499 Hz. Time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

This is further evidenced by looking at the power spectral density plot, shown in Figure 5.6 which also comply with the bandpass intervals, as expected, though the intensities appear generally higher for 15-499Hz than for 100-499Hz, even within the interval of the bandpass. Further discussion of this is had in Section 5.3, when discussing spectral analysis.

As these results are within what would be expected, one can also then conclude that from these pre-processing approaches it becomes possible to better study a clinical case: for instance, when looking at the 100-499Hz bandpass filter, while outside of the common range of 15-499Hz, it can be shown that it leads to pre-processed signals that better allow for differentiating between FOG states on the EMG1 and EMG4 channels (specifically for subject 1, during task 1). Figure 5.7 shows this – a much less intense signal is found during frozen states, which is compatible with the idea that the individual, being "frozen", is unable to move as has their muscle contractions

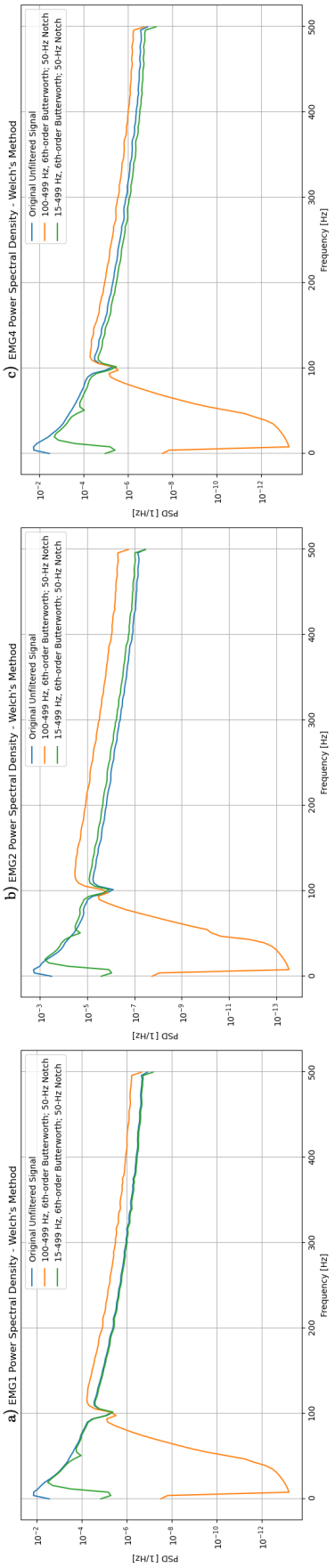


Figure 5.6: Power Spectral Density, through Welch's Method, of the original FOG signal, and of the signal after applying 15-499Hz bandpass or 100-499Hz bandpass, for (a) EMG1, (b)EMG2, (c)EMG4.

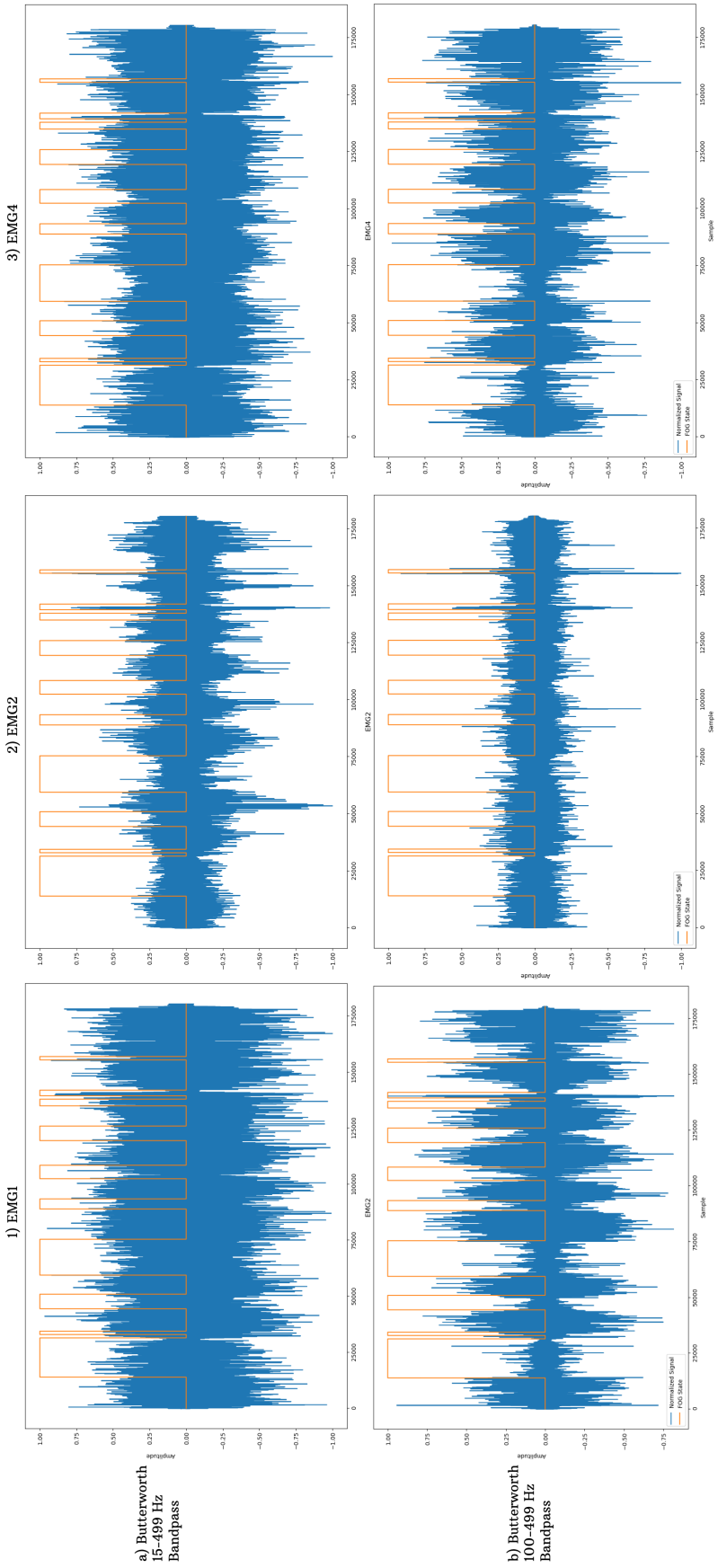


Figure 5.7: Comparison of entire normalized signals' timeline after a bandpass of (a) 15-499Hz, and (b) 100-499Hz; on all three EMG signals. Additionally, an indication of FOG state, with 0 being "mobile" and 1 being "frozen". Time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

canceled out (though not fully, which might explain non-null intensities even during freezing).

Without much processing required, save for the previous filtering steps, one could conceivably utilize these methods for monitoring a PD patient.

DMD Dataset

For the DMD dataset, specifically DMD patient 3 and healthy patient 2, during flexion and extension, this pre-processing can also be done (summarized in Figure 5.8) on each of the 64 channels. What can indeed be seen is that, after detrending, and on a 20-500Hz bandpass (keeping in mind that these signals are recorded at a 2048-Hz sampling rate), what was once a nigh-impossible task of identifying each muscle activation (though periodic) – especially during its raw stage of Figure 5.8(1) –, can now be done even with a thresholding approach.

Once normalized with respect to each subject's MVC, this threshold would be around 0.01 and 0.15 for extension of healthy patient and DMD patient, respectively; and around 0.02 and 0.2 for flexion, also respectively. While not much difference is found between gestures (intrasubject-wise). clinically, the starker differences found of MVC-bound thresholds between the two subjects (about 12 times higher for DMD patient) may indicate that, not only is a healthy subject's MVC much higher, but that perhaps a lot more effort need be done by the DMD subject in order to perform the same task. Once more, through the implemented pre-processing tools, it becomes possible to assert assessments that would otherwise prove unfeasible.

5.2 Amplitude Analysis

FOG Dataset

For the three EMG signals of FOG dataset, amplitude features were acquired, as displayed in Figure 5.9, which is using epoch windows of 1000 samples (1 second), in this case, focusing in on a smaller interval of time, where mobile state and frozen state happen twice each (during task 1). Something to note is that, because the signal is already rectified, its epoch average coincides with the ARV.

What's shown is that, during mobile states, the intensity metrics (including ARV, RMS, and mean square) are generally higher (by over 30% on all metrics, for EMG1), though variance remains roughly the same (as STD indicates in the error bars of the bar plots which show average of epochs inside each FOG state). For EMG2 and EMG4, variances are much higher during mobile state than frozen state, though, while still higher, the intensity metrics aren't much greater than that of frozen sometimes; this might show not only that, even for muscles that may not be as active during the performance of a task (muscle from EMG2 barely ever having RMS cross 0.2), variance may increase as a task is being successfully performed – perhaps due agitations or even PD tremor (which mostly occur when a muscle is at rest or less active).

Similarly, the implementation also allows for differentiation features (*i.e.*, derivative and integrals), as in Figure 5.10. In this case, what is shown is that the derivative displays a similar

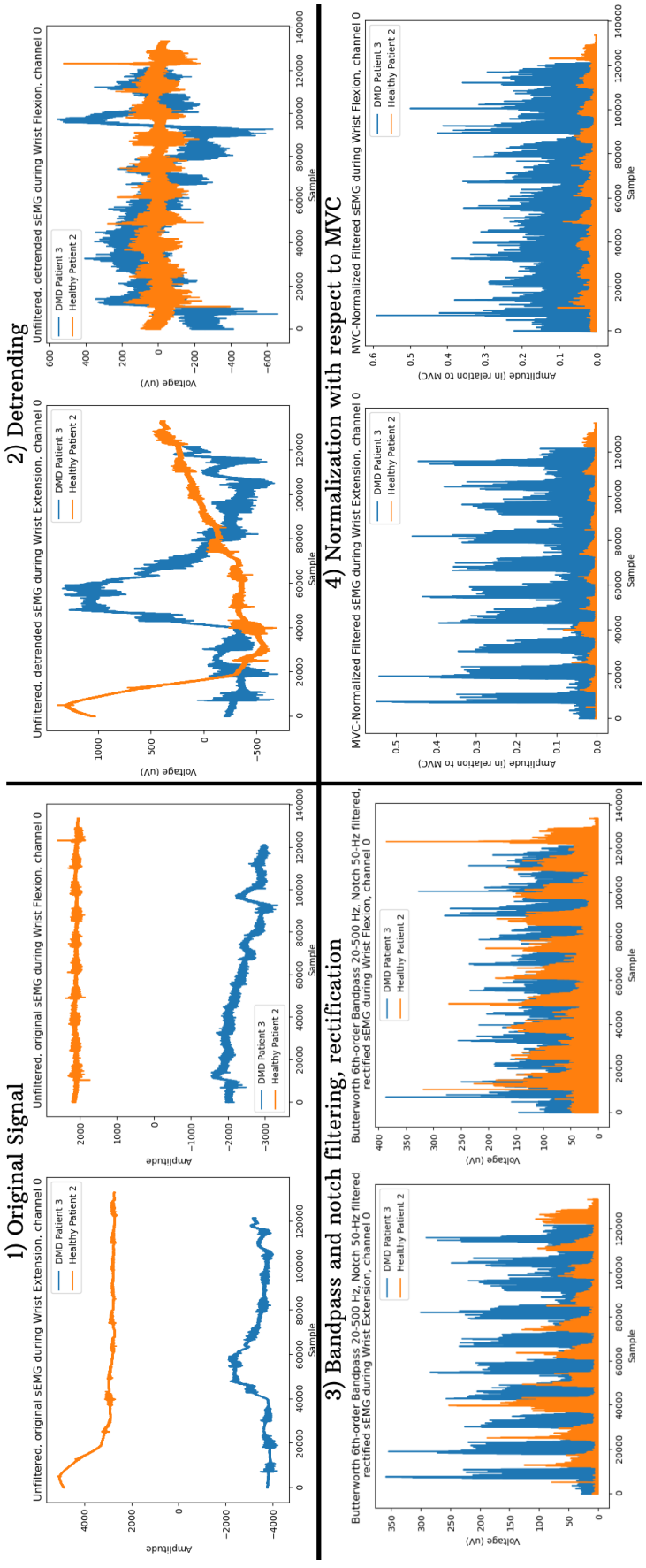


Figure 5.8: Pre-processing pipeline as applied to channel 0 of two patients (dystrophic and healthy) during wrist extension and flexion, from the DMD dataset; raw amplitude in μV. Time in samples (100000 samples corresponds to roughly 49 seconds, for instance).

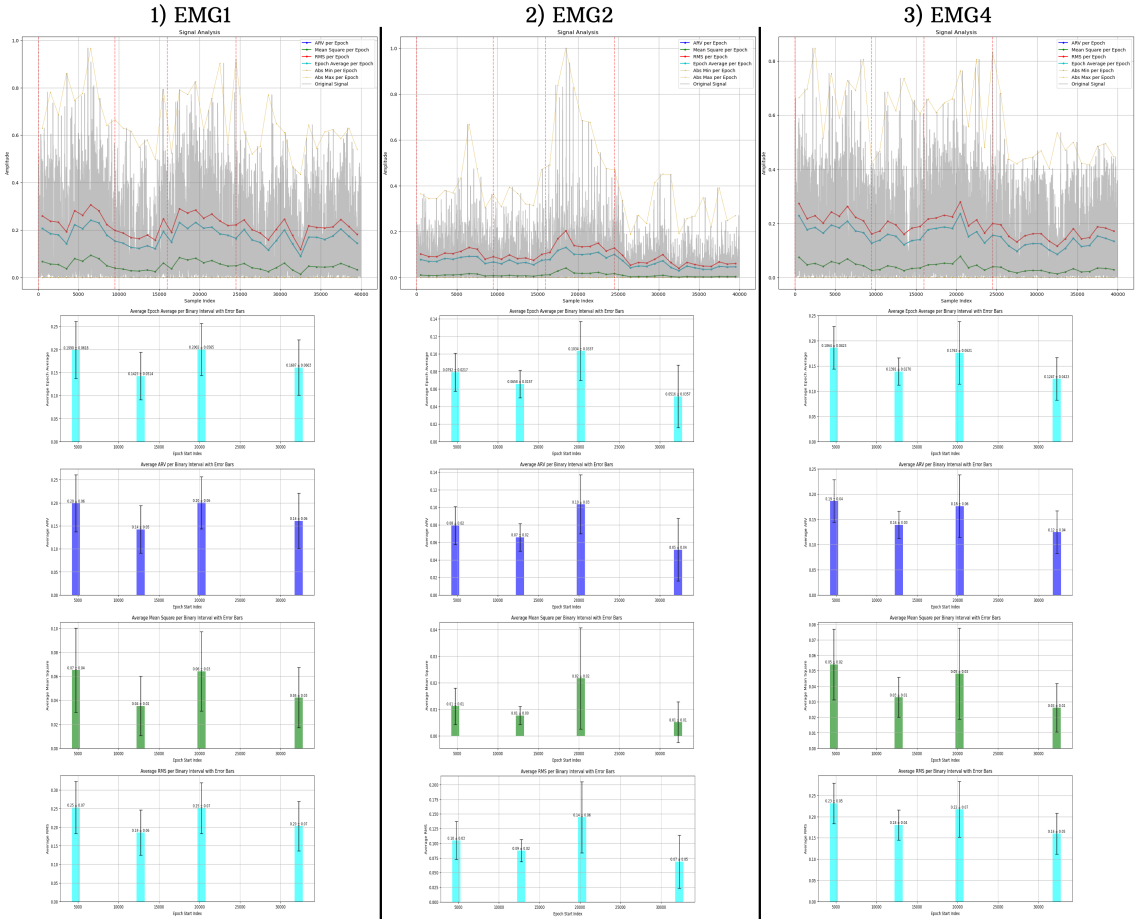


Figure 5.9: Amplitude features on a specific interval (not entire timeline), on 1000-sample epoch window, of subject 1 of FOG dataset, upon a 15-499Hz bandpass pre-processing, on (1) EMG1, (2) EMG2, and (3) EMG4. At the top, the rectified signal along ARV, Mean Square, RMS, Epoch Average, Minima and Maxima of each epoch; under it, bar plots of the average values of each metric at each FOG state switch. Error bars corresponding to 2STD; minmax-normalized signal amplitude. Time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

behaviour to that of the original signal, though for some cases (EMG1) perhaps facilitating in identifying FOG states, where ratio between intensities of derivative between mobile and frozen goes from about 2 (regular signal) to about 2.7 (derivative). Similar behavior described for RMS and the other features can be seen with the integral features, this is compatible with the notion that the signals that are more intense within an interval will lead to higher area under the envelope that describes it (*i.e.*, integral).

As a way to assess different bandpasses across the entire timeline (given 3000-sample epochs), the same can be done for the two cases (15-499Hz and 100-499Hz). As shows Figure 5.11a, the general behaviour between FOG states remains consistent, with amplitudes and variances being generally lower during freezing. These differences between mobile and frozen are all the starker upon a 100-499Hz bandpass (save for the EMG2 signal which appears to lead to better distinction at 15-499Hz) – that said, some noticeable differences can be found between the actual amplitudes

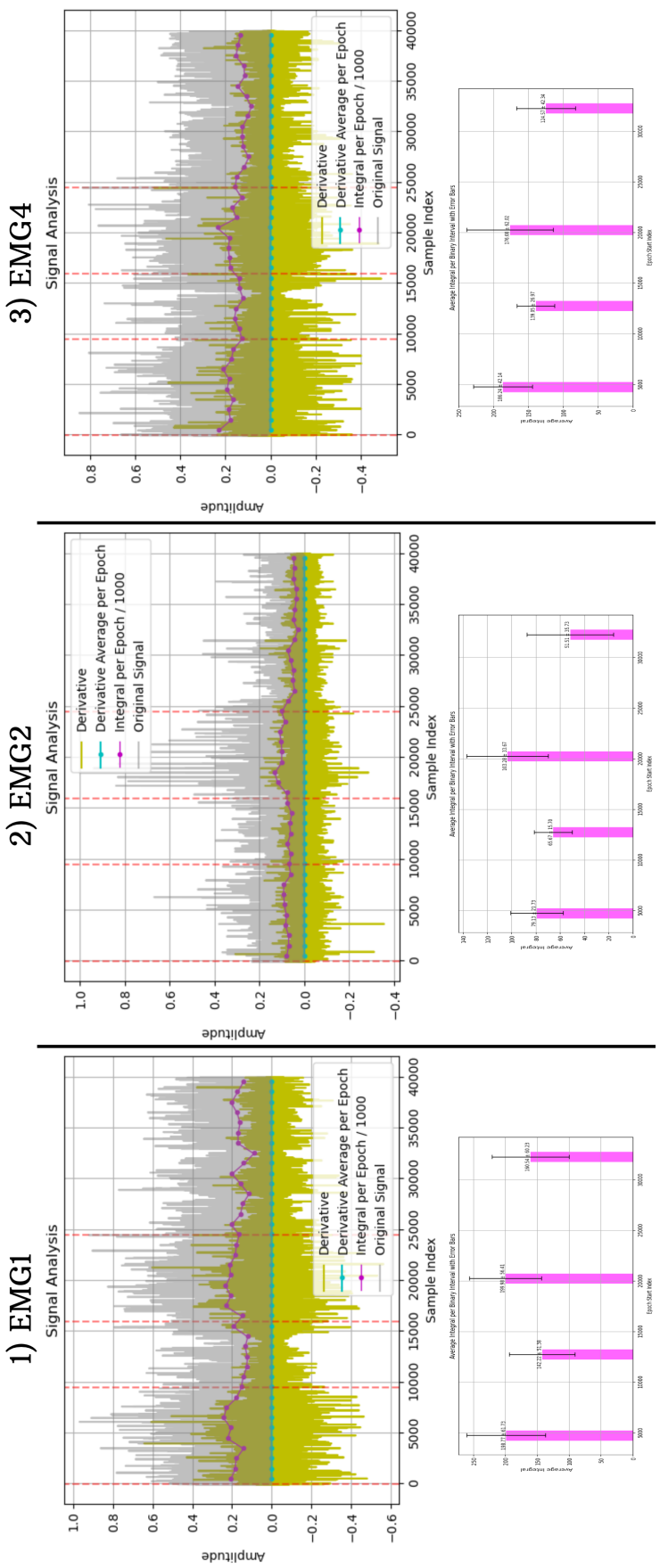
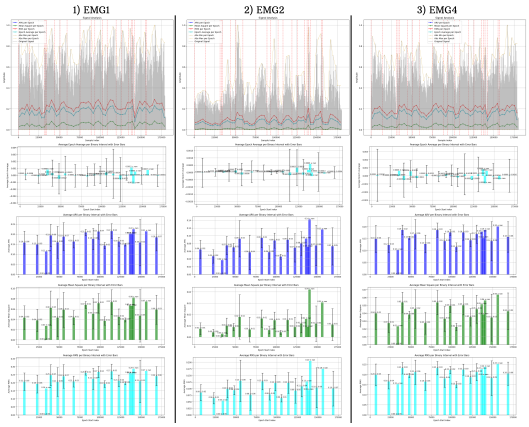
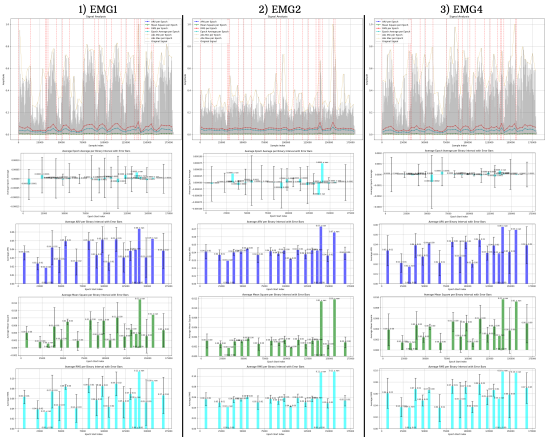


Figure 5.10: Differentiation features on a specific interval (not entire timeline), on 1000-sample epoch window, of subject 1 of FOG dataset, upon a 15-499Hz bandpass pre-processing, on (1) EMG1, (2) EMG2, and (3) EMG4. At the top, the rectified signal along derivative, and integral of each epoch; under it, the bar plots of average value of integral at each FOG state switch. Error bars correspond to 2STD; minmax-normalized signal amplitude. Time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

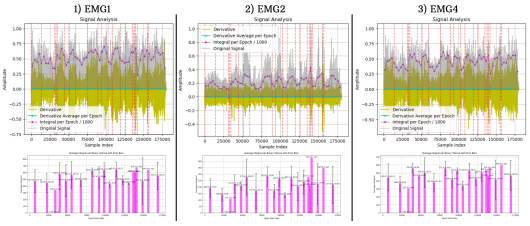
themselves.



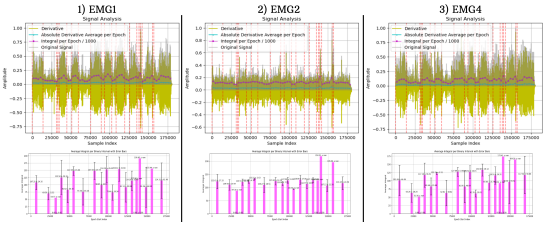
(a) Amplitude features, 15-499Hz bandpass



(b) Amplitude features, 100-499Hz bandpass



(c) Differentiation features, 15-499Hz bandpass



(d) Differentiation features, 100-499Hz bandpass

Figure 5.11: Amplitude and differentiation features on entire timeline, with 3000-sample epochs, for two different bandpass intervals; minmax-normalized signal amplitude. Time in samples (10000 samples corresponds to roughly 10 seconds, for instance).

This aspect is further summarized in Figure 5.12, which shows the overall averages for FOG states, given the two different bandpasses. What is shown is that the ratios between mobile and frozen are higher for 100-499Hz filtering, but that the average amplitude reduces greatly on all metrics (which would be expected, but may inform that either most of the 0-100Hz is mostly noise and that most of the task performed by the subject occupies higher frequencies, or that it is excluding some subtle activations that are less noticeable, yet still occur). For the task of identifying FOG, this may be adequate, but for the task of distinguish other particular aspects (such as gait cycles), perhaps some information is lost.

This Figure also shows that between 1000-sample epoch and 3000-sample epochs, beyond the STDs (and save for the integral metric, which shows differences due to the smaller interval used by the 1000-sample epoch case), the averages don't change much (though they are generally lower for 3000-sample).

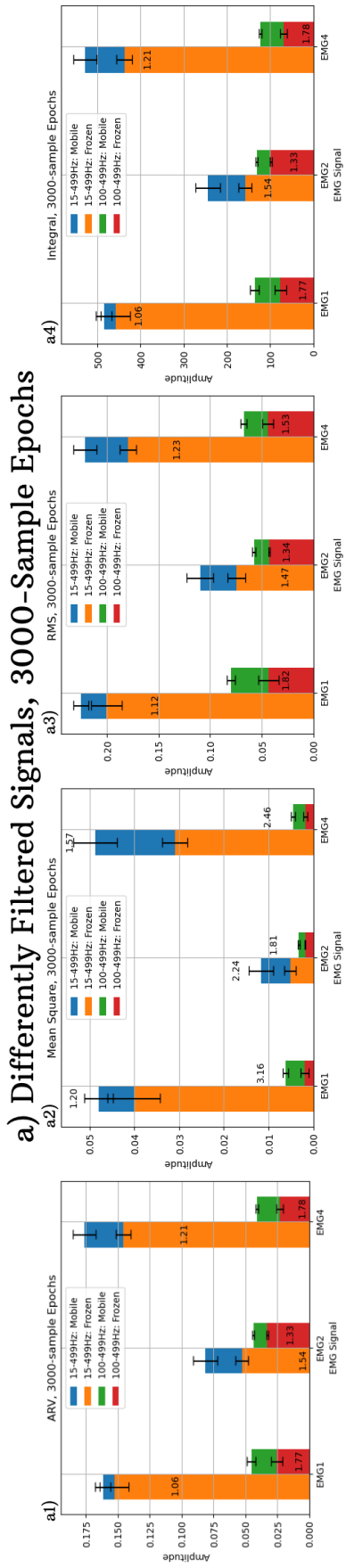


Figure 5.12: Bar plot summarizing resulting amplitude averages. (a) Compares different bandpasses on epochs with same timeline and size; (b) compares same bandpasses on epochs with different time interval and size.

HDsEMG Topographical Mapping & CoG

DMD Dataset

With the DMD dataset, the implemented tools allow for 2D visualization of the grid (along corresponding axis distances), as can be seen in Figure 5.13's HDsEMG intensity row. When comparing it to the remaining metrics (RMS and ARV, which, unlike the first that shows the signals at a given instant, these show an average from the epoch), it becomes possible to determine the regions of higher intensity; what can be seen is that, through time, activation distributions seem to remain roughly the same, though intensities lower, which is compatible with the idea that, as fatigue happens, the ability to perform as task may decrease. When distinguishing flexion and extension, distributions do change somewhat, though they don't often coincide between DMD and healthy subject. This can indicates that activation of muscles depends on motor ability, as DMD patients show to exert more effort (as also shown by the much higher values of MVC-normalized signals and RMS within the Figure) and require activation of more part of the muscle than the healthy individual.

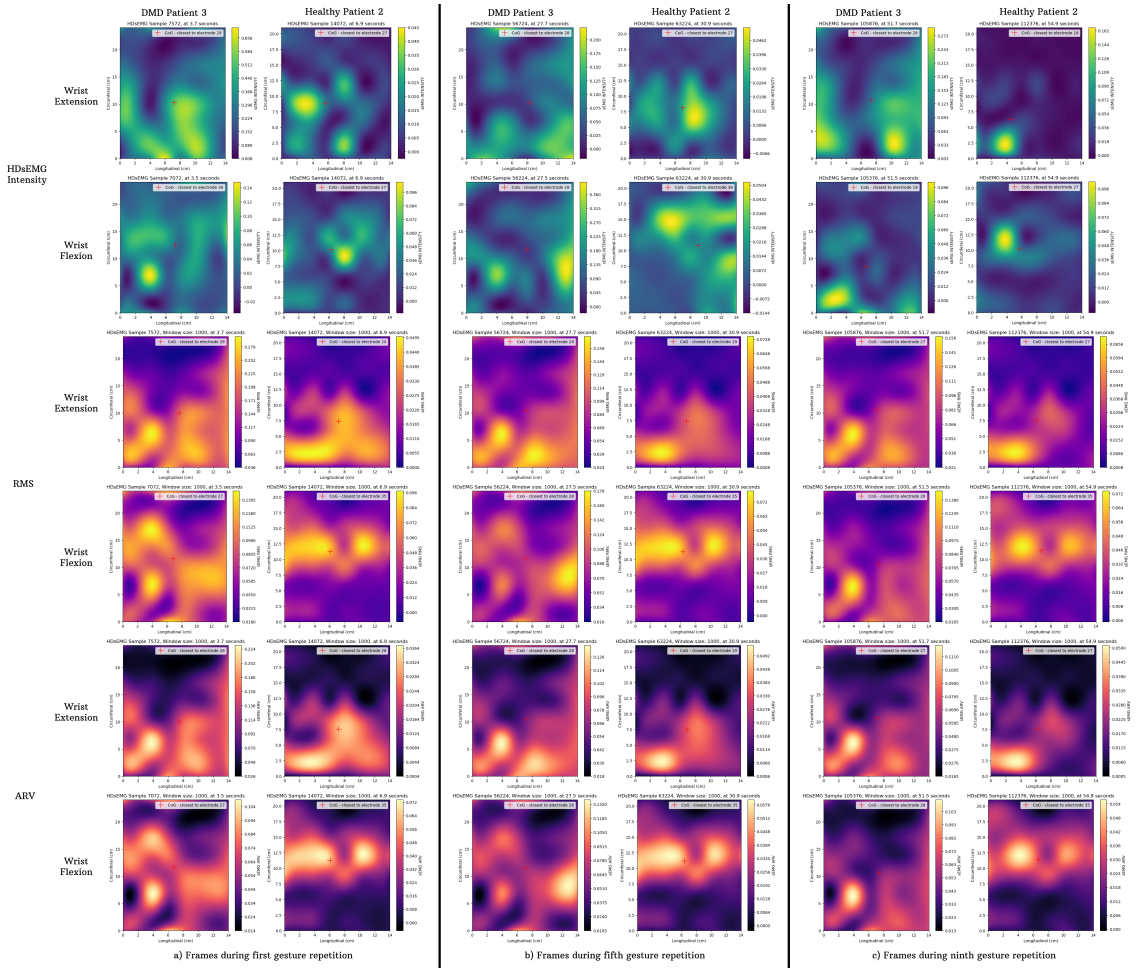


Figure 5.13: Topographical mapping on DMD dataset, at three different gesture repetitions; MVC-normalized signal amplitude.

This change in distribution can be evidenced by the CoG, also shown in the Figure 5.13, and plotted in Figure 5.14. By the way it varies, one may even be able to utilize it for detecting moments of activation, as the y-axis of the grid (along the fibers of the arm) displays changes of over 2cm from gesture to rest.

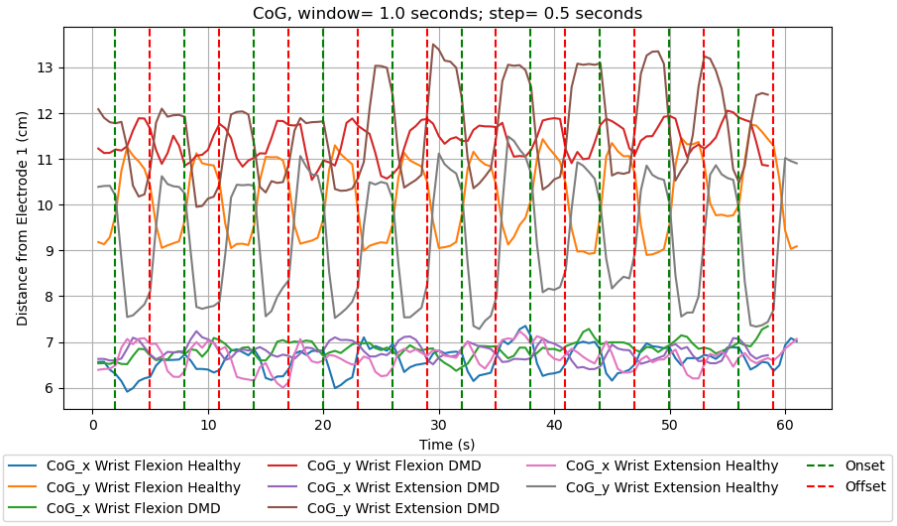


Figure 5.14: Center of Gravity in x and y axis on entire timeline.

These ideas are summarized in Table 5.1, which shows quantitatively that, as a metric, CoG can be used for analyzing movements on different health conditions. DMD, for instance, showed a lesser ability to redistribute its muscle activations (as evidenced by lower mean peak-to-peak displacement on both x and y axis), even despite the higher amplitude seen before (and presumed higher effort). In fact, during flexion, the DMD patient was unable to perform even 49% of the displacement seen in healthy flexion, with its STD low as well, which would suggest that barely any movement was, on average, performed between rest and gesture.

Table 5.1: Summary of CoG during Wrist Flexion and Extension, in Healthy (patient 2) and DMD (patient 3) Conditions.

Condition	**Movement**	**Maximum Distance (cm)**		**Minimum Distance (cm)**		**Mean Peak-to-Peak Displacement ± STD (cm)**	
		CoG_x	**CoG_y**	**CoG_x**	**CoG_y**	**CoG_x**	**CoG_y**
Healthy	Wrist Extension	7.2	11.5	6.0	7.3	0.3 ± 0.2	2.3 ± 1.5
	Wrist Flexion	7.4	11.8	5.9	8.9	0.4 ± 0.3	1.3 ± 1.0
DMD	Wrist Extension	7.2	13.5	6.4	10.0	0.2 ± 0.2	1.4 ± 1.2
	Wrist Flexion	7.3	12.1	6.4	10.5	0.2 ± 0.1	0.5 ± 0.4

With that, amplitude features have been shown to be able to assess the conditions by which EMG has been generated, consistent with what one would expect. The tools implemented herein show to be robust for analyzing how a person's gait may change (upon a freezing of state, for example) or how its activation distribution may be affected by gesture being performed, or health condition.

5.3 Spectral Analysis

FOG Dataset

In studying frequency and time-frequency aspects of the FOG state, knowing how the original signal's spectra compare to their processed version is of relevance, an assessment which has been made in Subsection 5.1, wherein FFT spectra are utilized and display behaviours compliant with the expected bandpasses given. Further analyses can be done using the PSD spectra, also shown in that subsection, in Figure 5.6, using Welch's method (by the package *scipy.signal.welch*), which, if comparing with a more traditional method of implementation (`np.abs(fft_filtered)[:len(freqs)//2]`), can give an idea of these implementations' reliability. Summarized in Table 5.2 are the PSD averages (within the corresponding bandpass interval), computed on both methods. What's shown is that both demonstrate a marked lowering of the average (ratios below 1), though the traditional method appeared to be more efficient at dissipating these frequencies (much smaller ratios, closer to 0); even still, Welch's method shows to be an adequate implementation, displaying a mean reduction below 35% of the original raw-signal mean for 15-499Hz, and mostly below 1% for 100-499Hz.

Table 5.2: Resulting average PSD before and after pre-processing (two different bandpasses), as computed by two different PSD methods (traditional and Welch's method)

EMG	**15-499 Hz (pre)**	**15-499 Hz (post)**	**100-499 Hz (pre)**	**100-499 Hz (post)**
Traditional PSD				
EMG1	35342.3161	0.00087421	5301.3474	5.2969e-06
Ratio	2.4736e-08		9.9915e-10	
EMG2	826.2616	2.9578e-05	826.2616	2.6919e-06
Ratio	3.5797e-08		3.2579e-09	
EMG4	34927.0498	0.0096673	826.2616	2.6919e-06
Ratio	2.7678e-07		3.2579e-09	
PSD (Welch's Method)				
EMG1	0.00376689	0.00080884	0.00057967	1.0698e-05
Ratio	0.21472		0.01846	
EMG2	0.00048201	0.00016242	7.4855e-05	7.7654e-06
Ratio	0.33697		0.10374	
EMG4	0.00469488	0.00060763	0.00072082	9.3053e-06
Ratio	0.12942		0.01291	

When wanting to plot spectral coherence, with this implementation, one can do so too, as shown in Figures 5.15 and 5.16; as it shows, coherence seems to coincide between the EMG1&2 pair and EMG2&4, but not EMG1&4 – this may indicate that the muscles captured by EMG1 and EMG4 may not be as synchronized as those captured by EMG1 & EMG2 or EMG2 & EMG4. Even so, the difference in coherence is quite minute.

For time-frequency, in Figure 5.17 are shown different spectra that can be acquired. In the case of 5.17b, this one plots STFT as well as mean, median, and power. What they show is that, through time, the intensities do change, with higher medians being observed when the PD subject is mobile. All spectra display behaviors that are in conformity with what would be expected from wide ranging signals that don't necessarily display many periodic movements.

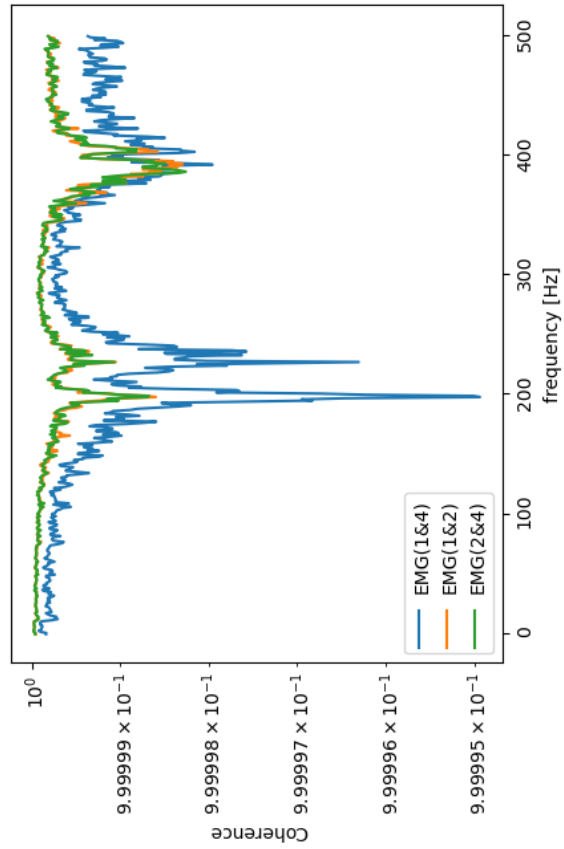


Figure 5.16: Spectral coherence, on FOG signals of patient 1, task 1, for bandpass 100-499Hz.

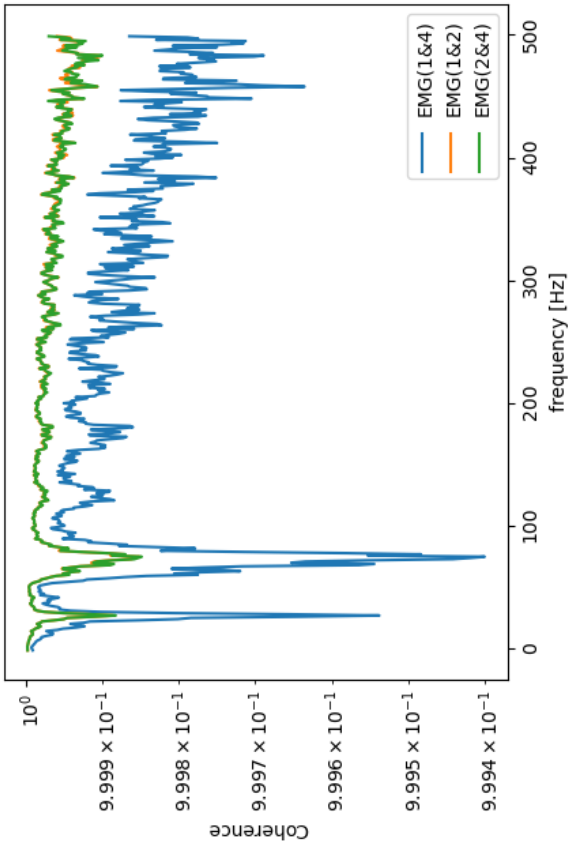


Figure 5.15: Spectral coherence, on FOG signals of patient 1, task 1, for bandpass 15-499Hz.

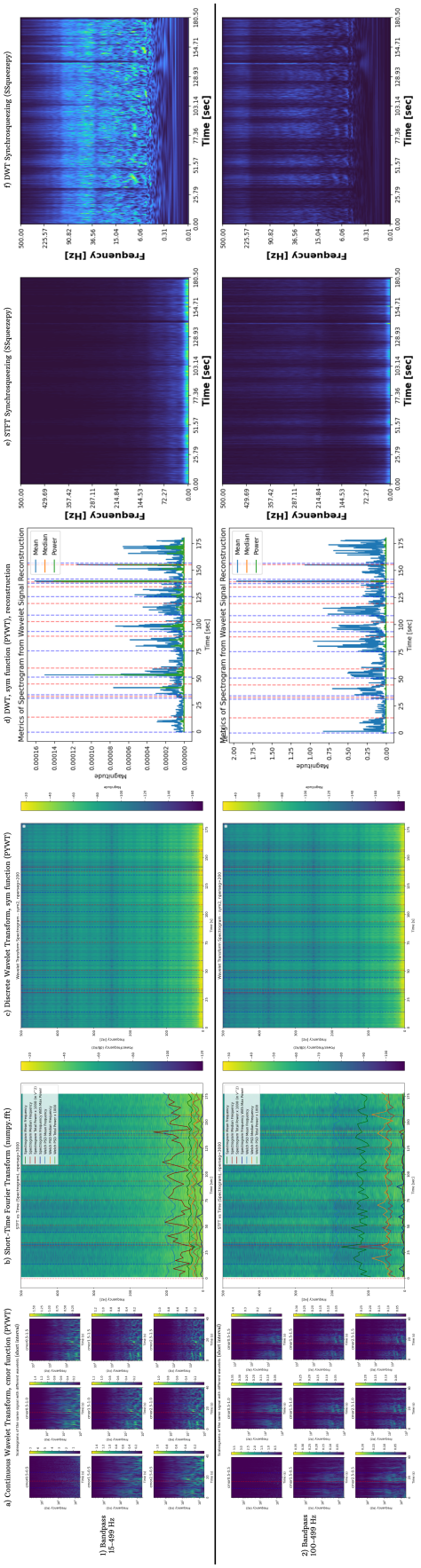


Figure 5.17: Time-frequency spectra for FOG patient 1 task 1, on signal EMG1, given the two bandpasses.

DMD Dataset

When looking at its pre-processed signals, the PSD in Figure 5.18 can be seen. When pre-processing, a relevant aspect is the fact that EMG signals do indeed have frequencies which require notch-filtering. The herein used DMD signals showed that the PSD of the raw signal, as shown in Figure 5.18, displays a peak of frequency at 50Hz, likely related to the baseline noise.

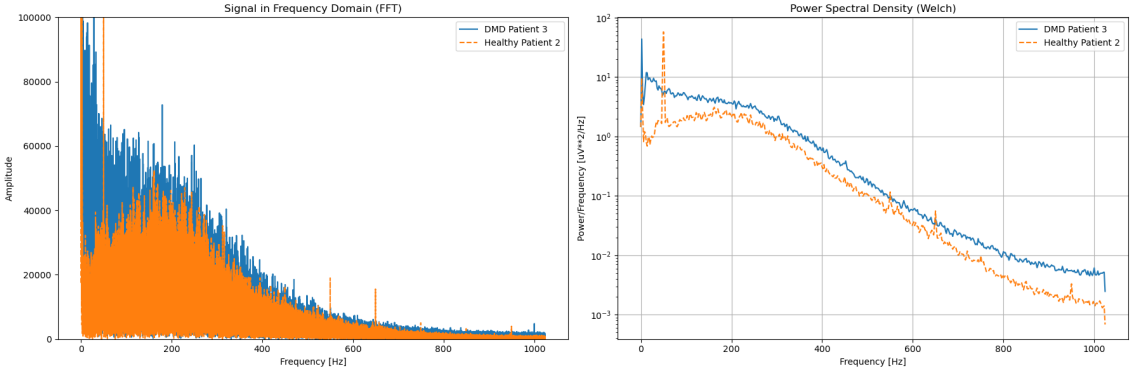


Figure 5.18: Power spectral density of raw signal for DMD and Healthy patients, during flexion.

Once filtered, the implementations also allow for visualization of CWT, STFT and HHT, as seen in Figure 5.19. What is quite evident are the intensities which increase upon each gesture, utilizing higher density of frequencies at the lower end of its spectrum (0 to 100 Hz).

Because a part of this work explores using these spectra upon a downsampling from the current sampling rate to 1000 Hz, and segmentation between 500-sample windows, in Figure 5.20 one can view the CWT and STFT spectra resulting from this downsampling and segmentation. What is shown is that, at different gesture repetitions, the distribution does not appear to change much, prioritizing lower frequencies at the start and end of the time window; the amplitudes themselves differ from DMD to healthy patient.

Similarly, for the HHT results within the same time windows, shown in Figure 5.21, values display a similar behaviour, going from noisier to smoother with each IMF (in these cases, IMF 10 are fully null in all cases), though wrist extension by DMD appears to be generally higher at most times, followed by wrist flexion also by DMD.

That said, these data are often harder to analyze visually or even quantitatively; to understand if a Deep-Learning model is able to learn from these images, Section 5.9 delves into this discussion.

With these spectral tools, implemented both for quantitative and visual assessments, it becomes possible to understand minutiae that can be found upon differing tasks and pathologies, and the herein shown spectra, from the distributions that may commonly be found in similar signals and settings, do evidence a reliable implementation.

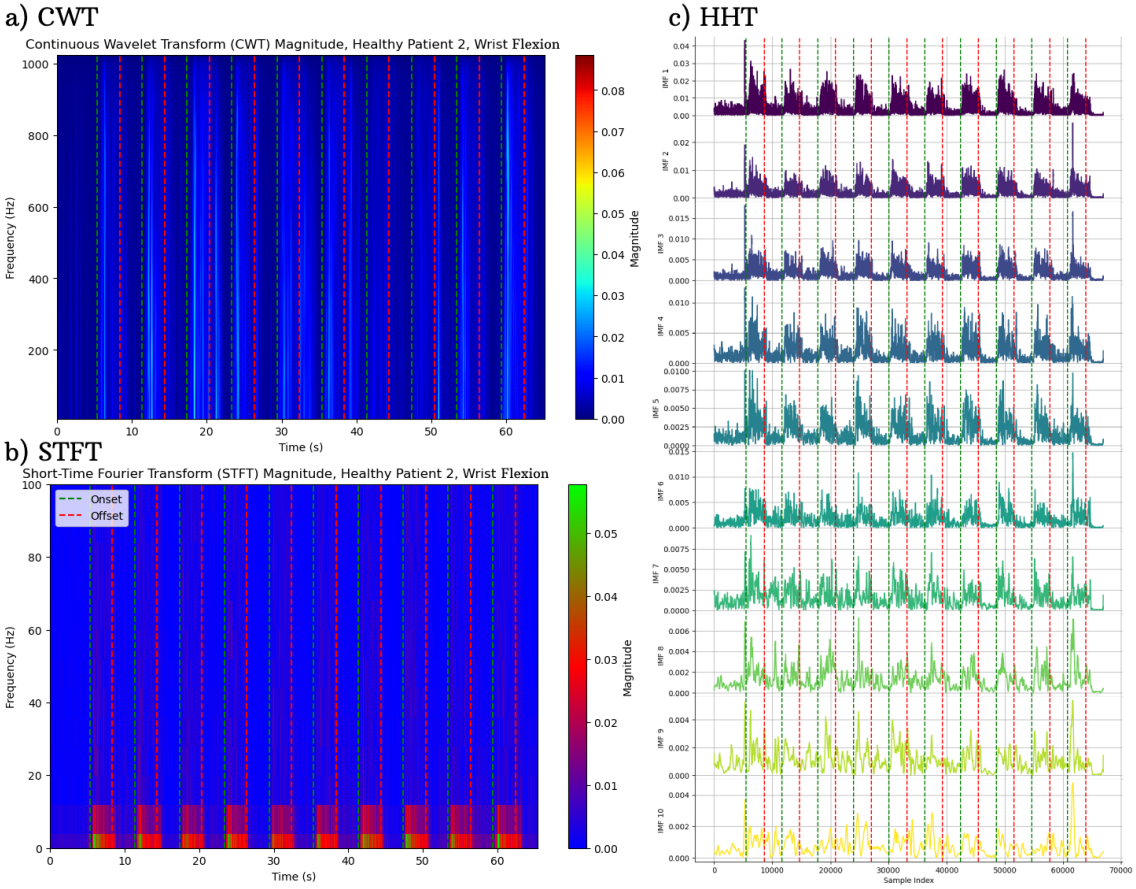


Figure 5.19: Time-Frequency Analysis On All Repetitions at electrode 28 (entire timeline, normalized signal at 2048-Hz sampling rate) – Healthy Patient 2, Wrist Flexion, with (a) Continuous Wavelet Transform; (b) Short-Time Fourier Transform, 256-sample windows; (c) Hilbert-Huang Transform (10 IMFs).

5.4 Timing Analysis

After running each detection algorithm (having defined each one's parameters so that they correctly identify at least one activation), the resulting deviations are shown in Table 5.3.

Table 5.3: Time deviations, and mean error, of each onset/offset detection method.

Detection Algorithm	**Average deviation ± STD (seconds)**		**Mean Error (%)**
	Onset	**Offset**	
hodges_bui	0.3 ± 0.3	0.4 ± 0.3	7
bonato	0.3 ± 0.1	0.3 ± 0.2	6
lidierth	0.4 ± 0.1	0.5 ± 0.3	9
abbink	0.4 ± 0.2	0.4 ± 0.2	8
solnik	0.3 ± 0.1	0.3 ± 0.2	6
silva	0.3 ± 0.1	0.4 ± 0.2	7
londral	0.3 ± 0.1	0.4 ± 0.1	7
TKEO	0.2 ± 0.1	0.2 ± 0.1	4

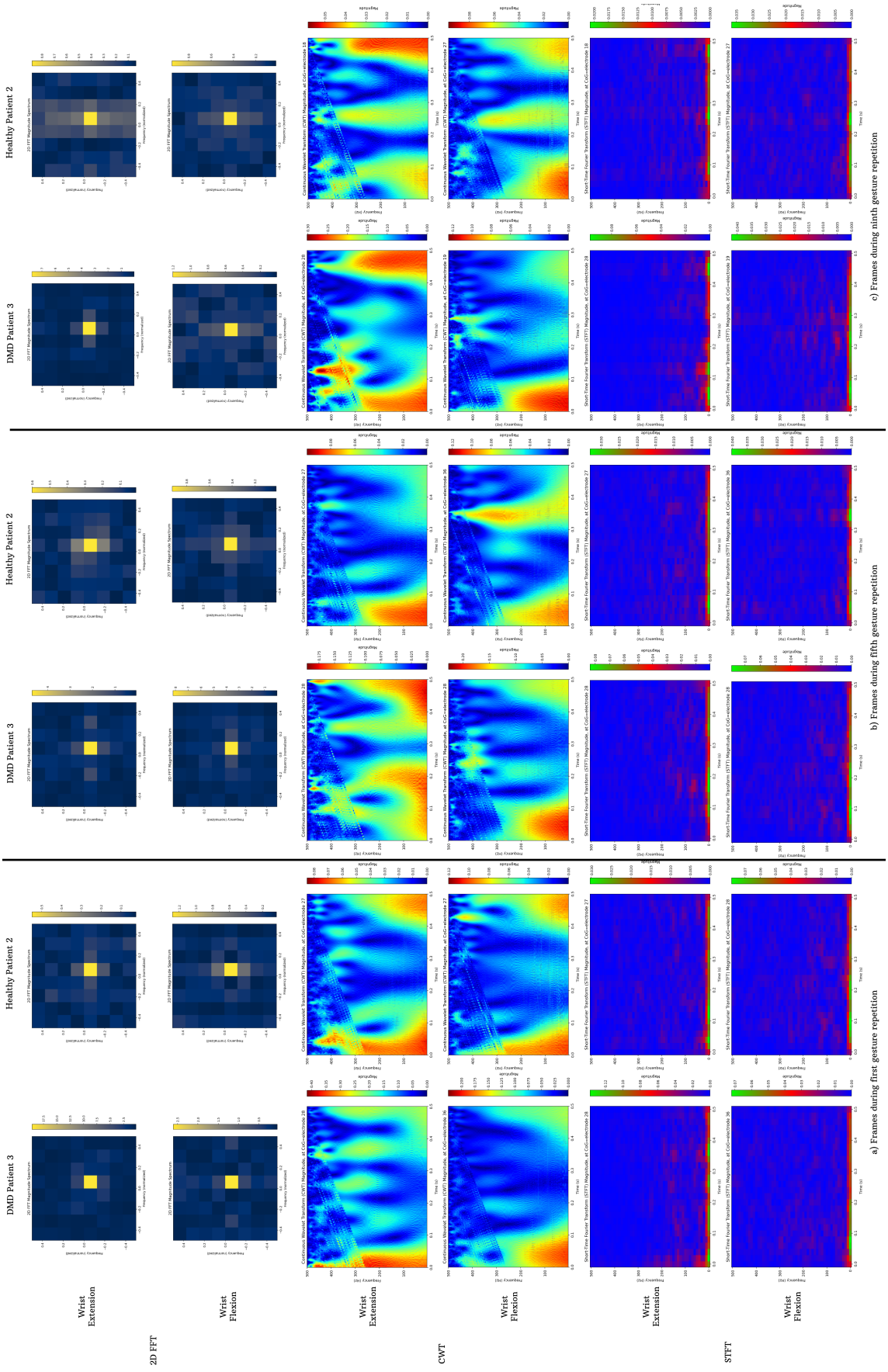


Figure 5.20: Time-Frequency Analysis through time, of 2D FFT (from original 2048-Hz sampling rate signal's 2D image) and of CWT and STFT spectra from a 500-sample window (1 second of the downsampled signal from channel nearest to CoG), at three different repetitions.

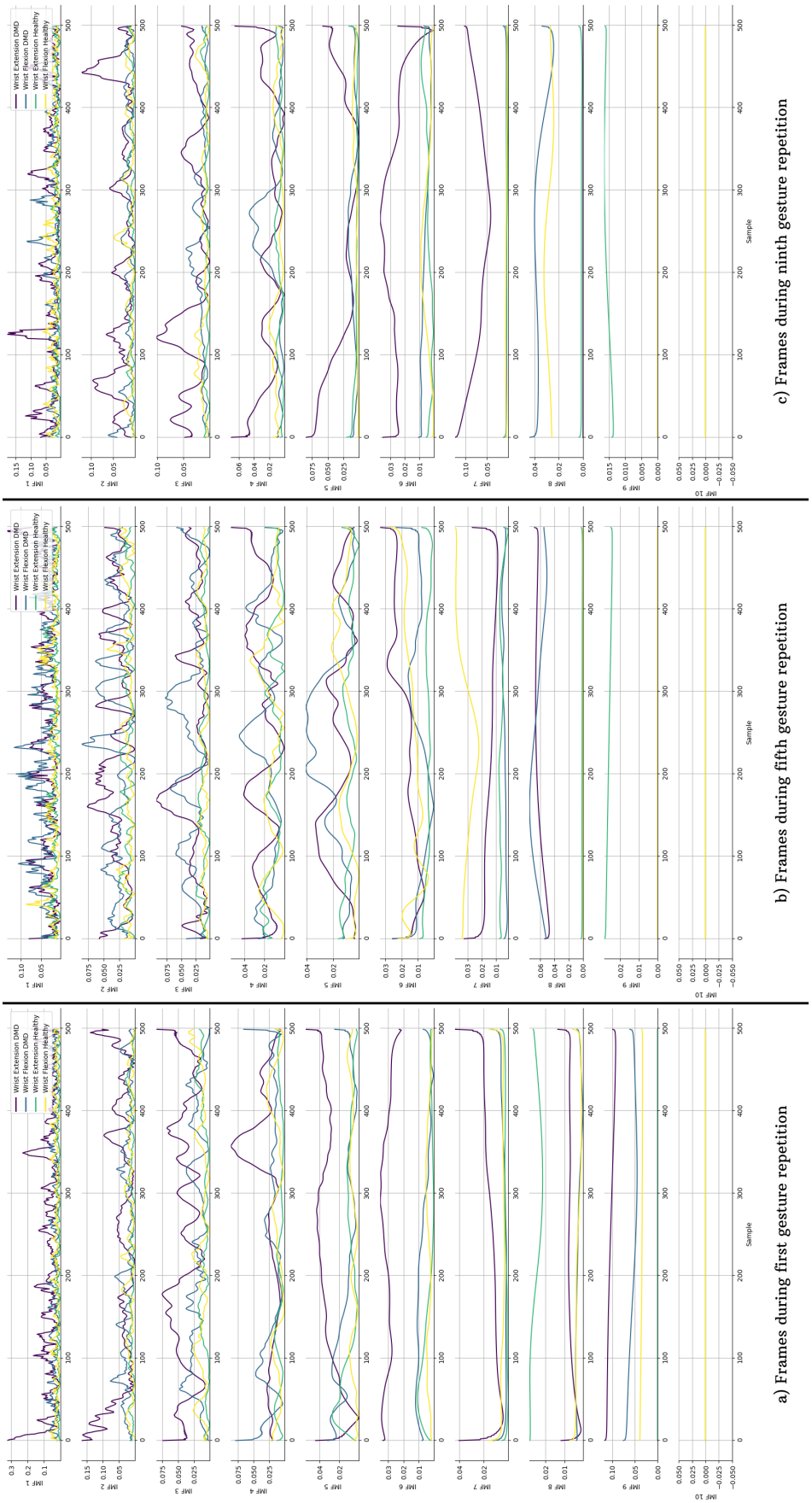


Figure 5.21: Hilbert-Huang Transform's Intrinsic Mode Functions (10 IMFs) through time, from a 500-sample window (1 second of the downsampled signal from channel nearest to CoG) at three different repetitions.

As can be seen, these time deviations (in relation to expected instant known to be when the individual was gesturing) barely ever went over 0.4 seconds (or about 800 samples), either at onset detection or offset detection; that said, some percent errors went above 5%, possibly due to transient moments during which the individual is still changing from rest to gesture, which would mean algorithms such as abbink and lidierth are not as robust to these moments. With that, one could say that these deviations represent response time of the actual individual performing the gesture: when changing from rest to a wrist position, a delay of upward of 150ms[240] can sometimes happen between brain intention, and actual movement.

With that, and never having surpassed 10%, one may consider all these algorithms reliable in identifying the instant of activation, with TKEO's displaying the best overall results, both in time-wise deviation, STD, and mean error.

5.5 Conduction Velocity Estimation

5.5.1 XC Algorithm

Synthetic Signals

When applying the XC algorithm to synthetic signals of different configurations, results are as shown in Figure 5.22. What was seen was that, generally speaking, given a single CV, the algorithm never failed to determine the CV, to errors no higher than 0.2%, regardless of actual settings or complexity (number of impulses, shape of signal, firing rate; provided CV remains the same; except for decay rates and SNR, which intentionally can be analogous to signal quality – low quality will generally always lead to incorrect estimations). This changes when two impulses of different CVs are provided: as the Figure shows, the algorithm is unable to "focus" solely on one impulse, leading to errors sometimes surpassing 100%.

What ought to be mentioned, however, is that, even in the case of simpler signals, one must ensure that IED and sampling frequency are such that, for velocities between 1 and 10 m/s (the common interval of real CVs), the distinction between two channels can be made within the given time window. In other words, and as an example, the algorithm is unable to assess CV from two signals that are too close together and given a low sampling rate (that may mean that a firing may essentially be "skipped" between channels, for example). The same can be said for window size.

Because signals tend to be more complex, resulting from several firings of differing CVs, it is to be expected that the algorithm's reliability may have a harder time identifying CVs in a real life context, though it may still be able to estimate a general velocity across the muscle fiber. Despite that, and despite the two-impulse-of-different-CVs estimation inability, for more complex signals of differing velocities, the algorithm does show the tendency to focus in on one of the propagations, sometimes switching between them (assuming these behaviours remain for real signals, one may average results through time to get an estimate of the average CV across all CVs; as such, this algorithm may prove to be an adequate way of obtaining MFCV).

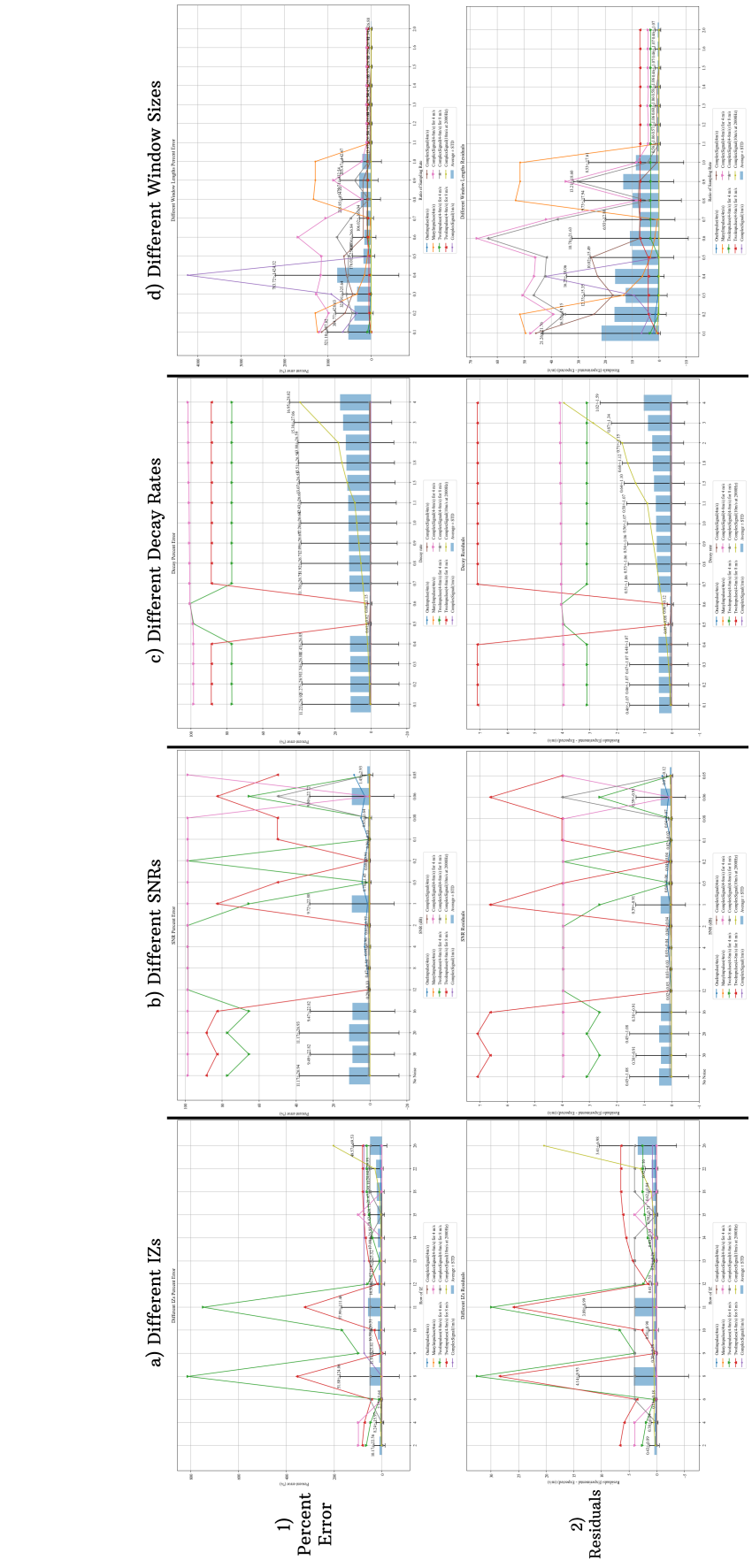
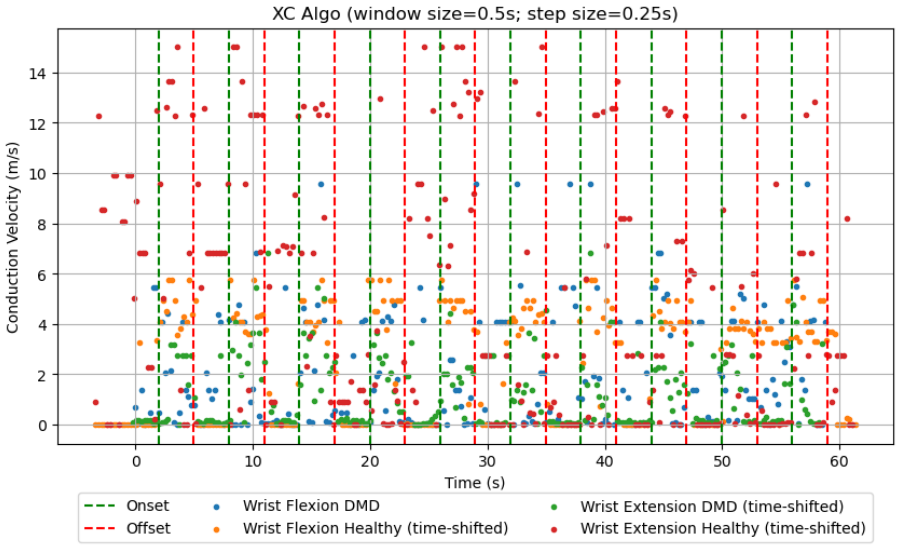


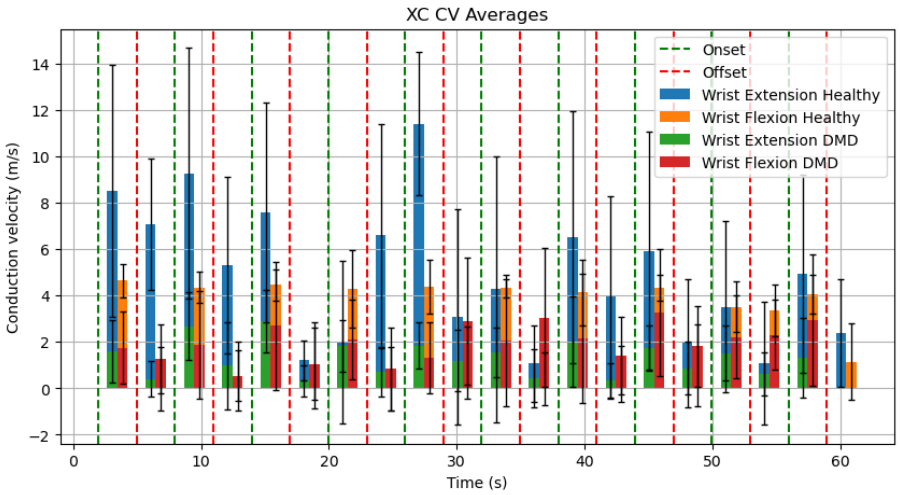
Figure 5.22: XC algorithm results on different configurations of the simulated HDsEMG signals (IZ locations shown for an array with 26 rows). To note that "ComplexSignal(4-8m/s)", for instance, corresponds to the results from complex signals with CVs of 4 and 8 m/s, and "for 4 m/s" corresponds to the results in relation to 4m/s.

DMD Dataset

Though the DMD dataset provides no ground-truths, it can still be a valuable asset for assessing if the algorithm is estimating values within the often expected CVs, roughly from 1m/s to 10m/s, and if they happen at expected intervals. Figure 5.23a shows the results acquired across the entire timelines of the two subjects performing each of the two gestures. These results come from the averages of CVs from combinations of different columns and different rows (as, in this case, columns go along the fiber's direction, computations are done within a single row, given two rows whose distance between each other is known).



(a) XC CV computed through time given window and step sizes.



(b) XC CV averages at every interval (at rest and while gesturing).

Figure 5.23: XC algorithm results on DMD dataset

To summarize these results, Figure 5.23b shows that velocities are lower outside of the periods of gesturing, as would be expected (reaching so low as a quarter of its mean during flexion by

healthy individual, as shown in Table 5.4) – even so, they sometimes are not zero, which might suggest that either some contraction is still taking place, or part of the signal noise is being considered for the estimation. Furthermore, velocities – mostly within 1m/s and 10m/s – appear generally higher for healthy individuals than DMD, regardless of gesture (though flexion gesture gives lower CVs than extension), by about twice the average; this is compatible with the idea that, due DMD being characterized by the absence of dystrophin, muscle fibers become damaged more easily, leading to progressive muscle degeneration, fibrosis, and infiltration by fat tissue. Such disruption is expected to affect efficiency of electrical signal propagation within-fiber. Not only that, but, as muscle fibers atrophy and lose functionality, the overall ability of the muscle to contract properly is diminished, contributing to the decreased conduction velocity.

Table 5.4: Overall averages of CV results of Extension and Flexion for Healthy and DMD conditions, during performing and at rest, and ratio between the two moments, for XC-based algorithm.

Condition	State	Mean $\pm$ SD (m/s)	Ratio
Extension Healthy	Performing	6.4 ± 4.6	1.9
	At rest	3.4 ± 3.1	
Flexion Healthy	Performing	4.3 ± 0.9	4.2
	At rest	1.0 ± 1.4	
Extension DMD	Performing	1.9 ± 1.3	2.9
	At rest	0.7 ± 1.1	
Flexion DMD	Performing	2.2 ± 2.3	1.3
	At rest	1.7 ± 1.9	
Ratio (Extension/Flexion)	Performing	1.3	
	At rest	1.5	

As would be expected from an estimation that seeks to obtain an average across several firings with different CVs, the results come to show that the STDs are quite high, even if the average itself aligns with what one may expect. While this shows that this algorithm may falter at being a precise method of MFCV estimation, it can still be a somewhat good approach at obtaining a representative average.

5.5.2 STA Algorithm

Synthetic Signals

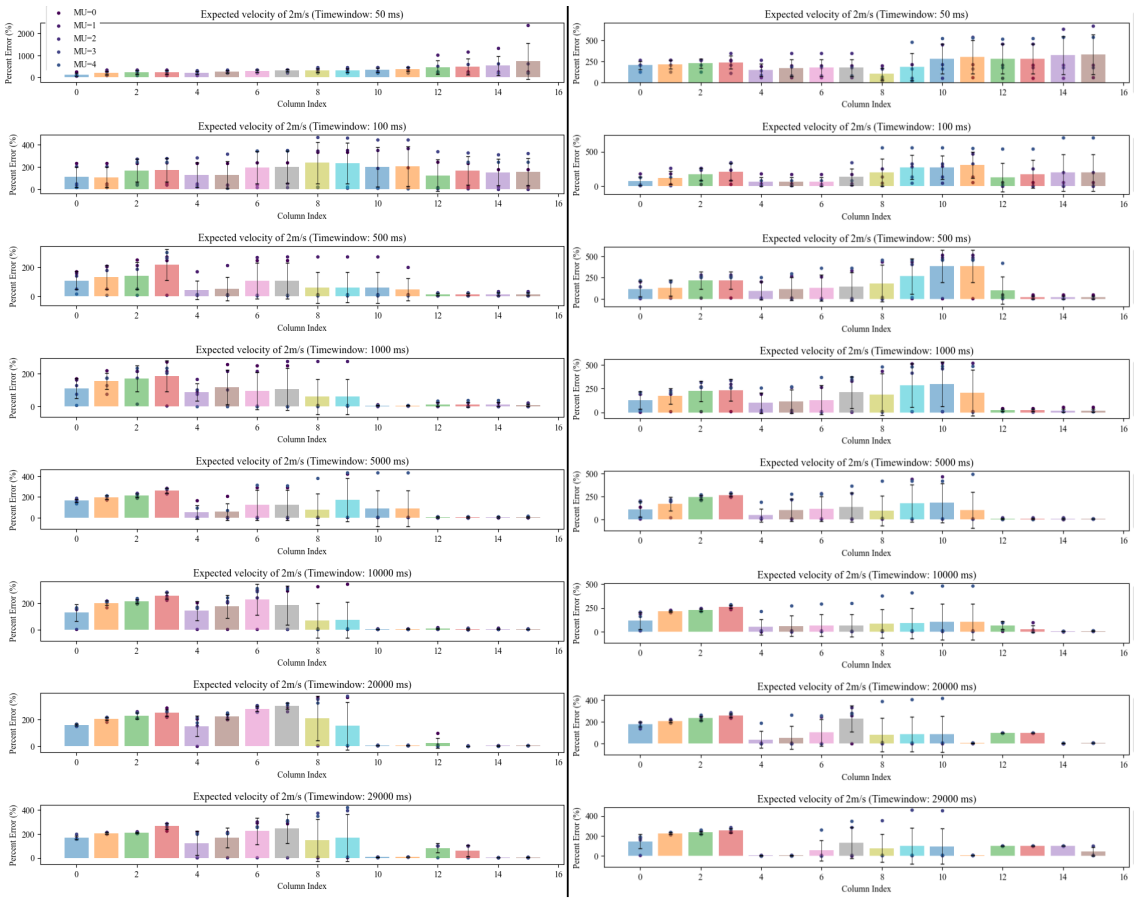
When evaluating the STA algorithm with synthetic signals, it was required that the MUSTs be known – so as to remove the variable of MU decomposition from the assessment –; with that, from a 4 by 16 HDsEMG grid, wherein APs propagate along rows (of which there are 4), the generated signals, alongside MU firing instants (same ones used for 5-MU, lower FR and higher FR of Section 5.7, and further described therein), led to the results found in Figure 5.24.

What they show is that, for each MU, and averaging all CV estimations across all columns, on all timewindows, results for CVs of 4m/s and of 10m/s very rarely went over 5% mean percent error, for all motor units whose MUCV was estimated; in the case of MUCVs between 1m/s and

Figure 5.24: Bar plots of the percent error results of STA-based algorithm upon different timewindows for estimation, and different simulated signals of MUSTs with different CVs. Each bar corresponds to the average on each MU of all CVs computed per column of the grid.

10m/S, all estimated CVs averaged to about 5m/s, which is about the midway point of the given CV interval, though it may indicate that the algorithms ability to distinguish between differing MUCVs of a specific MU may also suffer from cross-correlation related issues, as the ones presented in the XC-approach. Even still, one could still accept this algorithm as a reliable way of estimating an overall CV as produced by a given MU, within a time window; additionally, because the average is done across several columns, it is not impossible that each column's result may be nearer to other CVs of a given MU. To note that, for this analysis, different timewindows are shown to not affect percent error greatly (with 29000 ms a 29-second window, since the simulated signals had an activation lasting 30 seconds), provided that the MUCV remains constant.

For velocities of 2m/s and under, the algorithm started showing an inability to correctly predict each MUCV, barely every going under errors of 30% when averaging it across columns per MU; when averaging it across MUs per column, as seen in Figure 5.25, some results do reach as low as 3% (as in column 14 of timewindow 29000ms), however, the errors mostly reside around



a) Simulated Signals of Lower Complexity

b) Simulated Signals of Higher Complexity

Figure 5.25: Bar plots of the percent error results of STA-based algorithm, on 2m/s-simulated signals of (a) a lower and (b) a higher complexity. Each bar corresponds to the average of the MUs CV on each column of the grid.

100%; because 2m/s is a lower velocity, this also means that error computation suffers from higher

variability (100% implied the STA algorithm is determining 2m/s to be 4m/s (only a 2m/s difference)). In evaluating how different complexities of the signals (higher complexity here referring to aspects such as higher firing rate, but still same number of MUs (5)) may influence STA's ability to determine MUCV, it was found that, while higher complexity does lend itself more averages with errors lower than 10%, overall some errors were still high, if not higher; furthermore, the innervation zones seemed to affect results: in Figure 5.25, both (a) and (b) were from signals innervated at column 15 – it is also around those columns that percent errors are improved: this may show that, in dealing with situations where CV is low, one ought to perhaps choose which columns to compute MUCV from, by being aware of the innervation zones.

DMD Dataset

Once MU decomposed (leading to 16 identified MUs), with decomposition results that can be found in Subsection 5.7, the DMD dataset was used to estimate CV through the STA method, leading to values present in Figure 5.26. Because this decomposition led to a spike trains with essentially no firings outside of activation period, this method indicates that no MUCV is detected during rest – which makes sense considering no muscle activation. As for the velocities estimated within each 3-second gesture, one for each MU, what can be seen is that CVs appear to decrease through time, which may be an indication of fatigue and lactic acid concentration, from which the CV often decreases.

When comparing values between subjects, averaging shows, as in Figure 5.27, that CV for the healthy patient was higher than for DMD patient, by a ratio of 1.4 for extension and of 1.2 for flexion, with all velocities somewhat close to 1.5 m/s. Even still, STD also shows that, especially for healthy individuals, velocities may reach as high as 6m/s; additionally, healthy's STD is higher than DMD's in extension; lower in flexion than DMD's, though it remained about the same, whereas DMD's greatly changed, perhaps showing that performing these two tasks, as done by a healthy individual, is something that can be done with more ease that requires a "standard" equal in both. Flexion for the DMD subject, however, did show an increase both in CV average and STD, which may in turn reveal that more effort had to be put into one of the tasks.

5.5.3 Comparison between Algorithms

Comparing both algorithms, it becomes apparent that, while MFCV led to higher CVs overall, the CV relations found between subjects and between tasks remained the same; further, as the Bland-Altman plot of Figure 5.28a shows, no apparent trend is observed (save for the linear trend that seems to describe wrist extension pf healthy individual) and most values are somewhat close to their mean difference, with a bias that did not go below -1.5 (save for the -3.7 bias of extension by healthy individual). Generally the level of agreement is higher for a lower range, with greater variability or scattering at the higher range, which may suggest that the measurement error or differences between the two methods being compared increase with higher values of CV, perhaps

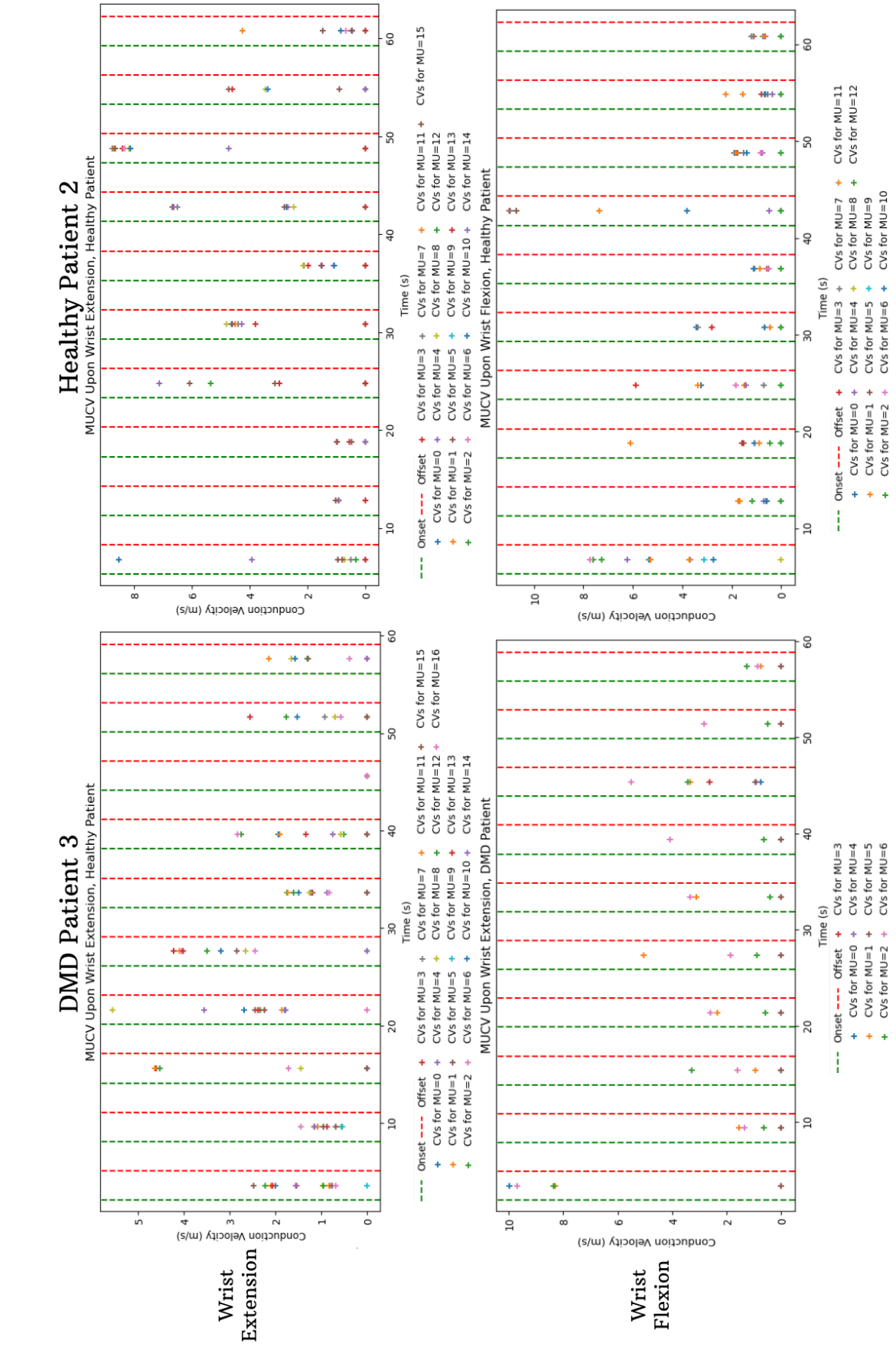


Figure 5.26: STA-based CV estimation results on DMD dataset.

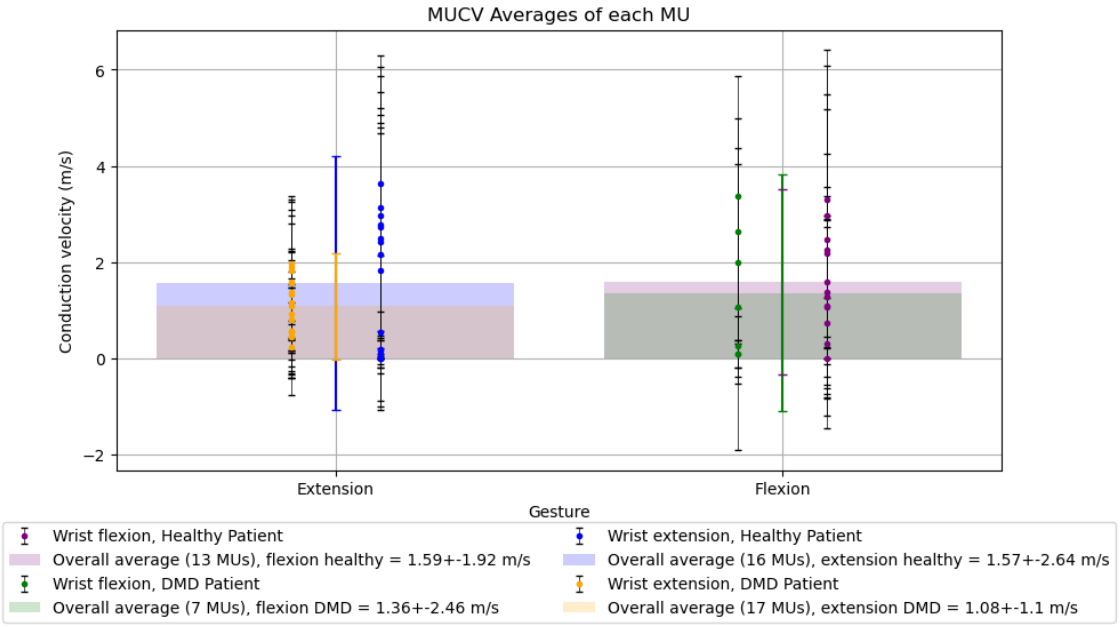
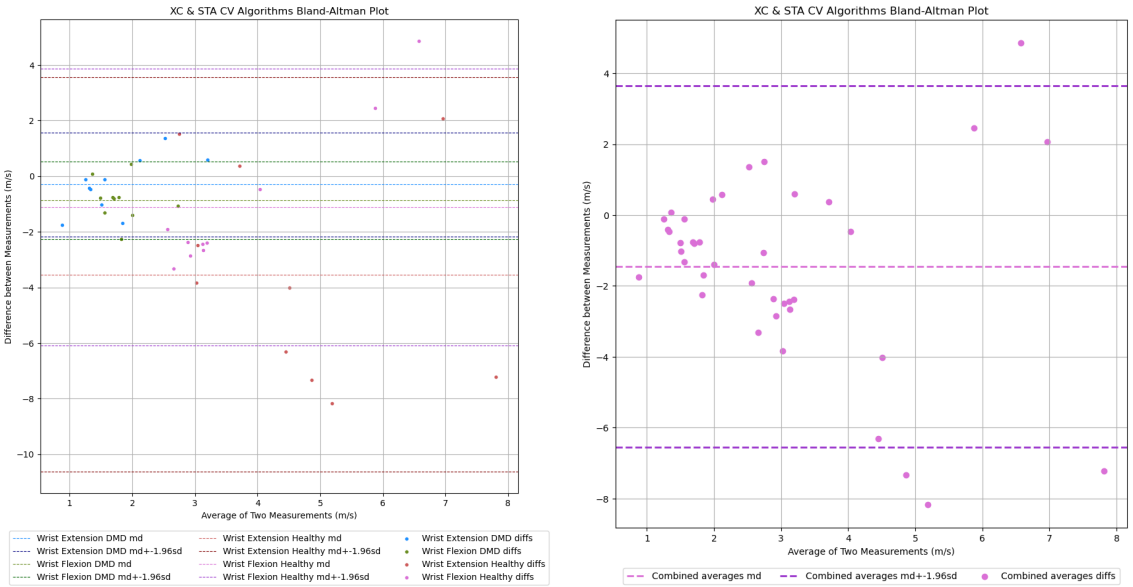


Figure 5.27: Bar plot of the averages of the STA CVs, on DMD dataset; error bars corresponding to STD of each average.

indicating some heteroscedasticity that shows that, if one is to assume an algorithm is robust, so too will the other one, at least for observations conducive to lower CVs.



(a) Separating each case (subject, gesture) of each method

(b) Combining all cases (all observations) of each method

Figure 5.28: Bland-Altman plots from the comparison of CV results; resulting in mean difference (md) and standard deviation of the difference (sd)

In evaluating the XC algorithm, in Figure 5.29 one can see the duration it takes for it to

perform its computation, barely ever reaching 1 second up until the number of samples surpasses 10^4, Because CV is often computed for a short interval of time, the over-1-second computation time is one that rarely would occur in a research context.

Similarly, for the STA algorithm, Figure 5.30 shows that, even under perhaps the most complex of real-life situations (65 channels, 2048 Hz, 30 seconds of timewindow), the algorithm would barely reach 0.8 seconds, proving computationally efficient in calculating many MUCV, which are often estimated in much simpler signal conditions.

With that, it is shown that these algorithms are reliable tools for estimating CV; differences found between one another may reveal that some inaccuracies may be present in at least one of the algorithms; however, because one method proposed to estimate MFCV and the other MUCV, it is not inordinate to conclude that this is what caused these differences, especially given the fact that these were in value only, not in general "behavior" (ratios, variability, change in CV amplitude through time) between gestures and subjects.

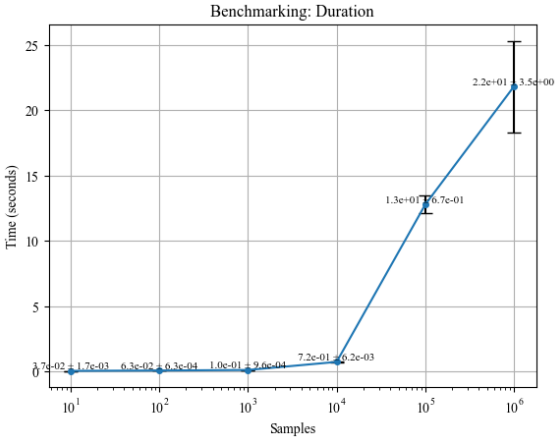


Figure 5.29: Duration of XC algorithm on different sample sizes.

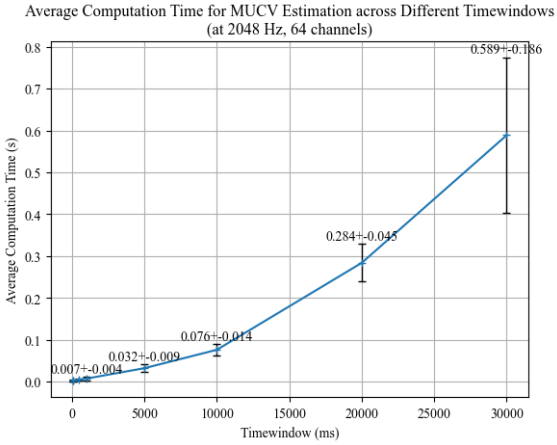


Figure 5.30: Duration of STA-based algorithm on different time windows, at 2048Hz.

5.6 Innervation Zone Estimation

When estimating IZ on simpler signals (single AP, for instance), results are as shown in Figure 5.31; as expected, the steepest extremum in each case is found at the index of the row of innervation; this, however, mostly only happens in cases where IZ is found farther from the edges of the grid. This indicates that, though this algorithm may be mostly accurate, it loses some of its robustness for specific IZ locations upon the grid.

Not only that, but its accuracy is limited to signals that don't display many IZs, or higher firing rates – a grid with complex signals such as the aforementioned ones do lead to PCA coefficients that show no systematic behavior, nor clear correlation in regards to IZ zone, as Figure 5.32 reveals.

Even so, some specific IZ zones within a grid, as well as some specific grid sizes (number of rows), even for more complex signals, can often times lead to estimation that are near the expected IZ. As an example, from Figure 5.33 one can see that, as IZ is closer to the center of a given grid,

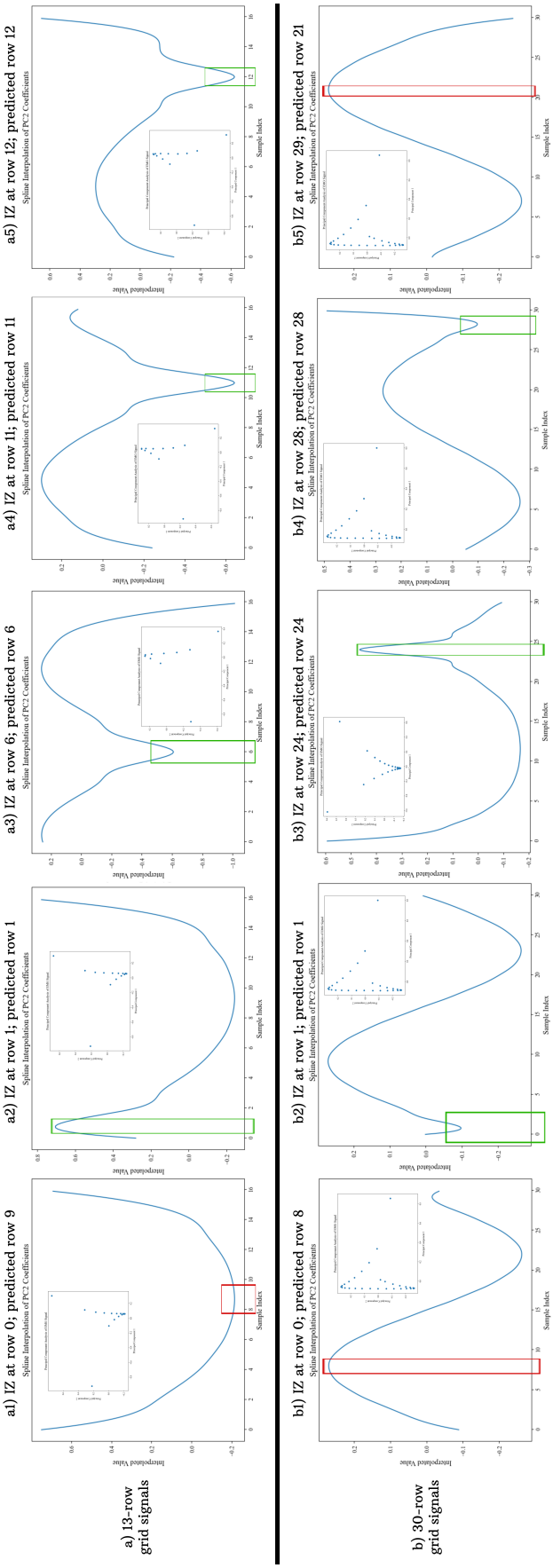
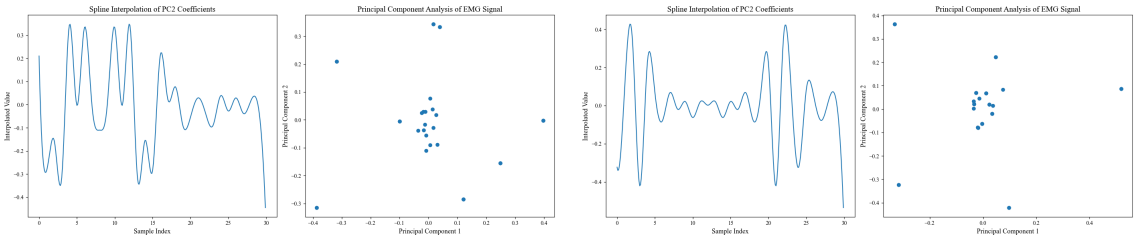


Figure 5.31: Interpolated plot of the 2nd principal component coefficients on simulated signals with IZ at different rows, on simpler signals. Red squares indicative that the steepest extremum incorrectly identified the IZ; green squares indicative that it correctly identified it. Within each image, a plot of the relation between the second principal component and the first.



(a) Result on a signal with IZ at row 8, for a 30-row grid

(b) Result on a signal with IZ at row 12, for a 30-row grid

Figure 5.32: Plots of the 2nd principal component coefficient on more complex signals.

the ability for this PCA algorithm to estimate IZ increases – on simpler signals reaching a full-accuracy peak at around 20% closeness; on complex signals, reaching sometimes as much as 80%, with higher variability, though farther distances perform equally inaccurately, even if IZ is not near the edge.

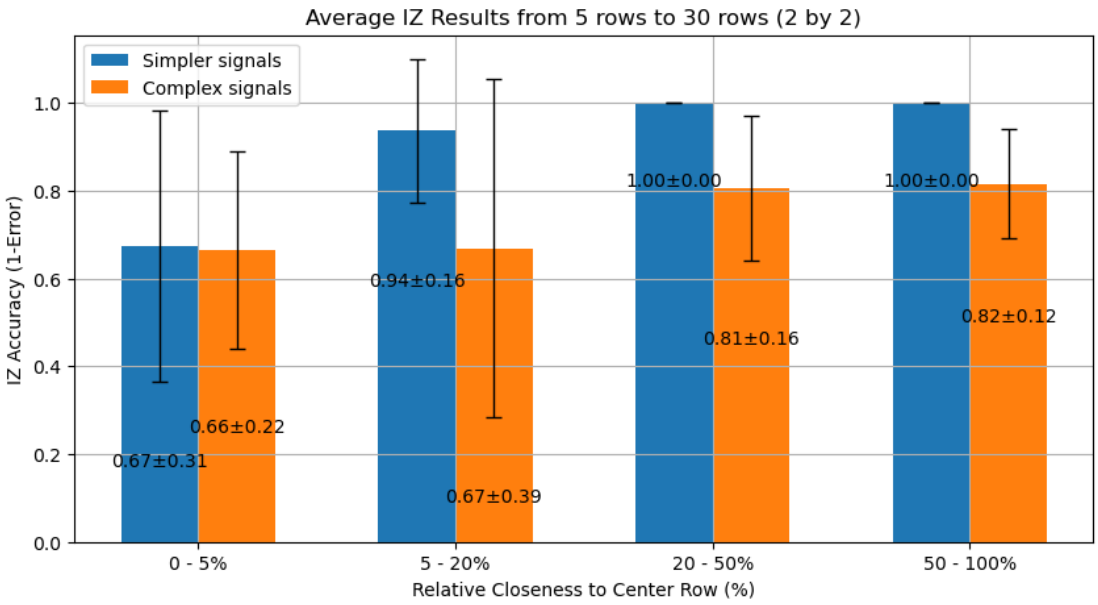


Figure 5.33: Average IZ accuracy per closeness of IZ to center row; 0% indicates that the channel is at a corner row (farthest from center); 100% corresponds to center row.

With that, for simpler signals, it can be assessed that this algorithm proves to be accurate on most situations, provided that the grid is ensured to be displaced such that its center is closer to the IZ.

For more complex signals, an adequate approach ought to be first decomposing the complex signals (provided that the used algorithm is reliable) and, from the simpler MUAP shapes extracted therefrom, estimate IZ for columns of a given MU.

5.7 Motor Unit Decomposition

Upon running the MU decomposition algorithm, times may vary depending on factors such as number of iterations, thresholding parameters, number of samples. Figure 5.34 it is seen that decomposition time appears to increase linearly with sample number, and that, for conditions where 64 channels are being used, these may reach as high as 400 seconds – however, this would only happen upon decomposing signals that occupy a 40000-sample window; even for high sampling rate situations, one would not expect a time-window of upward of 19 seconds to be decomposed; as such, average times may lie around 25 to 60 seconds.

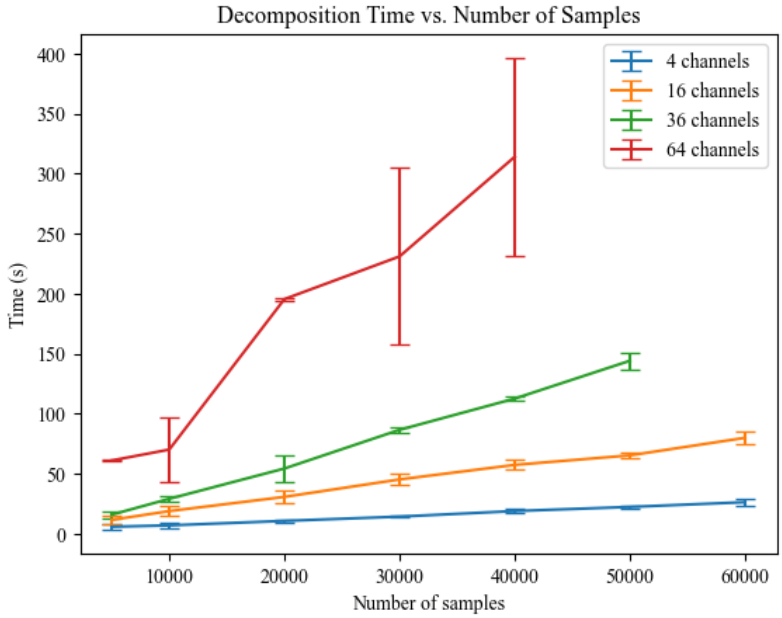


Figure 5.34: MU Decomposition algorithm duration at different number of samples and four distinct grid sizes.

Synthetic Signals

Upon generating the synthetic signals, further described in the previous chapter, comparisons of results were made as can be shown in Figure 5.35.

When it came to MUSTs from decomposition – for differing tolerances and MU numbers leading to Table 5.5, with some overlapped results present in Figure 5.36 –, the algorithm demonstrated, to a tolerance of 50 ms, a performance achieving a rate of agreement (RoA) above 50%, with an average of only 36% unmatched firings. For tolerance values greater than 50 ms, the RoA increased significantly, reaching up to 0.91 ± 0.11, while the number of unmatched firings dropped to an average of 9%. This shows a clear improvement in firing detection accuracy as the tolerance is relaxed.

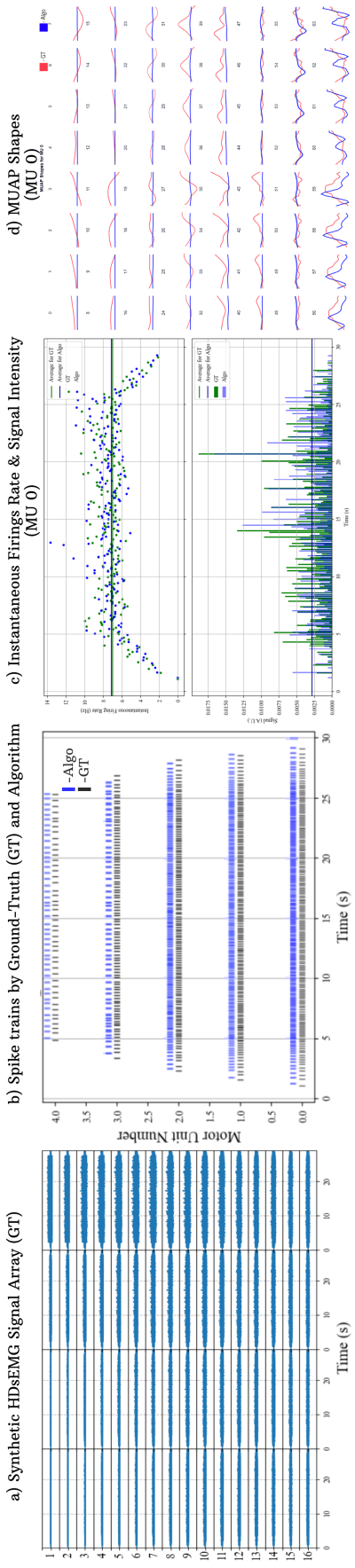


Figure 5.35: Comparison of synthetic signals (GT) with its corresponding decomposition (algo), for 5 MU-generated signals of lower firing rates.

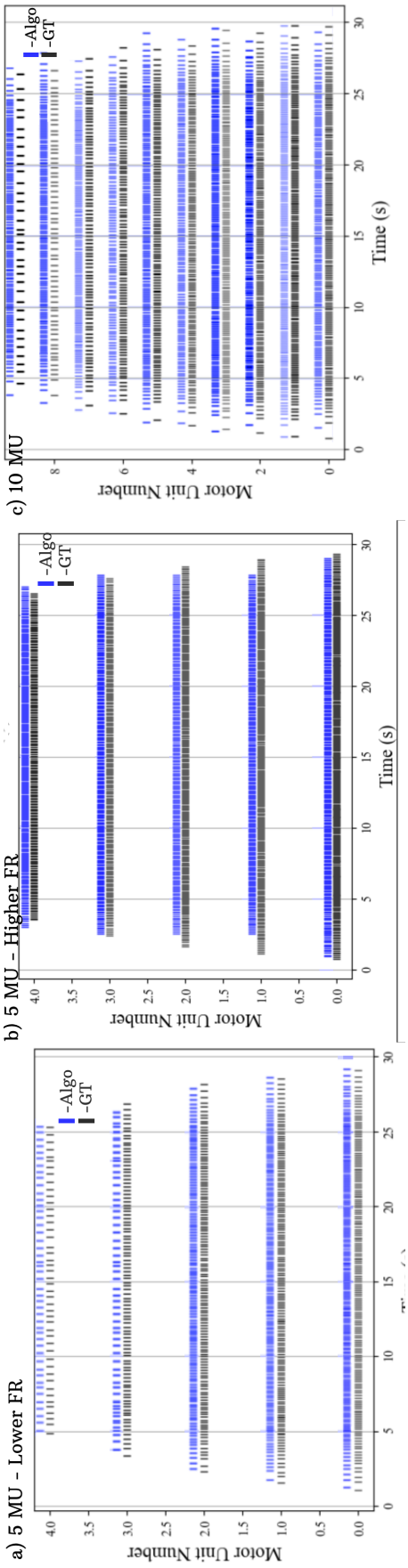


Figure 5.36: Comparisons of spike trains between Algo and GT, for three different conditions.

Furthermore, the algorithm was able to identify firings with temporal errors between the ground truth and the algorithm, rarely exceeding approximately 150 ms. By adjusting the algorithm's parameters, the number of detected MUs matched the ground truth precisely. In cases where more than 10 MUs were present, the algorithm identified some clearly repeated MUs (usually excluded by the researcher themself).

In assessing impact of firing rate on RoA, it was seen that for higher FR, the RoA increased (average difference of 20%). Additionally, for higher tolerances, percentage of firings without matches as determined by Algo start to become more and more similar ($3\% \pm 2\%$ for lower FR; $6\% \pm 8\%$ for higher FR) – this information, summarized in Table 5.6.

In regards to how MU firing rate may affect overall average of instantaneous firing rates and MUAP shapes, it is seen that, for signals of lower and higher firing rates, the average instantaneous firing rates mostly coincided (differences no higher than 0.2 Hz for lower FR – about 3 to 8 Hz; no higher than 2 Hz for higher FR – about 15 Hz) between the Algo result and the GT; similar can be said for signal strength. Plus, in each simulation, over 50% of motor units initiated and ceased firings within similar intervals (roughly 1 second).

Additionally, the MUAP shapes from the algorithm's results did have similar envelopes to the expected – over 60% being considered as nearly-identical –, though "spikiness" differed, being less common in Algo than GT; additionally, some shapes were less correctly identified when looking at channels that are further from the innervation zone; it was also found that often times the Algo's MUAP shapes were similar to GT but with a channel offset – for instance, Figure 5.37a shows that the row of channels 48-55 appear to display algorithm shapes that are more aligned with those found in row of channels 40-47 of GT.

DMD Dataset

Upon running decomposition on the two subjects, two gestures, across entire timeline, results, displayed Figure 5.38a, showed that number of motor units detected was higher for DMD compared to healthy upon wrist extension, yet lower for wrist flexion (in general, for both cases, MU number was higher in wrist extension, perhaps indicating the necessity of higher effort for this action); further, when looking at initial gesture performances, the motor unit spikes appear to be of lower density on first repetition followed by higher density on second repetition in DMD patient for extension only – compatible with the idea that a higher recruitment threshold is needed to initiate motion and that, after that, recruitment of more motor units, and a higher firing rate, are needed to achieve the same level of muscle force or activity –, though this one later subsides and, similar to flexion, starts firing less – this reduction in firing may reflect on DMD subject's inability to continue performing gestures fully, perhaps even indicating fatigue onset. On the other hand, healthy patient's firings remain consistent throughout all gesture repetitions.

When trying to identify common motor units, finding the highest RoA allowed for comparing instantaneous FR and signal intensity. What Figure 5.38 shows is that the firing rate of the healthy individual is often much lower than that of DMD (even reaching averages that are 3 times lower,

as in 5.381.1). Similarly, signal intensity is also much lower, once again corroborating the idea the much more effort need be done by DMD for performing the same tasks.

Overall, this algorithm, beyond efficient, proved to be reliable, from results on synthetic signals that reached RoAs as high as 90%, to decomposing real signals leading to assessments compatible to those expected when dealing with DMD patients.

Table 5.5: Summarized rate of agreement and matching firings results of the algorithm, averaged across the 1-MU, 5-MU (high FR and low FR), 10-MU and 20-MU simulated signals, over 5 distinct tolerances.

Tolerance (ms)	Mean Rate of Agreement ± STD	% Error	Mean Time Errors (ms) ± STD (ms)	% Error	Mean # Firings without Matches ± STD	% Error	Normalized Mean # Firings without Matches ± STD	% Error	Mean Algo # Firings without Matches ± STD	% Error	Normalized Mean Algo # Firings without Matches ± STD	% Error
5	0.07 ± 0.04	52	5 ± 1	30	167 ± 73	44	0.93 ± 0.04	4	165 ± 59	36	0.93 ± 0.04	5
10	0.14 ± 0.06	43	10 ± 2	20	152 ± 62	40	0.86 ± 0.06	7	150 ± 48	32	0.85 ± 0.07	9
50	0.6 ± 0.2	28	49 ± 7	14	54 ± 19	34	0.36 ± 0.18	51	51 ± 33	64	0.35 ± 0.23	68
100	0.85 ± 0.15	18	71 ± 21	29	25 ± 20	81	0.15 ± 0.15	100	23 ± 30	134	0.15 ± 0.19	129
1000	0.91 ± 0.11	12	94 ± 47	50	18 ± 22	124	0.09 ± 0.11	125	15 ± 32	211	0.08 ± 0.17	209

Table 5.6: Rate of agreement and matching firings results of the algorithm, for two sets of 5-MU simulated signals; one of higher FR, the other of lower FR.

	Mean Rate of Agreement ± STD				Mean Time Errors (ms) ± STD (ms)				Mean Algo # Firings without Matches ± STD				Normalized Mean Algo # Firings without Matches ± STD			
Tolerance (ms)	**Higher FR**	**% Error**	**Lower FR**	**% Error**	**Higher FR**	**% Error**	**Lower FR**	**% Error**	**Higher FR**	**% Error**	**Lower FR**	**% Error**	**Higher FR**	**% Error**	**Lower FR**	**% Error**
5	0.14 ± 0.02	12	0.03 ± 0.02	70	4.7 ± 0.4	8	5 ± 2	36	282 ± 22	8	110 ± 54	49	0.86 ± 0.03	3	0.96 ± 0.02	2
10	0.26 ± 0.02	6	0.09 ± 0.06	67	9.8 ± 0.4	4	10 ± 4	39	238 ± 23	10	102 ± 48	47	0.71 ± 0.05	7	0.91 ± 0.05	6
50	0.89 ± 0.07	8	0.4 ± 0.2	53	36 ± 3	7	55 ± 8	15	23 ± 24	104	50 ± 14	28	0.08 ± 0.08	95	0.5 ± 0.2	38
100	0.90 ± 0.07	8	0.7 ± 0.3	42	37 ± 2	5	91 ± 20	22	21 ± 25	117	18 ± 8	43	0.08 ± 0.08	104	0.3 ± 0.2	96
1000	0.91 ± 0.8	8	0.9 ± 0.2	23	44 ± 6	13	157 ± 73	47	15 ± 26	175	3 ± 2	62	0.06 ± 0.08	140	0.03 ± 0.02	87

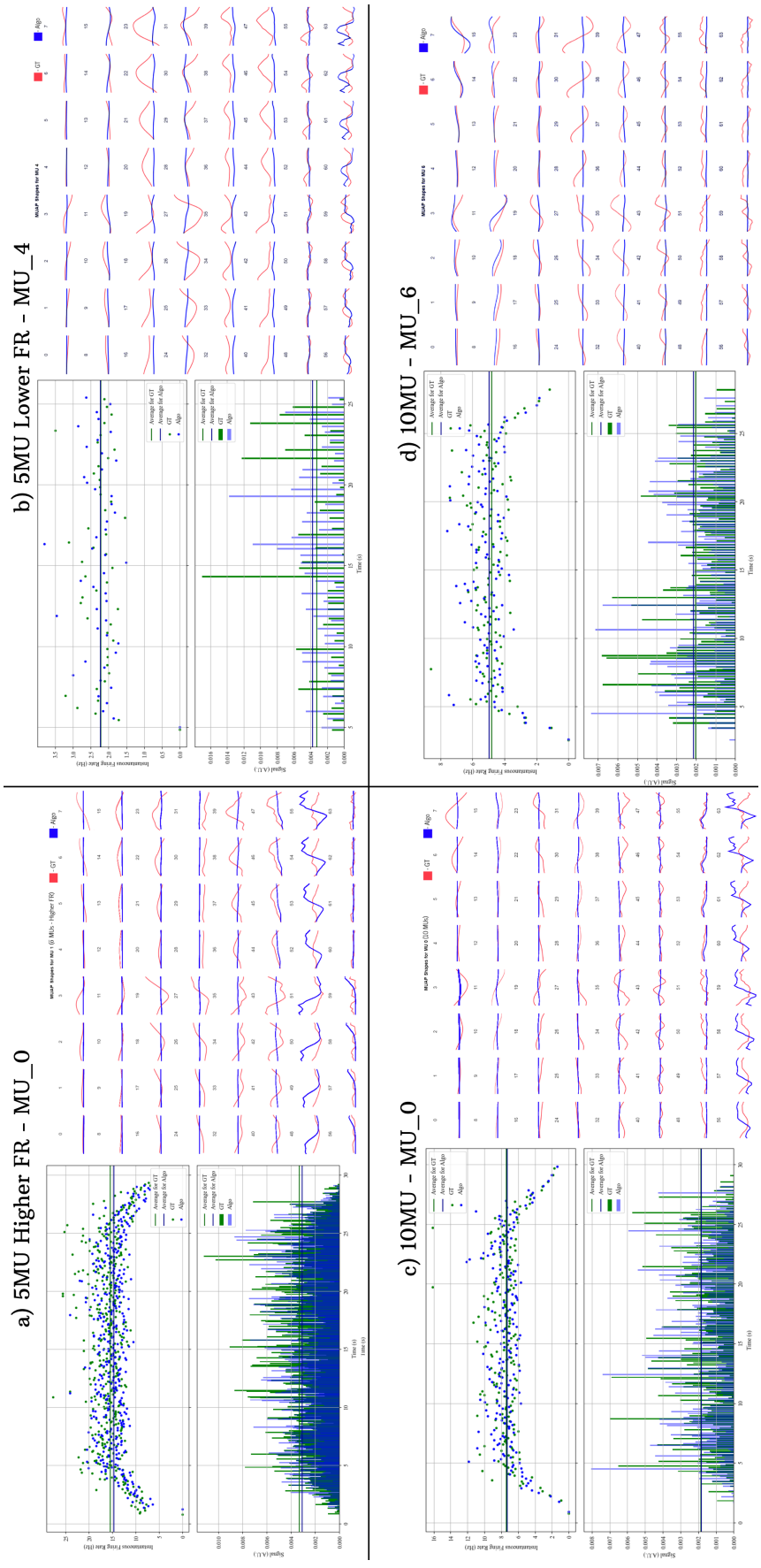
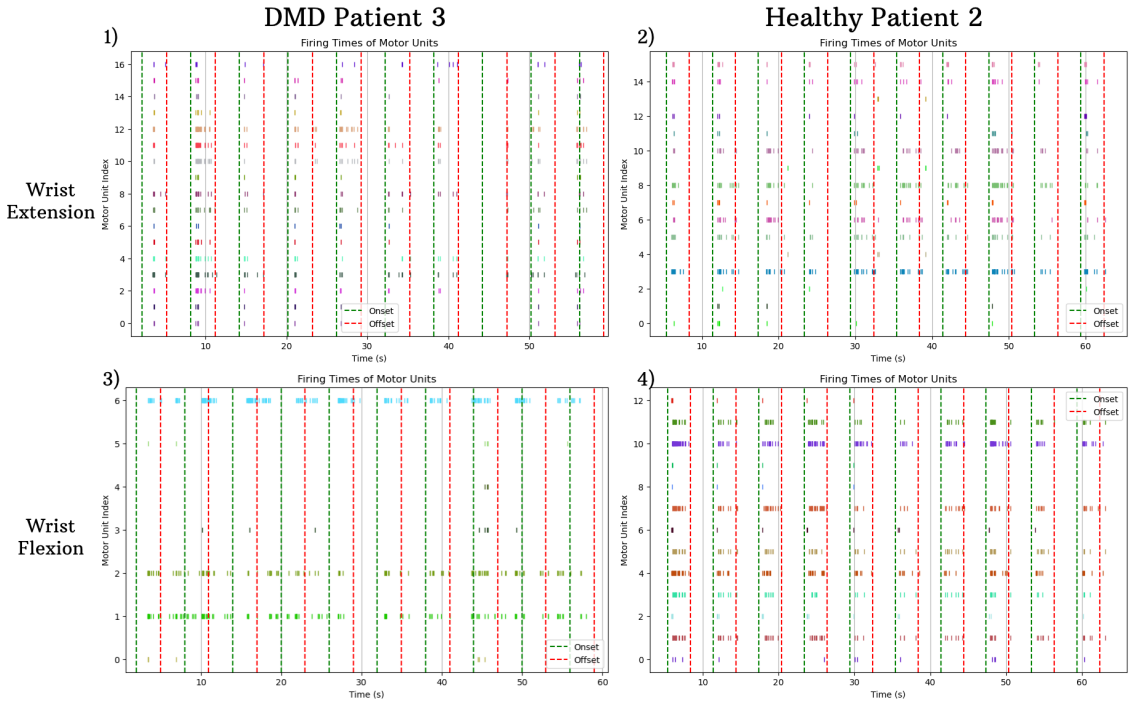
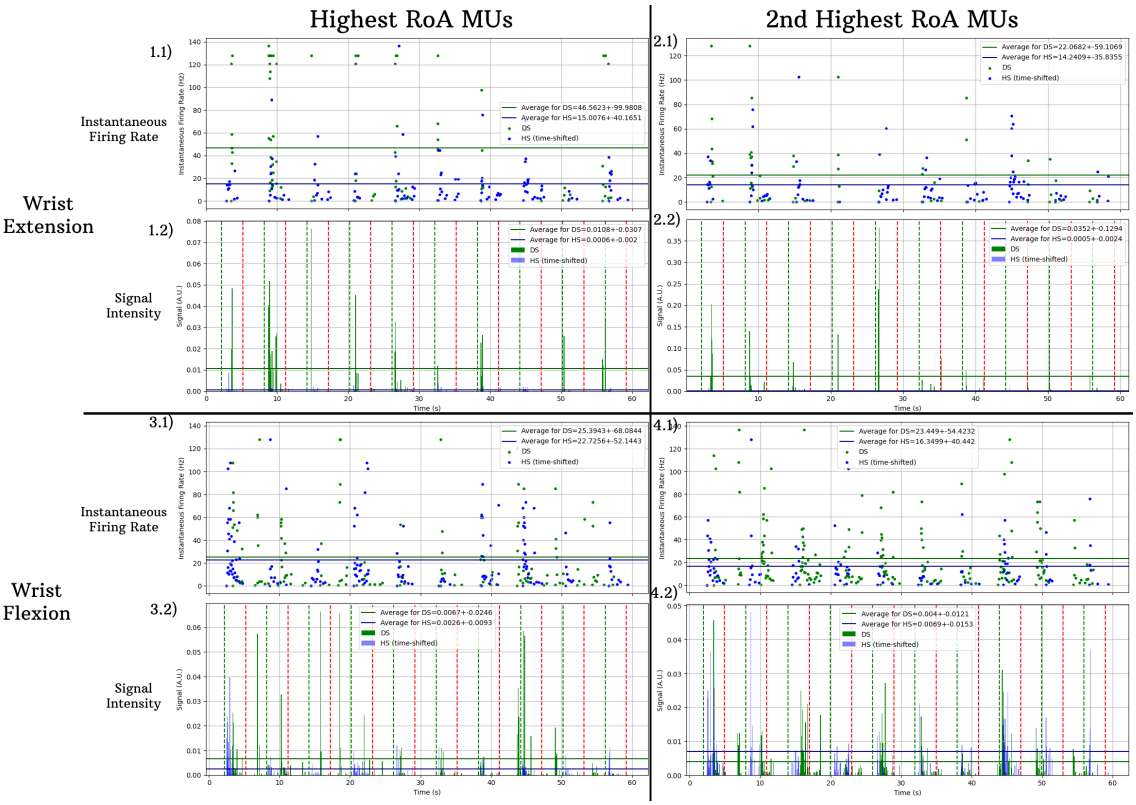


Figure 5.37: Comparisons of firing rate, signal strengths and MUAP shapes on three different conditions (two from the same 10-MU signal, displaying two specific MUs).



(a) Spike trains of all four cases



(b) Instantaneous firing rates and signal intensities found at MUs with highest RoAs for same gesture, different subject

Figure 5.38: Results of the MU decomposition algorithm on DMD dataset.

ML Applications

5.8 Gait Monitoring

At every step, it is relevant to evaluate the impact of each new decision on the accuracy of the model being trained and identify possible measures to take to either increase it or, at least, mitigate processing time.

Furthermore, most evaluations were done using the Random Forest classifier, thanks to the fact that it provided the highest accuracy at every point during this work. The results given by other models is addressed further in Subsection 5.8.4.

As one of the first steps was the choice of window length, ultimately choosing 3000 samples (3 seconds), it is perhaps pertinent to inspect how that can affect the predictive ability of the model. Table 5.7 shows that, while shorter intervals provided more segments from which the model could learn, the reduced windows did lower the ACC (while not affecting STD much). A drastic increase is found from 1000 samples to 2000 samples (of about 11%), and also from 2000 samples to 3000 samples (of about 5%); this shows that the information attained from bigger windows is indeed more impactful than the one gained from 2 to 3 thousand more observations; results for ”4000 samples” are slightly better than for ”3000 samples” by 0.3% (though no change in STD). Despite this, the latter was used for the study, being seen as a better balance between “high accuracy” and “time needed for classification”.

Table 5.7: Impact on model accuracy upon using different window lengths (from 1 second to 4 seconds), with step size of 300 samples, with task feature known (“+1”).

Window size	**Total Observations**	**Model Evaluation**		
		No. features	**Model**	**Highest Accuracy**
1000 samples	15997	111+1 (all)	RFC	0.817 ± 0.003
2000 samples	14279	111+1 (all)		0.921 ± 0.005
3000 samples	12846	111+1 (all)		0.969 ± 0.002
4000 samples	11628	111+1 (all)		0.972 ± 0.002

To note that, throughout the feature selection process, only validation data (*i.e.*, data from the same distribution of the train set, that resulted from the 80/20 split) is used for determining accuracy; once determined the best features, only then was the unseen data (from subjects 10 and 11) tested, in Subsection 5.8.4.

5.8.1 Feature Selection by Channel

When it came to choosing how many channels would be conducive to best results, combinations of all three were done. As Table 5.8 shows, one single channel never led to accuracies higher than 93%, and, in fact, showed worse standard deviations (with a percent uncertainty of about 1%). Adding one more channel proves to increase accuracy by about 3-4%, and with lower uncertainty; even so, some pairings lead to better results (mainly using channel 1 and 2, which only differed

from the highest accuracy by 0.6%). The highest accuracy is had when all three channels are considered, with accuracies reaching so much as 96.9%, and with much lower deviations (0.2% – less than half of previous iterations).

Table 5.8: Results of model accuracy upon using different combinations of the four channels, with task feature known ("+1").

Channel	Model Evaluation		
	No. features	Model	Highest Accuracy
1	111+1 (all)	RFC	0.92 ± 0.01
2			0.93 ± 0.01
3			0.91 ± 0.01
1, 2			0.963 ± 0.008
1, 3			0.953 ± 0.006
2, 3			0.957 ± 0.005
1, 2, 3			0.969 ± 0.002

With that, for further feature selections, all three channels are included, seeing as how that provided the best results, and do not impact overall computational performance.

5.8.2 Feature Selection by Domain

Analogous to the channel selection done in Subsection 5.8.1, combinations of domains were done, with effects presented in Table 5.9: for singular domains, both time and frequency gave ACC above 95% (with STDs no greater than 0.6%), with only the FrFT domain leading to a lesser result of 75%±2%. Even so, its inclusion with either the time domain or frequency domain did increase overall ACC without increasing uncertainty (although, still lesser than the "Time + Freq" pair, at 96.9%±0.4%). Overall, using all three domains simultaneous provided the lowest percent uncertainty, with a result of 96.9%±0.2% – same ACC as "Time + Freq", but lower STD).

Table 5.9: Results of model accuracy upon using different combinations of the three domains, with task feature known ("+1").

Domain	Model Evaluation		
	No. features	Model	Highest Accuracy
Time	57+1	RFC	0.961 ± 0.004
Frequency	36+1		0.940 ± 0.006
Fractional Fourier Transform	15+1		0.75 ± 0.02
Time + Freq.	93+1		0.969 ± 0.004
Time + FrFT	72+1		0.967 ± 0.006
Freq. + FrFT	51+1		0.944 ± 0.006
T + F + FrFT	111+1 (all)		0.969 ± 0.002

As such, further feature selections are done considering the features from all three domains.

5.8.3 Feature Selection by Score

Performing feature selection, 4 by 4 from 4 to 112 features, showed that, as the number of features increased, so too did the accuracy — but only up to a certain point, after which it either got slightly worse, or it barely changed; additionally, the standard deviations consistently decreased. When looking at Figure 5.39, which displays the case of the Random Forest classifier run through the three selection techniques, it is shown that Mutual Information was able to reach high accuracies with fewer features than the others, but that the peak accuracy happened from the ANOVA F-value technique at 88 features. Mostly all other models displayed similar behaviour, safe for the number of features corresponding to highest accuracy; as such, Figure 5.39 works as a good representation of the general behaviour.

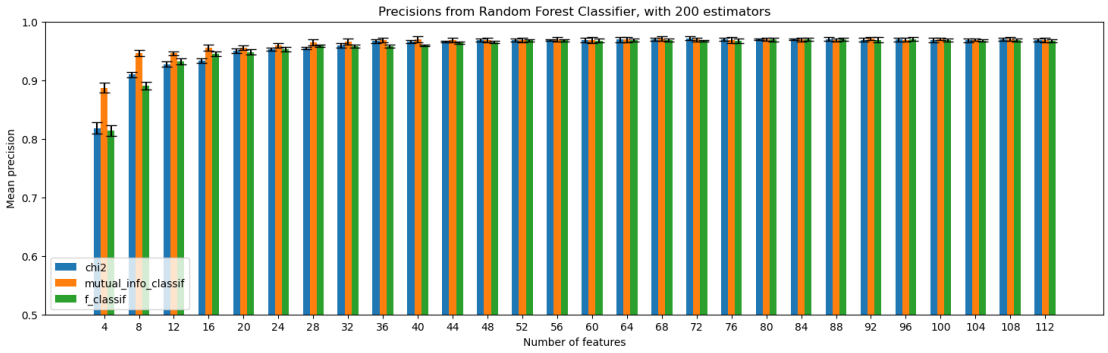


Figure 5.39: Bar chart of the accuracy of the Random Forest classifier, using different features (4 by 4, from 4 to 112) selected by the Chi-squared, ANOVA F-value and Mutual Information scoring techniques. Error bars correspond to the resulting standard deviations.

Table 5.10 shows the number of features that led to highest accuracy for each model, and the selection technique that chose the features used. As is evident, RFC provided highest accuracy, at 97.2%±0.2% (percent uncertainty of 0.2%),

Table 5.10: Results of model accuracy upon using different models and numbers of features, with features selected by the Chi-squared, ANOVA F-value and Mutual Information scoring techniques, with task feature known.

Model	Selector	No. features	Accuracy
RFC	ANOVA F-value	88	0.972 ± 0.002
SVM	Mutual Information	96	0.841 ± 0.005
LDA	ANOVA F-value	100	0.841 ± 0.006
GBC	Mutual Information	96	0.907 ± 0.006
NNC	Mutual Information	72	0.92 ± 0.01
KNN	ANOVA F-value	80	0.955 ± 0.002
MNB	Chi-Squared	100	0.704 ± 0.006

An additional exploration of the pipeline's ability to classify other aspects, namely "task" – knowing the FOG state –, and "FOG State and task", simultaneously, led to results found in Table

5.11. What is seen is that, despite the distinct nature of the target labels, "Task Only" (with more classes than "FOG State Only") showed an accuracy only 4% lower; likewise, "FOG State + Task" displayed the lowest results (both in ACC and STD), though only by 0.2% (even with double the number of classes). Additionally, in all three cases, RFC was the model conducive to highest accuracy, by margins of 2-8% in relation to the second best model (KNN, in all cases).

Table 5.11: Results of model accuracy upon classification of different aspects of PD patients' gait.

Classification Objective	Number of classes	Model	Selector	No. features	Highest Accuracy
FOG State Only (task known)	2	RFC	ANOVA F-value	88	0.972 ± 0.002
Task Only (FOG State known)	4	RFC	Chi Squared	92	0.930 ± 0.004
FOG State + Task	8	RFC	ANOVA F-value	84	0.932 ± 0.003

5.8.4 Final Assessment of Model Evaluation

With the previous steps, the entire feature selection process concludes, determining that, for classifying FOG state, and for window length of 3000 samples given 1000-Hz signal, the highest validation accuracy (accompanied by low standard deviation) happens when **all three channels** are used, **all domains** included, and **88 features as chosen by the ANOVA F-value technique** are included in training the **Random Forest model**, for an overall accuracy of 97.2%±0.2% (percent uncertainty of 0.2%).

It is with this model and features that the test-set (unseen data) classifications was followed through.

It is also important to know how many classifications the model is capable of providing along time. When given enough data for **18656 classifications**, it finished processing (filtering, computing features) and classifying all of them after an **elapsed time of 2295.7 seconds**. As such, the model is capable of providing roughly **8 classifications per second**, with each one taking about 0.125 seconds to process. In a real-life, online classification context, this classification rate proves that making predictions every 250 samples (0.25 seconds) – in a case where there would be predictions four times a second, as an example – is very feasible and isn't affected by processing time.

5.8.4.1 Evaluation of Classifications on Unseen Test Data

Applying the trained models on subjects 10 and 11's data led to predictions with few, if any, mismatches, regardless of task being performed, as is shown in Figure 5.40.

When looking at the corresponding confusion matrix, Figure 5.41, it becomes evident that, regardless of sample step, accuracies are always above 98%, as further summarized in Table 5.12. Indeed, as step size reduces, more observations are included, and though distributions may show

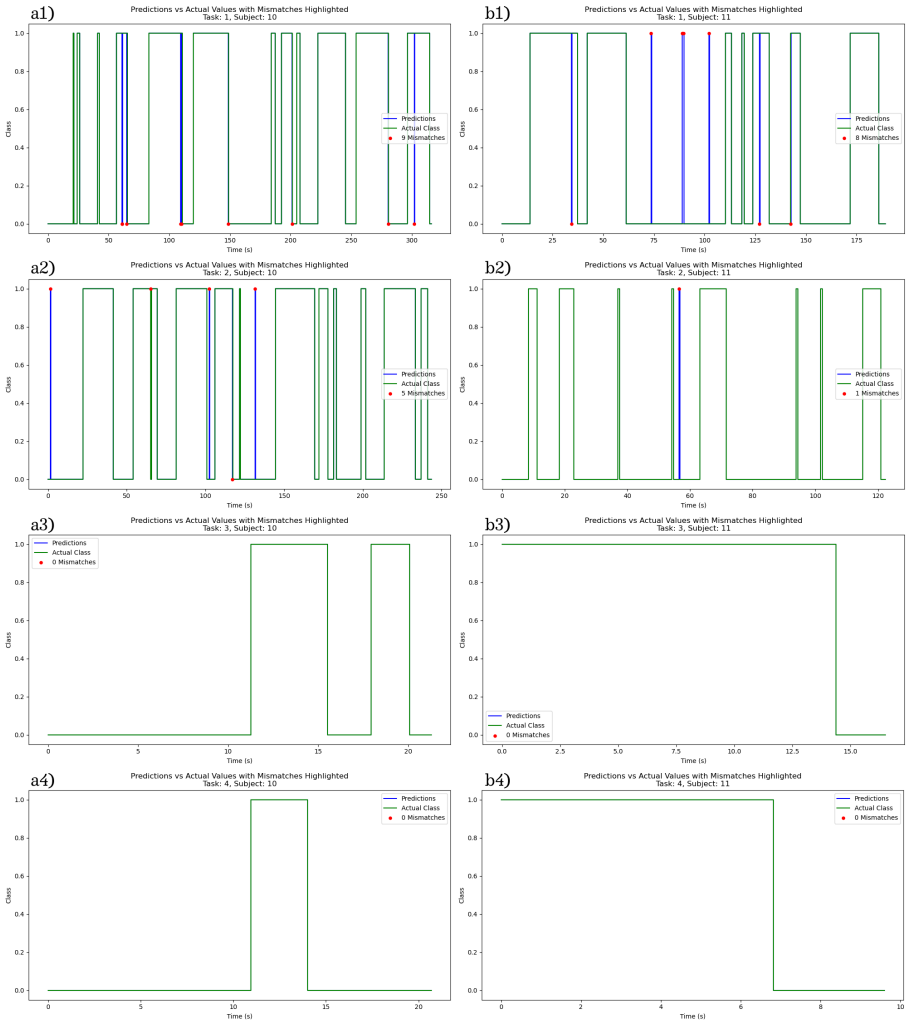


Figure 5.40: Predictions throughout entire timelines of all 4 tasks, on both subject 10 and 11, with highlighted mismatches.

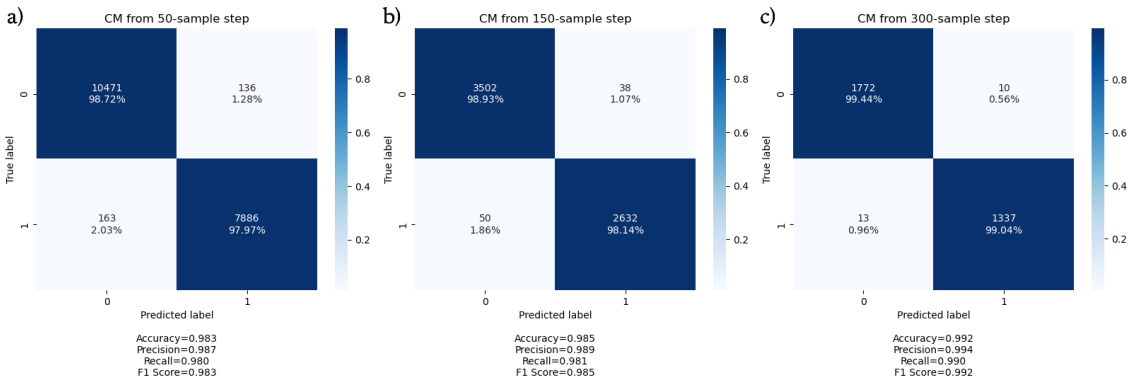


Figure 5.41: Normalized confusion matrices of FOG state prediction, on all tasks and the two test subjects, for different step sizes.

a higher bias toward task 1 and task 2, the model proves robust enough to classify FOG on other tasks, with weighted F1-scores that take into account the imbalance remaining as high as the

accuracy.

Table 5.12: Summary of results on unseen test subjects (10 and 11) over different step sizes, and corresponding distribution of observation numbers.

Step size	Total number of observations	Task	No. observations	No. mismatches	Accuracy	Weighted F1-score
300 samples	3132	1	1675	17	0.99	0.99
		2	1230	6	0.995	
		3	125	0	1	
		4	102	0	1	
150 samples	6222	1	3350	61	0.98	0.99
		2	2420	24	0.99	
		3	249	1	0.996	
		4	203	1	0.995	
50 samples	18656	1	10036	198	0.98	0.98
		2	7262	92	0.987	
		3	749	3	0.996	
		4	609	6	0.99	

Finally, when considering using this model (or pipeline) for FOG detection, one may even adopt an additional measure that may lead to increased classification accuracies. As done in this part of the work, each segment was classified and used for the identification of FOG, which Figure 5.42a shows. However, if one establishes a tolerance, such that FOG state ought only be considered mobile or frozen if a given number of sequential segments display the same classification, it can become possible to achieve a FOG timeline with no mismatches. For instance, once defined a tolerance of 50% of 1 second – meaning that, after 1 full second of classifications, the FOG state will only be considered one or the other if more than 50% of classifications are one or the other, respectively –, as shown in Figure 5.42b, the mismatches cease to be, thereby improving accuracy.

Through this pipeline, it has been shown that Machine Learning as a tool for automatic detection of gait events is a useful, accurate approach.

When it came to channels, as more are included, so too does the accuracy increase. This is compatible with the idea that each channel provides valuable information on the behavior of the muscle and that none of the additional information hinders the training (which could happen, if most of the signals from one channel was just noise or non-representative of the region of interest[241]). With that said, it is clear that the placement of electrodes at the tibialis anterior (EMG1 and EMG2) provide the most valuable information, judging by the higher accuracies resulting therefrom. In cases where only one or two channels could be used, these would be the channels to consider using.

A similar idea is shared upon domain selection. While it is evident that the time domain is most prevalent in terms of higher ACC (which isn't unexpected, since it does have more features and since EMG signals are functions of time, heavily affecting the characteristics in this domain), with fewer features the frequency domain was still able to provide nearly-equally high ACC (presumably, due to the specificity of how cyclic the behavior is upon each task, in stark contrast with

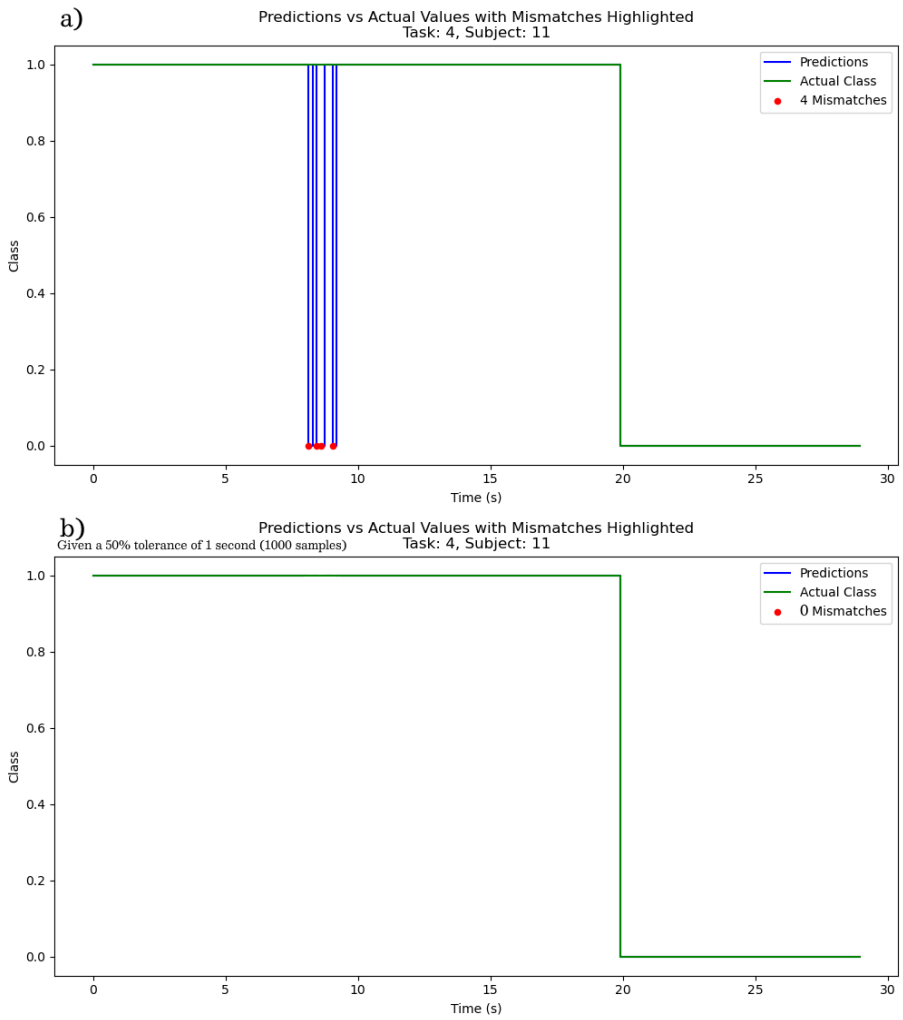


Figure 5.42: Correction of FOG detection mismatches through tolerance adjustment.

how it becomes upon freezing during task). It is arguable that the fractional Fourier transform domain could be excluded, since its inclusion alongside time and frequency only reduces the error, but the computation of its features is not very intensive, so its benefit is still worth enjoying.

When it came to feature selection by score, it is evident that the ANOVA F-value scoring technique provided the lower number of features conducive to higher ACC (still with low STD) — this might be explained by the fact that this is a technique that computes each feature's F-value and can assess the extent to which they contribute to the differences observed between groups of features, thereby determining how irrelevant some features may be overall.

Chi-squared barely ever showed as high of results, which may be explained by the fact that it is a technique whose choices are based on the degree of association between categorical variables (the variables worked with in this work are all numerical – save for task number –, with some having also some dependency to others)[242].

Something else to note is the fact that, at every point, the Random Forest led to the highest accuracies. It is also based on measurements of the importance of features; going in tandem with

the F-value feature selector is presumably what led to the highest results obtained.

Despite close to 100% (the ideal value), the results obtained are not without flaw, and further refinements of the steps taken in the work would surely lead to improvements.

As far as signal processing goes, some measures could be taken, such as readjusting the band-pass frequencies to slowly find the very best range, or introducing some filtering techniques to clear out noise that specifically acts during moments of lactic build-up.

Feature-wise, it is not impossible that adding several more features within the fractional Fourier transform domain (or even the other domains, like time-frequency), or slightly altering the alpha, could be conducive to higher ACC. While its contribution overall wasn't very noticeable, there is still potential for further tinkering.

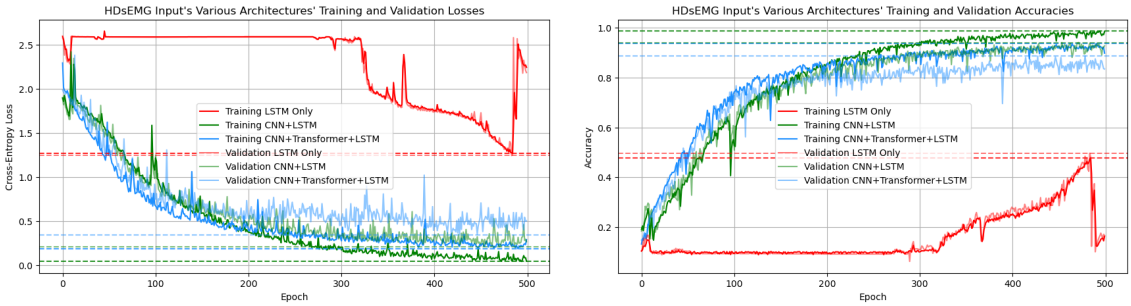
The same can be said about the models themselves. Each one is configured with certain hyper-parameters which, in this work, weren't explored very in-depth. Developing routines that attempt to fine-tune the hyperparameters used could prove useful in raising the models' resulting accuracy.

Finally, in terms of online acquisition, it would take being able to provide this model to a PD patient performing the tasks – even testing with more subjects and in more varied situations – to fully assert how generally accurate the model can be in more real-life context.

5.9 Deep Learning-Based Gesture and Health Condition Classification

5.9.1 HDsEMG Images as Input Data

Upon training, at every epoch resulted losses and accuracies, Figure 5.43, which displayed particular behaviours: when looking at LSTM-only model, it was only at around epoch 320 that losses started to decrease and, even once it reached in minimum, it returned to its initial cross-entropy losses.



(a) Losses through epoch

(b) Accuracies through epoch

Figure 5.43: Training and validation results for different architectures, given HDsEMG as input, upon DMD gesture and condition classification.

This behavior can be attributed to the LSTM's inability to efficiently capture spatial hierarchies in the data, especially when dealing with high-dimensional signals like HDsEMG. LSTMs are

well-suited for sequential processing but struggle with high-dimensional spatial input, causing slow convergence. The recurring increase in loss after epoch 320 might indicate that the model is prone to overfitting or encountering vanishing gradient issues, leading to unstable learning. This is a typical limitation of relying solely on LSTMs for high-dimensional temporal data, as it lacks the representational power needed to extract key spatial features early in training.

On the other hand, CNN+LSTM and CNN+Transformer+LSTM showed a much more efficient ability to learn from the HDsEMG input data. CNN+Transformer+LSTM did appear more efficient in the beginning, though by epoch 200, a crossover of losses happens, with CNN+LSTM now reaching lower (and lowest) losses; equivalently, for validation, the losses mostly followed the training losses trends, save for CNN+Transformer+LSTM which started to stagnate earlier than training.

This difference in behavior can be explained by the models' architectures. The addition of the CNN layers in both hybrid models provides the ability to extract spatial features from the input data early in the learning process, improving the learning efficiency. CNNs are particularly adept at detecting local patterns, which helps capture essential features from the HDsEMG signals before passing them to the LSTM or Transformer for temporal modeling. The initial superiority of CNN+Transformer+LSTM suggests that Transformers are more efficient at modeling temporal dependencies in the early stages of training. However, the eventual crossover could be due to the fact that, for this particular dataset, the simpler CNN+LSTM architecture was more suited to handling the longer-range dependencies after the spatial features were extracted (one may also suppose that the CNN+Transformer+LSTM may perform better on more complex data, or more volume of data). The Transformer's tendency to overfit or plateau when not tuned correctly for the data could explain the stagnation in losses during validation.

Analogously, the training and validations accuracies increase through epochs, though LSTM's only ever reaches accuracies of 50%, whereas the CNN+Transformer+LSTM reaches 94% on training and 89% on validation and test. With the highest accuracy results, CNN+LSTM showed 4% more than the previous architecture on all accounts – summarized in Table 5.13. To note that the test accuracy herein used is an intrasubject test accuracy, meaning that the test data constitutes subjects already trained upon (though with the test data portion not previously seen by the model).

Overall, the performance differences in accuracy can perhaps be explained by how well the models generalize. The CNN+LSTM's superior accuracy suggests that its combination of spatial and temporal modeling fits the data better, allowing for improved feature learning and generalization across both training and validation sets. While the Transformer offers powerful sequence modeling capabilities, its complexity might lead to overfitting, which could explain its slightly lower validation accuracy. Additionally, the LSTM-only model's poor performance reflects its difficulty in learning from high-dimensional data, where spatial relationships are crucial.

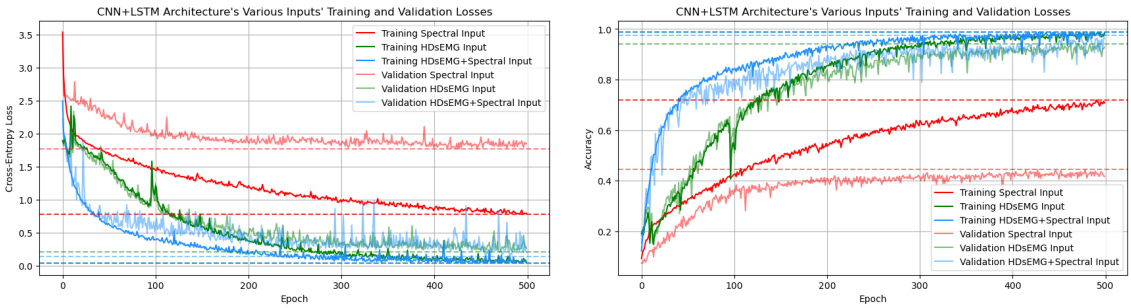
5.9.2 Spectral Images as Input Data

Due to the highest results acquired with CNN+LSTM, this architecture was also used for comparing results given different input information; regarding losses, which can also be assessed visually

Table 5.13: Summary of final accuracy and loss results over 500 epochs for different architectures, given HDsEMG as input.

Architecture	Train Loss	Train Accuracy	Validation Loss	Validation Accuracy	Test Accuracy
LSTM-only	1.26	0.48	1.2	0.50	0.48
CNN+LSTM	0.040	0.98	0.21	0.94	0.93
CNN+Transformer+LSTM	0.18	0.94	0.34	0.89	0.89

in Figure 5.44, it was found that using CWT+STFT+HHT (the spectral input) alone did not prove enough to provide lower losses, when compared to HDsEMG alone. Whereas the latter reached 0.040 loss, the former only got to 0.78 (about 20 times of the latter), to train and validation accuracies of 72% and 50%, respectively – about 25% and 45% lower than its HDsEMG input counterpart, with test accuracies only reaching half of said counterpart's – information summarized in Table 5.14.



(a) Losses through epoch

(b) Accuracies through epoch

Figure 5.44: Training and validation results for branched CNN+LSTM architecture over 500 epochs, given different inputs (HDsEMG, Spectral, HDsEMG+Spectral), upon DMD gesture and condition classification.

This performance gap can be attributed to the nature of the input data. HDsEMG signals inherently carry rich spatial and temporal information, making them better suited for CNN+LSTM architectures. In contrast, the spectral representations (CWT, STFT, HHT) primarily capture frequency-domain information, which may not be as effective in capturing the complex, multi-dimensional patterns present in HDsEMG data. Spectral inputs, while valuable for analyzing certain features, lack the direct spatial relationships that HDsEMG provides, leading to poorer

performance when used alone. Additionally, the relatively low validation accuracy of 50% and significantly higher loss values suggest that the model struggled to generalize from the spectral inputs, possibly due to overfitting or an inability to capture the necessary temporal dynamics of the signals.

On the other hand, when branching these inputs, into HDsEMG+(STFT,CWT,HHT), upon the CNN+LSTM architecture, is appears that not only was the model more efficient at learning (with a much steeper decreasing in loss, and increase in accuracy), it was able to go over HDsEMG-only's accuracies – going from 93% test accuracy to 96%.

This improvement can be explained by the complementary nature of the inputs. By combining HDsEMG with spectral representations, the model can leverage both spatial and temporal information from the raw HDsEMG signals while also gaining valuable frequency-domain insights from the spectral inputs. The spectral features provide a broader perspective on the signal's oscillatory behavior and underlying patterns, enriching the input data and enhancing the model's ability to differentiate between subtle patterns in the HDsEMG signals. The combination of spatial, temporal, and frequency-domain features allows the CNN to extract more meaningful representations, while the LSTM further improves temporal modeling, resulting in better generalization and higher test accuracy.

Table 5.14: Summary of final accuracy and loss results for branched CNN+LSTM architecture, given different inputs (HDsEMG, Spectral, HDsEMG+Spectral).

Input (Architecture: CNN+LSTM)	**Train Loss**	**Train Accuracy**	**Validation Loss**	**Validation Accuracy**	**Test Accuracy**
HDsEMG	0.040	0.98	0.21	0.94	0.93
(STFT, CWT, HHT)	0.78	0.72	1.8	0.50	0.46
HDsEMG + (STFT, CWT, HHT)	0.036	0.99	0.14	0.97	0.96

5.9.3 Final Assessment of Model Evaluation

The evaluation of the models across different architectures and input types provides crucial insights into their performance, scalability, and potential integration into a pipeline for training models from new signal uploads to a database.

The CNN+LSTM architecture consistently outperformed other models (LSTM-only, CNN +Transformer+LSTM), especially when applied to HDsEMG input. This suggests that the combination of CNN's spatial feature extraction and LSTM's temporal modeling is particularly well-suited for handling the complexity of HDsEMG data, which contains both spatial and temporal dependencies. The high training (98%) and validation/test accuracies (93%) of CNN+LSTM show its robustness and ability to generalize well, making it a reliable choice for future applications.

Moreover, when combining different input modalities, such as HDsEMG with spectral representations (CWT, STFT, HHT), the CNN+LSTM architecture demonstrated even stronger performance. By integrating the strengths of the spatial-temporal features from HDsEMG and the frequency-domain insights from the spectral inputs, the model achieved its highest intrasubject test accuracy of 96%. This suggests that a multi-modal input strategy is highly beneficial, allowing the model to capture different aspects of the data, which in turn improves classification accuracy and generalization.

From a deployment perspective – considering the pipeline built to train models on newly uploaded signals to a database – several conclusions can be attained:

- Model Efficiency and Training Time: The CNN+LSTM architecture showed efficient learning behavior, as evidenced by the steep decline in loss during training and its ability to reach high accuracies without overfitting. The use of spectral inputs alone was not as effective, but branching these inputs with HDsEMG produced the best results. This efficiency in learning is crucial when designing a pipeline to handle potentially large volumes of incoming data, allowing for faster training and model convergence.

- Modular Architecture for Future Input Flexibility: Since the CNN+LSTM model performs exceptionally well with combined input types (HDsEMG + spectral), the pipeline can be designed to be modular. This means that the model can be flexibly trained on different combinations of inputs based on the available data. New signal uploads to the database could include either HDsEMG data, spectral data, or both. The ability to easily switch between input modalities or branch them as needed may add flexibility for handling a wide variety of new signals and datasets in the future.

- Scalability and Generalization: The model's generalization capability is demonstrated by the relatively small performance gap between training and test accuracies. This indicates that the model is unlikely to overfit, making it well-suited for a pipeline where the goal is to handle new signal uploads from a variety of sources and subjects. As new signals are uploaded and processed, the model is expected to maintain its ability to generalize to unseen data (be it for training or classifying). This scalability makes the CNN+LSTM architecture a strong candidate for building an adaptable and efficient training pipeline.

Finally, the CNN+LSTM model, particularly when branching HDsEMG with spectral inputs (STFT, CWT, HHT), emerges as the best-performing architecture. Its high accuracy, efficient learning curve, and ability to generalize well to new data make it ideal for deployment in a pipeline

designed to handle new signal uploads. By incorporating automated preprocessing, flexible input configurations, and adaptive learning, this model can form the foundation of a robust system capable of continuously improving as more data becomes available either to train the current model for classifying between these 7 gestures and healthy or DMD conditions, either to train new models for classifying new movements or conditions.

Chapter 6

Conclusions

This dissertation presented a comprehensive analysis and development of HDsEMG signal processing techniques and ML models, targeting clinical applications. By implementing algorithms for pre-processing, and for amplitude, spectral, and timing analysis, the system can provide a detailed understanding of muscle dynamics, from being able to differentiate between different states of muscle activation, to being able to display the frequency components that describe it. Alongside these, conduction velocity estimation was made possible by resorting to two methods, one based on cross-correlating nearby signals, and the other on obtaining a spike-triggered average of the signals at moments of motor unit firings and performing iterative differential beamformings, resulting in estimates that, within the expected values of conduction, allowed for the assessment of how they may change upon different actions and when those actions are performed by a healthy individual or a individual with Duchenne muscular dystrophy. Innervation zone estimation was made possible through a PCA approach which proved robust for simpler signals, and Motor Unit Decomposition, through the use of a Blind Source Separation refinement-based algorithm, was able to discern different firings to high rates of agreement on relatively short tolerance intervals, as well as provide instantaneous firing rate, signal strength and motor unit action potential shapes close to the ground-truth. The integration of machine learning, particularly through traditional feature engineering, proved effective in detecting gait abnormalities, specifically for Freezing of Gait on patient with Parkinson's Disease, achieving high accuracy on unseen test subjects. Furthermore, the creation of a digital platform to simplify the uploading and preprocessing of HDsEMG data, enabling for model training or classification from a pre-trained model, was shown to be an adequate approach to facilitating training deep-learning models, by following a consistent pipeline; the results from the training of Neural Networks on data describing healthy or Duchenne patients as they performed seven different gestures highlight the potential for these methodologies to enhance clinical diagnostics in neuromuscular and rehabilitation medicine, demonstrating significant progress toward the practical application of HDsEMG for patient monitoring and care. Future work will aim to add new EMG analytical algorithms, refine the previous methods, increase scalability of the machine learning applications, and expand the software's tools use across diverse clinical environments.

Bibliography

[1] HfPT, "Health from Portugal Agenda HfPT." https://www.healthfromportugal.pt/pt/. Accessed: 2024-09-02.

[2] J. V. Basmajian and C. J. De Luca, *Muscles Alive: Their Functions Revealed By Electromyography. By J. V. Basmajian*. The Williams & Wilkins Company, Baltimore, 1962.

[3] L. Galvani, *Aloysii Galvani De viribus electricitatis in motu musculari commentarius*. Ex Typographia Instituti Scientiarium, 1791.

[4] M. B. I. Reaz, M. S. Hussain, and F. Mohd-Yasin, "Techniques of EMG signal analysis: detection, processing, classification and applications," *Biological Procedures Online*, vol. 8, p. 11–35, Dec. 2006.

[5] J. P. Clarys, "Electrology and localized electrization revisited," *Journal of Electromyography and Kinesiology*, vol. 4, p. 5–14, Jan. 1994.

[6] E. D. Adrian and D. W. Bronk, "The discharge of impulses in motor nerve fibres: Part II. The frequency of discharge in reflex and voluntary contractions," *The Journal of Physiology*, vol. 67, p. 9–151, Mar. 1929.

[7] D. Denny-Brown and J. B. Pennybacker, "FIBRILLATION AND FASCICULATION IN VOLUNTARY MUSCLE," *Brain*, vol. 61, no. 3, p. 311–312, 1938.

[8] C. Jeffrey, R. Glenn, S. Kasman, and J. Holtz, *Introduction to Surface Electromyography*. Gaithersburg Md: Aspen, 1998.

[9] X. Song, H. Li, and W. Gao, "MyoMonitor: Evaluating muscle fatigue with commodity smartphones," *Smart Health*, vol. 19, p. 100175, Mar. 2021.

[10] E. Henneman, G. Somjen, and D. O. Carpenter, "FUNCTIONAL SIGNIFICANCE OF CELL SIZE IN SPINAL MOTONEURONS," *Journal of Neurophysiology*, vol. 28, p. 560–580, May 1965.

[11] E. Henneman, "Relation between Size of Neurons and Their Susceptibility to Discharge," *Science*, vol. 126, p. 1345–1347, Dec. 1957.

[12] D. M. Halliday, B. A. Conway, S. F. Farmer, and J. R. Rosenberg, "Load-Independent Contributions From Motor-Unit Synchronization to Human Physiological Tremor," *Journal of Neurophysiology*, vol. 82, p. 664–675, Aug. 1999.

[13] W. Yao, R. J. Fuglevand, and R. M. Enoka, "Motor-Unit Synchronization Increases EMG Amplitude and Decreases Force Steadiness of Simulated Contractions," *Journal of Neurophysiology*, vol. 83, p. 441–452, Jan. 2000.

[14] R. Beck, "Muscle Fiber Conduction Velocity," *Wiley Encyclopedia of Biomedical Engineering*, Apr. 2006.

[15] S. D. Nandedkar and E. Stlberg, "Simulation of single muscle fibre action potentials," *Medical & Biological Engineering & Computing*, vol. 21, p. 158–165, Mar. 1983.

[16] T. Sadoyama, T. Masuda, H. Miyata, and S. Katsuta, "Fibre conduction velocity and fibre composition in human vastus lateralis," *European Journal of Applied Physiology and Occupational Physiology*, vol. 57, p. 767–771, Nov. 1988.

[17] S. Andreassen and L. Arendt-Nielsen, "Muscle fibre conduction velocity in motor units of the human anterior tibial muscle: a new size principle parameter.," *The Journal of Physiology*, vol. 391, p. 561–571, Oct. 1987.

[18] L. Arendt-Nielsen and K. Mills, "The relationship between mean power frequency of the EMG spectrum and muscle fibre conduction velocity," *Electroencephalography and Clinical Neurophysiology*, vol. 60, p. 130–134, Feb. 1985.

[19] R. Merletti, M. Knaflitz, and C. J. De Luca, "Myoelectric manifestations of fatigue in voluntary and electrically elicited contractions," *Journal of Applied Physiology*, vol. 69, p. 1810–1820, Nov. 1990.

[20] C. J. De Luca, "Physiology and Mathematics of Myoelectric Signals," *IEEE Transactions on Biomedical Engineering*, vol. BME-26, p. 313–325, June 1979.

[21] J. Van der Hoeven, M. Zwarts, and T. Van Weerden, "Muscle fiber conduction velocity in amyotrophic lateral sclerosis and traumatic lesions of the plexus brachialis," *Electroencephalography and Clinical Neurophysiology/Evoked Potentials Section*, vol. 89, p. 304–310, Oct. 1993.

[22] B. Gerdle, N. Östlund, C. Grönlund, K. Roeleveld, and J. S. Karlsson, "Firing rate and conduction velocity of single motor units in the trapezius muscle in fibromyalgia patients and healthy controls," *Journal of Electromyography and Kinesiology*, vol. 18, p. 707–716, Oct. 2008.

[23] D. Farina, A. M. Castronovo, I. Vujaklija, A. Sturma, S. Salminger, C. Hofer, and O. Aszmann, "Common Synaptic Input to Motor Neurons and Neural Drive to Targeted Reinnervated Muscles," *The Journal of Neuroscience*, vol. 37, p. 11285–11292, Oct. 2017.

[24] W. H. J. P. Linssen, D. F. Stegeman, E. M. G. Joosten, R. A. Binkhorst, M. J. H. Merks, H. J. T. Laak, and S. L. H. Notermans, "Fatigue in type I fiber predominance: A muscle force and surface EMG study on the relative role of type I and type II muscle fibers," *Muscle & Nerve*, vol. 14, p. 829–837, Sept. 1991.

[25] J. P. d. Paula, M. T. Navega, and D. H. Spinoso, "Neuromuscular efficiency of the quadriceps in women with and without patellofemoral pain," *Fisioterapia e Pesquisa*, vol. 30, 2023.

[26] J. A. Zellers, S. Parker, A. Marmon, and K. Grävare Silbernagel, "Muscle activation during maximum voluntary contraction and m-wave related in healthy but not in injured conditions: Implications when normalizing electromyography," *Clinical Biomechanics*, vol. 69, p. 104–108, Oct. 2019.

[27] R. E. Singh, K. Iqbal, G. White, and J. K. Holtz, "A Review of EMG Techniques for Detection of Gait Disorders," in *Artificial Intelligence* (M. A. Aceves-Fernandez, ed.), ch. 2, Rijeka: IntechOpen, 2019.

[28] A. Hibbs, K. Thompson, D. French, D. Hodgson, and I. Spears, "Peak and average rectified EMG measures: Which method of data reduction should be used for assessing core training exercises?," *Journal of Electromyography and Kinesiology*, vol. 21, p. 102–111, Feb. 2011.

[29] A. E. Hibbs, K. G. Thompson, D. French, A. Wrigley, and I. Spears, "Optimizing Performance by Improving Core Stability and Core Strength," *Sports Medicine*, vol. 38, no. 12, p. 995–1008, 2008.

[30] T. I. Arabadzhiev, V. G. Dimitrov, N. A. Dimitrova, and G. V. Dimitrov, "Interpretation of EMG integral or RMS and estimates of "neuromuscular efficiency" can be misleading in fatiguing contraction," *Journal of Electromyography and Kinesiology*, vol. 20, p. 223–232, Apr. 2010.

[31] E. Clancy and N. Hogan, "Single site electromyograph amplitude estimation," *IEEE Transactions on Biomedical Engineering*, vol. 41, no. 2, pp. 159–167, 1994.

[32] L. Liu, P. Liu, E. A. Clancy, E. Scheme, and K. B. Englehart, "Whitening of the electromyogram for improved classification accuracy in prosthesis control," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, IEEE, Aug. 2012.

[33] E. Clancy, E. Morin, and R. Merletti, "Sampling, noise-reduction and amplitude estimation issues in surface electromyography," *Journal of Electromyography and Kinesiology*, vol. 12, p. 1–16, Feb. 2002.

[34] D. Farina and R. Merletti, *Surface Electromyography: Physiology, Engineering, and Applications*. JOHN WILEY & SONS INC, Apr. 2016.

[35] G. Rasool, B. Afsharipour, N. L. Suresh, X. Hu, and W. Z. Rymer, "Spatial analysis of muscular activations in stroke survivors," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Aug. 2015.

[36] M. Rojas-Martínez, M. Mañanas, J. Alonso, and R. Merletti, "Identification of isometric contractions based on High Density EMG maps," *Journal of Electromyography and Kinesiology*, vol. 23, p. 33–42, Feb. 2013.

[37] U. Kuruganti, A. Pradhan, and J. Toner, "High-Density Electromyography Provides Improved Understanding of Muscle Function for Those With Amputation," *Frontiers in Medical Technology*, vol. 3, Aug. 2021.

[38] G. van Elswijk, B. U. Kleine, S. Overeem, B. Eshuis, K. D. Hekkert, and D. F. Stegeman, "Muscle imaging: Mapping responses to transcranial magnetic stimulation with high-density surface electromyography," *Cortex*, vol. 44, p. 609–616, May 2008.

[39] D. Farina, F. Leclerc, L. Arendt-Nielsen, O. Buttelli, and P. Madeleine, "The change in spatial distribution of upper trapezius muscle activity is correlated to contraction duration," *Journal of Electromyography and Kinesiology*, vol. 18, p. 16–25, Feb. 2008.

[40] B. R. Schlink, A. D. Nordin, and D. P. Ferris, "Human myoelectric spatial patterns differ among lower limb muscles and locomotion speeds," *Physiological Reports*, vol. 8, Dec. 2020.

[41] V. Agostini, F. Lo Fermo, G. Massazza, and M. Knaflitz, "Does texting while walking really affect gait in young adults?," *Journal of NeuroEngineering and Rehabilitation*, vol. 12, Sept. 2015.

[42] S. Uthvag, P. Vijay Sai, S. Dheeraj Kumar, H. Muthusamy, O. R. Chanu, and V. Karthik Raj, "REAL-TIME EMG ACQUISITION AND FEATURE EXTRACTION FOR REHABILITATION AND PROSTHESIS," *Biomedical Engineering: Applications, Basis and Communications*, vol. 31, p. 1950037, Sept. 2019.

[43] T. Kosaki, K. Atsuumi, Y. Takahashi, and S. Li, "A pneumatic arm power-assist system prototype with EMG-based muscle activity detection," in *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 793–798, 2017.

[44] C. D'Anna, T. Varrecchia, M. Schmid, and S. Conforto, "Using the frequency signature to detect muscular activity in weak and noisy myoelectric signals," *Biomedical Signal Processing and Control*, vol. 52, p. 69–76, July 2019.

[45] G. Allison, "Trunk muscle onset detection technique for EMG signals with ECG artefact," *Journal of Electromyography and Kinesiology*, vol. 13, p. 209–216, June 2003.

[46] D. Yang, Q. Huang, W. Yang, and H. Liu, "EMG Onset Detection Based on Teager–Kaiser Energy Operator and Morphological Close Operation," in *Intelligent Robotics and Applications* (H. Liu, N. Kubota, X. Zhu, R. Dillmann, and D. Zhou, eds.), (Cham), pp. 257–268, Springer International Publishing, 2015.

[47] J. Liu and Q. Liu, "Use of the integrated profile for voluntary muscle activity detection using EMG signals with spurious background spikes: A study with incomplete spinal cord injury," *Biomedical Signal Processing and Control*, vol. 24, p. 19–24, Feb. 2016.

[48] M. Santello and M. McDonagh, "The control of timing and amplitude of EMG activity in landing movements in humans," *Experimental Physiology*, vol. 83, p. 857–874, Nov. 1998.

[49] P. Zhou and X. Zhang, "A novel technique for muscle onset detection using surface EMG signals without removal of ECG artifacts," *Physiological Measurement*, vol. 35, p. 45–54, Dec. 2013.

[50] M. S. Tenan, A. J. Tweedell, and C. A. Haynes, "Analysis of statistical and standard algorithms for detecting muscle onset with surface electromyography," *PLOS ONE*, vol. 12, p. e0177312, May 2017.

[51] D. Yang, H. Zhang, Y. Gu, and H. Liu, "Accurate EMG onset detection in pathological, weak and noisy myoelectric signals," *Biomedical Signal Processing and Control*, vol. 33, p. 306–315, Mar. 2017.

[52] X. Zhang and P. Zhou, "Sample entropy analysis of surface EMG for improved muscle activity onset detection against spurious background spikes," *Journal of Electromyography and Kinesiology*, vol. 22, p. 901–907, Dec. 2012.

[53] M. Lyu, C. Xiong, Q. Zhang, and L. He, "Fuzzy Entropy-Based Muscle Onset Detection Using Electromyography (EMG)," in *Intelligent Robotics and Applications* (X. Zhang, H. Liu, Z. Chen, and N. Wang, eds.), (Cham), pp. 89–98, Springer International Publishing, 2014.

[54] G. Rasool, K. Iqbal, and G. A. White, "Myoelectric activity detection during a Sit-to-Stand movement using threshold methods," *Computers & Mathematics with Applications*, vol. 64, pp. 1473–1483, Sept. 2012.

[55] H. Bengacemi, A. Mesloub, A. Ouldali, and K. Abed-Meraim, "Adaptive Linear Energy Detector based on onset and offset electromyography activity detection," in *2017 6th International Conference on Systems and Control (ICSC)*, (Batna, Algeria), pp. 409–413, IEEE, May 2017.

[56] S. Spasojevic, A. Rodrigues, K. Mahdaviani, W. D. Reid, A. Mihailidis, and S. S. Khan, "Onset and Offset Detection of Respiratory EMG Data Based on Energy Operator Signal," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, (Mexico), pp. 121–124, IEEE, Nov. 2021.

[57] V. Bairagi and A. B. Kanwade, "Novel slope-based onset detection algorithm for electromyographical signals," *International Journal of Biomedical Engineering and Technology*, vol. 36, no. 4, p. 311, 2021.

[58] S. Bhardwaj, A. A. Khan, and M. Muzammil, "Onset difference between vastus lateralis and knee extension during sit to stand task," in *AIP Conference Proceedings*, AIP Publishing, 2019.

[59] G. Vannozzi, S. Conforto, and T. D'Alessio, "Automatic detection of surface EMG activation timing using a wavelet transform based method," *Journal of Electromyography and Kinesiology*, vol. 20, p. 767–772, Aug. 2010.

[60] P. McCool, N. Chatlani, L. Petropoulakis, J. J. Soraghan, R. Menon, and H. Lakany, "Lower Arm Electromyography (EMG) Activity Detection Using Local Binary Patterns," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, p. 1003–1012, Sept. 2014.

[61] G. M. Allen, S. C. Gandevia, and D. K. McKenzie, "Reliability of measurements of muscle strength and voluntary activation using twitch interpolation," *Muscle & Nerve*, vol. 18, p. 593–600, June 1995.

[62] J. Cannon, D. Kay, K. M. Tarpenning, and F. E. Marino, "Comparative effects of resistance training on peak isometric torque, muscle hypertrophy, voluntary activation and surface EMG between young and elderly women," *Clinical Physiology and Functional Imaging*, vol. 27, p. 91–100, Jan. 2007.

[63] M. Ghislieri, G. L. Cerone, M. Knaflitz, and V. Agostini, "Long short-term memory (LSTM) recurrent neural network for muscle activity detection," *Journal of NeuroEngineering and Rehabilitation*, vol. 18, Oct. 2021.

[64] S. V. Perumal and R. Sankar, "Gait monitoring system for patients with Parkinson's disease using wearable sensors," in *2016 IEEE Healthcare Innovation Point-Of-Care Technologies Conference (HI-POCT)*, IEEE, Nov. 2016.

[65] M. V. d. Costa, H. Bortolotti, T. V. Camata, J. R. Cardoso, R. Da Silva, T. Abrão, A. Moraes, and L. Altimari, "Análise espectral do sinal EMG de exercício incremental em ciclistas e não ciclistas usando as transformadas de Fourier e Wavelet," *Revista Brasileira de Cineantropometria e Desempenho Humano*, vol. 14, Nov. 2012.

[66] P. Bonato, G. Gagliati, and M. Knaflitz, "Analysis of myoelectric signals recorded during dynamic contractions," *IEEE Engineering in Medicine and Biology Magazine*, vol. 15, no. 6, pp. 102–111, 1996.

[67] I. Hostens, J. Seghers, A. Spaepen, and H. Ramon, "Validation of the wavelet spectral estimation technique in Biceps Brachii and Brachioradialis fatigue assessment during prolonged low-level static and dynamic contractions," *Journal of Electromyography and Kinesiology*, vol. 14, p. 205–215, Apr. 2004.

[68] S. Karlsson, J. Yu, and M. Akay, "Time-frequency analysis of myoelectric signals during dynamic contractions: a comparative study," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 2, p. 228–238, 2000.

[69] D. Kumar, N. Pah, and A. Bradley, "Wavelet analysis of surface electromyography," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 11, p. 400–406, Dec. 2003.

[70] S. Karlsson, J. Yu, and M. Akay, "Enhancement of spectral analysis of myoelectric signals during static contractions using wavelet methods," *IEEE Transactions on Biomedical Engineering*, vol. 46, no. 6, pp. 670–684, 1999.

[71] T. W. Beck, T. J. Housh, G. O. Johnson, J. P. Weir, J. T. Cramer, J. W. Coburn, and M. H. Malek, "Comparison of Fourier and wavelet transform procedures for examining the mechanomyographic and electromyographic frequency domain responses during fatiguing isokinetic muscle actions of the biceps brachii," *Journal of Electromyography and Kinesiology*, vol. 15, p. 190–199, Apr. 2005.

[72] E. Sarno Filho, A. Santos, E. Simas Filho, A. C. L. Fernandes Junior, and J. d. Seixas, "Transformada de Hilbert-Huang Modificada com Redução do Ruído para Sinais de Sonar Passivo," in *Anais do XXXIX Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*, SBrT2021, Sociedade Brasileira de Telecomunicações, 2021.

[73] A. Del Vecchio, F. Negro, F. Felici, and D. Farina, "Associations between motor unit action potential parameters and surface EMG features," *Journal of Applied Physiology*, vol. 123, p. 835–843, Oct. 2017.

[74] C. Krogh-Lund and K. Jørgensen, "Myo-electric fatigue manifestations revisited: power spectrum, conduction velocity, and amplitude of human elbow flexor muscles during isolated and repetitive endurance contractions at 30% maximal voluntary contraction," *European Journal of Applied Physiology and Occupational Physiology*, vol. 66, p. 161–173, Mar. 1993.

[75] G. Drost, "Propagation disturbance of motor unit action potentials during transient paresis in generalized myotonia: A high-density surface EMG study," *Brain*, vol. 124, p. 352–360, Feb. 2001.

[76] E. Martinez-Valdes, D. Farina, F. Negro, A. Del Vecchio, and D. Falla, "Early Motor Unit Conduction Velocity Changes to High-Intensity Interval Training versus Continuous Training," *Medicine & Science in Sports & Exercise*, vol. 50, p. 2339–2350, Nov. 2018.

[77] D. Farina, M. Pozzo, E. Merlo, A. Bottin, and R. Merletti, "Assessment of Average Muscle Fiber Conduction Velocity From Surface EMG Signals During Fatiguing Dynamic Contractions," *IEEE Transactions on Biomedical Engineering*, vol. 51, p. 1383–1393, Aug. 2004.

[78] K. C. McGill and L. J. Dorfman, "High-Resolution Alignment of Sampled Waveforms," *IEEE Transactions on Biomedical Engineering*, vol. BME-31, p. 462–468, June 1984.

[79] I. W. Hunter, R. E. Kearney, and L. A. Jones, "Estimation of the conduction velocity of muscle action potentials using phase and impulse response function techniques," *Medical & Biological Engineering & Computing*, vol. 25, p. 121–126, Mar. 1987.

[80] W. Muhammad, O. Meste, and H. Rix, "Comparison of single and multiple time delay estimators: application to muscle fiber conduction velocity estimation," *Signal Processing*, vol. 82, p. 925–940, June 2002.

[81] D. Farina, W. Muhammad, E. Fortunato, O. Meste, R. Merletti, and H. Rix, "Estimation of single motor unit conduction velocity from surface electromyogram signals detected with linear electrode arrays," *Medical and Biological Engineering and Computing*, vol. 39, p. 225–236, Mar. 2001.

[82] I. W. Hunter, R. E. Kearney, and L. A. Jones, "Estimation of the conduction velocity of muscle action potentials using phase and impulse response function techniques," *Medical & Biological Engineering & Computing*, vol. 25, p. 121–126, Mar. 1987.

[83] D. Farina and R. Merletti, "Estimation of average muscle fiber conduction velocity from two-dimensional surface EMG recordings," *Journal of Neuroscience Methods*, vol. 134, p. 199–208, Apr. 2004.

[84] I. Yaar and L. Niles, "Improved techniques for measuring muscle fiber conduction velocity," *Muscle & Nerve*, vol. 15, p. 410–418, Mar. 1992.

[85] F. Lange, T. W. Van Weerden, and J. H. Van Der Hoeven, "A new surface electromyography analysis method to determine spread of muscle fiber conduction velocities," *Journal of Applied Physiology*, vol. 93, p. 759–764, Aug. 2002.

[86] D. Farina and R. Merletti, "Methods for estimating muscle fibre conduction velocity from surface electromyographic signals," *Medical & Biological Engineering & Computing*, vol. 42, p. 432–445, July 2004.

[87] S. J. Day and M. Hulliger, "Experimental Simulation of Cat Electromyogram: Evidence for Algebraic Summation of Motor-Unit Action-Potential Trains," *Journal of Neurophysiology*, vol. 86, p. 2144–2158, Nov. 2001.

[88] R. Merletti and D. Farina, "Analysis of intramuscular electromyogram signals," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 367, p. 357–368, Nov. 2008.

[89] M. J. Zwarts and D. F. Stegeman, "Multichannel surface EMG: Basic aspects and clinical utility," *Muscle & Nerve*, vol. 28, p. 1–17, Apr. 2003.

[90] A. Del Vecchio, A. Holobar, D. Falla, F. Felici, R. Enoka, and D. Farina, "Tutorial: Analysis of motor unit discharge characteristics from high-density surface EMG signals," *Journal of Electromyography and Kinesiology*, vol. 53, p. 102426, Aug. 2020.

[91] M. Gazzoni, D. Farina, and R. Merletti, "A new method for the extraction and classification of single motor unit action potentials from surface EMG signals," *Journal of Neuroscience Methods*, vol. 136, p. 165–177, July 2004.

[92] A. Holobar and D. Zazula, "Correlation-based decomposition of surface electromyograms at low contraction forces," *Medical & Biological Engineering & Computing*, vol. 42, p. 487–495, July 2004.

[93] A. Holobar, M. A. Minetto, A. Botter, F. Negro, and D. Farina, "Experimental Analysis of Accuracy in the Identification of Motor Unit Spike Trains From High-Density Surface EMG," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, p. 221–229, June 2010.

[94] F. Negro, S. Muceli, A. M. Castronovo, A. Holobar, and D. Farina, "Multi-channel intramuscular and surface EMG decomposition by convolutive blind source separation," *Journal of Neural Engineering*, vol. 13, p. 026027, Feb. 2016.

[95] M. Chen and P. Zhou, "A Novel Framework Based on FastICA for High Density Surface EMG Decomposition," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, p. 117–127, Jan. 2016.

[96] E. Martinez-Valdes, F. Negro, C. M. Laine, D. Falla, F. Mayer, and D. Farina, "Tracking motor units longitudinally across experimental sessions with high-density surface electromyography," *The Journal of Physiology*, vol. 595, p. 1479–1496, Feb. 2017.

[97] J. A. Gallego, J. L. Dideriksen, A. Holobar, J. Ibanez, V. Glaser, J. P. Romero, J. Benito-Leon, J. L. Pons, E. Rocon, and D. Farina, "The Phase Difference Between Neural Drives to Antagonist Muscles in Essential Tremor Is Associated with the Relative Strength of Supraspinal and Afferent Input," *Journal of Neuroscience*, vol. 35, p. 8925–8937, June 2015.

[98] A. Del Vecchio, C. M. Germer, L. A. Elias, Q. Fu, J. Fine, M. Santello, and D. Farina, "The human central nervous system transmits common synaptic inputs to distinct motor

neuron pools during non-synergistic digit actions," *The Journal of Physiology*, vol. 597, p. 5935–5948, Oct. 2019.

[99] A. Holobar and D. Farina, "Noninvasive Neural Interfacing With Wearable Muscle Sensors: Combining Convolutive Blind Source Separation Methods and Deep Learning Techniques for Neural Decoding," *IEEE Signal Processing Magazine*, vol. 38, p. 103–118, July 2021.

[100] D. F. Feeney, D. Mani, and R. M. Enoka, "Variability in common synaptic input to motor neurons modulates both force steadiness and pegboard time in young and older adults," *The Journal of Physiology*, vol. 596, p. 3793–3806, July 2018.

[101] R. M. Enoka, "Physiological validation of the decomposition of surface EMG signals," *Journal of Electromyography and Kinesiology*, vol. 46, p. 70–83, June 2019.

[102] A. Hyvärinen and E. Oja, "A Fast Fixed-Point Algorithm for Independent Component Analysis," *Neural Computation*, vol. 9, p. 1483–1492, Oct. 1997.

[103] L. Meng, Q. Chen, X. Jiang, X. Liu, J. Fan, C. Dai, and W. Chen, "Evaluation of decomposition parameters for high-density surface electromyogram using fast independent component analysis algorithm," *Biomedical Signal Processing and Control*, vol. 75, p. 103615, May 2022.

[104] J. Lundsberg, A. Björkman, N. Malesevic, and C. Antfolk, "Compressed spike-triggered averaging in iterative decomposition of surface EMG," *Computer Methods and Programs in Biomedicine*, vol. 228, p. 107250, Jan. 2023.

[105] A. Holobar and D. Zazula, "Multichannel Blind Source Separation Using Convolution Kernel Compensation," *IEEE Transactions on Signal Processing*, vol. 55, p. 4487–4496, Sept. 2007.

[106] A. Holobar and D. Zazula, "Gradient Convolution Kernel Compensation Applied to Surface Electromyograms," in *Independent Component Analysis and Signal Separation* (M. E. Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, eds.), (Berlin, Heidelberg), pp. 617–624, Springer Berlin Heidelberg, 2007.

[107] A. K. Clarke, S. F. Atashzar, A. D. Vecchio, D. Barsakcioglu, S. Muceli, P. Bentley, F. Urh, A. Holobar, and D. Farina, "Deep Learning for Robust Decomposition of High-Density Surface EMG Signals," *IEEE Transactions on Biomedical Engineering*, vol. 68, p. 526–534, Feb. 2021.

[108] Y. Wen, S. Avrillon, J. C. Hernandez-Pavon, S. J. Kim, F. Hug, and J. L. Pons, "A convolutional neural network to identify motor units from high-density surface electromyography signals in real time," *Journal of Neural Engineering*, vol. 18, p. 056003, Apr. 2021.

[109] A. Casolo, S. Maeo, T. G. Balshaw, M. B. Lanza, N. R. W. Martin, S. Nuccio, T. Moro, A. Paoli, F. Felici, N. Maffulli, B. Eskofier, T. M. Kinfe, J. P. Folland, D. Farina, and A. D. Vecchio, "Non-invasive estimation of muscle fibre size from high-density electromyography," *The Journal of Physiology*, vol. 601, p. 1831–1850, Mar. 2023.

[110] D. Farina, L. Arendt-Nielsen, R. Merletti, and T. Graven-Nielsen, "Assessment of single motor unit conduction velocity during sustained contractions of the tibialis anterior muscle with advanced spike triggered averaging," *Journal of Neuroscience Methods*, vol. 115, p. 1–12, Mar. 2002.

[111] A. Del Vecchio, A. Casolo, F. Negro, M. Scorcelletti, I. Bazzucchi, R. Enoka, F. Felici, and D. Farina, "The increase in muscle force after 4 weeks of strength training is mediated by adaptations in motor unit recruitment and rate coding," *The Journal of Physiology*, vol. 597, p. 1873–1887, Feb. 2019.

[112] X. Zhang and P. Zhou, "High-Density Myoelectric Pattern Recognition Toward Improved Stroke Rehabilitation," *IEEE Transactions on Biomedical Engineering*, vol. 59, p. 1649–1657, June 2012.

[113] C. Amma, T. Krings, J. Böer, and T. Schultz, "Advancing Muscle-Computer Interfaces with High-Density Electromyography," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, ACM, Apr. 2015.

[114] T. Sun, Q. Hu, J. Libby, and S. F. Atashzar, "Deep Heterogeneous Dilation of LSTM for Transient-Phase Gesture Prediction Through High-Density Electromyography: Towards Application in Neurorobotics," *IEEE Robotics and Automation Letters*, vol. 7, p. 2851–2858, Apr. 2022.

[115] S. Tam, M. Boukadoum, A. Campeau-Lecours, and B. Gosselin, "A Fully Embedded Adaptive Real-Time Hand Gesture Classifier Leveraging HD-sEMG and Deep Learning," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, p. 232–243, Apr. 2020.

[116] A. Moin, A. Zhou, A. Rahimi, S. Benatti, A. Menon, S. Tamakloe, J. Ting, N. Yamamoto, Y. Khan, F. Burghardt, L. Benini, A. C. Arias, and J. M. Rabaey, "An EMG Gesture Recognition System with Flexible High-Density Sensors and Brain-Inspired High-Dimensional Classifier," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, IEEE, 2018.

[117] R. Crepin, C. L. Fall, Q. Mascret, C. Gosselin, A. Campeau-Lecours, and B. Gosselin, "Real-Time Hand Motion Recognition Using sEMG Patterns Classification," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, July 2018.

[118] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, and J. Li, "Gesture recognition by instantaneous surface EMG images," *Scientific Reports*, vol. 6, Nov. 2016.

[119] U. Cote-Allard, C. L. Fall, A. Drouin, A. Campeau-Lecours, C. Gosselin, K. Glette, F. Laviolette, and B. Gosselin, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 27, pp. 760–771, Apr. 2019.

[120] F. Ryser, T. Butzer, J. P. Held, O. Lambercy, and R. Gassert, "Fully embedded myoelectric control for a wearable robotic hand orthosis," in *2017 International Conference on Rehabilitation Robotics (ICORR)*, IEEE, July 2017.

[121] A. Boyali, N. Hashimoto, and O. Matsumoto, "Hand posture and gesture recognition using MYO armband and spectral collaborative representation based classification," in *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*, IEEE, Oct. 2015.

[122] K. Sehat, S. M. Shokouhyan, N. K. Abdallah, and K. Khalaf, "Deep Network Optimization Using a Genetic Algorithm for Recognizing Hand Gestures via EMG Signals," *Preprints*, January 2023.

[123] H. A. Jaber, M. T. Rashid, and L. Fortuna, "Elicitation hybrid spatial features from HD-sEMG signals for robust classification of gestures in real-time," *Australian Journal of Electrical and Electronics Engineering*, vol. 18, p. 249–259, Aug. 2021.

[124] R. N. Khushaba, A. H. Al-Timemy, A. Al-Ani, and A. Al-Jumaily, "A Framework of Temporal-Spatial Descriptors-Based Feature Extraction for Improved Myoelectric Pattern Recognition," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, p. 1821–1831, Oct. 2017.

[125] Y. Du, W. Jin, W. Wei, Y. Hu, and W. Geng, "Surface EMG-Based Inter-Session Gesture Recognition Enhanced by Deep Domain Adaptation," *Sensors*, vol. 17, p. 458, Feb. 2017.

[126] W. Wei, Y. Wong, Y. Du, Y. Hu, M. Kankanhalli, and W. Geng, "A multi-stream convolutional neural network for sEMG-based gesture recognition in muscle-computer interface," *Pattern Recognition Letters*, vol. 119, p. 131–138, Mar. 2019.

[127] J. Chen, S. Bi, G. Zhang, and G. Cao, "High-Density Surface EMG-Based Gesture Recognition Using a 3D Convolutional Neural Network," *Sensors*, vol. 20, p. 1201, Feb. 2020.

[128] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, "A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition," *PLOS ONE*, vol. 13, p. e0206049, Oct. 2018.

[129] N. K. Karnam, S. R. Dubey, A. C. Turlapaty, and B. Gokaraju, "EMGHandNet: A hybrid CNN and Bi-LSTM architecture for hand activity classification using surface EMG signals," *Biocybernetics and Biomedical Engineering*, vol. 42, p. 325–340, Jan. 2022.

[130] T. Zhong, D. Li, J. Wang, J. Xu, Z. An, and Y. Zhu, "Fusion Learning for sEMG Recognition of Multiple Upper-Limb Rehabilitation Movements," *Sensors*, vol. 21, p. 5385, Aug. 2021.

[131] D. Bai, T. Liu, X. Han, and H. Yi, "Application Research on Optimization Algorithm of sEMG Gesture Recognition Based on Light CNN+LSTM Model," *Cyborg and Bionic Systems*, vol. 2021, Jan. 2021.

[132] Z. Yang, D. Jiang, Y. Sun, B. Tao, X. Tong, G. Jiang, M. Xu, J. Yun, Y. Liu, B. Chen, and J. Kong, "Dynamic Gesture Recognition Using Surface EMG Signals Based on Multi-Stream Residual Network," *Frontiers in Bioengineering and Biotechnology*, vol. 9, Oct. 2021.

[133] M. R. Islam, D. Massicotte, F. Nougarou, P. Massicotte, and W.-P. Zhu, "S-Convnet: A Shallow Convolutional Neural Network Architecture for Neuromuscular Activity Recognition Using Instantaneous High-Density Surface EMG Images," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, IEEE, July 2020.

[134] B. Leelakittisin, M. Trakulruangroj, S. Sangnark, T. Wilaiprasitporn, and T. Sudhawiyangkul, "Enhanced Lightweight CNN Using Joint Classification With Averaging Probability for sEMG-Based Subject-Independent Hand Gesture Recognition," *IEEE Sensors Journal*, vol. 23, no. 17, pp. 20348–20356, 2023.

[135] M. R. Islam, D. Massicotte, and W.-P. Zhu, "All-ConvNet: A Lightweight All CNN for Neuromuscular Activity Recognition Using Instantaneous High-Density Surface EMG Images," in *2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, IEEE, May 2020.

[136] X. Peng, X. Zhou, H. Zhu, Z. Ke, and C. Pan, "MSFF-Net: Multi-Stream Feature Fusion Network for surface electromyography gesture recognition," *PLOS ONE*, vol. 17, p. e0276436, Nov. 2022.

[137] M. R. Islam, D. Massicotte, F. Nougarou, and W.-P. Zhu, "HOG and Pairwise SVMs for Neuromuscular Activity Recognition Using Instantaneous HD-sEMG Images," in *2018 16th IEEE International New Circuits and Systems Conference (NEWCAS)*, IEEE, June 2018.

[138] N. Celadon, S. Došen, I. Binder, P. Ariano, and D. Farina, "Proportional estimation of finger movements from high-density surface electromyography," *Journal of NeuroEngineering and Rehabilitation*, vol. 13, Aug. 2016.

[139] A. Aranceta-Garza and B. A. Conway, "Differentiating Variations in Thumb Position From Recordings of the Surface Electromyogram in Adults Performing Static Grips, a Proof of Concept Study," *Frontiers in Bioengineering and Biotechnology*, vol. 7, May 2019.

[140] G. Hajian, E. Morin, and A. Etemad, "Convolutional Neural Network Approach for Elbow Torque Estimation during Quasi-dynamic and Dynamic Contractions," in *2021 43rd Annual*

International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), IEEE, Nov. 2021.

[141] K. Fujikawa, S. Shirafuji, B. Su, E. Piovanelli, and J. Ota, "Estimation of fingertip forces using high-density surface electromyography," in *2017 International Symposium on Micro-NanoMechatronics and Human Science (MHS)*, pp. 1–5, 2017.

[142] X. Jiang, K. Nazarpour, and C. Dai, "Explainable and Robust Deep Forests for EMG-Force Modeling," *IEEE Journal of Biomedical and Health Informatics*, vol. 27, p. 2841–2852, June 2023.

[143] M. Houston, A. Wu, and Y. Zhang, "Optimizing Input for Gesture Recognition using Convolutional Networks on HD-sEMG Instantaneous Images," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, IEEE, Nov. 2021.

[144] I. F. Ghalyan, Z. M. Abouelenin, and V. Kapila, "Gaussian Filtering of EMG Signals for Improved Hand Gesture Classification," in *2018 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*, pp. 1–6, 2018.

[145] O. W. Samuel, X. Li, Y. Geng, M. G. Asogbon, P. Fang, Z. Huang, and G. Li, "Resolving the adverse impact of mobility on myoelectric pattern recognition in upper-limb multifunctional prostheses," *Computers in Biology and Medicine*, vol. 90, p. 76–87, Nov. 2017.

[146] T. W. Beck, J. M. DeFreitas, J. T. Cramer, and J. R. Stout, "A comparison of adaptive and notch filtering for removing electromagnetic noise from monopolar surface electromyographic signals," *Physiological Measurement*, vol. 30, p. 353–361, Feb. 2009.

[147] Arul, "A Review on Noises in EMG Signal and its Removal," 2017.

[148] T. Soni and N. Rathor, "Removal of High Density Impulse Noise using Efficient Median Filter for Digital Image," *International Journal of Computer Applications*, vol. 115, p. 25–31, Apr. 2015.

[149] Y. A. Jarrah, M. G. Asogbon, O. W. Samuel, X. Wang, M. Zhu, E. Nsugbe, S. Chen, and G. Li, "High-density surface EMG signal quality enhancement via optimized filtering technique for amputees' motion intent characterization towards intuitive prostheses control," *Biomedical Signal Processing and Control*, vol. 74, p. 103497, Apr. 2022.

[150] A. Moin, A. Zhou, A. Rahimi, S. Benatti, A. Menon, S. Tamakloe, J. Ting, N. Yamamoto, Y. Khan, F. Burghardt, L. Benini, A. C. Arias, and J. M. Rabaey, "An EMG Gesture Recognition System with Flexible High-Density Sensors and Brain-Inspired High-Dimensional Classifier," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, IEEE, 2018.

[151] P. Kanerva, "Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors," *Cognitive Computation*, vol. 1, p. 139–159, Jan. 2009.

[152] "Analysis of Robust Implementation of an EMG Pattern Recognition based Control," in *Proceedings of the International Conference on Bio-inspired Systems and Signal Processing*, SCITEPRESS - Science and and Technology Publications, 2014.

[153] E. Farago and A. D. C. Chan, "A Pilot Study on Quantifying Signal Quality in High-Density Surface Electromyography," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, p. 1–9, 2023.

[154] H. Zhang, C. Wang, W. Gou, L. Guo, and C. Lin, "DFNN-based Gesture Recognition with the Shift and Damage of the HD-sEMG Electrodes," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, IEEE, Dec. 2019.

[155] L. Wu, X. Zhang, K. Wang, X. Chen, and X. Chen, "Improved High-Density Myoelectric Pattern Recognition Control Against Electrode Shift Using Data Augmentation and Dilated Convolutional Neural Network," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, p. 2637–2646, Dec. 2020.

[156] R. Hu, X. Chen, X. Zhang, and X. Chen, "Adaptive Electrode Calibration Method Based on Muscle Core Activation Regions and Its Application in Myoelectric Pattern Recognition," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, p. 11–20, 2021.

[157] A. Stango, F. Negro, and D. Farina, "Spatial Correlation of High Density EMG Signals Provides Features Robust to Electrode Number and Shift in Pattern Recognition for Myocontrol," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, p. 189–198, Mar. 2015.

[158] A. Ameri, M. A. Akhaee, E. Scheme, and K. Englehart, "A deep transfer learning approach to reducing the effect of electrode shift in EMG pattern recognition-based control," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 28, pp. 370–379, Feb. 2020.

[159] T. Sun, J. Libby, J. Rizzo, and S. F. Atashzar, "Deep Augmentation for Electrode Shift Compensation in Transient High-density sEMG: Towards Application in Neurorobotics," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6148–6153, 2022.

[160] B. Xue, L. Wu, A. Liu, X. Zhang, X. Chen, and X. Chen, "Detecting the universal adversarial perturbations on high-density sEMG signals," *Computers in Biology and Medicine*, vol. 149, p. 105978, Oct. 2022.

[161] A. E. Olsson, P. Sager, E. Andersson, A. Björkman, N. Malešević, and C. Antfolk, "Extraction of Multi-Labelled Movement Information from the Raw HD-sEMG Image with Time-Domain Depth," *Scientific Reports*, vol. 9, May 2019.

[162] Y. Zhou, C. Chen, M. Cheng, Y. Alshahrani, S. Franovic, E. Lau, G. Xu, G. Ni, J. M. Cavanaugh, S. Muh, and S. Lemos, "Comparison of machine learning methods in sEMG signal processing for shoulder motion recognition," *Biomedical Signal Processing and Control*, vol. 68, p. 102577, July 2021.

[163] R. Andrei, "Investigating the Performance of Deep Learning Algorithms for Muscle Activation On/Off-Set Detection in Horse Surface Electromyography (sEMG) Data," in *39th Twente Student Conference on IT*, July 2023.

[164] X. Chen, X. Zhang, X. Chen, and X. Chen, "Decoding Silent Speech Based on High-Density Surface Electromyogram Using Spatiotemporal Neural Network," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 31, p. 2069–2078, 2023.

[165] D. Zhang, Z. Chen, L. Xiao, B. Zhu, R. Wu, C. Ou, Y. Ma, L. Xie, and H. Jiang, "Stretchable and durable HD-sEMG electrodes for accurate recognition of swallowing activities on complex epidermal surfaces," *Microsystems & Nanoengineering*, vol. 9, Sept. 2023.

[166] M. Montazerin, E. Rahimian, F. Naderkhani, S. F. Atashzar, H. Alinejad-Rokny, and A. Mohammadi, "HYDRA-HGR: A Hybrid Transformer-Based Architecture for Fusion of Macroscopic and Microscopic Neural Drive Information," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, June 2023.

[167] M. Montazerin, E. Rahimian, F. Naderkhani, S. F. Atashzar, S. Yanushkevich, and A. Mohammadi, "Transformer-based hand gesture recognition from instantaneous to fused neural decomposition of high-density EMG signals," *Scientific Reports*, vol. 13, July 2023.

[168] Y. Zhao, X. Zhang, X. Li, H. Zhao, X. Chen, X. Chen, and X. Gao, "Decoding finger movement patterns from microscopic neural drive information based on deep learning," *Medical Engineering & Physics*, vol. 104, p. 103797, June 2022.

[169] C. Chen, Y. Yu, S. Ma, X. Sheng, C. Lin, D. Farina, and X. Zhu, "Hand gesture recognition based on motor unit spike trains decoded from high-density electromyography," *Biomedical Signal Processing and Control*, vol. 55, p. 101637, Jan. 2020.

[170] Y. Xie, P. Wang, W. Chen, W. Wang, and L. Liu, "A neural-based approach to hand gesture recognition with HD-sEMG," in *2022 41st Chinese Control Conference (CCC)*, pp. 3144–3149, 2022.

[171] K. Nizamis, N. H. M. Rijken, R. van Middelaar, J. Neto, B. F. J. M. Koopman, and M. Sartori, "Characterization of Forearm Muscle Activation in Duchenne Muscular Dystrophy

via High-Density Electromyography: A Case Study on the Implications for Myoelectric Control," *Frontiers in Neurology*, vol. 11, Apr. 2020.

[172] X. Li, X. Zhang, X. Chen, X. Chen, and A. Liu, "Cross-user gesture recognition from sEMG signals using an optimal transport assisted student-teacher framework," *Computers in Biology and Medicine*, vol. 165, p. 107327, Oct. 2023.

[173] M. Nikolic, *Detailed analysis of clinical electromyography signals: EMG decomposition, findings and firing pattern analysis in controls and patients with myopathy and amytrophic lateral sclerosis*. PhD thesis, University of Copenhagen, the Faculty of Health Science, 2001.

[174] E. TUNCER and E. DOĞRU BOLAT, "Classification of Myopathy and Normal Electromyogram (EMG) Data with a New Deep Learning Architecture," *Balkan Journal of Electrical and Computer Engineering*, vol. 11, p. 267–276, Aug. 2023.

[175] M. S. Gaso, S. Cankurt, and A. Subasi, "Electromyography Signal Classification Using Deep Learning," in *2021 16th International Conference on Electronics Computer and Computation (ICECCO)*, IEEE, Nov. 2021.

[176] A. R. Verma and B. Gupta, "Detecting Neuromuscular Disorders Using EMG Signals Based on TQWT Features," *Augmented Human Research*, vol. 5, Nov. 2019.

[177] A. Belkhou, A. Jbari, and O. E. Badlaoui, "A computer-aided-diagnosis system for neuromuscular diseases using Mel frequency Cepstral coefficients," *Scientific African*, vol. 13, p. e00904, Sept. 2021.

[178] A. Mokdad, S. M. E. A. Debbal, and F. Meziani, "Diagnosis of amyotrophic lateral sclerosis (ALS) disorders based on electromyogram (EMG) signal analysis and feature selection," *Polish Journal of Medical Physics and Engineering*, vol. 26, p. 155–160, Sept. 2020.

[179] E. Bitar, F. ElSayyadi, and A. Diab, "Duchenne muscular dystrophy detection using EMG analysis and machine learning," in *2023 Seventh International Conference on Advances in Biomedical Engineering (ICABME)*, IEEE, Oct. 2023.

[180] S. M. Sid'El Moctar, A. Diab, I. Rida, K. Kinugawa, and S. Boudaoud, "Active aging prediction from muscle electrical activity using HD-sEMG signals and machine learning," in *2023 IEEE 36th International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 734–737, 2023.

[181] F. Urh, D. Strnad, A. Clarke, D. Farina, and A. Holobar, "On the Selection of Neural Network Architecture for Supervised Motor Unit Identification from High-Density Surface EMG," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, IEEE, July 2020.

[182] F. Urh, D. Strnad, A. Clarke, D. Farina, and A. Holobar, "On the Need for Spatial Whitening of High-Density Surface Electromyograms in Motor Unit Identification by Neural Networks," in *IFMBE Proceedings*, p. 915–922, Springer International Publishing, Nov. 2020.

[183] U. Rashid, I. K. Niazi, N. Signal, D. Farina, and D. Taylor, "Optimal automatic detection of muscle activation intervals," *Journal of Electromyography and Kinesiology*, vol. 48, p. 103–111, Oct. 2019.

[184] X. Jiang, X. Liu, J. Fan, X. Ye, C. Dai, E. A. Clancy, M. Akay, and W. Chen, "Open Access Dataset, Toolbox and Benchmark Processing Results of High-Density Surface Electromyogram Recordings," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, p. 1035–1046, 2021.

[185] C. Dai, Y. Cao, and X. Hu, "Prediction of Individual Finger Forces Based on Decoded Motoneuron Activities," *Annals of Biomedical Engineering*, vol. 47, p. 1357–1368, Mar. 2019.

[186] G. Valli, P. Ritsche, A. Casolo, F. Negro, and G. De Vito, "Tutorial: Analysis of central and peripheral motor unit properties from decomposed High-Density surface EMG signals with openhdemg," *Journal of Electromyography and Kinesiology*, vol. 74, p. 102850, Feb. 2024.

[187] M. B. Jensen, J. A. B. Manresa, K. S. Frahm, and O. K. Andersen, "Analysis of muscle fiber conduction velocity enables reliable detection of surface EMG crosstalk during detection of nociceptive withdrawal reflexes," *BMC Neuroscience*, vol. 14, Mar. 2013.

[188] S. Y. Shirazi, "hdEMG-Decompostion (github.com/neuromechanist/hdEMG-Decomposition/tag/0.1)," 2022. Retrieved January 15, 2024.

[189] M. Nikolic and C. Krarup, "EMGTools, an Adaptive and Versatile Tool for Detailed EMG Analysis," *IEEE Transactions on Biomedical Engineering*, vol. 58, p. 2707–2718, Oct. 2011.

[190] S. Avrillon, F. Hug, C. Gibbs, and D. Farina, "Tutorial on MUedit: An open-source software for identifying and analysing the discharge timing of motor units from electromyographic signals," *Cold Spring Harbor Laboratory*, July 2023.

[191] A. Alosta and J. Djärf, "sEMG-Sim version: 0.1." https://github.com/Josef-Djarf/sEMG-Sim, 2024. Accessed: 2024-04-10.

[192] W. Zhang, Z. Yang, H. Li, D. Huang, L. Wang, Y. Wei, L. Zhang, L. Ma, H. Feng, J. Pan, Y. Guo, and P. Chan, "Multimodal Data for the Detection of Freezing of Gait in Parkinson's Disease," *Scientific Data*, vol. 9, Oct. 2022.

[193] C. Carreiras, A. P. Alves, A. Lourenço, F. Canento, H. Silva, A. Fred, *et al.*, "BioSPPy: Biosignal Processing in Python," 2015. [Online; accessed 10 April 2024].

[194] P. Bonato, T. D'Alessio, and M. Knaflitz, "A statistical method for the measurement of muscle activation intervals from surface myoelectric signal during gait," *IEEE Transactions on Biomedical Engineering*, vol. 45, p. 287–299, Mar. 1998.

[195] M. Lidierth, "A computer based method for automated measurement of the periods of muscular activity from an EMG and its application to locomotor EMGs," *Electroencephalography and Clinical Neurophysiology*, vol. 64, p. 378–380, Oct. 1986.

[196] Abbink, V. D. Bilt, and V. D. Glas, "Detection of onset and termination of muscle activity in surface electromyograms," *Journal of Oral Rehabilitation*, vol. 25, p. 365–369, May 1998.

[197] A. Londral, H. Silva, N. Nunes, M. Carvalho, and L. Azevedo, "A wireless user- computer interface to explore various sources of biosignals and visual biofeedback for severe motor impairment," *Journal of Accessibility and Design for All*, vol. 3, pp. 118–134, Nov 2013.

[198] H. Silva, R. Scherer, J. Sousa, and A. Londral, "Towards Improving the Usability of Electromyographic Interfaces," in *Converging Clinical and Engineering Research on Neurorehabilitation, Biosystems & Biorobotics*, vol. 1, p. 437–441, Springer Berlin Heidelberg, 2013.

[199] S. Solnik, P. Rider, K. Steinweg, P. DeVita, and T. Hortobágyi, "Teager–Kaiser energy operator signal conditioning improves EMG onset detection," *European Journal of Applied Physiology*, vol. 110, p. 489–498, June 2010.

[200] P. Hodges, "A comparison of computer-based methods for the determination of onset of muscle contraction using electromyography," *Electroencephalography and Clinical Neurophysiology*, vol. 101, p. 511–519, Dec. 1996.

[201] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. O'Leary, "PyWavelets: A Python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, p. 1237, Apr. 2019.

[202] J. Muradeli, "ssqueezepy," *GitHub. Note: https://github.com/OverLordGoldDragon/ssqueezepy/*, 2020.

[203] D. Farina, E. Fortunato, and R. Merletti, "Noninvasive estimation of motor unit conduction velocity distribution using linear electrode arrays," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 3, pp. 380–388, 2000.

[204] C. Huang, M. Chen, Y. Zhang, S. Li, C. S. Klein, and P. Zhou, "A Novel Muscle Innervation Zone Estimation Method Using Monopolar High Density Surface Electromyography," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 31, p. 22–30, 2023.

[205] J. Rodriguez-Falces, "A new method for the localization of the innervation zone based on monopolar surface-detected potentials," *Journal of Electromyography and Kinesiology*, vol. 35, p. 47–60, Aug. 2017.

[206] E. Formento, P. Botros, and J. M. Carmena, "Skilled independent control of individual motor units via a non-invasive neuromuscular–machine interface," *Journal of Neural Engineering*, vol. 18, p. 066019, Nov. 2021.

[207] carmenalab, "emgdecomp." https://github.com/carmenalab/emgdecomp, 2024. Accessed: 2024-05-21.

[208] D. King, J. Ortega, R. Rudyak, and R. Sivanandam, "EMGdecomPy." https://github.com/The-Motor-Unit/EMGdecomPy, 2022. Accessed: 2024-05-21.

[209] R. Saravanan and P. Sujatha, "A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification," *Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 945–949, 2018.

[210] H. Tankisi, D. Burke, L. Cui, *et al.*, "Standards of instrumentation of EMG," *Clinical neurophysiology : official journal of the International Federation of Clinical Neurophysiology*, vol. 131, no. 1, pp. 243–258, 2020.

[211] C. J. D. Luca, L. D. Gilmore, M. Kuznetsov, and S. H. Roy, "Filtering the surface EMG signal: Movement artifact and baseline noise contamination," *Journal of biomechanics*, vol. 43, no. 8, p. 1573–1579, 2010.

[212] A. van Boxtel, "Optimal signal bandwidth for the recording of surface EMG activity of facial, jaw, oral, and neck muscles," *Psychophysiology*, vol. 38, no. 1, p. 22–34, 2001.

[213] Ţarălungă Dragoș Daniel and M. N. (Ungureanu), "Cancelling Harmonic Power Line Interference in Biopotentials," *InTech*, 2018.

[214] H. Azami, B. Bozorgtabar, and M. Shiroie, "Automatic Signal Segmentation using the Fractal Dimension and Weighted Moving Average Filter," *International Journal of Electrical & Computer Sciences IJECS-IJENS*, vol. 11, no. 6, 2011.

[215] L. Rosencrance, "DEFINITION feature engineering," *TechTarget*, 2021.

[216] F. Naim, M. Mustafa, N. Sulaiman, and N. A. A. Rahman, "The Study of Time Domain Features of EMG Signals for Detecting Driver's Drowsiness," *Recent Trends in Mechatronics Towards Industry 4.0*, p. 427–438, 2020.

[217] P. Welch, "The use of the fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 70 – 73, 1967.

[218] T. N. S. T. Zawawi, A. R. Abdullah, W. T. Jin, R. Sudirman, and N. M. Saad, "Electromyography Signal Analysis Using Time and Frequency Domain for Health Screening System Task," *International Journal of Human & Technology Interaction*, vol. 2, no. 1, 2018.

[219] E. Sejdić, I. Djurović, and L. Stanković, "Fractional Fourier transform as a signal processing tool: An overview of recent developments," *Signal Processing*, vol. 91, no. 6, pp. 1351–1369, 2011.

[220] S. Maddali, "frft," *GitHub repository*, 2022.

[221] L. A. C. Kallenberg and H. J. Hermens, "Behaviour of motor unit action potential rate, estimated from surface EMG, as a measure of muscle activation level," *Journal of neuroengineering and rehabilitation*, vol. 3, no. 15, 2006.

[222] K. Nazarpour, A. H. Al-Timemy, G. Bugmann, and A. Jackson, "A note on the probability distribution function of the surface electromyogram signal," *Brain Research Bulletin*, vol. 90, no. 100, 2013.

[223] D. C. Toledo-Pérez, J. Rodríguez-Reséndiz, and R. A. Gómez-Loenzo, "A Study of Computing Zero Crossing Methods and an Improved Proposal for EMG Signals," *IEEE Access*, vol. 8, 2020.

[224] A. Phinyomark, C. Limsakul, and P. Phukpattaranont, "EMG feature extraction for tolerance of white Gaussian noise," *International Workshop and Symposium Science Technology*, 2008.

[225] M. Mouzé-Amady and F. Horwat, "Evaluation of Hjorth parameters in forearm surface EMG analysis during an occupational repetitive task," *Electroencephalography and Clinical Neurophysiology/Electromyography and Motor Control*, vol. 101, no. 2, 1996.

[226] K. T. Q. Dang, H. L. Minh, H. N. Thanh, and T. V. Van, "Analyzing surface EMG signals to determine relationship between jaw imbalance and arm strength loss," *BioMedical Engineering OnLine*, vol. 11, no. 55, 2012.

[227] X. Zhang and P. Zhou, "Sample Entropy Analysis of Surface EMG for Improved Muscle Activity Onset Detection against Spurious Background Spikes," *Journal of electromyography and kinesiology : official journal of the International Society of Electrophysiological Kinesiology*, vol. 22, no. 6, pp. 901–907, 2012.

[228] B. Kim, J. Kim, H. Chae, D. Yoon, and J. W. Choi, "Deep neural network-based automatic modulation classification technique," *International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 579–582, 2016.

[229] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with Noisy Student improves ImageNet classification," *Google Research, Brain Team, Carnegie Mellon University*, 2019.

[230] SudoPurge, "4 Reasons Your Machine Learning Model Is Underperforming," *Towards Data Science*, 2021.

[231] C. Güzelgün, "A Deep Dive into Spark's Univariate Feature Selector," *Insider Engineering*, 2022.

[232] A. Gholamy, V. Kreinovich, and O. Kosheleva, "Why 70/30 or 80/20 Relation Between Training and Testing Sets: A Pedagogical Explanation," *Departmental Technical Reports (CS)*, vol. 1209, 2018.

[233] L. A. Jeni, J. F. Cohn, and F. De La Torre, "Facing Imbalanced Data–Recommendations for the Use of Performance Metrics," in *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, IEEE, Sept. 2013.

[234] N. Instruments, "Filtering and Downsampling (Advanced Signal Processing Toolkit or Control Design and Simulation Module)," *National Instruments Corp*, 2018.

[235] A. de Cheveigné and I. Nelken, "Filters: When, Why, and How (Not) to Use Them," *Neuron*, vol. 102, no. 2, pp. 280–293, 2019.

[236] M. Pilz and S. Parolai, "Tapering of windowed time series," *New Manual of Seismological Observatory Practice 2 (NMSOP-2)*, 2012.

[237] M. A. Ozdemir, D. H. Kisa, O. Guren, and A. Akan, "Hand gesture classification using time–frequency images and transfer learning based on CNN," *Biomedical Signal Processing and Control*, vol. 77, p. 103787, Aug. 2022.

[238] X. Liu, M. Zhang, J. Wang, X. Wang, T. Liang, J. Li, P. Xiong, and X. Liu, "Gesture recognition of continuous wavelet transform and deep convolution attention network," *Mathematical Biosciences and Engineering*, vol. 20, no. 6, p. 11139–11154, 2023.

[239] A.-C. Tsai, J.-J. Luh, and T.-T. Lin, "A novel STFT-ranking feature of multi-channel EMG for motion pattern recognition," *Expert Systems with Applications*, vol. 42, p. 3327–3341, May 2015.

[240] B. L. Day, J. C. Rothwell, P. D. Thompson, A. M. De Noordhout, K. Nakashima, K. Shannon, and C. D. Marsden, "Delay in the execution of voluntary movement by electrical or magnetic brain stimulation in intact man: Evidence for the storage of motor programs in the brain," *Brain*, vol. 112, no. 3, p. 649–663, 1989.

[241] I. Campanini, A. Merlo, *et al.*, "Effect of electrode location on EMG signal envelope in leg muscles during gait," *Journal of Electromyography and Kinesiology*, vol. 17, no. 4, pp. 515–526, 2007.

[242] N. Tran, "Fundamental of The Chi Square in Statistics," *Medium*, 2019.