Identifying Patterns of Physical Activity in Musculoskeletal Disease

A thesis submitted to the University of Manchester for the degree of Doctor of Philosophy in the Faculty of Biology, Medicine and Health.

2021

KATY S WEIHRICH

School of Biological Sciences,

Division of Musculoskeletal & Dermatological Sciences,

Arthritis Research UK Centre for Epidemiology

>Blank page<

Contents

Word Count: 74,324

List of Tables

List of Figures

Abbreviations

AC ..... *normalised cross-AutoCorrelation*
ADL ..... *Activities of Daily Living*
AGDA ..... *Aston Gait Detection Algorithm*
AGG ..... *AGGravating activity*
ATR ..... *Acceleration Transient Rate*
BMI ..... *Body Mass Index*
CDBN ..... *Convolutional Deep Belief Networks*
CI ..... *Confidence Interval*
CWT ..... *Complex Wavelet Transform*
DAS ..... *Disease Activity Score*
DASA ..... *Dominant Axis Selection Approach*
-DL ..... *Deep Learning features*
DMF ..... *Dichotomous Mapped Forest*
DWT ..... *Discrete Wavelet Transform*
DyF ..... *DailY Function*
EMG ..... *ElectroMyoGraphy*
FSI ..... *Fractal Scaling Index*
GB ..... *GigaByte*
GDA ..... *Gait Detection Algorithm*
GRF ..... *Ground Reaction Forces*
h ..... *Hour/s*
HMM ..... *Hidden Markov Model*
iCSF ..... *Interactive Computational Shared Facility*
ID ..... *participant/user IDentification*
IMP ..... *IMPortant activity*
IMU ..... *Internal Measurement Units*
InResponse ..... *INsoles in RESPONSE*
IPA ..... *Initial Peak Amplitude*
KB ..... *KiloByte*
KOALAP ..... *Knee OsteoArthritis: Linking Activity and Pain*
KOOS ..... *Knee injury and Osteoarthritis Outcome Score*
KSS ..... *Knee Society Score*
LIA ..... *Light-Intensity Activity*
MEMS ..... *Micro-Electro-Mechanical Systems*
MET ..... *Metabolic Equivalent of Task*
-metric ..... *METRIC learning mapping*
mHAQ ..... *Modified Health Assessment Questionnaire*
MIA ..... *Moderate-Intensity Activity*
msec ..... *MilliSECond / MilliSEConds*
MSK ..... *MusculoSKeletal conditions*
MTA ..... *Material Transfer Agreement*
MvM ..... *MoVing Mean filter*
MvSTD ..... *MoVing STandard Deviation*
NASC ..... *Normalised Autocorrelation Step Counting*
NHS ..... *National Health Service*
NSAID ..... *Non-Steroidal Anti-Inflammatory Drugs*
OA ..... *OsteoArthritis*
OAM ..... *morning pain score – pain Overall, morning (AM)*
obs ..... *OBServations / samples*
OPM ..... *afternoon pain - pain Overall, afternoon (PM)*
PCA ..... *Principal Component Analysis*
PLOE ..... *Placement, Location and Orientation Evaluation method*

p-Welch ..... *P-Welch spectrum analysis*
QoL ..... *Quality Of Life*
QUASAR ..... *QUAlity of Life, Sleep and rheumatoid ArthRitis*
RA ..... *Rheumatoid Arthritis*
RAM ..... *Random Access Memory*
RAT ..... *Ratio of Acceleration Transmission*
RB-SVM ..... *Radial Basis Support Vector Machines*
RF ..... *Random Forests*
RMD ..... *Rheumatic and Musculoskeletal Diseases*
RMS ..... *Root Mean Square*
ROC ..... *Receiver Operating Characteristic*
ROM ..... *Range Of Motion*
SC ..... *Step Count*
sec ..... *SECond / SEConds*
SI ..... *Symmetry Indices*
SPE ..... *Step Parameter Extraction algorithm*
SRDS ..... *Self-Recorded DataSet*
STD ..... *STandard Deviation*
STD-threshold ..... *STandard Deviation-THRESHOLD*
STFT ..... *Short-Term Fourier Transformation*
Switching AR ..... *SWITCHING AutoRegressive model*
TB ..... *TeraByte*
TENS ..... *Transcutaneous Electrical Nerve Stimulation*
TKA ..... *Total Knee Arthroplasty*
UKA ..... *Unicompartmental Knee Arthroplasty*
VIA ..... *Vigorous-Intensity Activity*
WHO ..... *World Health Organisation*
WOMAC ..... *Western Ontario and Mcmaster universities Arthritis Index*

Abstract

We aimed to find a novel approach for diagnosing functionality in patients with rheumatic and musculoskeletal diseases (RMDs). Accelerometer sensors embedded in consumer devices, such as Fitbits and smartwatches, open new possibilities for continuously tracking patients' symptoms. This can be achieved by developing apps that enable active tracking of functions by requesting questionnaires to be filled out routinely and by developing algorithms that can passively take accelerometer data and analyse patient activity patterns. The Knee Osteoarthritis: Linking Activity and Pain (KOALAP) study provides us with an example of such a potential diagnostic tool. Our goal was to develop a gait detection algorithm (GDA) to separate walking behaviours from non-walking behaviours continuously recorded by accelerometer sensors located on the wrist in free-living conditions. Our GDA performed with 97.20 % accuracy on a self-recorded, labelled dataset (SRDS). However, the translatability of this accuracy onto other datasets needs to be further explored, since the data was only recorded on one young and healthy participant. Furthermore, we aimed to extract step parameters to track a patient's physical function and developed a step parameter extraction algorithm (SPE) for this purpose. We identified the following parameters of interest: *Walking Episode Length* in minutes, *Acceleration Range* of gait cycles in m/sec^2, *Step Rate* in minutes, *Symmetry of Step Time* in seconds between the left and right step cycle and *Variance of the Acceleration Range* in m/sec^2. To assist the development of the algorithms and to examine the impact of different walking behaviours on accelerometer data, we used the SRDS. We identified two major ways in which the wrist location of the accelerometer influences the record accelerometer data. Walking can be divided into behaviours with free arm *swing* and *stiff* behaviours where the arm is fixed in relation to the body. Stiff behaviours tend to be noisier but also resemble data collected on the leg more closely. Swing behaviours tend to exhibit larger acceleration ranges. Applying our algorithms on the KOALAP dataset we found that we were unable to relate patients' step parameter values directionality to their pain reports. We did, however, find occurrences of significant correlations with some pain scores depending on the participant. We conclude that the step parameter we extracted could help track patients' performance over time, however, the effectiveness and the parameters used would need to be adjusted to the individual patient. To investigate the usefulness of the step parameter further, we applied our algorithms on the accelerometer dataset of the UK BioBank study. The parameters mean and standard deviation differed only slightly between groups of participants with different RMDs and selected control groups. However, conducting a paired t-test between RMD patients and their matched controls revealed a significant difference for most step parameters with mostly small effect sizes. We conclude that age, sex and Townsend scores (which we used to match the control groups) have a larger effect on individual participant step parameters than their RMD. However, if these patients' characteristics are accounted for we do find our step parameters to reflect patients' functionality in large datasets.

Declaration

No portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright Statement

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the “Copyright”) and s/he has given the University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made

iii. The ownership of certain Copyright, patents, designs, trademarks and other intellectual property (the “Intellectual Property”) and any reproductions of copyright works in the thesis, for example graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=24420), in any relevant Thesis restriction declarations deposited in the University Library, the University Library’s regulations (see http://www.library.manchester.ac.uk/about/regulations/) and in the University’s policy on Presentation of Theses.

Acknowledgement

I would like to thank my supervisors Dr Mark Lunt, Dr Max A. Little, Dr Suzanne Verstappen and Prof William Dixon.

Within the University of Manchester, I would like to thank the members working on the KOALAP project, in particular Dr Anna L. Beukenhorst, who helped me to get integrated into the project and with accessing the data. Furthermore, I received coaching in searching literature systematically from our librarian Mary Ingram, technical support from our Senior Software Engineer Owen Stewart and help to coordinate my supervisors from the Office Administrator Carol Sewell. I also received support navigating the Interactive Computational Shared Facility (iCSF, also known as Incline) system from Dr Stephanie Ling. Michael Cook and Dr Alex Casson helped me with accessing and understanding the UK BioBank procedures. When it came to identifying walking behaviours through accelerometry, Dr Alexander Oldroyd also shared ideas his ideas with me.

I would like to thank the Aston University research group that helped brainstorm ideas and provided a supportive environment for me to work in. In particular, I would like to thank Dr Reham Badawy, Dr Yordan P. Raykov and Dr Yazan Qarout. Dr Badawy worked with me on the recording of the self-recorded dataset. We brainstormed the walking behaviours chosen for the recordings together and she did the documentation for the staircase and coffee trials. Dr Qarout also volunteered for data collection. Dr Raykov provided me with a lot of guidance in navigating the different algorithms and mathematical concepts used to identify gait patterns in accelerometer data. He also provided me with various Matlab codes that I was able to test on my data.

At last, I would like to thank my parents, family and friends that supported me during the time of my PhD. My parents (Dr Anette Weihrich and Dipl.-Ing. Kay E. Weihrich) provided me with a safe workplace and much needed mental support during the corona pandemic. My father even volunteered for a data recording trial. In general, all my friends and Mareike Christensen in particular, tolerated me talking with them about problems and ideas I had for the gait detection and step parameter extraction, even if they did not understand a word I was saying. I am grateful for Olliver Christensen's proofreading of the KOALAP and UK BioBank chapters and lots of thanks in particular to Kieran Jones who proofread my whole thesis for spelling and grammar mistakes before submission.

Preface

In 2016 I completed a Bachelor's degree in "Cognitive Neuroscience and Psychology" followed by a Master's degree in "Neuroimaging for Clinical and Cognitive Neuroscience", both at the University of Manchester. My objectives in undertaking a PhD were to increase my statistical and computer science knowledge. I was also motivated to work on a project that requires analysing and working with large datasets and to build on my previous knowledge of programming in Matlab.

During the first year of my PhD, I had to make myself familiar with the fields of RMDs, a medical field that I had no prior experience with. Simultaneously, I looked into the different ongoing projects in the CfMR to obtain a dataset that I could analyse for my PhDs objectives. In the end, I selected the KOALAP study, since it provided both continuous recorded accelerometer data and records of self-reported pain scores. The raw accelerometer data was downloaded in February 2018 and the following months were mostly spent wrangling the data downloaded from the Google servers into a standard format. I also started looking into bioinformatics principles and machine learning processes in particular. In June 2018, I conducted the first attempts of data analysis on the KOALAP dataset. I applied pre-processing steps to the transformed raw data and even ran a short-term Fourier transformation (STFT) and applied a switching autoregressive model (Switching AR).

In September 2018, I moved to Birmingham to spend a year at Aston University with my supervisor Dr Little. There, I was introduced to various data analysis approaches (feature thresholding, p-Welch spectrum analysis (p-Welch), normalised cross-autocorrelation (AC), normalised autocorrelation step counting (NASC), autoregressive Hidden Markov Model (HMM) and K-means clustering). However, to assess the effectiveness of these methods, I needed labelled gait data. Between February and July 2019, we recorded our own dataset of walking and non-walking behaviours. The finished dataset was then used to test the above-named methods and to develop a GDA for wrist-worn accelerometer data collected in free-living conditions. I named the resulting methodology the "Aston Gait Detection Algorithm" (AGDA).

At the same time, my supervisors and I discussed applying for the UK BioBank accelerometer dataset. This would provide me with a larger dataset to test and apply the AGDA on. Simultaneously, I continued working on the AGDA, including completely reinventing it in June 2020, changing my methodology focus from the Switching AR approach to the Dominant Axis Identification approach.

During my PhD, I attended the Advances in Data Science conference, the UK-RiME conference, the Clinical Science & Engineering for Digital Health Workshop and the Digital Heath Event. Furthermore, I presented the poster "Data wrangling of the KOALAP Study: Process and Challenges" at the UK-RiME and the UoM SBS Event and held internal presentations about the topic of "Data Wrangling" and "Extracting Gait from Noise".

1. Introduction

We started the project to determine the relationship between physical activity and rheumatic and musculoskeletal diseases (RMD) with the aim to develop an algorithm for this purpose. The first step was to identify data types suitable for analysing this relationship. There were already some projects within the Division of Musculoskeletal & Dermatological Sciences of the University of Manchester that had collected some form of physical activity data from patients with RMDs. Furthermore, we decided to apply for access to the UK BioBank accelerometer dataset, which contains accelerometer data collected over one week from over 100.000 participants. Within the department, we looked at the Insoles in Response (InResponse), QUAlity of life, Sleep and rheumatoid ArthRitis (QUASAR) and the Knee OsteoArthritis: Linking Activity and Pain (KOALAP) studies as potential datasets to analyse. In the end, we decided to investigate the KOALAP dataset. Its accelerometer dataset was the closest in format to the UK BioBank study and it provided us with a total of 2.18 years worth of accelerometer data from 26 Osteoarthritis (OA) patients to investigate. However, we also discovered that wrangling data taken from consumer devices can be difficult, time-consuming and sometimes even hostile.

We started our investigation into physical behaviour by asking ourselves what kind of behaviours can be identified using accelerometers. Identifying gait patterns became our goal since walking behaviours are a good indicator of mobility and the periodicity of gait is a feature of the behaviour we were able to build an algorithm around for its identification. Furthermore, we were interested in identifying parameters of gait that might enable us to make new observations that go beyond passively recording levels of physical activity in the daily life of patients with RMDs.

Our overall aim during the course of my PhD was to develop a gait detection algorithm that uses accelerometer data, which was recorded on the wrist in free-living conditions, to identify walking behaviours and extract gait parameters. This algorithm has the potential usefulness to be applied to data collected by consumer devices, such as the Huawei Watch 2 smartwatch, which was used in the collection of the KOALP dataset. With devices like smartwatches and Fitbits becoming more popular with the general public, collecting data with the above-described properties will become cheaper and more accessible to implement in future studies. Developing apps or algorithms for self-monitoring of patients' health could increase the economical validity of physical function tests after surgery. This could also save healthcare professionals and patients time and money. We, therefore, aim for our algorithm to be easily applicable to accelerometer data collected at the wrist in free-living conditions.

The following chapters are dedicated to providing the necessary background for a basic understanding of gait measures, accelerometry and RMDs. We will focus on the relationship between OA and gait in detail. Most investigation of gait patterns using accelerometry are done on OA patients and most of the participant pool we investigated in this thesis consist of OA patients. After the introduction chapters, we will present the literature review, where we looked at the previous use of accelerometers to investigate physical functioning in RMD patients. Afterwards, we will present the dataset used in this thesis. This is followed, by the description of our self-reported dataset, in which we explore the difference between research-grade and consumer devices, and

the patterns of gait behaviour recorded from wrist-worn accelerometers. Following that, we will present our own gait detection and step parameter extraction algorithm. We then present the discoveries that were made using the KOALAP dataset. They include wrangling raw accelerometer data from a consumer device and correlating actively collected pain report data with passively recorded step parameter data. At last, we also apply our algorithm to the UK BioBank dataset, enabling us to compare different participant groups with RMDs with matched non-RMD control groups.

1.1. Gait in Humans

The term gait refers to a form of locomotion that is defined as the manner of a person's walking. Gait can be *periodic* when walking in a smooth, uninterrupted pattern or *non-periodic* when walking over rough terrain where the gait pattern needs to be interrupted or adjusted constantly (Song & Waldron, 1987). Analysis of gait patterns is usually done in the lab and focuses on periodic gait. In the following section, we will describe the typical bipedal gait pattern and list some gait parameter terminology.

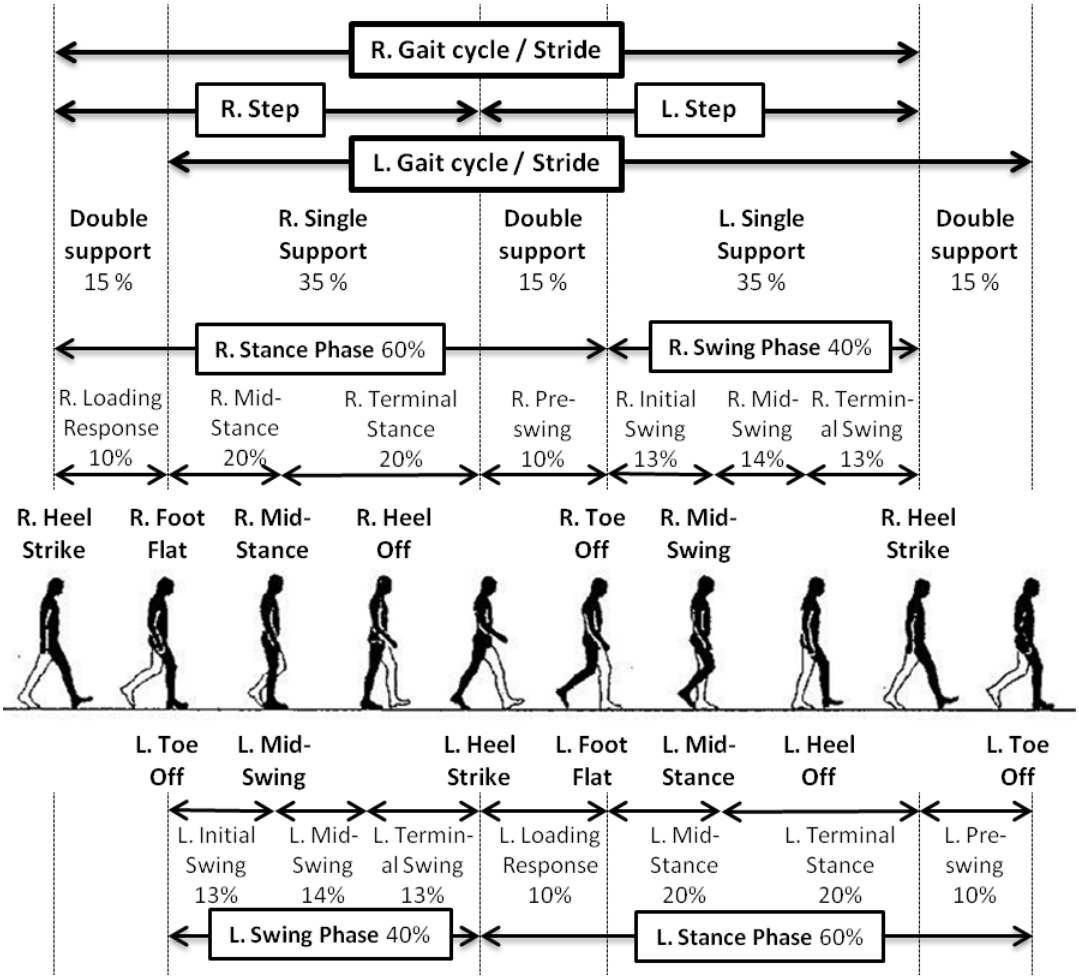


Figure 1 – ***Gait Phases of a typical Gait Cycle, naming events, periods and phases within the gait cycle.*** *Percentage ratio is given for periods (Neumann, 2010) and phases (Inman, Ralston, & Todd, 1981), they can vary between sources and do not necessarily overlap. Adapted from Inman, Ralston and Todd (1981) and Neumann (2010); R. = right, L. = left.*

Gait consists of consecutive gait cycles. A visualisation of a normal gait cycle can be found in Figure 1. The gait cycle starts with the *heel strike* (also known as initial contact) of the leading leg,

followed by the *foot flat*. The interval between the heel strike and the foot flat element is called the *loading phase*. During that phase, the bodyweight is rapidly shifted onto the leading leg. Since both legs are still connected to the ground the loading phase is a *double support phase*. After the foot flat incident, the opposite leg takes off from the ground and the *single support phase* starts with the *mid-stance* phase. During that phase, the body is shifted forward until the leading leg reaches the *mid stand*. During the following *terminal stance* phase, the heel of the leading leg breaks contact with the floor (*heel off*). The terminal stance phase, as well as the single support phase, ends when the *heel strike* of the opposite leg occurs. The first step with the leading leg is now completed and the opposite leg starts its stance phase. A double support time follows that includes the *pre-swing* phase of the leading leg. With the *toe-off* occurrence, the leading leg starts its *swing phase* and the opposite leg is now carrying the body's full weight. During the single support time portions of the gait cycle, the gait of the individual is at its least stable. The swing of the leg is used to propel the body forward. The swing phase starts with the *initial swing phase*, followed by the *mid-swing phase* and ends in the *terminal swing phase*. The terminal swing phase ends with the completion of the step of the opposite leg and starts the next gait cycle with the heel strike of the leading leg (Rueterbories, Spaich, Larsen, & Andersen, 2010).

Table 1 - Gait Parameters: Terms, descriptions and calculations.

Term/s	Description	Calculation
Spatial Descriptors		
Step length	Distance between heel-strike of opposite legs	Step length = Velocity / Frequency
Stride length / Gait cycle	Distance between heel-strike of the same leg	Stride length = 2 * Step length
Temporal Descriptors		
Gait cycle time/ Stride time	Duration of one gait cycle	The movement starts at ground contact with a starting leg (e.g. right heel-strike) to the next ground contact with the same limb
Step time	Duration of a left or right step	Time between the right and left initial ground contact
Cadence/ Frequency/	Rate of walk Frequency of gait	Frequency = Step Count / Time
Spatial-Temporal Descriptors		
Velocity or Speed	Speed of walk	Velocity = Total distance / Total time
Gait Quality		
Symmetry	Ratio between the left and right leg gait phases	see Figure 1, i.e. difference between left and right single-support phase
Variability & Regularity	Smoothness of walk	see Figure 1, ratio between double support and single support length

A summary of typical gait parameters, their definitions and how to calculate each of them can be found in Table 1. A person's gait parameters are dependent on the person's gender and age.

For example, speed and step length are lower in females and slower and shorter with increased age (Öberg, Karsznia, & Öberg, 1993).

1.2. Physical Activity and Physical Function

In accordance with the World Health Organisation (WHO), **physical activity** is defined as "any bodily movement produced by skeletal muscles that require energy expenditure" (WHO, n.d.). Four important aspects of physical activity (summarised as FITT) need to be considered when investigating the relationship of physical activity or inactivity with chronic diseases. They consist of the **frequency** of sessions with an activity level of a certain **intensity**, the length of the sessions (**time**) and the **type** of activity. They are often combined in the measure of Metabolic Equivalent of Task (1 MET = 1 kcal/(kg*h)), to estimate the overall cost of energy expended during one session (Barisic, Leatherdale, & Kreiger, 2011). A score of under 3 MET is considered light-intensity activity (LIA), from 3 to under 6 MET as moderate-intensity (MIA) and over 6 MET as vigorous-intensity (VIA). Different types of activities are associated with certain MET scores, for example, walking at about 5 km/h has an energy expenditure of 3.3 MET per hour (Office of Disease Prevention and Health Care, n.d.).

An individual's ability to perform physical activities can be assessed by looking at different aspects of their **physical functioning**. A test protocol with set functional measures can be used in a controlled environment to assess the individual's "**Capacity**" of performing physical tasks. Their "**Performance**" on the other hand describes how the individual's physical abilities hold up in their everyday environment (Tomkins-Lane & Haig, 2012).

Measuring the amount of walking the patient engages in during the course of the day would therefore be a measure of the patient's physical activity. On the other hand, measures of the patient's cadence, speed or gait symmetry are measures of physical functioning.

1.3. Accelerometry

The following chapter will explain what acceleration is and what measures can be inferred from the acceleration that is mentioned in the literature review. Then different types of acceleration and the effect of gravity on accelerometers are described. Next, different types of accelerometers are listed and the functioning of Micro-Electro-Mechanical Systems (MEMS) is described.

Acceleration is defined as the rate of change of velocity per unit of time. Therefore acceleration represents a change in the speed or direction of the movement of an object. Acceleration happens if the object is speeding up, slowing down or changing its movement direction. The average acceleration of an object is calculated by dividing the velocity change (initial subtracted from final velocity) by the total time as measured in m/sec^2. For example, if a car speeds up from 10 m/sec to 25 m/sec in 20 sec the acceleration will be 0.75 m/sec^2:

$$a = \frac{\Delta v}{\Delta t} = \frac{(vf - vi)}{\Delta t} = \frac{\frac{25m}{sec} - \frac{10m}{sec}}{20\ sec} = 0.75\,\frac{m}{sec^2}$$

Acceleration measures are usually given in units of ***g*** with 1 g being approximately 9.8 m/sec^2.

We can distinguish between several different sources of acceleration: motion, vibration and shock. Small magnitude accelerations such as human motion or sustained acceleration such as speeding up a car are **motions**. Note that, when a car is already in motion and moves at a constant speed on a linear road no acceleration occurs. **Vibrations** are oscillating movements around a centre point, such as a running engine. The last measure is **shock**, as defined by an abrupt change in acceleration, such as the impact of an object falling and hitting the ground (Hanly, 2016).

There are different types of acceleration that can be measured. The above-described sources of motion, shock and vibration can be identified when looking at **dynamic acceleration** (d), while the **static acceleration** (g) is caused by the force of gravity acting on the accelerometer even when it is stationary (Dimension Engineering, n.d.). We are interested in the acceleration caused by motions when using accelerometers to investigate activity patterns. In order to extract the dynamic acceleration from ***raw accelerometer data (a)***, the effect of gravity on the accelerometer is accounted for by removing the static acceleration:

$$d = a - g$$

It should be mentioned that static accelerometer data can not be used to determine the accelerometer's positioning towards the earth's surface nor can it detect tilt. Redell (1998) explained this problem by stating that an accelerometer does not detect gravitational acceleration, but instead detects aberrations from the gravitational force of the freefall, meaning that an actual free-falling accelerometer would not record any acceleration (acceleration = 0 g), while an accelerometer laying on a surface would measure the force that works against the gravity holding the accelerometer on the surface (acceleration = 1 g). Local gravitation can only be inferred as the opposite of its acceleration when the accelerometer is stationary. The direction of gravity can therefore not be detected by an accelerometer without already knowing its orientation.

Furthermore, the dimensions of acceleration data collected from tri-axial accelerometers can be reduced to one dimension for analysis. **Resultant acceleration** (or the magnitude of the accelerometer data, dr) is the averaged acceleration over the three axes of a tri-axial accelerometer:

$$dr = \sqrt{(d_x)^2 + (d_y)^2 + (d_z)^2}$$

Different types of accelerometers have different sensitivities to detect different types of acceleration. **Piezoelectric accelerometers** are **AC-Response accelerometers** that can measure dynamic events, but not static or dynamic acceleration. They are most suited to measure vibrations or shock and are the most resilient type of accelerometer. **DC-response accelerometers**, on the other hand, can detect static acceleration and g-forces. They can be either **piezoresistive or capacitive**. The former has a large bandwidth to handle high frequencies or g-forces and has the best signal-to-noise ratio. However, it is very temperature sensitive and expensive. The capacitive accelerometer has a worse signal-to-noise and is more suitable for low-frequency motion and low g-forces. It is the most commonly used accelerometer in commercial applications such as iPhones due to its low cost, small size and weight and ability to capture acceleration under 200 g. DC-

response accelerometers use **MEMS** to detect accelerations (Hanly, 2016) (TE Connectivity, 2017).

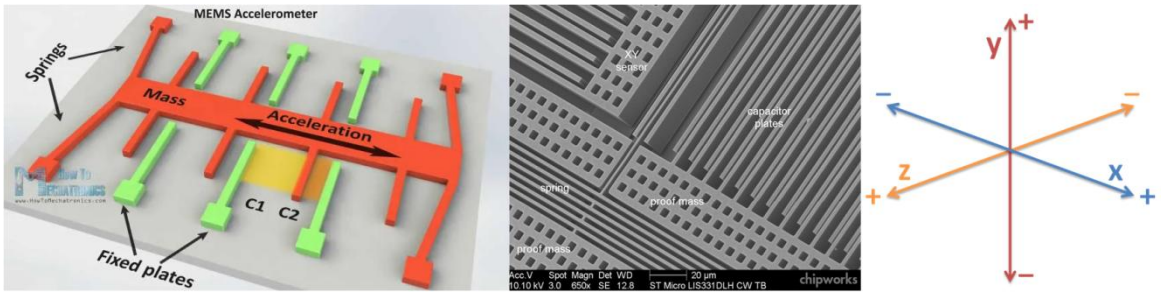


*Figure 2 – **MEMS Accelerometer**. Left: Graphical representation of a MEMS accelerometer (Dejan, 2015). Middle: Microscopic image of a MEMS accelerometer (Dixon-Warren, 2010). Right: Directions of acceleration measured by a tri-axial accelerometer (Kalane, 2018)*

The MEMS concept was first presented by Howe and Chang (1989). The devices consist of micromechanical structures made of electrically conductive silicon that can detect the mechanical acceleration of gravity (see Figure 2). The capacitor consists of conductive plates that are either fixed in position or can move freely. A series of these plates interlock with each other in a combed finger arrangement but are still electrically separated from one another. This structure forms a capacitor that experiences a change in capacitance when the distance between the fingers changes. In order to enable a change of distance between the fingers in accordance with acceleration, the freely moving plates are connected to a weight via spring contacts. When movement, vibration or gravity acts on the weight, it pulls on the movable plate and therefore changes the capacitance between the plates. Applying Newton's second law of motion to the spring force we can calculate the acceleration as a function of capacitance. The tri-axial MEMS accelerometer consists of three MEMS units arranged in the x-, y- and z-directions (John, 2011).

Using an accelerometer for data collection commonly requires selecting the **Sample Frequency** (in Hz) and the **Accelerometer Range** (in g). A "sample" or "observations" (obs) described one measurement made by the accelerometer across all three axes at the same point in time. At an example frequency of 50 Hz a second of data would be reflected by 50 obs.

1.4. Use of Accelerometry in Gait Analysis

The following terms will be used in the literature review when talking about accelerometer and gyroscope measures:

Acceleration Force	Acceleration magnitude of a movement (i.e. joint loading, heel-strike amplitude or transient, initial peak amplitude (IPA))
Acceleration Range	Peak-to-peak acceleration
Acceleration Power	Measure of control over movement
Acceleration Moment	Total movement or stress
Angular Velocity	Rate at which a rotating body changes its angular position (represents

the range of possible rotation)

Range of Motion (ROM) Joint abduction and flexion range, the extent to which the affected joint can reach the full range of movements a non-affected joint would be able to do

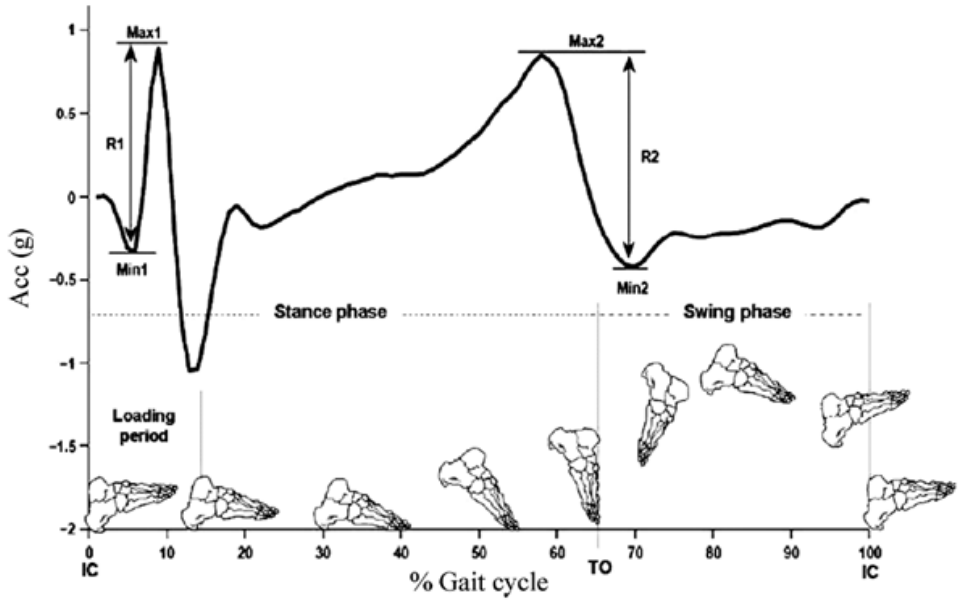


*Figure 3 - **Mean accelerometer curve during a gait cycle**. Depicted in Turcot, Aissaoui, Boivin, Pelletie, Hagemeister and de Guise (2008). Data was recorded using a tri-axial accelerometer on the leg averaged over 15 gait cycles. TO = toe-off, IC = initial contact, R = acceleration range, Max & Min = local acceleration maximum and minimum.*

Figure 3 shows the acceleration associated with one gait cycle, as described in Turcot et al. (2008). First, the heel-strike occurs during the loading phase, indicated in the accelerometer data by a smaller local minimum (Min1) at the initial peak amplitude (IPA) followed by a local maximum (Max1) and ending in a larger minimum when entering the mid-stance with the foot-flat posture. The magnitude of acceleration between Min1 and Max1 is described as the acceleration range (R1). Next, a change towards positive acceleration transpires during the terminal-stance and pre-swing phase when the heel-off occurs. The second local maximum (Max2) is just reached before the toe-off that initiates the swing phase, leading to another local minimum (Min2) with a second range (R2). A full gait cycle starts again with the heel-strike of the starting foot. Max1 and Max 2 represent the acceleration force measured at the foot at which the accelerometer is located (Max1) and the opposite foot (Max2). Looking at the accelerometer data presented in Figure 3, we can postulate that accelerometers located on the arms and legs will record asymmetrical acceleration forces during symmetrical gait. Accelerometers located at the centre of the body, on the other hand, will measure equal acceleration forces during symmetrical gait.

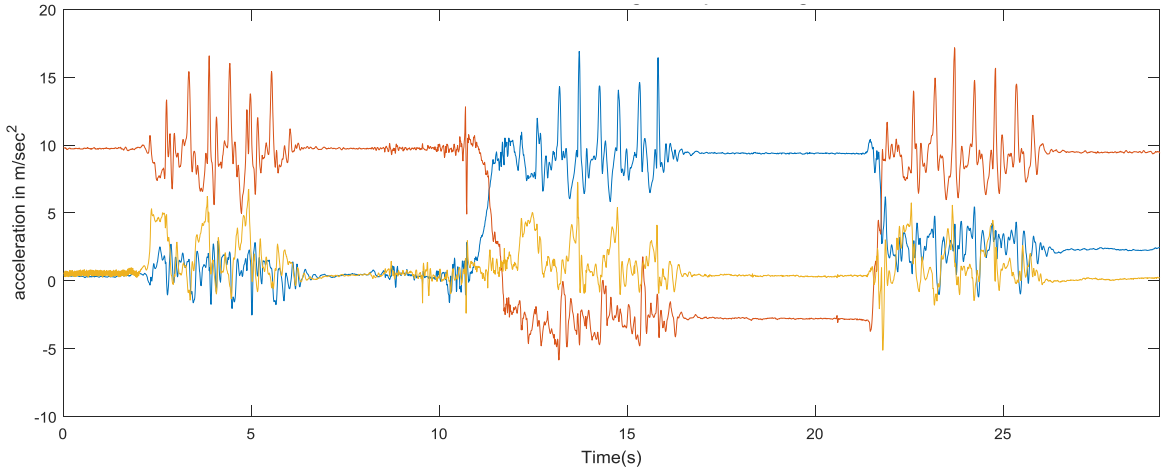


*Figure 4 – **Accelerometer data during interrupted walking**. Data was recorded from a smartphone by Badawy, Raykov and Little (2017). Blue = x-axis, red = y-axis, yellow = z-axis.*

An example of accelerometer data recorded by a smartphone during gait is depicted in Figure 4. The participant was requested to walk for 5 seconds (sec), then to stop for 5 sec, turn the smartphone, and repeat. Notable about the visuals of the raw acceleration data is the clearly visible static acceleration. At the 11-sec and the 22-sec mark, a change of orientation occurs with gravity acting first on the y-axis, then on the x-axis and again on the y-axis with a force of 9.8 m/sec^2 (equal to 1 g).

A common feature extracted to analyse gait patterns is the magnitude of acceleration. While commonly the magnitude dr is calculated with $dr = \sqrt{\sum_{i=0}^{n} d^2}$, taking the logarithm of the magnitude (drl) using the equation $drl = \log_{10} dr$, is also in praxis. *dr* is useful when analysing the data for variations at the larger frequency ranges, while *drl* is used when focused on smaller changes in the data pattern. For example, to differentiate episodes of walking from episodes of non-walking the dr will be more appropriate. On the other hand, the drl could be used to differentiate between different walking types, where the focus lies more in the detail of the pattern.

1.5. Rheumatic and Musculoskeletal Diseases

According to the 2017 evaluation of the "State of Musculoskeletal Health" compiled by Versus Arthritis (then called Arthritis Research UK), RMDs are the leading cause of years lived with disability, with about 3.7 million working-age people in the UK suffering from them. Every year, that leads to a loss of 30 million working days. 18% of people diagnosed with Musculoskeletal conditions (MSK) had to give up working and for about 43% of sufferers working ability is negatively impacted. According to the latest estimates, osteoarthritis (OA) and rheumatoid arthritis (RA) alone directly and indirectly cost the economy about £21.6 billion. MSK, therefore, has a serious impact both on the individual patient and on society as a whole (Arthritis Research UK, 2017). Furthermore, the above-mentioned impact will only increase with an ageing population. In the following literature review, we will focus on RA and OA since they are the most common forms of arthritis in the UK.

1.5.1. Osteoarthritis

OA is a degenerative joint disease that affects about 33% (around 8.75 million) of individuals over the age of 45 years in the UK. With increasing age, the proportion of OA patients rises and reaches 50% in the 75 years plus population (Arthritis Research UK, 2017). OA is foremost defined as a disease of joint overuse. Though it is not classified as a joint inflammation disease like RA, joints with OA do show an inflamed and thickened synovium which will show an increased fluid production. This will cause the joint capsule to stretch and thicken, likely in an attempt to stabilise the joint. The wear and tear aspect of the disease can be seen in the thinning and roughening of the cartilage, a structure that is important for shock absorption. The bone is no longer protected by the cartilage, leading to joint mincing, bone polishing and the formation of osteophytes that cause the bone to grow outwards. In later stages that can lead to deformation of the bone through bone angulation (Arthritis Research UK, 2012).

Age is a risk factor for the development of OA, mostly due to damage that accumulates in the joint over time and the weakening of cartilage. Women tend to suffer from OA more often and at higher severity levels than men (Zhang & Jordan, 2010). Obesity and consequently a lack of physical activity is another common risk factor. Looking at 525 patients with knee OA, Coggon et al. (2001) found a consistently higher body mass index (BMI) in the patient compared to the control group, with nearly 3 times as many individuals with OA being obese. They estimated that about 57% of knee OA surgery could have been avoided in the overweight OA group by reducing their weight to recommended BMIs. Not only is obesity more common in patients with OA, but it also is related to the progression of the disease (Marks, 2007) and is thought of as the most modifiable risk factor. Other risk factors for OA include previous injuries to the joints (Palazzo, Nguyen, Lefevre-Colau, Rannou, & Poiraudeau, 2016). Also, having an occupation that involved joint straining activities, like lifelong high levels of physical activities as found in professional football players can contribute (Gouttebarge, Haruhito, & Kerkhoffs, 2018). Zhang and Jordan (2010) furthermore summarise risk factors caused by the patients' physiology. While mechanical factors such as muscle weakness can increase the risk of symptomatic knee OA, increased grip strength was found to be associated with an increased risk of hand OA (Chaisson, Zhang, Sharma, Kannel, & Felson, 1999). Also, miss-distribution of load on the knee joints, called knee malalignment, is associated with radiographic OA. For example, medial alignment is linked to a two-fold increase in knee OA risk (Brouwer, et al., 2007). However, it is disputed if the malalignment is a risk factor or a result of OA severity. Other factors that contribute to OA through their impact on joint usage are knee laxity and limb length inequality.

For the treatment of OA, the National Health Service (NHS) recommends foremost lifestyle changes that reduce risk factors, medications to reduce pain and supportive treatments for everyday disease management (NHS choices, 2016). Increases in muscle strength, improvement of body posture and loss of weight can be achieved through an increase in exercise. Medications for pain relief start with the painkiller paracetamol. If this is not effective, non-steroidal anti-inflammatory drugs (NSAID), such as ibuprofen; opioids, such as codeine or tramadol; or capsaicin cream might be used. For severe cases, corticosteroid injections and viscosupplementation may be injected directly into the affected joints. Supportive treatments include transcutaneous electrical

nerve stimulation (TENS), the use of hot or cold packs, visiting a physiotherapist to prevent muscle wasting, and devices that assist with the patient's mobility. In the latter case, the choice of footwear and the use of insoles can be beneficial to help with shock-absorbing. In the most severe cases, surgery might be required. The most common one is arthroplasty when the joints will be replaced by prosthetic joints. However, they will need to be revised around every 20 years. Joint resurfacing replaces only the damaged part of the joint with metal components. During an osteotomy, small parts of the bone are taken off to change the alignment of the knee and therefore reduce pressure on the joint. In extreme cases, arthrodesis may be performed in which the joint is fused into a stable but unmovable permanent position (NHS choices, 2016).

1.5.2. Benefits of Physical Exercise

The knee and hip are the most commonly affected joints in OA, followed by the neck, back, hands and feet. The symptoms can be bilateral or unilateral, with the latter being more common (Arthritis Research UK, 2012). Consequently, the asymmetry of the symptoms can lead the patient to load one side of their body more than the other during activities in order to avoid pain and to account for joint instability (Swift, 2002). Gait and progression of the disease alike are affected by loading mechanical factors that are related to joint vulnerability and extreme joint loading. For example, in moderate to severe knee OA malalignment can cause a valgus (lateral) or varus (medial) thrust pattern that again can increase the cartilage loss (Hunter & Eckstein, 2009).

While the joint pain experienced by patients with arthritis can have a negative impact on the patient's willingness to commit to exercise, exercise is still an important feature to prevent the further degeneration of the joints. This might feel counterintuitive when considering that OA is a degenerative joint disease and a decrease of engagement in physical activity is a common adverse consequence of OA. However, a lack of use can cause stiffer and weaker muscles which are a risk factor for disease progression, since lack of muscle strength will cause the joints to become more unstable. Nevertheless, exercise needs to be adapted to the patient’s individual condition and should, therefore, depend on a personalised fitness plan made by a physiotherapist (Arthritis Research UK, 2011).

2. Literature Review

2.1. Abstract

A cohort of studies investigated the effect of physical activity as measured in energy levels and step counts of people with rheumatoid- (RA) and osteoarthritis (OA). However, these approaches do not look beyond sorting people into low, moderate and vigorous groups in order to compare them with disease symptoms. The use of accelerometers to investigate gait in patients with Osteoarthritis opens up the potential to track physical function variations over time and to relate it back to pain reports. Extracting information about the physical functions of the patient can provide more inside than physical activity data on its own. Combined with the advancing field of digital epidemiology and patients' self-monitoring on consumer devices, this opens the opportunity to relate patients' immediate physical functioning to self-reported pain data.

2.2. Introduction

The present literature review will describe the use of sensory monitoring devices in arthritis research: more specifically, the research that was done in the last 30 years using accelerometer data to investigate physical functioning in patients with OA and RA. The goal of the review is to examine how accelerometers are applied in rheumatic and musculoskeletal diseases (RMD) research. Further interests are the aspects of physical functioning that were most commonly investigated and what kind of measures were collected.

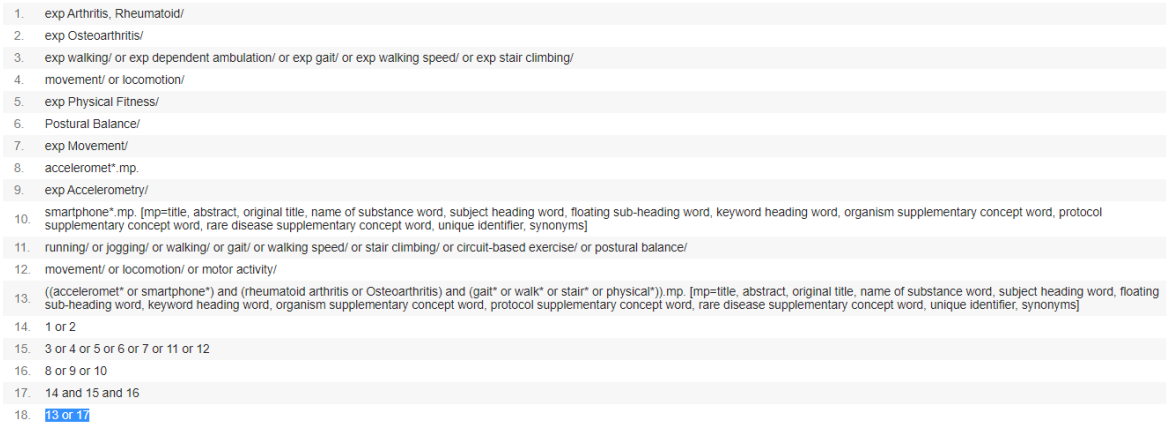
We will focus on studies that examined physical functioning in individuals with OA and RA with the use of accelerometers. Over time the availability of accelerometers, gyroscopes and magnetometers has become a common feature in smartphones and the popularity of smartwatches and activity monitors is steadily increasing. Passive monitoring of activity has the potential to provide insight into the effect of the disease in a more natural environment compared to measuring physical function in a laboratory. Furthermore, monitoring physical activity over the course of days, weeks or even months can provide a more objective and in-depth picture of patients' abilities than self-reported data alone. Finally, the focus of RMD was restricted to OA and RA. They were selected since they are the most common form of Arthritis in the UK.

The body of this literature review and its conclusion were written in December 2017. Newer articles are therefore not included in the body of the review and not considered in the conclusion. We repeated the literature search in March 2022 and commented on notable papers in the added Chapter 2.6. Furthermore, we will explain how the literature review informed the development of this thesis in Chapter 2.7.

2.3. Methodology

For the body of the literature review, we searched the Medline electronic database in November 2017. The same search and exclusion conditions were applied in March 2022 with the additional filter of only looking for papers published in 2018 or later, and to exclude papers mentioning “Physical Activity” automatically instead of manually.

The Ovid MEDLINE(R) thesaurus was used to identify studies investigating the physical functions of patients with RA and OA using accelerometers. We used the following search commands displayed in Figure 5. The term “explode” means that all subcategories of the following medical subject heading are included. Medical subject headings are indicated by closing with the slash (/) symbol. They do not require specifying their multi-purpose. For an advanced search, the term “multi-purpose” determined which fields are searched.



1.	exp Arthritis, Rheumatoid/
2.	exp Osteoarthritis/
3.	exp walking/ or exp dependent ambulation/ or exp gait/ or exp walking speed/ or exp stair climbing/
4.	movement/ or locomotion/
5.	exp Physical Fitness/
6.	Postural Balance/
7.	exp Movement/
8.	acceleromet*.mp.
9.	exp Accelerometry/
10.	smartphone*.mp. [mp=title, abstract, original title, name of substance word, subject heading word, floating sub-heading word, keyword heading word, organism supplementary concept word, protocol supplementary concept word, rare disease supplementary concept word, unique identifier, synonyms]
11.	running/ or jogging/ or walking/ or gait/ or walking speed/ or stair climbing/ or circuit-based exercise/ or postural balance/
12.	movement/ or locomotion/ or motor activity/
13.	((acceleromet* or smartphone*) and (rheumatoid arthritis or Osteoarthritis) and (gait* or walk* or stair* or physical*)).mp. [mp=title, abstract, original title, name of substance word, subject heading word, floating sub-heading word, keyword heading word, organism supplementary concept word, protocol supplementary concept word, rare disease supplementary concept word, unique identifier, synonyms]
14.	1 or 2
15.	3 or 4 or 5 or 6 or 7 or 11 or 12
16.	8 or 9 or 10
17.	14 and 15 and 16
18.	13 or 17

*Figure 5 - **Search terms used for the literature search in Ovid MEDLINE(R)**. The line marked in blue was executed for the literature search in November 2017. exp = explode, mp = multi-purpose, / = marks medical subject headings.*

We excluded articles when no actual accelerometers were used in the study. For example, some studies used pedometers instead of accelerometers to measure step count. Furthermore, articles that only measured physical activity and not physical functions as defined in the introduction were excluded. After first scanning the output of our literature search, we determined that physical activity measures such as step count and separating physical activity into time spent in light-intensity, moderate-intensity or vigorous-intensity was commonly researched and reviewed. Therefore, we decided to look beyond physical activity and focus on physical functions. At last, we excluded studies looking at underage participant groups or animals.

In March 2022, for the repeat of the literature review, we wrote out the search terms as written below. We read all abstracts and looked at papers that piqued our interest as defined by search number 5.

1) exp Arthritis, Rheumatoid/ or exp Osteoarthritis/ or rheumatoid arthritis or osteoarthritis
2) exp Accelerometry/ or acceleromet* or smartphone*
3) exp Movement/ or exp Locomotion/ or exp Motor activity/ or exp walking/ or exp running/ or exp gait/ or exp stair climbing/ or Postural Balance/ or Physical Fitness/ or circuit-based exercise/ or gait* or walk* or stair* or physical*
4) “Physical Activity” or “physical activity” or “Physical activity”
5) (1 and 2 and 3) not 4

2.4. Body

2.4.1. Article Selection

A total of 33 papers were selected for inclusion in the review (see Figure 6). They are summarised in Table 2. In particular information about the types of accelerometers used in the studies and how they were used for data collection was extracted. Measures that were recorded and the means of collecting these measures are summarised in the table. The activities participants were tested on are listed and a note is made of any interventions that were part of the methodology. The table is sorted by date and then alphabetically by the first author. Exceptions to this rule are articles that had identical author groups and very similar methodology (see Table 2).

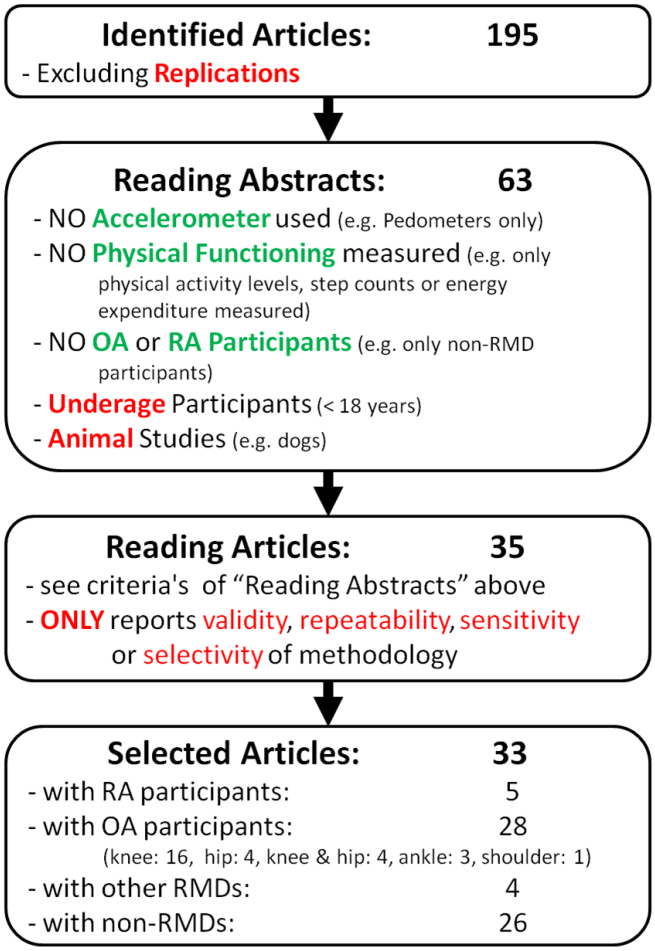


*Figure 6 - **Article selection process with exclusion criteria**.*

The following review will describe the studies in the context of how the development of more advanced accelerometry and gait detection algorithms changed the use of technology when measuring physical functioning in OA, the use of accelerometers to track changes in physical abilities of the patients after surgery, how accelerometers are used to evaluate insole intervention, comparisons between patients and controls or between groups with different disease severity, the effect of increased walking speed on outcome measures, the use of activity pattern identification or classifications and studies looking at no-gait related mobility test. Except for the last chapter the physical activity's that the participants are tested on are all gait-related.

Table 2 - Summary of Articles included in the Literature Review. Sorted primarily by date. Table specific abbreviations and acronyms: accl = accelerometer; accn = acceleration; AP = antero-posterior; ASES = American Shoulder and Elbow Surgeons score; d = days; DASH = Disability of the Arm, Shoulder and Hand score; gyro = gyroscope; m = metre; mth = month; ML = medial-lateral; pre = previous; RMS = root mean square; SS = Simple Shoulder test; V = vertical; VAS = Visual Analogue Scale; w = weeks; // = not given. Additional sources: [1] Van den Dikkenberg et al. (2002), [2] Van Hees et al. (2009), [3] Bussmann et al. (2001).

Article	Participants (number)	Objectives	Intervention	Sensory Devices (Name) (Body site)	Sensory Device Measures	Other Measures	Investigated Activity
Smidt, et al., 1977	RA (24) degenerative joint disease (22)	progression	tibiofemoral knee implants (pre; 2, 6 & 12mth)	**Statham Acc**. (1) (Accl.: 3 x uni-axial ,//, //,//,±2.56g) **Pelvis**: second sacral vertebra	harmonic ratio accn. range	Force-plates: step length, stance time, swing time,	walking
Rööser, Ekbladh, & Lidgren, 1988	RA (12) healthy (5)	different heels & shoe-inlays,	bicompartmental knee arthroplasty (6mth) shoe-inlays	**EMT 25** B Elema-Schonander (2) (Accl: // (V), piezoelectric, 12 g., 100Hz; //) **Leg**: knee (tibial tuberosity), foot (medial malleolus)	IPA; mean speed	None	walking (20 m)
Ogata, Yasunaga, & Nomiyama, 1997	medial & lateral knee OA (48: 38/10) healthy (40)	insoles vs no insoles, compare groups	valgus insoles for medial OA or varus insoles for lateral OA (1mth)	**NEC San-ei, 1823** (1) (Accl.: uni-axial (ML), //, //, //, //) **Knee**: tibial tubercle	IPA; secondary peak accn.	Radiography: OA severity; Questioning: insoles effectiveness; Strain gauge: time of heel-strike	walking (3 gait cycles)
Aminian, et al., 1999	unilateral hip OA (12) healthy (30)	progression, compare groups	hip arthroplasty (pre; 3, 6 & 9mth)	**IC sensors 3021** (2) (Accl.: uni-axial (AP), piezoresistive, //, *0-350Hz*, ±5g) **Knee**	gait cycle duration & phases; gait symmetry (left to right leg ratio) & ratio (gait cycle to phase ratio) for stance & double support	Pressure-Sensors: identify heel-strikes	walking (35m)
Auvinet, Chaleil, & Barrey, 1999	hip & knee OA (42) other RMDs (21) healthy (139)	effect of gender, height & age, compare groups validity	None	**Locometrix** (2) (Accl,: 2 x uni-axial (ML & V), //,20g (140g total), 50Hz, //) **Back**: L3-L4 level	cycle frequency; stride symmetry; stride regularity	Lequesne's functional index: disease severity for OA patients	walk (30 m; ~ 20 gait cycles)
Reddy, et al., 2001	RA (11) spondylo-arthropathy (11)	compare groups	None	**Entran, Model EGA** (1) (Accl.: uni-axial (AP), //, 0.5g, *0-500Hz, ±0-10g*) **Knee**: patella	time-domain analysis; mean accn. power	None	rotate leg while sitting
Turcot, et al., 2008	knee OA (9) healthy (9)	compare groups	None	**ADXL320** (Acc.) & **Murata, ENC-03J** (Gyro.) (2)	peak accn. (minimum, maximum) & accn. range	Force-plates: identify gait cycle	walk (25 sec, ~ 15 gait cycles)

Turcot, et al., 2011	knee OA (20) healthy (9)	↑	↑	(Accl:.tri-axial, //, //, //, ±5g) (Gyro.: ±400º/sec) **Leg**: femoral & tibia	linear accn.; angular velocity	Force-Platforms: identify centre of pressure	single-limb stance of affected limb (3 x 4 sec)
van Hemert, et al., 2009	knee OA with resurfaced & not resurfaced patella (53: 31/22)	compare groups	unilateral total knee arthroplasty (6mth)	[1] Dynaport: **IC sensors 3031** (5) *(Accl.: uni-axial (V, AP (only waist)), piezo-resistive, //, 32Hz, ±5g)* **Body**: sternum, waist, lower & upper legs [2] Minimod: **ADXL202** (1) (*Accl.: tri-axial, //, //, 100Hz, //*) **Pelvis**	Dynaport Knee Test: score between 0-100 Minimod Gait Test: speed, asymmetry, irregularity & inefficiency	KSS: clinical outcome	Dynaport: locomotion, rise and decent, transfers, lift and move Minimod Gait Test: walking (20 m)
Liikavainio, et al., 2010	knee OA (54) healthy (53) (male only)	compare groups	None	**Meac-x** (3) (Accl.: tri-axial, //, //, //, ±10g) **Knees**: below joint & above joint (symptomatic only)	IPA, accn. range	Force-plates: GRF; EMG: muscle activation; VAS: subjective knee pain; Radiographic assessment	corridor-level (10 x 2-3 gait cycles x 1.2, 1.5 & 1.7 m/sec) and laboratory-level (only at 1.2 m/sec) walking; stair-walking (10 x up & down x 5 gait cycles x 0.5 & 0.8 m/sec)
Hodt-Billington, et al. **2011 →**	primary hip OA (34)	progression	total hip replacement (pre; 3, 6 & 12mth)	**Logger technology HB** (1) (Accl,.: tri-axial, piezoresistant, 15g, //, //) **Back**: L3-level	trunk symmetry	Electronical-walkway: footfall step-length and single support symmetry	walking (4.3 m x 3 speeds (slow, preferred, fast)
2012 →	end-stage primary hip OA (37) healthy (56)	compare groups & limbs, validity	None				
Jolles, et al., 2011	glenohumeral OA (7) rotator cuff disease (27) healthy (31)	progression, compare groups	total shoulder replacement (pre; 3, 6 & 12mth)	**ADXL 210** (Acc.: 3) & **ADXRS 250** (Gyro.:3) (Accl.: tri-axial, //, //, //, ±5g) (Gyro.: ±400º/sec) **Upper Body**: each humerus and sternal manubrium	angular velocity scores; power score; moment score	VAS, SST, DASH, ASES and Constant score: clinical outcomes	7 shoulder movement test based on the SST
Rouhani, et al. **2011 →**	ankle OA (12) healthy (10)	validity, compare groups & joints	None	**Physilog, BioAGM** (2 or 4) (Accl.: tri-axial, //, //, 200Hz, //) (Gyro.: //) **2011: Leg**: shank, forefoot **2012 & 2014: Leg**: shank, hindfoot, forefoot, toe	joint kinematics: force, moment & power	Force-plates and cameras: identification of gait cycle	walk (100 m)
2012 →	ankle OA (12) healthy (10)	compare groups & joints			joint kinematics: angular rotation	Force-plates and cameras: validation of the system	
2014 →	ankle OA (12) healthy (10)				joint kinematics: force, moment & power	None	

Vissers, et al., 2011	end-stage hip OA (30) healthy (30)	progression, compare groups	total hip arthroplasty (6w pre; 6mth)	[3] **ADXL202** (3) *Accl.: uni-axial (PA (leg), bi-axial (PA & V (sternum)), piezoresistive, //, //, //)* **Body**: sternum, upper legs	percentage, total time & occasions of time in activity, walking & sitting	None	continues measures for one day (~ 15 h), extracting walking and chair rising incidences.
2012 → **Bolink, et al.,**	end-stage knee OA (20) healthy (30)	compare groups, validity	None	**Micro Strain Inertia-Link** (1) (Accl.: tri-axial, //, 39g, 100Hz, ±5g) (Gyro.: ±300º/s) **Pelvis**: posterior superior iliac spines	speed, cadence, step time (irregularity & asymmetry) & length; ROM, peak accn.	None	gait: walking (20 m) sit-to-stand: chair transfers (3 x) step block (3 x 2 legs)
2015 →	end-stage unilateral knee & hip OA (40: 20/20) healthy (20)	compare groups			see Bolink et al., 2012; pelvic ROM, asymmetry overall & at heel-strike		walking (20 m)
Yamada, et al., 2012	RA (39) healthy (20)	correlation with clinical outcome scores	None	**Smartphone: Android** (1) (Accl.: //, //, 139g (total mass), 33Hz, //) **Back**: L3 level	IPA (gait cycle duration), autocorrelation peak (gait balance), coefficient of variance (gait variability)	DAS, mHAQ, walking ability: clinical outcome scores	walking (20 m, 7.75 sec)
Khan, et al., 2013	TKA patients (27 with 38 knees) shoulder patients (18 with 35 knees)	compare groups	total knee arthroplasty (6mth)	**GLI Interactive LLC** (1) Accl.: tri-axial, //, //, 100Hz, ±2g) **Knee**: tibial tubercle	mean accn.; total acc. magnitude	WOMAC & Patient Activity Scale: report instability and pain	walking (3 steps), chair transfers (1 x), box stepping (1 x), walking with a 90º turn (1x) for both legs
Atallah, et al., 2014	knee & hip replacement OA patients (38: 21/17) OA (12) healthy (21)	compare groups	knee/hip replacement (12mth)	***//*** Accl.: tri-axial, //, 7.4g, 130Hz, //) **Head**: ear	gait cycle time; step-period asymmetry	Force-plates: GRF for validation	walking between 4 km/h to 8 km/h in 0.5km/h steps; treadmill incline and decline at 4 km/h
Calliess, et al., 2014	primary knee OA (6)	surgery success assessment for individuals	knee replacement (1d pre; 12mth)	**Shinner 2R** (3) (Accl.: tri-axial, //, //, //, //) (Gyro.: //) (Magnetometer) **Body**: pelvis (L5 level), leg (thigh & tibia)	speed; step cadence, length & symmetry; flexion angles, angular velocity	KSS and Oxford Knee Score: evaluate surgery success	time up and go (3 x) walking (100 m); run (50 m), sprint with abrupt stop; stair walking (4 x)
Staab, et al., 2014	unilateral knee OA (12) healthy (7)	compare groups	None	**Freescale; MMA76260 Q** (Acc.) & **ADXRS 300** (Gyro.) (3) (Accl.: tri-axial, piezoelectric,//, 1000Hz, ±2g) (Gyro.: ±*300º/s*) **Body**: back (L3-level), legs (lateral malleoli)	speed, cadence, step time asymmetry; spectral analysis	Force-plates and cameras: validation	walking (500 m)
Barrois, et al., 2015	hip & knee OA (47) healthy (12) (severity, age & shoe type groups)	compare groups	None	**Xsens MTw** (1) (Accl.: tri-axial, //, *27g*, 100Hz, ±*16g*) **Pelvis**: L4-L5 level	main walking frequency	WOMAC: assess severity groups;	walking (20 m)

Barrois, et al., 2016	Lower limb OA (48) Healthy (12) (grouped by severity)	compare groups, identify effective measuring locations	↑	**Xsens MTw** (4) (Accl.: tri-axial, //, 27g, 100Hz, ±16g) (Gyro.: ±1200º/s) (Magnetometer) **Body**: forehead, back (L4-L5 level), feet (dorsal foot face)	mean accn. & RMS for: angular velocity & linear acc. in the horizontal plane & vertical axis	↑	↑
Christiansen, et al., 2015	unilateral knee OA (24) healthy (12 SCT & 19 6MWT)	progression, compare groups	unilateral total knee arthroplasty (1-2w pre; 5 & 26w)	**Delsys** (2) Accl.: tri-axial, //, 4g, 1000Hz, ±10g) **Legs**: tibia	IPA; absolute limb asymmetry (average right & left IPA)	None	SCT: wtair walking 6MWT: walking (6 min)
Barden, et al., 2016	bilateral knee OA (15) healthy (15)	compare groups and sex between three accn. directions	None	**GENEActiv** (1) (Accl.: tri-axial, //, //, 100 Hz, //) **Back**: L3 level	stride & step time; stride & step regularity; gait symmetry	None	walking (9 min)
Clermont & Barden, 2016	knee OA (15) healthy (15)	compare groups	None	**GENEActive** (1) Accl.: tri-axial, //, //, 100Hz, //) **Back**: lower back	speed, step count; step & stride time & variability (variability measures as STD / fractal scaling index)	KOOS: clinical evaluation	walking (10 min)
Lyytinen, et al., 2016	knee OA (9) healthy (9) (male only)	validity only; but group comparison possible	None (repeated 2w apart for repeatability)	**Meac-x** (1) Accl.: tri-axial, piezoresistive, //, 1000Hz, //) **Leg**: tibia	IPA; IPA range; mean accn. around IPA	WOMAC: self-reported joint pain; Radiographs: OA severity; EMG: mean activity over gait cycle	walking (6 x 15 m, at 1.2 m/sec & 0.5 m/sec) stair walking (3 x 12 steps at 0.5 m/sec)
Zhang, et al., 2016	TKA patients (12) healthy (12)	progression, compare groups & limbs	total knee arthroplasty (pre; 6w, 6mth)	**IDEEA 3** (8) (Accl.: tri-axial, //, //, //, //) **Lower Body**: thighs, ankles, feet, sternum	walking velocity, stride length; single support time; footfall & swing power	AKS: includes knee score (symptoms) and knee function score (walking ability)	walking (40 m)
Andreu-Perez, et al., 2017	RA (10) healthy (20)	classify static, acyclic and transitional activities; validity	None	**Axivity AX3** (1) (Accl.: tri-axial, //, //, 100Hz, //) **Back**: L5 level	machine learning & deep learning; accuracy	None	data tagging: sit - lie - stand transitions
Kobsar, et al., 2016	knee OA (39) (divided into non-, low- & high-responders)	predict responder group, determine best sensor location or combination	hip-strengthening exercise (6w) (pre)	**iNEMO inertial module** (4) (Accl,: //, //, variable, 100Hz. ±16g) (Gyro.: ±2000º/sec) **Body**: back (l3 level), leg (thigh, shank, foot)	classification accuracy	KOOS: grouping responders after intervention	walking (1 min)

2.4.2. Impact of Technical and Methodological Advances on the use of Accelerometers

Three main advancements can be seen throughout the articles selected:

- Accelerometers can replace force-plates for the detection of force measures, such as heel-strike amplitude.
- Technical advances towards tri-axial MEMS accelerometers made the accelerometers lighter, more accurate and therefore more reliable and practical.
- While accelerometers needed to be tightly fitted onto the legs, usable data can now be obtained from various locations on the body.

In the earliest paper identified in the literature search, Smidt et al. (1977) proposed an accelerometer system for gait analysis but still had to use force-plates to identify walking velocity. Force-plates are sensors that are integrated into plates on a walkway. They measure ground reaction forces (GRF) and can, therefore, be used to detect the approximate location, time and power of steps. Smidt et al. (1977) had to mount three uni-axial accelerometers together into a tri-axial accelerometer to detect peak acceleration and the harmonic ratio in three dimensions. Following the publication of this paper, the acceleration range at heel-strike became an established measure to identify joint loading at the knee in RA and OA. Improvements in accelerometer technology in the following years enabled the investigation of different aspects of gait. While Smidt et al. (1977) relied on force-plates to extract temporal data for the identification of gait phases, Aminian et al. (1999) were able to extract heel-strike and toe-off events from uni-axial accelerometers in the anterior-posterior plane. Likewise, they identified the gait phases through an algorithm that used local minima and maxima. Positive acceleration indicated the swing-phase with a global maximum peak at mid-swing. Toe-off and heel-strike on the other hand showed as sharp negative accelerations with a larger amplitude at heel-strike. Consequently, they were able to identify the durations for the gait cycle, stance and double support. To validate their findings, measures were taken from five healthy participants with both an accelerometer and pressure sensors in the shoes. The validity of the method was confirmed when the same temporal results were recorded for both methods. Accelerometers were found not to suffer from the same artefacts as the force sensors caused by the pathological walk of the patients. Furthermore, they determined that at least 20 successful gait cycles are necessary to estimate temporal features of gait, which corresponds to about 30 metres of walking distance for the participant. Nevertheless, one of the problems Aminian et al. (1999) ran into was determining a sufficient threshold to distinguish between heel-strike and toe-off acceleration peaks.

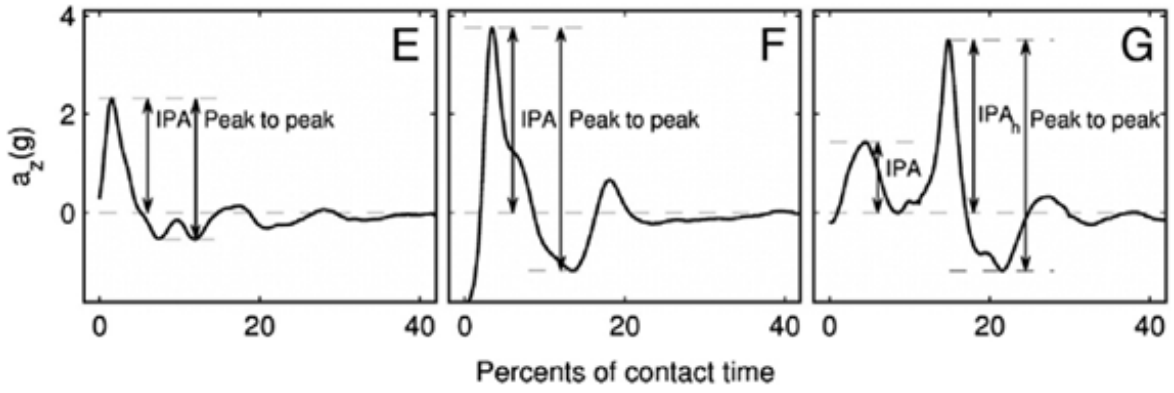


Figure 7 – ***Vertical acceleration during level-walking*** *(E)****, stair ascend*** *(F)* ***and stair descend*** *(G). Taken from Liikavainio et al. (2010). a_z = vertical acceleration, IPA = initial peak acceleration, IPA_h = IPA at heel strike.*

While Turcot et al. (2008) were able to derive peak acceleration and acceleration range from the accelerometers alone, they still relied on force-plates to identify the exact start and end of the gait cycles during analysis. Liikavainio et al. (2010) used skin-mounted, low-weight accelerometers to investigate the gait analysis of level walking, joint loading in OA during stair ascend and descend. However, they also recorded GRF from force-plates and muscle activation from electromyography (EMG) simultaneously. As depicted in Figure 7, the IPA peaked at the loading phase during level-walking and the peak amplitude increased during stair ascend. However, during stair descend, a smaller IPA followed by a larger secondary peak acceleration occurs in the vertical plane. Van Hemert et al. (2009), on the other hand, extracted all their measurements from a lower back based tri-axial accelerometer. They extracted the gait parameters of speed, symmetry and irregularity through the Minimod Gait Test and the Dynaport Knee Test which use five accelerometers at various body locations. The latter consists of 29 daily activities belonging to four categories (locomotion, rising and descending, lifting and moving objects, and transfers). According to Mokkink et al. (2005), it is a well-validated test to assess physical function after total knee arthroplasty (TKA) for OA. Six accelerometers are located on the trunk and leg to evaluate the patients' task performance on a 0 to 100 scale.

Between 2001 and 2014, the use of force-plates to identify gait parameters became progressively less common and was reduced to a validation method. Ironically, one study even used accelerometer data to validate the GRF findings that they obtained from force-plates. They used accelerometers to identify heel-strikes that the force-plate system failed to record, replacing the force-plates, walkways and camera monitoring as a cheaper alternative (Schmid, et al., 2013). The trend away from force-platforms and camera analysis is well illustrated when looking at the studies conducted by Rouhani's research team. Over three studies they tested their ability to assess joint kinematics of patients with ankle OA. First, in 2011, Rouhani et al. (2011) investigated ankle OA by attaching internal measurement units (IMU) consisting of tri-axial accelerometers and gyroscopes to the forefoot and shank, assisted by force-plates and camera detection. They used the latter to identify gait cycle onset while the participants walked a total of 100 metres. A year later, Rouhani et al. (2012) added two additional IMUs at the hind-foot and toe. Furthermore, they relied solely on the IMU to identify the beginning of a gait cycle, instead of using the stationary force-platform and camera recordings to detect the stand phases. Only joint ankle rotation data

was reported this time, while force, moment and power were published in a later article by Rouhani et al. (2014). The new wearable system made of four IMUs was used in the 2014 study and the analysis was split across the shank-hindfoot, hindfoot-forefoot and forefoot-toe. Overall the studies conducted by Rouhani, Favre, Crevoisier and Aminian led to a cheap, quick and easy wearable system for clinical evaluation and treatment monitoring of ankle OA.

While most of the advances that cause accelerometers to be able to replace force sensors and camera monitoring of patients' gait are the result of advanced algorithms that make gait identification and analysis more effective, accelerometers did not start out as a convenient, non-invasive method to measure movement. In the 1980s acceleration range at heel-strike became an established measure to identify joint loading at the knee. Findings in animal studies described how experimentally increased impact loads on animals influenced cartilage degeneration and can, therefore, be considered a contributing factor to OA in humans (Collins & Whittle, 1989). However, it took some time and studies of healthy individuals to bring the accelerometer technology to a standard that was sufficient to give insight into bone acceleration. Collins and Whittle (1989) discussed the necessary developments and described how early studies had to mount the accelerometer surgically to healthy participants' bones. Pronounced differences were found when comparing accelerometer data collected from the skin to bone acceleration data. Better agreement between the two acceleration measures was obtained when accelerometers with a lighter mass were used and when a tight attachment was ensured through custom-made splints. With the use of lighter and stronger mounted accelerometers, the investigation of force loading at the joints could be conducted without surgical intervention, which made the investigation of patients easier and more common. The development of tri-axial accelerometers in particular was an important step to make data collection more accurate and informative. While Amamin's research group had to rely on one or two accelerometer axes and Smidt et al. had to mount accelerometers together before the new millennia, Turcoet et al. (2008) could use accelerometers that were already built tri-axially. They were able to describe a typical mean acceleration curve for a whole gait cycle (see Chapter 1.4, Figure 3) and extract gait parameters relevant for OA and control group comparison. A speculative assumption could be made that the later mentioned study conducted by Rööser, Ekbladh and Lidgren (1988) lacked the ability to discriminate between the effects of different shoe insoles on heel-strike transient because their uni-axial accelerometer only measured acceleration in the vertical direction. Measures in the anterior-posterior direction could have potentially led to more conclusive results.

In the last decade, a new challenge for accelerometer research is measuring meaningful data in more convenient ways for participants. One way to achieve this goal is to change the location of the accelerometers from the legs and hips to places where accelerometers can be attached to common wearable objects, such as phones, watches and even glasses. Yamada et al. (2012) study used accelerometers present in smartphones for gait analysis. While gait symmetry is commonly measured at the leg or the hip, there have even been attempts to measure it through accelerometers attached to the ear. Atallah et al. (2014) identified steps when the local minima reached a threshold at all three axes of the tri-axial, ear-worn accelerometer at once and compared them to GRF measures that were collected through force-plates on a treadmill. They reached about

90% agreement for gait cycle time with a mean difference of about 0.02 sec (0.05 STD), and only a 0.01 sec mean difference (0.07 STD) for step-period asymmetry. Even with treadmill incline or decline, the agreement stayed unchanged, showing that it is possible to make judgements about gait parameters from accelerometer data from the head.

2.4.3. Tracking Progress after Surgery

One potential application of accelerometer data in the clinic is the tracing of recovery after surgery. Smidt et al. (1977) aimed to investigate pathologies like RA and their effect on gait. RA and degenerative joint disease patients were analysed before and repeatedly after surgery. Walking velocity was measured by force-plates and three uni-axial accelerometers located at the pelvis detected peak acceleration and the harmonic ratio. The harmonic ratio indicates how often force changes occur during a gait cycle and can be used to judge patients walking variability. Surgery resulted in an increase in the walking velocity in both groups, but a reduction of peak-to-peak acceleration and a major improvement in the harmonic ratio were found only in the degenerative joint disease group. The RA group, however, only showed a slightly smoother walk and an increased acceleration range. It is notable that the higher the heel-strike amplitude in RA at six months decreased again at twelve months after surgery. Looking at the temporal aspect of gait, knee implants benefited both groups, but gait properties like foot descend and gait variability indicated a stronger benefit of surgery for degenerative diseases such as OA than for the inflammatory RA.

By calculating the percentage of time spent in the stance and double support phase, Aminian et al. (1999) proposed a method to evaluate the recovery of gait symmetry after surgery. They measured unilateral hip OA patients before and after hip arthroplasty. Of particular importance was the double support time, since in patients with left unilateral hip OA the left double stance is shorter than the right double stance, resulting in an asymmetry. The operation decreased 88% of stance and 250% of double stance asymmetry

Van Hemert et al. (2009) main goal was to compare knee OA patients undergoing different kinds of surgery through the Minimod, the DynaPort Knee Test and the Knee Society Score (KSS) to examine the value of resurfacing the patella during TKA. Minimod and DynaPort were already briefly described in the previous chapter. The KSS is a clinical rating scale to objectively rate patients' functional abilities before and after TKA (Scuderi, Bourne, Noble, Benjamin, Lonner, & Scott, 2012). Van Hemert et al. (2009) made the case that simple determination of surgery success through the established KSS is not sufficient to show all potential benefits. Patella resurfacing does not show a significant benefit when looking at the KSS results. However, when functionality was assessed using the DynaPort test the overall score was significant and performance increased in ascending and descending activities, such as stair climbing. Furthermore, the Minimod Gait Test demonstrated a significant increase in step frequency and walking speed in favour of patella lifting, but not in gait symmetry. The study demonstrates how measuring the physical functions of patients should be an important part of treatment evaluation, even if no symptomatic improvement is reported by the patient. However, it should be noted that the KSS was completely revised in 2012 to address problems with validity, reliability and responsiveness of the score (Scuderi, Bourne,

Noble, Benjamin, Lonner, & Scott, 2012) and that the above-mentioned study was published in 2009. This could explain the above studies' inability to find a significant change in the KSS after surgery.

Hodt-Billington et al. (2011) used an accelerometer to record trunk acceleration. Single-support percentage, step length and gait velocity were assessed by an electronic walkway. They looked at total hip replacement patients before, three, six and twelve months after surgery. An autocorrelation procedure of correlating the acceleration curve with itself at a phase shift of a step or stride was used to calculate the between-step (first peak) and between-stride (second peak of the correlation) regularity. The symmetry indices (SI) were calculated by subtracting between-step from between-stride regularity. Noteworthy is the temporal variation in improvement after surgery. An overall significant improvement of anterior-posterior SI occurred between the before and one post-surgery assessment with the largest differences between three and six months. The vertical SI improved overall with the most improvement in the first three months, but medial-lateral SI only showed a significant difference between the sixth and twelfth months of recovery. Step length and single support symmetry showed significant improvement over the whole twelve months, as did gait velocity, but only starting three months after surgery. On the other hand, symptom scores like Quality of Life and pain scores improved immediately after surgery.

Christensen et al. (2015) examined level- and stair-walking to assess loading of the joints measured by the asymmetry between limbs in the few weeks before and after TKA surgery. They calculated an absolute symmetry index between the two limbs by comparing the averaged IPA of the right and left legs. Stair ascend and walking after five minutes showed a significant increase in asymmetry five weeks after the operation, which was gone again by 26 weeks. These two measures and stair descending were also significantly less symmetrical than in the control group only at five weeks. However, the first minute of the five-minute walk did not show any difference in patients' performance compared to the control group or over time. The limb asymmetry resulted from a decrease of IPA measured in the fifth week followed by an increase towards the 26th week. Looking at the limbs during the fifth week showed that IPA decrease in the operated limbs was larger than the decrease in the healthy legs. Gait symmetry measures during stair descend became worse with increased task demands and during longer walking periods as measured by the six-minute walk test. How the choice of test can have an impact on the participant's performance and their distinctness from control groups was also demonstrated by Kahn et al. (2013) and Calliess et al. (2014). Kahn et al. (2013) tested the knee performance of people who underwent TKA at least six months after the operation with a short protocol. Participants had to step forward, do sudden stops, take turns, box stepping and sit up or down. They simply calculated the mean acceleration of the different activities and found a significant difference from shoulder OA patients for the stepping down, the turning and nearly for the sudden stop task, which are movements that put more force on the knee than standing or stepping up. Calliess et al. (2014) on the other hand included running and sprinting in their investigation of level and stair walking and found them to correlate before and 12 months after knee arthroplasty. In their analysis, they looked at six patients' individual profiles which included clinical examination scores (KSS and Oxford Knee Score) and physical functioning outcomes. Overall the temporal parameters of gait improved for the

individuals, especially the running speed and sprint parameters, and correlated well with the clinical examination scores and self-reported satisfaction with surgery outcome. Both studies demonstrated that tasks with increased demands work better to assess group differences in post-operative patients and to assess patients' recovery. This is an important finding since temporal parameters of gait, such as walking velocity and cadence, generally, improve within a year to the same levels as controls which can cause more demanding physical function recovery to be overlooked.

Recently an investigation of post-TKA patients with clinical scores and gait parameters recorded at the legs and chest done by Zhang et al. (2016) found a significant increase of single support time for the operated knee over six months. Furthermore, it was the only tested parameter that was significantly correlated with the patients' clinical scores. Both legs showed a significant increase in footfall between six weeks and six months after surgery. Footfall occurs at the end of the swing phase and gait cycle and joint pain patients tend to be more careful when setting off their feet to compensate for dysfunctionality and to reduce pain, leading to a decrease in mean acceleration. When comparing patients to healthy individuals, footfall measures stayed significantly different even six months after surgery as did walking velocity. Stride length, however, was only lower before and six weeks after surgery. The authors state that, even though footfall showed a significant increase six months after surgery, the patients should continue rehabilitation exercise, due to the remaining difference to the healthy control group.

2.4.4. Use of Insoles

Two studies in the literature search looked at the effect of different shoe types and insoles on the heel-strike impact in patients with RA and OA to identify risk factors of walking on hard surfaces and possible intervention targets. Rööser, Ekbladh and Lidgren (1988) compared the heel-strike transients by measuring the acceleration range of RA patients using two different shoe types with eight variations of insoles. They found no difference in acceleration amplitude at heel-strike for the thin insoles but found a reduction in heel-strike for the thicker compliant- compared to the hard rubber-heel shoes. In the vertical axis, the heel-strike of the leg with the knee attached accelerometer forms a clear acceleration peak, followed by a smaller amplitude indicating the heel-strike of the opposite leg. They point out that RA patients walked more slowly than healthy participants and that the reduced speed was associated with a reduced heel-strike magnitude.

Insoles are also used to correct common foot displacement issues in the lateral and medial direction, as they are commonly found in OA. The investigation of the lateral and medial thrust by Ogata, Yasunaga and Nomiyama (1997) lead to the identification of three thrust patterns: the lateral thrust pattern, where the first and larger acceleration peak occurred in a lateral direction followed by a smaller medial second peak; the medial thrust pattern, with an opposite pattern to the lateral thrust pattern; and the unclassifiable pattern, where no thrust peaks can be identified. The latter was most common in the healthy control group, while patients with medial (lateral) knee OA showed a consistent lateral (medial) thrust pattern. The use of laterally and medially elevated insoles for one month decreased the medial and lateral first peak amplitude as well as reported

pain in both patient groups. They also found that treatment with insoles was more effective at reducing pain in early-stage OA than in later stage OA participants.

2.4.5. Between-Group Comparisons

Just before the start of the new millennia, Auvinet, Chaleil and Barrey (1999) reported strong evidence for a reduction in gait cycle frequency, stride symmetry and regularity in OA and other mechanical RMD disorders compared to healthy individuals. They used 139 healthy controls and 63 patients (42 of them with hip or knee OA) and found an effect of gender and height, but not of age on gait cycle duration when investigating pathological gait of RMD patients. The absence of an effect of age on the measured gait parameter in patients was used to justify the age difference between the patients and the control group in their study. Comparing the groups, they found all patient groups to perform worse than the healthy controls in every measured gait parameter. Furthermore, the gait parameters of the hip OA groups were negatively correlated with disease severity, while this was only the case for stride symmetry in knee OA. They raise some accelerometer data limitations; mainly that the axes are fixed to the participants' body location. Therefore the accelerometer axis arrangement does not coincide with the floor axis and changes with the movement of the participant.

The factor of different speeds during gait analysis played a role in studies conducted by Hodt-Billington's research group. The overall consensus is that gait parameters can be measured reliably and effectively when participants choose their preferred walking speed, as previously noted by Aminian et al. (1999). Nevertheless, Liikavainio's (2010) group demonstrated that investigating gait at different speeds might show an effect on the participants' ability to execute normal gait through compensation strategies. Hodt-Billington et al. (2011) looked at the impact of different speeds after surgery. They requested participants to walk at their preferred speed and then at a slower and a faster speed. The instructed walking speed did not seem to cause a lot of difference overall to the improvement of gait velocity. For the patients' preferred speed the results improved significantly from the third to sixth and to the twelfth month, but for the faster speeds, a significant improvement after surgery mostly occurred between the third and sixth month, while for slower gait the main improvements occurred from the sixth to twelfth month. Like Kahn et al. (2013) and Calliess et al. (2014), Christensen et al. (2015) demonstrate not only the effect of increased task demands on gait symmetry measures during stair descend but also the wearying effect of longer walking periods with the six-minute walk test.

While Hodt-Billington et al. (2011) study looked at surgical outcomes, Hodt-Billington et al. (2012) compared the end-stage primary OA patients with a healthy participant group and found the patient group to show significantly more gait asymmetry than the control group. These changes were detectable for trunk acceleration in vertical, medial-lateral and anterior-posterior directions. Furthermore, the patients' steps were longer and the single support duration on the affected limb was shorter. No judgment about the Atallah et al. (2014) studies' ability to distinguish between OA patients or healthy controls can be made. The patients recruited for the study were either only tested twelve months after surgery when most measures of gait symmetry are on the same level as

the control group or were knee OA patients whose disease was not specifically stated to be unilateral.

Bolink et al. (2012) looked at end-stage knee OA patients using accelerometers and gyroscopes. They not only assessed gait data but also tested standing up from a chair and block stepping. Overall the patients were slower at the task than the control group. During walking they showed worse step length and irregularity, but not asymmetry and their range of motion (ROM) was increased in the sagittal but decreased in the frontal plane. When sitting up patients showed increased ROM, but no significantly different peak acceleration. Vertical and anterior-posterior peak acceleration and asymmetry scores for block stepping, on the other hand, were higher for OA patients, with the affected knee showing much more sagittal ROM. The authors also calculated the areas under the curve to evaluate the sensitivity and specificity of the functional parameters they investigated. They found the significant temporal parameters to reach good to excellent scores, while step time irregularity, ROM and peak acceleration only received fair to poor scores. The exceptions are the good scores for ROM of the gait analysis and the sagittal ROM for the affected leg in block stepping. In a later study, Bolink et al. (2015) conducted a gait study where they included a hip OA patient group, as well as knee OA patients and healthy participants and added pelvis asymmetry and ROM as measures. This time the knee OA group showed a significant difference of step time asymmetry to the control group, though their pelvis asymmetry was not larger. The hip OA group, on the other hand, lacked significant differences in step time irregularity, asymmetry and even in the purely temporal parameter cadence and step time to the control group. Interestingly the study revealed a significant difference between the hip and knee OA groups with a worse step time asymmetry for the knee OA group and decreased pelvic rotation and increased pelvic asymmetry for the hip OA group. Overall the authors demonstrated that a reasonable distinction between patients with different diseases or healthy individuals can be drawn with the right task and functional parameters.

How variable knee OA patients' gait symmetry can be was demonstrated in both of the studies conducted by Bolink's research group. However, this difference might have been due to the more careful selection of unilateral knee and hip OA patients in the latter study (Bolink, et al., 2015). Barden et al. (2016) used only bilateral knee OA patients and healthy controls and found no significant difference in gait symmetry between the groups. Furthermore, they used an unbiased autocorrelation procedure to look both at group differences and the effect of sex on stride and step regularity and gait symmetry. In this case, gait symmetry was calculated using step and stride regularity measures. They found OA patients had less step regularity in the vertical and anterior-posterior plane than controls, while there was a significant interaction effect between groups and sex in the medial-lateral plane. Overall the control groups had higher step regularity, except for the men in the medial-lateral plane. The other measures did not reach significance between sex or groups and neither did stride or step time.

Another study that used unilateral knee OA participants to conduct a spectral analysis of the difference between trunk and limb acceleration during gait was conducted by Staab et al. (2014). Their analysis looked at the number of amplitudes at each frequency as calculated with the Fourier transformation. While the control group's spectrum showed two distinct peaks, one each for the

accelerometer and gyroscope data, the patient group's gait asymmetry can be identified by an additional accelerometer peak overlapping the gyroscope data. The method has the potential to identify pathological gait. Furthermore, they found the gait parameters of gait velocity, cadence, trunk symmetry and limb symmetry to be significantly different between groups. While trunk symmetry was assessed using spectral analysis, limb symmetry judgements were based on footfall measures. The authors proposed that the contradictory higher OA group results for angular degree and time of limb symmetry were due to the compensation through the healthy knee in unilateral knee OA. Compensations through the healthy knee in unilateral knee OA may, therefore, be an important factor for the presence of gait symmetry.

Potential reasons for an inability to measure a significant difference between the patients' and the control groups' gait parameters were given by Clermont et al. (2016). In their study patients were requested to walk for 10 min and measures were averaged over nearly the total walking time. They studied temporal gait parameters of participants with knee OA and control participants and evaluated for gait variability with both linear (STD) and non-linear measures (fractal scaling index, FSI). While the stride and step time was significantly slower in the patients, the variability of these measures did not show any significance, neither with the STD nor with the FSI approach. The authors tried to attribute the lack of significant difference in gait variability to their small sample size of only 15 individuals per group. On average the studies with knee or hip OA patients had 28.35 OA participants (STD = 15.78). However, it should also be mentioned that Clermont et al. (2016) did not specify if the patients' knee OA in their study was uni- or bi-lateral.

The relationship between mean walking frequency and OA severity (as assessed by the Western Ontario and McMaster Universities Arthritis Index (WOMAC)), age and footwear were investigated by Barrois et al. (2015) through pelvic accelerometer data. For mean walking frequency, they found a significant difference between the control group and medium or higher severity groups, and between lower and higher severity groups. All results were independent of participants' age and footwear. The authors emphasise the clinical relevance of a short ten-metre walk test that represents a more objective assessment than clinical scores, which must be independent of age and footwear and should be easier to conduct and not reliant upon skin mounted accelerometers. In another experiment, Barrois et al. (2016) divided patients into two severity groups according to their WOMAC scores and tested acceleration and angular velocity in the horizontal plane and on the vertical axis. They only found a significant difference between the acceleration mean and root mean square (RMS) between OA severity and control group in the horizontal axis at the sensory units attached to the legs, but not at the ones attached to the back or head. Walking velocity was not significantly different from the control and low severity group, but the high severity group was distinguishable from both of them.

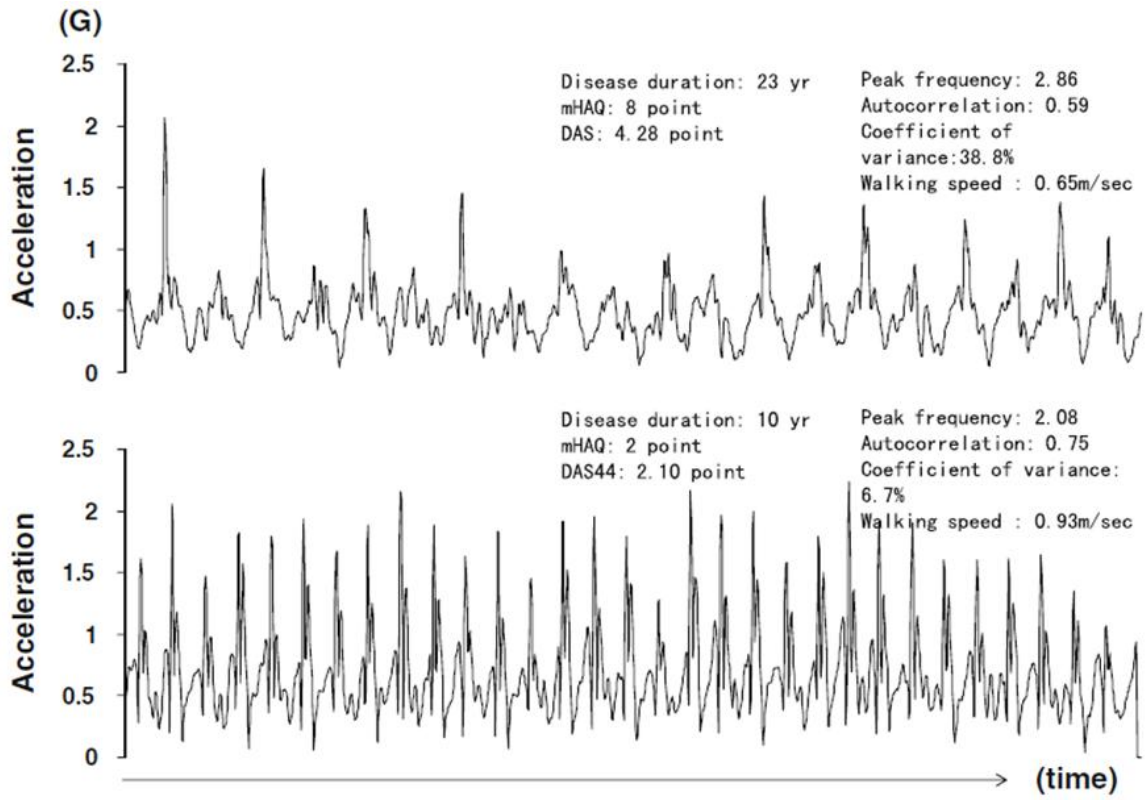


Figure 8 – ***Walking acceleration pattern of RA patients with different disease severities.*** *Adapted from Yamada et al. (2012). Above: patients with more severe RA; below: patient with less severe RA, DAS = disease activity score, mHAQ = modified health assessment questionnaire.*

Yamada et al. (2012) used trunk acceleration data of RA patients to extract the gait cycle peak frequency, the gait balance through the autocorrelation peak, the gait variability through the coefficient of variance and the walking speed. Furthermore, they correlated these values with symptom scores such as the Disease Activity Score (DAS), modified Health Assessment Questionnaire (mHAQ) and walking ability. The peak frequency only showed a correlation with walking speed, which was in itself correlated with walking ability and the mHAQ. Gait balance was correlated with speed and both of the RA assessment scores, while gait variability not only correlated with the DAS but also with both walking speed and ability. They also observed that the more severe the RA, the more irregular the acceleration waveform patterns, as shown in Figure 8.

Turcot et al. (2008) measured the minimum-maximum and range of acceleration of the gait cycle's first and second acceleration peak in all three dimensions and found a significant difference between the OA and healthy control group in the loading phase. More specifically the peak acceleration of the femur in the anterior-posterior direction showed an increase of acceleration in OA patients, even when controlled for bodyweight of the individuals. Liikavainio et al. (2010) used accelerometers, force-plates and EMG in their gait analysis. They found no significant difference between the knee OA and healthy control group during moderate level walking in the heel-strike IPA, the peak-to-peak acceleration and the ratio of acceleration transmission (RAT). OA severity was not correlated with these values either. However, the OA group showed a significantly higher IPA for the accelerometer below the knee when walking the stairs. Overall IPA and peak-to-peak range increased when the participants' level- or stair-walking speed increased. Also, the RAT was lower for OA participants during faster stair descend. Together with the finding that patients used

different strategies for walking, as shown by the EMG, the authors conclude that OA patients load their lower legs more forcefully and that exhausting the compensation mechanisms at higher speeds leads to the more pronounced difference between the two groups.

The three studies conducted by Rouhani's research team found that their patient group showed significantly lower force amplitudes (global maxima and minima) during gait cycles in the anterior-posterior and vertical, but not in the medial-lateral directions. The moment in the sagittal and transverse plane and the power of the ankle was also decreased in the patient group. The authors tested for error, sensitivity and repeatability of the measures and found only the coronal moment to be lacking, even though power and force showed only low error values (Rouhani, Favre, Crevoisier, & Aminian, 2011). To ensure the validity of the new wearable system in their 2012 study (Rouhani, Favre, Crevoisier, & Aminian, 2012), they used a group of three healthy participants to validate it against the stationary system. The rotation ability of the patients between the shank, forefoot and hind-foot was significantly lower in all planes. Finally, in the third study (Rouhani, Favre, Crevoisier, & Aminian, 2014) they found mostly the same significant decrease of amplitude in the patients for both the joints around the hindfoot, though they could not find a difference in the moment of the transverse plane and did not bother to report the coronal plane anymore. For the fore-foot toe joint, only the power and anterior-posterior force was significantly depressed in the patient group. Overall the ankle OA patients' joint kinematics of acceleration force, moment and power performed worse than the control group. Using the gyroscope as an additional sensor-enabled them to also make judgements about the angular rotation of the joint, a measurement that is not obtainable with an accelerometer.

Another study investigating acceleration power was by Lyytinen et al. (2016). They tested knee OA patients during level and stair walking. IPA, acceleration slope preceding IPA (named maximal acceleration transient rate (ATR)) and the mean acceleration 50 milliseconds (msec) before and 300 msec after IPA (named RMS acceleration) were calculated using skin mounted accelerometers to assess impulse loading. Their patient pool of only nine individuals per group led to high STDs in the reported data. Nevertheless, reconstructing the significant levels of the reported accelerometer data revealed increased acceleration peaks in the patient group for the z-axis (vertical direction) and the resultant acceleration but not averaged over the x- and y-axis (medial-lateral and anterior-posterior directions) during level-walking. RMS acceleration was also significantly larger for the OA group, but the acceleration slope was not. None of these measures was significant for walking upstairs, even though looking only at the means and not the STD they clearly appeared to be higher for the patient group. Downstairs walking showed a significant increase in acceleration for patients' overall values. The paper by Lyytinen et al. (2016) was only included in this review because they reported their acceleration data. Their main goal was to assess the repeatability of their methodology; hence, they only processed a small participant count. Their focus also was on peak acceleration, a measure much more likely to show a significant difference between patient and control groups than gait symmetry.

2.4.6. Activity Pattern Identification and Classification

Vissers et al. (2011) used established activity monitoring methods based on accelerometer data that identified body posture and motion to single out 10 random walking periods with a minimum of 10 gait cycles and 20 chair rising moments. It is the first study selected in this review that used data collected from the participants' daily life. Furthermore, they identified how long and often the participants spent in walking or sitting positions and counted the number of longer or shorter periods in these activities. The intensity of movement during walking was measured as the body motility of the participant. They compared the participants' behaviour before and six months after surgery and to a control group. The results showed that stride frequency, body motility and chair rising time was worse in pre-operative patients compared to the control group, but the surgery led to significant improvement in all three aspects of the movement. However, the percentage of time post-operative patients spend on movement-related activities or walking decreased slightly and was significantly different from the control group. The authors demonstrated that improvement after surgery cannot just be evaluated by an increase in physical activity levels in their daily life, but that looking at the improvements in physical functioning is necessary as well.

Kobsar et al. (2016) gave an insight into the way identifying gait parameters could be a valuable resource for treatment decisions. They attempted to use accelerometer data and the Knee injury and OA outcome score (KOOS) questionnaire to divide knee OA patients into three groups according to their responsiveness to hip strengthening exercise. The data was recorded from the affected knee at the foot, shank and thigh, and the back. After the intervention participants were sorted into the responder groups according to the absolute change in their KOOS scores. The best classification accuracy for a single sensor of 74.4% was achieved by the thigh sensor, though combining the back, thigh and shank sensors increased the result to 81.7%. These results indicate that accelerometer data could help to identify individuals that might respond better to certain physical exercise interventions. Identifying potential intervention success for the individual patient could not only save the patient time and money but could also aim to prevent negative treatment effects since non-responders' KOOS scores decreased on average after invention. Including an objective measure is especially helpful since judging the response groups on the KOOS score alone only lead to 53.3% of accuracy and accounted for only 25.6% of the final result.

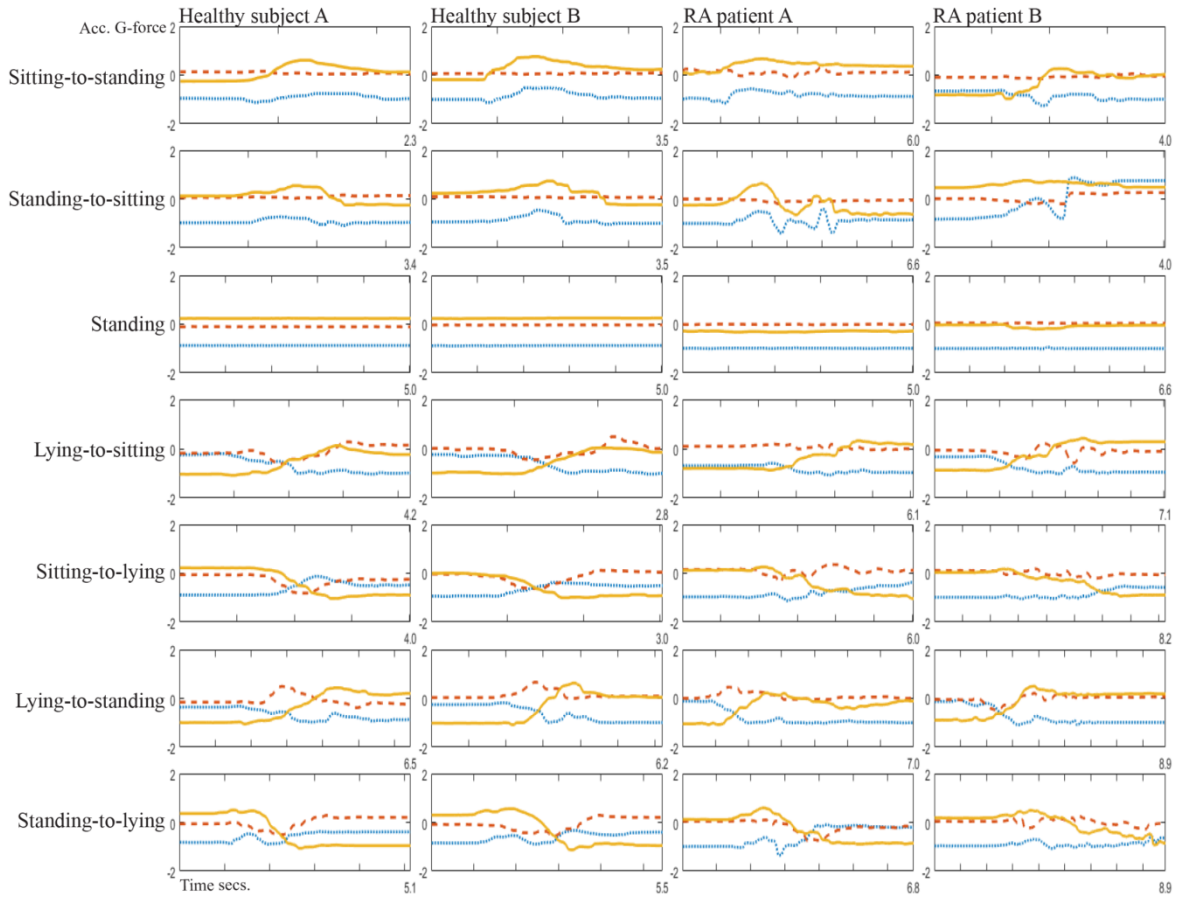


Figure 9 - ***Accelerometer data examples for various transitions.*** *Taken from Andreu-Perez et al. (2017). Sensor location on the lower back (L5 level), x-axis (blue) corresponds to vertical, y-axis (red) to medial-lateral and z-axis (yellow) to anterior-posterior acceleration.*

Recently, machine learning approaches were used to identify transition periods. Andreu-Perez et al. (2017) recorded the acceleration forces and rotation angles from a single accelerometer placed on the lower back. However, instead of using gyroscopes and magnetometers to record angular rotation, they estimated these values from the accelerometer data alone. Being able to calculate this data without the use of gyroscopes would save recording memory and battery and enable longer studies. However, the authors do admit to the resulting loss of accuracy. An example of the recorded accelerometer data can be found in Figure 9. Overall the healthy control groups' acceleration data is unmistakably more fluent than the RA groups' acceleration data. That could be explained by movement irregularity and change of posture due to pain avoidance behaviour. The data collected for these activities was first processed into signal descriptors that contained information about acceleration mean, variance, RMS, peak count, acceleration range and many more. Mapping learning models were used to select a variable subset for the classifiers' training and the Dichotomous Mapped Forest (DMF) method with either Metric Learning Mapping (DMF-metric) or Deep Learning Features (DMF-DL) was used for tag classification to recognise a true label for the data transitions. The classifier, in this case, is based on a hierarchical procession, where the to-be-labelled data goes through multiple independent classifier nodes. For example, walking activity would first be classified into the "continuously active" node, and then in the "cyclical" node before the "walking" prediction is made at the leaf. Different methods were used for activity recognition, namely the Radial Basis Support Vector Machines (RB-SVM), the Random

Forests (RF, also with RF-metric and RF-DL variations), the Convolutional Deep Belief Networks (CDBN) and continuous Hidden Markov Models (HMM). Furthermore, a logical filter was applied to the data to discount infeasible labelling of data. For example, while the possibility of a lying-to-sitting transition can be followed by a sitting activity, a sitting-to-standing or a sitting-to-lying transition, but not by a standing activity. The logical filter increases the confidence in the prediction made on unlabelled data. Also, the effect of different training sets made up of the use of A) both RA patients and healthy participants, B) only patients and C) both the patients and controls but the feature selection was only done on patients and the mapping only on healthy participants to prepper classifier training. The labelling accuracy of the separate RA patient test set data was reported for the different methods. The most accurate activity recognition was achieved by using DMF approaches (about 90%), applying a logic filter even increased the accuracy further (to about 94% for DMF-DL). Notable, B increased the accuracy of the DMF-metric approach but decreased the DMF-DL approach. Again applying a logic filter increased the accuracy, with the DMF-metric approach of set C with a logic filter achieving 90.37 % accuracy.

2.4.7. Non-Gait related Mobility Tests

Not a lot of studies looked at measures other than gait functionality to assess physical function in patients with OA and RA. Chair transitions (sitting down or standing up from a chair) are often added to gait functionality tests. Studies that included chair transitions reported a wider ROM for patients than controls, but no difference in acceleration peaks (Bolink, van Laarhoven, Lipperts, Heyligers, & Grimm, 2012) or mean acceleration (Khan, et al., 2013).

The other studies belonging to this chapter mainly focus on the joint rotation ability and balance of the patients. The first attempt to investigate arthritis joint rotations with accelerometers had the goal to find an easier and cheaper method to distinguish between RA and spondyloarthropathy. When filtering for the flexion motion-related sinusoidal wave, the spondyloarthropathy patients showed high-frequency micro-vibrations that were not present in RA patients. Also, they had a larger mean power of the acceleration compared to the RA group (Reddy, Rothschild, Verrall, & Joshi, 2001).

Post-surgical performance of joint rotations was examined by Jolles et al. (2011). They used gyroscope and accelerometer data to collect angular velocity, power score and moment score of the shoulder in people with OA, rotator cuff disease and healthy individuals. They found the data to correlate with the Simple Shoulder Test, especially with the power score, which is based on both gyroscope and acceleration data. However, it is stated that the clinical use of kinematic sensor data in their study is based on the observation of patients' post-surgical progression and not on the diagnostic or differentiation between pathologies. Nevertheless, the data can be a good indicator of pain-related limitations in physical functioning.

Furthermore, individuals with knee OA were found to show an increased body sway caused by knee joint instability. Turcot, Hagemeister, de Guise and Aissaoui (2011) were able to demonstrate this sway by collecting linear accelerometer, angular velocity gyroscope and centre of pressure force-platform data. They found the centre of pressure to have a lower velocity in OA. Also, the accelerometer range was higher in the sagittal (anterior-posterior) and the coronal (medial-lateral)

plane. The additional use of gyroscopes in this study revealed a larger angle of movement of the knee in the medial-lateral direction

2.5. Conclusion

The review illustrates how improvements in technology can lead to more investigative potential in the assessments of pathologies. MEMS made smaller, lighter and cheaper accelerometer devices possible that can detect movements with more accuracy in all three dimensions. Judgements about measures of gait speed, frequency, variability and symmetry can be made with accelerometers at various locations on the body. Most studies placed the devices on the lower back (total number N = 9) or the legs (N = 10). However, 2 studies also recorded data on the upper body half and 7 spread out multiple devices on various body parts. Furthermore, accelerometers have proven to record data that can hold up to data collected with laboratory-based, more expensive and complicated equipment such as force-plates, electronic walkways and camera systems.

The review showed the importance of tracking patients' disease progression after surgery to evaluate treatment effects. Especially after TKA, the self-reported perception of surgery benefits can be recorded early onwards, while some physical functions take up to a year to show significant improvements (Hodt-Billington, Helbostad, Vervaat, Rognsvåg, & Moe-Nilssen, 2011). Since pathological gait can be more easily detected when greater stress is put on the body, laboratory investigation of gait might not return very realistic results of the patient's day to day physical abilities. It is furthermore worth pointing out that gait analysis can help to determine the patient's suitability to exercise treatment approaches as demonstrated by the Kobsar et al. (2016) study.

Measurements of temporal gait parameters such as walking velocity and cadence can help to identify individuals that have problems with their joints, even if they might not admit to it or are not aware of them as unusual and therefore are not searching for treatment. Furthermore, they could identify cases in which patients use compensatory strategies that don't keep up with more exhausting activities, like long walks, sprinting or stair descending. In addition, investigating gait variability and symmetry can be used to monitor treatment progression since they are parameters that take longer to adjust after surgery. However, measuring step symmetry led to some quite different results in the selected studies. Some studies found a significant difference to the control group (Hodt-Billington et al., 2012; Staab et al., 2014), an improvement after surgery (Aminian et al., 1999; Hodt-Billington et al., 2011), a correlation with disease severity (Auvinet, Chaleil and Barrey, 1999) or even between different kinds of OA (Bolink at al., 2015). However, other studies reported no surgery improvements or group differences (Bolink et al., 2012; Barden et al., 2016, Christensen et al., 2015; Atallah et al., 2014). With others, it is questionable if they should be counted as finding symmetry differences or not. Van Hemert et al. (2009) compared improvements between two kinds of surgery, and Bolink et al. found a significant difference for block stepping but not for gait in 2012 and for step time, but not for pelvis symmetry in 2015. In the latter case, they were even able to use pelvis and step time symmetry to distinguish knee and hip OA. In general, studies seem to be more likely to report symmetry differences if they look at unilateral over

bilaterally affected patients and when specific sections of the gait cycle were selected, such as the stance, double support or single support phases.

In conclusion to the literature review, we propose that wrist-worn accelerometers could be used to paint a more comprehensive image of a patient's actual physical abilities. Accelerometers are commonly integrated into smartwatches, which could already be owned by a study participant or patient. In such a case, it would provide a cheap diagnostic tool for continuous tracking. For example, a machine learning approach can be applied to identify activity episodes and determine gait parameters of patients under different conditions (e.g. during normal speed walking or running), during different activities (e.g. at level- or stair-walking) and over different periods of time (e.g. comparing morning to evening performance). Furthermore, judgements about limb rotation could be made through the use of gyroscopes. Limitations of the accelerometer, such as the undeterminable exact axis orientation in relation to the ground, can be compensated for by added manometers and barometers. Together these different sensory devices are more commonly getting integrated into IMUs and therefore becoming more accessible as disease tracking devices.

2.6. Literature Review Update

The use of accelerometry has steeply increased since the formal literature review was completed. Our new search identified 63 papers published between January 2018 and December 2021. If we do not filter out "physical activity" in order to focus on physical function, we even identified a total of 166 articles. This means a similar number of articles were written in the last three years as was in the preceding 40 years. The following chapter will discuss interesting observations we made while scanning their abstracts. We were astounded, however, by the increased number of studies (9 in total) using accelerometers for studying the health of dogs.

Unsurprisingly accelerometers are still used for evaluating physical functioning in patients. For example, Homma et al. (2020) investigated trunk and pelvis motion in hip OA patients with abnormal gait by straightforwardly extracting measures like gait speed, ROM and step length. Van der Straaten et al. (2020) validated the use of inertial sensors with a camera-based motion analysis system and were able to identify differences between knee OA patients and healthy controls. Another interesting finding was Kluge et al.'s (2018) ability to predict TKA responder groups using pre-surgery gait parameters with an accuracy of 89%.

Approaches going beyond straightforward measures are also on the rise. Brenneman and Maly (2018) used principal component analysis (PCA) to identify changes in gait waveforms in women with knee OA. An increase in machine learning approaches can also be seen, such as in Kobsar and Ferber (2018). They used a one-class support vector machine on accelerometer data collected on the lower back, thigh and shank to identify changes in OA patients' gait patterns pre- and post-intervention. Hartog, Harlaar and Smit (2021) trained machine learning algorithms to identify stumbling events recorded by an IMU at the shank during treadmill walking and while performing tasks of daily living. They proposed using their methodology to help treat the walking pattern of patients with OA but trained and developed their approach using recorded behaviour of healthy individuals only.

Accelerometry is further used to access treatment methods. Tomite et al. (2021) included accelerometer analysis of anterior-posterior acceleration to assist in their outcome comparison between anterior cruciate ligament TKA and bi-cruciate stabilized TKA. They concluded that bi-cruciate stabilized TKA lead to better outcomes. Çankaya, Aktı, Ünal and Sezgin (2021) compared TKA to unicompartmental knee arthroplasty (UKA) by using a Samsung Galaxy Note 10 Plus to record measures such as gait velocity, step time symmetry and step length symmetry. They found UKA to lead to better gait patterns than TKA.

The use of smartphones in the context of eHealth research is steadily increasing. In Germany, the RECOVER-E app is an evidence-based mobile app that is used to educate patients after total knee or hip replacements (Stauber, et al., 2020). But eHealth can go further, for example, smartphone applications can be used for tracking patients' symptoms. Nowell et al. (2021) investigated which pain-related outcomes RA patients preferred to track digitally. They identified sleep disturbance, fatigue, joint stiffness duration in the morning, pain interference, pain intensity and physical functioning as the most important outcomes to track. When developing a smartwatch application to track patients' symptoms these measures should be targeted. Crawford et al. (2021) demonstrated that a postoperative smartphone-based care management system can be as sufficient as traditional post-surgery care models. In particular, using smartphone applications reduced the need for in-person physiotherapy. Care provided using smartphone applications can reduce post-operative costs and was not found to be inferior to traditional care (Crawford, et al., 2021). Goniometers within smartphones can be used to collect ROM measures. Mehta, Bremer, Cyrus, Milligan and Oliashirazi (2021) demonstrated that even inexperienced examiners were able to use their app to extract reliable ROM measures. Another example of making clinical diagnostic tools more accessible is the remote automatic system developed by Zhang and Zhou (2021) that identifies a patient's KSS score and sends it automatically to their doctor.

Further research was done to track patients' physical functioning in the free environment. Wouda, Jaspar, Harlaar, van Beijnum and Veltink (2021) developed an approach to determine the foot progression angle of patients with gait disorders outside of laboratory conditions using an accelerometer and gyroscope sensor worn on their foot. Receiving feedback about their foot progression angle in real-time can support patients in reducing their knee adduction moment. He, Lippmann, Shakoor, Ferrigno and Wimmer (2019) used a pressure detecting insole in combination with auditory clues to detect knee adduction movement. Training participants using the auditory clues significantly reduced adduction. Accelerometers can also be used covertly in studies. Nicolson, Hinman, Wrigley, Stratford and Bennell (2019) investigated exercise adherence by embedding accelerometers in an ankle cuff weight used for the exercise. Surprisingly, they did not find a link between adherence and changes in reported pain or function assessed through the WOMAC.

More important for the scope of this thesis are the findings of Vangeneugden et al. (2020). They continuously tracked participants for between a week and 10 days with an accelerometer taped to the upper thigh. While they found that the overall activity profile was not significantly different between healthy controls, lean knee OA and obese knee OA patients, stride interval dynamics did show a significant difference between the healthy and patient groups. They propose that their

findings could be used for non-intrusive knee OA diagnosis. This paper illustrates the potential of continuous tracking as a diagnostic tool for OA

2.7. Aims for Further Studies

The initial literature review was helpful in guiding our focus to walking behaviours as a measure of physical functioning. In the initial search, we discovered a lot of studies that looked for physical activity in free-living conditions using measures of step counts or activity levels. Physical functioning was mostly only observed or measured in laboratory settings. Furthermore, only one study (Vissers, Bussmann, De Groot, Verhaar, & Reijman, 2011) looked at physical functioning and recorded data without instructed behaviours in the free environment. The general trend of the last few years goes towards moving accelerometers out of the laboratory settings towards continuous tracking in the real world. While Vangeneugden et al. (2020) stress the benefit of their extensive setup including multiple sensors as being a non-invasive method of identifying OA, it is still not a practical setup or a very accessible one. The next step would be to use devices more easily accessible to patients, such as the use of smartphone apps by Crawford et al (2021). Studies collecting accelerometer data from the wrist location did not increase in recent years. Of the 166 studies found in the literature search update, all of the 6 studies that used wrist-worn accelerometers focused mainly on physical activity measures and not physical functioning. It is uncertain whether this is due to a lack of attempts or a lack of success in extracting gait parameters from wrist-worn accelerometer data. The goal of this thesis henceforward is to examine the feasibility of using wrist-worn accelerometers to extract gait parameters.

In the review, we demonstrated how technical advances in accelerometry lead to a cheaper and more accurate investigation of patients' physical functioning. We see the application of apps in smartwatches as the next step for the use of accelerometry in the assessment of physical functioning in patients with RMDs. Consumer devices such as smartwatches and Fitbits that are capable of recording accelerometer data have the potential to provide the setup needed for tracking behaviours without additional need for hardware. Enabling patients to track their physical functioning via an app on their own smartwatch offers the benefit of a diagnostic tool at a reduced cost to the healthcare system. This could mean a reduced need for doctor’s visits and provide a more realistic everyday reading of the patient's capabilities. If the physical functioning of the patient can also be related to their pain experiences, a passive method of tracking wellbeing and medication effectiveness could be inferred. However, these benefits rely on the patient possessing a suitable consumer device. Accessibility is therefore a point of debate and would favour wealthier and younger patients. Nevertheless, tracking patients via consumer devices is becoming more prevalent, such as with the RECOVER-E app. Steps in this direction are connected to the rising trend of eHealth in healthcare practice.

Our most optimistic aim would, therefore, be to develop a diagnostic tool. For example, Reddy, Rothschild, Verrall, & Joshi (2001) were able to use accelerometry to distinguish between RA and spondyloarthropathy. The value of their discovery lies in the easiness and cheapness of the methodology. However, achieving this with a wrist-worn accelerometer is unlikely due to the location of the sensors; since the data collected needs to be of very high quality. Therefore,

focusing on extracting data with high quality should be a priority. But even if this does not turn out to be feasible, tracking an individual patient's daily physical functioning can be of value for them, since a decrease in physical functioning can be both a risk factor and an indicator of worsening disease progression. An app could also be programmed to regularly request the patient to perform certain behaviours to test specific physical functioning or regularly fill our diagnostic questionnaires. Potential tests that could be integrated are the KSS, Minimod Gait Test or the six-minute walk test.

After surgery, the tracking of physical functioning is of particular importance. Having regular access to a diagnostic tool can therefore be very valuable to the patient's recovery. For example, Van Hees, Slootmaker, De Groot, Van Mechelen, & Van Lummel (2009) found a change in physical function after TKA even when the patients did not report symptomatic improvements anymore. Recording of patients' physical functioning in a free-living environment also allows for longer monitoring and consequently can track deterioration of functioning throughout the course of the day. Recovery of more demanding physical functions after surgery can be overlooked when only relying on standard physical functioning tests in the clinic, as suggested by Khan, et al. (2013) and Calliess, Bocklage, Karkosch, Marschollek, Windhagen and Schulze (2014). Furthermore, after detecting gait abnormalities treatment options could be recommended. For example, testing for the requirement of insoles, or training healthier walking behaviour by sending warning signals to the patient. Especially symmetry parameters are likely affected when the disease unilaterally affects one side of the body.

To summarise, we believe that our focus should lie on bringing the investigation of gait parameters as a measure of physical functioning into the world of smartwatch applications in order to make an accessible, passive and continuous diagnostic tool. The next chapter will look into different accelerometer datasets used in this thesis. The literature review was important in guiding our focus onto the collection of wrist-worn accelerometer data in Chapter 4. Within the step parameter extraction algorithm development (Chapter 6), it helped us in identifying step parameters to extract and especially guided our attempt at gait symmetry extraction.

3. Overview of the Accelerometer Datasets

Within this thesis we worked with the data extracted from three sources: the Knee OsteoArthritis: Linking Activity and Pain (KOALAP) study, the self-recorded dataset (SRDS) and the UK BioBank database. We first gained access to the data in the KOALAP dataset. Table 3 below displays a summary of the important characteristics of the datasets.

Table 3 - Overview of the Accelerometer Datasets. SRDS = self-recorded dataset, KOALAP = Knee Osteoarthritis: Linking Activity and Pain; RMD = rheumatic and musculoskeletal diseases, OA = osteoarthritis, RA = rheumatoid arthritis.

	Name	SRDS	KOALAP	UK BioBank
Dataset	**Collection dates**	February to July 2019	September to December 2017	May 2013 to December 2015
	Total length	4.28 hours	Active Wear Time: 2.18 years	336.46 years
	Length per participant		10 days to 118 days; Mean (SD): 73 (23) days	7 days
Population	**Size**	1	26	17,544
	Medical condition	no RMD	Knee OA	RMDs & matched Controls: OA, RA, gout, others
	Inclusion criteria	accessibility within the department	Age: > 50 years; able to visit Greater Manchester; self-reported knee OA diagnosis; own an Apple/Android smartphone	Age: 40 to 69; Accelerometer data was collected; Have a RMD or are a matched control (year of birth, sex, Townsend score)
	Demographics mean (SD)	Age: 24; female	Age: 64.2 (8.8); Sex ratio: 50 %	Born: 1948 (~7 years); Sex ratio: 58.91 %
Device	**Sensory device**	AX3 Sensor (Axivity); Fitbit	Huawei Watch 2	AX3 Sensor (Axivity)
	Sampling rate	50 Hz	50 Hz	100 Hz

As previously mentioned, we originally looked at the Insoles in Response (InResponse), Quality of Life, Sleep and Rheumatoid Arthritis (QUASAR) and the KOALAP studies as potential datasets to analyse within the now called “Centre for Epidemiology Versus Arthritis” department of the University of Manchester. InResponse was not chosen since the sensor was located on the legs and data was cached over every second (sec) to save memory. QUASAR had a similar limitation where data was collected at 50 Hz but cached, processed and saved in 30-sec intervals only. This was done to save the whole duration of each participant’s data collection onto the memory of the sensory device. In both cases, raw data was therefore either not saved or hard to access. In the end, KOALAP was selected over the other studies in the department mainly due to its similar setup to the UK BioBank study and the quantity of data it provided.

In 2017 the data collection for the KOALAP dataset was conducted by the KOALAP research team within the then called “Arthritis Research United Kingdom Centre for Epidemiology”. Data was continuously recorded over a period of multiple months for 26 participants with knee osteoarthritis (OA). In total, we obtained 2.18 years of accelerometer data collected at 50 Hz. Participants were given a Huawei Watch 2 smartwatch containing the KOALAP app. The app requested participants to regularly self-report pain scores and passively recorded sensory data. The active self-reports included filling out the knee injury and osteoarthritis outcome score (KOOS) questionnaire monthly, rating their “Quality of Life” weekly, and reporting daily “morning” pain, “afternoon” pain, pain during “aggravating actives” and pain during “important activities”. At the beginning of the study, each participant had to determine the activities they found most pain aggravating and the most important activities being impaired by pain. The sensory data was passively collected by the smartwatch and consisted of accelerometer data, gyroscope data, magnetometer data, barometer data, heart rate data and step counts. More detail about the study setup can be found in Chapter 7.3 and the publication of Beukenhorst et al. (2019). The first challenge was to wrangle the raw data. The KOALAP app was developed in collaboration with the Google Android Wear team and the smartwatch was only able to store around 4 days of data. Therefore, data was uploaded daily to the Google Cloud database via Wi-Fi connection. The format of the raw sensory data files that we downloaded from the Cloud was not sorted by time or sensory data type and only superficially sorted by participants. Furthermore, each day's worth of data was randomly split into 20 data files. Our wrangling attempts can be found in Chapter 7.5.4. Apart from the use of the KOALAP dataset for our thesis, we furthermore contributed to the publication of the KOALAP engagement paper (Beukenhorst, et al., 2020) by extracting the participants’ smartwatch wear times (see Chapter 7.5.4.5). We used the KOALAP dataset to investigate the distribution of step parameters that we extracted using our gait detection and step parameter extraction algorithm. Furthermore, we looked at correlations between the step parameters extracted from the accelerometer data and the self-reported pain scores of the participants.

While our initial plan was to develop our own gait detection algorithm using the KOALAP dataset, we quickly had to realise that it was hard to find gait patterns within the accelerometer data. The gait patterns we did manage to extract using the Switching AR (see Figure 21, Chapter 5.2.1) were often quite dissimilar to the patterns we expected to see referencing our literature review. It was, therefore, necessary for us to get more acquainted with gait patterns recorded at the wrist location. We, therefore, planned the recording of the SRDS in order to achieve the necessary familiarity. Due to time constraints, we did not look for participants but used ourselves as subjects for the recording of the dataset. We made a plan of different walking behaviours to investigate and record non-walking reference data. In the end, we successfully recorded data during four distinct sessions: a staircase walking trial, a corridor walking trial, an outside walking trial and a non-walking coffee trial. During the staircase trial, we walked up and down a set of stairs with and without shoes with heels. The corridor trial was conducted similarly, with the addition of walking while caring a cup of water and while caring a bag over the shoulder. During the free walking trial, we went outside and walked continuously for one hour. We also added confounding behaviours to the walk, such as using our hands to readjust loose hair strains, using a smartphone, or caring a

shopping bag. The coffee trial consisted of sitting and animatedly talking with a colleague so that we could ensure that the trial did not contain gait behaviour. This trial served as a non-gait behaviour data reference. A more in-depth description of the trials and methods can be found in Chapter 4.3. In total, we recorded 78.21 min of data during the non-walking trial and 69.93 min of walking trials. While we were recording the dataset, we mostly focused on having reference data to help us label behaviours within the KOALAP dataset. However, hand-labelling sections of the KOALAP dataset turn out to be ambitious but futile. In consequence, we extended the use of the SRDS to be our main asset in the development of our gait detection algorithm and we used it to explore the performance of the algorithm. We used the SRDS to test various common gait detection methods and to make observations about the pattern of gait behaviour recorded at the wrist. These observations help inform the development of our own gait detection and step parameter extraction algorithm (see Chapters 4.4 and 5.2.1). Especially exploring the features of the patterns of the different gait behaviours helped us develop the dominant axis selection approach (DASA), which is a novel approach to assist gait detection from wrist-worn devices (see Chapter 5.3.5). Initially, we wanted to use the SRDS as an opportunity to compare the accelerometer data collected from the KOALAP app on the Huawei Watch 2 consumer device with the research-grade Axivity AX3 Sensor. The latter was also used for data collection in the UK BioBank dataset. Disappointingly, we were not able to use the Huawei Watch 2 provided by Google, since the use of apps on the watch was blocked after the end of the study. Instead, we recorded step count data using a Fitbit to examine the accuracy of step counts on a common consumer device (see Chapter 4.4.1).

While the KOALAP dataset gave us the possibility to explore the relationship between pain and step parameters, the UK BioBank offered us the possibility to compare the extracted step parameters between RMD patient groups and matched control groups and between RMD patient groups. Based in the United Kingdom, the UK BioBank's goal is to investigate the role of genetic predispositions and environmental exposures in the development of a wide range of medical conditions and diseases. Since 2006 they have been following around 500,000 volunteers between the ages of 40 to 69. Around 100,000 of these volunteers were also equipped with a research-grade accelerometer sensor (the Axivity AX3 Sensor) for 7 days. Data was collected at a frequency of 100 Hz. It differs from the other two datasets, which were both collected at 50 Hz. For more information please see Chapter 8.3 and the UK BioBank's "Physical activity monitor: description and methods" resource (Brage, et al., 2016). The major advantage of the UK BioBank dataset is the population size, something that was lacking in both the SRDS and the KOALAP datasets. We selected 17,544 participants from the UK BioBank accelerometer dataset and sorted them into 8 groups: OA, RA, gout, other RMD conditions (other), and healthy matched control groups for each group. The control groups were matched according to birth year, sex and Townsend scores. We identified 6,459 OA participants, 833 RA participants, 1,121 gout participants and 570 other participants. We used both the KOALAP and UK BioBank datasets to evaluate the effectiveness of the step parameters we managed to extract using our own algorithm. In particular, we examined our ability to find differences between the patient and control groups using our identified step parameters of interest on the UK BioBank dataset (see Chapter 8.4). One problem we encountered

when using the UK BioBank dataset was downloading accelerometer data. In general, the process between being granted access to the dataset and being able to finish downloading the dataset can be quite long (see Chapter 8.3.2).

At last, we would like to mention some points about our data handling to make reading this thesis easier. Most data extraction, analysis and visualisation were done using MATLAB R2018b. We also used the Interactive Computational Shared Facility (iCSF) of the University of Manchester to run our final data wrangling, gait detection and step parameter extraction codes on the KOALAP and UK BioBank datasets. The SRDS and algorithm development was mostly handled outside the iCSF. The complete gait detection algorithm and step parameter extraction codes can be found in the appendix. We decided to visualise the use of Matlab codes within this thesis by using the font `Courier New`. We will, therefore, refer to features and variables used as input or extracted from the accelerometer data using Matlab by both their descriptive name (using font: Arial) in their variable name (font: `Courier New`) interchangeably within the text. For example, the sampling rate of the accelerometer data could be referred to as `var_feq`. Furthermore, Matlab coding scrips are built on "functions". A common function, for example, would be `x_max = max(x)` where the function `max` outputs the maximum value `x_max` found within the variable `x`. Therefore, Matlab would output `x_max = 25` when you input `x = [1, 5, 17, 20, 25, 15, 9]; x_max = max(x)`. Functions are mostly already included within Matlab, but can also be downloaded from the Matlab exchange centre or created. The gait detection algorithm and step parameter extraction codes each consist of the main coding documents called `AstonGDACode_GaitDetectionAlgorithm` and `AstonGDACode_StepParameterExtraction` with 6 and 9 of our own functions needed to run the codes respectively.

The following chapters will first look at the development and observations derived from the SRDS, then explained the development of the gait detection algorithm and step parameter extraction codes. Afterwards, we will look at the KOALAP dataset, the wrangling of its raw data, the extraction of wear times and the application of our algorithm. At last, we will apply our algorithm to the UK BioBank dataset.

4. The Self-Recorded Dataset

4.1. Abstract

When looking for examples of accelerometer data in the literature, most figures display data collected from sensors located on the legs or hips. Finding depictions of accelerometer data collected on the wrist proved to be limited. To get an idea about how sensor location on the wrist affects the recorded data during walking, we recorded a set of walking behaviours ourselves. We also compared step counts recorded by a Fitbit to the real steps taken, in order to judge the step counting performance of a common consumer device.

Data that we were able to correctly label to the second (sec) was combined into the self-recorded dataset (SRDS). It could, therefore, be used to examine algorithm performance or for training machine learning algorithms. We used the SRDS to examine the data pattern and descriptive statistics of different walking behaviours. Our most notable observation was the difference between walking behaviours where the arm “swings” freely and where it is fixed in a “stiff” position towards the body. While the walking behaviour with “swing” conditions reliably displays gait patterns on the y-axis, in “stiff” conditions the pattern can often be observed the most clearly on the other axes, or even on multiple axes. Looking at the features of the moving standard deviation (MvSTD), we discovered that a high MvSTD on the one axis compared to the MvSTD of the magnitude of the accelerometer data (dr) can identify the “dominant” axis displaying the gait pattern. This is important since we recorded the accelerometer data at the wrist location. For sensors located on the legs, the dr is sufficient to reduce the accelerometer data dimensions to one for analysis. But our observations of the SRDS show that this is not the case for data recorded at the wrist.

Furthermore, we applied common gait detection methods. But, none of the methods we tested were satisfying. STD-thresholding proved the most accurate, at a threshold of 0.6 m/sec^2, with 26.32 % of the data being misidentified and a ROC true positive rate of 0.72 and false positive rate of 0.20. The CWT-thresholding overall proved the least accurate, misidentifying 55.70% of the SRDS.

Overall, we determined that creating a gait detection algorithm suitable for data recorded at the wrist location should involve identifying a dominant axis, looking for periodicity in the data and setting a minimum episode length.

4.2. Background

4.2.1. Motivation, Aims and Plans

In the beginning, the plan was to use the data collected in the Knee OsteoArthritis: Linking Activity and Pain (KOALAP) study to test and develop gait algorithms. In order to evaluate how well the algorithms perform, the data needs to be labelled as gait or non-gait. Neither the KOALAP nor the UK BioBank datasets are labelled for walking behaviour. Searching the literature and talking with colleagues in Aston gave us some idea of what gait data should look like. Therefore, we attempted to hand label some of the KOALAP data. However, that proved quite difficult since data

collected at the lower body and the wrist tend to differ from one another (see Figure 10). Notably, data recorded at the wrist is messier than data recorded at the hip or legs. Furthermore, our data is a continuous recording of the participant's full day with a lot of data that shows non-walking behaviours. Most example data we found in the literature was quite clean and had very clear start- and endpoints to the walking behaviour even without looking at labels.

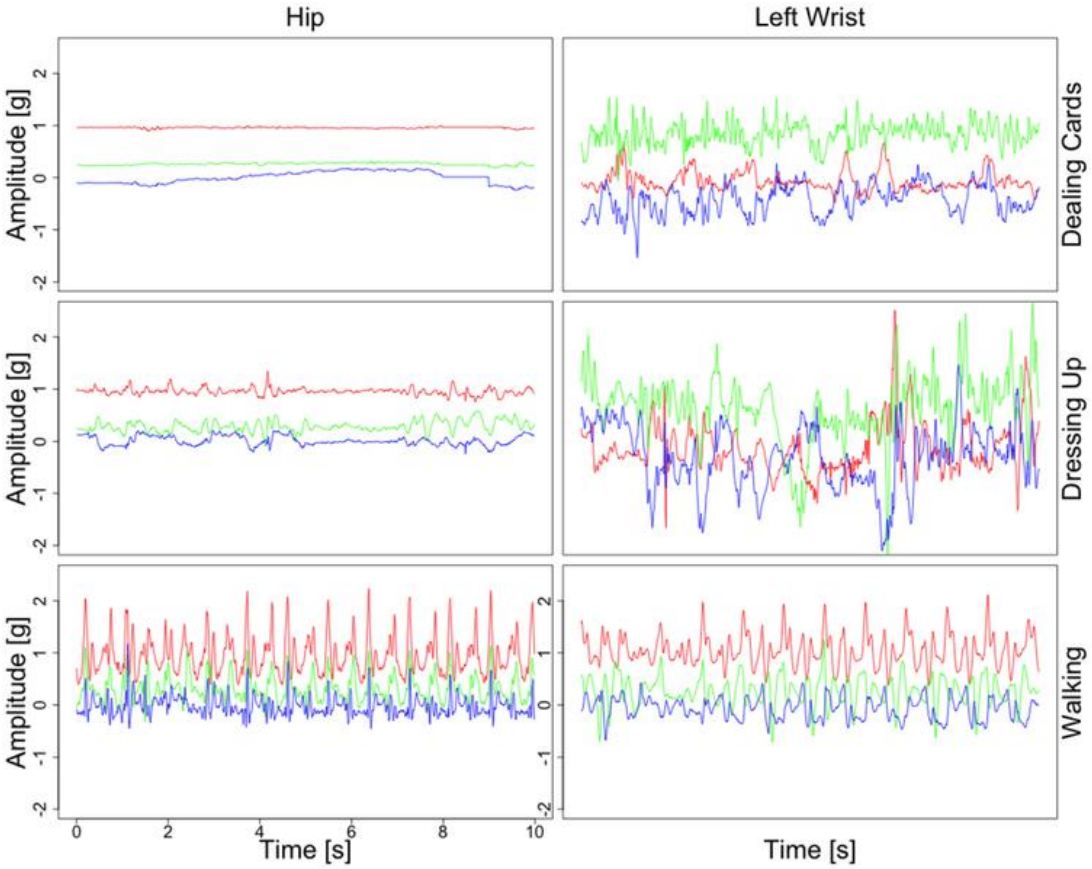


Figure 10 - ***Comparison of acceleration signal recorded at the hip and wrist during different activities****. Taken from Karas et al. (2018).*

Furthermore, while a lot of resources describe the stages of gait and the corresponding accelerometer signal as collected on the lower body in detail (for example, see Chapter 1.4, Figure 3), we were not able to find any exploration about how arm movements correspond to gait and affect the accelerometer pattern. Also, we were interested in how different types of walking or activities during walking might lead to a difference in pattern and found even fewer visual references that would have helped hand-labelling the data. Hence, over the course of the year spend at Aston University we created a self-recorded dataset of various walking behaviours. We set the following aims:

- Get a feel for what walking patterns recorded at the wrist by an accelerometer would look like.
- Examine how different walking behaviours would affect the pattern.
- Produce a labelled data set of various walking behaviours.

The initial plan was to use both the Huawei 2 smart watch (used in the KOALAP study) and Axtivity sensor (used in the UK BioBank study) to compare the two devices. In the Manchester

department, we were also interested if we could identify the reason behind a minor problem with the KOALAP data collection. Our college Dr Anna Beukenhorst and the author wrote a draft for a validation study with the additional aim to compare data quality and validity of the data collected by the KOALAP app with a wrist-worn and shank-worn dedicated research device. However, we could not enable additional data collection with the KOALAP app after the official end of the KOALAP data collection and therefore had to discard the proposal.

Furthermore, the Aston research team was also interested in evaluating the accuracy of the Fitbit as a commercial device used to count steps in research settings. Our interest was piqued after our colleague Dr Yordan P. Raykov observed his step count rising while holding meetings in his office. The smartwatch gait-test protocol was more loosely written than the validation study protocol and we were only able to execute some of it.

In the end, we collected our own accelerometer data during the course of the year in Aston. The following behaviours were recorded and combined into a dataset to test gait detection algorithms on: walking up and & downstairs (with and without heels), conversation in a coffee shop while sitting down (as a control), walking up & down a corridor (including with/without heels, balancing a mug, holding a bag over the shoulder) and walking around town (including carrying a bag with the hand, handling a phone, hands in pockets). In addition, data was collected for driving in a car, a full day with a diary and playing football, however they were not included in the final SRDS due to mistakes made during data collection or labelling. We decided not to re-record the data, but never the less made observations about the excluded trials. The driving trial did not display data patterns that were notably different to resting behaviour or the coffee trial. Behaviours within the dairy trial informed the walking behaviours tested in the corridor trial. In regards to the football trial, we narrowed our focus to walking behaviour and excluded sports activities such as running.

4.2.2. Background for evaluating Axis Orientation

Raw accelerometer data recorded by the AX3 sensor or smartwatches consists of three axes representing the acceleration experienced on the x-axis, y-axis and z-axis (see Figure 11). Cole, Anshari, Lambert, Thrasher and Valafar (2013) illustrate how the axes of accelerometer data recorded by a smartwatch correspond to the orientation of the consumer device. The same axis orientation is also present in AX3 sensors, which can be fixed to the participant's wrist with a wristband (see Figure 12). We can, therefore, relate the acceleration on these axes to the orientation of the sensor within the wristband, the wristband to the participant's wrist, the wrist to the arm movement and the arm movement to the walking behaviour. Movements of the hand going in the direction from the radial (medial, thumb side) to the ulnar (lateral, pinkie finger side) zone lead to an increase in acceleration on the y-axis (lateral). Movement from ulnar to radial leads to negative acceleration of the y-axis (medial). For example, the y-axis axis would see strong changes in acceleration when the arm swings freely next to the body while the palm faces the hip. Moving the wrist from volar (palm side) to dorsal (hand back) leads to an increase in acceleration on the z-axis. For example, reaching to grab something would cause positive acceleration on the z-axis. Movement from the posterior (elbow) to the anterior (finger) will lead to an increase on the x-axis.

For example, pulling open a door would lead to negative acceleration on the x-axis. Figure 11 helps to visualise the axes within a smartwatch and their relationship to the wrist.

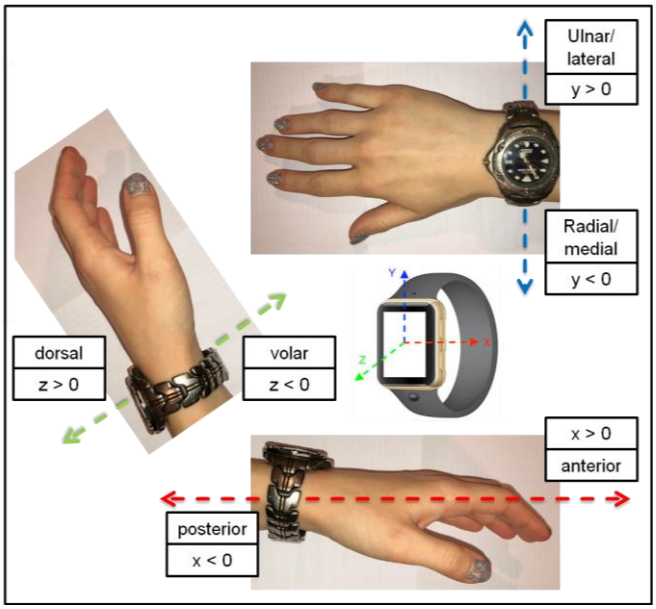


Figure 11 - ***Axis orientation of smartwatches and the corresponding impact of movements in various directions on acceleration****. Adapted from Cole, Anshari, Lambert, Thrasher and Valafar (2013).*

It needs to be noted that the Axtivity (AX3 sensor) accelerometer device does not have the orientation characteristics that a smartwatch has due to its featureless design. Therefore it is unclear if Axtivity data with positive acceleration results from lateral movement (as is the case with the smartwatch) or medial movement. Taking the sensor out of the wristband (as is necessary for reading the data and for charging the device) can also potentially lead to the sensor being put back into the wristband with a different orientation. This can cause the same problem of inferring movement direction from acceleration direction for the other axis as well. However, since the sensor is rectangular we can at least rely on the axis orientations to be fixed.

Furthermore, the exact orientation and location of the sensor can vary between participants or even for the same participant at different times. The KOALAP and UK BioBank studies both rely on the participants to put on the sensor themselves and a placement protocol is not given. The participants could switch between dominant and non-dominant wrists, turn the device, secure the wristband too loose or tight, or attach the watch at different distances to the wrist on their arm. Even if a protocol is provided, it is not guaranteed that it is always correctly executed by the participants. Straczkiewicz, Glynn and Harezlak (2019) found that 15.6 % of their 45 older participants violated protocol at least once, which lead to miscalculated physical activity scores on these days. They developed the “Placement, Location and Orientation Evaluation method” (PLOE) to determine the orientation of wrist-worn tri-axis accelerometers in free-living conditions.

4.2.3. Gait Detection Algorithm

This subchapter will give a short overview of the different data analysis methods used on the SRDS. We tested the following gait detection algorithms on the data: normalised autocorrelation step counting (NASC), STD-thresholding, short-term Fourier transformation (STFT) thresholding, complex wavelet transform (CWT) thresholding and switching autoregressive model (Switching AR). We also attempted to replicate the methodology used in Brajdic and Harle (2013). However, this attempt had to be discarded due to ambiguity over their methodology. For example, it was hard to identify how they defined the “energy of acceleration signal” and our attempts yielded widely different optimum thresholds than the ones reported in Brajdic and Harle's (2013) paper. Our identified optimal *energy threshold* was 1983.63 m/sec^2, while they reported an optimum threshold of 0.04 m/sec^2. We achieved similar results for the *acceleration threshold* (0.87 m/sec^2 vs 10.5 m/sec^2), but ended with similar *standard deviation thresholds* (STD-threshold, 0.69 m/sec^2 vs 0.6 m/sec^2). A STD-threshold of 0.6 m/sec^2 is commonly judged to be an adequate threshold for identifying gait behaviour.

Thresholding is the practice of extracting *features* from the data and setting a threshold at which the data is considered walking behaviour. The threshold deems the data to be either gait or not-gait. To avoid being susceptible to rapid changes in the data profile, these features can be extracted over a *window*. For walking behaviours, a window of 1 sec is usually in line with the average step rate of humans.

A STD is able to express the variability of the data. The dynamic data of walking behaviours consist of spikes coinciding with each step. Also, in order for a gait pattern to be identifiable, the walking behaviour needs to represent the strongest impact on the accelerometer. Noise that results in stronger amplitudes than the gait behaviour itself would prevent us from identifying and analysing said gait pattern. Usually, identifiable walking behaviour causes large spikes in amplitude on one axis and smaller similar spikes on the other axis. These spikes can also occur in opposite directions. For example, while the y-axis has a large negative spike, the x-axis simultaneously peaks in the positive range.

The NASC takes the STD with a *moving window* of 1 sec and conducts an autocorrelation at certain frequency ranges (between 1 and 2 Hz). A limitation of the NASC is that interrupted gait will slip through, causing short occurrences of noise in the data to cut into the walking episodes. This is a particular concern with patient data, whose gait can have an increased ratio of noise by nature. However, if it is a repetitive problem, as is the case with asymmetrical gait, the autocorrelation can catch the behaviour in the frequency range of the gait cycle instead of the step cycle.

The STFT applies a spectrogram to compute the power spectrum of the behaviour. If the maximum of the power spectrum to passes the threshold, the data is identified as walking behaviour. It relies very heavily on the trade-off of choosing a window size. Smaller window sizes will lead to a loss of accuracy while larger window sizes will cause a loss of information. If the window size that is chosen does not correspond to the periodicity of the data, then a periodic behaviour will not be identified as such. A sliding window needs to be implemented to somewhat counteract this limitation.

The CWT and DWT (Discrete Wavelet Transform) are ANOVA techniques that analyse the variance of a time series dataset. It decomposes the signal into sets of data at different frequency bands and identifies the wavelet spectrum. Its advantage is its temporal resolution which enables a precise episode starting point.

4.3. Methods

The data was recorded in various trials between the 22nd of February and the 15th of July 2019. Each trial has a different main focus for the behaviours investigated, but different walking conditions were conducted within the same trial. An overview of all trials can be found in Table 4. The trials will henceforth be addressed by their title given in the table. We selected four trials to be used for the development of the GDA and we combined them into the self-recorded dataset (SRDS).

Trial Title	Date (2019)	Subject	Documentarian	Step Count Documentation Methods	Included in the SRDS
Initial / Diary Trial	22nd Feb 13:00 - 18:30	KSW	KSW	notes app	No
Driving Trial	15th Mar 13:30 - 19:30	KEW & KSW	KSW	notes app, online Fitbit report	No
Staircase Trial	25th Mar 13:55 – 15:00	KSW	RB	excel table, video	Yes
Corridor Trial	08th Apr 19:45-20:40	KSW	KSW	video	Yes
Coffee Trial	10th Apr 15:45 – 17:20	KSW	RB	excel table, video	Yes
Football Trial	04th Jul 20:55-22:25	YQ	KSW	video	No
Free Walking Trial	15th Jul 19:50 – 20:50	KSW	KSW	notes app	Yes

Table 4 - Dates of Trial Recordings. Participants: Katy Sarah Weihrich (KSW), Dipl.-Ing. Kay Erhard Weihrich (KEW), Reham Badawy (RB) and Dr Yazan Qarout (YQ).

4.3.1. Participants and Documentation

Most of the data recorded and all of the data that was selected for the SRDS used the thesis author (Katy S. Weihrich, female, age 24) as a participant. Furthermore, we found willing participants in the author's father (Dipl.-Ing. Kay E. Weihrich, male, age 55) and our colleague (Dr Yazan Qarout, male, age 26). To our knowledge and the participants' knowledge, none of them had any RMDs.

We documented two different kinds of step counts: 1) the real step count taken during the course of the trial as countered by the documentarian; 2) the step count recorded by the Fitbit. For the latter, the documentarian asked the participant to read the current step counter of the Fitbit's display. Documentation was done by our colleague Dr Reham Badawy or by the author. Dr Badawy mainly used an Excel table in which she recorded step counts and recording times. Furthermore, we recorded some trials using the author's smartphone camera. When the author was both the participant and documentarian, we used the smartphones' note app to manually document step counts, or they voiced Fitbit step counts out loud to be recorded on video. Step

counts recorded on the note app or recorded by video were later added to the excel table used for the staircase trial and coffee trial.

“Triggers”, that enable the synchronisation between the recorded labels and the accelerometer data, were set by quickly rotating the wrist. This leads to a very distinct visual with high frequency and high amplitude acceleration changes. Table 4 provides an overview of the trials conducted during the SRDS development. Furthermore, an in-depth description of all trials and behaviours within the trials can be found in Chapter 4.3.3.

4.3.2. Sensor Devices

We used two different sensory devices simultaneously for the recording of the trials: the research-grade AX3 Axivity sensor and the Fitbit Charge 2 consumer device. Both devices were placed on the left wrist, with the Axivity being placed anterior to the Fitbit. This way we could ensure that both devices recorded the same behaviour at the same location.

The **AX3 sensor** produced by **Axivity** is a MEMS 3-axis accelerometer and was also used in the UK BioBank study. According to Axivity (2015), the sensor can be configured to record an accelerometer range of ±2/4/8/16 g and a resolution of up to 13-bit. Furthermore, the sensor is water-resistant up to 1.5 m and can operate at temperatures from 0 to 65°C. The sensor's size is 23x32.5x8.9 millimetre and weighs 11 g. It has a battery capacity of 150mAh, a battery charge current of 150mA and a memory of 512 Mb. The logging frequency is configurable between 12.5 Hz to 3200 Hz. The manufacturer reports that 14 days of continuous data recording can be achieved at 100 Hz. The sensor's internal quartz real-time clock works at a 32.768 kHz frequency and has a precision of ± 50 ppm. Furthermore, the device contains a light and temperature sensor which we did not make use of. The sensors can be charged and their data can be accessed via the micro-USB. The sensor is mounted onto the wristband made from a soft-touch silicone material (see Figure 12). The OmGui Open Movment [V1.0.0.37] software (Jackson, 2015) was used to set up the recording session, and to extract and reformat the data from the sensor. We configured the sensor to the frequency of 50 Hz, which is the same frequency used in the KOALAP study. The Range was set at ±8 g. We exported the raw data as a .cwa file and transformed it into a .cvs file.

*Figure 12 – **SRDS sensory deceives used**. Left: AX3 Sensor (Axivity Ltd, 2015) and wristband (Axivity Ltd, nan); Right: Fitbit Charge 2 (Fitbit, 2018).*

The **Fitbit Charge 2** contains a MEMS 3-axis accelerometer but does not record raw data. It tolerates an operating temperature of -10° to 45°C but is not waterproofed (Fitbit, 2018). Since the Fitbit is a consumer device and not a research-grade device like the Axtivity, we did not have access to the same amount of detail regarding the sensor configuration. Data was extracted either

directly from the Fitbit by the documentarian recording the current step count onto a note-taking app done, or by announcing the step count out loud and later extracting the timestamps and step counts from the video recordings. Alternatively, we looked up the data recorded by the Fitbit, which can be uploaded onto an online profile, after synchronising with the Fitbit app via Bluetooth. However, on the online profile, the step count is only accessible summed in 15-minute (min) intervals. We are mostly interested in the step count output, but the Fitbit can also report heart rate, distance covered, calories burned, floors climbed, active minutes and hourly activity, using an optical heart rate tracker and an altimeter additionally to the MEMS (Fitbit, 2018).

4.3.3. Overview of Recording Trials

During the *Initial Trial*, real step counting was done by manual counting steps while walking and regularly documenting them on the smartphone note-taking app. We used this trial mainly to evaluate the accuracy of the Fitbit. The trial followed the participant's behaviour for 5.5 hours (h) on a Friday afternoon. It included walking, eating, office work, stair walking, sitting in a lecture and socialising. Sadly the exact starting time and end time of walking episodes were taken in a resolution of minutes and therefore did not have a high enough accuracy to enable us to label the steps in the accelerometer data. This was exacerbated by recording some behaviour as a mixture between walking and other activities. We sorted the behaviours of this trial into three different categories: 1) behaviour including walking, 2) non-walking behaviour, and 3) a mix between walking and non-walking behaviour. The first category mainly included walking from home to the office and walking to and from different locations within Aston University. The non-walking behaviour category includes the following behaviours: putting on a jacket, eating food, working at the office, and sitting in a lecture. At last, behaviours such as getting food, getting coffee, going to the toilet, and socialising included a mix of walking and non-walking behaviours. For example, the condition to "get coffee" consisted of walking to the office kitchen, preparing the coffee and walking back. We did not include this trial in the SRDS, since we mainly use it to test recording data and to evaluate the accuracy of step counts using consumer devices. Nevertheless, the trial was helpful in guiding what behaviours were recorded in the staircase and corridor trial.

Next, we collected data during a six-hour road trip in the *Driving Trial*. But the step count ended up confounded by one of the participants' habits to get out of the car before reporting their Fitbit step count to the documentarian. In the end, we obtained the Fitbit step count for the trial by extracting the activity log from the Fitbit account online. However, they only report the total step count in 15 min intervals. Therefore only 4 h in total were evaluated for the difference between real and Fitbit step counts. We did not include this trial in the SRDS, since it did not offer us behaviour that was notably different in its pattern from the coffee trial. We were interested in seeing if the confounding movement of the car would influence the data in a significant way, due to vibrations of the engine. We did not find patterns in the data, which would influence our algorithm to wrongly detect gait in a different way from the coffee trial. Since we were able to extract step counts for 4 h of the data, we deemed the trial sufficient to look at the Fitbit step count performance. We did not deem the data important for the SRDS and therefore did not attempt to re-record the trial.

The last trial not used in the SRDS was the *Football Trial*. It was conducted during the weekly get-together of colleagues from the Aston University department. While the full trial was recorded on video and the participant reported their step count from the Fitbit diligently, the documentarian incorrectly programmed the Axtivity sensor and the sensor data was therefore not recorded for the whole playing session. Since the sensor started recording data one hour later than desired, only 11 minutes of the 1 h playtime was recorded. The initial plan of looking at the video footage of the participant, manually counting steps and comparing it to the Fitbit step count, proved quite difficult to conduct. In the end, we did not judge our manual step counts accurate enough to compare them with the Fitbit step count data. Since evaluating the performance of the Fitbit step count is not a primary focus of this thesis, and we already had a good illustration of the over counting of steps from the initial trial, we decided to completely discard the football trial. Looking at the 11 min of sports behaviour recorded from the trial, we found the step behaviour to be surprisingly consistent and not much different in pattern from the free walking trail under the “free up arm swing” condition, except for a strong increase in step speed and amplitude. Since applying accurate labels to the data had the same underlying problems as the manual counting of steps, we decided not to include sports behaviour in the SRDS. One of the decisions we made for our GDA development, was to focus on walking behaviour only. We decided to exclude frequency ranges associated with running instead of walking. We justify this decision, by arguing that participants are more likely to be aware of their physical functioning being limited during sports activities, and that engagement in sports activities is more likely to be accurately recalled, than their daily walking behaviour.

The first data recording included in the SRDS was the *Staircase Trial*. It consisted of a total of 20 stair ascends and 20 stair descends with 14 steps each. We conducted half of the trials with the participant wearing medium-sized heels and half of them in socks. Furthermore, in half of the trials, we recorded the step count separately between ascend and descend, while for the other half the participant reported the Fitbit steps after ascending, carefully turning around and descending the staircase. In retrospect, recording the step count data this way was unnecessarily complicated and did not provide additional information to the SRDS dataset. It was, however, interesting to see how the turn affected the Fitbit step count. We will henceforth with regards to the SRDS treat the walks with separate counting of stair ascends and descends the same as the combined ascends and descends walks.

In the *Corridor Trial,* the participant walked 15 steps forward, turned and walked 15 steps back. This was done roughly 5 times for each condition: 1) wearing medium heels, 2) wearing no heels, 3) carrying a cup filled with water and 4) carrying a bag over the shoulder. The latter three were done not wearing shoes and objects were carried on the side of the sensor location. Carrying a bag thereby simulates walking under conditions where the arms are fixed to the body and do not swing freely while carrying a cup of water simulates carefully walking while balancing objects. Each walk resulted in 32 steps, but one walk resulted in two extra steps and two times a walk was repeated.

The *Coffee Trial* took place in a coffee shop, where the participant and the documentarian talked and drank coffee. Every 15 min the documentarian asked the participant for the recorded step count on the Fitbit. The recording took 78 min in total.

The last trial contained data recorded while walking around outside the Aston University campus and the inner city of Birmingham. This *Free Walking Trial* consisted of walking around the campus and nearby streets continuously without taking breaks. Triggers were set in the data by shaking the wrist between conditions. These conditions consisted of: 1) letting the arm hang loose with a focus on not using it, 2) allowing use (e.g. to readjust hair), 3) keeping the hand with the sensor in the jeans pocket, 4) holding and using a phone, 5) carrying a backpack like one would carry groceries. Again, activities were conducted on the arm with the sensor. No video recording was made and no real step counts were taken. Due to the previous recording of the football trial, we decided that attempts in this direction would be futile. While an attempt for manual step count and Fitbit step count collected on the smartphone note app was made, it was discarded during the walk. Using the triggers, we were able to accurately label walking behaviour for the SRDS.

4.3.4. Accelerometer Data Wrangling

All data extraction, wrangling, labelling and analysis of the SRDS were done on MATLAB R2018b (The MathWorks, 2018). The data collected from the stair walking, corridor walking, coffee and free walking trials were combined into the *self-recorded dataset* (SRDS) in this order.

The methodology for the pre-processing of the accelerometer data is based on Badawy, Raykov and Little (2017). The gravitational field (g) was inferred with the use of a variation of L1-trend filtering and then subtracted from the raw data resulting in the dynamic data (d). We did not apply interpolation to the dataset. The 3-dimensional data was reduced into one dimension for analysis by adding the magnitude (the square root of the sum of the squared data) of the three axes together (*data magnitude (dr)*).

We labelled the self-recorded data in accordance with the notes taken either on paper or verbally announced on the video recordings. The noted walking episode start and end times were accurate for the minute. The indoor walking trials (staircase and corridor trials) were timed to start at the beginning of the minute, while we set triggers during the coffee trial and the outdoor trial (free walking trial) were set by rotating the wrist quickly for a few turns. This results in an acceleration pattern of quick direction changes with high amplitudes, looking a lot like strong vibrations on all three axes. The labels are set at the start of each sec and therefore might include up to one sec of non-gait data at the start- and endpoints of each walking episode, however, the difference between the exact start and end of the walking episode is minimal.

The output is a vector labelling `data_Lab` where each data point is labelled as:

0 - unknown behaviour and triggers
1 - non-walking
2 - walking

The unknown behaviour consists of the time between walking episodes in the indoor trials. Mostly the arm with the sensor was consciously kept at rest although these trials may include some steps. The non-walking data consists completely of the coffee trial.

Another Labelling Catalogue save under variable `data_l` was applied on the 2nd of April 2020 to enable a more in-depth analysis of the data according to different behaviours. The following Labels were applied:

0 - unknown/non-gait
1 - potential gait (not documented gait)
2 - triggers
3 - non-gait (documented – coffee trial)
4 - staircase walking (in heels - up)
5 - staircase walking (in heels - down)
6 - staircase walking (no heels - up)
7 - staircase walking (no heels - down)
8 - walking in heels (in heels - arm free swing)
9 - walking in heels (no heels - arm free swing)
10 - walking in heels (no heels - carrying mug with water in the hand)
11 - walking in heels (no heels – carrying a bag over the shoulder)
12 - Coffee trial (documented non-gait)
14 - Free walking (artefacts)
15 - Free walking (arm free swing)
16 - Free walking (hand in pocket)
17 - Free walking (hand uses phone)
18 - Free walking (arm free swinging with random hand use)
19 - Free walking (carry bag in hand (like a shopping bag))
20 - Free walking (random hand usage)

The data was further labelled for all behaviours seen in Table 5. These labels were used as categories to visually and statistically compare walking behaviours with one another. Within the Free Walking session noise in the data that was longer than a second was labelled as artefacts and excluded. Additional labels were created by combining behaviours into the following categories: going upstairs (18) or downstairs (19), walking in heels (20) or barefoot (21) and letting the arms swing freely (22) or making use of the arm (23).

Behaviour (of ***trials***, *conditions* and **categories**)	**New Label**	**Time (length in obs & min)**	**Step Count (Real)**	**Step Count (Fitbit)**	**Description & Notes**
Stair	15	21,948 7.32	600	646	made up of `data_l` label: 1, 2, 3 & 4
Up in heels	1 'Stair Swing Heel Up'	5,413 1.80	150	165	10 walks (14 steps up, turn, 14 steps down each)
Down in heels	2 'Stair Swing Heel Dw'	5,021 1.67	150	159	
Up barefoot	3 'Stair Swing Flat Up'	6,151 2.05	150	165	

Down barefoot	4 'Stair Swing Flat Dw'	5,363 1.79	150	159	
CorrW	16	22,142 7.38	662	603	made up of `data_l` label: 5, 6, 7 & 8
in heels	5 'Corrw Swing Heel'	4,834 1.61	150	156	5 walks (15 steps, turn, 15 steps each)
barefoot	6 'Corrw Swing Flat'	4,791 1.60	150	153	
mug (barefoot while balancing water in a mug)	7 'Corrw Stiff Flat Mug'	6,385 2.13	180	105	6 walks (15 steps, turn, 15 steps each)
shoulder bag (barefoot while carrying a bag over one shoulder)	8 'Corrw Stiff Flat BagShoulder'	6,132 2.04	182	189	
Coffee	14	234,618 78.21	0	395	steps count recorded in intervals of ~ 15 min
FreeW	17	141,685 47.23	-	~ 5500	made up of `data_l` label: 9, 10, 11 & 12
clean	9	57,459 19.15	-	~ 1121	clean arm swing without other activities
pocket	10	25,578 8.53	-	~ 832	hands in jean's pocket
phone	11	9,799 3.27	-	~ 551	using a phone
handbag (caring a bag like a shopping bag)	12	13,973 4.66	-	~ 563	carrying a bag imitating a shopping bag
random	13	34,876 11.63	-	~ 2433	arm swing with random arm movement
upstairs	18	11,564 3.85	300	330	made up of `data_l` label: 1 & 3
downstairs	19	10,384 3.46	300	318	2 & 4
heels	20	15,268 5.01	450	480	1, 2 & 5
barefoot	21	16,305 5.44	450	477	3, 4 & 6
swing (letting the arm swing freely)	22	123,908 41.30	-	~	5, 6, 9 & 13
stiff (making use of the arm)	23	61,867 20.62	-	~	7, 8, 10, 11 & 12

Table 5 - Information about the Recording and Labeling of the Self-reported Data.

4.3.5. Feature Extraction

In order to examine how gait data can be affected by different forms of gait and behaviour impacting the place of data recording (the wrist), we ran a statistical comparison of gait features between behaviours. The data was divided according to the behaviours, categories and sessions seen in Table 5 and the following features were extracted: Windowed Mean, Windowed STD and Spectral Support.

The Windowed Mean (Feature Mean) was calculated by applying a Moving-Average Filter (`filter((1/50).*ones(1,50),1,data)`) to the data magnitude using a rational transfer function. The window size was set at 1 sec (at 50 Hz). A running, windowed STD was used to extract the Windowed STD (Feature STD, `movingstd(data,50)`), with a window size of 1 sec.

The Spectral Support Features were identified by applying a spectrogram using the short-time Fourier (STFT) transform (`spectrogram(data,50,0,[],50,'yaxis')`) with a window size of 1 sec and no overlapping of samples. The resulting STFT is then expressed as absolute numbers, squared and divided by half of the window size (`(abs(support_gait).^2)/(50.*0.5)`). The data is analysed in the 0.5 to 10 Hz range. Lastly, the STFT of each frequency was added together for each time window resulting in the Sum of the Spectral Support (Feature SumP) and the maximum was taken to receive the Maximum of the Spectral Support (Feature MaxP).

4.3.6. Analysis

We displayed and examined the different features of selected walking behaviours using histograms, conducted a Receiver Operating Characteristic (ROC) curve using selected gait detection methodologies and examined the resulting z-vectors. The z-vector is a variable that labels data as gait (`z = 1`) or non-gait (`z = 0`) for a selected gait detection algorithm.

4.4. Results

4.4.1. Fitbit Results

Looking at the Fitbit data of the initial trial recording we can see that the Fitbit tends to undercount steps slightly while walking (Real Step Count (SC) = 1,706; Fitbit SC = 1,616) by missing 90 steps. However, some steps will be recorded even when activities mostly involve sitting down (Real Step Count =0; Fitbit SC = 110) and during mixed activities (Real Step Count = 365; Fitbit SC = 613). The difference is particularly high during socialising where 225 steps were counted by the Fitbit while the actual step count was only 8 steps. This has mainly to do with gestures done using hands during conversations. Furthermore, it should be noted that while on average the step count is only off by a few steps in every episode, the accumulated effect over the day can be much larger (see Figure 13, black line). During the 5.5 h of this recording, the Fitbit counted 268 extra steps (approximately 348 metres).

We also recorded the Fitbit step data during a long car trip with two drivers. A total of 31 steps were recorded for the first driver within 1.5 h (21 steps/h) and 12 steps for the second driver within 3 h (4 steps/h). It shows that the same behaviour of two people can be measured by step counting algorithms with a wide difference. These differences might not be large, but can be substantial when comparing step counts between individuals.

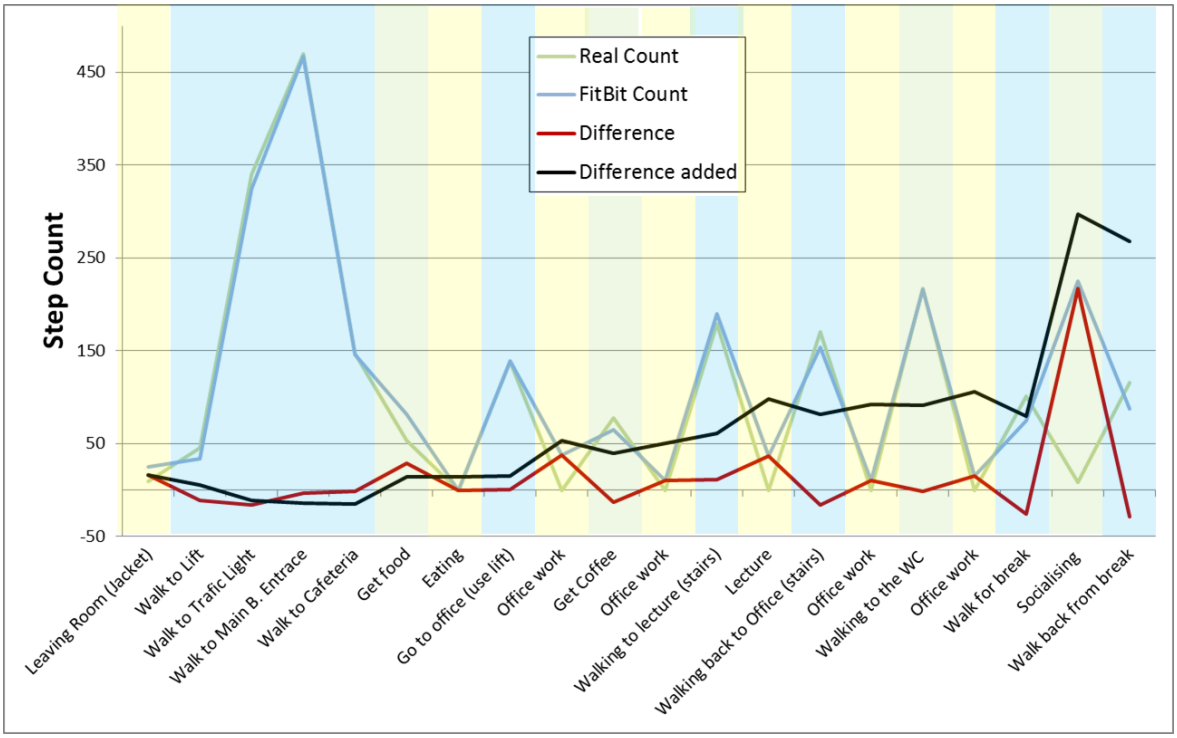


Figure 13 - ***Comparison of Real Step Count to Fitbit Step Count within the initial trail.*** *Activities classified as walking (blue background), non-walking (yellow background) and mixed (green background).*

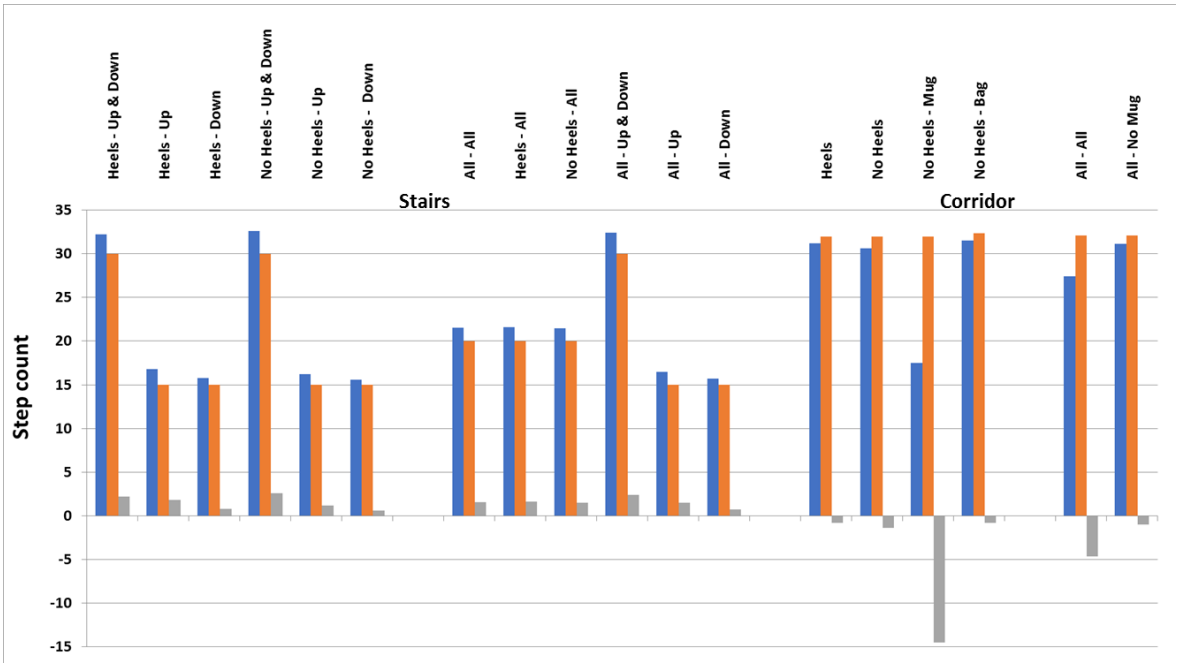


Figure 14 - ***Average step count during stair and corridor trials under different conditions****. Blue = Fitbit Step Count; Orange = Real Step Count; Grey = Difference between Fitbit and Real Step Count.*

During the staircase trial (see Figure 14, left) 646 steps were counted by the Fitbit, while only 600 steps were taken. The Fitbit seemed to overcount more during stair ascend (M = 1.5, STD = 2.12) then during descend (M = 0.7, STD = 0.95). On the other hand, while 706 steps total were done during the corridor trial (see Figure 14, right), only 603 were recorded by the Fitbit. However,

most of the miscount can be attributed to the “mug” condition. While the other trials only missed one step on average per walk (STD = 0.71), only one of the 6 mug walks counted close to that range of error, with the other walks missing more than half of the steps per trial. Overall, the trials show that the Fitbit slightly undercounts steps when walking on stairs and overcounts slightly when walking downstairs in experimental settings. When holding a mug of water the step counting algorithm of the Fitbit was not able to identify walking. This is most likely due to the difference in hand position and the reduction of the amplitude of the dominant peaks (see Figure 15). A smaller magnitude of the signal will not reach the necessary thresholds to be detected by some step counting algorithms.

While no real steps were taken during the coffee trial, the Fitbit counted 395 total, with a large difference between the 15 min windows (M = 79, STD = 42.33). The misattribution of around 5 steps/min for this non-gait behaviour illustrates that upper body movements during sedentary behaviours can produce an acceleration pattern, that is picked up on by the Fitbit algorithm as steps. Looking at the video some arm movements executed while talking about preparing food might have led to the misattribution of walking behaviour.

4.4.2. Gait Pattern Visualisation

The following chapter is made up of self-observations if not otherwise referenced. The observations were derived from looking at the dynamic data, video recordings and re-enactments (i.e. walking up and down the office corridor to determine typical arm motions). They, therefore, include speculations and should be judged critically. The data contained within the various gait labels is depicted in whole in the Appendix, Figure 88.

We selected some behaviours that were particularly distinct from one another to illustrate the way how accelerometer data depicting gait can look quite different depending on the situation independently of impairment of gait (see Figure 15). Ten sec of data selected from the middle of the data section of each behaviour are displayed below.

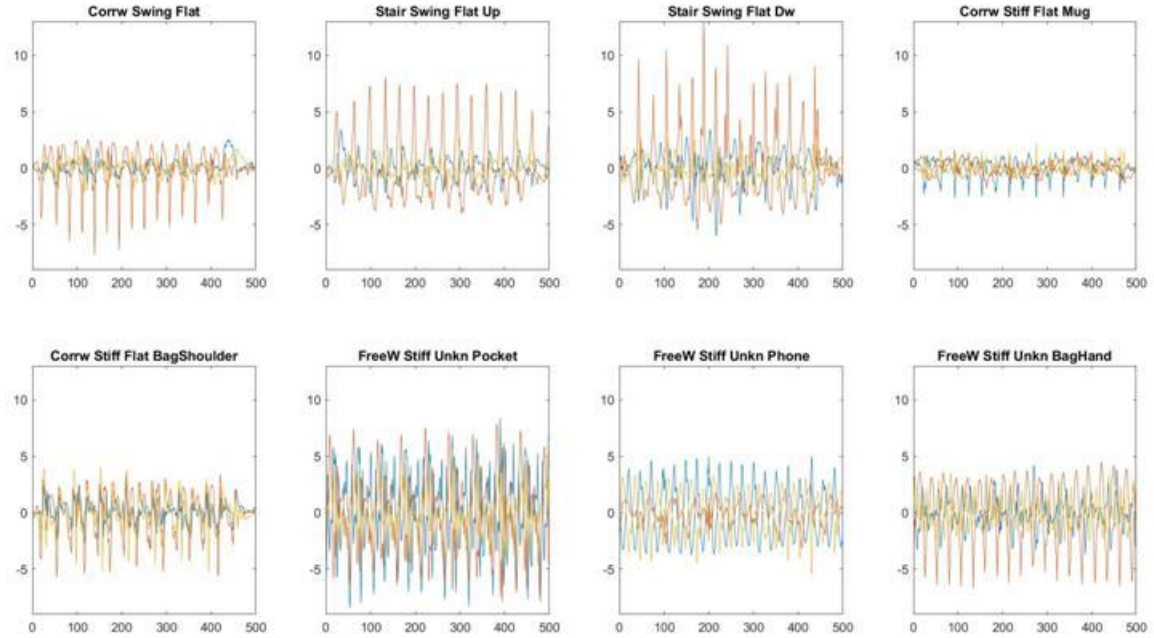


Figure 15 - ***Example dynamic data of selected behaviours of the SRDS****. X-axis = time in samples (obs, 50 obs = 1 sec), Y-axis = acceleration (m/sec^2), length of each window = 10 sec, blue = x-axis, red = y-axis, yellow = z-axis.*

It needs to be noted that while we are interested in the pattern arising from gait-related behaviours, we do not measure leg movements directly, but rather infer walking behaviour from wrist movements. The most consistent behaviour acting on the wrist movements during simple walking is the arm swing. Data depicting the “swinging” condition shows clear spikes dominantly on the y-axis (Figure 15 a, b and c). This can be explained by the orientation of the axis on the watch in relation to the movement of the arm (see Figure 11, Chapter 4.2.2). Movements going in the direction from the radial (medial, thumb side) to the ulnar (lateral, pinkie finger side) zone leads to an increase in acceleration on the y-axis (lateral), with a movement from ulnar to radial leading to negative acceleration (medial). Moving the wrist from volar (palm side) to dorsal (hand back) leads to an increase of acceleration on the z-axis. Movement from the posterior (arm) to the anterior (finger) will lead to an increase on the x-axis.

Arm swing is theorised to potentially serve for stability optimisation and energy consumption minimization during walking (Meyns, Bruijn, & Duysens, 2013). Arm swing can be impaired indirectly due to impaired gait, but also directly through neurological pathologies such as hemiplegia after stroke, Cerebral Palsy, Parkinson’s Disease, spinal cord injury, etc. They can either reduce the arm swing or prevent its occurrence completely. Furthermore, the unaffected arm might show an increase in swing (for more detail see Meyns, Bruijn and Duysens (2013)). The normal arm swing occurs out of phase with the leg movements. Thereby the arm can either be straight or bent at the elbow joint. During more intense or faster walking the arm swing becomes more pronounced and bending is more likely to occur. Bending of the elbow during free-swing of the arm is only possible in the direction of the anterior (walking direction) and not the posterior. Therefore, during more energetic walking the wrist will swing further towards the medial direction but the lateral backswing will be stopped by the elbow joint (Park, 2008). In the data, this can be seen in the corridor trial (Figure 15a, Corrw Swing Flat), where the acceleration shows large

negative spikes on the y-axis corresponding to wrist movement from ulnar to radial. The other axes only show minimal fluctuations in acceleration.

Staircase walking requires more control over the body than walking on flat surfaces, preventing the body from falling backwards after ascend or forward during descend. During ascending periods the body leans more forward and the arm movement, therefore, swings stronger backwards than forwards. Even without the ability to bend the elbow backwards, acceleration is stronger in the lateral direction, which causes stronger positive spikes on the y-axis (see Figure 15b). During descend this behaviour intensifies since it is much harder to balance out the forces acting on the body during descend than ascend. An increased need to balance the body out leads to more variability in the body movements and consequently more variability in the data on all axes (see Figure 15c). Walking on flat surfaces, therefore, produces a very smooth, uninterrupted walk, while dealing with uneven grounds leads to more variability in the gait.

However, arm-swing is not a requirement for walking. Among others, the arms can be bound to the body, held stiff during walking, move in an anti-normal fashion (arm moves with the leg on the same side instead of the opposite side) or be extended in unstable surroundings for added balance. Furthermore, hands can be in use during walking, for example by carrying bags, handling mobile devices or balancing drinks. We therefore also examined various behaviours where the hand is in use or fixated onto the body. We categorise these walking behaviours as “stiff” conditions (see Figure 15d, e, f and g).

Carrying objects in front of you is a daily necessity and oftentimes these objects need to be balanced while being carried. The example behaviour we used is the carrying of a mug full of water (see Figure 15d). The behaviour requires reducing unwanted movement as much as possible. The balancing of the water causes the participant to correct for the shock from heel-strike, resulting in reduced acceleration. This can be well observed in the accelerometer data. Spikes are clearly visible but much more subdued than with any of the other behaviours. These spikes, however, are not located on the y-axis as in the swing-condition behaviours, but mostly occur on the x-axis and occasionally on the z-axis. In some “mug” walks the spikes on the z-axis are even larger than the spikes on the x-axis, but they are not depicted in Figure 15. The grip of the mug handle changes the orientation of the wrist during standstill from fingers pointing towards the floor to the ulnar pointing towards the floor. While balancing a mug the wrist moves parallel with the rest of the body instead of swinging with the arm movement. Therefore the axis most affected by the heel-strike is the x-axis.

Carrying a bag over the shoulder on the side of the accelerometer is represented in Figure 15e. The action of pressing the bag against the body connects the accelerometer to the upper body, enabling the heel-strike to have the most effect on the data. However, a subdued swing still occurs, since the bag is not glued to the body and therefore slightly swings much like an arm with the body motion. Consequently, a slight swing motion is also experienced at the wrist, which is resting on the bag. The swing occurs in the same direction as the leg on the side of the accelerometer, as opposed to the “swing” conditions. In the accelerometer data, we see noticeable

spikes of all three axes at the same time with similar amplitudes, something that is not the case in the swing conditions. However, these spikes are not as great in amplitude and quite noisy.

The behaviour of putting your hand in your pocket in Figure 15f causes a slight change in wrist position between steps. During walking, the pockets that are stitched onto the jeans change the wrist angle to the body with the angle of the leg to the hip. During heel-strike of the leg on the same site with the sensor, the sensor orientation will have the fingers pointed to the floor. Gravity, therefore, acts on the anterior (x-axis). Furthermore, the fixed location on the hip causes not only the heel-strike of the leg on the side of the sensor to cause the largest acceleration in the data, but the heel-strike of the opposite leg also strongly vibrates through the hip. This results in the x- and y-axis showing peaks of similar size opposing each other. The location in the pocket did not cause the acceleration signal to decrease; instead, it shows the largest acceleration difference between the peak and the valleys of all of the gait behaviours shown in Figure 15, due to the location being closer to the heel-strike location.

Holding and using a phone while walking leads to a similar pattern to holding a mug, however, the acceleration is not as subdued, since it is not important to prevent the phone from shaking in relation to gravity as it is to keep the phone from shaking in relation to the eyes for reading. The phone, therefore, is held to experience the same movements as the head and therefore the rest of the body. Swing is prevented. The phone used in these experiments was an iPhone 6. Holding is during use angles the wrist between the ulnar (y-axis) facing towards the ground and the dorsal (z-axis) facing the ground. Therefore, in the acceleration, the axes showing a slightly noisy gait pattern are the z- and y-axes, with both axes showing similar peak heights or one axis being more pronounced than the other (see Figure 15g).

The last condition shown in Figure 15h is carrying a bag like a shopping bag. While the signal of the y-axis behaves quite similar to the swing conditions, the other axes are much more impacted by the behaviour than in the pure swing conditions shown in Figure 15a-c.

4.4.3. Walking Behaviours – Descriptive and Statistic

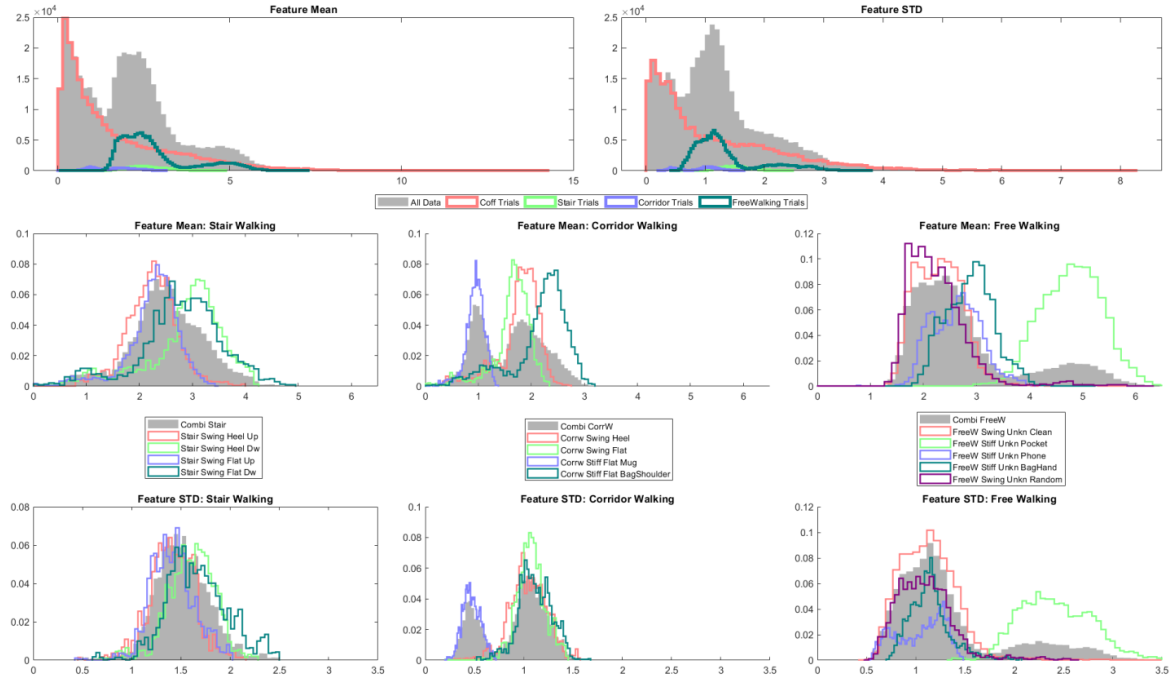


Figure 16 – ***Histograms of the Feature Mean and Feature STD of the Labelled data of the SRDS****. Top = Whole SRDS by trail, middle = Feature Mean by behaviour, bottom = Feature STD by behaviour. X-axis = acceleration (m/sec^2), Y-axis = relative probability.*

Figure 16 displays the Feature Mean and Feature STD of the different walking labels and the non-walking label. The non-walking behaviour shows a much smaller mean and STD than any of the walking behaviours. In the staircase trial, we can see a tendency for stair ascend to display a slightly lower amplitude and variability. We were not able to see much difference between wearing heels or walking in socks. This can also be seen in the corridor trial. Notable in a corridor trial is significantly decreased feature mean and STD for the mug carrying behaviour. This is most likely due to being more careful when balancing water. In comparison, carrying a bag over a shoulder led to an increase in the feature mean. Most of the free walking trial behaviours overlap with most of the corridor walking and stair walking behaviours, the exception being holding hands in pockets. The fixation of the arms on the lower body caused the heel strike to have a very significant impact on the feature mean. At the same time, the still somewhat loose fixation on the body caused an increase in variability. Particularly interesting to observe is the viability of the phone using behaviour, which fluctuates between higher and lower viability, while staying in the normal range for other walking behaviours. This might be due to trying to hold the phone steady while reading and noise being introduced when interacting with the phone.

The spectral support, however, did not yield any interesting results (see Figure 17). Looking at the sum of the spectral support, we were not able to find any difference between the labels. The maximum spectral support tended to show a slight increase for the free walking behaviours mostly caused by swing behaviours. However, this difference was only slight and not consistent between the labelled behaviours.

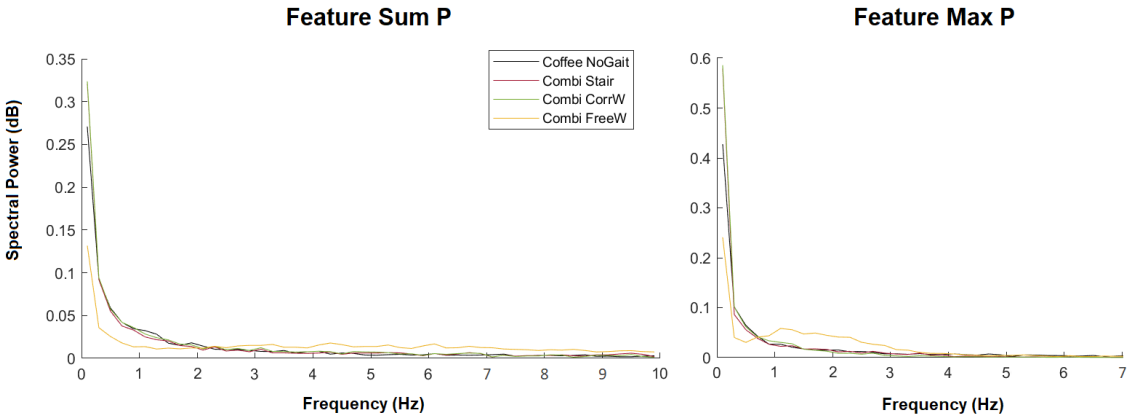


*Figure 17 – **Histograms of the Feature of Spectral Support Sum and Max**. Left: Sum P; Right: Max P.*

We also made some interesting observations comparing the Feature STD of selected behaviours (see Figure 18). The axis on which the walking behaviour was expressed most dominant has the highest moving STD of the 3-axis and a larger moving STD than the magnitude of the acceleration. If the behaviour was similarly expressed on two axes, such as the pocket behaviour, their moving STD would overlap. In the appendix Figure 88, we display the data of all walking behaviour labels of the SRDS, the corresponding moving STD and the histogram for the moving STD of the various axes and the magnitude of the data.

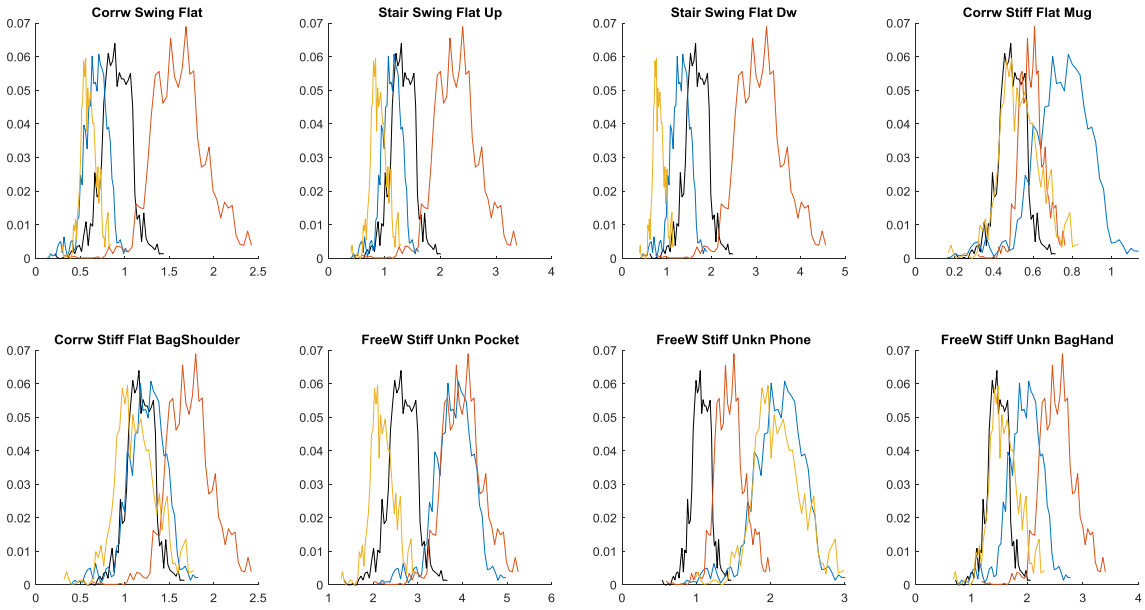


*Figure 18 - **Histogram of Feature STD for selected behaviours**. X-axis: acceleration in m/sec^2, Y-axis: normalized probability, black = magnitude of the acceleration (dr), blue, red, yellow = x-, y- & z-axis acceleration.*

4.4.4. Applying Gait Algorithms

The results of applying a receiver operating characteristic (ROC) of various gait detection algorithms on the data are reported in Table 6 and Figure 19.

	ROC		Labelled			
Methods	**True positive rate**	**False positive rate**	**Walking**	**Rest**	**Rest when Walking**	**Walking when Rest**
Hand Labels			42.36%	57.64%		
NASC	0.84	0.17	40.83%	25.73%	1.15%	31.90%
STD-thresholding	0.72	0.20	37.13%	36.56%	5.24%	21.08%
STFT-thresholding	0.92	0.29	41.03%	30.65%	1.33%	26.99%
CWT-thresholding	0.99	0.34	38.88%	3.53%	1.60%	54.10%
Switching AR	0.96	0.36				

Table 6 – True positive and false positive Rate for Gait Detection Methods

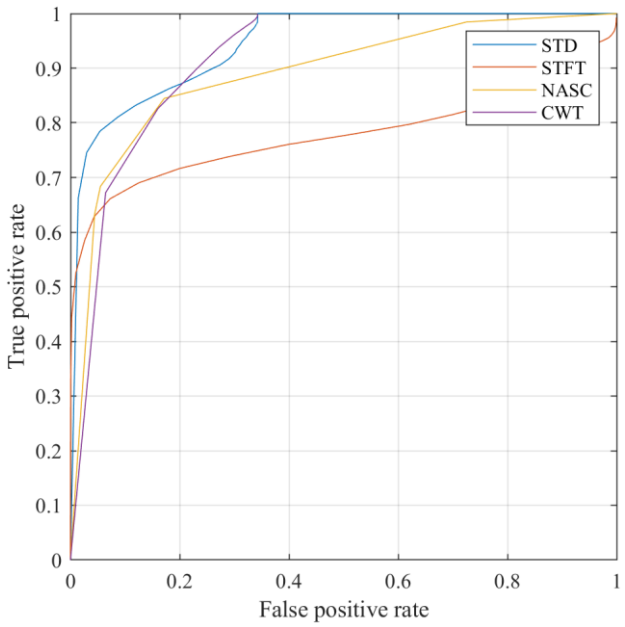


Figure 19- ***ROC curve results for various gait detection methods.***

Figure 20 displays some examples of the algorithms' performance on non-walking and walking behaviours. While thresholding is very useful to identify potential walking behaviour from resting behaviours, it is limited in excluding walking behaviour from non-walking behaviours with high activities. For data collected at the legs, there are not a lot of sustained high-level activities that do not consist of walking. However, data located at the wrist is exposed to a lot of non-walking behaviours in comparison. Furthermore, the inflexibility of singular thresholds can lead to careful walking behaviours being misidentified as resting behaviour. The NASC is good at not missing walking behaviour but identifies too much non-walking behaviour as walking. The STFT is similar to the NASC and a little bit better in excluding the rest from the walking data, albeit not by much. While the C/DWF is precise in identifying starting points of the walking episodes, it is not very selective at identifying gait vs non-gait. Table 6 shows that most of the algorithms tend not to

misidentify walking behaviours as non-walking behaviours, with Thresholding displaying the worst performance with around 5% of data wrongly identified as rest. However, all methods perform very poorly when it comes to correctly identify non-walking behaviours, with CWF performing the worst by misidentifying over half of the data. Most common Gait detection algorithms are more suited to differentiate activity from non-activity than to identify actual episodes of gait when recorded at the wrist.

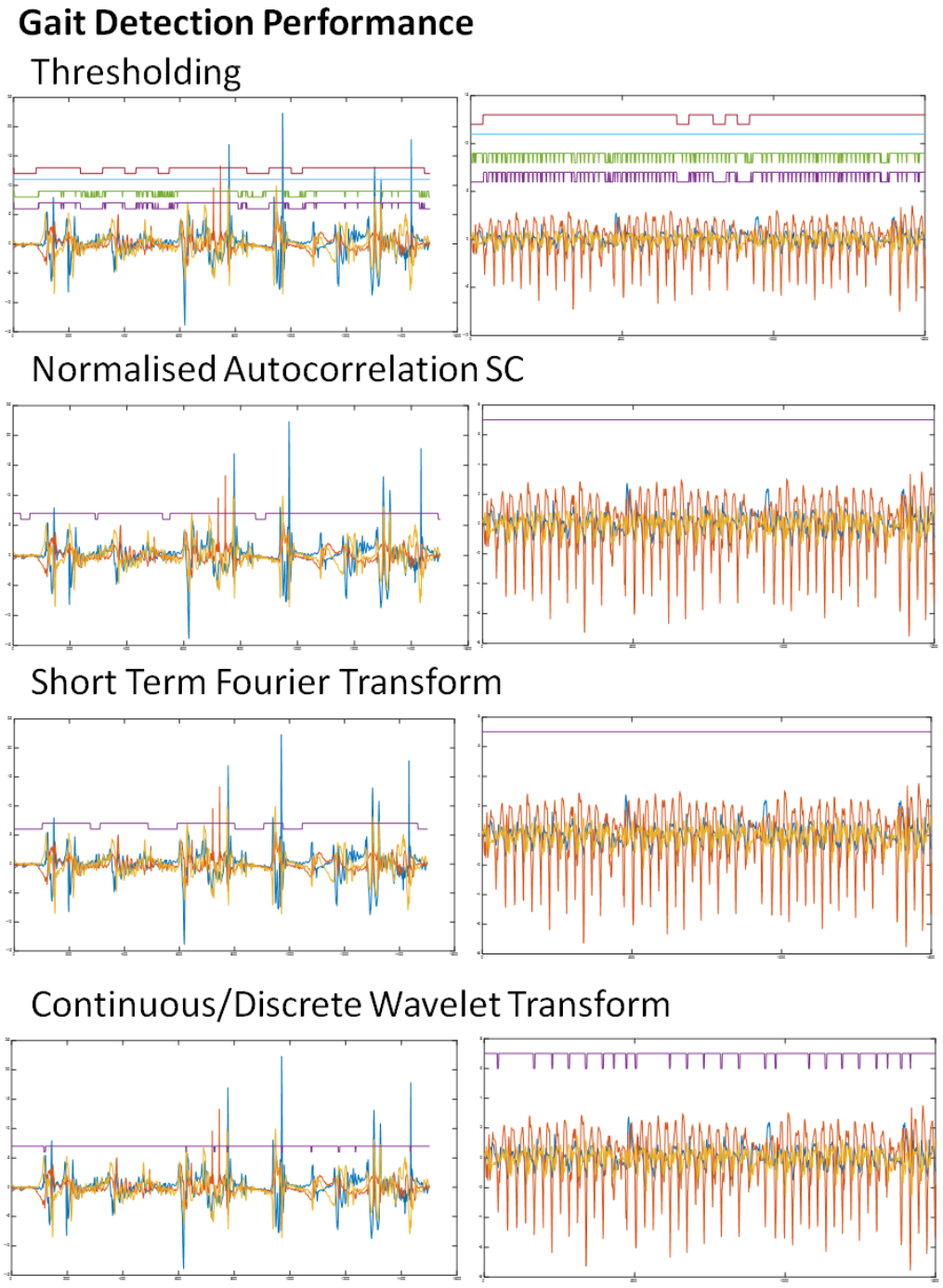


Figure 20 - ***Performance of selected gait detection methods****. Left: non-walking behaviour, Right: walking behaviour. X-axis: observations (obs), Y-axis: acceleration in m/sec^2 and display of z-vectors. Blue, red & yellow = dynamic data of x-, y- & z-axis purple = Z-vectors (Gait vs Non-gait),. In "Thresholding": dark red = acceleration magnitude mean, light blue = acceleration energy, green = acceleration magnitude STD, purple = combined thresholds.*

4.5. Discussion

Looking at the reported step counts of a common consumer deceive, we observed that depending on the walking activity the Fitbit will slightly over-count or under-count steps or will not be able to pick up some forms of walking at all. This is also dependent on the individual wearing the watch. Overall step count algorithms tend to overcount steps over a day. However, this can also be highly dependent on the individual. This leads to interpretation decisions that have to be made when interpreting step count data of consumer devices. A participant that uses their upper body to express themselves heavily will have a higher step count consistently. The One-day Diary data was used primarily to compare self-counted steps to Fitbit Step count, but the stair walking, corridor walking and coffee sessions results are also evaluated. It needs to be considered that the sample size of trials was quite small and only tested on one person and can therefore not be used to make any meaningful observations about the method of step counting. However, the results are similar to previous studies in the literature. The comparison is made to expose potential errors and problems with gait analysis from wrist-collected data.

We tested some common gait detection algorithms on the SRDS and observed, that while most algorithms perform where we will in identifying gait behaviour, they massively underperformed in identifying non-walking behaviour. Often gait algorithms are used to identify start- and endpoints of walking behaviours in lab recorded settings. There the algorithms are not commonly exposed to high activity non-walking behaviour. Furthermore, algorithms and consumer devices are most likely optimised to not miss walking behaviour to the cost of overestimating step count. It is valid to assume that consumers would most likely complain when encountering the former than the latter.

It needs to be noted that while we are interested in the pattern arising from gait-related behaviours, we do not measure leg movements directly but rather infer walking behaviour from wrist movements. The data collected is therefore not mostly a direct result of the heel-strike as it is when sensors are placed on the lower body, but most often of the arm swing. Looking at our self-recorded data we made some interesting observations.

1) Walking behaviour is periodic in nature.
2) Axes can display periodicity that opposes each other. In these cases taking the magnitude of the data leads to a decrease in acceleration magnitude.
3) Careful walking will cause the amplitude of the data to be suppressed.
4) Most of the gait detection algorithms rely on either the amplitude of the data to be continuously increased and/or on the periodicity of the walking behaviour.
5) We separated walking behaviours into two categories “swing” and “stiff”. While most swing behaviours lie on the y-axis and are exposed to less noise, stiff behaviours can lie on multiple axes and tend to exhibit more noise in the data.
6) Axes on which the walking behaviour lies can be identified as dominant by comparing their moving STD.
7) Less dominant axes exhibit more noise.

Reducing the three-dimensionality of the data to one dimension by taking the magnitude of the acceleration reduces our ability to extract gait parameters. However, comparing the magnitude of

the data to the moving STD of the individual axis enables us to identify a dominant axis that we judge the walking behaviour to be represented on with the least amount of noise. This dominant axis would also enable us to extract step parameters from episodes of walking behaviours.

In conclusion, when developing an algorithm we can take benefit from our dominant axis observations, investigate the periodicity of the data and to not rely on thresholds that will most likely exclude careful walking.

The dataset is limited by the single participant that the SRDS dataset was recorded from. Only one young healthy female participant's data was included in the SRDS. Therefore, while we recorded a lot of walking data with diverse behaviours, the diversity of the dataset itself is limited. During the course of our data collection, we narrowed our focus to walking behaviours and excluded behaviour of vigorous activity, such as running and sports. We justified this through our aim to passively monitor daily living behaviour. The likelihood of patients correctly remembering the amount of sport they engage in is much higher than remembering their willingness to take the stroll through the neighbourhood. Especially sports activities are much more ritualised and therefore easier to keep track of. While running is also a type of walking behaviour, we decided to exclude vigorous levels of activity. The ideal type of walking behaviour we, therefore, aim to extract are sustained and mostly not interrupted sections of walking behaviour displayed during the participants daily living.

Additionally, since most of the data recording and brainstorming were done by women, there might be some biases in the behaviour that was recorded. For example, walking in heels turned out to not result in a pronounced difference in gait patterns recorded at the wrist but was extensively analysed. In defence, reading about the effect of using insoles also inspired the investigation of different footwear and the creation of the dataset. After looking at the dataset, we doubt that we will find patterns in the data sufficient enough to recommend the use of insoles, a potential diagnostic aim proposed in the literature review conclusion. We predict that only data recorded over a long period of time will be sufficient to find reliable gait patterns within participants. Fortunately, the KOALAP dataset offers us data recorded from individual participants over a period of multiple months and the UK BioBank dataset offers a sufficiently large population for analysis. The following chapters will describe the development of the GDA and SPE algorithms, which were informed using the observation and label data of the SRDS. Afterwards, the algorithms will be applied to both the KOALAP and UK BioBank datasets.

Furthermore, when labelling the dataset the accuracy of when to start and end each walking episode was flexible by 1 sec. Correctly classifying these sections of data as non-walking while we had labelled it walking is very likely, especially since we started walking behaviour from a resting position. Another point to consider is that the positioning of the recording device on the wrist is not perfectly well regulated. In the UK BioBank study participants had to put on the wrist device themselves and might have taken it off and put it back on in a different orientation or even on a different arm. In the KOALAP study, the participants were shown how to correctly put on the watch, but had to remove them daily for charging. We think that sensor orientation changes like this would be a common problem when measuring participants in free-living conditions where they will take off

sensors for various reasons. We are sure that we re-inserted the sensor differently between the staircase and corridor trials ourselves, but the switch in axis orientation did not influence our data analysis. This problem, however, would need to be addressed if pattern analysis is used to distinguish between different walking behaviours. For example, in our dataset, the axis orientation would influence our interpretation of classifying walking behaviour as stair-ascend or descend. Consequently, while we did make interesting observations about the axis orientation which could give us more information about the type of walking behaviour, we did not have time further explore this topic in the algorithm development. Some attempts to further examine the data were made, such as taking the gravity data into account when identifying the orientation of the device but were discarded. Exploring these methods might be interesting for future work.

5. Gait Detection Algorithm Development

5.1. Abstract

We developed a gait detection algorithm (GDA) with an accuracy of 97.20% on our self-recorded dataset (SRDS). Our novel approach involves identifying the axes on which the gait behaviour is expressed dominantly, applying the windowed STD of the magnitude of the dynamic data and the individual axes and by removing periods of high activity that are too short to contain a 5 step gait episode. The periodicity of the data was determined using our autocorrelation. Furthermore, our algorithm is distinguished by its fast processing time of 4.13 sec/h for data sampled at 50 Hz. One-third of this time was spent saving the data, therefore further optimisation is possible. The data extracted from the GDA can be used to extract step parameters. Therefore, we prioritised data that exhibits a clear gait pattern. The main limitation of this algorithm is, therefore, that it potentially excludes strongly confounded walking behaviours. We made this decision, after realising that attempting to extract every gait behaviour from data collected by a wrist-worn sensor is futile, due to the confounding hand movements while walking. Furthermore, gait data that is too impaired to be picked up by this algorithm, would also not be possible to extract step parameters from.

5.2. Background

Our aim is to develop a gait detection algorithm that is able to extract periods of walking behaviour from accelerometer data recorded at the wrist in free-living conditions. We explored a variety of different common gait detection methods and developed our own approaches as well. In the end, we developed a gait detection code that combined pre-existing and original methods. This involved identifing patterns in the gait data to guide our methodology. We also had to discard many attempts, sometimes because they were not suited to the wrist-recorded data, sometimes because we found more effective methods and sometimes because we failed to apply the algorithm to our data. The following background chapters will first describe the development timeline and talk about most of the approaches that we discarded. Then we will summarise what focuses and conditions we used to develop our final gait detection algorithm. Afterwards, we will describe the steps that made it into the algorithm in the end by going through the code step by step. The Algorithm Development chapter can also be seen as a guide to the code and will include advice on how to adjust the code to one's own dataset. Finally, we will explore the final algorithms' performance, by looking at the accuracy, importance and processing time of each algorithm step.

5.2.1. Algorithm Development Timeline and Discarded Gait Detection Approaches

The development of the Aston Gait Detection Algorithm (AGDA) to extract walking behaviour and gait parameters from data collected at the wrist in the free environment was the main goal of this thesis. The development of the ADGA was, therefore, influenced by every topic we covered in this thesis. The conditions and goals were determined by the literature review and the format of the KOALAP dataset. The latter was selected for analysis after setting our goal of using gait parameters to examine physical functioning in patients with musculoskeletal conditions (RMD) in a

free-living environment. The following subchapter will illustrate the timeline in which the AGDA code was developed and describe some approaches that were discarded.

First, we need to explain why the AGDA was separated into two parts, the gait detection algorithm (GDA) and the step parameter extraction (SPE). Due to the massive size of the KOALAP and UK BioBank datasets, we already run the GDA to extract data exhibiting walking behaviours before the development of the SPE was done. This had the unfortunate side effect of reduced optimisation. Some steps that were done in the GDA had to be repeated in the SPE, due to the data required by the SPE not being saved. Furthermore, output data was safe for the GDA that we did not use later on. In the following two chapters, we will describe the algorithm as it was used for the data extraction in this thesis, however, we will also discuss how it can be further optimised in the concluding subchapters. All codes will be displayed in the appendix and published on Github (Weihrich, 2022). The latter will be used to publish updated versions of the code.

We examined a lot of different approaches, many of which we discarded. The structure of this thesis does not reflect the temporal order in which we made observations that helped us develop the AGDA. The previous chapter examined the self-recorded dataset (SRDS) in detail, while Chapter 7 will illustrate the handling of the KOALAP dataset. Originally, we first gained access to the KOALAP dataset in the early spring of 2018. After wrangling the raw sensory data (see Chapter 7.5.4), we pre-processed the data and attempted our first gait detection algorithm. Due to this timeline, the pre-processing of the KOLAP dataset was already completed by the time we put together the AGDA code. The code, therefore, loads in already pre-processed data, instead of loading the wrangled data.

For the first gait detection attempt conducted in the early summer of 2018, we choose two days (13^{th} to 15^{th} September) and one participant (User 8) to conduct the pre-processing and a switching autoregressive model (Switching AR) on. Originally we wanted to pre-process a week worth of data, but the process was still slow and kept crashing. User 8 was selected since they had chosen “Sitting for long periods” as their most aggravating activity affected by pain and “Playing Sports” as their most important activity affected by pain (see Chapter 7.8.3., Figure 79). Especially choosing sports as the most aggravating activity speaks for their physical ability to be strongly affected during high-intensity activities. One could assume that User 8 will most likely show normal or only slightly impaired gait during normal short walks. Their gait might be impaired at faster or longer walking episodes.

In preparation for the Switching AR we extracted the magnitude of acceleration (dr) from the dynamic acceleration (d). Then the dr data need to be downsampled. We decimate the data by a factor of 20, meaning that only every 20^{th} observation (obs) was kept. This reduces the sample rate of the data to 2.5 Hz. Before downsampling an anti-aliasing via moving average filter is applied to the dr. Anti-alias filters are low pass filters that reduce the higher frequencies. This step is important to make sure that large outliers do not skew the data. The Switching AR is a nonparametric, unsupervised machine learning model that can identify between 1 and N different patterns within the complete dataset. We set the maximum number of patterns that can be identified to 30. We judged a maximum of 30 different data segments to be sufficient. In our

example data, the Switching AR identified 17 different patterns in 7.07 h of data of participant 8 on the 13th of September. Figure 21 illustrates one min section of selected segments of User 8. A helpful observation when looking at the segments was the difference in length of continuous data episodes between clear gait and clear non-gait behaviours. Due to gait's periodical and sustained pattern, the identified data segments tend to be longer. Non-gait behaviours on the other hand seem to vary more in their movements and therefore are more likely to get separated by the Switching AR.

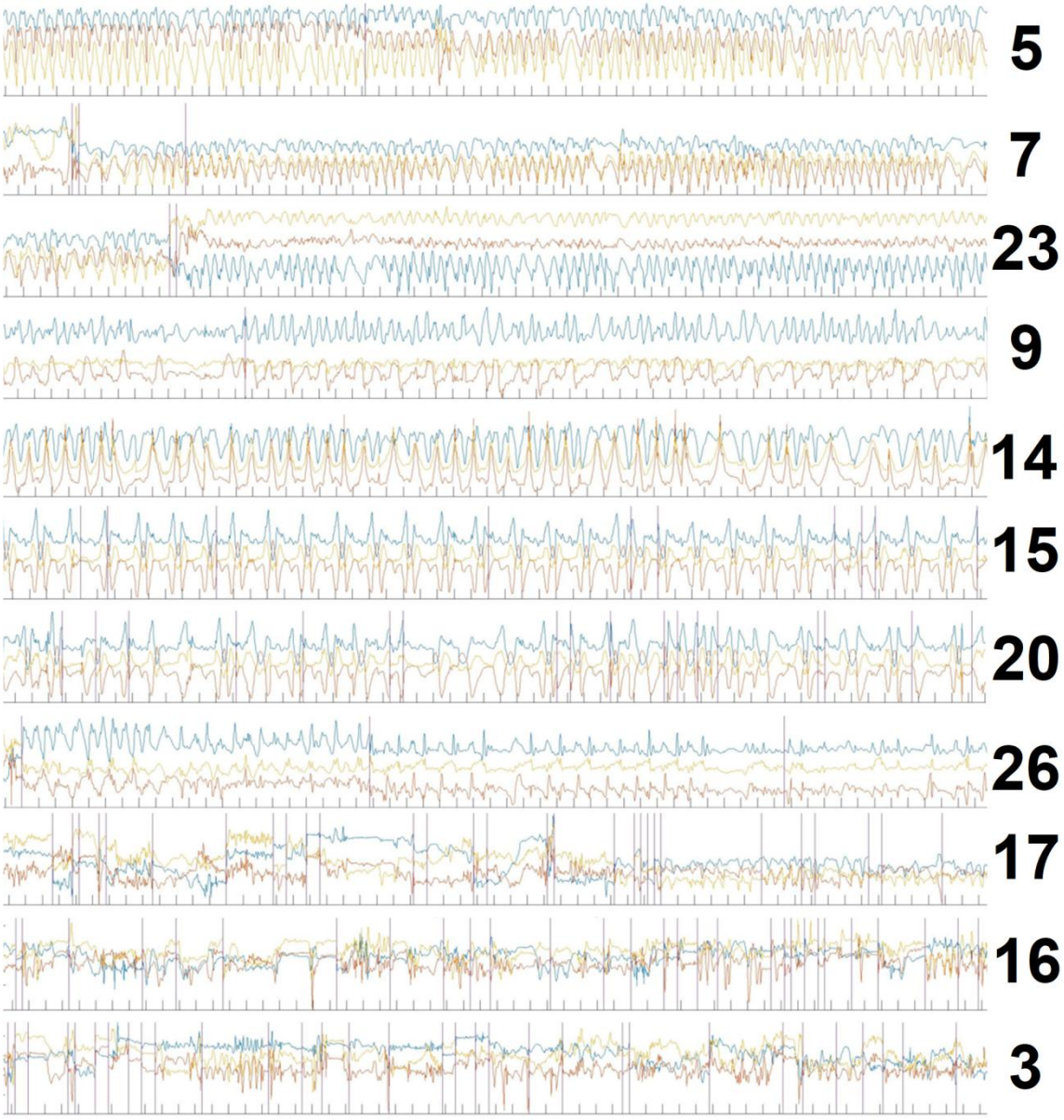


*Figure 21 - **Example of Switching AR output data**. User 8 from the 13th of September. One min section of selected segments visually sorted from most likely gait behaviour to least likely gait behaviour. Right: Segment label, Y-axis: raw accelerometer data (ai with blue = x-axis, red = y-axis, yellow = z-axis), X-axis: time in sec, purple vertical lines = interruption of consecutive data segments.*

But this feature also leads to some problems with the Switching AR. The window size of the segmentation was variable. A fixed flexible window size can cause different data patterns to be put into the same segment which would cause a loss of accuracy in classification. However, in at least one case of applying the Switching AR to the KOALAP dataset, the algorithm separated the left

and right steps of a clear asymmetrical walking behaviour into separate segments. We were, therefore, forced to include a minimum window size to avoid this from happening. Furthermore, we were not able to analyse the whole dataset of a person in one go. Feeding multiple days of the dataset into the Switching AR at once caused the segments to be overwhelmed. The segments displayed visually obviously not related data patterns. Therefore, we had to segment the dataset of a single participant into chunks of up to 7 h.

In addition, we decided to try to remove any clearly non-gait data before performing the Switching AR, using very simple and quick methods. During the late autumn of 2018, we explored the Short-Term Fourier Transformation (STFT) and the Normalised Autocorrelation Step Counting (NASC) as methods to remove resting behaviours from the dataset (see Chapters 4.2.3 and 4.4.4). The STFT has the window size trade-off where a smaller window size loses accuracy while a larger window size loses information in the frequency profile it analyses. The STFT output data is therefore very volatile and can be vastly changed depending on the thresholds used. The NASC was less likely to be effected by noise in the gait data. While we previously listed the NASC tendency to not identify interruption of periodic behaviour, for our purposes as a sedentary behaviour exclusion method this becomes a benefit: segments consisting of a mixture or gait and non-gait are not dropped, and can be further refined later. We, therefore, decided upon using the NASC.

Having used Switching AR to classify the remaining data into segments, the next step was to classify the segments as gait or non-gait. Our first approach used p-Welch. P-Welch enables us to perform a spectral analysis by computing the power spectral density estimate. The advantage of p-Welch is that it does not require you to know the spectral content of the signal. Figure 22 shows an example of the power spectral density estimate detected at the y-axis of the SRDS. A spectral power increase can be seen in the frequency range around 2 Hz, but also around 6 Hz and 4 Hz. This pattern reflects the periodicity of the gait behaviour and identifies periodicity as one of the main features to look for when identifying gait patterns. His pattern gets interrupted when confounding behaviours interrupt the regular gait pattern. During the walking behaviour, this occurs two times, first when the participant changes their walking style.

The maximum power drops into a lower frequency range (~1 Hz) when the walking behaviour was confounded by non-gait related movement of the arm. In this case, the participant adjusted a loose strand of hair. Looking at the non-walking behaviour data, the power spectral density estimates are lower and the frequency ranges with high power are wider.

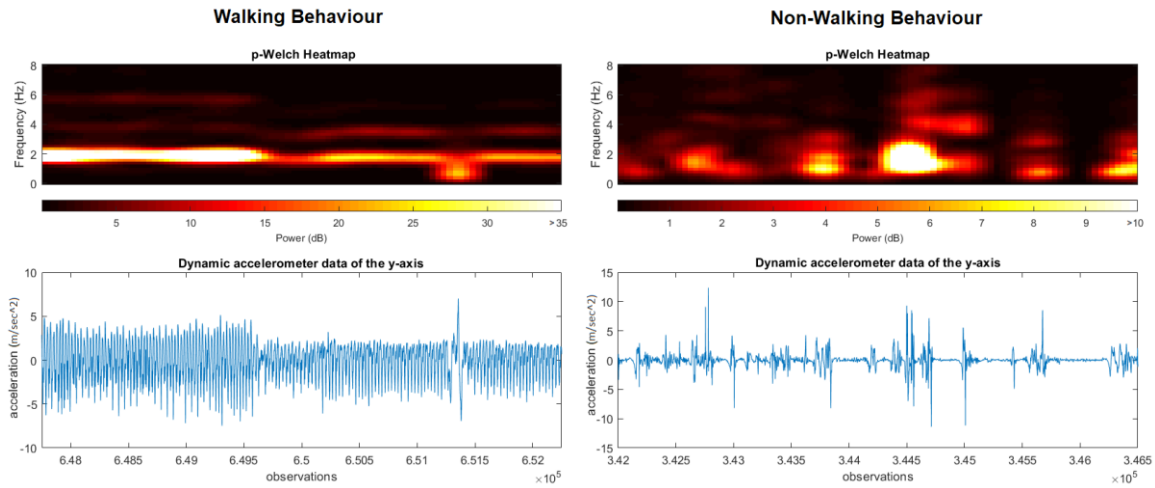


*Figure 22 – **P-Welch heat map of the SRDS data**. Walking behaviour examples taken from the free walking trial with free arm swing, non-walking behaviour taken from the coffee trial. Sliding window with a width of 6 sec sliding by 1 sec. Sample frequency = 50 Hz.*

While p-Welch gave us a good framework of what to consider while analysing the periodicity of the data, we had some problems translating the visual information from the heat map into a decision making tree to classify sections of data as either gait or non-gait. One difficulty was setting appropriate sliding window widths. Furthermore, different gaits exhibit different frequency ranges with different maximum power maximums. Finding thresholds appropriate for all subjects was, therefore, challenging. As can be seen in Figure 22, the active non-gait behaviour also tends to exhibit maximum power within the common walking frequency range of 1.4 to 2.5 Hz (Aitaterini & Ji, 2005). Asymmetrical walking can cause the frequency band of the gait cycle (2.8 – 5 Hz) to be more pronounced than the step cycle (1.4 - 2.5 Hz). All of these issues informed our gait pattern identification requirements further elaborated on in the next chapter. Nevertheless, the main problem we had with the p-Welch method was the long processing time. Later, in early winter 2019, we started looking into running a normalised cross-correlation (autocorrelation, AC) to further examine the gait data identified by the Switching AR. Disappointingly, short episodes of obvious non-gait behaviour still got combined with obvious gait behaviour in the same cluster. We found the AC to be able to give us the same information of interest as the p-Welch with a much shorter processing time.

Around this time we decided that the KOALAP data was not sufficient for the development of the algorithm. Therefore, we decided to create our own dataset with labelled behaviour data to get a better understanding of what gait (and non-gait) patterns look like when collected at the wrist. Between February and July 2019 we created the SRDS. The dataset enabled us to observe some of the above-mentioned problems. It also made it possible to examine the accuracy of our algorithm. This led to the decision to try to use the AC to “clean up” non-gait behaviours that slipped through the Switching AR.

Furthermore, we attempted to replicate the gait detection methods described in Brajdic’s and Harle’s (2013) paper. While we were unsuccessful in replicating the paper, it guided us in exploring gait detection and step counting methods, such as the NASC, STFT, Continuous/Discrete Wavelet Transform (CWT/DWT) and acceleration magnitude, energy of the acceleration signal and

standard deviation thresholding methods. Especially using standard deviation thresholding to explore the SRDS (see Chapter 4.4.3) inspired our own contribution to gait detection methodology. We developed the Dominant Axis Selection Approach (DASA) in late spring 2020. One main problem we run into that is specific to the location of the data recording is the reduction of three axes into one-dimensional data. This is most commonly done by calculating the acceleration magnitude of the dynamic data (dr). However, since the arms tend to introduce a lot of confounding behaviour to the data (see Figure 22) and the positioning of the sensor in relation to the user's body and the ground can change a lot, we determined that it would be better to extract data from one selected axis. Figure 23 illustrates the problem of using the acceleration magnitude to reduce tri-axial accelerometer data dimensions. While clear spikes of larger magnitude corresponding with each step can be seen in the dr data under the “free-swinging arms” condition, the condition with confounding behaviour (using a smartphone) lost this feature. This problem would not occur in data collected at the centre of the body or at the legs and is unique to the wrist-worn location. The DASA was therefore a methodology that we developed to overcome this challenge and is to our knowledge a completely novel approach.

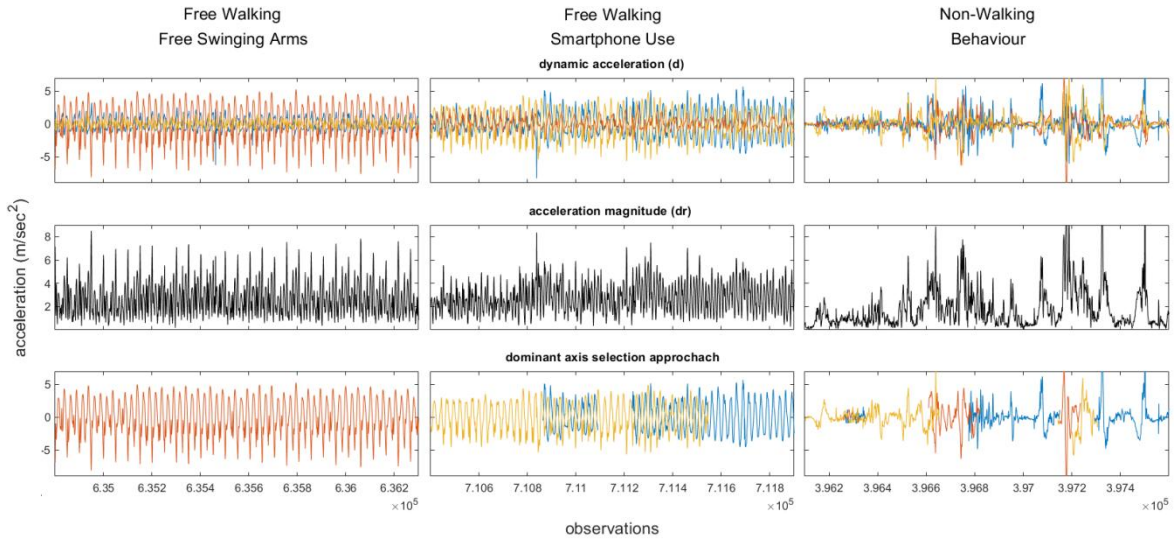


Figure 23 - ***Comparison of the acceleration magnitude and the dominant axis selection approach in reducing three-dimensional accelerometer data to one dimension****. Example of 90 sec of selected data for non-confounded gait and confounded gait. Blue = x-axis, red = y-axis, yellow = z-axis, black = acceleration magnitude (dr).*

Looking at Figure 23, we can see that clear gait recorded under free-swinging arm conditions results in an easy selection of the y-axis as the dominant axis by extracting the maximum percentage threshold difference to the dr data. However, under stiff conditions, the dominant axis can be more ambiguous and change abruptly when simply taking the maximum threshold difference (see Figure 24). Especially in the case of the ”hand in pocket condition” both the x- and y-axis have similar prominence After a lot of experimenting with conditions, we decided to keep multiple dominant axes and save them as individual “episodes of walking” for later steps in the algorithm. This leads to the next algorithm steps we developed with the DASA and the AC algorithm steps: extracting “Episodes” from the walking behaviour. In Figure 23, the clean “free-swinging arms” condition leads to the dominant axis being continuously identified as the “dominant” one. But looking at the coffee trail data, the length of a dominant episode sustained for each axis is

very short. In general, the hand orientation changes a lot more during non-gait non-sedentary behaviour but tends to stay constant during episodes of walking behaviours. Applying a code to remove “short episodes” is therefore a simple efficient way to remove non-gait data.

Figure 24 – ***Problems and advantages of reducing the 3-D accelerometer data to 1-D by selecting the axis with the largest percentage threshold****. X-axis = observations, Y-axis = acceleration of the (from top to bottom) selected dominant axis (DASA), uncut dynamic data, x-axis, y-axis and z-axis identified by the DASA; Blue, red & yellow = x-, y- & z-axis, grey = data discarded by the DASA, red cycles = selected axis change was bad, green cycles = selected axis change was good, blue cycles = selected axis change was neutral.*

At the end of summer 2020, the previous AGDA was mostly done. Its overview can be seen in Figure 25 but was never quite sufficient. After some introspection, we realised that the NASC and the episode length restrictions had the largest impact in excluding non-gait data and the DASA and the AC had the highest impact in identifying gait patterns. Furthermore, we repeated the use of the AC and the Switching AR took a massive amount of processing power and time. Consequently, we decided to optimise the code for processing time. This was especially important since we had applied to use the UK BioBank dataset by this time. That dataset is made up of one week of continuous accelerometer data recordings of over 100,000 participants. In the end, we decided to

include 17,000 of them in our investigation. Optimising the code's efficiency for processing time was therefore a relevant endeavour. Nevertheless, we would recommend the use of a Switching AR to examine the walking behaviour of individual patients or to differentiate between different working behaviours.

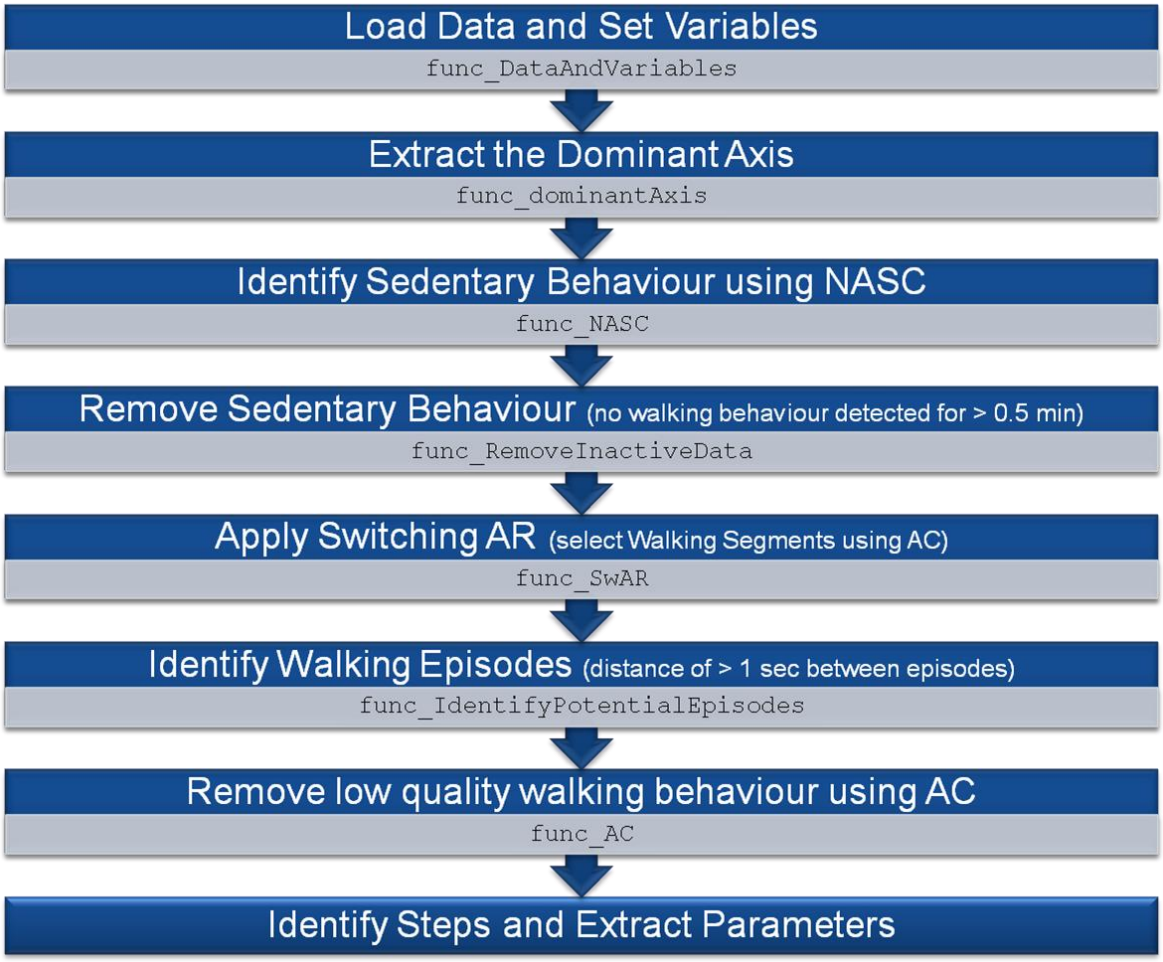


*Figure 25 - **Previous Aston Gait Detection Algorithm attempt**. Arial font = algorithm steps, Courier New font = MATLAB function names.*

In the end, we decided to discard the original code and used the building blocks we developed to completely re-write it. A description of the function of the final code will be discussed in the following chapters. First, we are going to summarise the requirement this final GDA was built around. Then we will go through the code and explain how it works. Finally, we will describe the efficiency of each step of the GDA, its processing speed and accuracy.

5.2.2. Requirements for the Algorithm

We decided that the AGDA needs to fulfil the following requirements:

- can be applied to data recorded on wrist-worn devices
- is applicable for data recorded in free-living conditions
- can run automatically without operator intervention
- does not rely mostly on inflexible cut off values
- needs to have a fast processing time

- needs to balance sensitivity and specificity
- takes advantage of the pattern underlying all gait behaviours

Applied to wrist-worn devices in free-living conditions

The above SRDS, KOALAP and UK BioBank datasets were recorded on wrist-worn accelerometers. Previous studies that identified gait parameters for data analysis tended to record the data in a laboratory environment and the sensors were mostly either located at the leg or fixed to the centre of the body. Accelerometers located on the legs or back do not tend to experience as much non-gait related movement as accelerometers located on the arms. Everyday activities such as cooking or eating can therefore introduce a lot of additional noise to the data that sensors located on the legs would not pick up on.

Also, sensors on the legs and back are fixed in position during walking and we can assume that gait will always affect one axis in particular. Since hand movements can change the wrist orientation towards the body and the ground this principle cannot be applied here. Furthermore, the hand can be occupied with other activities during walking, such as carrying a bag or handling a mobile phone, which introduces further noise sources and orientation changes.

Nevertheless, using wrist-worn accelerometers is valuable for research. The increased use of smart-watches, Fitbits and other consumer devices with integrated accelerometers can provide access to data recorded in free-living conditions with access to larger pools of participants and at lower costs. It is also more natural for participants to wear a wrist located sensor with additional functions, such as showing the time, than a sensor strapped to the leg or back due to the prevalence of watches, smart or not.

Flexibility between Participants and within the Datasets

Another requirement for the algorithm is to be flexible for use on various different individuals without adjustment. For example, inflexible cut-off scores such as setting an acceleration magnitude minimum of 0.6 g can lead to careful walking being discarded. If thresholds are required they should be sourced from the literature (see `fastWalk_obs` or `slowWalk_obs`), adjusted to the participant's individual data (see `PctDif_thld` and `peakprom_threshold`), or only consist of very low thresholds (see `STDthld_standart`).

Processing Time

Due to the large size of the KOALAP and UK BioBank datasets, we decided to make processing speed a priority of the algorithm. From the UK BioBank study, we selected 17422 participants to conduct analysis on, each with one week of data. Taken together this is around 334 years' worth of data. The KOALAP dataset “only” sums up to 2.18 years. Because of this, there are differences between the KOALAP and UK BioBank algorithms. They consist of a change in pre-processing that we do not consider a part of the gait detection part of the algorithm, as well as changes in the format data is saved in. This was also done to reduce the storage space required for this data.

While developing the algorithm, we were mainly concerned with speeding up processing steps that were repeated for each data file, while long-running commands that were only run once during the analysis were considered acceptable. An example is the command `clean all` at the start and end of the code, which is quite computing-intensive and not strictly necessary but is good housekeeping.

Identify Gait Patterns

Since we want to extract parameters of gait from the identified data, the algorithm needs to be **highly specific**. The data identified as gait needs to show a clear peak-valley pattern to identify step count, amplitude and speed. The accelerometer containing devices were worn at the wrist during the course of the day. Therefore, a lot of the data will not be gait pattern behaviour. A robust definition of gait and its expected parameters is necessary to ***exclude noise*** from intended hand movements from the identified gait. The size of the datasets (both KOALAP and UK BioBank) also requires that the algorithm is reliable enough to be ***applied to different participants*** without having to adjust parameters or manual adjustments. For our intended proposes it is more important to select data that is ***reliably identifiable as gait*** than to be able to spot every instance of walking behaviour.

Consequently, we aim to create an algorithm that will identify gait that is of a certain quality, rather than all instances where a participant was walking. This approach has consequences when interpreting the data later on. Low rates of walking behaviour could be the result of a lack of walking done by the participant, the participant's walking being too impaired to display an identifiable gait pattern, or due to confounding actions that impact the walking pattern. However, we can argue that nevertheless, we can draw similar conclusions about fluctuations in the severity of the patient's condition both in the case where walking decreases in quality or when the amount of walking decreases. Confounding actions effecting gait identification in healthy individuals will be minimal (e.g. stop and go walks, holding something, etc.) or representative of impaired walking behaviour (e.g. walking aids, holding on to railings, etc.).

We decided on giving the specificity of gait detection priority over sensitivity in the algorithm overall. However, in the first step of the algorithm that extracts potential walking behaviour from the dataset and removes them from the rest of the dataset, we favour sensitivity over specificity. During Gait Parameter extraction we favoured specificity over sensitivity.

Step Frequency

A minimum and maximum step frequency variable needs to be determined for running the algorithm. Identifying gait depends on its periodicity and setting limits on the frequency range at which this periodicity occurs can exclude periodic behaviours with too high frequencies (e.g. vibrations caused by shaking something) or too low frequencies. Öberg, Karsznia and Öberg (1993) extracted the gait parameters from a total of 233 healthy subjects between 10 and 79 years of age. They documented Step length, Step frequency and Gait speed at slow, normal and fast walking speeds by age and gender. We extracted their values from the Tables "Step frequency. Slow gait. Men.", "Step frequency. Slow gait. Women.", "Step frequency. Fast gait. Men." and "Step frequency. Fast gait. Women." We calculated the 99% confidence interval (mean±(STD/√n)*2.58)

and found Women aged 60-69 to be the slowest walkers with 1.31 steps/sec and Women aged 40-49 the fastest walkers with 2.78 steps/sec. In comparison, Aitaterini and Ji (2005) observed 800 uninformed people in two locations and observed walking frequencies between 1.4 and 2.5 Hz (step/sec). Zijlstra et al (2008) observed the walking behaviour of 40 physically active community-dwelling women aged 64 to 85. The lowest Step frequency they reported was 84.7 steps/min. Converting to sec and applying a 99% CI results in a Step frequency of 1.23 steps/sec. Furthermore, the slowest and fastest gait speed in Öberg, Karsznia and Öberg (1993) of 0.70 m/sec and 1.84 m/sec are close to the normally reported slow gait speed of 0.6 m/sec and bordering running gait speed of 2.0 m/sec (Wu, Simpson, van Asseldonk, van der Kooij, & Ijspeert, 2019). Wu et al. (2019) reported only a limited effect of gait speed on step variability, while Ground reaction force and step frequency got smaller with slower walking speeds. However, this data describes the behaviour of healthy individuals. Since we want to use the data to investigate walking behaviours in individuals with RMDs we will have to determine a minimum step frequency that considers people with impaired walking.

5.3. Algorithm Development

5.3.1. Algorithm Overview

The algorithm is divided into two main sections, the Gait Detection Algorithm (GDA, `AstonGDACode_GaitDetectionAlgorithm.m`) and the Step Parameter Extraction (SPE, `AstonGDACode_StepParameterExtraction.m`) algorithm (see Appendix). Before the gait detection, the data needs to be pre-processed and loaded in a format suitable for the rest of the code. This part of the code needs the most adjustments when being applied to a new dataset. Between the Gait Detection and Parameter Extraction algorithm parts, some reformatting needed to take place. For the UK BioBank data, this reformatting is part of the Gait Detection Code, while for KOALAP reformat was handled by a separate code.

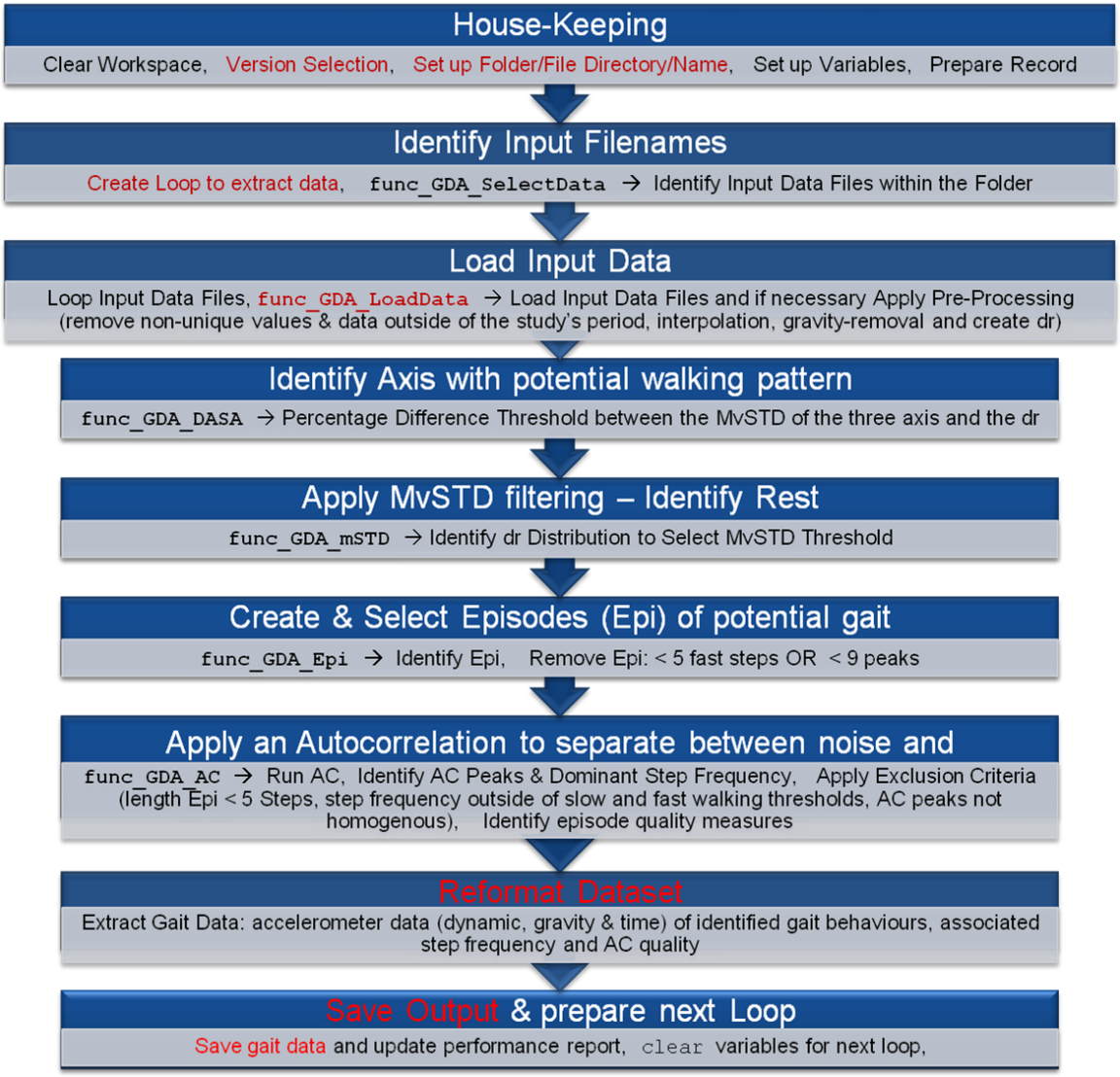


*Figure 26 - **Overview of the final Gait Detection Algorithm**. Functions used are marked in the `Courier New` font. Red = parts of the algorithm that need adjustments between datasets.*

All coding for the Gait detection algorithm is performed within Matlab. Figure 26 provides an overview of the Gait detection code and its functions. A Matlab function is a code that runs with the variables put into the function and only outputs the resulting variables of interest. In the figure steps names with the prefix "`func_GDA_`" contain functions. The whole code and its functions can be found in the Appendix. Steps marked in red contain substantial changes between datasets.

The functions help to keep the main code more readable. Furthermore, they helped during programming, since we were able to try different versions of sections within a function, without accidentally damaging the main code. Using functions was something that we implemented later in our programming endeavours and that greatly helped organise the code. For example, we were able to experiment more easily with the order in which functions were executed.

5.3.2. Housekeeping

General housekeeping is included in the code and most functions. The algorithm starts by clearing the workspace with `clear all`, closing all still open files with `fclose('all')` and images with `close all`. All of them are important to ensure maximum efficiency by clearing the

memory space used by Matlab. In addition, the function `tic` starts a stopwatch that can be used to extract information about the processing times in combination with `toc`.

The code has an option to create images that illustrate the individual steps of the algorithm: this was used to create the figures in this chapter. Since creating too many images can cause Matlab to crash, we programmed failsafes into the code that will set this variable back to zero if you attempt to make images of more than 5 files at once or on a file that contains more than 12 h of data.

The dataset version the code should run on is determined by `Version_Dataset`. The selected version mainly determines which directory the code used. `CHOICE_dir_basis` describes the folder the dataset is located in, while `CHOICE_dir_Output` selects the folder for the output data. `filename_load` identifies a component in the filename that is the same for all input files within the input folder. The code is built to identify files with data to analyse within a folder, to run the algorithm on these files and create an output file with the results. For the KOALAP dataset, where data is saved in a folder for each participant, a loop is activated that goes through the folders in `User_List`. The variables `x_str` and `x_end` can select specific files to be run with a folder. If `x_end` is left empty (`x_end = [];`) all files are run. For the UK BioBank dataset data is separated into 31 folders with 562 files each. `folder_n` selects which folder the code should run on and `iteration_add_on` will add to the name of the Variable and Processing output files to prevent overwriting these files when the code needs to be run multiple times within the same folder.

The location of functions needed for the code is written in `CHOICE_dir_basis_sup`. The function `addpath` grants Matlab access to both the input files and the function codes. Output files are saved in the right location by using `cd` to open the output folder to write in.

Only some Variables need to be defined for the Gait detection algorithm. The variable `var_feq` is the value of the frequency in Hz at which the data was collected.

The next variables are dependent on our definition of gait in regards to the requirements on the Gait Detection algorithm. We decided to set the minimum number of steps for this episode (`minEpisode_steps`) to 5 steps. The threshold for the slowest and fastest walking speed are recorded in the variables `slowWalk_obs` and `fastWalk_obs`. All Variables are saved in a Matlab file called `'SetUP_Variables_idx[iteration_add_on].mat'` for future reference. Furthermore, the variable `RunTime` is set up to keep track of the processing times for each data file.

Lastly, most functions that are used to identify walking behaviour need the input `z_Base` and `z_Base_3D`. These two variables create a vector of ones and zeros that label each data point as either potential walking behaviour (1) or non-walking behaviour (0). The code used these vectors to determine which data to analyse. In the beginning, all samples should be set to 1 and each step reduces the amount of data considered as potential gait. If the variables are left empty they will be treated as vectors of ones. The functions also output these vectors as, for example, `z_domAxis` and `z_domAxis_3D`. These variables will be referred to as z variables for the rest of the chapter.

5.3.3. Select Data Files for Analysis

`func_GDA_SelectData`

While we only need to read in two files for the Self-recorded dataset, the original and the randomised version, for the KOALAP and UK BioBank datasets the files are organised in multiple folders. The function identifies the documents inside the folder of interest and excludes files one by one that do not have the same format or naming components that we expect to see in the data that we want to analyse. When this algorithm is used on a different dataset this file should be checked to control the naming conventions of the data input and output. The function used `Version_Dataset`, `CHOICE_dir_basis` and `filename_load` to output `filename_load_pre`, `filename_load_pos` and `filename_save_pre`. These variables provide a list with file name components where 'load_pre' is the start and 'load_pos' is the end of the file. While not needed for UK BioBank where raw accelerometer and time data are saved within one .cwa file, previous pre-processing done on the KOALAP data caused the data to be separated into files by data type.

`filename_save_pre` gives the option to change the naming convention of the file the data is later saved in. Within the code, `filename_save_pre` can be used when creating the output-filename-variable `fn_save`. The output lists of this function are run in a loop to process the data one file after another.

5.3.4. Loading and Pre-Processing the Data

`func_GDA_LoadData`

We input the previously described filename list variables (`fn_load_pre`, `fn_load_pos`, `CHOICE_dir_basis`), the sampling frequency (`var_feq`) and the dataset Version indicator (`Version_Dataset`) and output the dynamic acceleration data (`data_d`), magnitude of the data (`data_dr`), the time-stamp data in the Matlab datenum format (`data_t`), the gravitational component (`data_g`), the total dataset length (`lgt_TotalData`). Furthermore, the function will output the walking labels for the data from the Self-recorded dataset (`data_Lab` and `data_l`, see Chapter 4.3.4) and it has the option to output the raw data (`data_raw`).

KOALAP and SRDS

This function contains the most changes between datasets. In the case of the SRDS and KOALAP the data was saved in previous pre-processing steps. The process is explained in more detail in the wrangling sections of the SRDS (see Chapter 4.3.4) and KOALAP (see Chapter 7.5.4.4) methodology chapters. The data is saved in a binary format. For example, to load the dynamic data the file needs to be opened using `fid_load = fopen(filename,'rb', 'ieee-be')` read using `data_d = fread(fid_load, [3 inf], 'float32')` and closed using `fclose('all')`. Fclose is especially important to relieve the memory used by the algorithm that keeps the file open. Forgetting to close the opened files would put a lot of stain on Matlab, slowing it down and making it crash when looped too many times. The fread function requires

information about the format the data was saved in. The following formats are required to read in the data files:

- Raw accelerometer data `fread(fid_load,[3 inf],'float32')`
- dynamic data `fread(fid_load,[3 inf],'float32')`
- gravity component data `fread(fid_load,[3 inf],'float32')`
- time data: `fread(fid_load,[1 inf],'float64')`

In case of the SRDS the data labels are also loaded:

- Hand labels: `fread(fid_load,[1 inf],'float32')`

Some dynamic data was lost during the previous KOALAP data wrangling step. We decided to delete the previously calculated dynamic data and added the calculation of the dynamic data for the KOALAP dataset in this function.

UK BioBank Dataset

For the UK BioBank dataset, we only had the raw .cwa files, since we applied this algorithm as soon as the data was downloaded from the UK BioBank servers. To load the cwa. file we used the open-source code by Nils Hammerla (2012). We extracted the raw accelerometer and time data, removed non-unique values in the data, removed recorded data outside of the one-week experimental recording, applied interpolation and removed gravity.

A standard cubic spline interpolation utilising the Matlab `interp1` command is used on the raw data at the sampling frequency. Interpolation is done to preserve the temporal form of the data during analysis. Processing of the data required the dynamic data to have exactly the same time interval (the sample frequency) between each data point. However, sensor batching issues during data recording can lead to slight differences in the sample frequency recorded and can even lead to several samples having exactly the same timestamp. Therefore, it is not possible to determine the accurate temporal order of the samples. This was a particular problem with the KOALAP data (see Chapter 7.5.4) since the raw data was not stored in the same order as it had been generated in. While we could have assumed the UK BioBank data to be in the temporally correct order, we still remove non-unique simultaneous values to be able to apply interpolation. Applying a histogram of the difference between timestamps (`diff(data_t)`) on one UK BioBank file, we observed that the sampling rate fluctuates between 95, 100 and 105 Hz. We, therefore, decided to apply an interpolation.

The UK BioBank data is recorded continuously for one week. Since there are no breaks in the data recording we don't have to consider breaking the data into different recording instances before applying interpolation. However, we found cases where some timestamps lay outside of the one-week recording window. Since these days only contained a few sec of data each, we solved this issue by removing every day with less than 5 min of data recordings.

While the KOALAP and self-recorded dataset applied an L1-trend filter to remove gravity from the dataset, we used an IIR filter to make the code more time-efficient. We set the second-order lowpass filter to 0.5 Hz (`[b,a] = butter(n,lowPassF/(var_feq*0.5),'low')`) and ran

the data both forward and backwards through the signal (`data_g = filtfilt(b,a,data_ai)`). The L1-trend filter stays more constant between detected changes in gravity and is, therefore, less susceptible to slight fluctuations within the walking behaviour. A slight up and down corresponding to the walking behaviour can be observed in the IIR filter, however, due to the regularity of the gait pattern it only causes the amplitude of the gait pattern to be slightly repressed. On the other hand, the IIR filter has a slightly better cut-off between gravity changes than the L1-trend filter. Overall, the IIR filter and the L1-trend filter produce near identical results when looking at the data labelled as walking behaviour in the self-recorded dataset. However, non-walking behaviour was shown to be better presented by the L1-trend filter. Since we are only interested in data representing walking behaviour and due to the large performance increase of the IIR filter over the L1-trend filter we decided to run the IIR filter on the UK BioBank data. Since we are not comparing the KOALAP and UK BioBank data with each other and since pre-processing was already done, we kept the L1-trend filter for the KOALAP dataset. In order to prove that the change to the IIR filter did not influence our results, we started a 4th Dataset version, in which we applied the IIR filter to the self-recorded dataset. We noticed that it slightly reduced peak acceleration. In data displaying walking behaviour, this reduction was mostly not noticeable, but it was noticeable in the non-walking behaviour, due to its irregular walking pattern. Using the IIR filter instead of the L1-trend filter was, therefore, justifiable to improve processing speed significantly. However, we would not recommend comparing step parameter output data of datasets with those that used different filters.

At last, for all datasets, the magnitude of the data (`data_dr`) is calculated.

5.3.5. Dominant Axis Selection Approach

The Dominant Axis Selection Approach (DASA) is based on observations from Chapter 4.4.3 and we already explained the use of the DASA in the background Chapter 5.2.1. We have to reduce the three-dimensional accelerometer data to one dimension. Gait behaviour recorded from a wrist-worn accelerometer will most prominently act on the axis along which the arm swings in “swing” conditions or the axis that is orientated towards the ground in the “stiff” condition. The other axis can show a similar pattern to the dominant axis; however, they will be much more subdued.

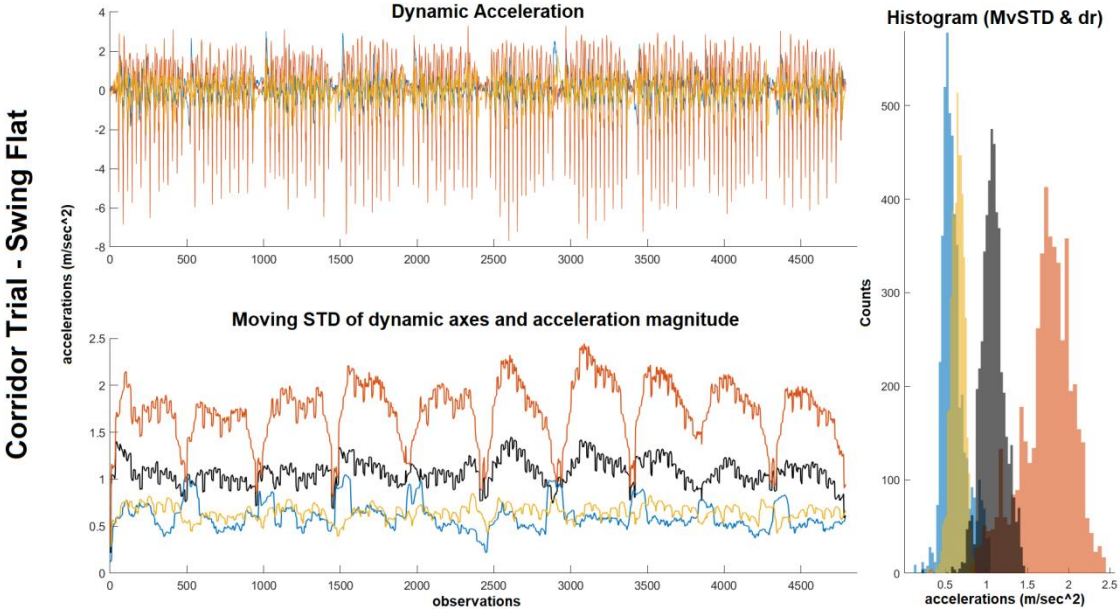


*Figure 27 - **Moving STD on a "swing" walking behaviour**. Blue = x-axis, red = y-axis, yellow = z-axis; black = dr. 50 observations = 1 sec.*

Looking at the histograms of the moving STD of the magnitude data and the individual axis of the dynamic data, we can see a relationship between the amplitude of the axis that shows the most pronounced gait pattern and the magnitude data. Comparing the STD of each axis enables us to identify the axis that produces the highest amplitude. In Figure 27 we can see that the walking behaviour is displayed dominantly on the y-axis. Looking at the histogram of the data the y-axis STD is much higher than the STD of the magnitude of the data. However, when looking at “stiff” walking behaviour multiple axes can display a standard variation higher than the STD of the data magnitude (see Figure 28). The moving STD data of all labels can be found in the appendix, Figure 88.

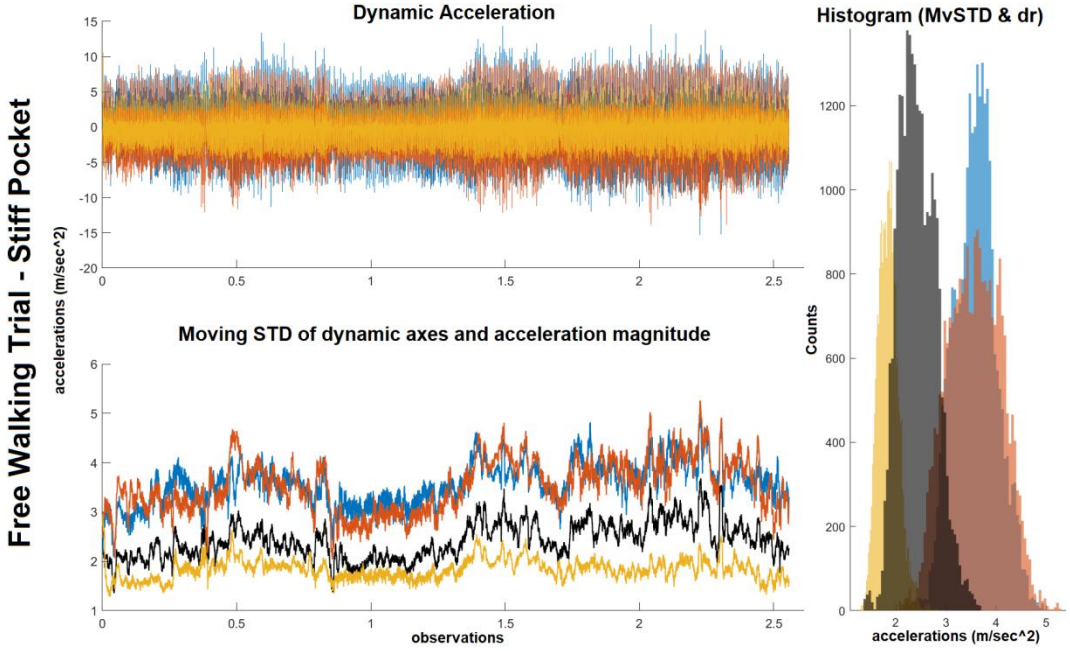


*Figure 28 - **Moving STD on a "stiff" walking behaviour**. Blue = x-axis, red = y-axis, yellow = z-axis; black = dr. 50 observations = 1 sec.*

`func_GDA_DASA`

We aim to evaluate the four acceleration data types (dynamic data of the x-, y- and z-axis and the magnitude data (dr)) against one another to create a vector that indicates on which axis the dominant gait behaviour occurs. The function requires the inputs of the z variables, dynamic data (`data_d`), magnitude data (`data_dr`), sample frequency (`var_feq`) and the length of the dataset (`lgt_TotalData`). The output consists of the z variables and the `data_thldPct`, which indicate the distance between the dominant axis and the magnitude STD.

The first step is to calculate the *moving standard deviation* (MvSTD) of each axis. We are using the function `movingstd`, written by D'Errico (2021), with a window size of 1 sec.

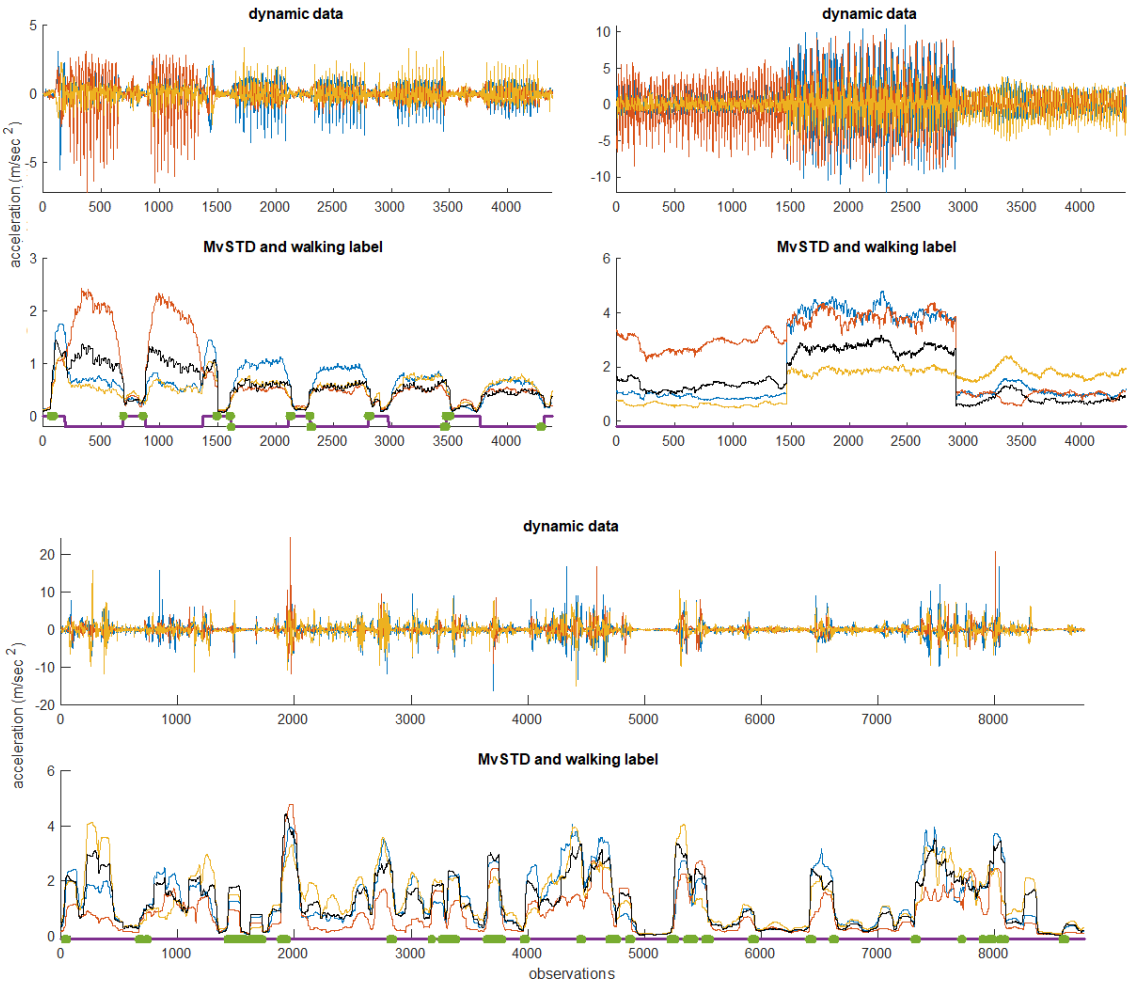


Figure 29 - ***Examples of MvSTD for walking behaviours*** *(above)* ***and non-walking behaviours*** *(below). Blue = x-axis, red = y-axis, yellow = z-axis, black = dr data, purple = gait label (0 = non-gait unknown behaviour, -0.1 = non-gait coffee drinking, -0.2 = walking bahaviout), green = locations where dr MvSTD > max(axis MvSTD), 50 observations = 1 sec.*

Figure 29 displays a plot of all four MvSTDs and the walking label. We can observe that the MvSTDs of the dominant axis is always larger than the dr's MvSTD. This can be observed very well in the "Free Walking examples". However, this difference gets smaller with the smaller the amplitude of the gait behaviour is. In the Corridor walking example, the first two walking episodes represent a gait with free-swinging arms, while the last four episodes represent the "stiff" behaviour of balancing a cup of water. Due to the careful walk, the signal amplitude is suppressed. However,

the MvSTD of the dominant axis consistently stayed above the dr's MvSTD. On the other hand, noisy data, such as between walking episodes or within the "Coffee Drinking examples" have a lot of instances where the dr MvSTD becomes larger than any other dynamic MvSTD. These occurrences can be seen in the green dots in Figure 29 and coincide often with the start and end of walking episodes. Furthermore, if a dominant axis is identified it is not sustained for longer than a few sec.

Afterwards, a *moving mean filter (*MvM*),* was applied to smooth the data to be used to determine the dominant axis (see Figure 30). Smoothing the data ensures that changes between dominant axis selection occurs less rapidly.

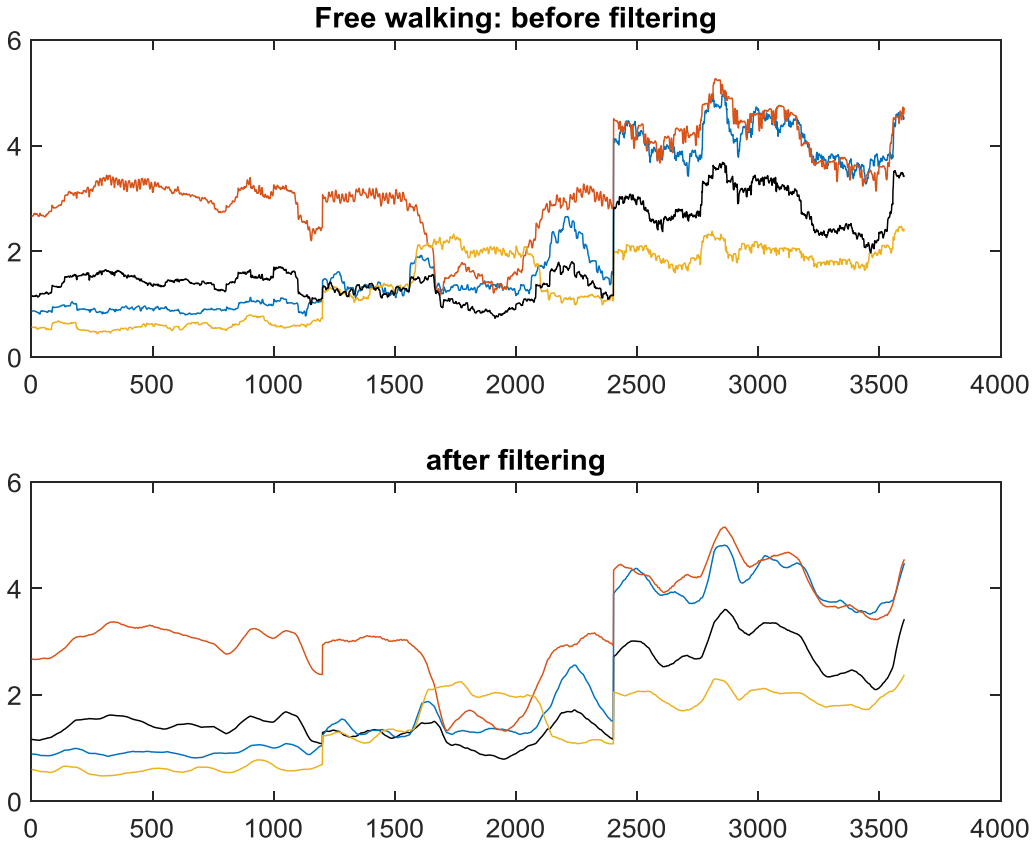


Figure 30 – ***MvSTD before*** *(above)* ***and after*** *(below)* ***filtering****. Y-axis: acceleration (m/sec2), X-axis: observations (obs, 50 obs = 1 sec). Blue = x-axis, red = y-axis, yellow = z-axis, black = dr data.*

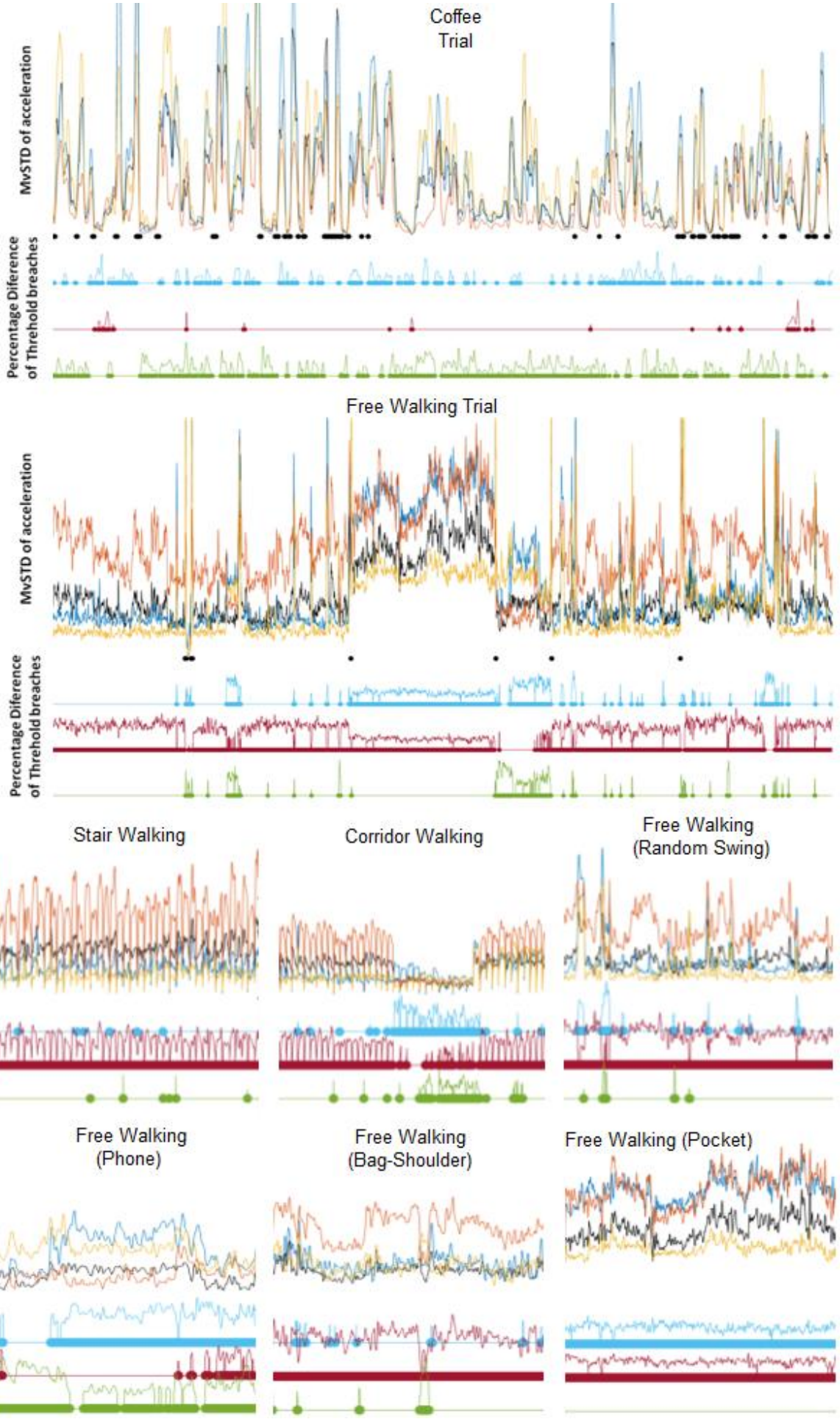
To determine the dominant axis, the distance and the percentage difference of the 3d axes from the dr axis were calculated by subtracting the dr MvSTD from each d MvSTD and by dividing the resulting distance by the dr MvSTD (i.e. for the x-axis: `data_PctDiffMvSTD_ax3 = (data_fMvSTD_ax3 - data_fMvSTD_dr)./data_fMvSTD_dr;`).

The maximum value of the *Percentage Difference* of the dynamic axes is taken and divided in half to receive the *Percentage Difference Threshold* (`PctDiff_thld`). It is then used to identify the axes that are closer to the dominant MvSTD than the dr MvSTD. Every axis that breaches the *percentage difference threshold* is labelled a potential dominant axis (`z_domAxis_3D`) and the *percentage difference* is also recorded (e.g. `data_thldPctDiffMvSTD_ax1`, Figure 31).

Figure 31 displays the percentage difference of data that reached the percentage difference threshold. As previously observed, we obtained longer episodes of data labelled as gait in the gait labelled data (above) than in the non-gait labelled data (below). However, nevertheless, some larger episodes on non-gait data remain. Figure 31 provides further examples for all gait behaviours.

It should be noted that we compare the magnitude of the whole data with the dynamic data of the individual axis. While this does sound odd, we also attempted this method by transforming the axis dynamic data into magnitude data before the MvSTD. This causes the dominant axis to overlap with the magnitude of all axis significantly. Both approaches come to the same results so we stuck to the first method.

Figure 31 - ***MvSTD and corresponding percentage difference for each axis of the coffee, free walking trail and selected conditions****. Dark blue, light red & yellow = MvSTD of the x-,y-& z-axis; light blue, dark red & green = axis MvSTD > dr MvSTD x-, y- & z-axis, black = dr MvSTD > all axis MvSTD. ↓*


Coffee
Trial
MvSTD of acceleration
Percentage Diference of Threhold breaches
Free Walking Trial
MvSTD of acceleration
Percentage Diference of Threhold breaches
Stair Walking
Corridor Walking
Free Walking
(Random Swing)
Free Walking
(Phone)
Free Walking
(Bag-Shoulder)
Free Walking (Pocket)

5.3.6. Removal of Low-Activity Data

We remove sections of the data whose amplitude is too low to be evaluated for step patterns. Low-level noise can reach frequencies that fall between the slow walking and fast walking thresholds and can accidentally get labelled as gait. This step will be particularly important when handling the UK BioBank dataset. Since the participants were encouraged to wear the Axtivity wristband 24/7, we will have a lot of data that will consist of the participants sleeping or otherwise resting. This will aid in excluding data where the amplitude is too low to be considered gait.

Beyond the reduction of data, there is a second reason this step is important for our algorithm. The DASA-AC approach is mainly concerned with the pattern of the data and does not make use of amplitude information. In previous runs, this resulted in data being identified as gait with implausibly low amplitudes of around 0.05 g.

`func_GDA_mSTD`

The function takes as input the z variables, the magnitude data (`data_dr`), the sample frequency (`var_feq`) and the maximum MvSTD threshold (`STDthld_standart`). It outputs the z variables and the adjusted MvSTD threshold (`output_stdthld`). While both 1D and 3D z variables are given, the data only gets analysed at the 1D level.

The first step to exclude seated/sedentary behaviours is to apply a STD-threshold. The moving STD filter was applied in 1-sec windows on the data. Figure 32 shows a histogram with probability normalisation of the magnitude of the data (dr) of the full self-recorded dataset, the data labelled as gait, data labelled as the coffee trial, and data between trials labelled as non-gait. The latter is derived mostly from sedentary data since the person who was recorded was particularly careful not to move too much between walking trials. But even the non-walking coffee trial data tends to be primarily in a lower acceleration range than the walking trial data. Looking at the histogram of the moving STD data, this difference gets even more pronounced with the intersection between the increasing gait labelled data and the decreasing no-gait labelled data lying at around 0.6 g (0.59 g for the sedentary and 0.65 g for the coffee trial data). This re-enforces the notion that 0.6 g makes for a good moving STD threshold.

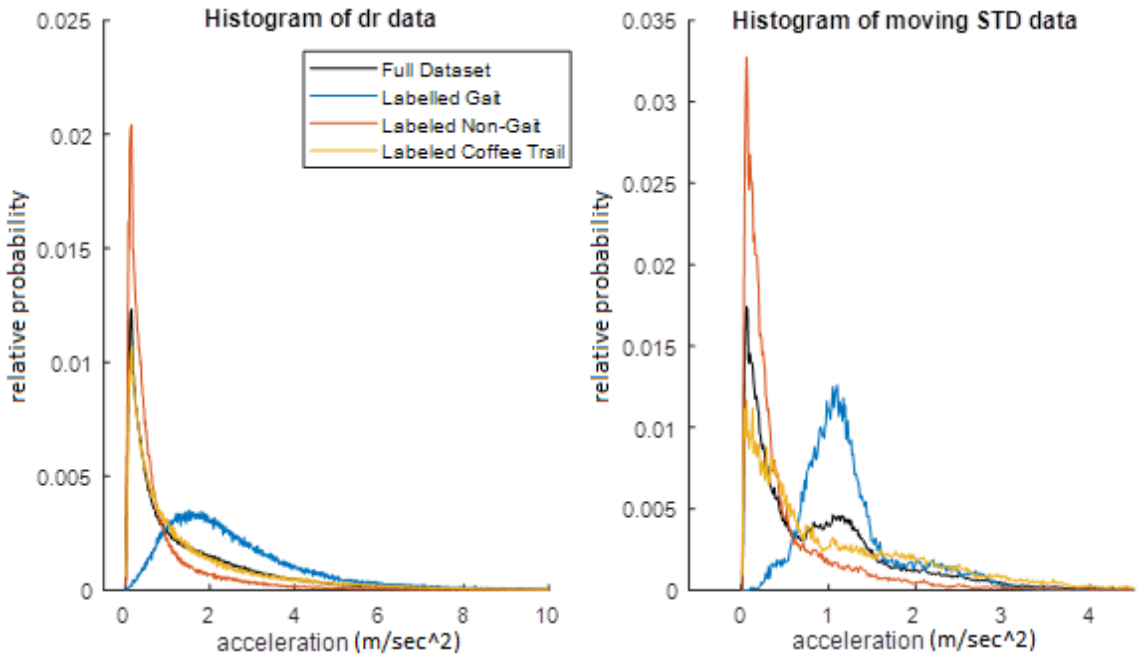


Figure 32 - ***Histogram with probability normalization of the magnitude of the data*** *(dr, left)* ***and the moving STD*** *(right). Black = of the full SRDS, Blue = of the SRDS labelled as gait, red = of the SRDS labelled as non-gait, yellow = of the coffee trial data.*

Usually, the threshold used for gait detecting STD thresholding is 0.6 g. However, we do not plan to identify gait through this step, but simply exclude anything that could not possibly be gait. We also run into the problem that the gait pattern in the “mug” condition did not reach the amplitude necessary to breach the 0.6 g threshold (see Figure 34). Consequently, we wanted to set a much lower threshold for the data. Setting the threshold at 0.3 g worked but felt somewhat arbitrary. As we wanted to make sure that the threshold was flexibly determined by the participant’s data, we, therefore, decided to make the threshold dependent on the histogram of the dataset for each file.

While the histogram of the moving STD of the full dataset shows a nice dip at 0.68 g that would seemingly make a for a good starting point to identify a threshold, our dataset does not represent a realistic ratio of walking/non-walking behaviour. 25.52% of the dataset consists of walking behaviour. That would mean the participants had to have walked an average of 6.13h a day, which we do not assume to be the common behaviour. Looking at the participant data we were not able to see such a dip in data. Hence we look at the large spike at 0.15 g for the dr data.

We calculated the histogram by using `histcounts` on the dr data using a bin width of 0.01 g. Then we used `findpeaks` to identify the location of the low amplitude spike. We multiplied the resulting amplitude by 2 to obtain a new potential moving STD threshold (`i_thld_pass`). If this threshold did not exceed the recommended 0.6 g threshold, we used this new threshold to identify potential walking behaviour by labelling data whose MSTD did not reach the threshold as non-walking behaviour. For the UK BioBank data, we set the maximum threshold to 0.3 g.

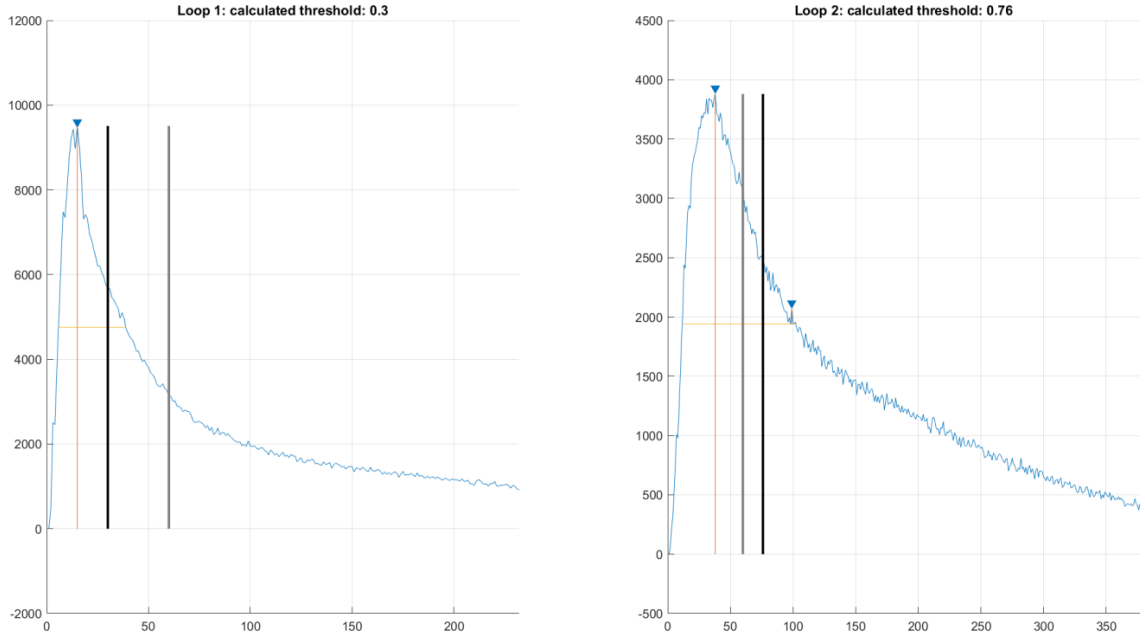


Figure 33 – ***Histogram of the dr data and the thresholds identified with each loop****. Y-axis: counts, X-axis = acceleration (m/sec^2). Blue = dr, blue triangles = identified peaks, red & yellow = identified peaks height & width, grey = maximum STD threshold, black = identified STD threshold.*

While running this approach on the self-recorded dataset worked immediately (see Figure 33), the thresholds identified for the KOALAP participants were very low. Generally speaking, the more sedentary behaviour we have in the dataset, the closer the identified threshold will be to 0g. We found that removing the non-walking behaviour data identified with the threshold and running the histogram analysis again got us closer to a suitable threshold. We repeat the process and stop before the threshold identified is larger than the maximum threshold (`STDthld_standart`) that we set as 0.6 g for the KOALAP and 0.3 g for the UK BioBank data.

We tried the same approach using the MSTD data instead of the dr data, but the results were less satisfactory and more volatile. Furthermore, we acknowledge that using `findpeaks` to identify the peaks is not the best approach to minimise the processing speed that we could have taken. It remains as an artefact when we tried to pursue finding a valley in the MSTD data. Since the processing time for this function is very small, we forgot to look for improvements here. `peakseek` could be used instead and was used in the `func_GDA_Epi` function. However, the std function only used `findpeaks` for a maximum of 7 times, while the `peakseek` in the episode function runs 2489 times on the self-recorded dataset.

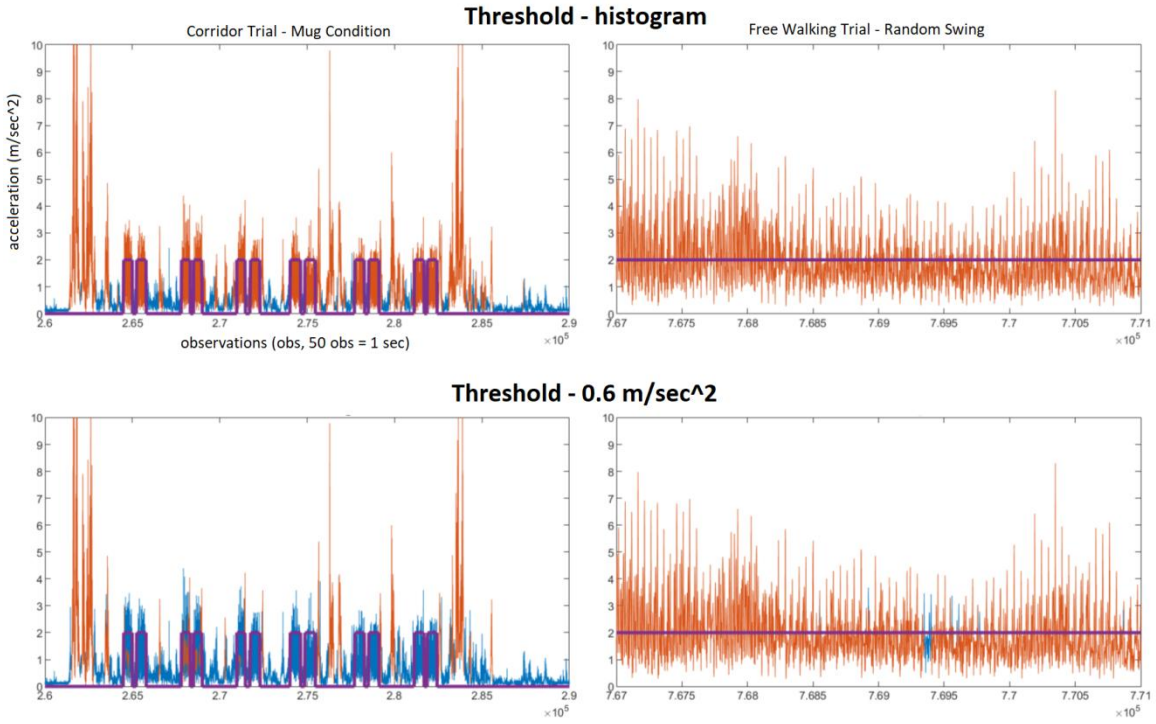


Figure 34 - ***Difference between 0.6 g*** *(below)* ***and histogram threshold*** *(above). Blue = dr data below the threshold, red = dr data above the threshold, purple line = labelling of the data with 0 → non-walking and 2 → walking behaviour.*

The difference between the 0.6 g and the histogram threshold can be seen in Figure 34. It illustrates the above-mentioned problem of excluding low amplitude walking behaviours with a typical threshold of 0.6 g. This problem was not only present in the intentionally careful walk of the "mug" condition but could also be found in the free walking trial.

5.3.7. Create Episodes of Potential Gait Data

In the next function, we identify the start and endpoints of the potential walking episodes and clean them up a bit by removing episodes that are too short or do not show enough spikes to be potential walking episodes of more than 5 steps.

`func_GDA_Epi`

In order to run an autocorrelation on the identified potential walking episodes, we first need to identify the start and end-points of each potential episode. We input the 3D z variable, the dynamic data (`data_d`) and the total data length (`lgt_TotalData`) and out threshold variables `slowWalk_obs`, `fastWalk_obs` and `minEpisode_steps`. The output consists of both 1D and 3D z variables and a list of start and end points, the axis and length (`epiList_loc`) of the identified potential walk episodes. This list consists of episodes of potential gait on the x-axis, followed by the y-axis and z-axis.

First change points are identified for the 3D z variables using `find(diff(z_Base_3D(i_axis,:)) ~= 0)`. We look at the value of z at identified start and endpoints to remove non-walking episodes. Then we calculate the length of the episodes.

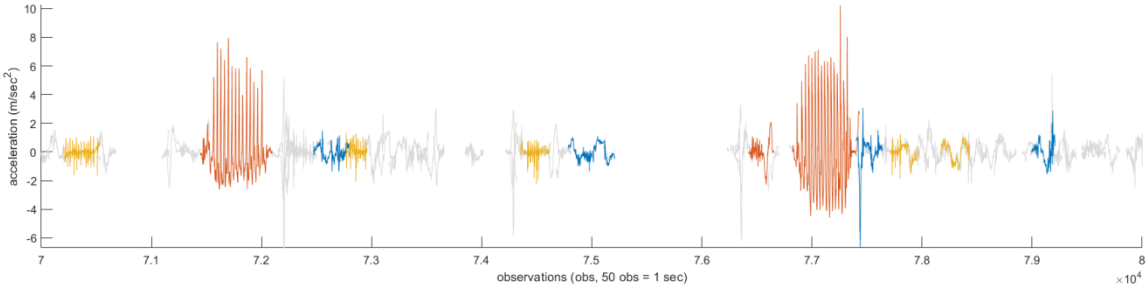


Figure 35 - ***Comparison between data identified as walking by the mSTD and Episodes function****. Grey = dynamic data identified as potential gait by "mSTD"; blue, red & yellow = x-,y-,z-axis of dynamic data identified as potential gait by "Episodes"*

Any episodes that did not reach the minimum time needed to make 5 fast steps of 1.8 sec (`var_minEpi_lgt_fastWalk`) were excluded. Then we looked at episodes that were shorter than the maximum time needed to make 5 steps of 4 sec (`var_minEpi_lgt_slowWalk`) and used the Matlab function `peakseek` (O'Connor, 2021) to identify peaks and valleys in the data with a minimum distance of one fast step (`var_MinPeakDistance`) and a minimum height of the mean data magnitude of the episode (`thld_amp_hill` and `thld_amp_vall`). If there were combined fewer than 9 peaks and valleys, we excluded the episode. Figure 35 shows the data that was identified by the mSTD function as potential gait and overlays it with the data identified by the Episode function as potential gait.

As described beforehand, we had some performance issues with the `findpeaks` function that we used previously. We compared the processing speed of the "Episodes" function and the "AC" function on the self-recorded data under the three conditions of a) not checking for at least 5 steps; b) using `findpeaks` to check for at least 5 steps; and c) using `peakseek` to check for at least 5 steps. Not looking for episodes with less than 5 steps increased the processing time of AC to 4.62 sec and decreased the time of Episodes to 0.02 sec for a sum of 4.63 sec. Using the `findpeaks` function the AC only took 1.08 sec, but the episode's performance time increased considerably to 2.95 sec to a total of 4.03 sec. Using `peakseek` resulted in a processing time of only 0.18 sec for the episodes and 1.98 sec for the AC and gives the best processing time performance of 2.16 sec total. While `peakseek` does not perform quite as well as `findpeaks` in removing episodes with less than 5 steps, it is more effective in achieving the primary concern of cutting down on processing time considerably.

5.3.8. Identify Potential Walking Behaviour using Autocorrelation (AC)

The sustained periodicity of the gait pattern within a certain frequency range is the defining feature we are looking for when separating gait from non-gait behaviour. All walking behaviours have a repeating pattern in common that relates to the Gait Cycle while few other behaviours demonstrate such periodicity, especially over longer periods of time. An exception would be vibrations of driving vehicles; these however should have already been caught by the moving STD function.

Due to the variety of walking behaviours practised by an individual, the variety of gait between people and the variety introduced by noise factors, we see periodicity as the most promising approach to identifying gait. However, it has the disadvantage of excluding noisy or particularly bad

walking episodes. We have to concede that it is not realistic to identify all walking behaviours within a dataset. Focusing on patterns that can be reliably judged as gait has to be sufficient. Furthermore, even if only healthy gait is extracted from the participant, we can still observe changes in the amount of healthy gait, speed of healthy gait and regularity of healthy gait that should be of interest.

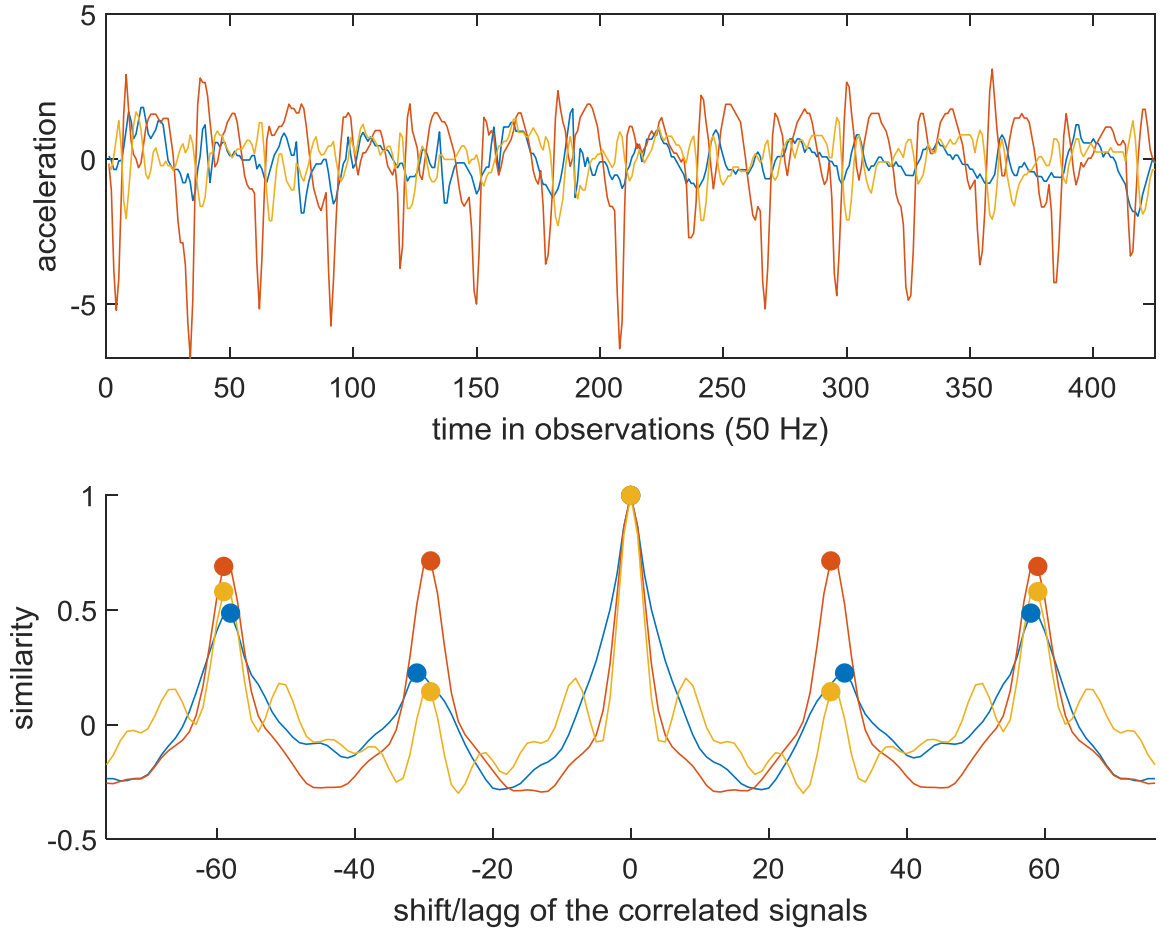


*Figure 36 – **AC** (below) **of the dynamic data** (above) **of the corridor walking, flat heel segments** (loc: 224217:224641). Blue = x-axis, red = y-axis, yellow = z-axis dynamic data and AC correlation. Dots = identified AC peaks.*

Figure 36 provides an example of a gait episode and the correlation of each of its axes. The dominant axis (the y-axis marked as red) in particular shows a high correlation at around 30 observations (0.6 sec). The AC correlates similarity between the data and the same data sifted by x samples. Thereby, the signal is correlated with itself at 0 lag. The spikes of the y-axis present at a lag of around 30 and 60. Thereby the peak at 30 lag is the dominant peak with the following peak at 60 lag being homogenous to the dominant peak. Since the gait pattern peaks are the strongest signal in the data at the y-axis, the AC data will show high similarity at the lags with a multitude of the distance between gait peaks.

Therefore, for AC correlation to be considered walking behaviour we have to first find the dominant peak that is followed by homogenous peaks. The distance between these peaks should be the lag between the area where the signal correlates with itself and the dominant peak. While this is a simple concept in theory that is easy for a human to judge with just one look, programming

the algorithm to do this was quite tricky. We spent many months trying to adjust this algorithm. In general, we prioritised sensitivity over specificity.

Examples of AC correlations of gait and non-gait pattern

The following images can be outputted by the algorithm and are all examples taken from the self-recorded dataset. We will use these images to show how the AC function works first and then give a rundown of the programming steps taken to achieve these results. The upper graph always shows the dynamic data that was labelled as gait within a range around the episode of interest. The range is defined as the length of the episode added to the start and the end of the episode, however, the maximum range extension is set to one minute of data. We then plotted the data analysed within the episode in black on top of the dynamic data. The lower graph displays the corresponding AC of the black data. A lag of 0 is located at 193. The grey box represents the area the dominant peak needs to fall into, in order to be located within the slow and fast walking thresholds. The black lines represent platforms on which the homogenous peaks need to fall to be considered as such.

In general longer episodes of gait resulted in a more reliable clear AC pattern that is easy to identify. A perfect example is Figure 37 above. The dominant peak lies within the thresholds area of the fastest and slowest acceptable walking speed and each flowing homogenous peak hits the platforms nearly in the centre. We call the lag of the dominant peak to the centre of the AC frequency. It describes the frequency at which the individual steps cause the signal to repeat and is equivalent to the Step frequency. Figure 38 provides a good example of gait that is very asymmetric. There the dominant peak and the first homogenous peak show a clear pattern where each step (dominant peak) has a much lower correlation than the gait cycle as a whole (homogenous peak). The episode displays walking behaviour while the hand was located in the trouser pocket. This explains the difference in signal amplitude between the left and right steps.

However, longer walking episodes can also be more susceptible to changes in walking speeds, as shown in Figure 39. Disappointingly, the platform was closely missed by the 2nd homogenous peak. However, while the episode was not identified as gait on the z-axis, the y-axis was also identified as a potential dominant axis and was able to identify most of this episode as gait. It should also be noted that we incorporated different walking speeds as a part of the free walking trial and these gait episodes were all picked up by the AC function.

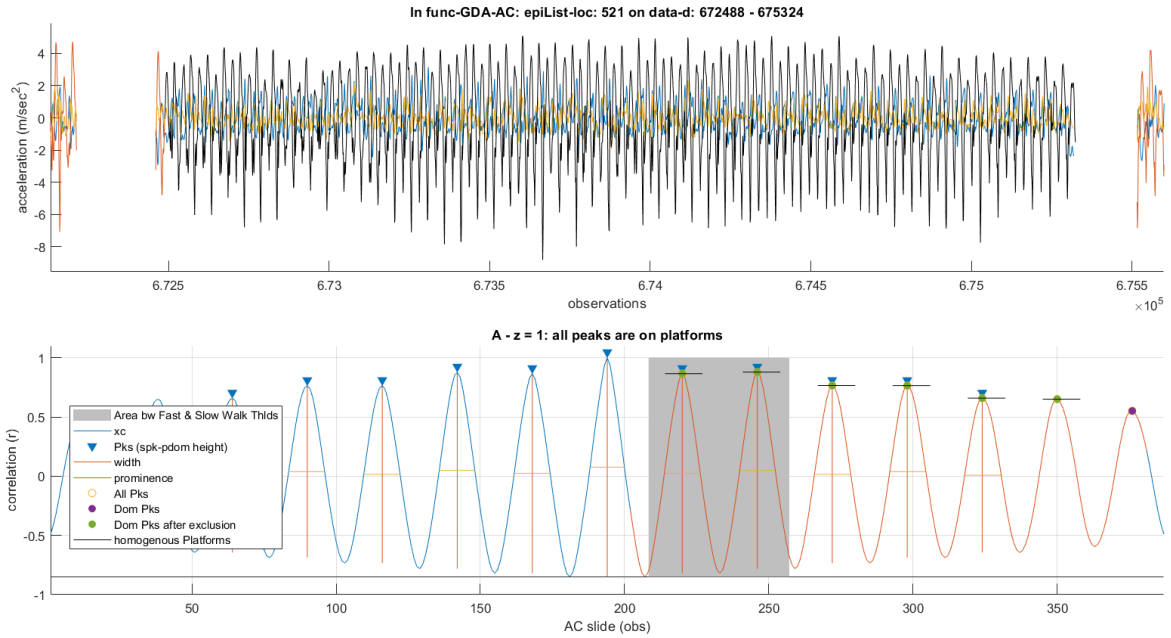


*Figure 37 – **AC example: Perfect Gait – Clean Swing Condition, concentrated on non-interrupted clean gait**. Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

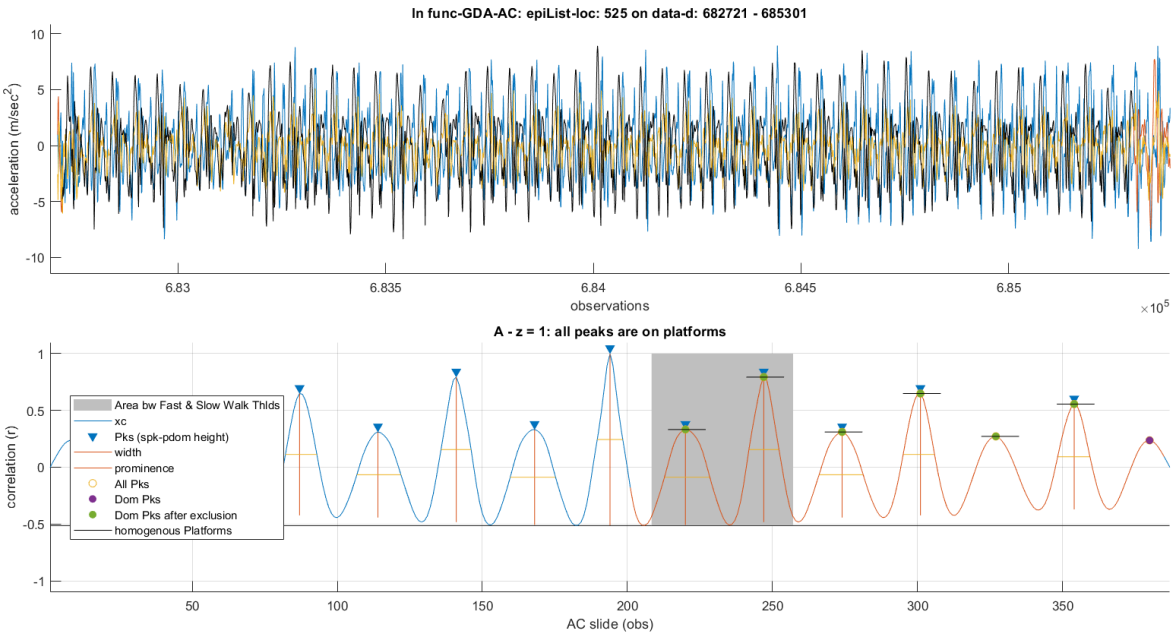


*Figure 38 - **AC example: Good Asymmetric Walk – Hands in Pocket Condition**. Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

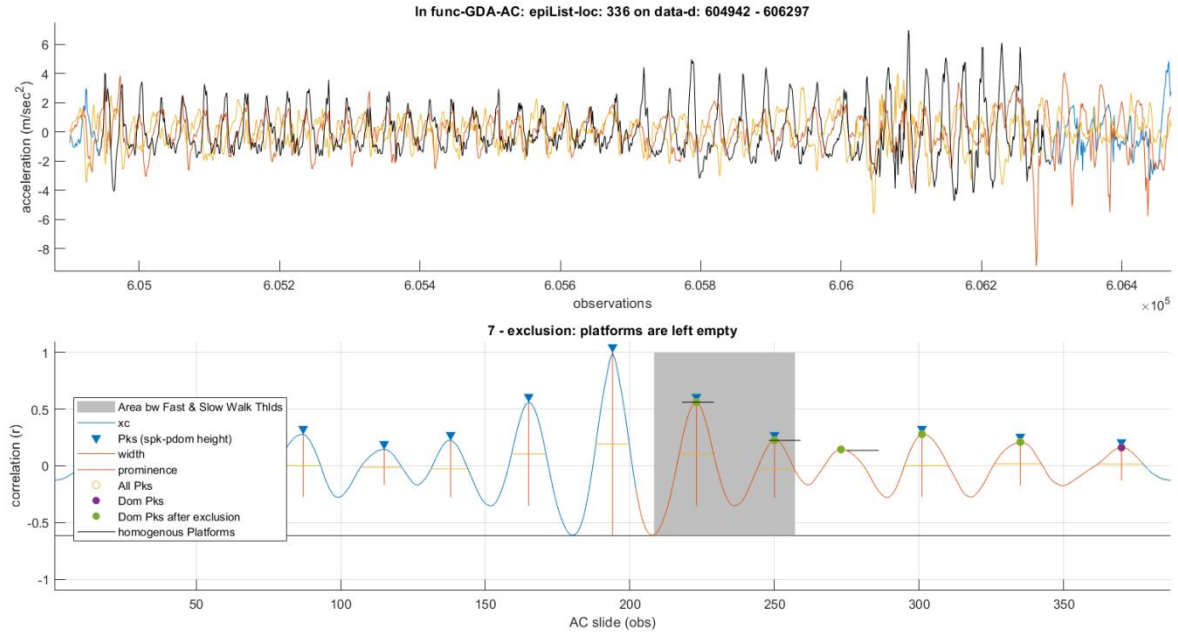


Figure 39 - ***AC example: Wrongly Excluded Walking Behaviour – not labelled as gait*** *(z_Lab = 1), located between 2 triggers. Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

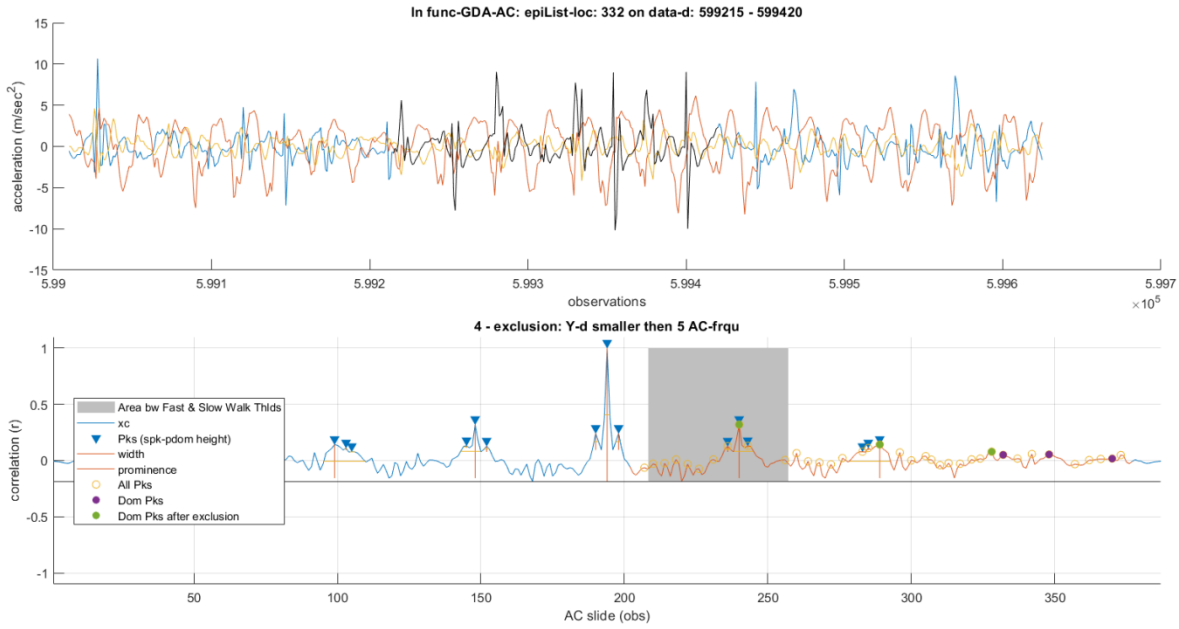


Figure 40 - ***AC example: Wrongly Excluded Data – not labelled as gait*** *(z_Lab = 1), most likely walking downstairs to start the free walking trial. Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

We tried different approaches in separating the data in these cases, however, efforts to do so led to shorter episodes that were too noisy to be identified as gait and caused a lot of non-gait labelled data to be incorrectly labelled as gait. Figure 40 shows an example of how noise can affect short episodes of walking behaviour. The whole length of the episode is 2 sec and 7 steps can be observed. Each step took therefore around 0.29 sec. That usually corresponded to an AC frequency of around 30 observations (obs, 30 obs = 0.6 sec), however, the AC frequency identified was around 50 obs. The noise on the axis combined with the shortness of the episode lead to the episode that looks like gait to be discarded. However, the y-axis dynamic data was much cleaner than the x-axis behaviour, therefore this episode was not lost as potential gait. This is also a reason why we keep more than just the axis with the highest moving STD as the dominant axis since noise can increase axis amplitude significantly.

It should be noted that both of the “missed gait” examples were walking episodes to the start of the free walking trial and therefore did not formally belong to our walking labelled data. We labelled them as gait since they clearly showed walking behaviour, however, they include a lot more noise that was not properly recorded.

The importance of the proper selection of dominant and homogenous peaks can be seen in Figure 41. It represents walking while having a bag hanging from the shoulder with the sensor. Instead of clean up and down spikes, the positive spikes are interrupted with a small negative spike that can cause a slight increase in AC values at the frequency of this positive peak suppression. We can see the small spike at a lag of around 8 obs (marked with a yellow circle), followed by peaks around the dominant and homogenous peaks with a distance of around 7 obs. When we measure the width of the negative noise within the positive spikes of the dynamic data we also observe a distance of around 7 obs.

The challenge that this poses on the peak identification is especially important since we need to make a clear distinction between these noise-affected peaks and noise suppressed peaks that do represent peaks of interest (see Figure 38, asymmetric signal).

On the other hand, there is also data that was labelled non-gait but was picked up as gait by the walking algorithm. Figure 42 is an example of non-walking behaviour that tricks our algorithm into believing that it is gait behaviour. While in this case keeping the “fizzling” would have helped the algorithm in its exclusion of the episode, it would have also thrown out the episode in Figure 43. This episode shows how shorter episodes with some noise in them will show decreased AC amplitude with each homogenous peak. We, therefore, have to stop the analysis of homogenous peaks before the signal “fizzles” out.

Figure 44 displays an example of data that was not labelled gait but has a very similar look to gait behaviour and is also picked up by the AC function as gait. While this case has a negative effect on the accuracy measures of our algorithm, we consider the algorithm's judgment as accurate.

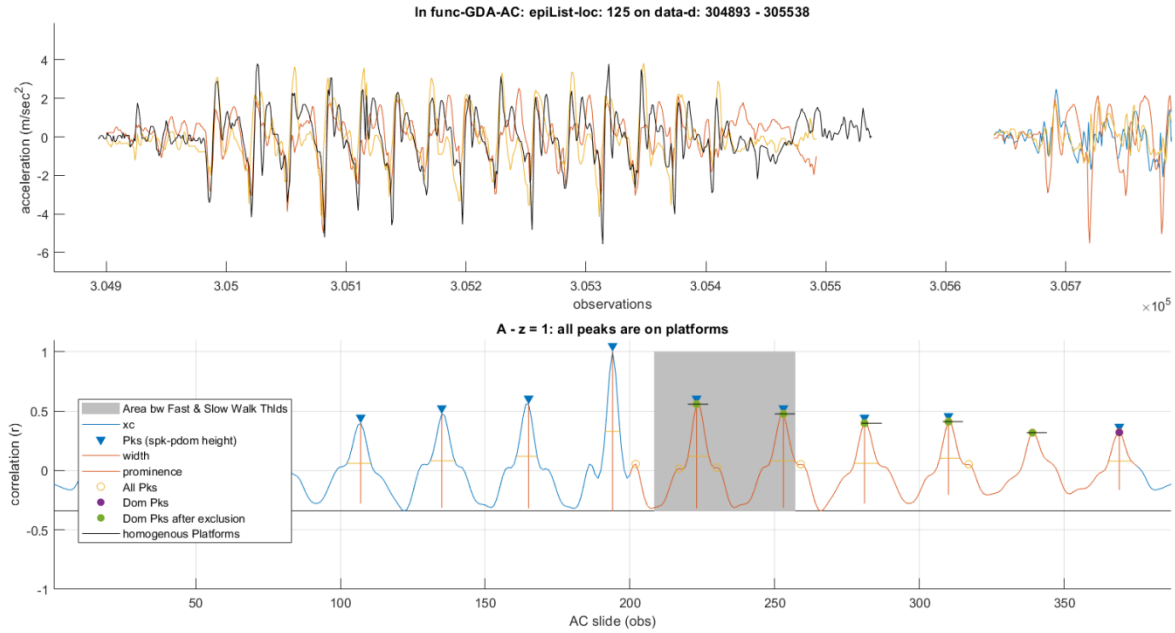


Figure 41 - ***AC example: Sub-Quality Gait – Walking with a Bag over the Shoulder****. Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

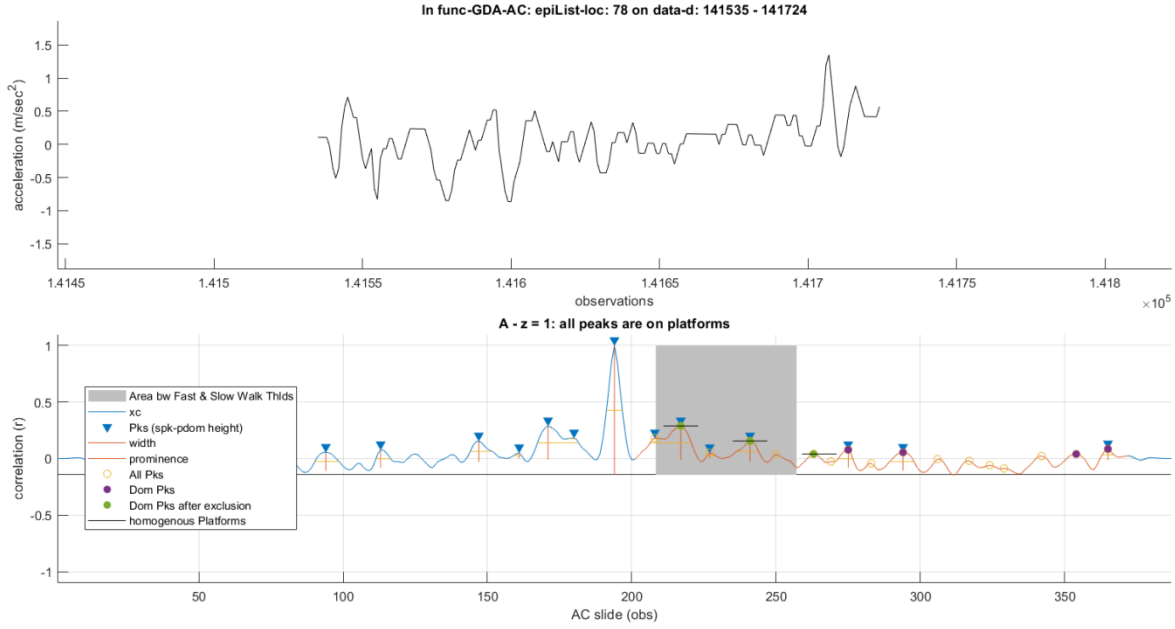


Figure 42 - ***AC example: Misidentified as Gait – At rest during the Staircase Trial****. Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

Lastly, Figure 45 shows some AC Outputs from episodes that were correctly labelled as non-gait. Non-gait AC mostly shows low correlation below 0.5 the further away it goes from correlation with itself at 0 lag. Furthermore the AC tends to fluctuate a lot.

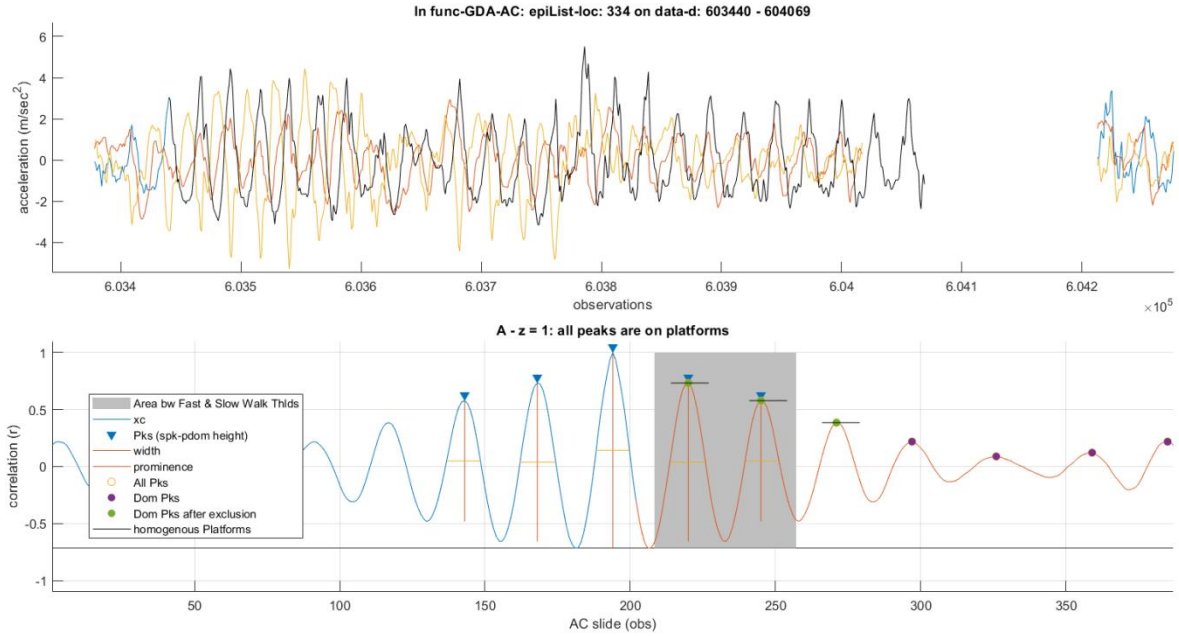


Figure 43 – ***AC example: Accounting for "Fizzling" out AC – Not labelled as gait*** *(z_Lab = 1). Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

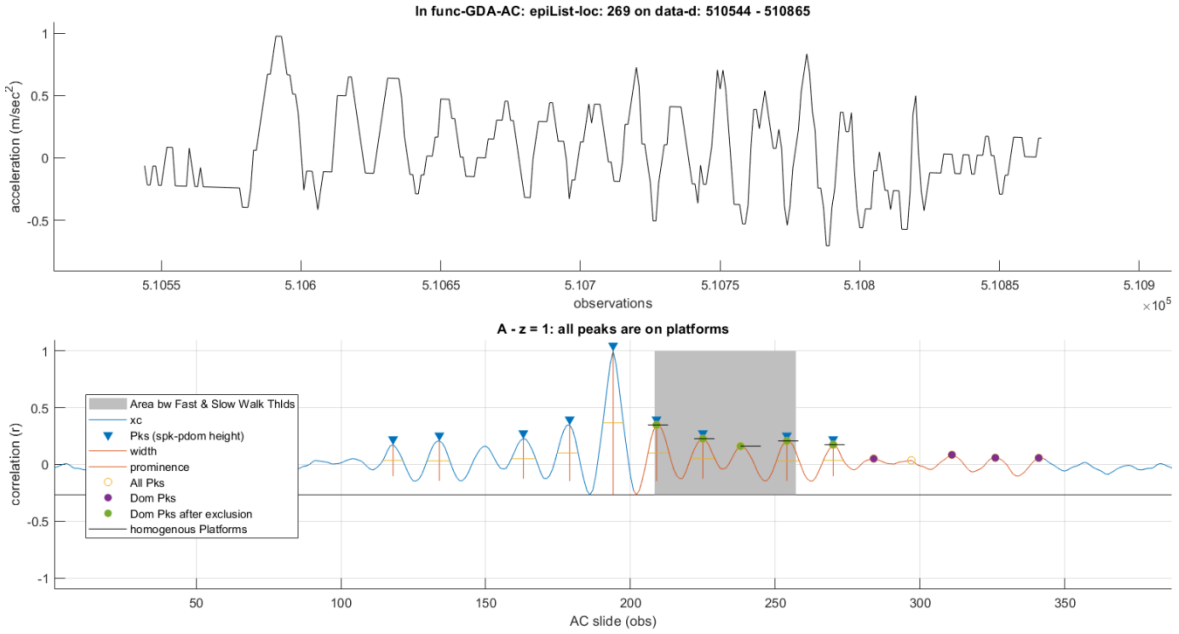


Figure 44 - ***AC example: Behaviour that is non-Gait, but looks like Gait – during the Coffee Trial****. Above: dynamic data, blue, red & yellow = x-, y- & z-axis dynamic data labelled as gait, black = dynamic data of the episode. Below: cross-correlation.*

To summarise, longer episodes of gait data are less susceptible to noise and rapid changes in walking speed. The exclusion criteria will now be explained in more detail while describing the AC function steps.

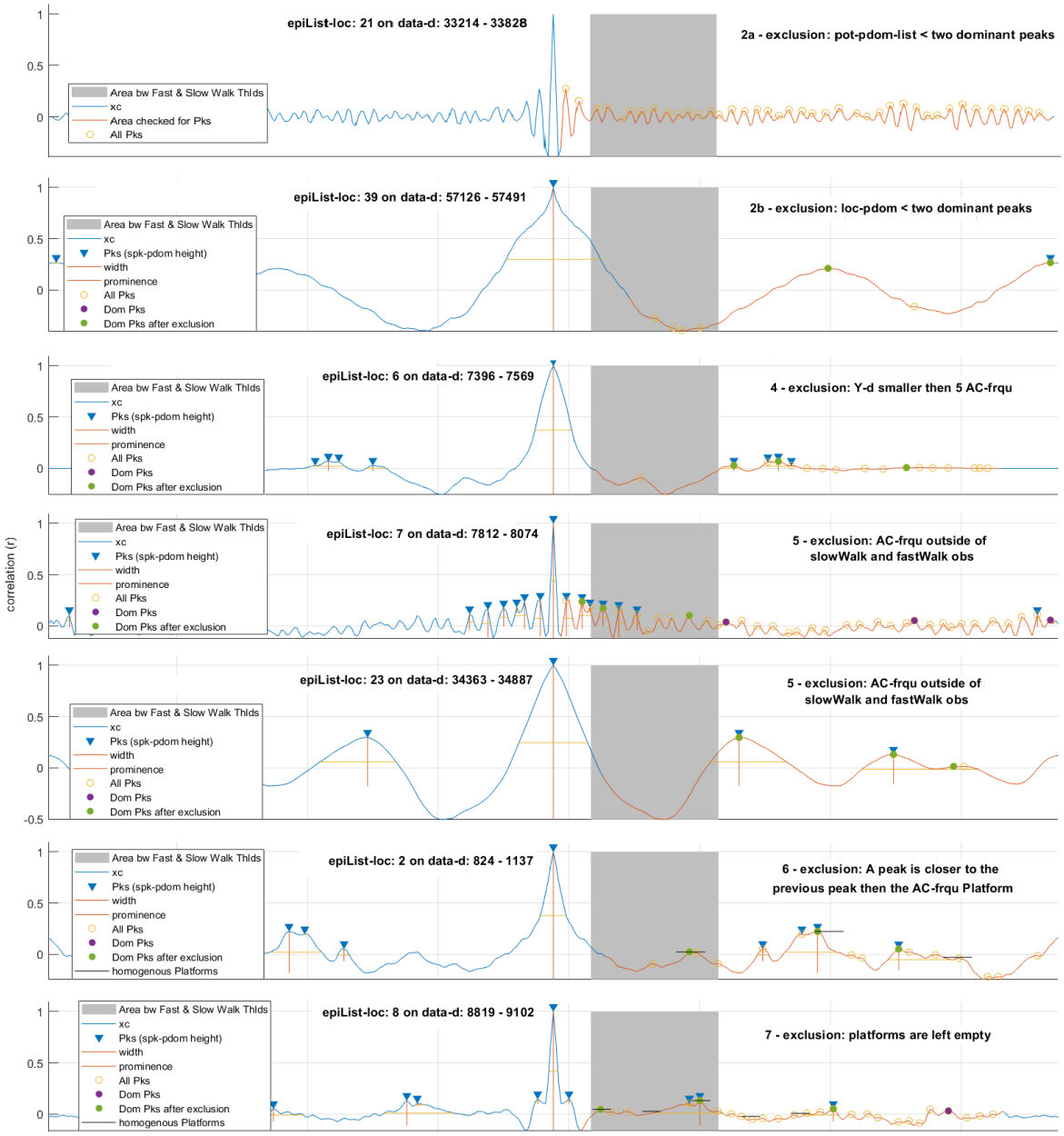


Figure 45 - ***AC example: Various Excluded Episodes****.*

func_GDA_AC

The input required for this function is the epiList_loc output from the “Episodes” function, the dynamic data (data_d), the total dataset length (lgt_TotalData), and the slow and fast walk thresholds (slowWalk_obs and fastWalk_obs). The function outputs the z variables and a list of episodes that were identified as potential gait with further information (AC_epiList). This includes the episode start, end, axis and length already recorded in epiList_loc and supplements them with the identified AC frequency of the potential gait, a quality index of the AC frequency and the index of the corresponding index of this episode on the epiList_loc. Furthermore, we save information about the AC’s quality within the variable Quality_extend. It records the dominant and homogenous peak amplitude, location, width and prominence in .spk_dom.

Algorithm preparation and Structure of i_z

First, we prepare the output Variables for the loop to save processing speed. Matlab works better when the space a variable will need is pre-allocated before running loops. This is generally done using the nan or zeroes function. We also included a variable called ExclusionCriteria that counts the underlying reasons for excluding certain episodes. Within each loop, we first clear the variables bwpks_list, spk_pdom, loc_p0, loc_pdom and sc_dpks_hgt_pct to make sure that peaks from a previous episode are not imposed on the next episode. This is needed since we keep track of our decision to identify an episode as gait or non-gait with the variable i_z. is set to nan at the start of each loop and each step of the AC function checks if i_z is still equal to nan before continuing. If an episode is therefore excluded early on, it is important the variables listed above are cleared, since they would not be overwritten by the code. If a step identifies the episode as non-gait i_z is set to 0. At last, if the last exclusion criteria is passed i_z is set to 1. Furthermore, we spell out the Exclusion Criteria for plotting purposes in plot_title and increase the ExclusionCriteria variable by one. This process is done for each Exclusion Step.

Autocorrelation

We load in each episode's dynamic data (as Y_d) and run a normalised cross-correlation using a window size of three slow steps (xc = xcorr(Y_d(1,:), ceil(slowWalk_obs*stepfactor),'coeff')). The stepfactor is a variable defining how wide of a range we want the AC to be conducted on. We decided upon a factor of 3 that enables us to find at least 3 dominant/homogenous peaks under the slowest walking behaviour. In order to focus on the AC data of interest, we select the xc data from the plotcenter to the end of the plot, thereby setting the lag at the place where the data correlates itself to 0. Any future reference to the location of a peak will therefore be synonymous with the AC frequency in obs. A different approach that we contemplated was to consider the autocorrelation of all three axes together, like in Figure 36. However, we did not manage to obtain good results this way, and this approach became too convoluted.

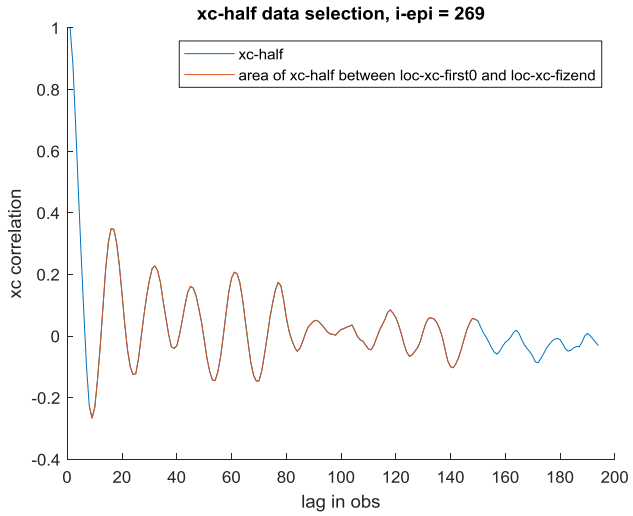


Figure 46 – ***AC method: Area of potential peak selection****.*

Exclusion Criteria 1: XC does not reach 0

A dominant peak needs to be proceeded by a valley that needs to be in the negative correlation range since the area before the fast walking threshold would correlate the increasing and decreasing accelerations with one another. We, therefore, identify the area before `xc` hits zero (`loc_xc_first0`) and set the data preceding this point to `nan` to exclude it from analysis (see Figure 46). If the correlation never reaches 0, then our first exclusion criteria is triggered. This did not occur in the SRDS, but did happen in the KOALAP dataset. This process also prevents pronounced early peaks from being identified as the dominant peak.

Identify all peaks and their properties

After this, we identify the endpoint of the data of interest as the last time the xc graph reaches the mean of xc (`loc_xc_fizend`, see Figure 46) and create a vector that excludes data beyond this point named `xc_half2`.

We then use `findpeaks` to identify the amplitude, location, width and prominence of each potential peak in `xc_half`. From Figure 37 to Figure 45 these peaks are marked with yellow cycles. We use `findpeaks` instead of `peakseek`, since it is able to return these measures. Furthermore, `findpeaks` is able to perform quicker when no criteria for the peaks (such as minimum peak distance) are given. We tried using `findpeaks` directly to identify our dominant and homogenous peaks by using thresholds such as a min distance of `fastWalk_obs`, a maximum width of `slowWalk_obs`, a minimum peak height and prominence of the maximum between `xc` within the fast and slow walking thresholds. However, using these qualifiers lead to a peak selection that forced our target pattern onto the data and therefore lead to a lot of false positives.

We, therefore, use `findpeaks` to identify properties of interest of all peaks in the data and then identify potential peaks with a different method described below. We can then use the identified location of the selected peaks to obtain their properties from the `findpeaks` output.

Identify Potential dominant/homogenous peaks

Next, we identify potential dominant and homogeneous peaks. We identify ranges that consecutive homogenous peaks would potentially be located within. A dominant peak has to have a large prominence. The ideal dominant peak (see Figure 37) would display a negative correlation valley, followed by a positive hill and another negative valley. Identifying where `xc_half` crosses 0 would therefore give a good indicator for potential peaks and valleys. Since the negative correlation peaks are not as big as the positive correlation peaks, the width at the point where the correlation hill passes 0 should be around half the length of the AC frequency. For episode 521 (see Figure 37), which has a very clean and steep valley and hill pattern, the area size identified in `pot_pdom_list` is 13 obs, while the AC frequency identified is 27 obs. Since more noisy data tends to suppress the positive correlations and creates larger widths, a threshold of half the fastest Walking speed should be sufficient, however in accordance with our guidelines for the code we halved the threshold to 1/4th of the `fastWalking_obs`.

It would have been easier to set the threshold as ¼ of the location of the maximum peak of the data or to discard any peaks that have a high below 0.5 as dominant peaks. However, the gait cycle pattern in Figure 38 would not satisfy this criterion. Figure 47 displays examples of walking data with a large noise effect that suppresses the positive acceleration. Episode 355 has very nice regular peaks and valleys around the magnitude of -0.2 and 0.2. This is caused by the noise surrounding the episodes, which is most likely due to hand motions preparing for the free walking trial. Episode 637 on the other hand was too noisy

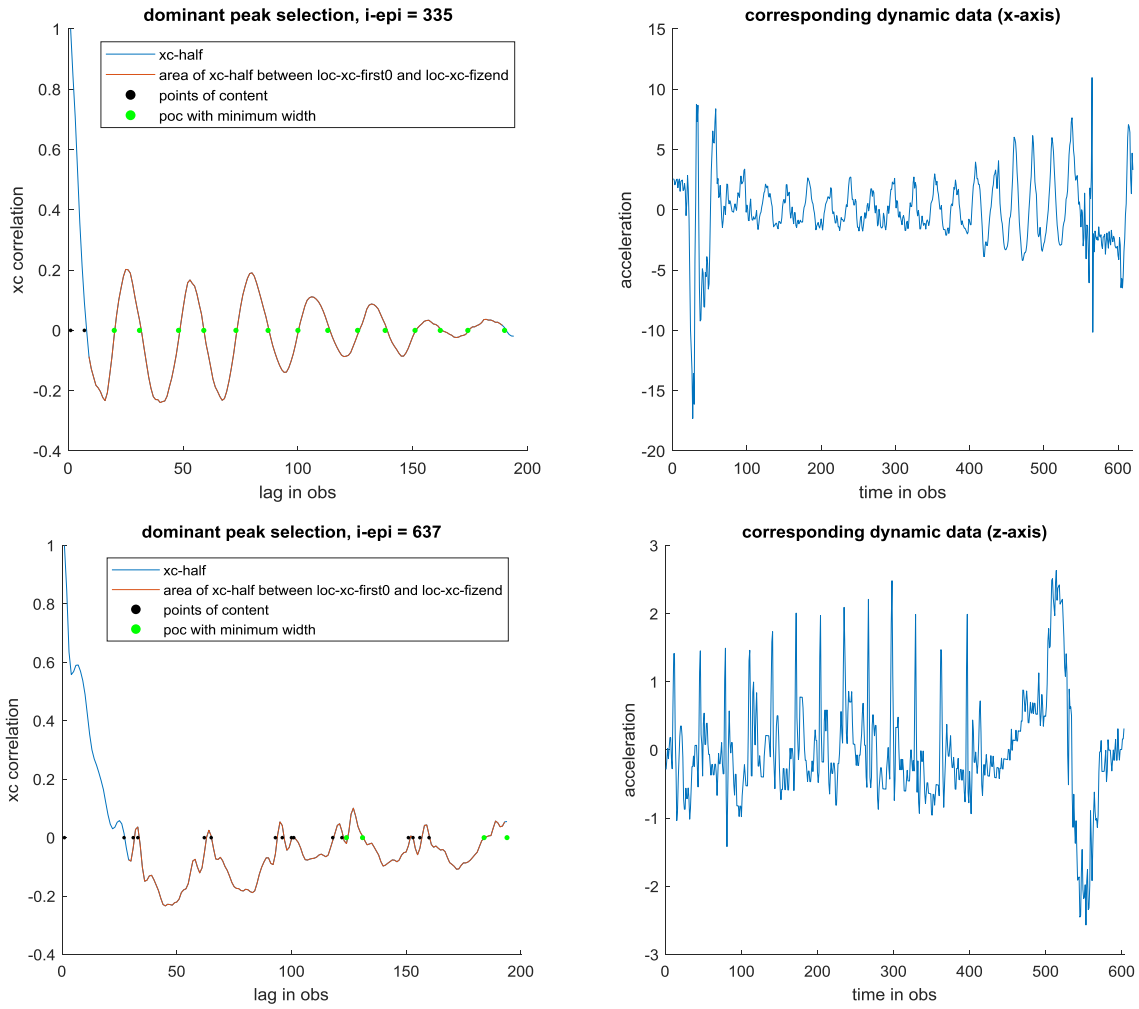


Figure 47 – ***AC example: Comparison between two axes on the same section of data*** *– the z-axis was discarded, but the x-axis was correctly labelled as gait.*

In the code, we identify each instance where xc crosses 0 and call these instances `pointsofcontent` (see Figure 47 black dots). We then look at the data between each point of content and see if the minimum distance between the points is at least 1/4 of the length of `fastWalk_obs`. Then we check if the area between the points is a valley or peak and exclude any points smaller than `loc_xc_first0`. The resulting areas (`pot_pdom_list`, see Figure 47 green dots) are our potential dominant and homogenous peaks. If less than 2 areas are identified the episode is labelled non-gait (Exclusion Criteria 2a).

Next, we select the dominant peak by looking at the areas identified by `pot_pdom_list` one after another. We receive the output `loc_pdom`, which is the location of the maximum peak in each area. As a failsafe, if different peaks were identified than with the initial `findpeaks`, they were discarded. If not at least two peaks were found the episode was again labelled non-gait (Exclusion Criteria 2b). From Figure 37 to Figure 45 `loc_pdom` is marked as purple dots.

Select Peaks

We use the identified location of the peaks from `loc_pdom` to create a matrix (`spk_pdom`) with the properties identified by `findpeaks`. We then look at the maximum height and prominence of the peaks and take one-third of their value as a threshold to identify the point where the peaks

“fizzle out”. For example in Figure 48 (see also Figure 43) the dominant peak has the overall maximum height and prominence of 0.73 and 1.39 respectively. We, therefore, look at the last peak that reaches the threshold high of 0.24 and remove any potential peaks that come afterwards. In this case, the 3rd peak was the last to reach the threshold and the 4th to 7th peaks were removed. However, we included a condition that not more than 3 peaks can be removed. This process is repeated for prominence as well.

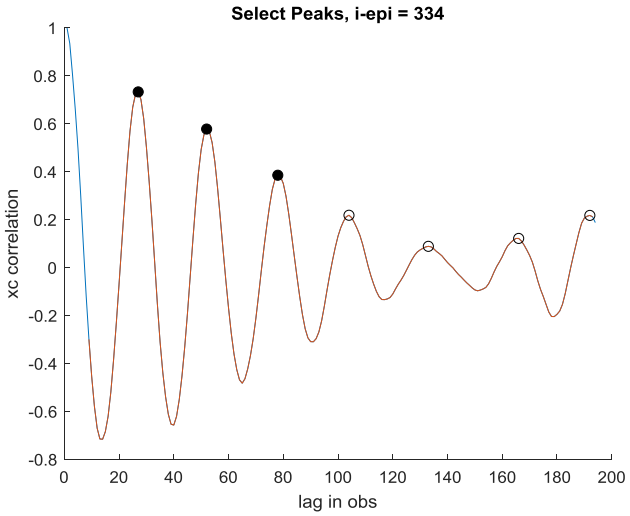


*Figure 48 – **AC method: AC peak selection.** Blue = xc_half, red = areas of xc_half between loc_xc_first0 and loc_xc_fizend, filled cycles = peaks accepted, empty cycles = peaks rejected.*

Furthermore, we checked that there were no outlying prominence peaks in the data anymore. We took the mean of the prominence of the `spk_pdom` spikes as the variable `i_mean` and compared it to the 1st centile of all peaks that were above the `i_mean` threshold. If the newly identified threshold `thldmean` was closer to 0 than the `i_mean` threshold, then all peaks that did not reach the new threshold were removed from the `spk_pdom` matrix.

We now have a matrix `spk_pdom` where `spk_pdom(:,1)` is the dominant peak and `spk_pdom(:,2:end)` contains the homogenous peaks. Again, if less than 2 peaks are left the episode is labelled non-gait (Exclusion Criteria 2c).

Identify Frequency and a measure of Quality of the Episode

We record the location of the dominant peak as the AC frequency (`AC_frqu`). This measure is the expected step frequency of the episode. Furthermore, we record a measure that represents the quality of the episode. We, therefore, identify the negative peak amplitude of the valley preceding the dominant peak (`xc_lower_thld`) and calculate its distance from a perfect correlation at lag 0 (`xc_ysz`). For example, the `xc_ysz` for the ideal gait episode 521 was 1.85. We then calculated the distance between the height of each peak of `spk_pdom` and the `xc_lower_thld` and divided it by `xc_ysz` to obtain the percentage of there the peaks lie between the lowest valley and a perfect correlation. We recorded this value as `sc_dpks_hgt_pct` and will add it to the `Quality_extend` variable later on.

Exclude Episodes according to AC frequency

Next, we apply two Exclusion Criteria. First, the length of the episode has to fit at least 5 steps. Therefore episodes that are shorter than the `AC_frqu*5` are labelled non-gait (Exclusion Criteria 4). Then we check if `AC_frqu` lies within the fast and slow Walking thresholds. If `AC_frqu < fastWalk_obs` or `AC_frqu > slowWalk_obs` then we label the episode non-gait (Exclusion Criteria 5).

Platforms

Next, we investigate whenever the homogenous peaks are homogenous by creating "Platforms" on which these peaks should fall. To do this, we extract the width of the dominant peak as the basis of our platform width (`bwpks_wid`). However, if the identified platform width is larger than half of the identified AC frequency then half of the AC frequency is taken. Then we identify the range around the dominant peak (`accprange`) by centring the platform on top of it. A matrix `bwpks_list` is created that has the location of the peaks from `spk_pdom` in the first row, the start of the platform on the second row, the end of the platform in the third row and an indicator of if the peak lies on the platform or not. We create this matrix by applying a loop that runs as long as there are still peaks that can be assigned a platform or an exclusion criteria is hit. We create platforms by taking the location of the peak on the previous platform and move the start and end of the next platform by the AC frequency plus and minus half the platform width. Then we check if the peak lies on the platform. If the peak of the current loop is located in front of the platform we declare the episode as non-gait (Exclusion Criteria 6).

If the peak is located after the platform, it could be that a peak was wrongly excluded due to a low prominence, which can be caused by excessive noise. To counter this, we insert a new column after the current peak into `bwpks_list`, move the peak over to the new column, remove the peak in the current column and identify the start and endpoint of the platform in the new column by adding the AC frequency to the platform start and end of the old column. The next loop now evaluates the peak again for the new platform. If the peak misses two platforms in a row, the episode is declared non-gait (Exclusion Criteria 7).

Finally, we check if any peaks were not matched to a platform by looking at the 4th row. If any row was marked with a 999 we identify that a platform or peak stayed unmatched and declare the episode as non-gait (Exclusion Criteria 7). Any episodes that were not declared non-gait at this point are now identified as gait episodes.

5.3.9. Reformatting the Output, Clean up and Figures

In comparison to the other functions where the code for figures is added at the end of the function, in the AC function a figure can be created for each potential walking episode. To reduce the amount of data that would have had to be saved in memory for the creation of the figure, the figure creation is placed at the end of the episode loop. If figures were created we saved the ExclusionCriteria Variable with labels in a Matlab data document called "`Results_AC_EC_' IMG_name '.mat`".

If the episode was identified as gait we record this in the z variables. Furthermore, we extracted the `epiList_loc` for the episodes identified as gait and added the AC frequency and the rough measure of Quality to the matrix, creating the `AC_epiList` output variable.

5.4. Algorithm Performance

5.4.1. Data Reduction of the Individual Algorithm Steps

To examine the validity and importance of each step of the analysis we will summarise how much of the SRDS was excluded as non-walking behaviour in each step of the algorithm.

%	KOALAP Algorithm		UK BioBank Algorithm	
	original dataset	randomised dataset	original dataset	randomised dataset
DomAxis	100.00 %	100.00 %	100.00 %	100.00 %
mSTD	81.10 %	81.98 %	67.58 %	70.68 %
Episodes	57.55 %	56.98 %	49.68 %	49.26 %
AC	28.11 %	29.36 %	28.55 %	29.31 %
Data_Lab	29.77 %			

Table 7 - Percentage of Dataset identified as Potential Gait according to 1D z-vectors

The total length of the data is 771,070 observations (obs) (`lgt_TotalData`), of which 29.77% was labelled as gait data (`data_Lab`). Table 7 depicts the percentage of the data identified as potential gait by each function's 1D z variable output and Table 8 displays the same for the 3D z variables.

%	KOALAP Algorithm			UK BioBank Algorithm		
	x	y	z	x	y	z
DomAxis	49.87 %	48.25 %	48.29 %	51.91 %	48.32 %	46.79 %
mSTD	37.81 %	40.24 %	35.74 %	30.66 %	34.73 %	26.77 %
Episodes	19.95 %	26.03 %	19.24 %	17.95 %	24.92 %	14.20 %
AC	6.95 %	23.37 %	2.77 %	6.80 %	23.32 %	3.26 %

Table 8 - Percentage of the original Datasets identified as potential Gait according to 3D z variables broken down for each axis.

The **moving STD threshold** (`z_mSTD`) identified 515,334 obs as potential walking behaviour and reduced the data therefore by 33.17%. After applying the **dominant axis threshold** (`z_domAxis`) 452,073 obs were left. This is a reduction of 12.28% of the data. Since the main purpose of the dominant axis function is to examine the three axes independently from one another instead of looking at the data magnitude, we will now look at the 3 axes as well. The STD threshold identified 1,546,002 obs on the three axes as potential gait, while the dominant Axis approach (`z_domAxis_3D`) only identified 562,008. This is a reduction of 63.65%. The **identification of episodes** further eliminated 44.55 % of the data identified by the dominant axis threshold. Lastly, the **AC** reduced the amount of identified episode data by 50.74%. It is therefore evident that each step in the algorithm is meaningful and necessary for the elimination of non-gait behaviour.

5.4.2. Accuracy of the Individual Algorithm Steps

Figure 11 displays the accuracy between the Algorithms Steps and the hand labelled data of the SRDS. Overall the Performance of the algorithm does not misidentify a significant portion of the dataset. We managed to achieve very high accuracy for labelling walking behaviour correctly (97.57 %) and for labelling non-walking behaviour correctly (96.83 %). That means that in total only 2.80 % of the data was misclassified by the GDA, with 97.20 % being correctly classified. The moving STD, Episode and AC functions each accounted for a substantial exclusion of non-gait data. The Episode function thereby relies on the inputs of the DomAxis function for operation.

Code	**Percentage of data identified as Gait**	**Overlap** (z_Lab = 2 && z_x = 1) **Walk correct**	**Overlap** (z_Lab < 2 && z_x = 0) **Rest Correct**	**Overlap** (z_Lab = 2 && z_x = 0) **Rest Wrong**	**Overlap** (z_Lab < 2 && z_x = 1) **Walk Wrong**
DataLoad	Hand Labels 25.52 %				
DomAxis	92.89 % on 1D 41.69 % on 3D	99.41 %	9.33 %	0.58 %	90.66 %
mSTD	77.30 % on 1D 32.89 % on 3D	99.40 %	30.27 %	0.59 %	69.72 %
Episode	54.54 % on 1D 20.75 % on 3D	98.83 %	60.64 %	1.16 %	39.35 %
AC	27.26 % on 1D 10.71 % on 3D	97.57 %	96.83 %	2.42 %	3.16 %

Table 9 - Resulting Accuracy of the GDA

Since the STD Thresholds (`z_mSTD`) main purpose is to cut down on data that needs to be analysed, the most important factor regarding accuracy is that no gait data gets excluded. Therefore, we need to look at the amount of data incorrectly identifies as non-walking while it was labelled as walking. Only 9.2 sec of 257 min of data were incorrectly identified as rest according to the hand labels. However, looking at the data in context, this seems to be mostly the case due to the nature of the data labelling. The start and end of the gait periods were labelled with an inaccuracy of up 1 sec (50 obs). Especially within the corridor trail setting the data around the start and end points of a walking period is better defined by the STD Threshold algorithm than by the hand labelling. This is especially true for the “Water Cup” condition (see Figure 49).

The Dominant Axis approach causes the cut around label onsets to be defined even more precisely. It cut out more imprecise hand labelling from nearly every walking on- and offset, in one case even removing only a single sample. It correctly cut out spikes caused by reading of the time and step count from the watch before the walking started that sometimes occurred too quickly in the walking episode to be removed efficiently from the data while labelling (see Figure 50). However, it sometimes did cut out genuine steps, but only at the start or end of the walking episode. We assume that the above-described gait episode-preceding wrist orientation change might be the cause.

The Episode function wrongly excluded 2.8 sec of addition data (Figure 51). Most of the data excluded consisted of either the start and end section of walking episodes or very noisy data.

Sadly, the AC function did exclude some data that was labelled gait and also clearly looked like walking behaviours (see Figure 52). In total 35.56 sec of data were lost. However, this is only a

small percentage of the total of 65.60 min for gait data in the SRDS and only a little more than half a min of data was misclassified as non-gait while gait was present in total.

Some data that labelled non-gait was not successfully excluded from the GDA, in total 7.6 min for non-gait behaviours were misclassified (see Figure 53). Often these areas that surround actual gait behaviour, and are likely due to start- and endpoints of episodes not being cut off successfully by the dominant axis approach. Most of the data was part of the coffee trial and upon closer examination of the data, this does show some periodicity characteristics. Sometimes the misidentified data clearly looked like gait behaviour and the algorithm was justified in labelling it as such. Other identified data consisted of very low amplitude behaviour whose periodicity falls into our walking speed thresholds. This misidentification of non-walking behaviour as gait is most likely due to some too low thresholds being selected for the data.

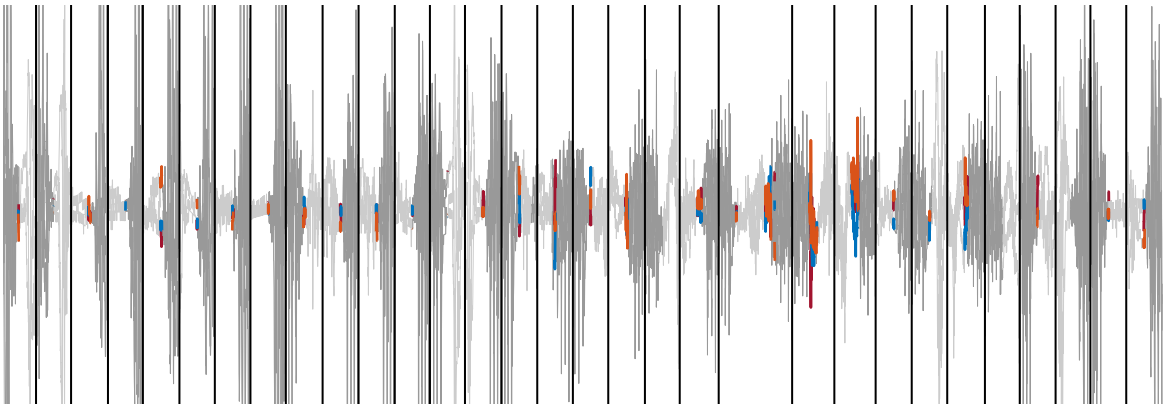


*Figure 49 – **All data the STD-Threshold misidentified as non-walking**. Coloured = misidentified data (data_Lab = 0 & z = 1), light grey = data labelled non-gait (data_Lab = 0); dark grey = data labelled gait (data_Lab = 2), black vertical lines = breaks between data instances.*

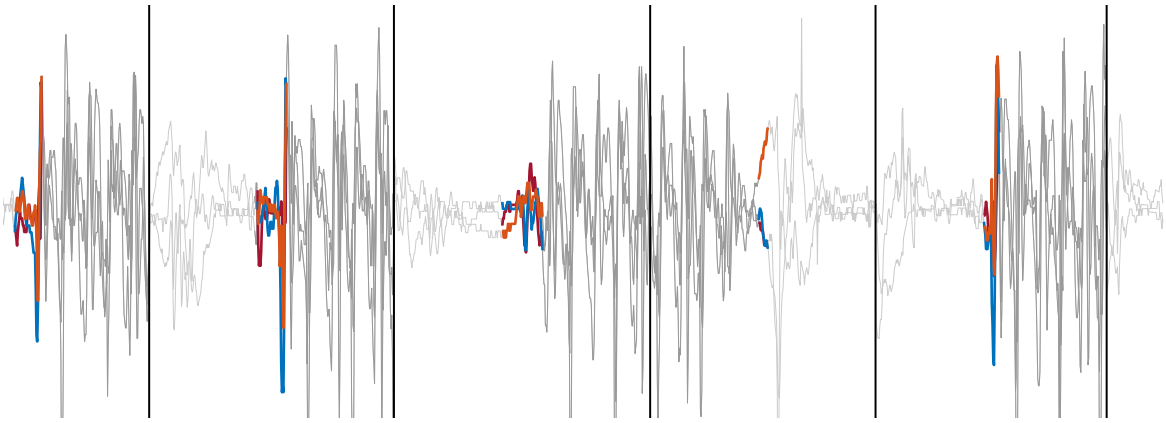


*Figure 50 **– Data Sections the Dominant Axis Approach misidentified as non-walking**. Coloured = misidentified data (data_Lab = 0 & z = 1), light grey = data labelled non-gait (data_Lab = 0); dark grey = data labelled gait (data_Lab = 2), black vertical lines = breaks between data instances.*

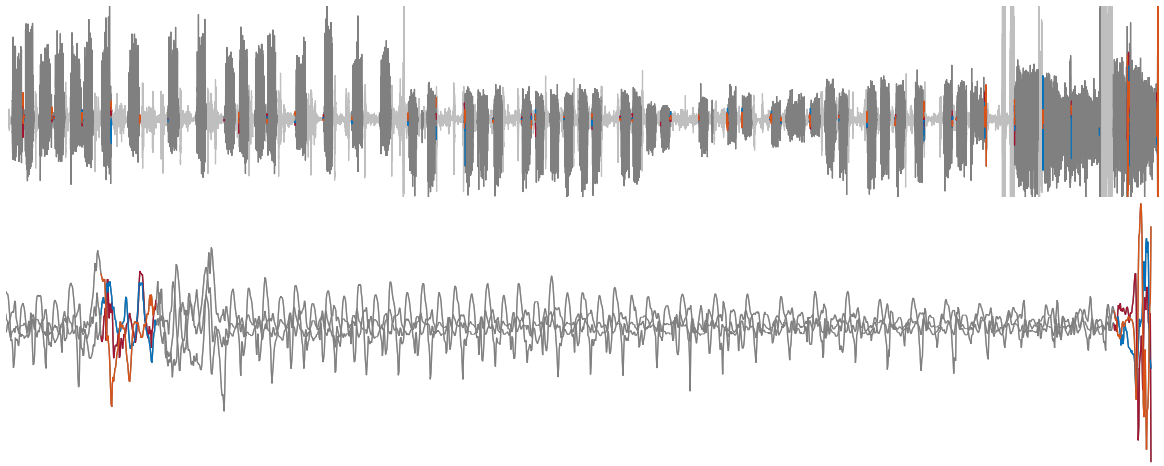


*Figure 51 – **All Data Sections the Episode function misidentified as non-walking.** Zooming into the last two occurrences (below). Coloured = misidentified data (data_Lab = 0 & z = 1), light grey = data labelled non-gait (data_Lab = 0); dark grey = data labelled gait (data_Lab = 2), black vertical lines = breaks between data instances.*

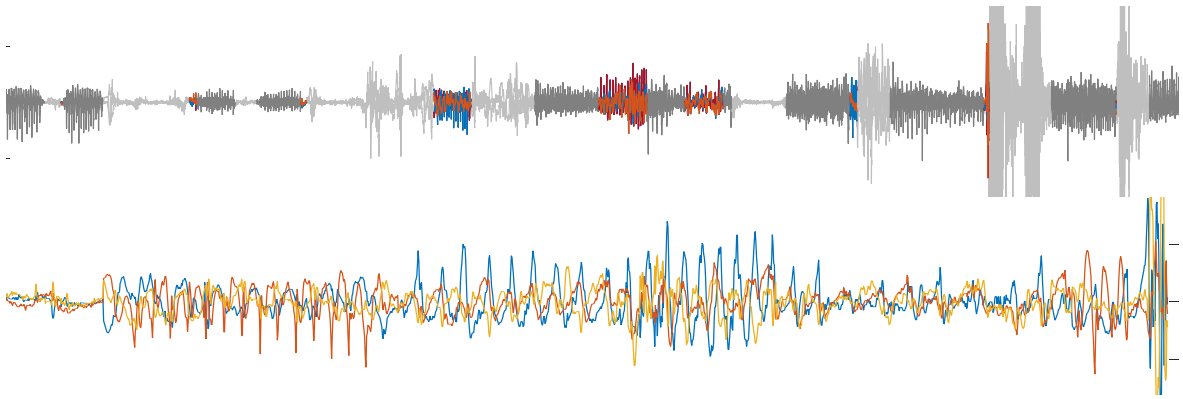


*Figure 52 - **All Data Sections the AC function misidentified as non-walking**. One episode with zoom. Coloured = misidentified data (data_Lab = 0 & z = 1), light grey = data labelled non-gait (data_Lab = 0); dark grey = data labelled gait (data_Lab = 2), black vertical lines = breaks between data instances.*

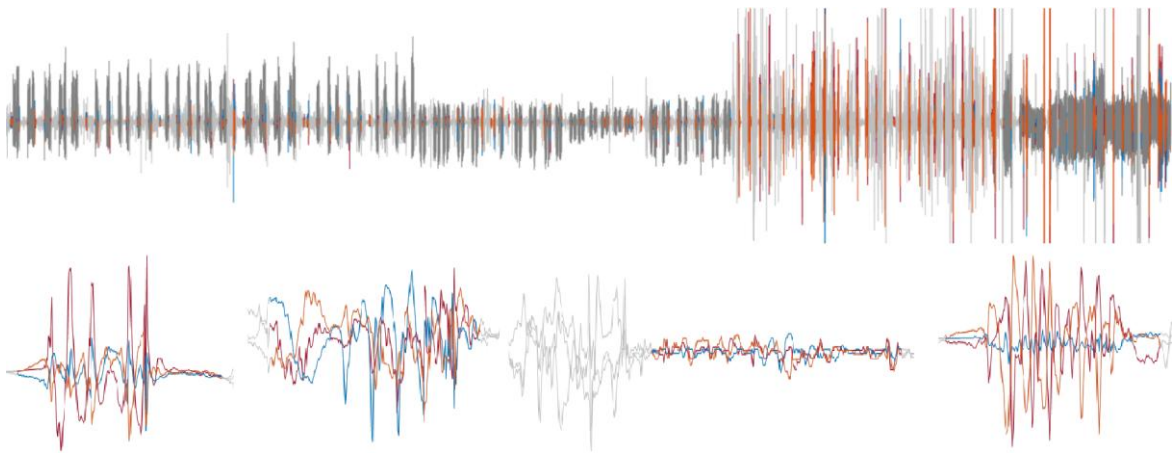


*Figure 53 - **All Data Sections the GDA misidentified as walking**. Some sections are zoomed in. Coloured = misidentified data (data_Lab = 0 & z = 1), light grey = data labelled non-gait (data_Lab = 0); dark grey = data labelled gait (data_Lab = 2), black vertical lines = breaks between data instances.*

5.4.3. Processing Time of the Individual Algorithm Steps

The dataset is 771,070 samples long or 4h 17 min 1.4 sec (4.28372h). The following measures of processing time for the steps within the algorithm will be given in sec per h of data (sec/h). The KOALAP data alone consists of ~ 2.18 Years' worth of data (~19,100 h); therefore one sec of processing time per h of data would need ~ 5.31h to be conducted. The processing time is measured on a common personal laptop. While the ICSF server will have fast processing times and will be able to handle processing multiple chunks of data at once, calculating the processing time will be useful to evaluate the processing speed of the algorithm during development. This should be helpful in estimating a factor of time each step will take that can later be used to calculate how long the full data processing will take. All processing times will be conducted on the self-recorded dataset. Times will be reported in sec/h of data. Processing times were derived using the functions `TIC` and `TOC`.

The STD Thresholding step took 0.20 sec (199.82 msec) to run on the self-recorded Dataset. This results in a processing time of 0.05 sec/h. The `TIC` and `TOC` were set before and after the function func`_GDA_mSTD`. Identifying potential dominant axes took 0.96 sec (958.16 msec). The processing time is 0.22 sec/h. Identifying episodes from the dominant axis and removing some of them took 3.58 sec (3576.71 msec), resulting in a processing time of 0.84 sec/h. The most time-intensive component of the code is the detection of peaks.

Sadly, while interpreting the data we had to discover that we made some mistakes extracting the runtimes of the individual function steps. While we wanted to compare the length of the functions between the self-recorded data and the KOALAP dataset, we will only be able to record the processing length of the files for the SRDS. We decided to record the runtime on the SRDS outside of the iCSF on the GDA. Both the version without pre-processing (KOALAP Algorithm) and with pre-processing (UK BioBank Algorithm) are reported in Table 10. Furthermore, we also created a randomised version of the SRDS. Processing times of the KOALAP and UK BioBank datasets were quicker per hour of data than the iCSF and included more resting behaviours, reducing the processing time per hour of data further. But, considering that future use of the algorithm is unlikely to have access to the iCSF, we decided that analysing the runtimes of the SRDS (with about equal amounts of non-walking behaviours and walking behaviours) on a computer running windows 10 are sufficient to illustrate processing requirements of the steps of the code. Actual processing times will vary widely between datasets and processing machines and information gained from reporting the KOALAP and UK BioBank processing times will only serve to illustrate the time requirements of analysing large datasets all at once, which we will cover in the KOALAP wrangling chapters.

The runtime of the KOALAP algorithm on the ordered dataset was 17.69 sec for 4.28 h worth of data. These results in a processing speed of 4.13 sec/h for data sampled at 50 Hz. The randomised dataset had a slightly slower processing time of 19.81 sec, translating to 4.62 sec/h. The UK BioBank algorithm on the ordered dataset took 18.00 sec (4.20 sec/h) and on the randomised dataset 18.05 sec (4.21 sec/h). The processing length for the individual algorithm steps for the datasets is reported in Table 10 below.

Processing time in sec	KOALAP Algorithm		UK BioBank Algorithm	
	original dataset	randomised dataset	original dataset	randomised dataset
Start to End	15.07	16.31	18.15	17.80
Extract Data	0.12	0.10	0.33	0.22
DomAxis	0.83	0.76	0.84	0.76
mSTD	0.45	0.19	0.64	0.30
Episodes	0.54	0.29	0.40	0.27
AC	4.01	4.61	3.48	3.44
Saving Data	9.11	10.36	12.46	12.81

Table 10 - Processing times of the individual steps on the SRDS in Sec

5.5. Summary

Using a novel approach to identifying a dominant axis on which gait behaviour lies in combination with an autocorrelation enabled us to create a gait detection algorithm that is both fast and accurate. We do have to concede that the SRDS that we tested our algorithm on was limited in its number of participants and number of behaviours. Further testing of the algorithm on labelled gait behaviour would be advisable.

We did test some alternative gait detection methods. We tried using an NASC to exclude resting data, however, using a simple moving STD-threshold proved to be faster and we did not need the marginally increased accuracy of the NASC. For judging the periodicity of the data, we also attempted a p-Welch Method although similar reasoning led to choosing an autocorrelation instead. One of the more troublesome decisions we had to make was the classification of whenever a gait behaviour was interrupted or sustained. Noise can lead to an interruption of a gait episode and we decided to separate episodes if the break between episodes was larger than one sec. Larger breaks would have made future step parameter extraction more difficult to navigate. Future versions of this algorithm could take this problem into account.

The autocorrelation was the most difficult step to program. The AC function has some limitations when gait behaviour varies within the same episode. For example, when a person varies in their speed within the same episode it will interfere with the correlation. Another example would be the handling of objects or holding onto a body part, as in the phone or pocket condition, that causes slight wrist rotations which will have an impact on the acceleration magnitude. Setting a maximum episode length to run the AC on should be considered although this would mean a trade-off in processing speed. In general, setting clear boundaries is necessary when applying an autocorrelation approach. Noise can hinder the ability to correctly identify the data when half of the data being correlated is walking behaviour and half is not.

One way we had to change the code was by reducing the amount of data that gets saved. In the course of the algorithm development, we saved the output data from each Matlab function in a separate data file to keep track of former algorithm approaches and to be able to test and compare different variables. Since we have limited space in the directory for the UK BioBank data analysis (8 TB) we had to reduce the amount of data to be saved from the output data alone. Redoing the individual steps in the algorithm during development every time something was changed would

have been impractical and time-intensive. Therefore, our method of saving data for the GDA is neither the best practice nor the most effective. This problem would need to be remedied before offering the code for publication. Since we were already applying the GDA to the KOALAP and UK BioBank set while the SPE algorithm was still in development, we decided to publish the version that we had run extraction of the gait data with. Repeating the gait data extraction process took a long time and was not feasible to repeat before the publication of this thesis. This was in particular the case for the UK BioBank, since we included the deletion of the raw .cwa datafiles in the code. Had we not included this step, the provided memory space would not have been enough.

6. Step Parameter Extraction Algorithm Development

6.1. Abstract

The SPE algorithm consisted of three major steps: the identification of the dominant axis; the identification of steps grouped in step and gait cycles; and the extraction of gait parameters. We identified 27 parameters, which we evaluated using correlations and Principal Component Analysis (PCA). We identified the following parameters as suitable for walking behaviour analysis: number of steps within a gait episode, acceleration range of the gait cycle, step rate, symmetry of the acceleration range and the step time and the variance of the acceleration range. While the algorithm's performance was deemed sufficient, we still see many potential points for improvement.

6.2. Aims

While we first planned to develop a single algorithm that both detects walking behaviour and extracts parameters of gait behaviour, we ended up separating these aims into two separate algorithms. However, the step parameter extraction (SPE) algorithm relies on the outputs from the gait detection algorithm (GDA) and should not be applied to raw accelerometer data.

We wrote a code with the aim to extract step parameters with the plan to identify step parameters which we will be able to use to analyse walking behaviour in patients with RMDs. We named this algorithm Step Parameter Extraction (SPE).

6.3. Algorithm Development

The 2nd part of the Aston Gait Detection algorithm identifies Steps and extracts gait parameters using the Code in `AstonGDACode_StepParameterExtraction.m`. While the gait detection algorithm was mainly developed by testing it on the self-recorded dataset, the parameter selection was done by looking at the GDA output from the KOALAP dataset.

6.3.1. Housekeeping, Variable Setting and Data Loading

The purposes of most of the housekeeping and data loading steps in this code were already described in Chapter 4.3.2 and will therefore not be repeated here. A new aspect of housekeeping we included was to use the `clear` function after every code section on variables only used for the section. This keeps the workspace cleaner and reduces memory usage. Since we plan to combine the parameter outputs of multiple participants for the UK BioBank dataset, this is an important consideration. It also prevents potential cross-contamination if the data output from the previous file remains before the next loop commences. The variables already defined in the previous algorithm were saved in the data file SetUP_Variables_idv_01.mat and loaded into the workspace as well. We furthermore set three variables that stay constant throughout the code:

- `var_TimeOfDay` time categories
- `minEpisode_steps_run` minimum Steps to keep an Episode for analysis
- `minEpisode_steps_eval` minimum Steps to extract Parameters from Episode

We determine four Time Categories (`var_TimeOfDay`), the *Morning* (from 3 am – 9 am), *Midday* (9 am – 3 pm), *Afternoon* (3 pm – 9 pm) and *Night* (9 pm – 3 pm). The TC will be helpful in distinguishing patients' walking behaviours during the course of the day. Considering a typical workday is from 9 am to 5 pm in the UK, the *Morning* category was set to capture morning ritual activities and the commute to work.

As part of the constant exclusion criteria, we set a minimum of five steps during the step identification process. If an episode became too short to contain at least five steps, or if less than five potential steps were recorded, the episode was discarded. Later on, we set a minimum of 15 steps for a sustained walking episode. While we determine five steps to be enough to classify an episode as walking, we decided to require at least 15 steps in order to guarantee the extraction of step parameters.

We input the data saved from the previous Gait detection algorithm and output a file containing a structure names `WalkEpi` that contains extracted gait parameters for each walking episode. We also save the variables `var_TimeOfDay`, `var_feq`, `t_range_min`, `walk_org` and `lgt_TotalData_all` for future reference.

The variable `walk_t` had to be reformatted as in the previous code we saved the numeric information containing the dates at which the data was collected and replaced it with a count of days. Thereby the number one corresponded to the first-day data was collected from each participant, the number two to the second day and so on. The information about the dates of data collection was saved in the variable `t_range_min`. We created a variable that contains the date of data collection (`walk_t_date`), the time of data collection (`walk_t_time`), a weekday category (`walk_t_wk`) and the reconstructed original full timestamp (`walk_t_org`) for each data point. The function `weekday`, which we used to create `walk_t_wk`, labels Sundays as day 1 and Saturdays as 7. Hence, we changed the labels to start the weekday variable with Mondays being set as 1. While this assignment of weekdays starting with Mondays felt more natural, it may not be best practice going forward and should be revisited.

6.3.2. Selecting Data Axis

While we had already identified a potential dominant axis in the GDA algorithm, sometimes more than one axis was labelled as showing identifiable gait pattern. The best example of this is the "stiff" behaviour of putting your hands in your pockets. In this case both the x- and y-axis displayed a strong walking behaviour. Since we do not want to analyse the same walking behaviour twice, we had to make the decision on which axis to extract the gait parameter from.

We first attempted to select episodes according to measures such as the episode quality, or acceleration amplitude but these attempts did not prove fruitful. Oftentimes, we would have lost a lot of identifiable steps to noisy data on the identified dominant axis, while the steps were clearly visible on another, similarly expressive axis. We, therefore, had to admit that the outputs from the GDA were not sufficient to identify the ideal dominant axis.

We did not go back to change the GDA, since we had already run the algorithm on the KOALAP and UK BioBank datasets. Instead, we decided to repeat simplified versions of some of the GDA

steps in the SPE. The following paragraphs will describe the difference between the functions used in the SPE and the GDA.

`func_SPE_mstd`

In the GDA algorithm, the `func_GDA_mSTD.m` function was used to distinguish between non-walking and walking activities. The threshold used was determined by the histogram profile of the data and the step was repeated until the amount of low-amplitude moving STD data was achieved. In the SPE the goal of using moving STD-thresholding is to select the axis which best represents the walking behaviour. The function `func_SPE_mstd` is therefore also a combination of the `func_GDA_mSTD.m` and the `func_GDA_DASA.m` functions of the GDA.

The function `func_SPE_mstd` inputs are the dynamic acceleration data that was identified as walking (`walk_d`), the sample frequency (`var_feq`) and the selected threshold factor (`thld_dr_factor`). It outputs the dynamic acceleration data where excluded axis was set to `nan` (`walk_d2`) and a matrix summarising the performance of the function at the selected threshold (`outp_rate`). The later output can be seen in Figure 54 where we run the function using threshold factors ranging from 0 to 3 in steps of 0.5.

Each axis and the magnitude of the data is run through a moving STD. The magnitude is multiplied by a threshold factor of 0.75. Figure 54 shows that a threshold factor of 0.54 leaves a majority of the data with two axes to choose from and at a threshold of 0.98 most of the data has one distinct axis selected. Setting the threshold to optimise on selecting one axis causes a significant amount of data (17.39 %) that had already been identified as gait correctly to be discarded completely. On the other hand, setting a low threshold diminishes the usefulness of this processing step. At a threshold of 0.75, the amount of data discarded is an acceptable 4.09% and the majority of the data is reduced to one dominant axis (53.42%).

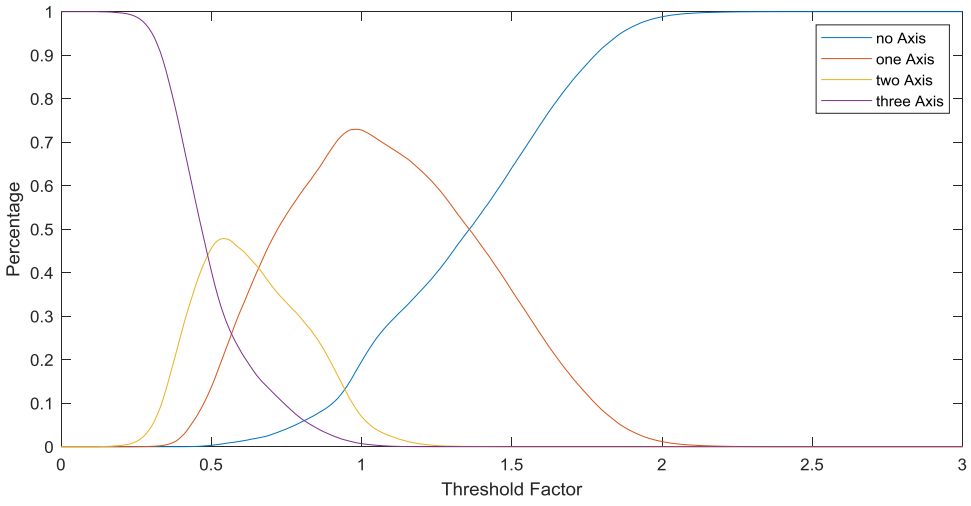


Figure 54 - ***Applying different Threshold Factors on the walking behaviour data of the SRDS****. The output displays how many axes were identified as the potential dominant axis as a percentage in the whole dataset.*

`func_SPE_removeShortEpi`

The next step is similar to the GDA function `func_GDA_Epi`. While this function was only able to remove episodes using the slow and fast walking thresholds peak counting, we can use the frequencies identified by the AC of the GDA for this new function. The function `func_SPE_removeShortEpi` inputs the dynamic walking data taken from the `func_SPE_mstd` function (`walk_d_01_mSTD`), the AC frequencies calculated by the `func_GDA_AC` function of the GDA (`walk_info_ACfreq`) and four thresholds. The thresholds are:

- 5 fast steps `thld_01 = minEpisode_steps_run * fastWak_obs;`
- 15 fast steps `thld_02 = minEpisode_steps_eval * fastWalk_obs;`
- 5 steps `thld_03_base = minEpisode_steps_run;`
- 5 slow steps `thld_03 = thld_03_base * slowWalk_obs;`

The function outputs the dynamic acceleration with excluded data set to `nan` (`walk_d3`).

`func_SPE_removeShortEpi` creates a list of start and end points for each episode of data for each axis as determined by the `func_SPE_mstd`. Then all episodes that do not reach the length of at least 5 fast steps were removed. Afterwards, we looked at episodes that were smaller than 15 fast steps and removed them if another axis completely overlapped with them. As a result, the longer axis with a sustained episode was chosen. Finally, we looked at episodes that were smaller than 5 slow steps. We looked at the minimum recorded AC frequency for these episodes and if the length of the episode was smaller than 15 steps at AC frequency, the episode was discarded.

After applying the `func_SPE_removeShortEpi` function we have removed 7.41% of the data completely and for 67.05% a final dominant axis was determined.

`func_SPE_ACslide`

While we had already extracted potential step frequencies in the GDA (see function `func_GDA_AC`), these frequencies were not flexible across longer episodes. A walking episode can contain slight variations in walking speeds. It, therefore, aims to run the sliding AC window to obtain the most likely step frequency at each data point. This method is quite computing-intensive and very sensitive to noise when applied to short episodes of data, hence it would not have been an appropriate method for the GDA.

We inputted the dynamic data derived from `func_GDA_Epi` (`walk_02_EpiAxis`), the corresponding timestamps (`walk_t_org`), the sample frequency (`var_feq`), the desired window length (`wdw_ac`, set at 10 sec), the desired slide of the window (`wdw_sl`, 1 sec), the thresholds for slow and fast walking behaviour (`slowWalk_obs` and `fastWalk_obs`), a threshold determining the minimum number of observations between episodes of data (`t_diff`), and the minimum amount of steps an episode needs to not be discarded (`min_Steps`). The output consists of the sliding AC frequency (`walk_AC_loc`) and its corresponding cross-correlation (`walk_AC_amp`).

Instead of the complicated and thorough selection process to identify peaks in the autocorrelation that indicate gait, we can code the AC sliding window with the assumption that the data analysed displays gait behaviour.

We ran the AC sliding window along each axis of the dynamic data. First, we removed the data excluded from the analysis by identifying and removing values set to `nan`. We saved the location of the analysed dataset (`i_idx`) to later reconstruct the output matrix. Then, we identified locations where the time variable indicated a break of at least five observations (`loc_forcedStop`). This method was necessary to prevent the AC window from analysing two different walking episodes at the same time. Next, for each data window, we had to check if there was a break between episodes within it. If that was the case, we divided the window and ran the AC separately. If the window was too short to contain at least 5 fast steps, we did not run the correlation. However, this did not prevent the analysis of these data segments, since previous or subsequent windows would cover them.

A normalised cross-correlation using a window size of one slow step was then applied to the data. Compared to the GDA AC, we are not looking for the homogeneity of the data; therefore, a larger window was not needed. The appropriate range to find the step frequency in was selected by removing the AC output corresponding to the fastest walk frequency. Afterwards, we tried out two ways of identifying the AC frequency by either: 1) identifying the location of the AC maximum and making sure the correlation is larger than 0.25, or 2) using `peakseek` to select the first peak that also had a higher correlation then 0. Selecting the location of the maximum correlation comes with the problem that an asymmetric gait leads to the correlation of the gait cycle being larger than the correlation at the step cycle. Therefore, the frequency of the gait cycle might be misinterpreted by this method as the step cycle frequency. Simply using the first peak comes with the hypothetical potential of noise being selected instead of the peak representing the step cycle. However, since we have already determined the episode to be gait, the likelihood of this is very minimal and we did not come across examples of this happening in the data that we explored.

Lastly, the AC frequencies identified for each window were recorded in a matrix and the median of the frequency representing the step frequency (`mtx_loc`) and correlation at a set frequency (`mtx_amp`) was calculated for each data point.

`func_SPE_SelectMainAxis`

The analysis step makes a final decision on which axis to use for the step parameter extraction. We inputted the dynamic and time data (`walk_02_EpiAxis`, `walk_t_org`), the AC frequencies both identified by the GDA and the `func_SPE_ACslide` (`walk_AC_loc`, `walk_info_ACfreq`) and the threshold for slow Walking (`slowWalk_obs`). Additionally, the variable `factor_wdwamp` multiplies the sliding AC frequency by two to identify the gait cycle length at each time point.

The function outputs the dynamic data (`DomWalk_d_preGDA`), timestamps (`DomWalk_t`), the final axis section (`DomWalk_axis`) and the associated AC frequency (`DomWalk_AC`) for each data point.

The functions run the `func_AmpOverWindowMax` function on the data, which will be described in more detail in the next Chapter. The distance between the minimum and maximum amplitude within the gait AC frequency window is determined for each data point and each axis. The axis with the highest amplitude size is selected as the dominant axis. However, to suppress rapid changes in

axis selection, the condition must be met that at least one step is taken before the dominant axis changes. Therefore, we first identify instances where the switching between dominant axes is faster than one slow step. Then we identify the minimum GDA determined AC frequency and if the section is smaller than it. Afterwards, we check if this section is surrounded by the same dominant axis on both sides. If all conditions are met the surrounding dominant axis overrides a small step dominant axis. This procedure is done to prevent instances of large noise spikes on one axis overriding the walking behaviour on another axis.

6.3.3. Identify Steps

We now have a single axis on which we can identify steps by looking for spikes in the data. While we tried different approaches using the Matlab codes `findpeaks` and `peekseek`, these functions proved to be quite computation-intensive and vulnerable to noise. One main problem was the inflexibility of setting the minimal amplitude and temporal threshold. We, therefore, developed our own method by identifying the maximum amplitude over the step frequency window.

`func_AmpOverWindowMax`

The function reads in the one-dimensional dynamic data (`DomWalk_d_preGDA`) and vector for each data point associated step frequency (`DomWalk_AC`). It outputs the maximum, minimum and range amplitude within the window surrounding each data point (`wdwamp_max`, `wdwamp_min` and `wdwamp_diff`), and lists recording the location of the amplitude of each identified hill and valley (`wdwamp_hill`, `wdwamp_vall`). Furthermore, the function can output how many hills and valleys were found (`wdwamp_lgt`) and the central location between windowed maximum and minimum amplitude (`wdwamp_center`).

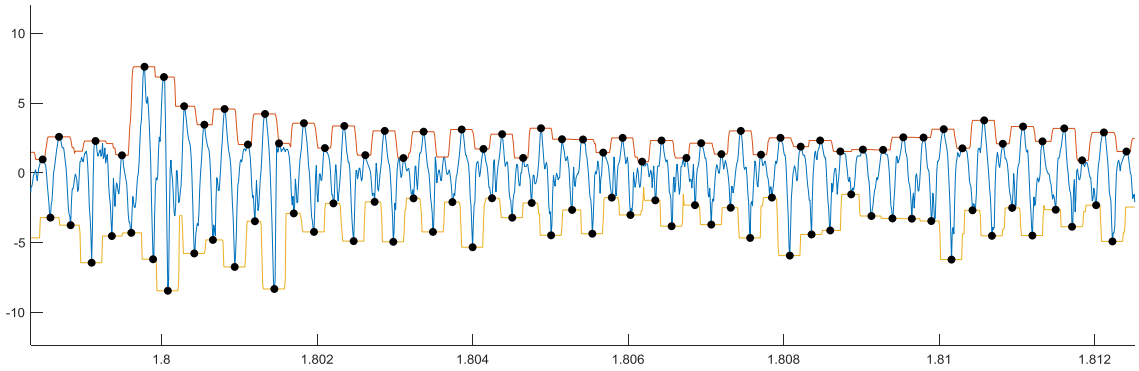


*Figure 55 - **Example of func_AmpOverWindowMax Output**. Y-axis = acceleration (m/sec^2), X-axis = observations (obs x10^5, 50 obs = 1 sec), blue = dynamic data; red = maximum windowed acceleration, yellow = minimum windowed acceleration, black = identified hills and valleys.*

First, each data point is centred into its corresponding AC window and the maximum and minimum amplitude are identified. Afterwards, we identify spikes in the data by identifying the locations where the dynamic data charges maximum and minimum acceleration windows. We named the former “hills” and the latter “valleys” (see Figure 55). The benefits of this method are that it is quick and only one hill and one valley are identified for each step.

Finally, one clean-up step needed to be conducted since spikes that consisted of two consecutive equal values would be registered as two spikes instead of one. This problem is usually

very unlikely but can occur when the data was interpolated during pre-processing. We, therefore, identified double spikes and selected the location that was closer to the centre between its surrounding spikes.

For further analysis, the outputs were recorded in one `nan` vector the length of the dataset with the amplitude of each hill and valley recorded at their respective locations. We also grouped steps into episodes. Episodes are divided by breaks in the data that were not identified as walking which were longer than two sec. The start and endpoints of each episode were recorded in the variable `strend_list`.

`func_SPE_HiVaViolation`

In order to make gait parameter extraction feasible, we had to ensure that each hill is followed by a valley. While this was mostly the case there are still exceptions to the rules: for example where the acceleration of the hill was suppressed by noise and did not reach the necessary positive acceleration to be identified as a hill. Therefore, we had to look at sections of data where this hill-valley pattern was violated and identify if it was caused by a missing spike, or by a wrongly identified spike.

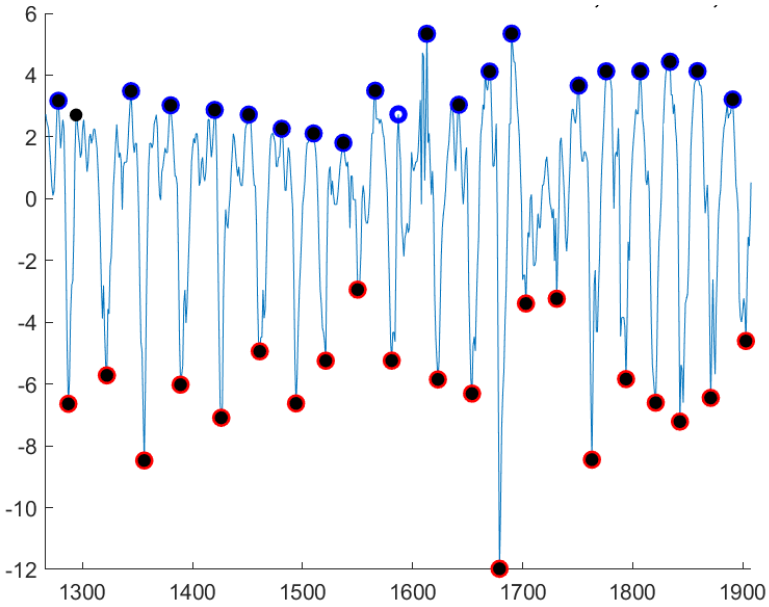


*Figure 56 - **Example of Hill-Valley-Violations**. Data sample from SRDS obs: 65587:68595. Y-axis = acceleration (m/sec^2), X-axis = observations (obs, 50 obs = 1 sec). Light blue = dynamic data, saturated blue/red = hills/valleys identified by func_AmpOverWindowMax; Black = hills/valleys corrected by func_SPE_HiVaViolation.*

We loop the `func_SPE_HiVaViolation` function for each identified episode. It inputs the dynamic data (`DomWalk_d_exp`), the hill (`DomWalk_hill_pre`) and the valley (`DomWalk_vall_pre`) vectors and returns corrected hill and valley vectors (`DomWalk_hill_pos`, `DomWalk_vall_pos`).

The function first identifies violations within the episode (`idx_hivaVio`). If a violation is part of the episode, it identifies the mean and STD of the amplitude and distance of the hills and valleys for the whole episode (`error_amp_mstd_hill`, `error_loc_mstd_hill`, `error_amp_mstd_vall` and `error_loc_mstd_vall`). We also determine an upper and lower

range by subtracting and adding the STD to the mean (`error_***_rang_**ll`). This range can be used to judge if the spike in question is or would be an outlier.

After each violation is looped, the violation location (`i_list_err`) and the dynamic data between them (`i_d`) is selected. For each violation, we have to make a decision whether to add or delete a spike. The variables `potential_deletion`, `potential_addition`, `sure_deletion` and `sure_addition` keep track of our decision. First, we identify whether the violation consists of two consecutive hills or valleys, then we assign the variables: `error_rang_loc_flip`, `error_mean_amp`, `error_mean_amp_flip`, `error_DomWalk`, `error_DomWalk_flip`, `error_amp_val` and `error_amp_loc`. If the violation consists of consecutive valleys the `*_flip` variables are assigned to the hill values and vice versa. The `*_range_*` and `*_mean_*` variables take the values of the corresponding `error_*` variables above, and the `*_DomWalk*` variables use the `DomWalk_hill_pre` or `DomWalk_vall_pre` input variable. At last `error_amp_val` and `error_loc_val` identify the potential new spike between the violation by taking the maximum (for two valleys) or the minimum (for two hills) of the dynamic data between the violation spikes. The flipping of the data ensures that we can always treat the violation as if either a hill is missing or an additional valley is present. This way we do not have to write the same code twice for both situations.

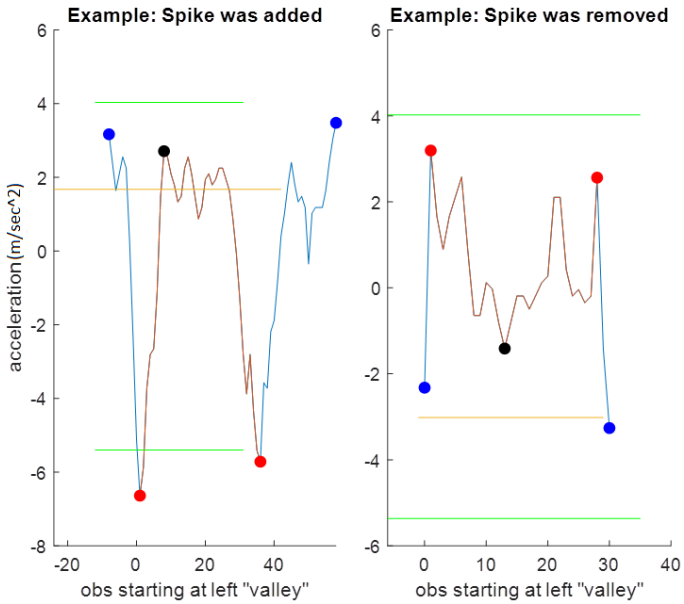


*Figure 57 - **Example of Hill-Valley-Violations zoomed in**. Left = adding a spike, right = deleting a spike. Blue line = area of violation, blue dots = surrounding hills, red line = area to be assessed, black dot = potential additional spike, red dots = potential surplus spikes. Green line = mean of the episode's hills and valleys (y-axis) & upper distance range limit (x-axis). Yellow Line = ¾ distance threshold (y-axis) & distance between surrounding hills (x-axis).*

Next, we extract the distance between the identified potential new spike and the average hill and valley amplitude (`distancetoVall` and `distancetoHill`). If the potential peak is closer to the mean hill amplitude than the mean valley amplitude, we add to the `potential_addition` variable, else we discarded the potential new spike (set `sure_deletion = 1`). If the new peak lays ¾ of the distance between the valley and the hill mean, we consider the new peak valid for sure (set `sure_addition = 1`, see Figure 57 left).

Then we looked at the distance between the hills surrounding the valleys in question. We investigate if at least one more hill can be pressed between the surrounding hills, without significantly impacting the average distance between hills of the episode. If the surrounding hills are further apart than the upper limits of the distance range, we can assume with a high likelihood that a hill is missing between them. If the distance between the surrounding hills (`i_loc_diff`) is larger than the upper limit of the range distance between hills (`error_rang_loc_flip(1,2)`), then we add to the `potential_addition` variable, else we discarded the potential new spike. If surrounding hills are missing and we cannot make this judgement, we add to the `potential_deletion` variable. Figure 57 (right) displays an example where the amplitude of the potential spike was not close enough to the average “hills” (in this case there are flipped valleys). Furthermore, the surrounding “hills” were not far enough apart to warrant a spike regardless.

If the decision was made to add a new spike, the new spike was added to the `error_DomWalk_flip` variable. If a new spike is not added, one of the spikes in question needs to be removed to ensure the hill-valley pattern is maintained. We identify (`iii_idx_delete`) and remove the spike from the `error_DomWalk` variable.

Finally, we update the output vectors (`DomWalk_hill_exp`) and (`DomWalk_vall_exp`) depending on the violations' former state of two consecutive valleys or hills. Each episode's output vector is applied to the whole dataset (`DomWalk_hill_02` and `DomWalk_vall_02`).

`func_SPE_GaitCycleIdf`

In order to extract step parameters such as symmetry, we need to not only identify step cycles but also gait cycles. Furthermore, we want to standardise the gait and step cycles to start with the hills that have the highest possible mean amplitude. We can assume that the hills with a higher amplitude will correspond to the heel-strike of the dominant leg (for “stiff” walking behaviours) or the forward swing of the dominant arm (for the “swing” walking behaviours). We also might have to flip the dynamic data and the hill and valley amplitudes around. This approach ensures that the identified acceleration force is somewhat standardised since we cannot deduce the orientation of the watch worn by the participant or the wrist orientation while walking (see Chapter 4.2.2).

`func_SPE_GaitCycleIdf` sorts the spikes from hills and valleys vectors into step and gait cycles. The function is looped for each episode. It inputs the episode’s dynamic data (`DomWalk_d_preGDA`), hill and valley vectors (`DomWalk_hill_02` and `DomWalk_vall_02`), and the threshold for the minimum amount of gait cycles to keep the episode in the analysis (`min_gaitCylces`). It outputs the adjusted one-dimensional dynamic data (`DomWalk_d_rslt`), an indicator for the orientation change of the data (`max_mean`) and four variables describing the location of the dominant and subsequent steps hills and valleys (`HillList_1st_pos`, `HillList_2nd_pos`, `VallList_1st_pos` and `VallList_2nd_pos`).

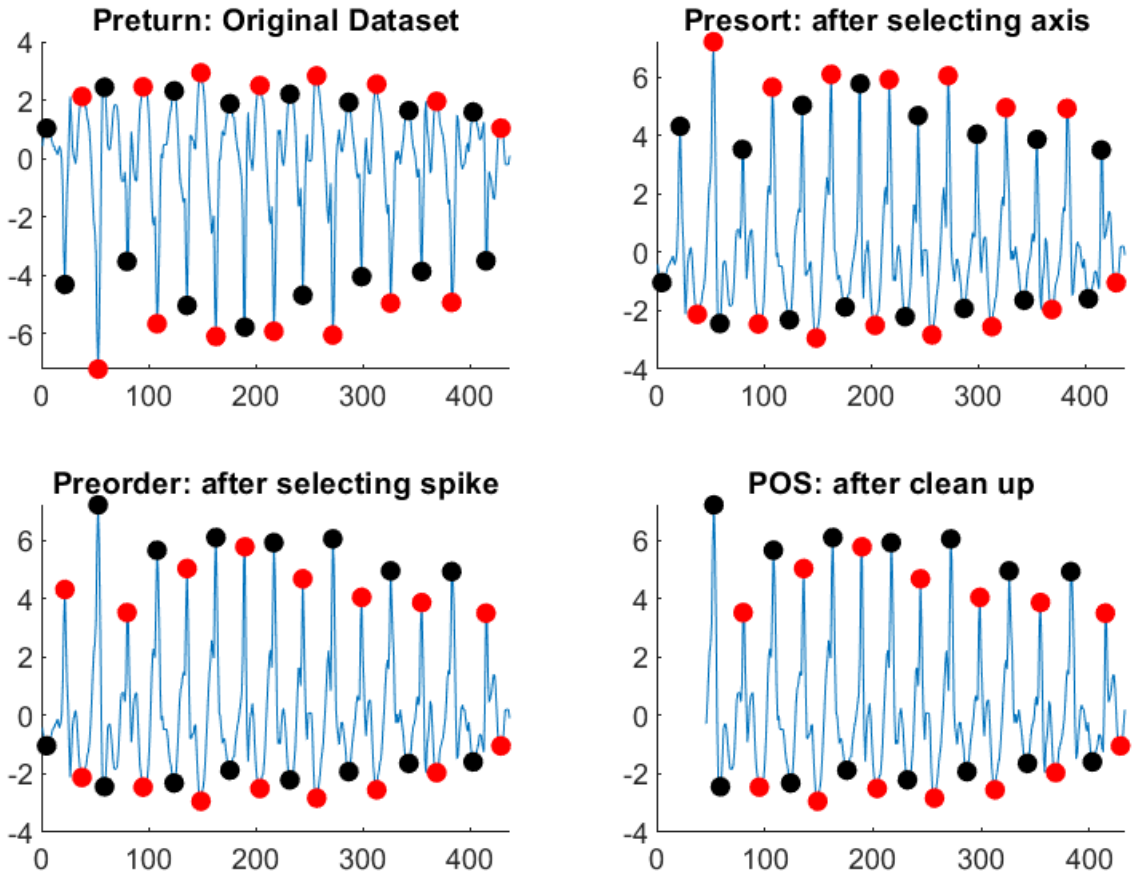


*Figure 58 - **Example of Gait Identification with MaxMean = 4**. X-axis: data points over time (obs), Y-axis: acceleration (m/sec^2). Dots with positive acceleration = hills, dots with negative acceleration = valleys. Blue line = dynamic data of the dominant axis, black dots = dominant steps (1st), red dots = subsequent steps (2nd).*

First, we extracted the location and amplitude from the hill and valley vectors and separated every second hill and valley into a separate list, resulting in four variables: `HillList_1st_preturn`, `HillList_2nd_preturn`, `VallList_1st_preturn` and `VallList_2nd_preturn` (see Figure 58 upper left). If all variables have a minimum length of 5, the required 5 gait cycles are possible, else the episode is discarded. We then calculate the mean and STD of the four variables (`Calc_MSTD`) and identify the variable with the highest mean (`max_mean`). If the maximum mean falls onto one of the valley variables (`if(max_mean==3)||(max_mean==4)`), we flip the dynamic data and the spike variables by multiplying them by -1 (see Figure 58 upper right (`*_presort`)). This way the dominant spikes are ensured to have positive acceleration amplitudes. Furthermore, we switch which spikes we consider the “dominant” (1st) or “subsequent” (2nd) hill if the hill with the highest mean amplitude was initially sorted into the “subsequent” hill variable (`if(max_mean==2)||(max_mean==4)`, see Figure 58 lower left (`*_preorder`)). If at this point the walking episode does not start with the dominant hill and ends with the subsequent valley, we remove the spikes necessary to achieve such a pattern (see Figure 58 lower right (`*_pos`)). We also replaced the dynamic data section before the first hill and after the last valley with `nan`. The data was cut off at the first value before passing 0 for the hill and the first value after passing 0 for the valley. At last, we updated the dynamic data output and the step location and amplitude variables.

`func_SPE_CreateLocListTo0_V2`

Staying in the same episode loop, we identify the start and end location of each step and gait cycle using the function `func_SPE_CreateLocListTo0_V2`. For the gait cycle, it inputs the location of the dominant hills and the subsequent valleys and for the step cycle, the sorted location of all hills and all valleys are inputted (`HillList_1st_loc` and `VallList_2nd_loc`). Additionally, the dynamic data (`DomWalk_d_rslt`) output from the `func_SPE_GaitCycleIdf` is needed. It outputs a variable containing the start and end location of each step/gait cycle of the dynamic data of the episode (`loclist_Gait_strend` and `loclistSPEstrend`).

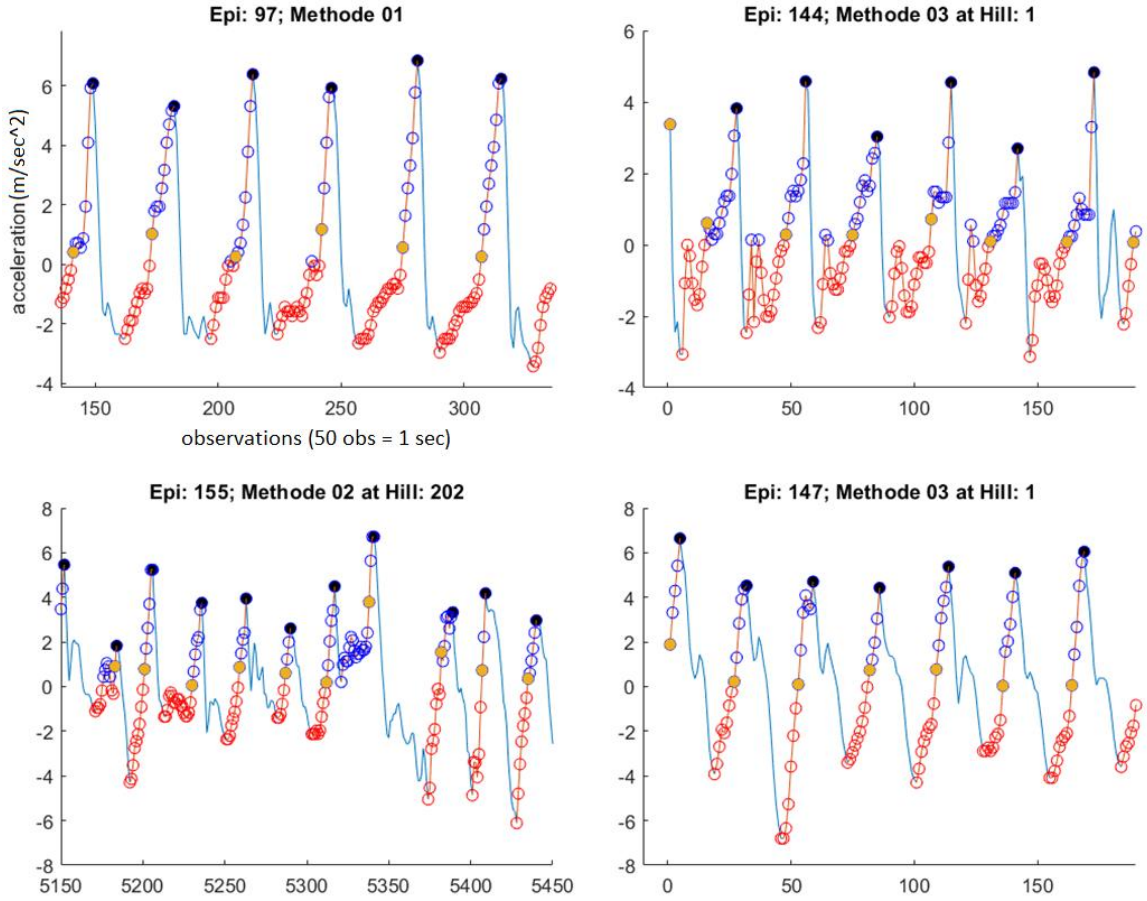


Figure 59 - ***Examples of Creating the Loc to 0 List for Step Cycles****. Black dots = hills of interest, empty dots = data points (obs) between the previous valley and hill of interest, blue dots = obs with positive amplitude, red dots = obs with negative amplitude, yellow dot = identified start point for each step cycle.*

The function loops each step/gait cycles starting hill, identifies the dynamic data preceding the hill (`i_d`) and sorts it into observations with positive amplitude (`i_str_pos`) and negative amplitude (`i_str_neg`). Then the first positive observation that occurs after the last negative observations is selected as the start of the step cycle (`i_str`, see Figure 59 upper left, e.g. 4th hill). However, if no negative/positive obs or a start point is found, a different method is required. The third method is applied when the first hill is examined and a previous step in the algorithm has cut off the dynamic data to close to the hill for preceding negative observations to be found (see Figure 59 right). In this case, we decided to use the start of the dynamic data as the start point of the step cycle. A more difficult decision was what to do if the preceding valley has a positive acceleration

(see Figure 59 lower left, 8th Hill). In this case, we looked at the acceleration of the hill in question and the preceding valley found the centre between them and declared it as the new threshold for the step cycle start.

The end of the step/gait cycle was then determined by taking the start time of the next cycle and subtracting one observation from it. For the last cycle of each episode, we determine the end of the cycle by extracting the location of the first time the acceleration data turns positive and selecting the observation before this occurs. Alternatively, we select the last data point containing dynamic data of the episode. All the resulting start and endpoints of each cycle are saved in the variable (`loclist_Gait_strend`).

6.3.4. Recording Data for each Episode

In order to save the output of the SPE we created a structure called `WalkEpi` to record the information of interest we extracted for each identified walking episode:

- Start and end location of each episode (`.strend`)
- Locations of the dynamic data that was set to `nan` (`.idx_nan`)
- Orientation identified by the function `func_SPE_GaitCycleIdf` (`.orientation`)
- The axis that the dynamic data was made of (dominant axis, `.axis`)
- Timestamp (`.t.datenum`), Weekday (`.t.WK`), Time of Day (`.t.meanTime`) and Time Category (`.t.TC`)
- The total number of Spikes/Steps identified within the episode (`.Calc_NSpk`)
- The variables describing the location and amplitude of the dominant and subsequent Hills and Valleys (`.Spk.Hill1st`, `.Spk.Hill2nd`, `.Spk.Vall1st` and `.Spk.Vall2nd`)
- The start and endpoints of each Step and Gait Cycle (`.Spk.locList_Step` and `.Spk.locList_Gait`)

Episodes that contained less than 5 identified steps were excluded. Furthermore, we saved the dynamic data as outputted by the function `func_SPE_GaitCycleIdf` as `DomWalk_d`.

6.3.5. Selection of Step Parameters

`func_SPE_EP`

The last function of the SPE extracts parameters from the previously identified step and gait cycles. It inputs the `WalkEpi` structure, the dynamic data (`DomWalk_d`) and the sample frequency (`var_feq`). It adds to the `WalkEpi` structure by adding the extracted parameter outputs to each episode within the structure. An earlier version of the function was used while processing the KOALAP dataset (`func_SPE_EP_originalKOALAPversion`), the `func_SPE_EP` function incorporates processing steps that were developed while examining the step parameters of the KOALAP data. It also simplifies the output for the UK BioBank dataset by only recording the mean of the step parameters of each episode instead of saving the parameter for each step and gait cycle.

First, the function prepared the output structure by copying over the information saved in the input structure. Then potentially interesting Step Parameters were extracted for all Steps Cycles (`Calc_allStep_***`), each dominant Step cycle (`Calc_domStep_***`), subsequent Step Cycle (`Calc_subStep_***`) and all Gait Cycles (`Calc_allGait_***`) of the episode. Then the means of these parameters were taken and added to the `WalkEpi_output` structure (`.Epis_allStep_***`, `.Epis_domStep_***`, `.Epis_subStep_***` and `.Epis_allGait_***` or `.Epis_Gait_***`).

In Chapter 1.1 Table 1 we described potential gait parameters, which we derived from the methodology of the papers analysed in the literature review. These were divided into measures of spatial descriptors, temporal descriptors, spatial-temporal descriptors and gait quality measures.

In order to be able to extract **spatial descriptors**, we considered calculating step and stride (gait cycle) lengths using trigonometry. That would have required the acceleration data and body measures from the participants, such as leg and arm length. While we did have those measures available for all of our datasets, we ran out of time to implement these calculations. Therefore, we did not obtain the **spatial descriptors** of step length, stride length, or walking speed parameters.

However, we were able to identify an alternative measure for the **spatial-temporal descriptor** of **Velocity** by measuring the area under the curve for each step. Therefore, we extracted the dynamic data of the step or gait cycle (`i_gd`). Then the `polyarea` function was used on both the positive and negative acceleration and added together (`polyarea([1 1:length(i_gd) length(i_gd) 1]/var_feq,[0 abs(i_gd) 0 0]`). We named the velocity output variables `***_Velocity`.

We also identified the **acceleration magnitude descriptors** of force and range**,** which can tell us something about the magnitude of the arm swing during “swing” walking behaviours or the magnitude of the heel-strike in “stiff” walking behaviours. **Acceleration Force** (`***_AccForce`) was identified as the acceleration maximum of each step. We did not report the `***_allGait_AccForce` variable since it would be the same as the `***_domStep_AccForce` variable. **Acceleration Range** (`***_AccRange`) was identified as the difference between the acceleration minimum and maximum for the step or gait cycle (`max(i_gd)-min(i_gd)`).

We were able to extract the **temporal descriptors** of stride time, step time and walking cadence directly by measuring the temporal distance between spikes in the data. While our first attempt consisted of extracting the location of the spikes by measuring the distance between them, we decided that taking the start and endpoints of the step and gait cycles would be more accurate and less prone to distortion due to noise. The **Step Times** and **Stride Times** (`***_TimeSec`) where measured as time in sec (`((loclistSPEstrend(:,2)-loclistSPEstrend(:,1))')/var_feq`). We furthermore extracted the episode length (`Epis_Epi_min`) in minutes (`(WalkEpi_input(i_epi).strend(2) - WalkEpi_input(i_epi).strend(1))/var_feq/60`) and the **Step Rate** (`Epis_SPE_RateMin`) as steps per minute

(`WalkEpi_input(i_epi).Calc_NSpk/WalkEpi_input(i_epi).Calc_Epi_min`) for the whole episode.

Finding measures for the **Gait Quality** was a little less straightforward. We managed to extract measures of **Symmetry** (`***_Symmetry_AccForce`, `***_Symmetry_AccRange`, `***_Symmetry_TimeSec` and `***_Symmetry_Velocity`) by using the parameters of dominant and subsequent steps to extract their difference (`diff([ WalkEpi_input(i_epi).Calc_subStep_***; WalkEpi_input(i_epi).Calc_domStep_***])`). Symmetry is a potentially useful measure to identify if a patient burdens one leg more than the other. In order to identify a measure of the smoothness of the walk, we decided to extract the **Variance** (`***_Variance_AccForce`, `***_Variance_AccRange`, `***_Variance_TimeSec` and `***_Variance_Velocity`) of the data by taking the STD of the acceleration magnitude and temporal descriptors (`std(WalkEpi_input(i_epi).Calc_allStep_***)`).

At last, we tied to identify a measure of **Quality** for the episodes, which could have the potential to be used for excluding gait episodes for analysis that are very noisy and therefore not optimal to extract gait parameters from. We extracted the additional **Changes over 0 per Spike** (`Epis_chng0PerSpike`) measure. It takes the steps identified within the episode, multiplies them by 2 and divided them by how often the dynamic data of the episode passes 0 g (`((WalkEpi_input(i_epi).Calc_NSpk*2)/ (length(find(diff(iepi_d < 0) ~= 0)))`). Since each step consists of a valley and a hill, the graph should pass 0 g two times per step in the case of high-quality data.

6.3.6. Saving the Data

For the KOALAP data, we identified the pain score for each pain questionnaire type for each episode. The outputs from the morning and afternoon pain were combined. We extracted the time and score of each pain data point and identified the closest consecutive report of pain for each walking episode. The outputs were recorded in the `WalkEpi_output` structure as `.Pain_Closest_***`. We also recorded the pain type for the aggravating and important function pain scores (`.Pain_Typestr_***` and `.Pain_Typestr_***`) and reported the time difference between the walking episode occurrence and its associated pain report (`.Pain_Timedif_***`). Each participant of the KOALAP dataset was saved in separate data files and combined later in a separate code. The individual participant files contained the `.Calc_***` step parameter outputs for each step and gait cycle.

For the UK BioBank dataset, we loaded the file that contained the participants' group identifiers and applied them to each participant. The `.Calc_***` output of each participant was not saved; only the mean of the step parameters for each episode was saved. Within the algorithm, the data extracted from each participant was saved within the `WalkEpi_output_AllUser` structure. At the end of the algorithm's run, the full dataset was saved.

Furthermore, the processing time for each file was calculated and saved as the variable `ProcessingTime`. We also wrote a little section of code that would update the estimated processing time for the rest of the algorithm's run.

6.4. Algorithm Performance

6.4.1. Data Reduction of the Individual Algorithm Steps

Table 11 - Axis Selection and Exclusion of Gait Data for Step Parameter Analysis of the SRDS.

Percentage of data No identified with x-axis after applying function:	0 Axes	1 Axis	2 Axes	3 Axes
func_SPE_mstd	4.09%	53.42%	33.09%	9.40%
func_SPE_removeShortEpi	7.59%	66.89%	23.39%	2.13%
func_SPE_SelectMainAxis	7.81%	92.19%	-	-
func_SPE_GaitCycleIdf	13.68%	86.32%	-	-

6.4.2. Data Reduction of the Individual Algorithm Steps

Table 11 describes the process of axis elimination for the selection of the dominant axis. The three functions consisting of a moving STD, the removal of short episodes and the final selection of the main axis each took a significant part in the axis selection. A total of around 8% of the data was excluded in the process. Most of this excluded data consisted of data that should have been identified as non-gait by the GDA but wasn't. In particular, a lot of the coffee trial data was correctly removed from further analysis. The rest of the removed data consisted of noisy data with the free walking trial, and data surrounding the starting point of the walking episodes in the staircase and corridor controls (see Figure 60). In total, these steps excluded 5.48 min of data out of 72.26 min of GDA identified walking behaviours.

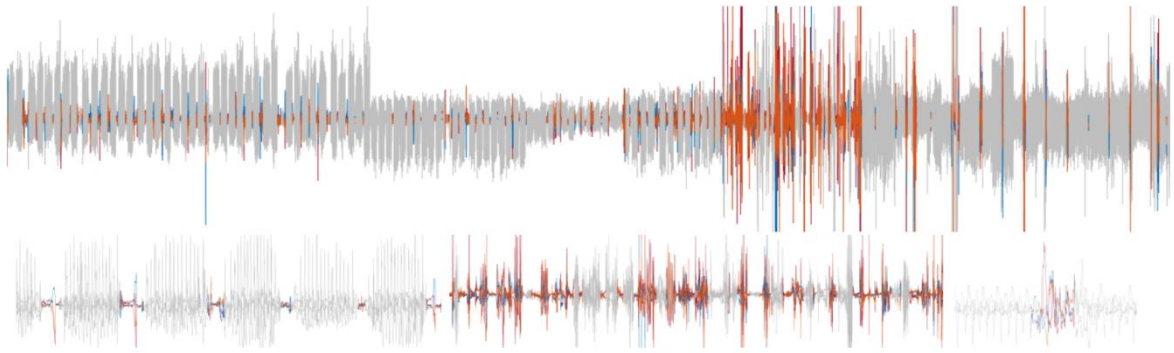


*Figure 60 - **All Data Sections the SPE identified as non-gait up to the func_SPE_removeShortEpi function**. Coloured = misidentified data (data_Lab = 0 & z = 1), light grey = data labelled non-gait (data_Lab = 0); dark grey = data labelled gait (data_Lab = 2), black vertical lines = breaks between data instances.*

The rest of the data that was excluded by the removal of short episodes and the selection of the main axis mainly consists of the same data patterns that were also excluded by the moving STD (see Figure 61). Very notably, nearly all of the sections of the coffee trial data that were still left after the GDA have now been removed. Also, two triggers that were not previously were finally successfully excluded from the analysis. A lot of small drops of data happen when a walking behaviour is interrupted by noise; hence the SPE needed to tolerate breaks of less than two sec

within episodes. However, at least one large section of data was excluded even though it looked like step parameter extraction should have been possible. In total, 4.29 min of data were removed between the `func_SPE_removeShortEpi` and the rest of the SPE.

Overall the step parameters of 62.37 min of data were extracted from the SRDS by the SPE, which is very close to the 65.60 min of data that were labelled as gait.

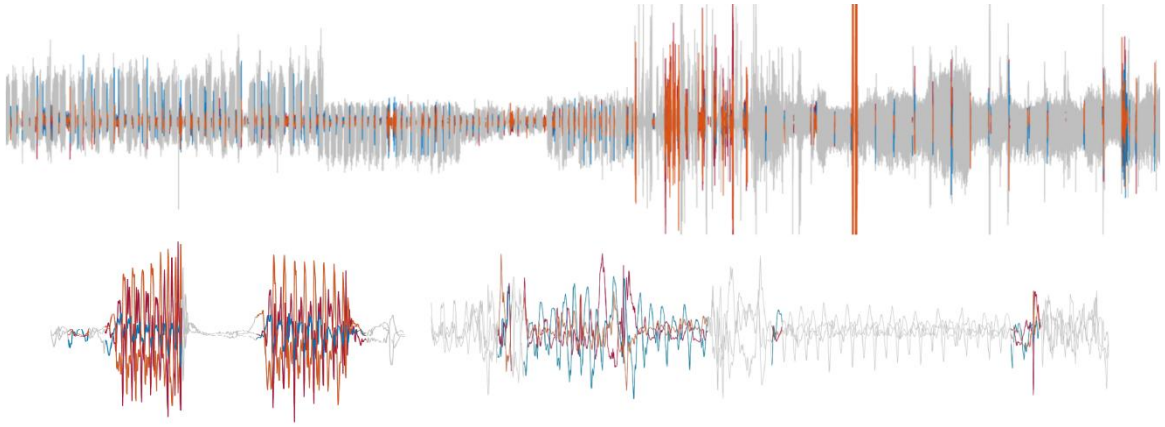


*Figure 61 - **All Data ections the SPE identified as non-gait after the func_SPE_removeShortEpi**. Zoomed in sections (below). Coloured = misidentified data (data_Lab = 0 & z = 1), light grey = data labelled non-gait (data_Lab = 0); dark grey = data labelled gait (data_Lab = 2), black vertical lines = breaks between data instances.*

6.4.3. Processing Time of the Individual Algorithm Steps

Running the SPE on the SRDS (72.26 min of data collected at 50 Hz after the GDA) took 5.94 sec total. This translates to a processing time of 4.93 sec per h of data. The individual steps took:

- Loading of the data 0.21 sec
- func_SPE_mstd 0.26 sec
- func_SPE_removeShortEpi 0.07 sec
- func_SPE_ACslide 2.60 sec
- func_SPE_SelectMainAxis 0.84 sec
- func_AmpOverWindowMax 0.57 sec
- Step episode creation including: func_SPE_HiVaViolation, func_SPE_GaitCycleIdf and func_SPE_CreateLocListTo0_V2 1.40 sec
- func_SPE_EP 0.62 sec
- Saving the data 0.61 sec

Most of the processing time of the SPE was sacrificed to the AC sliding window. It would therefore be not advisable to implement an AC sliding window on a dataset containing mostly non-walking behaviours since it would be a waste of processing time. Overall the SPE is an appropriately fast way of extracting step parameters and compliments the GDA's processing time well, adding up to 20.00 sec/h together.

In Incline, processing the complete KOALAP data set using the SPE on the gait data identified by the GDA took 1.33 h in total. The processing times varied greatly, with a minimum of 5.64 sec, a maximum of 10.50 min, a mean of 3.08 min and a STD of 2.38 min. This was mostly due to the

large variability in dataset size between participants. The UK BioBank dataset of 17.422 participants took a combined 169.94 h (7.08 days). The fastest individual file took 0.99 sec and the longest took 4.41 min. On average the processing took 35.12 sec (STD = 23.66 sec, see Figure 62). Each participant started with exactly 7 days' worth of data, therefore, we can explain the high variability of processing times on the varying amounts of data identified as gait by the GDA for each participant.

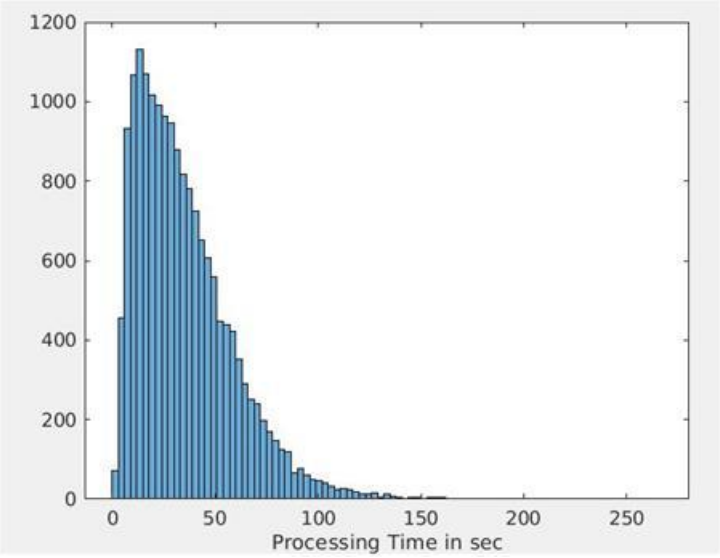


*Figure 62 – **Histogram of the UK BioBank SPE performance time per Participant File**. Y-axis = counts.*

6.5. Step Parameter Evaluation and Selection for Analysis

6.5.1. Variations between Datasets

The following results discuss datasets extracted from the SRDS, the KOALAP data and the UK BioBank data. The SPE outputs the `WalkEpi_***` structure, that contains:

- the participants' ID,
- 2 variables describe the dynamic data of the episodes,
- 4 variables describe the time of recording of the episodes,
- for the KOALAP data: 5 variables with pain score recordings, 5 variables with their associated time and 4 variables to describe the users' pain context,
- for the UK BioBank data: 5 variables indicating the participants' RMDs condition and 1 variable identifying the participants' matched control,
- and 27 Step Parameter Variables.

For the KOALAP data, we looked at the individual participants walking behaviour more closely by using the mean of each gait episode for analysis. Since a great deal of data was present for each participant, we removed the episodes that contained less than 15 steps.

For the UK BioBank dataset, on the other hand, we used the **weighted mean** of each variable for all episodes of the participants' data. We wrote an additional code outside of the SPE that uses the number of steps of each episode (`wmean_vect = ***.Epis_NSk`) to extract a weighted mean (`v_result = sum(v_data.*wmean_vect,2)/ sum(wmean_vect);`) of the individual

Step Parameters (`v`) extracted from all walking episodes (`v_data = extractfield(WalkEpi(i), VariableNameList(v)`) of a participant (`i`). We also adjusted the symmetry variable of the temporal and temporal-spatial variables (`Epis_Symmetry_TimeSec` and `Epis_Symmetry_Velocity`) by taking the absolute value. Therefore, the differentiation between a faster dominant step than the subsequent step and a faster subsequent than a dominant step was removed. We discovered while looking at the variable outputs of the KOALAP data (see Chapter 7.9, Figure 82, Symmetry of TimeSec (sec)) that individual participants temporal symmetry parameters tended to display a normal distribution stretched over both negative and positive symmetry values, but with a distinctly positive or negative mean. The symmetry of the acceleration range and force were not affected, since acceleration force determines the identification of dominant and subsequent steps. The data was saved in the `SPE_Data_wmean` structure, which is identical to the `WalkEpi_***` structure, except that each participant only holds one value per step parameter instead of one value for each episode.

6.5.2. Extracted Step Parameters

	Parameters		
	Episode Length	**Temporal**	**Temporal-Spatial**
Episode	Epis_NSpk Epis_Epi_min	Epis_SPE_RateMin	
Step Cycle **Dominant** **Subsequent**		Epis_allStep_TimeSec Epis_domStep_TimeSec Epis_subStep_TimeSec	Epis_allStep_Velocity Epis_domStep_Velocity Epis_subStep_Velocity
Gait Cycle		Epis_Gait_TimeSec	Epis_Gait_Velocity
Symmetry **Variance**		Epis_Symmetry_TimeSec Epis_Variance_TimeSec	Epis_Symmetry_Velocity Epis_Variance_Velocity
	Acceleration Magnitude		
Step Cycle **Dominant** **Subsequent**	Epis_allStep_AccForce Epis_domStep_AccForce Epis_subStep_AccForce	Epis_allStep_AccRange Epis_domStep_AccRange Epis_subStep_AccRange	
Gait Cycle	Epis_allStep_AccForce	Epis_allGait_AccRange	
Symmetry **Variance**	Epis_Symmetry_AccForce Epis_Variance_AccForce	Epis_Symmetry_AccRange Epis_Variance_AccRange	
	Other		
Episode	Epis_chng0PerSpike		

Table 12 – Overview of the SPEs identified step parameters variable names.

The Step Parameters extracted by the SPE can be sorted into some general categories. They are summarised in Table 12. Parameters describing the volume of an episode consists of length in minutes (`Epis_Epi_min`) and the number of steps (`Epis_NSpk`). Then we have the temporal parameters of the step rate (`Epis_SPE_RateMin`) in steps/min and the step or gait cycle time (`Epis_***_TimeSec`) in sec. The temporal-spatial parameter consists of the total velocity of the step or gait cycle in m/sec. The acceleration magnitude parameters consist of the acceleration force and range (`Epis_***_AccForce` and `Epis_***_AccRange`) in m/sec^2. Furthermore, each parameter is assessed under different conditions. Some values were taken once per episode, others were taken once per step cycle and once per gait cycle. The step cycle measures were

further divided into dominant and subsequent steps. Additionally, the symmetry and variance of the temporal, temporal-spatial and acceleration magnitude parameters were calculated once per episode.

Figure 63 displays episodes of data that were selected from the SRDS for having the minimum and maximum value of selected step parameters. We decided on giving examples of the data in this way to illustrate both the step parameters' effect on the data and to see if the step parameters themselves could potentially be used to judge the quality of an episode. The variable `Epis_chng0PerSpike` was initially created to make such a judgement, however, we did not find it to be sufficient for this task. Nevertheless, a lower rate of changes between positive and negative acceleration does affect the smoothness of the data (see Figure 63, upper left) and can be associated with a higher probability of the algorithm wrongly identifying the best valley locations. Looking at the data, the length of the episode (`Epis_Epi_min`), acceleration magnitude (`Epis_***_Velocity` and `Epis_***_AccRange`) and the variation of step time (`Epis_***_TimeSec`) do have potential. However, we decided to control for quality by only removing episodes of less than 15 steps from the KOALAP data.

The maximum symmetry of the acceleration range (see Figure 63, middle of the lower centre) and of the step time (see Figure 63, left of the lower centre) displays a good example of non-symmetrical walking, where the force of the subsequent step is much lower than the dominant step. The difference between the minimum and maximum variance of step time (see Figure 63, bottom left) at the acceleration range (see Figure 63, bottom centre) displays the difference between very regular gait and non-regular gait. The maximum variance of the step velocity (see Figure 63, bottom right) seems to be strongly affected by the symmetry of the gait, yet ironically the symmetry of the step velocity seems less so (see Figure 63, lower centre right). This is due to the properties of the symmetry of the step time variable. The more negative the value, the larger the subsequent step cycle time to the dominant step cycle time, while it is the other way around for more positive values. Values closer to 0, therefore, indicate more symmetrical step times, while higher magnitude negative and positive values indicate more asymmetrical step cycle times.

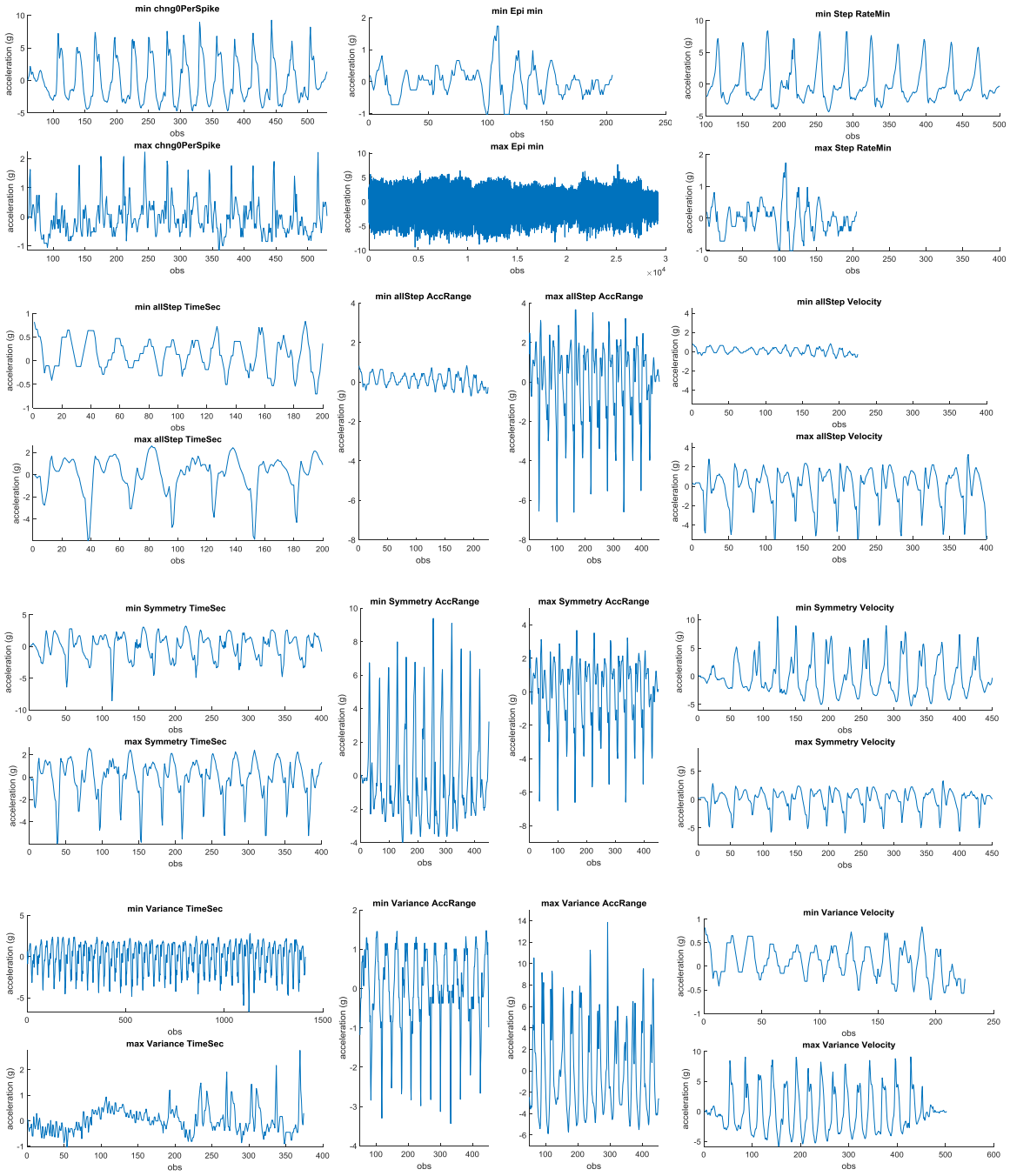


Figure 63 - ***Episode with the minimum and maximum value for selected step parameters****.*

6.5.3. Reduction of Variables for Analysis – Looking at Correlations

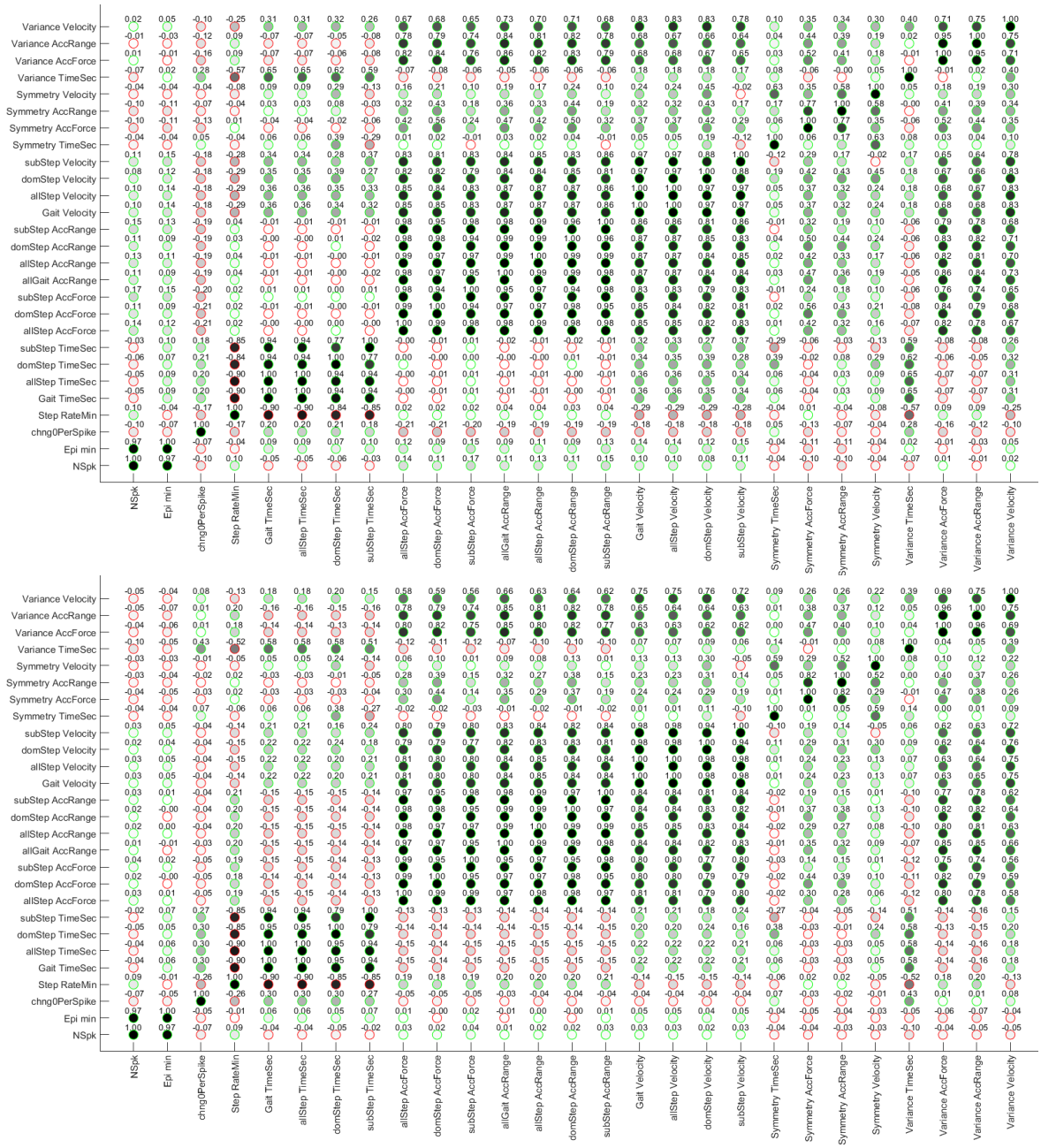


Figure 64 – ***Correlations between Variables of the KOALAP*** *(above)* ***and UK BioBank*** *(below)* ***datasets****. Red cycle = correlation coefficient is negative, green cycle = correlation coefficient is positive, black dots = correlation coefficient is 1, darker dots = correlation coefficient is closer to 1, lighter dots = correlation coefficient is closer to 0, while dots = correlation coefficient is 0. Correlation Coefficients (r) are reported above the corresponding dots.*

A lot of the 27 extracted Step Parameters are very similar, therefore we examined them to reduce the number of variables used for analysis. First, we checked the correlation between all the variables on both the individual episodes of the KOALAP data where the step parameters of all participants were combined and on the weighted means for all participants of the UK BioBank dataset (see Figure 64), in order to look for and confirm similarities between variables. Most notable is that the correlation coefficients between variables are very similar between the two datasets. Furthermore, we have very distinct positively correlated clusters.

The first cluster consists of the variables describing the episode length, which is not correlated at all with any other variable. The changes over 0 g per spike are also very distinct and only show a small correlation with temporal variables and a medium correlation with the variance of step time.

The second cluster of interest contains the temporal variables, which also show a medium correlation with the variance of step time and some of them show a small correlation with the symmetry of step time.

The third cluster contains the acceleration magnitude and temporal-spatial variables. The corresponding variance strongly correlated with them, but a distinction can be made between the correlation strength of the variance of the acceleration magnitude with the temporal-spatial variables and vice versa. The variables in the cluster only show a small correlation with the corresponding symmetry variables. Notably, the temporal-spatial variables also show a small correlation with the temporal variables.

The fourth cluster of interest contains the symmetry variables. This part of the data shows a variety of correlation strengths, notably acceleration force and range are strongly correlated and a medium correlation exists between velocity and step time, and velocity and acceleration magnitude. There do exist some small to moderate correlations between symmetry and corresponding variance variables. Other notable small to medium correlations are already discussed above.

The cluster of interests concerns the variance variables. Again, the variance between acceleration force and range shows a strong correlation, while velocity displays a medium correlation with the variance of acceleration magnitude and step time. The correlations with variables of other clusters were already described, but we mention again that the variables corresponding to each variance variable tend to be in the medium to strong correlation range.

In summary, the correlation matrix displayed in Figure 64 suggests that we extract at least one variable for the episode length parameters, the temporal parameters, and the acceleration magnitude and velocity parameters. We decided to exclude the changes over 0g per spike variable from now on from further analysis. Furthermore, while there is some overlap between the variables of symmetry and variance with the already identified categories of interest, we do have an interest in identifying some parameters for these step parameters, since they have the potential to explain behaviour variations in patients with RMD.

6.5.4. Reduction of Variables for analysis – Looking at Principal Component Analysis

We used a Principal Component Analysis (PCA, `[coeff,~,~,~,explained,~] = pca(X);`) to examine the previously identified variables clusters closer. Table 13 shows the results of the PCA of the data of each episode of the KOALAP and UK BioBank datasets. We document how much of the variance is explained by each component and report the coefficient for each of these components until the sum of the variance explained by the components is at least 99.00%.

		KOALAP					UK BioBank				
Cluster	Variable List	Coefficients of Compnents till sum(Explained) > 99.00				Explained	Coefficients of Compnents till sum(Explained) > 99.00				Explained
		1st	2nd	3rd	4th		1st	2nd	3rd	4th	
1: Length	Epis_NSpk	0.9998				99.9975	0.9999				99.9985
	Epis_Epi_min	0.0203				0.0025	0.0163				0.0015
2: Temporal	Epis_Step_RateMin	0.9993	0.0370			99.9612	0.9996				99.9805
	Epis_Gait_TimeSec	-0.0280	0.7553			0.0324	-0.0202				0.0166
	Epis_allStep_TimeSec	-0.0140	0.3776			0.0064	-0.0101				0.0030
	Epis_domStep_TimeSec	-0.0142	0.3922			0.0000	-0.0103				0.0000
	Epis_subStep_TimeSec	-0.0137	0.3631			0.0000	-0.0099				0.0000
3: Acceleation Magnitude and Velocity	Epis_allStep_AccForce	0.2416	0.1199	-0.1042		97.4815	0.2478	-0.1603	0.0987		96.8759
	Epis_domStep_AccForce	0.2624	-0.2455	-0.2154		0.9547	0.2607	-0.1816	-0.2916		1.5400
	Epis_subStep_AccForce	0.2209	0.4852	0.0069		0.7764	0.2348	-0.1389	0.4889		0.8113
	Epis_allGait_AccRange	0.5097	-0.2420	-0.1026		0.4623	0.4914	-0.0716	-0.3461		0.5148
	Epis_allStep_AccRange	0.4234	0.0815	-0.0124		0.1868	0.4218	-0.0283	0.0967		0.1591
	Epis_domStep_AccRange	0.4438	-0.4475	-0.1192		0.1031	0.4367	-0.0465	-0.4063		0.0699
	Epis_subStep_AccRange	0.4029	0.6104	0.0944		0.0351	0.4068	-0.0100	0.5997		0.0290
	Epis_Gait_Velocity	0.1213	-0.1395	0.7194		0.0000	0.1514	0.7217	-0.0011		0.0000
	Epis_allStep_Velocity	0.0603	-0.0702	0.3610		0.0000	0.0754	0.3617	-0.0008		0.0000
	Epis_domStep_Velocity	0.0634	-0.1641	0.3725		0.0000	0.0767	0.3772	-0.0794		0.0000
	Epis_subStep_Velocity	0.0571	0.0238	0.3496		0.0000	0.0741	0.3462	0.0777		0.0000
4: Symmerty	Epis_Symmetry_TimeSec	0.0118	-0.0529	0.2647	0.9628	85.6673	0.0268	0.8915	0.2844	-0.3515	76.8697
	Epis_Symmetry_AccForce	0.5845	0.7986	0.1437	-0.0028	10.8747	0.6005	-0.1988	0.7587	0.1555	14.2460
	Epis_Symmetry_AccRange	0.8005	-0.5390	-0.2601	0.0321	3.1975	0.7900	0.0593	-0.5600	-0.2424	7.3931
	Epis_Symmetry_Velocity	0.1320	-0.2627	0.9174	-0.2682	0.2605	0.1207	0.4027	-0.1727	0.8908	1.4911
5: Variance	Epis_Variance_TimeSec	0.0006	-0.0386	0.2268	0.9732	97.0447	0.0104	0.9005	0.2132		93.9181
	Epis_Variance_AccForce	0.5462	0.8342	0.0742	0.0155	1.9262	0.5483	-0.1235	0.8081		3.6485
	Epis_Variance_AccRange	0.8304	-0.5285	-0.1750	0.0194	0.9490	0.8259	0.0044	-0.5050		1.7004
	Epis_Variance_Velocity	0.1096	-0.1525	0.9552	-0.2287	0.0802	0.1311	0.4170	-0.2157		0.7330

Table 13 - PCA on the KOALAP dataset

We can see that the length cluster was mostly informed by the number of spikes within an episode. This makes the decision to add the variable of the **number of steps per episode** to the analysis easy. The temporal cluster was mostly informed by the **step rate**, which we decided to include in the analysis. The second component was informed by the other variables, mostly by the gait cycle (stride) time. The variables of the acceleration magnitude and velocity cluster have a very balanced input on the first component. The focus here lies on the acceleration range, where the acceleration range of the gait cycle is having the largest impact although not by much. We first proposed to extract the component for data analysis but later limited our analysis to the **acceleration range of the gait cycles**. The further components of this cluster consist of the difference between the dominant and subsequent steps (2nd KOALAP Cluster and 3rd BioBank cluster), and the variance of velocity (3rd KOALAP Cluster and 2nd BioBank cluster).

Making a decision about the symmetry cluster proved more difficult. The **symmetry of acceleration range** has the largest influence on the first component and we included it in our analysis. The second component (3rd for UK BioBank) reflects the difference between the symmetry of the acceleration force and range. Since the first component is already influenced by acceleration force, we decided to focus more on the fourth component (2nd for UK BioBank), in order to also consider the temporal symmetry of the gait. The fourth component of **symmetry of step time** is both present in the KOALAP PCA and the UK BioBank PCA. Since the dataset of the UK BioBank is much larger than the KOALAP dataset, we weighted our decision more towards the former.

At last, we looked at the variance cluster, whose component output is nearly identical to the PCA done on the symmetry data. The difference relies mostly on how much of the first component can explain the variance between the variance variables. We, therefore, decided to only look at the **variance of the acceleration range**.

We take these six identified variables to the analysis of the KOALAP and UK BioBank datasets.

6.6. Discussion

We identified the following parameters as suitable for walking behaviour analysis: number of steps within a gait episode, acceleration range of the gait cycle, step rate, symmetry of the acceleration range and the step time and the variance of the acceleration range. These six identified variables will be used in the analysis of the KOALAP and UK BioBank datasets.

We handled the housekeeping better in the SPE than in the GDA, mostly due to an increase in experience. However, we are still not quite satisfied with the formatting of our output data. The Matlab structure enabled us to have a better overview and clear labelling of the step parameters, which also made saving within a Matlab file easier. However, it is certainly not an accessible format from an outsider's perspective and is even hostile to people not familiar with Matlab. In general, both algorithms' output data need to be reformatted into binary file structures if we would want to share them with people outside of the research group. Another poor formatting choice that would need to be corrected is the non-standard way of labelling weekdays. Overall, aspects that were repeated in the SPE and the GDA should be combined into one code, with the option to save outputs of interest, such as dynamic data of the identified gait episodes, z-vectors, step locations and amplitude, associated step parameter values, etc.

In hindsight, we can also see some improvement opportunities for the SPE. The function of creating the step and gait cycle starting and endpoints could be more useful integrated into the hill-valley violation function since it was already performed there at the start and end of the episode. It also might be better to restrict the AC sliding window to only looking for frequencies in the range already identified by the GDA AC frequency. Furthermore, we tried, but in the end decided against, running the sliding AC on all three axes. It could be a helpful indicator of quality and to evaluate the identification of the dominant axis, since the walking behaviour tends to be represented on all three axes (or at least two of them) with very similar frequencies and with differing AC strengths. A further potential extension of the algorithm is the inclusion of spatial parameters, which we did not identify within our algorithm, even though they are often reported in the literature. Another critique of handling the data is the counterintuitive values of the step symmetry, where larger positive and negative values indicate more asymmetric behaviour, while values closer to 0 indicate more symmetric behaviour. In addition, we also experimented with potential noise reduction on the step data, using filters to smooth out spikes that were suppressed by noise and therefore most likely had their acceleration range depressed and the spike location displaced. However, our terms turned out to be processing time-consuming, developing time-consuming or not very accurate and were therefore discarded. Nevertheless, one filter that we applied can still be seen in the SPE code, albeit in a disabled form.

While our algorithm ended up very optimised for the SRDS, looking at the data identified from the KOALAP patients still sometimes contained identified gait episodes that were less qualitative than what we would wish for when analysing step parameters. We spent a lot of time looking for a method to exclude subpar episodes but had to concede defeat. It leaves the window open for improvements, and also illustrates the problem of both the GDA and SPEs. Both algorithms were nearly completely developed using the dataset of only one participant and were, therefore, most likely optimised for the SRDS and do not account for problems in gait identification caused by the walking behaviour of other people and people with RMDs in particular. Collecting labelled data from more participants and RMD patients and testing the performance of the algorithm on them would be highly advised.

For future analysis, classifying the gait patterns identified for each participant, for example by using a Switching AR, might yield interesting results. Analysing step parameter differences between gait behaviours identified within the same clusters could result in better identification of a reduction in the functioning of RMD patients. The recorded information on dynamic data orientation and dominant axis could also be used for this purpose. At the moment, the differences between gait episodes are more likely due to a difference in walking behaviours, than a difference in walking performance. Chapter 4.4.2 has already illustrated that the main difference between gait episodes is most likely caused by voluntary behaviour rather than a limiting cause by their disease. However, given a large enough dataset, we might still find associations between patients' step parameters and pain reports, or between RMD patients and controls.

The SPE algorithm performance on our SRDS was mostly very successful and only excluded less than 3 min of data that we would have considered walking behaviour. According to our hand labels, only 2.79% of the data was misidentified at the end of both algorithms. However, the translatability of this accuracy onto other datasets needs further exploration.

7. Knee Osteoarthritis: Linking Activity and Pain

7.1. Abstract

The Knee Osteoarthritis: Linking Activity and Pain (KOALAP) study aims to investigate participants' activity and pain levels. This is done by actively requesting participants to fill out daily pain level scores and by continuously passively recording sensory data using smartwatches. The pain was scored with six distinct pain questions: morning and afternoon pain, pain during aggravating and important activities, daily function and quality of life.

First, we made interesting observations about wrangling data extracted from a consumer device. Our proposal advises researchers to contact software developers when using consumer devices for research purposes. The technicians working on the programs have different priorities than researchers and will prioritise energy efficiency and reduction of memory usage over temporal accuracy. Awareness of potential data wrangling issues at the project planning stage, such as the possible occurrence of non-unique data points, can potentially improve the data collection process by ensuring a more ensuring thorough and expedient raw data collection. Furthermore, we looked at the wear time compliance of the participants, the results were published in (Beukenhorst, et al., 2019).

When applying the Aston Gait Detection Algorithm (AGDA) to the KOALAP dataset, we were unable to relate patients' step parameter values consistently to their pain scores. Nevertheless, we did find occurrences of significant correlations with some pain scores depending on the participant. We conclude that the step parameter we extracted could help track patients' performance over time. However, the effectiveness and the parameter used would need to be adjusted to the individual patient.

7.2. Introduction

As already discussed in the literature review, most OA studies investigate the effect of physical activity as measured in energy levels and step counts. We aim to go beyond the classification of low, moderate and vigorous activity to look directly at parameters derived from walking behaviours. The use of accelerometers to investigate gait in patients with OA opens up the potential to track physical function variations over time and to relate it back to pain reports. We hope to find parameters that enable us to identify impaired physical functions. Most likely we will observe decreases in walking speed, acceleration amplitude and symmetry, and increases in gait variability. Looking at patients with OA we would furthermore assume a decrease in walking engagement and performance due to fatigue throughout the day. Potentially we could even find patterns of correlation between step parameters and pain. A decrease in walking episodes and a decrease in physical functions could be associated with an increase in self-reported pain.

A secondary aim is to evaluate the use of consumer devices in passively monitoring the physical functioning of patients with Osteoarthritis (OA) both in clinical and research settings. Consequently, the data quality and user experience will be judged. The patient engagement with the KOALAP app was reported in (Beukenhorst, et al., 2019). They not only looked at the wear-

time that we extracted for their paper but also at participants' interviews about their experience with the smartwatches. One mayor complained was the time it took for the smartwatch to charge and GPS tracking had to be disabled consequently. Overall most participants were motivated by the self-tracking aspects of the app.

7.3. Methods – Data Collection

7.3.1. Participants Selection and Study Progression Overview

A selection of participants from the CLOUDY study was asked to also participate in the KOALAP study, given that they:

- Were 50 years or older
- Were living in the Greater Manchester area or able to travel there
- Had a self-reported diagnosis of Knee Osteoarthritis
- Possessed an Apple or Android smartphone

Individuals who showed interest in participating were invited to the University of Manchester, where a research team with help from Google equipped the participant with a smartwatch and installed the KOALAP phone app. Furthermore, they were asked to fill out a short paper questionnaire, take part in an approximately 1-h interview and to fill out a KOOS questionnaire to decide if the participant was eligible to take part in the study. Since the KOOS was filled out on the smartwatch, the visit was also used to determine if the participants were able to use the device adequately. Each participant recruited for the study was given a numeric value as a participant identification (ID). Since not all recruited participants ended up joining the study, the Participant IDs consisted of 26 numbers between 3 and 42. A total of 26 participants (13 males, 13 females) took part in the study between September 2017 and February 2018. The minimum age was 50 years, with a mean and STD of 64.2 ±8.8 years in the participant group.

The information recorded from the participants included physical characteristics (height, weight and leg length), information about their condition (self-reporting a diagnosis of knee OA, signs of disease, self-judged most impaired activity, use of walking aids and reporting joint replacements) and exercise preference. Regarding the premise of the study, participants were asked to give feedback on their motivation to participate and how self-monitoring activity impacted their health management.

Participants that ended up joining the study were asked to wear the watch from when they woke up until they went back to bed, with the instructions to “take the watch off as close to bedtime as possible” and to charge it overnight. In addition, they were asked to fill out daily, weekly and monthly questionnaires on the smartwatch. Every day the participants answered questions labelled “Morning Pain”, “Afternoon Pain”, “Pain when …” and “Daily Functioning”. Once a week “Pain interference with…” and “Quality of Life” was asked. On a monthly basis, the participant filled out the pain and activities of daily living subscales of the Knee Osteoarthritis Outcome Score (KOOS) questionnaire. The official length of the study was 3 months, with the start dates lying between the

13th of and 23rd of September. Start dates and events that disrupted the data collection (e.g. technical problems, holidays or dropouts) were recorded and are summarised in Table 14. Throughout the study, the watch recorded accelerometer, gyroscope, magnetometer, barometer, heart rate and step count data during wear time. Between 10 and 118 days of sensory data is present for each participant, with a mean and STD of 73 ± 23 days.

Participation time / Breaks	Day ranges / Days lost	N	Participants IDs
Total Participants		26	
Official start and end time	13/Sep/2017 - 13/Dec/2017	4	13, 14, 15, 19
	14/Sep/2017 - 14/Dec/2017	7	3, 7, 8, 9, 21, 25, 27
	15/Sep/2017 - 15/Dec/2017	10	5, 16, 17, 18, 20, 23, 26, 28, 34, 35,
	21/Sep/2017 - 21/Dec/2017	2	10, 30
	23/Sep/2017 - 23/Dec/2017	3	37, 38, 42
Total Participants with official breaks	180 days	9	3, 5, 9, 10, 15, 17, 20, 34, 35
Total Participants without official breaks	2186 days	17	7, 8, 13, 14, 16, 18, 19, 21, 23, 25, 26, 27, 28, 30, 37, 38, 42
Drop out	32 days	1	34
Technical difficulties	27 days, 24 days, 2 days	3	5, 9, 35
Holidays	31 days, 14 days, 7 days	3	10, 15, 17
Drop out & Technical difficulties (& Holidays)	21 days	1	20
Technical difficulties & Holidays	22 days	1	3

Table 14 - Official Participation Time and Breaks of Participants

At the end of the study, participants were invited for a second visit to the university, where they filled out a final 30-min questionnaire, returned the smartwatch and took part in a second interview. The participants' travel expenses were covered for up to £30 and they were compensated with a £10 Love2shop voucher after each interview.

7.3.2. Material

The Huawei Watch 2 smartwatch was loaned by Google for the duration of the study. The dimensions of the watch are 48.9 mm x 45 mm x 12.6 mm with a weight of approximately 40 g (excluding the watch strap, 57 g with the watch strap). The operating system is running on Android Wear 2.0, which supports the Android 4.4+ and OS 9.0+ mobile OS. It has a memory of 4 GB flash + 768 MB random access memory (RAM) and the average battery capacity is 420 mAh. The watch contains a 6-axis accelerometer and gyroscope sensor, a 3-axis compass, a heart rate sensor (photoplethysmogram), a barometer, a capacitive sensor and an ambient light sensor. Water resistance was described as 1.5 m maximum depth for up to 30 min (Huawei, n.d.).

All data collection was facilitated through the KOALAP app, which was developed in collaboration with the Google Android Wear team and the KOALAP research team. The watch recorded sensory data while it was worn and saved them in an internal SQLite database. Full data storage (maximum ~ 4 days of data) or an empty battery would cause the watch to stop saving

data. The Wi-Fi connection was used by the watch to upload the collected data daily onto the Google Cloud database.

The following paragraphs will contain a general description of the sensors in the smartwatch used to record data for the KOALAP study. An in-depth description of the accelerometers can be found in Chapter 1.3 and will not be repeated here.

Gyroscopes can detect the angular velocity of an object when it rotates around an axis, but they are not sensitive to linear movements or vibration. Angular velocity is measured in revolutions per sec or degrees per sec (°/sec). A tri-axial MEMS gyroscope, as found in smartphones and watches, detects rotation around the x- (called “pitch”), y- (“roll”) and z-axis (“yaw”). Each axis has a small rotating gyroscope sensor that has a mass attached to it with springs. When an outside rotation of the object occurs in alignment with the rotation of the sensor, the attached mass will pull towards the centre of the sensor. However, if the rotation of the object is against the rotation of the sensor, the mass pulls away. The shift of the mass is measured and the resulting physical displacement caused by the Coriolis force is measured through the MEMS sensors as the angular velocity (Esfandyari, De Nuccio, & Gang, 2010).

Magnetometers are used to identify the true orientation of a device with respect to an ambient magnetic field, most usually the Earth’s magnetic field. They most commonly work using the Hall effect by measuring the current flow through a conductive plate. In this way, a magnetic field disrupts the current flow in accordance with its strength and direction. This causes the electrons and positive poles on the opposite side of the plate to move within the conductor, resulting in a difference in voltage that can be measured with a voltage meter (Nedelkovski, 2015).

7.3.3. Data Collection

Participants were notified by the watches’ vibration when it became time to fill out the questionnaires. An overview of their interface can be seen in Figure 65.

Figure 65 - **KOALAP app interface.** Left: Notification for an active questionnaire on the home screen. Centre: Home screen with information about battery usage, heart rate, step count of the day and active questionnaire reminder. Right: Screen with the “aggravating activities” questionnaire.

Knee Injury and Osteoarthritis Outcome Score

On a monthly basis, the participant filled out the KOOS questionnaire, a 5-point Likert scale. While the full KOOS contains 5 subscales, only the “pain” and “activities of daily living” subscales

were requested from the participants. The questionnaire was found to be reliable, valid and responsive when evaluating surgery and therapy outcomes (Roos, Roos, Lohmander, Ekdahl, & Beynnon, 1998). The participants were reminded to fill out the KOOS at 14, 44 and 74 days after the start of the study and had one week's time before the questionnaire expired.

Pain Levels

All of the pain level questionnaires consisted of a single question answered on a 10-point Numeric Rating Scale with 10 indicating the worst pain.

Participants had to rate their "morning" pain during lunchtime (between 12:22 and 16:22), and their "afternoon" pain in the evening (between 18:22 and 22:22). Additionally, two further questions were asked in the evening (between 5 pm and midnight), making the participants rate their "daily function" and their pain during "aggravating activities". They had to select the activity they judged most aggravating on the first visit from the following activities: sitting for long periods; sitting to standing; squatting or bending or kneeling; standing, turning or twisting; walking or walking up stairs or inclines.

Once a week the participant rated their "Quality of Life" and how much their pain interfered with an activity the participant judged to be most "important" to them, as reported during the first visit. Participants could choose from the following activities: doing household tasks, getting washed and dressed, playing sports, walking, working effectively and socializing.

Sensory Data

Throughout the study, the watch recorded accelerometer, gyroscope, magnetometer, barometer, heart rate and step count data during wear time. While the plan was to also record distance travelled via GPS data, this measure had to be dropped since keeping the GPS active drained the battery too fast.

Data collection occurred while the participant wore the watch. If the watch was taken off, data collection stopped. The planned data collection frequency was 50 Hz for the accelerometer, gyroscope and magnetometer data, 1 Hz for the heart rate data and one measurement per minute for the barometer data.

Step data was calculated by an algorithm created by Google and the exact algorithm is unknown to the research team. Therefore step count data will at most only be used for validation purposes by comparing it to step count data extracted through algorithms produced by the research group.

7.4. Introduction - Data Wrangling

Kandel et al (2011) define *data wrangling* as "a process of iterative data exploration and transformation that enables analysis". It has the goal to make data "useable", "credible" and "useful". They furthermore comment on how the data wrangling process can be described as the "most tedious" part of data analysis. The process of Data Wrangling takes raw output data and aims to transform it into a format that can be used in common analysis packages. This data processing step makes the data more accessible and allows the data analysis to examine the data

on quality issues. Therefore, missing and duplicated data has to be identified and it has to be evaluated how feasibly realistic the collected data is, or if there are outliers that clearly indicate a problem in the data collection process (Kandel, et al., 2011).

We will describe the steps involved in data extraction, present the code we used to extract the data and present a manually conducted data extraction on the example of the 1st and 2nd January 2018. Situations that complicated the data extractions and prolonged steps in the data analysis for the Research Team will be highlighted. The aim of this chapter is to describe how time-consuming and taxing the data wrangling process can be when the person handling the data was not the one developing the data collection process.

In data wrangling, the obtained raw data is put through a process in which it is explored and transformed into a suitable format for data analysis. Furthermore, it entails identifying quality issues with the data. However, developing a process for data wrangling can be tedious when the dataset is large or unconventionally formatted.

7.5. Methods – Data Wrangling

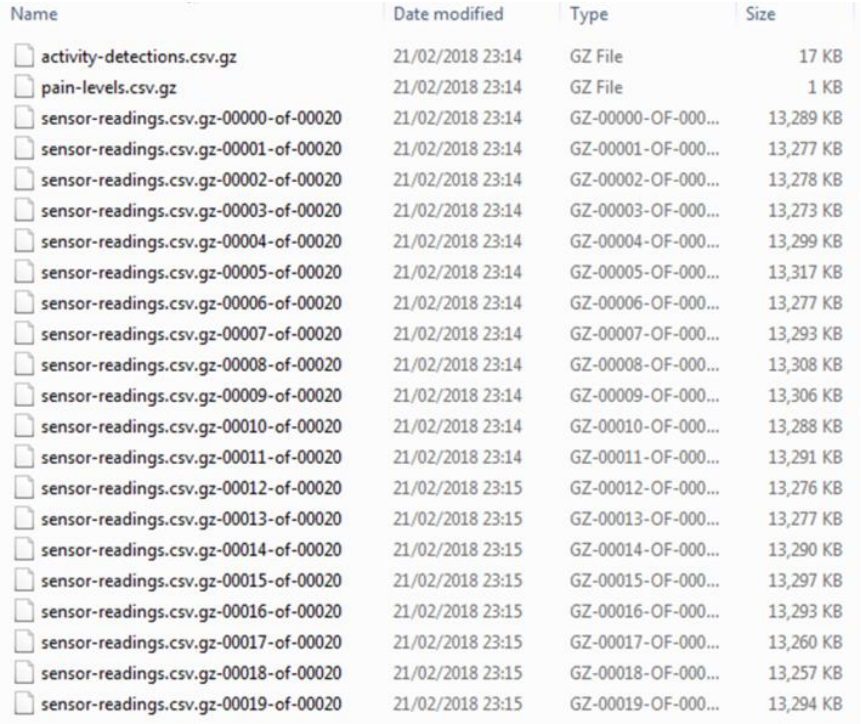
7.5.1. Data Analysis Software

Data wrangling was mostly done in MATLAB R2017a and through the use of Microsoft Excel 2010. KOOS data wrangling was done completely in Excel, while pain level data was facilitated through both Excel and Matlab and sensory data was handled solely in Matlab. Due to the size of the sensory data (about 900 gigabytes (GB)), data wrangling was done in an Interactive Computational Shared Facility (iCSF) also known as Incline. Incline enables computationally-intensive work. On the service, commands can be executed using a larger memory (256 GB) or 2 terabytes (TB) in iCSF vs an 8 GB RAM than on a standard computer. This makes data handling faster and larger chunks of data can be processed at once.

7.5.2. Obtaining the Raw Data

The data was loaded from the Google Cloud database between the 21st of February and the 28th of March 2017 onto a secure university database. The data was anonymised, therefore, participants cannot be personally identified by the research team. Sensitive data is put under restricted, password-protected access. All data processing will be in line with the Data Protection Act, 1998 and in accordance with Medical Research Council guidelines for ethical research. The data will be archived for 10 years.

With no prior input when the data recording methods were planned, understanding the data layout was the first obstacle. We downloaded a total of 166 folders from the Google servers. One folder contained the KOOS questionnaires while the other folders are labelled with the dates from the 09th of September 2017 to the 20th of February 2018. Each of the date-labelled folders contains zipped .csv files (see Figure 66). Within each folder, three types of files are present: a “pain-levels.csv” file, an “activity-detections.csv” file and twenty “sensor-readings.csv” files. The “activity-detections.csv” file will be ignored since they contain claims about the participant's behaviour at any given time that data was present that are the output of an undisclosed algorithm.



Name	Date modified	Type	Size
activity-detections.csv.gz	21/02/2018 23:14	GZ File	17 KB
pain-levels.csv.gz	21/02/2018 23:14	GZ File	1 KB
sensor-readings.csv.gz-00000-of-00020	21/02/2018 23:14	GZ-00000-OF-000...	13,289 KB
sensor-readings.csv.gz-00001-of-00020	21/02/2018 23:14	GZ-00001-OF-000...	13,277 KB
sensor-readings.csv.gz-00002-of-00020	21/02/2018 23:14	GZ-00002-OF-000...	13,278 KB
sensor-readings.csv.gz-00003-of-00020	21/02/2018 23:14	GZ-00003-OF-000...	13,273 KB
sensor-readings.csv.gz-00004-of-00020	21/02/2018 23:14	GZ-00004-OF-000...	13,299 KB
sensor-readings.csv.gz-00005-of-00020	21/02/2018 23:14	GZ-00005-OF-000...	13,317 KB
sensor-readings.csv.gz-00006-of-00020	21/02/2018 23:14	GZ-00006-OF-000...	13,277 KB
sensor-readings.csv.gz-00007-of-00020	21/02/2018 23:14	GZ-00007-OF-000...	13,293 KB
sensor-readings.csv.gz-00008-of-00020	21/02/2018 23:14	GZ-00008-OF-000...	13,308 KB
sensor-readings.csv.gz-00009-of-00020	21/02/2018 23:14	GZ-00009-OF-000...	13,306 KB
sensor-readings.csv.gz-00010-of-00020	21/02/2018 23:14	GZ-00010-OF-000...	13,288 KB
sensor-readings.csv.gz-00011-of-00020	21/02/2018 23:14	GZ-00011-OF-000...	13,291 KB
sensor-readings.csv.gz-00012-of-00020	21/02/2018 23:15	GZ-00012-OF-000...	13,276 KB
sensor-readings.csv.gz-00013-of-00020	21/02/2018 23:15	GZ-00013-OF-000...	13,277 KB
sensor-readings.csv.gz-00014-of-00020	21/02/2018 23:15	GZ-00014-OF-000...	13,290 KB
sensor-readings.csv.gz-00015-of-00020	21/02/2018 23:15	GZ-00015-OF-000...	13,297 KB
sensor-readings.csv.gz-00016-of-00020	21/02/2018 23:15	GZ-00016-OF-000...	13,293 KB
sensor-readings.csv.gz-00017-of-00020	21/02/2018 23:15	GZ-00017-OF-000...	13,260 KB
sensor-readings.csv.gz-00018-of-00020	21/02/2018 23:15	GZ-00018-OF-000...	13,257 KB
sensor-readings.csv.gz-00019-of-00020	21/02/2018 23:15	GZ-00019-OF-000...	13,294 KB

Figure 66 - ***Content of Folder "2018-01-01"***

The first task was to identify folders without content. While folder “2017-09-09” contained all the files present in the other folders, the files themselves only contained headers. Going through the files and identifying “sensor-readings.csv” files that exceeded 2 kilobytes (KB) in unzipped form gave an estimate of the number of days that contained data. Smartwatches collected data between the 12th Sep 2017 and 26th January 2018, but also on the 10th Sep and the 11th and 13th of February which shows that some participants clearly wore the watch even after their part in the study officially ended.

Since data extraction of the sensory data was completely handled through Matlab codes, the above-mentioned empty data files were a source of confusion. When we were unable to load any data besides the header, we took the data with us to the research meeting at Aston University (Birmingham) to solve the data unzipping problem. There we had to realise that no actual data came with us on our journey, wasting the first opportunity to examine the data in a team. This is an example of the simple yet time-expensive problems one can encounter during data wrangling. Small instances such as this can add up and causes the data wrangling process to become both time-consuming and frustrating.

All files had the same foundational structure. Each data observation was assigned a row with the first column containing the smartwatch ID and the second column containing a timestamp to indicate the time and date a questionnaire was filled out or a sensory measurement was taken. Therefore, for each data file, the smartwatch ID was translated to the patient and the date and time were extracted from the timestamp.

Each smartwatch provided by Google has its own randomly generated User ID, which consists of 32 randomly generates letters and number characters. It was recorded that the watch of participant 5 stopped working and had to be replaced, which lead this participant to have two

separate smartwatch IDs. The two smartwatch IDs were combined into one Participant ID. Afterwards, the date and time were extracted from the timestamp. While extracting only the date was sufficient for the questionnaire files, the sensory data needed the timestamp to be preserved accurately to the msec. In most cases, the timestamps were transformed into a numeric value. For example, the timestamp “2017-09-12T15:35:52.264Z” would receive the value 736950.649910463. The integer 1 would represent the date 1st January 0000, hence, the 12th September 2017 would be assigned the integer 736950. The fraction 0.649910463 indicates the time 3:35 pm and 52.264 sec (0.041666667 * 15 h + 0.000694444 * 35 min + 0.000011574 * 52 sec + 0.000000012 * 264 msec).

The next section will first describe the pain data extraction of KOOS and pain level data and then the sensory data extraction, even though the actual data extraction occurred in the opposite order.

7.5.3. Extracting Questionnaire Data

Knee injury and Osteoarthritis Outcome Score

The KOOS folder contained three files labelled “koos.cvs”, “koos_1.cvs” and “koos_2.cvs”. While the third file appeared to contain KOOS scores from all participants at all times, both the “koos.cvs” and “koos_1.cvs” files did not include all participants and only data collected on the 1st of October. We combined the three files and removed repetitions. Rows were removed when the Participant ID, timestamp and all individuals' KOOS scores were equal. This reduced the total number of filled out questionnaires from 113 to 69 (an average of 2.7 questionnaires per participant). An overview of the number of KOOS questionnaires filled out by the participants and the time points of questionnaires filled out are shown in Figure 67. Participant 16 filled out the questionnaire twice on the 1st of November.

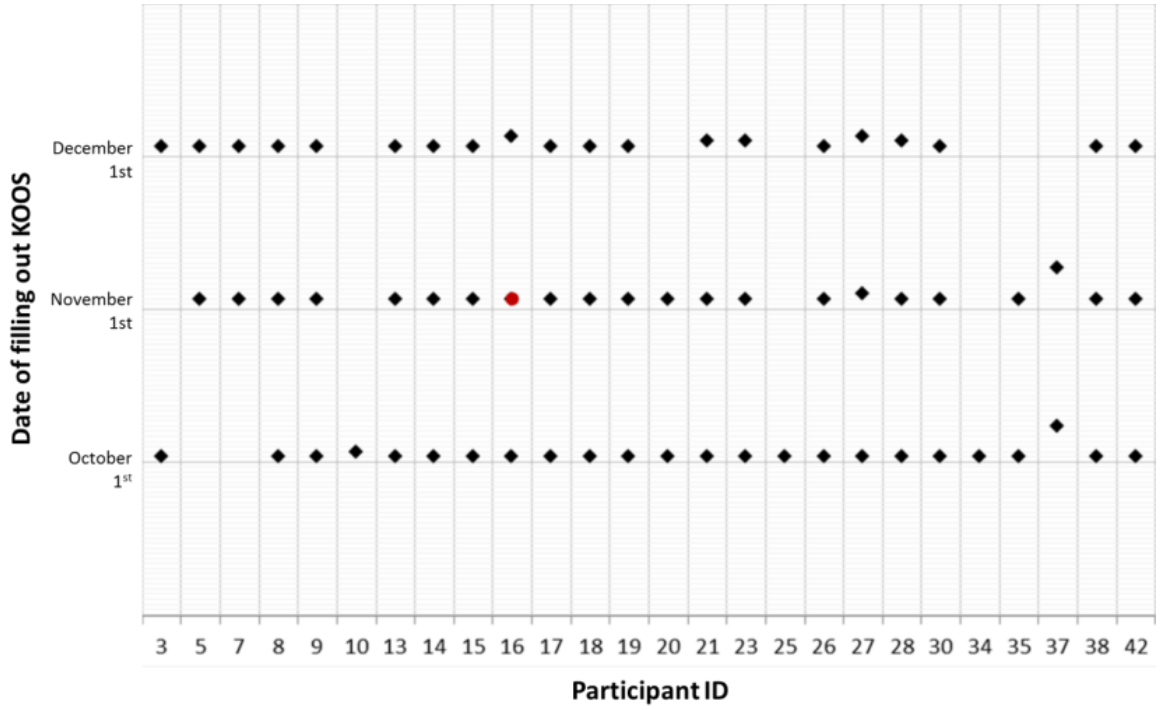
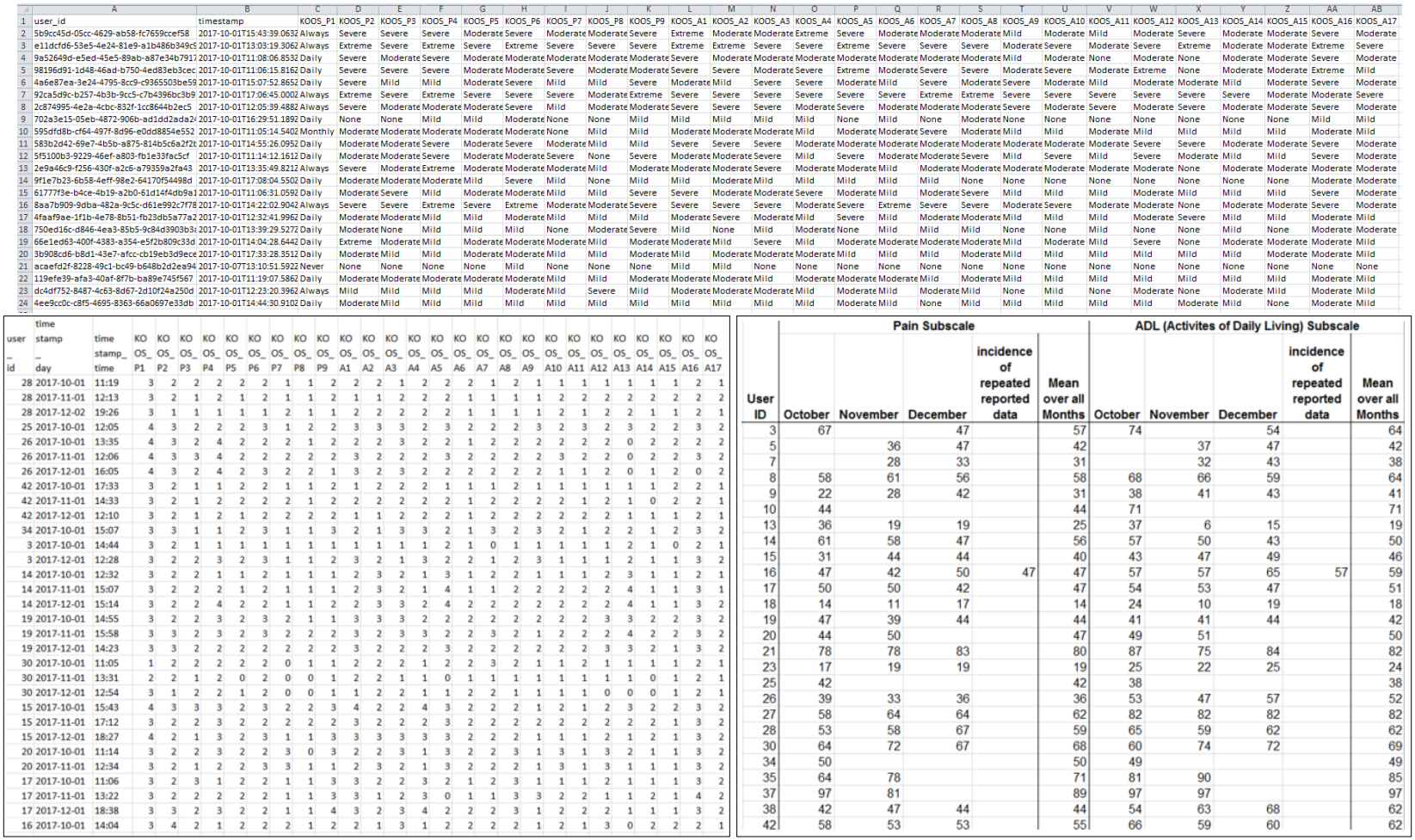


*Figure 67 – **Time points the KOOS questionnaire was filled out by each Participant**. Black Diamonds: Time point of each questionnaire, most of them on the 1st of each month (Outliers: 2nd Oct, 7th Oct, 2nd Nov, 11th Nov, 2nd & Dec, 3rd Dec); Red Dot: the questionnaire was filled out twice that day.*

Figure 68 shows both the raw data and extracted data after data wrangling for the KOOS questionnaires. Since answers to the 5 Likert boxes questionnaire were saved in a text format in the raw data ("None", "Mild", "Moderate", "Severe" and "Extreme"), we had to transform them into numeric values from 0 ("None") to 4 ("Extreme"). The data was then sorted by participant ID and date (see Figure 68, below, left). Furthermore, KOOS scores (see Figure 68, below, right) were calculated for each subscale using the equation (KOOS, 2012):

$$\text{Score} = 100 - \frac{\text{Mean Score(Subscale Socres) x } 100}{4}$$

A score of 0 indicates the most extreme symptoms, while a score of 100 would mean that the participant did not experience any symptoms (KOOS, 2012). Participants were only given the "Pain" and "Activities of Daily Living (ADL)" subscales of the KOOS. Scores for the "Symptoms", "Sport and Recreation Function" and "knee-related Quality of Life (QoL)" subscales were not taken.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB
1	user_id	timestamp	KOOS_P1	KOOS_P2	KOOS_P3	KOOS_P4	KOOS_P5	KOOS_P6	KOOS_P7	KOOS_P8	KOOS_P9	KOOS_A1	KOOS_A2	KOOS_A3	KOOS_A4	KOOS_A5	KOOS_A6	KOOS_A7	KOOS_A8	KOOS_A9	KOOS_A10	KOOS_A11	KOOS_A12	KOOS_A13	KOOS_A14	KOOS_A15	KOOS_A16	KOOS_A17
2	5b9cc45d-05cc-4629-ab58-fc7659ccef58	2017-10-01T15:43:39.063Z	Always	Severe	Severe	Severe	Moderate	Severe	Moderate	Moderate	Severe	Extreme	Moderate	Moderate	Extreme	Severe	Moderate	Moderate	Moderate	Mild	Moderate	Mild	Moderate	Severe	Moderate	Moderate	Severe	Moderate
3	e11dcfd6-53e5-4e24-81e9-a1b486b349c5	2017-10-01T13:03:19.306Z	Always	Extreme	Severe	Extreme	Severe	Extreme	Severe	Severe	Severe	Extreme	Extreme	Severe	Severe	Extreme	Severe	Severe	Severe	Moderate	Severe	Moderate	Severe	Extreme	Moderate	Moderate	Extreme	Severe
4	9a52649d-e5ed-45e5-89ab-a87e34b7917	2017-10-01T11:08:06.853Z	Daily	Severe	Moderate	Severe	Moderate	Moderate	Moderate	Moderate	Moderate	Severe	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Mild	Moderate	None	Moderate	None	Moderate	Moderate	Severe	Moderate
5	98196d91-1d48-46ad-b750-4ed83eb3cec	2017-10-01T11:06:15.816Z	Daily	Severe	Severe	Severe	Moderate	Moderate	Severe	Moderate	Moderate	Severe	Severe	Moderate	Severe	Extreme	Moderate	Severe	Severe	Moderate	Severe	Moderate	Extreme	None	Moderate	Moderate	Extreme	Mild
6	4a6e87ea-3e24-4795-8cc9-c9365503be59	2017-10-01T15:07:52.865Z	Daily	Severe	Mild	Mild	Moderate	Severe	Mild	Mild	Severe	Moderate	Mild	Severe	Severe	Moderate	Mild	Severe	Moderate	Severe	Moderate	Mild	Moderate	Moderate	Mild	Moderate	Severe	Moderate
7	92ca5d9c-b257-4b3b-9cc5-c7b4396bc3b9	2017-10-01T17:06:45.000Z	Always	Extreme	Severe	Extreme	Severe	Severe	Severe	Moderate	Extreme	Severe	Severe	Severe	Severe	Severe	Severe	Extreme	Extreme	Severe	Severe	Severe	Severe	Severe	Severe	Severe	Moderate	Severe
8	2c874995-4e2a-4cbc-832f-1cc8644b2ec5	2017-10-01T12:05:39.488Z	Always	Severe	Moderate	Moderate	Moderate	Severe	Mild	Moderate	Moderate	Severe	Severe	Severe	Moderate	Severe	Moderate	Moderate	Moderate	Severe	Moderate	Severe	Moderate	Severe	Moderate	Moderate	Severe	Moderate
9	702a3e15-05eb-4872-906b-ad1dd2ada24	2017-10-01T16:29:51.189Z	Daily	None	None	Mild	Mild	Moderate	None	None	Mild	Mild	Mild	Mild	Mild	None	Mild	None	Moderate	None	None	None	None	None	None	Mild	Mild
10	595dfd8b-cf64-497f-8d96-e0dd8854e552	2017-10-01T11:05:14.540Z	Monthly	Moderate	Moderate	Moderate	Moderate	Moderate	None	Mild	Mild	Moderate	Moderate	Moderate	Mild	Moderate	Moderate	Severe	Moderate	Mild	Mild	Moderate	Mild	Mild	Mild	Mild	Moderate	Mild
11	583b2d42-69e7-4b5b-a875-814b5c6a2f2c	2017-10-01T14:55:26.095Z	Daily	Moderate	Moderate	Severe	Moderate	Severe	Moderate	Mild	Mild	Severe	Severe	Severe	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Severe	Severe	Moderate	Moderate	Severe	Moderate
12	5f5100b3-9229-46ef-a803-fb1e33fac5cf	2017-10-01T11:14:12.161Z	Daily	Moderate	Moderate	Severe	Moderate	Moderate	Severe	None	Severe	Moderate	Moderate	Severe	Mild	Severe	Moderate	Moderate	Severe	Mild	Severe	Mild	Severe	Moderate	Mild	Mild	Severe	Moderate
13	2e9a46c9-f256-430f-a2c6-a79359a2fa43	2017-10-01T13:35:49.821Z	Always	Severe	Moderate	Extreme	Moderate	Moderate	Moderate	Mild	Moderate	Moderate	Severe	Moderate	Moderate	Mild	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	Moderate	None	Moderate	Moderate	Moderate	Moderate
14	9f1e7b23-6b58-4eff-98e2-64170f54498d	2017-10-01T17:08:04.550Z	Daily	Moderate	Moderate	Moderate	Mild	Severe	Mild	None	Mild	Mild	Moderate	Mild	Mild	Mild	Mild	Mild	None	None	None	None	None	None	None	None	Moderate	Moderate
15	61777f3e-b4ce-4b19-a2b0-61d14f4db9a1	2017-10-01T11:06:31.059Z	Daily	Moderate	Severe	Mild	Moderate	Moderate	Mild	Mild	Severe	Severe	Moderate	Moderate	Severe	Moderate	Mild	Moderate	Severe	Mild	Mild	Mild	Moderate	Mild	Mild	Mild	Severe	Moderate
16	8aa7b909-9dba-482a-9c5c-d61e992c7f78	2017-10-01T14:22:02.904Z	Always	Severe	Severe	Extreme	Severe	Extreme	Moderate	Moderate	Severe	Severe	Severe	Severe	Moderate	Severe	Extreme	Severe	Severe	Moderate	Severe	Moderate	Moderate	None	Moderate	Moderate	Severe	Moderate
17	4faaf9ae-1f1b-4e78-8b51-fb23db5a77a2	2017-10-01T12:32:41.996Z	Daily	Moderate	Moderate	Mild	Mild	Moderate	Mild	Mild	Mild	Moderate	Severe	Moderate	Mild	Severe	Mild	Moderate	Moderate	Mild	Mild	Mild	Moderate	Severe	Mild	Mild	Moderate	Mild
18	750ed16c-d846-4ea3-85b5-9c84d3903b3a	2017-10-01T13:39:29.527Z	Daily	Moderate	None	Mild	Mild	Mild	None	Moderate	Severe	Mild	None	Mild	Moderate	None	Mild	Mild	Mild	None	None	Mild	None	None	Mild	None	Moderate	Moderate
19	66e1ed63-400f-4383-a354-e5f2b809c33d	2017-10-01T14:04:28.644Z	Daily	Extreme	Moderate	Mild	Moderate	Moderate	Moderate	Mild	Moderate	Moderate	Mild	Severe	Mild	Moderate	Moderate	Moderate	Moderate	Mild	Moderate	Mild	Severe	None	Moderate	Moderate	Moderate	Mild
20	3b908cd6-b8d1-43e7-afcc-cb19eb3d9ece	2017-10-01T17:33:28.351Z	Daily	Moderate	Mild	Mild	Moderate	Moderate	Mild	Mild	Moderate	Mild	Moderate	Moderate	Moderate	Mild	Mild	Mild	Moderate	Mild	Mild	Mild	Mild	Mild	Mild	Mild	Moderate	Mild
21	acaefd2f-8228-49c1-bc49-b648b2d2ea94	2017-10-07T13:10:51.592Z	Never	None	None	None	None	Mild	None	None	None	Mild	Mild	None	None	None	None	None	None	None	None	None	None	None	None	None	None
22	119efe39-afa3-40af-8f7b-ba89e745f567	2017-10-01T11:19:07.586Z	Daily	Moderate	Moderate	Moderate	Moderate	Moderate	Mild	Mild	Moderate	Moderate	Moderate	Mild	Moderate	Moderate	Moderate	Mild	Moderate	Mild	Mild	Mild	Mild	Mild	Mild	Mild	Moderate	Mild
23	dc4df752-8487-4c63-8d67-2d10f24a250d	2017-10-01T12:23:20.396Z	Always	Mild	Mild	Mild	Mild	Moderate	Mild	Severe	Mild	Moderate	Moderate	Moderate	Mild	Moderate	Mild	Moderate	Mild	None	Mild	None	Moderate	None	Moderate	Mild	Moderate	Mild
24	4ee9cc0c-c8f5-4695-8363-66a0697e33db	2017-10-01T14:44:30.910Z	Daily	Moderate	Mild	Mild	Mild	Mild	Mild	Mild	Mild	Mild	Mild	Mild	Mild	Moderate	Mild	None	Mild	Mild	Mild	Mild	Mild	Moderate	Mild	None	Moderate	Mild

user_id	time stamp	time stamp_day	time	KOOS_P1	KOOS_P2	KOOS_P3	KOOS_P4	KOOS_P5	KOOS_P6	KOOS_P7	KOOS_P8	KOOS_P9	KOOS_A1	KOOS_A2	KOOS_A3	KOOS_A4	KOOS_A5	KOOS_A6	KOOS_A7	KOOS_A8	KOOS_A9	KOOS_A10	KOOS_A11	KOOS_A12	KOOS_A13	KOOS_A14	KOOS_A15	KOOS_A16	KOOS_A17
28	2017-10-01	11:19		3	2	2	2	2	2	1	1	2	2	2	1	2	2	1	2	1	1	1	1	1	1	1	2	1
28	2017-11-01	12:13		3	2	1	2	1	2	1	1	2	1	1	2	2	2	1	1	1	1	2	2	2	2	2	2	2
28	2017-12-02	19:26		3	1	1	1	1	1	2	1	1	2	2	2	2	2	1	1	1	1	2	1	2	2	1	1	2	1
25	2017-10-01	12:05		4	3	2	2	2	3	1	2	2	3	3	3	2	3	2	2	2	3	2	3	2	3	2	2	3	2
26	2017-10-01	13:35		4	3	2	4	2	2	2	1	2	2	2	3	2	2	1	2	2	2	2	2	2	0	2	2	2	2
26	2017-11-01	12:06		4	3	3	4	2	2	2	2	3	2	2	2	3	2	2	2	2	2	3	2	2	0	2	2	3	2
26	2017-12-01	16:05		4	3	2	4	2	3	2	2	1	3	2	3	2	2	2	2	2	2	1	1	2	0	1	2	0	2
42	2017-10-01	17:33		3	2	1	1	2	2	1	1	2	1	2	2	2	1	1	1	2	1	1	1	1	1	1	2	2	1
42	2017-11-01	14:33		3	2	1	2	2	2	1	2	2	2	2	2	2	1	2	2	2	2	1	2	1	0	2	2	2	1
42	2017-12-01	12:10		3	2	1	2	1	2	2	2	2	1	1	2	2	2	1	2	2	2	2	2	2	1	1	1	2	1
34	2017-10-01	15:07		3	3	1	1	2	3	1	1	3	2	1	3	3	2	1	3	2	3	2	1	2	2	1	2	3	2
3	2017-10-01	14:44		3	2	1	1	1	1	1	1	1	1	1	1	2	1	0	1	1	1	1	1	2	1	0	2	1	
3	2017-12-01	12:28		3	2	2	3	2	3	1	1	2	3	2	1	3	2	2	1	2	3	1	1	1	2	1	1	3	2
14	2017-10-01	12:32		3	2	2	1	1	2	1	1	2	3	2	1	3	1	2	2	1	1	1	2	3	1	1	2	1	
14	2017-11-01	15:07		3	2	2	2	1	2	1	1	1	2	3	2	1	4	1	1	2	2	2	2	2	4	1	1	3	1
14	2017-12-01	15:14		3	2	2	4	2	2	1	1	2	2	3	3	2	4	2	2	2	2	2	2	2	4	1	1	3	2
19	2017-10-01	14:55		3	2	2	3	2	3	2	1	1	3	3	3	2	2	2	2	2	2	2	2	3	3	2	2	3	2
19	2017-11-01	15:58		3	3	2	3	2	3	2	2	2	3	2	3	3	2	2	3	2	1	2	2	2	4	2	2	3	2
19	2017-12-01	14:23		3	3	2	2	2	2	2	2	3	2	2	2	3	2	2	2	2	2	2	2	3	3	2	1	3	2
30	2017-10-01	11:05		1	2	2	2	2	2	0	1	1	2	2	2	1	2	2	3	2	1	1	2	1	1	1	1	2	1
30	2017-11-01	13:31		2	2	1	2	0	2	0	0	1	2	2	1	1	0	1	1	1	1	1	1	1	1	0	1	2	1
30	2017-12-01	12:54		3	1	2	2	1	2	0	0	1	1	2	2	1	1	2	2	1	1	1	1	0	0	0	1	2	1
15	2017-10-01	15:43		4	3	3	3	2	3	2	2	3	4	2	2	4	3	2	2	2	1	2	1	2	3	2	2	3	2
15	2017-11-01	17:12		3	2	2	3	2	2	2	2	3	2	2	2	3	2	2	2	2	2	2	2	2	2	2	1	3	2
15	2017-12-01	18:27		4	2	1	3	2	3	1	1	3	3	3	3	3	3	2	2	2	1	1	1	2	1	2	1	3	2
20	2017-10-01	11:14		3	2	2	3	2	2	3	0	3	2	2	3	1	3	2	2	3	1	3	1	3	2	1	1	3	2
20	2017-11-01	12:34		3	2	1	2	2	3	3	1	1	2	3	2	1	3	2	2	2	1	3	1	3	1	1	1	3	2
17	2017-10-01	11:06		3	2	3	1	2	2	1	1	3	3	2	2	3	2	1	2	3	1	1	1	2	1	1	1	3	2
17	2017-11-01	13:22		3	2	2	2	2	2	1	1	3	3	1	2	3	0	1	1	3	3	2	2	1	1	2	1	4	2
17	2017-12-01	18:38		3	3	2	3	2	2	1	1	4	3	2	3	4	2	2	2	3	1	2	2	2	1	1	1	3	2
16	2017-10-01	14:04		3	4	2	1	2	2	2	1	2	2	1	3	1	2	2	2	2	1	2	1	3	0	2	2	2	1

	Pain Subscale					ADL (Activites of Daily Living) Subscale				
User ID	October	November	December	incidence of repeated reported data	Mean over all Months	October	November	December	incidence of repeated reported data	Mean over all Months
3	67		47		57	74		54		64
5		36	47		42		37	47		42
7		28	33		31		32	43		38
8	58	61	56		58	68	66	59		64
9	22	28	42		31	38	41	43		41
10	44				44	71				71
13	36	19	19		25	37	6	15		19
14	61	58	47		56	57	50	43		50
15	31	44	44		40	43	47	49		46
16	47	42	50	47	47	57	57	65	57	59
17	50	50	42		47	54	53	47		51
18	14	11	17		14	24	10	19		18
19	47	39	44		44	41	41	44		42
20	44	50			47	49	51			50
21	78	78	83		80	87	75	84		82
23	17	19	19		19	25	22	25		24
25	42				42	38				38
26	39	33	36		36	53	47	57		52
27	58	64	64		62	82	82	82		82
28	53	58	67		59	65	59	62		62
30	64	72	67		68	60	74	72		69
34	50				50	49				49
35	64	78			71	81	90			85
37	97	81			89	97	97			97
38	42	47	44		44	54	63	68		62
42	58	53	53		55	66	59	60		62

*Figure 68 – **KOOS data extraction**. Above: Part of the Raw Data; Below, left: Extracted Data of all Scores; Below, right: Extracted Data of KOOS with subscales scores averaged and transform into KOOS standard score.*

Pain Levels

While the KOOS scores were taken once a month, pain levels were accessed daily, twice a day or weekly. The content of the “pain-levels.csv” file for every day was extracted and combined in a single file. An example of pain level raw data from the 1st Jan 2018 is presented in Figure 69.

	A	B	C	D	E	F
1	user_id	timestamp	pain	pain_site	side	notification_type
2	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T17:00:27.690Z	6	KNEE	RIGHT	PAIN_DAILY_FUNCTION
3	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T12:48:44.405Z	7	KNEE	RIGHT	PAIN_OVERALL_MORNING
4	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T17:00:34.773Z	7	KNEE	RIGHT	PAIN_AGGRAVATING_ACTIVITY:SQUATTING_OR_BENDING_OR_KNEELING
5	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T20:36:50.074Z	9	KNEE	RIGHT	PAIN_OVERALL_AFTERNOON
6	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T20:36:57.156Z	9	KNEE	RIGHT	PAIN_AGGRAVATING_ACTIVITY:WALKING
7	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T20:36:53.789Z	9	KNEE	RIGHT	PAIN_DAILY_FUNCTION
8						

User. ID	Day. numeric	Pain. side	Pain Score. QoL(w)	PainScore. Daily Function(d)	PainScore. Morning(d)	PainScore. Afternoon(d)	PainScore. Important(w). score	PainScore. Important(w). type	PainScore. Aggravating(d). score	PainScore. Aggravating(d). type	PainScore. alternative. score	PainScore. alternative. type
3	736951	1	NaN	8	NaN	7	NaN	NaN	7	15	NaN	NaN
3	736952	1	NaN	8	6	NaN	NaN	NaN	8	15	NaN	NaN
3	736953	1	NaN	7	9	8	NaN	NaN	8	15	NaN	NaN
3	736954	1	NaN	5	6	5	NaN	NaN	4	15	NaN	NaN
3	736955	1	5	4	3	2	NaN	NaN	3	15	NaN	NaN
3	736956	1	NaN	6	5	6	NaN	NaN	7	15	NaN	NaN
3	736957	1	NaN	8	7	6	NaN	NaN	6	15	NaN	NaN
3	736958	1	NaN	7	6	7	8	7	8	15	NaN	NaN
3	736959	1	NaN	7	8	7	NaN	NaN	6	15	NaN	NaN
3	736960	1	NaN	5	NaN	6	NaN	NaN	5	15	NaN	NaN
3	736968	1	NaN	2	2	2	NaN	NaN	2	15	NaN	NaN
3	736969	1	3	3	3	3	NaN	NaN	3	15	NaN	NaN
3	736970	1	NaN	8	6	8	NaN	NaN	8	15	NaN	NaN
3	736971	1	NaN	6	5	7	NaN	NaN	6	15	NaN	NaN
3	736973	1	NaN	8	7	9	NaN	NaN	8	15	NaN	NaN
3	736974	1	NaN	5	6	7	NaN	NaN	6	15	NaN	NaN
3	736976	1	7	5	6	3	NaN	NaN	4	15	NaN	NaN
3	736978	1	NaN	7	7	8	NaN	NaN	7	15	NaN	NaN
3	736982	1	NaN	4	4	4	NaN	NaN	4	15	NaN	NaN
3	736985	1	NaN	2	2	NaN	NaN	NaN	3	15	NaN	NaN

*Figure 69 – **Pain Level data extraction**. Above: raw pain level data; Below: pain level data after extraction.*

Every “notification type” and “side” was assigned a numeric value to enable easier data handling. Looking at the data, “Pain Side” was always assigned the label “KNEE”, while the pain “side” could either be the “LEFT” or “RIGHT” knee. There were 16 different “Notification Types”, with the weekly measures of “QUALITY OF LIFE” (QoL) and pain interference with “IMPORTANT ACTIVITY” (IMP) and the daily measures of “DAILY FUNCTION” (DyF), “MORNING” pain (OAM), “AFTERNOON” pain (OPM) and “AGGRAVATING ACTIVITY” (AGG). We dropped the “pain_side” information and converted other text information into numeric values, which are indexed in Table 15.

Notification Type			Number
Pain Side		Left	1
		Right	2
Notification Type	**Pain Important Activity:**	Do Household Tasks	5
		Get Washed and Dressed	6
		Play Sport	7
		Walk	8
		Work Effectively	9
	Pain Aggravating Activity:	Sitting For Long Periods	10
		Sitting To Standing	11
		Squatting Or Bending Or Kneeling	12
		Standing	13
		Turning Or Twisting	14
		Walking	15
		Walking Up Stairs or Inclines	16
Other Notification types		Quality Of Life	1
		Pain Daily Function	2
		Pain Overall Morning	3
		Pain Overall Afternoon	4

Table 15 - Indexes for Notification Types of the Pain Level data

7.5.4. Extracting Sensory Data

In this section, the data wrangling process of the sensory data will be described and problems that we encountered during data extraction will be discussed in detail. However, the descriptions will not necessarily be in chronological order.

We will describe a manual data extraction approach using Excel 2007. This approach was performed on one day of data and will be helpful for explaining the formatting of the sensory dataset. The manual approach was used to test the accuracy of our data wrangling. In contrast to the recorded pain data, this is a task that is not only inconvenient to do manually via Excel but cannot technically be done without using a computing environment, such as Matlab on iCSF. There exist two Matlab codes. The first was functional but had to be completely rewritten due to performance issues.

First, the manual approach will be described, including various issues we encountered during data extraction, followed by the final Matlab code and an overview of its differences from the original one.

7.5.4.1. Manual Data Extraction Approach

When looking at the sensory data the most complex problem is the massive data size. It is worthwhile to note that the “sensor-reading.csv.gz-000xx-of-00020” files are quite large and that the data seems to be evenly split into 20 separate files (see Figure 66). Indeed, the largest of these files, recorded on the 30th of September 2017, is around 91,600 KB. Taken together the 20 zipped “sensor-readings.csv” files for that day take up around 1,832,000 KB of space. After unzipping the file the required data volume increases by 6.19 times. Therefore the unzipping of all “sensor-readings.csv” files for the 30th of September would require about 11,338,000 KB of space. On the

other hand, the “pain-level.csv” files and “activity-detections.csv” files take up much less space with 8 KB and about 2,000 KB respectively after unzipping the files. Given the above information, we can estimate that the total data volume of the “sensor-readings.csv” combined overall days would be around 900,000,000 KB (~900 GB) after unzipping the data (the zipped data volume size lay around 145,000,000 KB).

Another large issue is the amount of time processing the files takes. Unzipping the 1st sensory file on the 1st of January 2018 takes around 18 sec, which translates to unzipping at roughly 738 KB/sec. Therefore we estimate the time it would take to unzip all of the 145,000,000 KB of sensory data to about 196,000 sec or 54.4 h. It would take 2.27 days only to unzip the data. This would not be as much of a problem if the data was in a sensible format. However, when looking at the unzipped data, it becomes clear that this is not the case (see Figure 70). One file of raw data contains more than one participant’s data. In the case of the 1st January, there is data of three different participants and the data seems to be sorted by their ID. Looking at the other sensory data files in the same folder reveals that, when it comes to the smartwatch ID, all 20 show the same pattern. The green box in Figure 70 highlights the time of day corresponding to each row of data and it illustrates that the timestamps were not sorted in any obvious manner. Furthermore, the timestamp reveals that some data from the following day was saved in the file, contradicting the folder labelling. At first glance, we can see that five things need to be done to change the data format into something usable: the data of all 20 files for a day need to be combined, separated by the participant, separated into sensory types, divided into the proper days and sorted by timestamp.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	user_id	timestamp	accelerometer/x	accelerometer/y	accelerometer/z	gyroscope/x	gyroscope/y	gyroscope/z	magnetometer/x	magnetometer/y	magnetometer/z	barometer	heart/bpm	distance/change	step_count
2	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T13:26:48.000Z													27527
3	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T16:27:23.000Z							-26.622	36.852	-13.785				
4	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T10:16:57.000Z							35.673	-2.976	55.319				
5	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T19:40:50.000Z							-17.642	-8.429	-33.28				
6	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T15:25:22.000Z							-45.653	-5.681	-2.207				
7	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T20:30:56.000Z	-5.326	-8.31	-1.876										
8	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T18:05:52.001Z	-1.285	-9.978	-0.213										
9	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T14:19:45.002Z							-11.409	30.225	-13.142				
10	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T10:37:53.001Z							-14.75	26.102	29.692				
11	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T11:29:05.001Z				0.259	0.934	0.779							
12	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T21:28:33.000Z				-0.005	-0.001	0.001							
13	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T10:12:18.000Z				-0.033	-0.007	0.038							
14	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T11:18:26.000Z				-0.018	0.051	0.016							
15	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T20:30:58.001Z	-5.257	-8.255	-1.754										
16	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T16:17:06.001Z							-25.682	19.876	37.181				
17	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T22:00:34.001Z				0.002	-0.001	0							
18	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T13:17:54.000Z				-0.022	0.01	-0.017							
19	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T12:09:39.000Z	-1.639	-8.167	-0.809										
20	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T21:49:55.002Z				-0.067	-0.002	-0.021							

	A	B
1	user_id	timestamp
2	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T13:26:48.000Z
3	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T16:27:23.000Z
4	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T10:16:57.000Z
5	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T19:40:50.000Z
6	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T15:25:22.000Z
7	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T20:30:56.000Z
8	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T18:05:52.001Z
9	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T14:19:45.002Z
10	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T10:37:53.001Z
11	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T11:29:05.001Z
12	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T21:28:33.000Z
13	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T10:12:18.000Z
14	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T11:18:26.000Z
15	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T20:30:58.001Z
16	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T16:17:06.001Z
17	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T22:00:34.001Z
18	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T13:17:54.000Z
19	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T12:09:39.000Z
20	9a52649d-e5ed-45e5-89ab-a87e34b7917d	2018-01-01T21:49:55.002Z

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
920230	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T18:35:43.546Z							-51.67	102.5	179.2				
920231	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T18:35:43.763Z				-0.012	-0.013	0							
920232	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.991Z							-72.37	94.75	217.4				
920233	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.868Z							-71.06	94.5	215.6				
920234	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.619Z	9.808	-1.022	1.524										
920235	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.522Z							-72.44	95.21	216.2				
920236	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.516Z				-0.004	-0.011	-0.004							
920237	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.316Z				0.01	-0.016	0.004							
920238	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.597Z				0.002	0.011	-0.009							
920239	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:08:47.938Z				0.002	0.002	0							
920240	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-02T03:41:51.429Z				0.004	-0.004	-0.005							
920241	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-02T03:41:51.664Z				0.011	0.004	-0.001							
920242	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-02T03:41:51.722Z							-37.52	97.68	169				
920243	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-02T03:41:51.569Z				0.009	0.004	-0.006							
920244	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-02T03:41:51.759Z				0.005	0	-0.006							
920245	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T18:37:51.393Z							-52.8	111.7	213.7				
920246	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T18:37:51.716Z				0.001	-0.01	0.001							
920247	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T18:37:51.939Z				0.057	-0.013	-0.006							
920248	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T18:37:51.812Z	9.983	-0.627	0.4										
920249	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T18:37:51.528Z	9.945	-0.625	0.409										
920250	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:10:55.680Z				-0.063	-0.012	0.005							
920251	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:10:55.418Z	9.768	-1.201	1.713										
920252	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:10:55.520Z	9.741	-1.235	1.725										
920253	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-01T23:10:55.781Z				-0.053	-0.012	0.002							
920254	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-02T03:43:59.548Z							-50.56	100.8	193.7				
920255	e11dcfd6-53e5-4e24-81e9-a1b486b349c9	2018-01-02T03:43:59.949Z	1.388	-3.998	8.839										

*Figure 70 - **Examples for "sensor-readings.csv" raw data format**. Taken from: "sensor-readings.csv.gz-00000-of-00020" in folder "2018-01-01"; Above: full view of the first 20 rows of raw data; Below, left: Zoom onto smartwatch ID and timestamp; Below, right: last 36 rows, data of the following day is marked with red.*

To illustrate the way the file format causes the large size of the raw sensory data, we examined the file with the ending “00000-of-00020” of the day “2018-01-01” in Excel and observed changes in the file size. The size and length of the data files are reported in Table 16. First, we unzipped the data, which increased the data size by 524%. Saving the data in Excel, even in .csv format increased the data size slightly, causing a 0.49% increase to affect the calculations. By changing the 36 character smartwatch ID to a numeric user ID, the file size was reduced by 37%. After converting the timestamp into a numerical value (excel function: DATEVALUE and TIMEVALUE) the data was further reduced by 14%. Dividing the data into the different sensors resulted in another 4% reduction. This step removed all the empty cells in the data seen in Figure 70. Overall implementing these steps did not change anything about the information value in the files but reduced the data size by 48%. Furthermore, the data was reduced by dividing it into the individual participants. Participant 38 makes up 33.12% (304,800/920,315 lines of data) of the data. Therefore just removing the column with the smartwatch ID in the data means a nearly 5% reduction of the overall data.

	Of one file (“[…]00000-of-00020”)			
	All Users		User 38	
	Size in KB	Lines	Size in KB	Lines
zipped	13,289	920,254	-	304,782
unzipped	82,964			
after saving	83,371			
smartwatch ID to user ID	52,501		16,537	
Timestamp to Time Value	45,311		12,635	
Accelerometer	14,345	306,948	4573	102,955
Gyroscope	13,881	301,779	4275	99,358
Magnetometer	15,068	304,853	4647	100,285
Barometer	2	43	1	16
Heart Rate	203	6,551	61	2,145
Step Count	5	141	2	41
Sum of Sensors	43,504	920,315	13,559	304,800
Estimated sum for 20 files	870,080	18,406,300	271,180	6,096,000

Table 16 - Manual extracted Information

Table 16 also illustrates why the data is impossible to handle manually, even when viewing the data is the only objective. Excel can only load 1,048,576 data rows at once, corresponding to around 94,500 KB of unzipped data (15,100 KB of zipped data). Consequently, about 101 days of data in our study cannot be opened completely in Excel. The 1st January 2018 was selected as an example since its data size made it just readable in Excel and it contained more than one participant. However, even when combing the data of the sensory devices individually for one participant, combining 20 similar large files will still be too large to open fully. Therefore, we had to sort the accelerometer, gyroscope and magnetometer data in 2 steps. Taking a maximal size of around 52,400 rows of accelerometer data in each file into account (33.78% of the data). At this point manually handling the 1st January 2018 was not feasible anymore. Therefore, another day was selected with the criteria of containing exactly two participants in the day, one for accurate data extraction and one to see if the code might be thrown off by when looking at multiple participants. We knew how many participants were in each day from a Matlab code that was written to extract the smartwatch IDs for each day. The code was written as a basis to extract the sensory

data before the data file with the key between smartwatch ID and participant ID was obtained. Going through the files manually we identify the 07th January 2018 as suitable for our purposes.

Unzipping all 20 files of a folder at the same time is impractical, firstly due to the massive amount of space the data would take up and second because of the nomenclature of the files, where every “sensor-reading.csv.gz-000xx-of-00020” is unzipped to “sensor-reading.csv”, causing the program to overwrite the file once it is unzipped. The first time we encountered the problem we used the “Command Prompt” window in Windows, selecting the folder with the raw data with the “cd” command and combining all files with the command “copy /b sensor-readings.csv.gz* sr.csv.gz” (or “copy /b *.gz* sr.csv.gz”). However, this method resulted in too large files, which caused the Matlab program to crash when attempting to open them. In the manual approach, we renamed the data files before unzipping whereas in the Matlab codes we wrote a loop that unzipped and processed the data of one file, before going on to the next one.

7.5.4.2. Extraction of Information

During the data wrangling process, two additional Matlab codes were written to create an overall view of the data. The first one was concerned with identifying the smartwatch IDs present in the sensory data files of every day. It was a step needed for the 1st version of the data extraction code but was not needed anymore in the final one.

The second Matlab code was concerned with identifying the length of time each participant wore the watch every day. It was written with the goal in mind to obtain a quick and accurate estimate of the times the participants wore the watch without the need to fully wrangle the data of the whole study. During the data transfer, the data of all active participants was equally (but seemingly randomly) divided into 20 files. Consequently, all 20 files need to be combined, divided by smartwatch ID and sorted by timestamps to reconstruct the full data set. Wearing the watch for one minute creates 9000 data points (3000 per axis) each for the accelerometer, gyroscope and magnetometer, 1 data point for the barometer and 60 data points for the heart rate. Therefore, every file contains around 1.253 data points of the 27.061 data points total for this minute. Figure 71 illustrates the random way data is sorted into the 20 sensory data files of each day. Since the data is distributed randomly, extracting the smartwatch ID and timestamp of the first file for each day and rounding the timestamp to the minute is sufficient to provide an accurate representation of the time that the watch was worn in minutes.

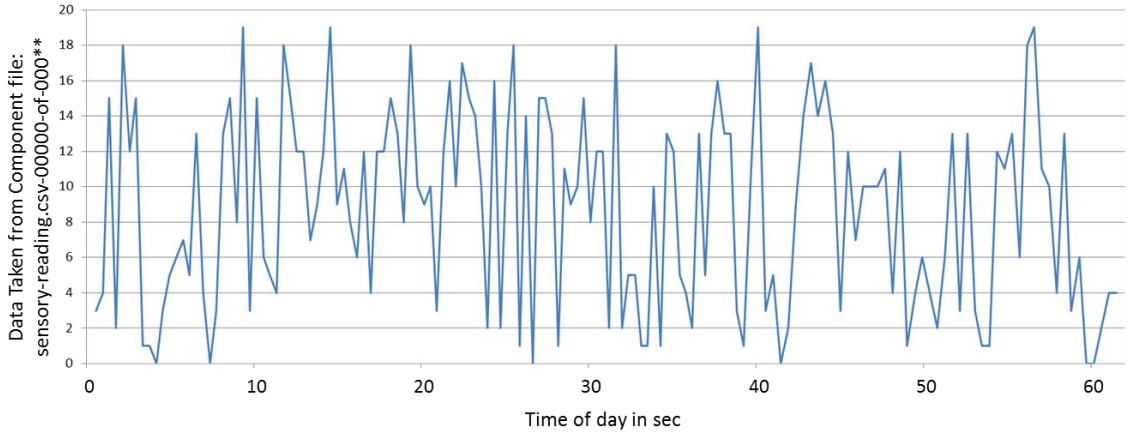


*Figure 71 - **Origin of data from the 20 sensory data files**. Taken from one min of accelerometer data of participant 28 of the 14th Sep 2017.*

The Wear Time extraction code was completely rewritten multiple times because the first approach was not successful. It first separated the timestamps into timestamps of the "current" and the "following" day. Afterwards, the timestamps were sorted and the first and last value was extracted as the "first activity" and "last activity" time point of the day. Then "breaks" between timestamps that exceeded 1 minute were identified, their "start" and "end" times documented and their length calculated (break length = end time – start time). Afterwards, the total length of all breaks was subtracted from the difference between the "last activity" and "first activity" in order to obtain the length of time the participant wore the smartwatch that day. The data of participant 8 is an example of the problems with this approach (see Figure 72) but also provides an interesting view on how the data was collected and saved. Data, before 7 am in the morning, was saved in the data file of the previous day. Therefore the data with the timestamp "2017-09-14T06:50:30.259Z" would be saved within the folder "2017-09-13". We can conclude that the data upload onto the Google servers occurred at 7 am in the morning before and at 8 am after the 6th of November. Apart from the quite inelegant way the information about wear time was saved by the 1st code, it also had problems handling the data of the 1st of every month (see Figure 72). Another aspect that stands out is the overlap of "current" and "following" day data at the beginning of each month. The data of the "current" and "following" that was collected before 7/8 am seems to be labelled both with the date of the 30th/31st and the timestamps after 7/8 am both ended up in the column of the 1st. This was not a constant occurrence. 62% (n = 16) of the participants did not show this pattern around the 1st of the month in their data and it did not appear at all in 27% (n = 7) of participants. The error only occurred in 3 participants (12%) consistently.

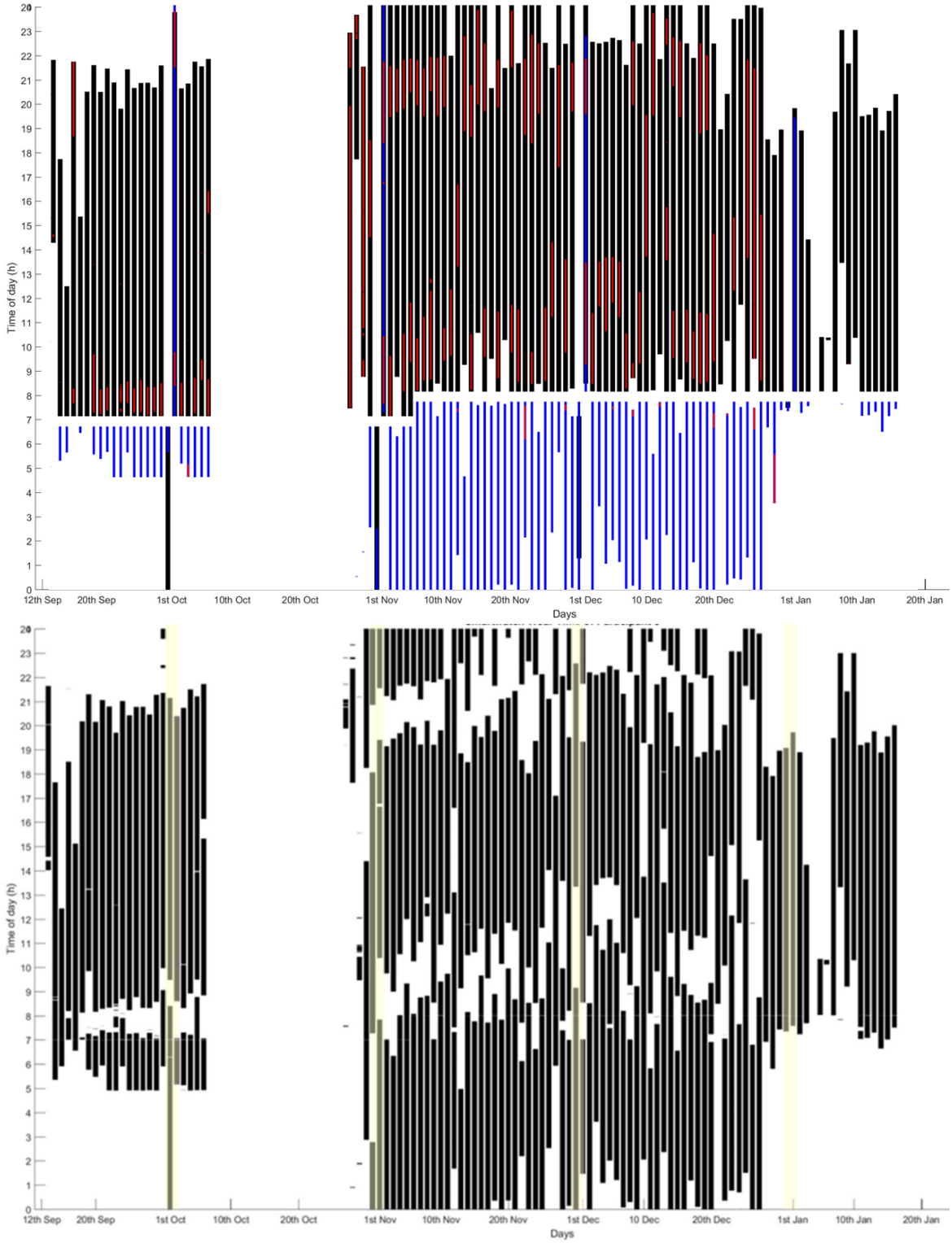


Figure 72 - ***Attempts to extract Wear Time****. Wear Time for Participant 8. Above: first attempt; Black columns: time between the "first" and "last" activity of the "current" day; Blue columns: time between the "first" and "last" activity of the "following" day. Red columns: time between the "start" and "end" of every "break". Below: final result; Black columns: time between the "first" and "last" activity of the every data collection episode; Yellow area: problematic areas from the 1st attempt.*

Examining the data manually and other approaches to identify the origin did not result in an explanation of why this unexpected behaviour occurred and why it was so inconsistent across participants. The most likely explanation would be that for some participants uploading the data onto the Google servers was delayed by a day due to problems with accessing the servers on the

correct day. A completely new Matlab code was written where the raw data was divided into clusters containing the data of the same participant identifiers for the same day. Time points were then sorted and divided into segments containing continuous-time points that were not further apart than 1 minute from each other. When the last time point of a segment occurred before 4 am we assigned the segment to the cluster of the previous day for the participant. We were able to infer the length of time that the watch was worn by subtracting the first time point of a segment from its last time point. Adding together the time length of all the segments of a cluster gave the participant's total wear time for the day. The time of day the participants first put the watch on and the last time it was taken off was documented. Furthermore, the number of times and length of times the watch stopped recording data in between the first and last activity of the day was noted. The final working Matlab code does not save the start- and endpoints of the breaks in the data as the first code did but saves the first and last timestamp of every time episode that the watch was worn. It then looks at the timestamp itself for information about the date of the episode, rather than assuming that a timestamp with a different date than the "current" date is the following date. The resulting wear time of Participant 8 can be seen in Figure 72, below. Comparing the data of the first of every month (marked in yellow) shows that the wear time episodes are now assigned the right date. Visualisation of the wear time episode for all participants can be found in the Appendix, Figure 89).

7.5.4.3. Previous Data Wrangling Approach

The first Matlab code is written to extract the sensory data from the raw data imported through the Matlab "importdata" command. This approach is not advised since it is only designed to import numeric data (as well as column and row headers) and therefore had problems handling the non-numeric data of the smartwatch ID and the timestamps. Furthermore, it relied on knowing which smartwatch IDs would be present in the file, information that was supplied by the prior mentioned smartwatch ID extraction code. Rather than using the "datenum" function, a standard way of transforming timestamps into numeric values, the original code added up the msec of the day as a value for the timestamps (for example, 7:30 am and 46.272 sec = 3600000*7 + 60000 * 30 + 1000 * 46 + 272 = 27046272) and saved the date turned into a numeric value separately. The resulted data was then saved in a file with the date and the participant's name using "dlmwrite". The next step was to combine the 20 files from the same participant on the same day, loading the data using "csvread". Data was separated by date, divided by sensor, sorted by time and saved using "dlmwrite" again. "importdata", "dlmwrite", "csvread" on all load and save data in a Delimited ASCII (DEL) file format as a csv-file. It is a text file format and therefore much slower than using the binary format chosen in the final data-wrangling approach. Extracting the data for the 19th of September 2017 this way took around one day to extract and transform the data for 18 participants. The data files for participant 28 added up to around 610,000 KB of data. The final data wrangling code on the other hand only takes around 2 h to wrangle the same amount of data and requires only around 221,000 KB of space. In summary, the old code was slower, required more space to save the data and was less elegant.

7.5.4.4. Final Data Wrangling Approach

We had to write the data wrangling code in two steps:

1) Unzipping the data and dividing it by participant and data type
2) Sorting the data and bringing data saved in different days together

Due to the large data mass, sorting the data from the start was not possible, since it would have overloaded the memory. Loading all the sensory files of the 30th of September would require the computer to handle 1,832,000 KB of data at once. The university's Windows computer allows Matlab a maximal memory of 12365 MB (12,365,000 KB) and a physical memory of 8149 MB (8,149,000 KB). Therefore while the computer would have the physical memory to handle the data all at once, it would vastly slow the processing speed down.

First, we created a map that associated the smartwatch ID with its corresponding user ID. Due to the previous "participant ID" code, we know that the data of 6 additional smartwatches was present from technical failure during the patient equipment phase. Further general variables are set up that can be changed by the user. The "select choice" variable lets the data analysis chooses the range of days that shall be processed at once. Furthermore, the user can change the "basis directory" to enable the quick use of the code on different devices (for example the University PC, the Incline system or a Laptop for group demonstrations), as long as the recommended folder structure in the user guide is followed.

A loop for the selected days and months is set up and variables that are used for the day for data used are set up or reset to be empty. This prevents data from previously processed days from accidentally ending up in the wrong file. Also, indicator variables are set up to keep track of data that was saved on a wrong day. Furthermore, a directory to save the wrangled data is created for the selected and following day. Then a loop is started that goes through the 20 zipped files, extracts their data, divides and transforms it and saves them. First, the "component loop" unzipped the current file and opens it. When handling binary files, the commands "fopen", "textscan", "feof", "fwrite" and "fclose" are of importance.

In the file opening process "fid1 = fopen(filename1,'rb')" opens the file we want to read (permission: 'rb') from and gives it an index number, so that multiple files can be kept open at the same time. We used "fid2 = fopen(filename2,'wb','ieee-be')" to create files for each participant and sensor type to write (permission: 'wb') in. The machinefmt: 'ieee-be' ending causes the file to be written in a big-endian ordering, it describes the order in which the "bytes are stored in computer memory[. In this case] the "big end" (most significant value in the sequence) is stored first (at the lowest storage address)" (Rouse, 2005). We create a loop where a certain number of lines is read in from the raw data, transformed and saved in the target folders. "textscan(fid1, 'formatSpec', N, 'Delimiter',',','MultipleDelimsAsOne', false)" reads 1000 rows (N: 1e5) from the raw data in the format (formatSpec: '%s%s%f%f%f%f%f%f%f%f%f%f%f%f'). The textscan command is looped using "~feof(fid1)" to identify when the end of the file is reached. The transformed data can be saved into the already opened target file using "fwrite(fid2,data,precission)". The precision

determines the “class and size in bits” the data is written in. Afterwards, when one is finished reading or writing into a file it has to be closed using “fclose(fid)”.

In the context of the Matlab code, we first open the file with the “00000-of-00020” ending and extract all smartwatch IDs from it. Then the ID map is used to determine which participant’s data was recorded during the day and files for each participant are opened. For each participant, 12 files are opened, two files for each sensor. Sensory data and time of the data are saved separately in the same order since they require different precisions. Time requires a precision of 64-bits floating time points (precision: 'float64’), while sensory data only requires 32-bits floating time points (precision: 'float32'). Saving them separately, therefore, saves memory space and therefore reduces saving and loading times. After this, the date from the timestamps is extracted and further files are opened for each participant whose data contains dates that are not the same as the date of the current folder. These files are labelled “morning data”.

The next step is to unzip and load all sensory files in a loop. Loading the data is facilitated using the “textscan” command again in chunks of 50,000 rows at once. This keeps the amount of data that is processed at once smaller than it would be otherwise. The data of each chunk is divided by participant, date and sensor type and is saved into the already opened 12 binary files. For the second step of the data wrangling process, the data from the 12 binary format files is loaded again. If “morning data” is present, it is added to the data. For each sensor, the sensor data and time data are reconnected into one matrix. The entire data is then sorted by time and saved in binary format.

7.5.4.5. Extracting Wear Time

In order to obtain a quick overview of the participant’s compliance when it came to wearing the watch, we extracted the participants' “Wear Time”. The Sensory data was only recorded while the watch was worn and was uploaded to the servers once a day. From the format of the raw data files, we assume that the raw data was uploaded at the beginning of the study at 7 am and then from the 6th of November onwards at 8 am to the Google servers. During the data transfer, the data of all active participants were equally (but seemingly at random) divided into 20 files. Consequently, all 20 files need to be combined, divided by smartwatch ID and sorted by timestamps to reconstruct the full data set. Wearing the watch for one minute creates approximately 9000 individual data points. The sample frequency of the sensors are: Accelerometer/Gyroscope/Magnetometer (~ 50 data points/sec); Barometer (~0.5 data points/min); Heart Rate (~ 1 data points/sec) and Step Count (with no regular patters). Therefore every file contains around 450 data points per minute. Hence, extracting the smartwatch ID and timestamp of the first file for each day is sufficient to provide an accurate representation of the time that the watch was worn. Processing speeds for unzipping the files is approximately 738 KB/sec. The smartwatch data collected over the course of the whole study is 138 GB large in zipped format. The unzipping process to open all files would take roughly 2.2 days alone. Considering loading and transforming the data, extracting wear time would most likely double the processing time. Only unzipping one file out of 20 for each day reduced the data volume to around 7.25 GB, which was done in around 3 h. The value extraction time took around 6 h.

Raw data was divided into clusters that contained the data of the same participant identifiers for the same day. Time points were then sorted and divided into segments containing continuous-time points that were not further apart than 1 min from each other. When the last time point of a segment occurred before 4 am we assigned the segment to the cluster of the previous day for the participant. We were able to infer the length of time that the watch was worn by subtracting the first time point of a segment from its last time point. Adding together the time length of all the segments of a cluster gave the participants total wear time for the day. Both the time of day the participants first put the watch on and the last time it was taken off was documented. Furthermore, the number of times and length of times the watch stopped recording data in between the first and last activity of the day was noted. Wear time length of each data segment in a cluster was obtained by subtracting its last time point from its first time point. The following wear time parameters were extracted for each participant for each day: the first/last time point of the first/last segment was taken as the “First activity” and “Last activity”; the wear time length of all segments was summed into “Minutes of sensor data”; $N_{Segments}$-1 describes the “Number of breaks” and subtracting the minutes of sensor data from the time difference in minutes between the first and last activity was used to calculate the “Break times”.

7.6. Results Data Wrangling

7.6.1. Sensory Data Observations

An example of the raw output data of the accelerometer sensor is depicted in Figure 73.5. Examples of the gyroscope and magnetometer raw output data can be found in the Appendix, Figure 90.5 and Figure 91.5. After the data wrangling process, we looked at the time difference between consecutive timestamps to infer at what sample frequency the data was collected. Figure 73 displays the frequency at which the data was collected from participant 7 on the 19th of September. Usually, a data point was recorded every 20-21 msec (see Figure 73.3). This agrees with the requested sampling frequency of 50 Hz. However, the sampling rate frequency is not stable throughout the whole time of the data collection. Figure 73.2 shows that a few times a minute the sampling rate increases to up to 16.67 Hz, but mostly reaches a value between 25 and 30 Hz (see Figure 73.1).

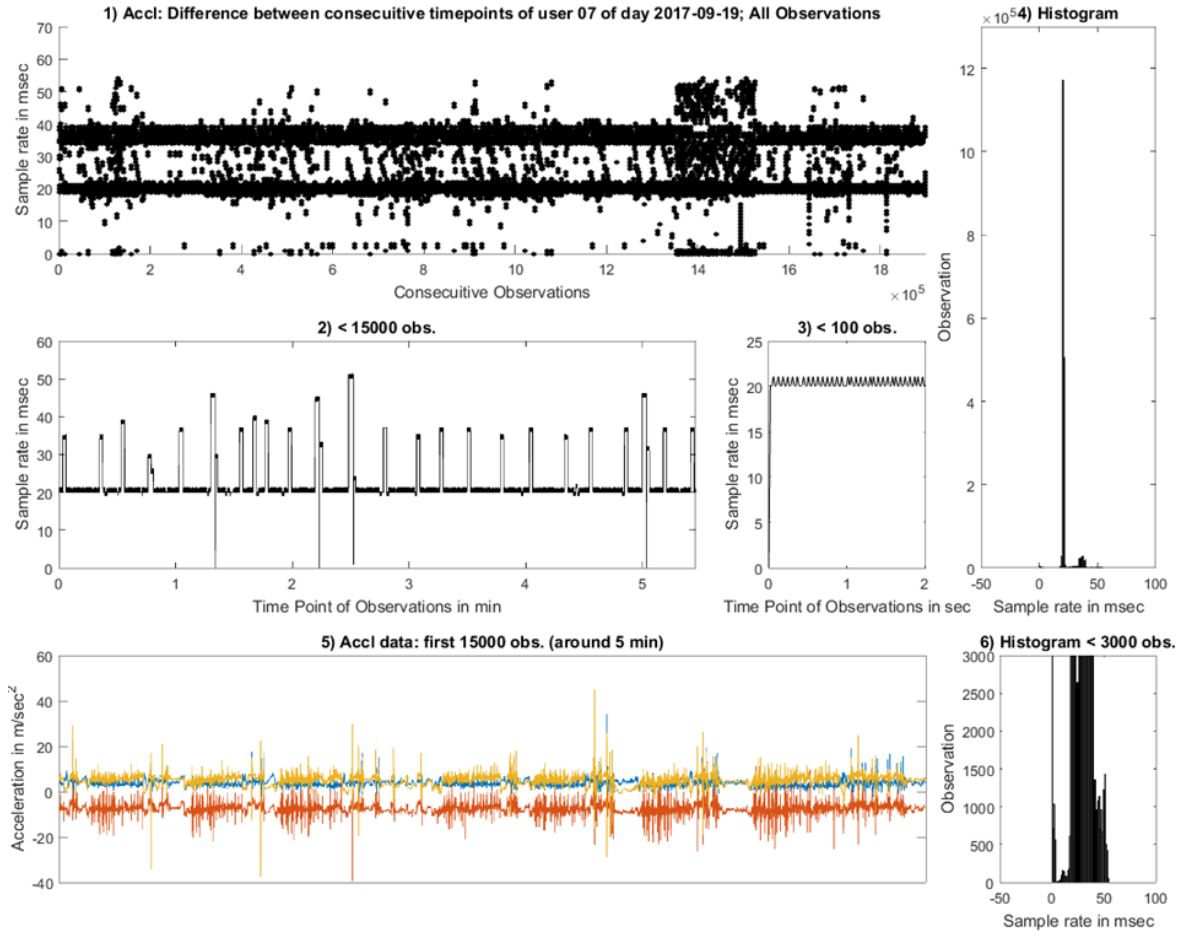


Figure 73 – ***Accelerometer data example and Sample frequency observations*** *of participant 7 for the 19th September.*

The short changes in sample frequency seem to last for about 50 observations (~ 1 sec) and then revert back to the standard 50 Hz sampling frequency. Two typical patterns can be observed around non-unique observation occurrences. The most common one is a non-unique observation episode, which will usually be preceded by a decrease in sample frequency of 50 observations in the 20 Hz range, followed by 50 observations with the same timestamp. The sampling rate then drops back to anywhere between 30-50 Hz for another 50 observations, before

settling back to around 50 Hz (see Figure 74.1). The non-unique observation episode pattern makes up 69% of all pattern around the non-unique observations and can be followed or precede a single non-unique observation pattern (see Figure 74.4). The single non-unique observation pattern occurs in 31% of cases and is quite similar to the previous pattern, except for the non-unique observations not lasting for longer than 2 observations (see Figure 74.2&4). Finally, there are only two outliers in the 149 events of non-unique data observation which are depicted in Figure 74.3&6.

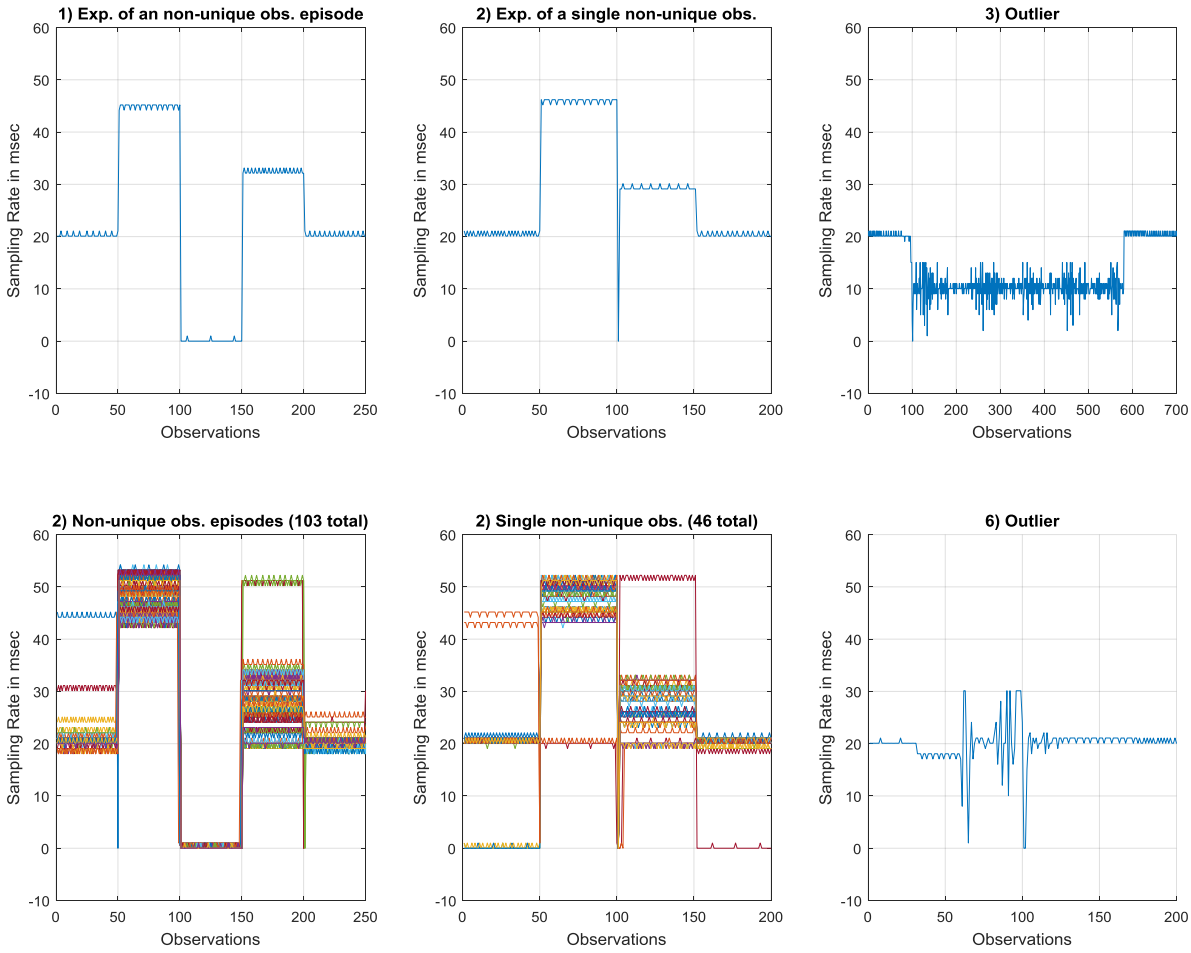


Figure 74 – ***Pattern around non-unique observations*** *(149 patterns total).*

Examples of barometer, heart rate and step count data can be found in the Appendix, Figure 90 to Figure 94. The barometer data was collected every 1.5 min with some outliers in the 15-30 sec range. Therefore it deviates from the expected 1 min intervals. The heart rate was collected around every 0.8, 0.9, 1 or 1.1 sec. Outliers are mostly present in the 0-2 sec range. On average the sampling rate was 1 Hz which agrees with the sampling rate described in the methodology. Step count data was recorded continuously at random time intervals and would reset once the data was uploaded to the Google servers.

7.6.2. Time of Pain Reporting

Table 17 displays the times of participants' reported their pain. We documented the earliest, latest, mean and the STD of the time reports for each pain score category. The participants managed to mostly respond to the request of filling out the questionnaire within one or two hours.

Most questionnaires were filled out during the lunch hours (IMP, OAM and QoL) or during dinner hours (AGG, DyF and OPM).

Pain Type	Earliest	Latest	Mean/STD
Aggregating Pain (AGG)	07:24	23:45	17:31 / 01:28
Important Function Impaired (IMP)	11:00	23:15	13:35 / 02:30
Daily Function (DyF)	08:38	23:45	17:31 / 01:27
Morning Pain (OAM)	11:22	18:05	12:52 / 01:15
Evening Pain (OPM)	17:22	23:12	18:34 / 01:08
Quality of Life (QoL)	11:00	23:22	13:17 / 02:15

Table 17 - Time of Pain Reporting

7.7. Discussion – Data Wrangling

Data Wrangling

The KOALAP dataset consists of about 900 GB of data. The following three main problems were identified during the transformation of the data: 1) the time-consuming unzipping, loading, processing and saving of files in text format; 2) the unnecessarily large size of the raw data; 3) the raw data not being sorted by timestamp and consecutive data points being saved randomly across 20 files. Our wrangling process reduced data extraction time by 99% and the data size by 79%. Implementing code that keeps track of the sampling rate, the start and end time of data collection episodes and the total amount of data points during the wrangling process is essential for the evaluation of data quality. Exploring the data after transformation revealed two problems with the original data: 1) sampling frequency is not constant at 50 Hz and 2) non-unique observations are present in the data. The small amount of non-unique observations (about 2% of around 2.18 years of data) supported the decision to delete them in the pre-processing stage. Changes in sampling frequency, however, occur too frequently and will have to be interpolated during pre-processing. Clearly, the time effort of the data wrangling process should not be underestimated when planning studies with large datasets. Furthermore, examination of the data can help identify and correct for errors in the data collection. While research graded devices (i.e. Axivity AX3) are more likely to save raw data in a format suitable for research purposes, raw data collection of consumer graded devices (e.g. Googles Huawei Watch 2 smartwatch) are not necessarily designed to be user-friendly. After all, the latter focuses their main outputs on energy consumption measures derived from their own algorithms, such as step counts, rather than prioritising accurate raw data output. With an increasing interest in patients' digital health self-monitoring, the use and evaluation of data collected by consumer graded devices will become more common. Awareness of potential data wrangling issues at the project planning stage, such as the possible occurrence of non-unique data points, can potentially improve the data collection process by ensuring a more ensuring thorough and expedient raw data collection.

The median sampling rates for the different sensors mostly behave as planned in the methodology with the exception of the barometer data, which was sampled at 1.5 instead of 1 min intervals. We had a lot of difficulties extracting the sensory data from the Google server, which happened to prolong steps in the data analysis. The main problems were due to the commercial aspect of the device and the size of the data. At the end of the study, the data was downloaded from the Google server and saved at two different locations in the research facility. While this is a

standard research practice, the step itself should be emphasized to researchers whose data collection is saved by a second party. After a certain period, Google would delete the data and it would no longer be accessible on their servers. Our proposal is to advise researchers to contact software developers when using consumer devices for research purposes. The technicians working on the programs have different priorities than researchers and will prioritise energy efficiency and reduction of memory over temporal accuracy. Furthermore, they will most likely spend a minimal amount of resources on making the data accessible for people working externally due to the lack of widespread or commercial demand.

Consequences for the KOALAP Dataset

A case can be made to exclude some participants. Most data of participant 27 is missing and the pattern indicates an error in during the data collection. The technical difficulties experienced by participant 7 would lead to the exclusion of at least 10 out of 51 days and the 41 days for which data collection works contain comparably few data points at unusual and irregularly times of the day. For participant 10 only a few days have recorded sensory data and even fewer questionnaires were answered. Further participants with only small amounts of days who filled out questionnaires are participants 25, 34 and 37. In conclusion, the exclusion of participants 27 and 10 is recommended, due to a lack of recorded sensory data. Furthermore, data interpretation in regards to linking activity with pain might become difficult for participants 7, 25, 34 and 37.

The occurrence of non-unique intervals is another source of missing data. However, considering that they only make up a tiny fraction of the whole data set, as can be seen in the sampling rate histograms, the pattern is quite predictable; therefore, it is assumed that the behaviour is related to sensor batching. The non-unique episode preceding the decrease in sampling rate could be related to the subsystem hardware storing sensory data in the local memory before the main processor retrieves the data and saves it onto a flash memory. This behaviour would occur when the sensor goes into a lower energy resting state to preserve energy (Little, 2018)

7.8. Results – Participants, Participation and Pain Scores

7.8.1. Participant Demographics

A total of 26 participants (13 females) took part in the study. The group had an average age of 64.2 ± 8.8 years. The mean and STD of weight, body height and inner leg length are 86.8 ± 21.4 kg, 1.68 ± 0.13 m and 78.3 ± 6.8 cm respectively. When asked if participants engaged in exercise, 22 reported regular low/moderate exercise, 8 engaged in vigorous-intensity sports/activity, 6 own wearables for activity monitoring, 6 used a smartphone app for activity monitoring and 6 use a smartphone health app. Furthermore, 10 participants rely on walking aids, including knee braces (1), sticks (5), crutches (2) and walking frames (2). Also, 5 participants underwent knee or hip joint replacement prior to the study and one participant had surgery during the study.

7.8.2. Participants Participation

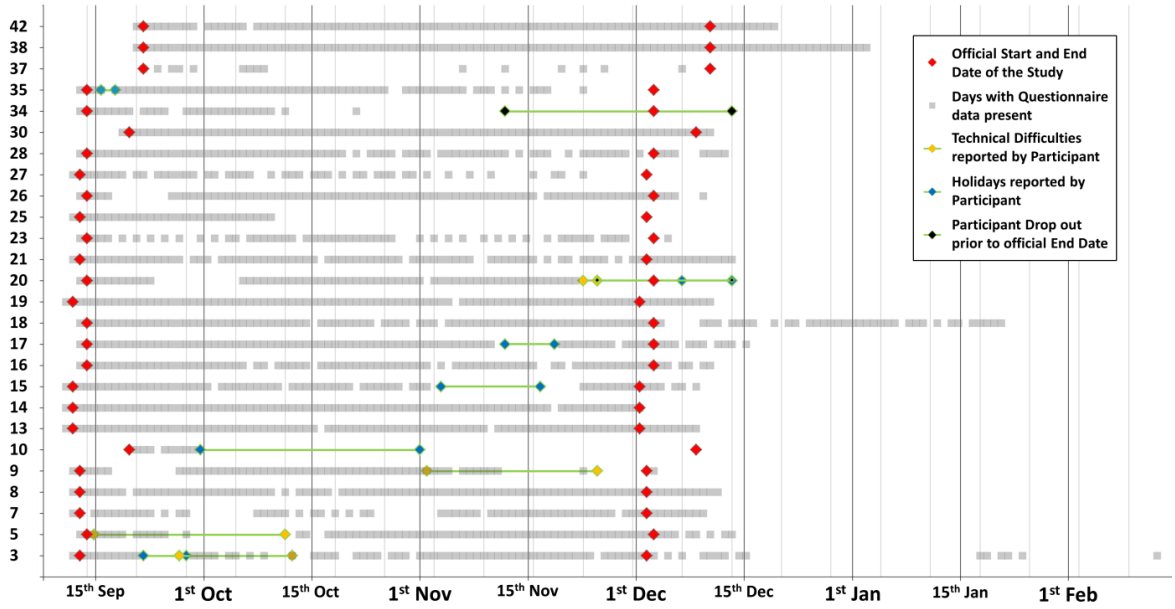


Figure 75 - ***Comparison between official start, end and breaks for each participant and their compliance on filling out the questionnaires****.*

The official data collection start dates for the 26 participants were recorded between the 13th and the 23rd of September 2017. The planned participation time was 3 months, with the official end date being set to the same day as the start day in December. However, a total of 9 people were noted to have breaks in their data collection, with 2 dropouts of the study, 5 holidays and 5 occurrences of technical difficulties (see Chapter 7.3.1, Table 14).

There is a difference between the official study start and end for each participant and their actual compliance in wearing the watch and answering the questionnaires (Figure 75). While most participants only had a few days where they did not fill in any questionnaires, some missed full weeks or even stopped answering the questionnaires entirely. The days' participants wore the watch shows similar but not always identical patterns. Between 10 and 118 days, sensory data is present for each participant, with a mean and STD of 73 ± 23 days. Figure 76 shows at what days sensory data is present for each participant and how many minutes of sensory data was collected in total.

Comparing the two figures reveals that answering questionnaires and wearing the watch did not necessarily overlap all of the time. For example, participant 37 only occasionally answered the questionnaires while wearing the watch for most days. Participant 42, on the other hand, filled out the questionnaires nearly every day but lacks sensory data records for some days at a time. Overall we recorded data from some participants well past the official end date of the study, but for some participants, data stopped being collected before the official end of the study (notably participant 10).

Taking a closer look at the amount of sensory data that was collected every day, we can infer how regularly the participants wore the watch. The data was collected by the “Wear Time” Matlab code. Overall, participants wore the smartwatch for an average of 10.15 h (STD = 1.24, 95% CI

[9.67 h, 10.62 h]) daily. On average they started wearing the watch at 9.07 am (STD = 1.21 h, 95% CI [8.60 am, 9.53 am]) and took it off at 6.31 pm (STD = 2.44 h, 95% CI [5.37 pm, 7.25 pm]). Between the start and end of their daily wear time participants took off the watch 1.92 times on average (STD = 7.93, 95% CI [-1.13, 4.97]). The results for each individual participant are reported in Figure 77.

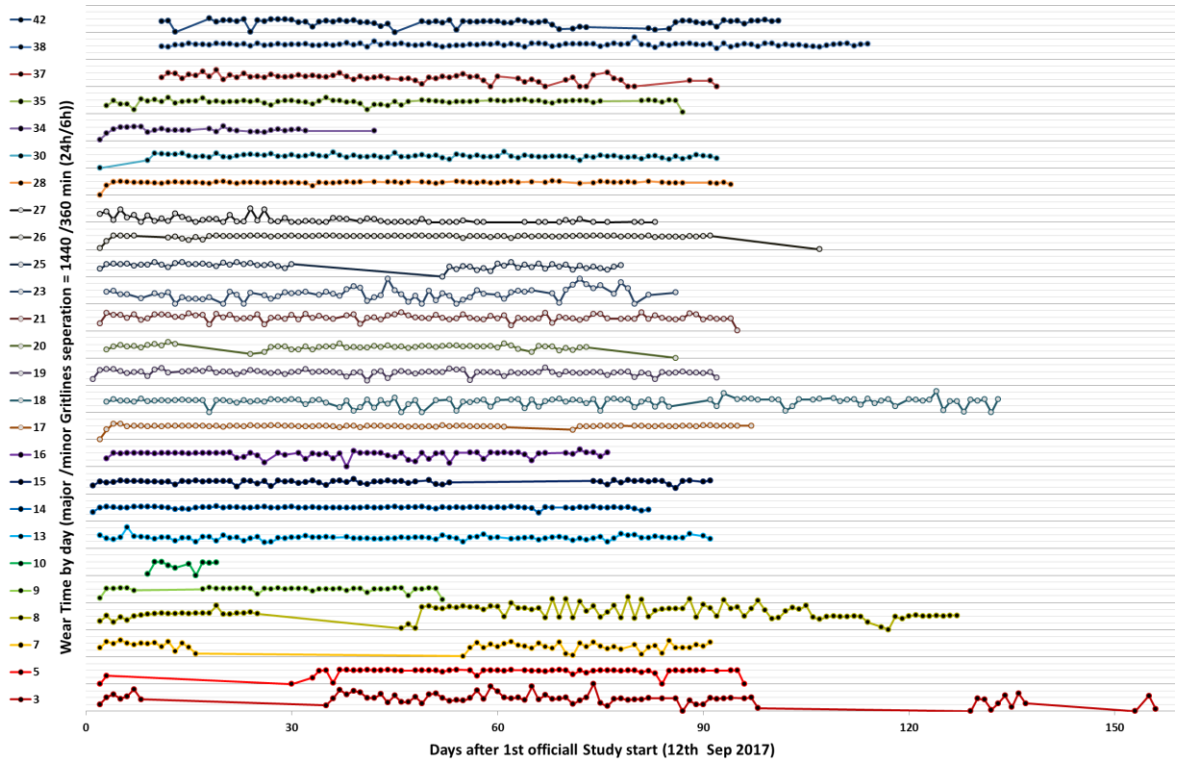


Figure 76 - ***Amount of time the watch was worn every day in minutes****. Circles indicate days of wear time.*

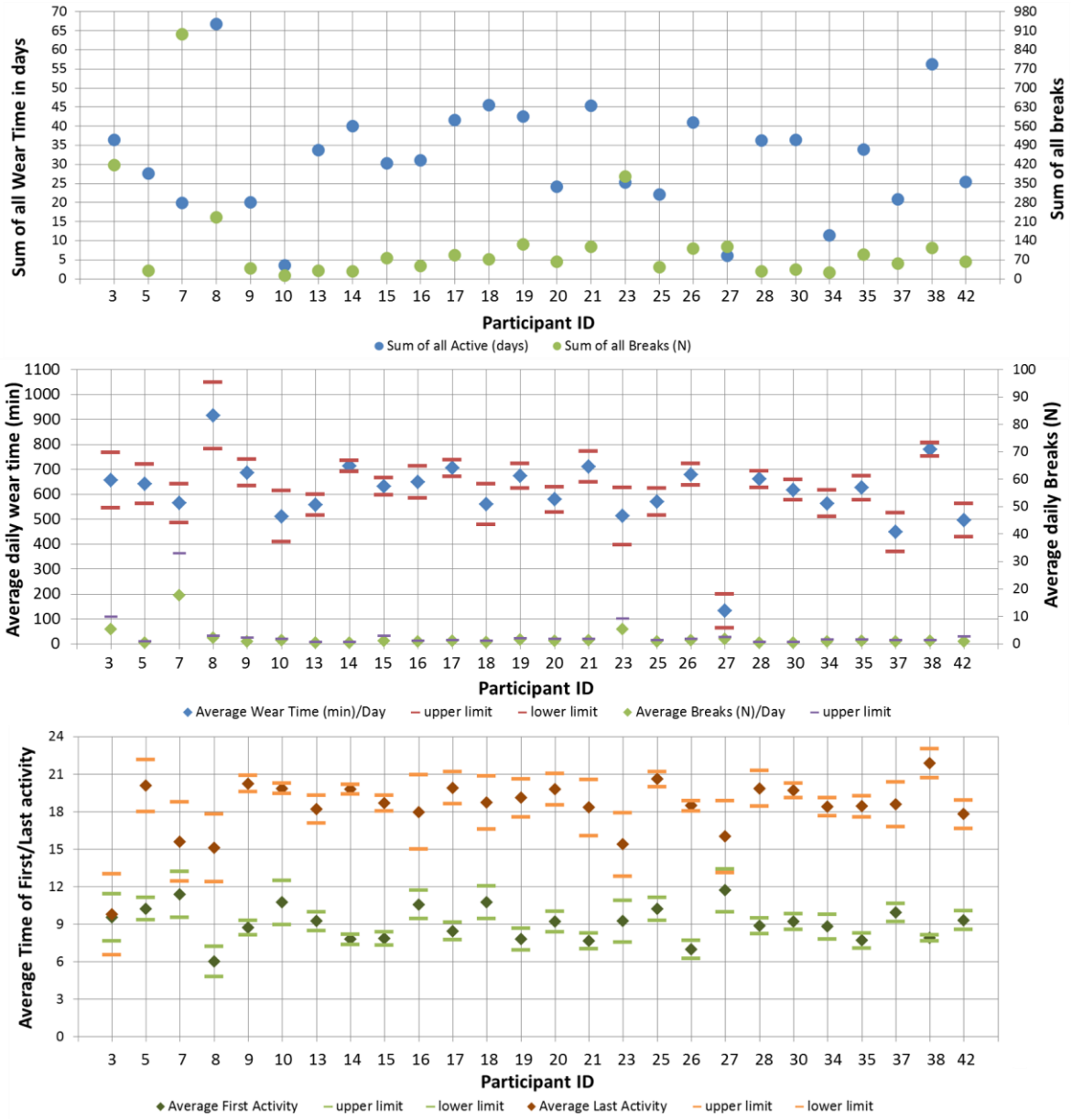


Figure 77 - ***Average daily wear time, wear time interruptions ("Breaks") and average time of first wear episode and last wear episode of the day****.*

Some participants appear to differ strongly in their individual pattern of wear time from what would be expected. For example, Participants 3 and 7 exhibit very unpredictable wear patterns, from which start and end times can hardly be inferred (see Figure 78). On the other hand, participant 8 seems to have worn the watch overnight and loaded it in the morning and evening (see Figure 72). This behaviour makes it impossible to choose an appropriate value for the "first" and "last" activity of the day. Therefore participant 8 has some days on which the "first" activity is judged as around 8 am in the morning, but their last activity is noted as 6 am in the morning of the following day. Consequently, this causes the value assigned to each day's total wear time to fluctuate between around 700 min (~ 12 h) and 1250 min (~21 h) in the time range of the 30th October and 25th December (see Figure 76). On 8 days the wear time is larger than 1440 min (24h). These days are usually preceded and followed by days with around 800 min (~13h). In reality, the participants wore for around 1100 min (~ 19h) a day.

Opposite to participant 8, participant 27 affects the average wear time by only having small one minute episodes of activity on most days. This pattern of data collection under ~ 10 min at random times of the day is marked in green in Figure 78.

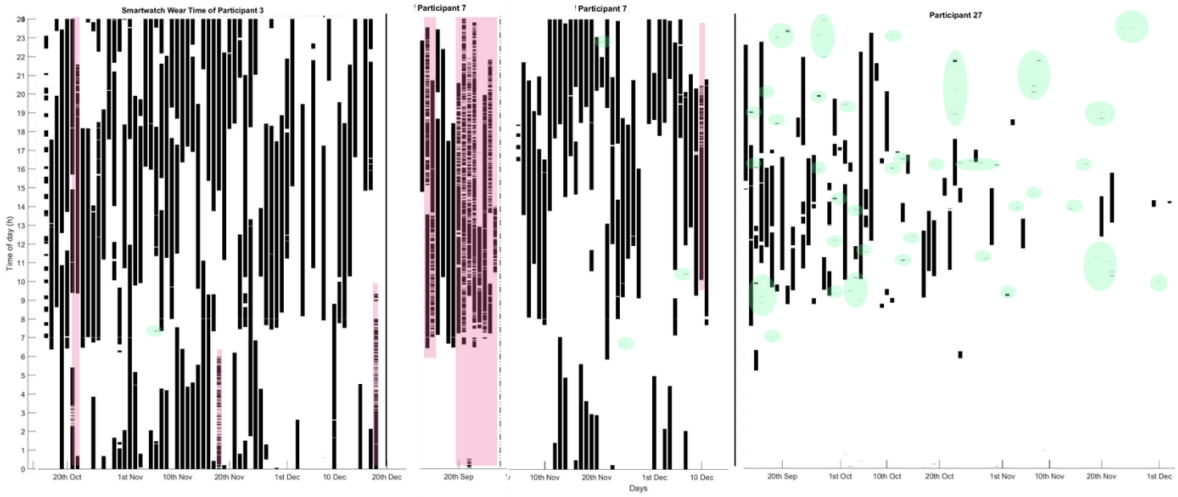


Figure 78 - ***Wear Time examples of participants 3, 7 and 27.*** *Green: data episodes smaller than 10 min, Pink: multiple short data collection episodes.*

For some participants the smartwatch seems to have problems recording continuous data, resulting in the high spikes of breaks as shown for participant 7 (highlighted in pink in Figure 78). However, for most participants, only one or two days display this pattern. Participants 3 and 23 are the only other participants where at least 3 days are affected with short breaks over at least an hour of data collection. Excluding these three participants reduces the breaks to an average of 0.87 a day and reduces variability (STD = 1.16, 95% confidence interval (CI) [0.40, 1.35]).

Lastly, there are two participants with less than 10 days' worth of data collected (see Figure 76). For participant 27 the reasons are already explained above, but participant 10 most likely simply stopped using their watch after their holidays (see Figure 75).

7.8.3. Pain Score Results

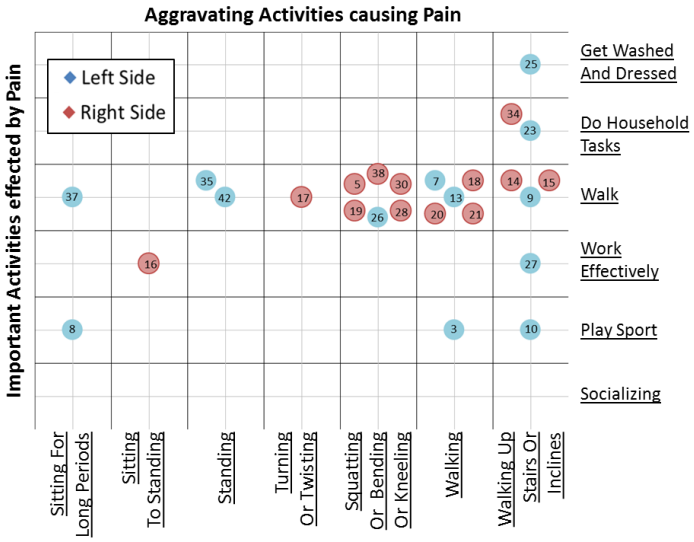


Figure 79 - ***Participant selection of important and aggravating pain level questions and side of knee most affected.***

There was an even divide between participants with the left (N = 13) and right (N = 13) knee as the most dominant side of their condition. Most participants selected walking as the most important activity affected by pain (N = 18, 69%) and as the activity that causes the most aggravating pain (N = 6, 23%). More specific information about the selection of "important" and "aggravating" activities by the participants can be found in Figure 79. No individual indicated "socializing" as their most important activity impacted by pain.

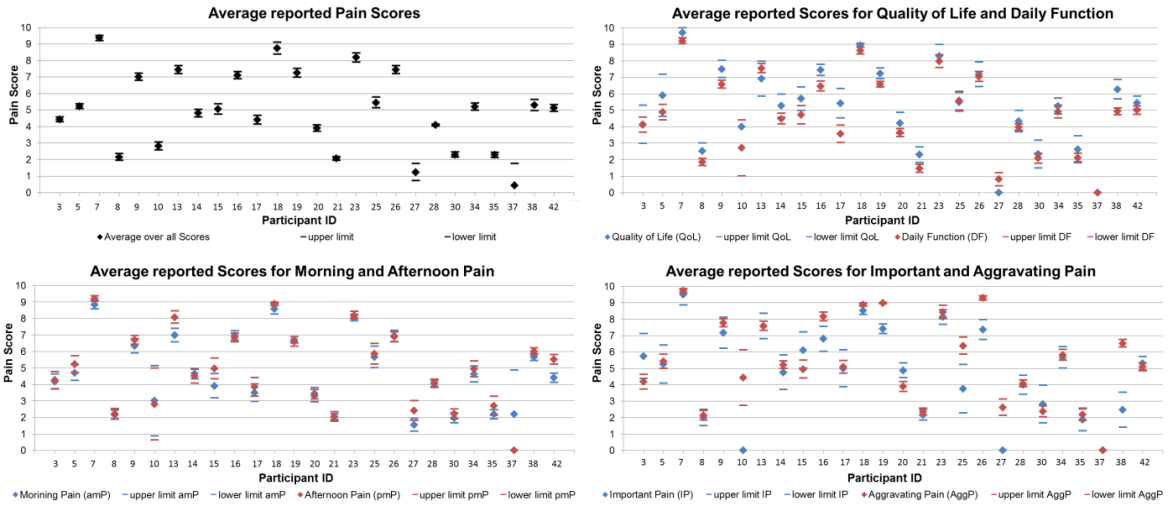


Figure 80 – ***Average pain level scores****.*

The average pain scores over all participants are 5.38 (STD = 2.34, 95% CI [4.46, 6.30]) for Quality of Life, 4.65 (STD = 2.38, 95% CI [3.70, 5.60]) for Daily Function, 4.74 (STD = 2.22, 95% CI [3.89, 5.59]) for Morning Pain, 4.93 (STD = 2.40, 95% CI [4.01, 5.86]) for Afternoon Pain, 4.78 (STD = 2.77, 95% CI [3.71, 5.85]) for Important Pain and 5.44 (STD = 2.66, 95% CI [4.42, 6.47]) for Aggravating Pain. The average scores for each participant can be found in Figure 80.

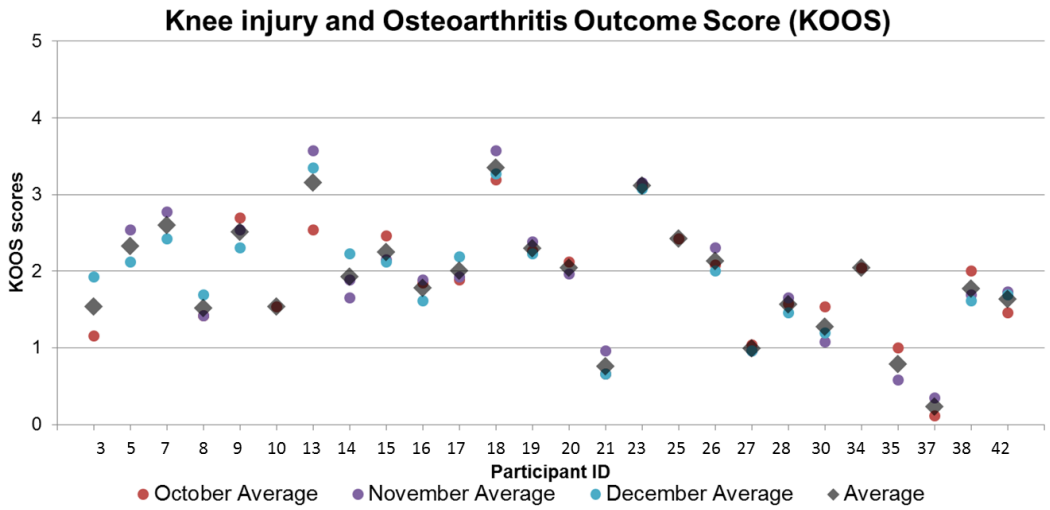


*Figure 81 – **Average KOOS scores**.*

An overview of the reported KOOS scores averaged for each month is reported in Figure 81. Overall, participants reported an average KOOS score of 48.22 (STD = 17.21, 95% CI [41.60, 54.83]) on the Pain subscale and 56.48 (STD = 19.49, 95% CI [47.18, 62.17]) on the Activities of Daily Living subscale.

7.9. Results – Step Parameter Extraction

For the Methods of the GDA and SPEs please see Chapters 4 and 5. When interpreting the data we should consider that participants 7 and 10 lacked a substantial amount of sensor data and participants 25, 34 and 37 had not filled out many pain score questionnaires. However, we decided against excluding them from the analysis.

7.9.1. Step Parameter Distribution

We looked at each gait episode of each participant that contained at least 15 steps and extracted the weighted mean of the episode with the number of steps taken within the episode. The step parameters for the length of the episode (`Epis_Epi_min`), the acceleration range of the gait cycle (`Epis_allGait_AccRange`), the step rate (`Epis_SPE_RateMin`), the symmetry of the step time (`Epis_Symmetry_TimeSec`), the symmetry of the acceleration range (`Epis_Symmetry_AccRange`), and the variance of the acceleration range (`Epis_Variance_AccRange`) were selected for analysis.

We created a histogram with normalized probability for each of those variables for the whole dataset of each participant and displayed them in Figure 82. We can see that the distribution of episode length is very similar between participants, and skews toward smaller episode lengths. Participants 18 and 34 stick out for the increased number of shorter episodes. The big difference in distribution displayed by participant 10 is most likely due to the small amount of data present for them. For all participants, the majority of episode lengths fall between 0.25 -0.5 min and 0.5-0.75 min.

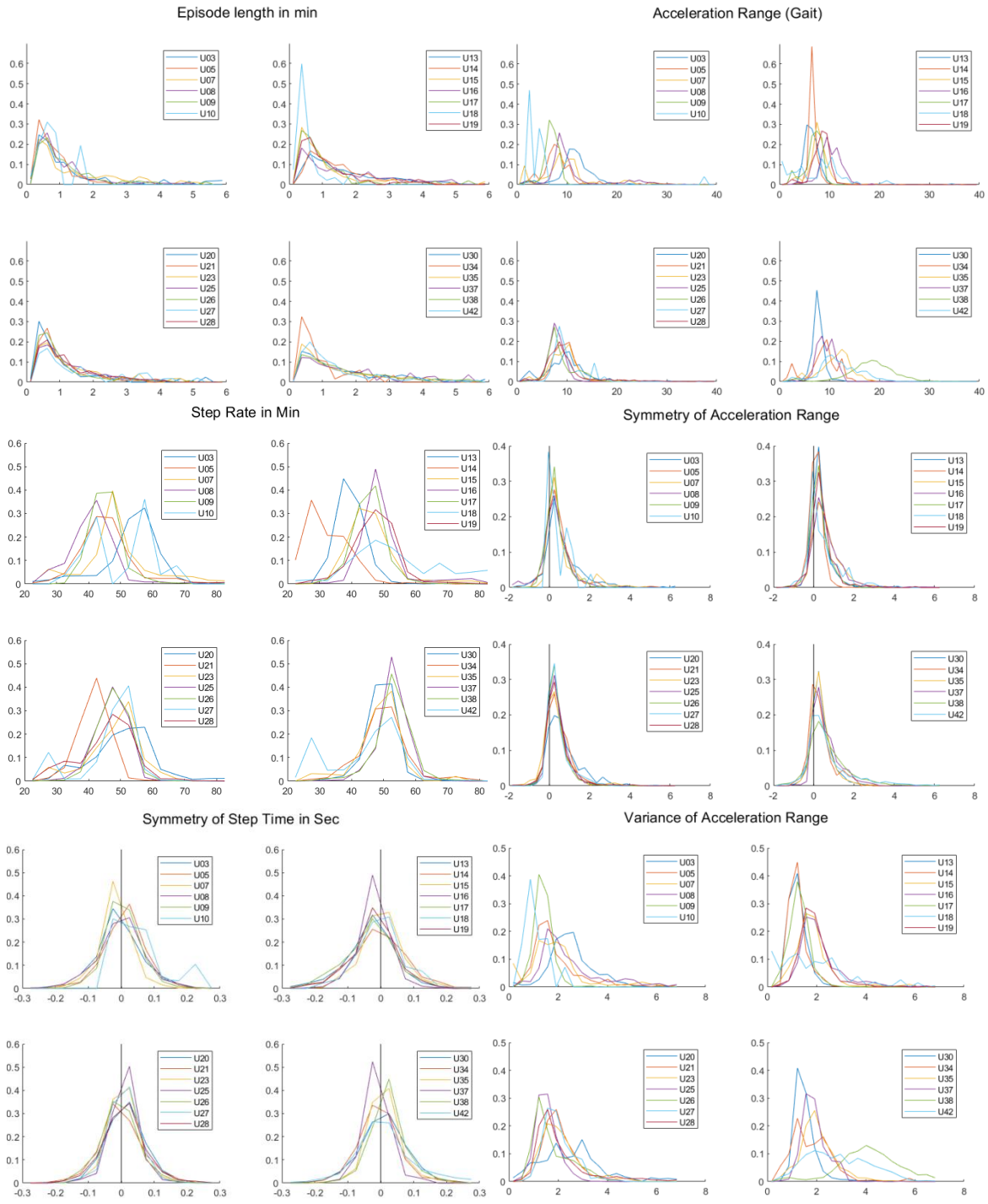


Figure 82 - ***Histograms of the individual Participants extracted Step Parameters with normalized probability****.*

The acceleration range displays a much starker difference between participants. Most notable is the high amplitude of participant 38, but we can also distinguish participants 9, 13 and 30 from participants 3, 16 and 35. They show stark differences in the force applied while walking. The difference between participants is even starker when looking at the step rate. There seems to be quite a divide between participants whose step rate is closer to 40 steps a minute and participants with a step rate of around 50 to 60 steps a minute with the latter being closer to the average of a healthy individual. Interestingly, the data is not normally distributed but shows a bimodal distribution for participants 27 and 42.

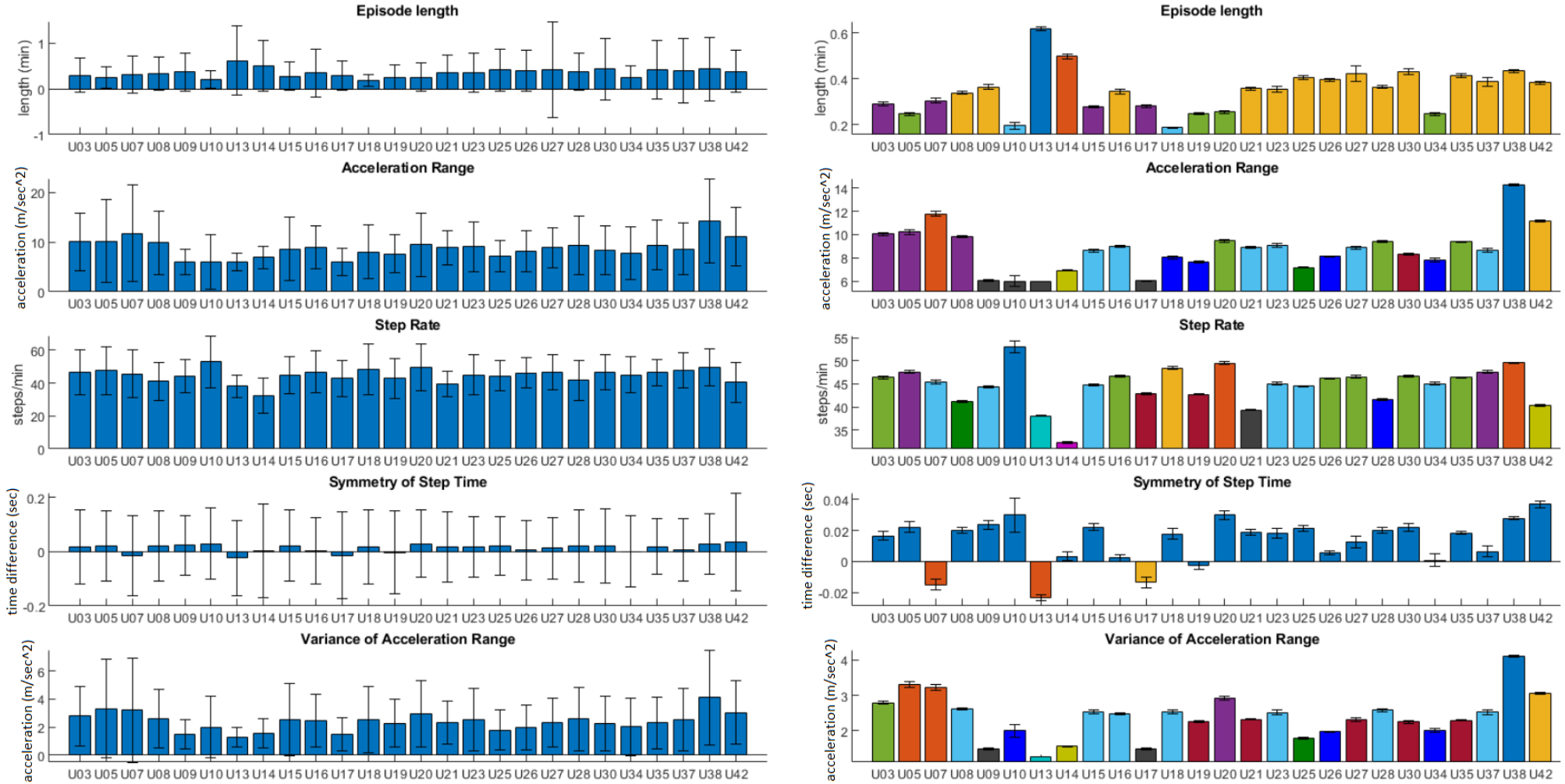


Figure 83 – ***Bar Graphs of the Mean and STD*** *(left)* ***and the zoomed in Mean and Standard Error*** *(right)* ***of the individual KOALAP participants selected Step Parameters.*** *The participants were grouped by colour on the right according to the overlapping standard error ranges.*

The symmetry of the acceleration range on the other hand looks very normally distributed and not a lot of difference between participants can be seen. We decided to drop this parameter in favour of the symmetry of step time, which shows that participants seem to have a tendency towards positive (see participants 5, 25 and 38) or negative (see participants 7, 16, and 37) symmetry. It should be noted here, that for symmetry values closer to 0 speak for higher symmetry of the gait, while values further away from 0 speak of an increase in asymmetry. However, it is hard to judge if this tendency will turn out to be significant. Lastly, the variance of acceleration range is another promising looking variable. In particular, participants 13, 14 and 17 show a stark contrast to participants 15, 16 and 19. Furthermore, participant 38 sticks out again for their distribution around higher variance. Most of the participants are normally distributed, with participants 18, 20 and 34 being outliers.

The mean, STD and standard error can be seen in Figure 83. When looking at the standard error, we coloured the participants whose standard errors overlapped into groups with the same colour. As we already assumed when looking at histogram data, the most distinct groups can be found for the acceleration range, the step rate and the variance of the acceleration range. On the other hand, most participants had a positive symmetry of step time and were not easily sorted into non-overlapping groups, mostly due to some of the participants' high standard errors.

7.9.2. Step Parameter Correlations with Pain Scores and Time of Day.

We ran a correlation with each participant's selected Step Parameter and Pain Scores and the Time of Day (see Figure 84). Overall, it is hard to interpret a meaningful correlation between time-of-day and step parameters or pain scores. Depending on the participant, correlations can be either positive or negative, however, about half of the participants have correlations that are significant (128 out of 260). We selected the morning and evening pain reports (Pain_Closest_oAMoPM) to correlate with the step parameters (see Figure 85). We did find some participants to show significant correlations, however, the directionality of these correlations varied.

Figure 84 – ***Correlation of Step Parameter and Pain Scores with Time of Day****. Red = not significant; green = significant.*

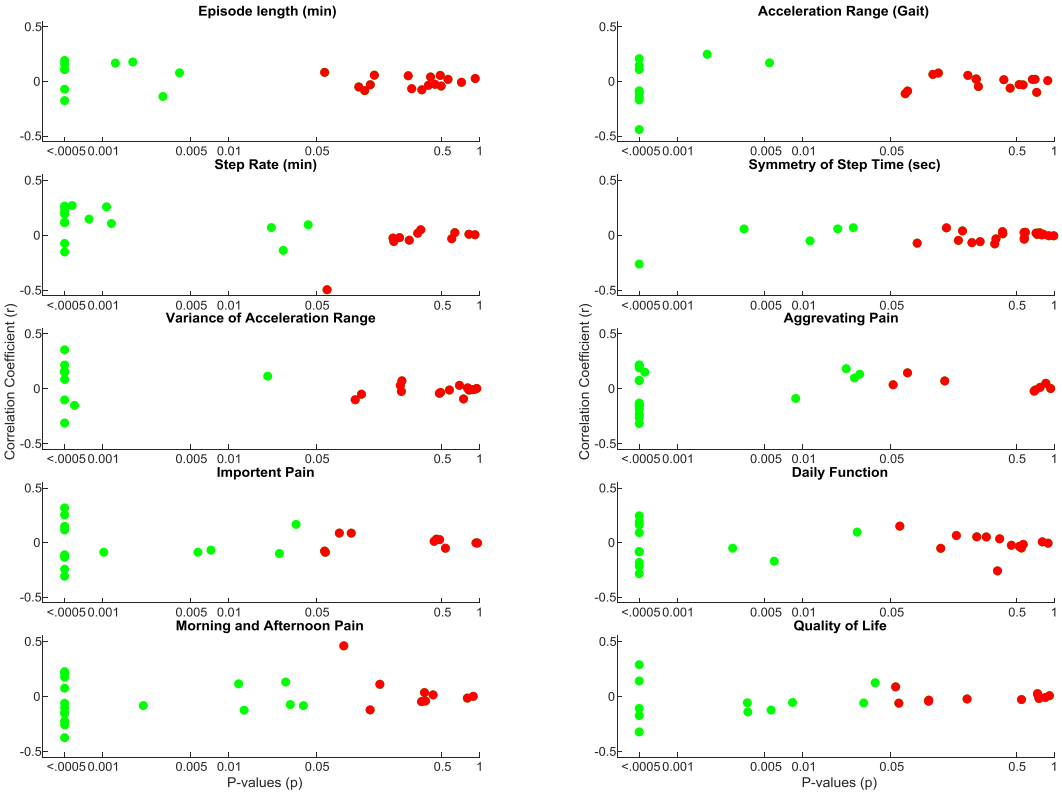
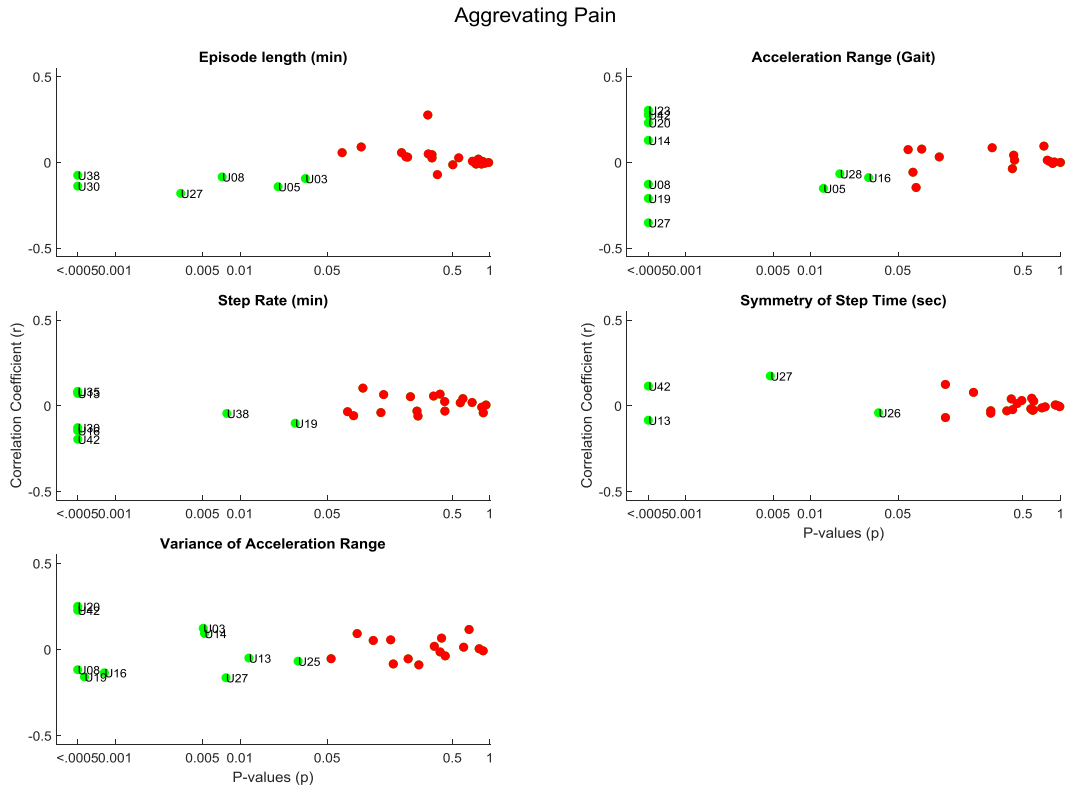
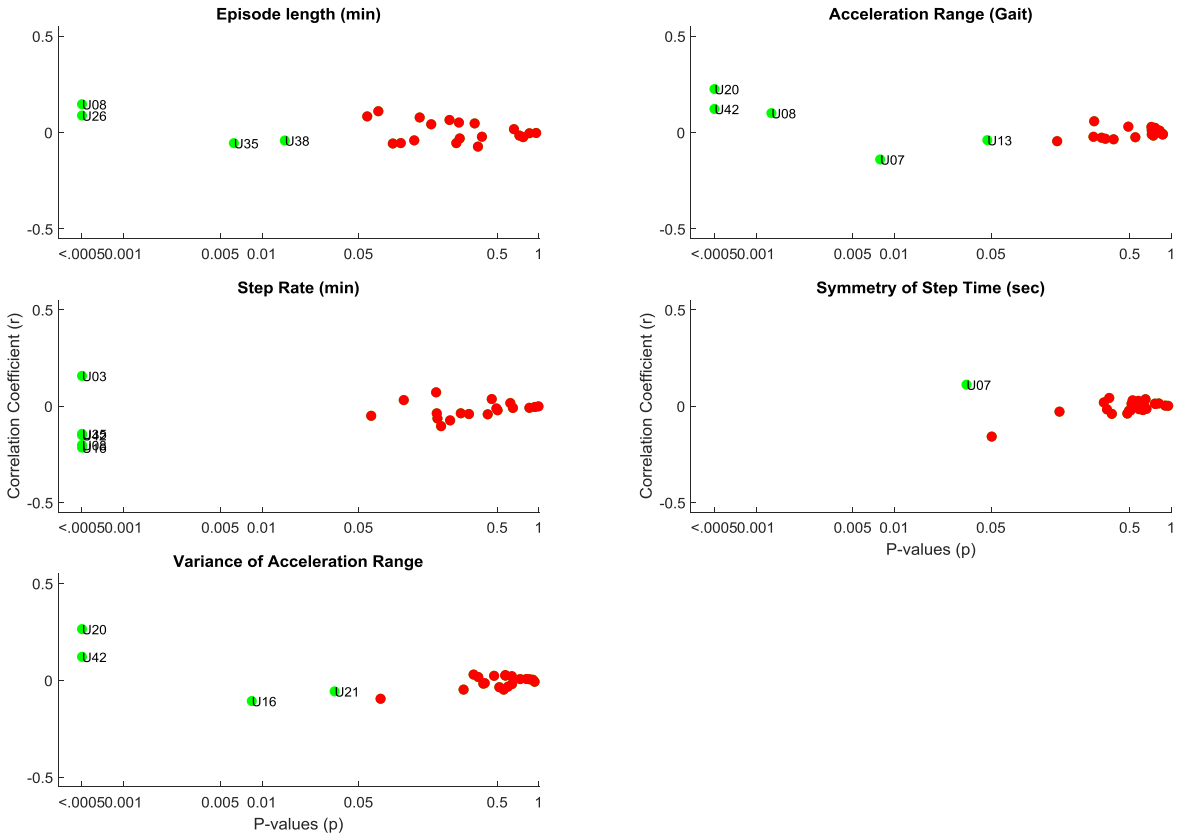


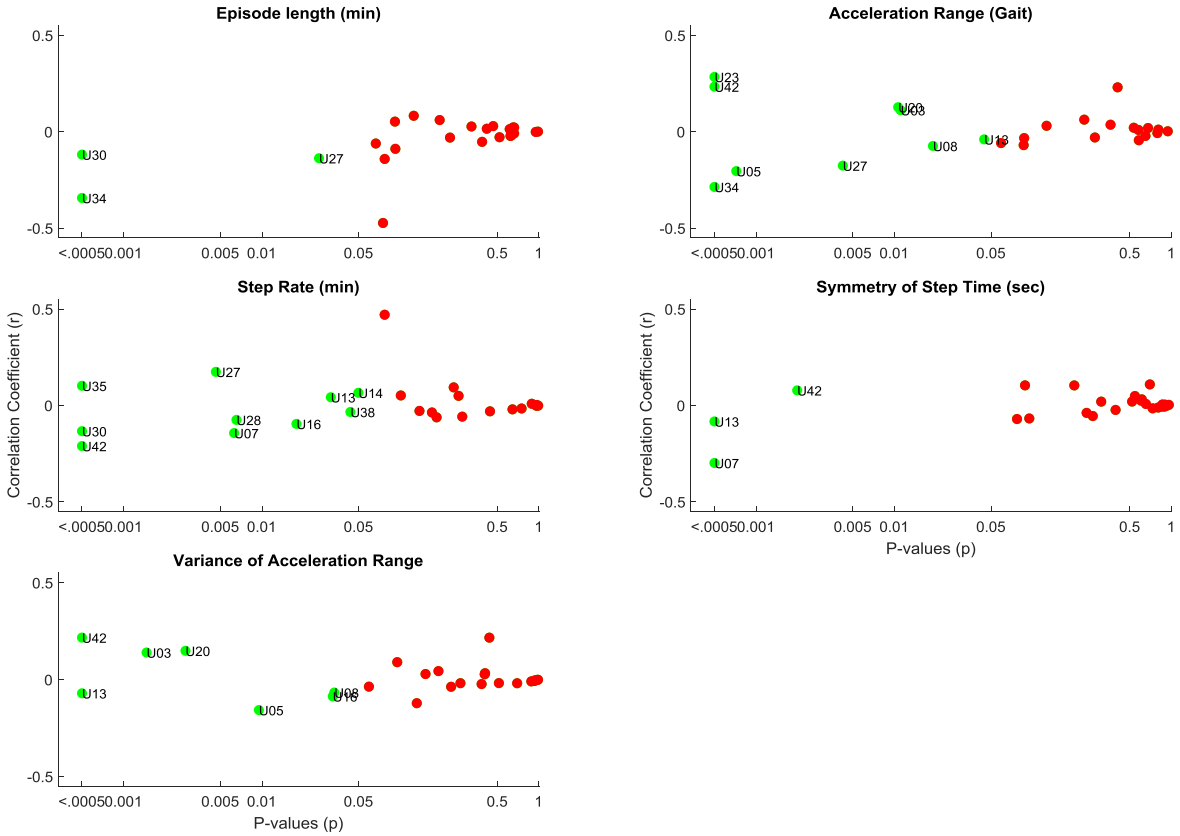
Figure 85 - ***Correlation Coefficients correlating Step Parameter and Pain Scores****. Red = not significant; green = significant. Continued on the next pages. ↓*



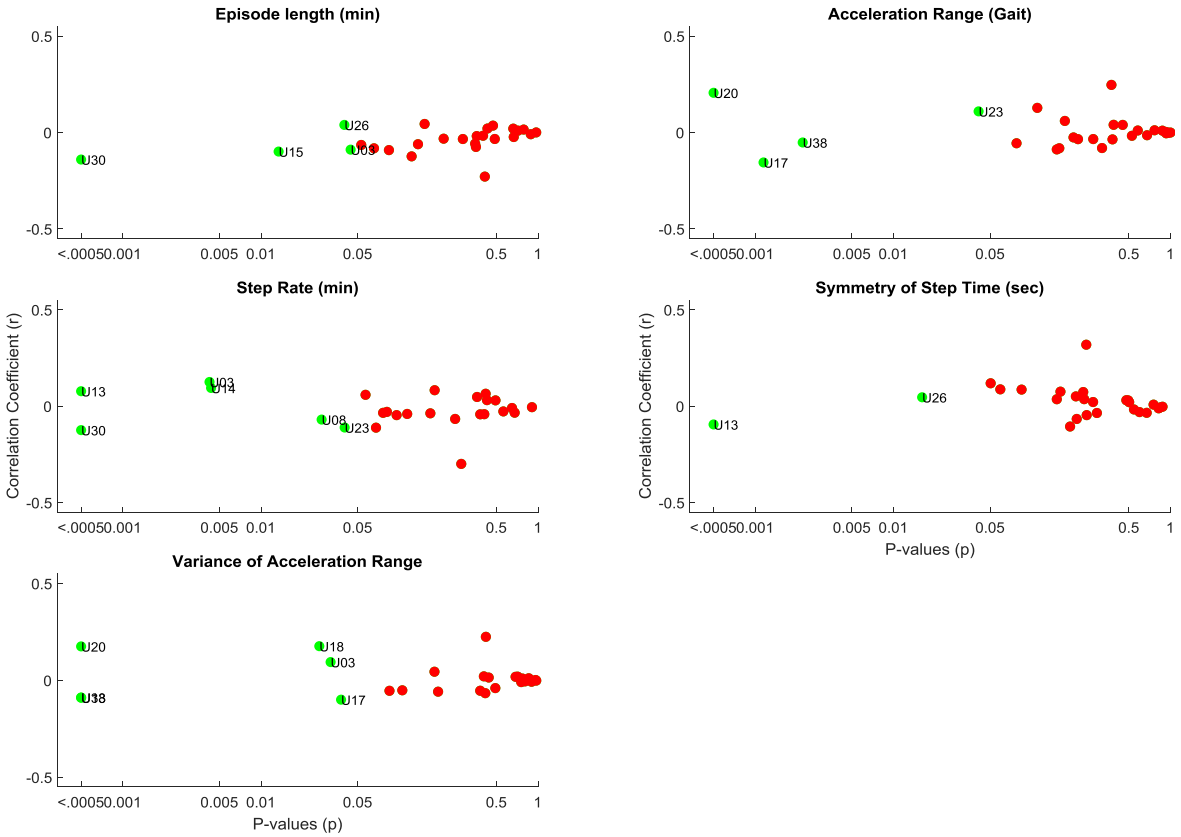
Importent Pain



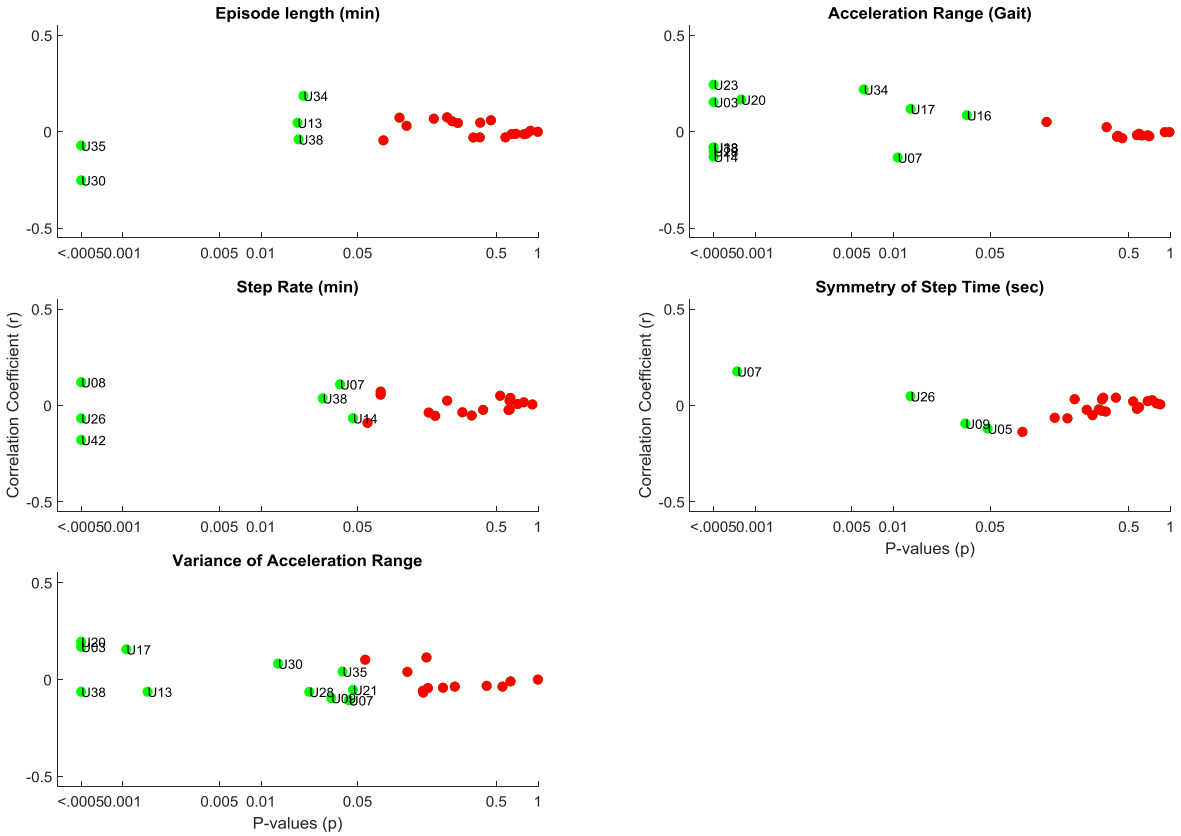
Daily Function



Morning and Afternoon Pain



Quality of Life



7.10. Conclusion – Step Parameters and Pain

We were not able to directly link an increase in pain scores to a step parameter's increase or decrease. However, we were able to show that some of our step parameters were distinctive for individual patients and correlated to individual patients' pain scores. Looking at the correlation patterns that we extracted from the KOALAP data, we have to conclude that our step parameters are most likely only helpful in accessing the behavioural pattern of patients on an individual level.

However, before we discuss the implications of the KOALAP data analysis for our algorithm and the development of the potential smartwatch app as a diagnostic tool, we will have to discuss how the development of the algorithm might have influenced the data we extracted. The extraction of step parameters was optimised using the SRDS, which reflects walking behaviour of a healthy participant. It is therefore likely that only walks with less impaired gait patterns where correlated with the pain scores reported by the participants. Episode length was mostly negatively correlated with morning and afternoon pain, quality of life, daily function and pain during aggravating activities. That means that episode length was more consistently negatively correlated with pain measures than the other parameters. Since we weighted the step parameters measures for each episode according to the length of the episode, longer sustained higher quality walks were biased to step parameters favourably. Episode length is not affected by the weighting, since it itself reflects episode length. This measure is therefore helpful in judging the amount of gait the participant engages with, and the ability of the participant to sustain it. In consequence, the algorithm would benefit from a better way of weighting extracted step parameters of each step within the output data. A better way to measure the amount the patient engaged in measurable walking behaviour would also be beneficial, before correlating the extracted step parameter with reported pain scores. If during pain flares, patients' engagement in walking behaviours is reduced and only higher quality walking episodes can be identified, then the algorithm's bias towards better quality walking might result in step parameters wrongly reflecting higher functioning. On the other hand, during periods of less excruciating pain patients might engage more in walking behaviour and are more likely to exhibit walking behaviours that are of a high enough quality to be pick picked up by the other, which might bias the step parameters to reflect decreased functioning.

Our prediction that we can generally relate our step parameters with a decrease in performance throughout the day could not be supported. For patients with OA we would have predicted the performance would decrease during the day, however, the directionality of the correlations speaks against this conclusion. We furthermore have to address that none of the correlations that were found were particularly strong, and therefore the usefulness of the application of the parameters for the tracking of patients' functions on a large scale would be limited. Another problem is the directionality between pain and walking behaviour. We were not able to identify if patients walking behaviour and walking quality decreased because of pain, or if the increase in walking behaviour leads to an increase in pain in the participants. It could still potentially be used to track a patient's individual disease progression; this would however have to be done over a large amount of time. First, at least a few months of data should be collected for a patient to establish their individual, specific parameters of interest. If the patient's established step parameters can predict their submitted pain scores, this could then be further examined and further tests could be conducted to

identify correlations. The high amount of significant correlation with various pain scores promises such an attempt to be fruitful. If we can link an increase or decrease in walking performance to the participants' pain scores, then our algorithm could be a helpful tool in personal disease tracking.

However, this tracking tool would most likely only be supplementary and only financially beneficial for the participants if they already possess a smartwatch to run the data collection on. Our algorithm’s properties such as having low processing times and only relying on the recording of accelerometer data can be seen as a positive since it can likely be adjusted to not demand too much of the smart watch's battery capacity. Collecting data from multiple sensors was quite demanding on a smartwatch during the KOALAP study, and the smartwatch had all other functionality disabled. An app aimed at consumers or patients would have to minimise battery consumption. Consequently, the algorithm should have options focusing on some step parameters over others, after enough test data was collected from a consumer to determine the most appropriate.

8. UK BioBank

8.1. Abstract

The UK BioBank provides a large collection of accelerometer data from a diverse participant pool. We identified 17,422 participants of interest that were sorted into eight groups: participants with osteoarthritis (OA), rheumatoid arthritis (RA), gout, other rheumatic and musculoskeletal diseases (RMD), and a matched control group for each of these RMD groups. We compared the results of the following step parameter between the RMD groups with their control groups: Walking Episode Length in minutes, Acceleration Range of gait cycles in m/sec^2, Step Rate in minutes, Symmetry of Step Time in minutes between the left and right step cycle and Variance of the Acceleration Range in m/sec^2. The parameters' mean and standard deviation (STD) differed only slightly between groups of participants with different RMDs and the selected control groups. Conducting a paired t-test between RMD patients and their matched controls revealed a significant difference for most step parameters with mostly small effect sizes. We conclude that age, sex and Townsend scores, which we used to match the control groups, have a larger effect on individual participant step parameters than their RMD. If these participant characteristics are accounted for, we do find our step parameters to reflect patients' functionality in large datasets. Therefore we consider the step parameter extraction (SPE) algorithm useful to extract step parameters from various gait behaviours.

8.2. Introduction

Measuring physical activity has historically been limited to questionnaires. These can be imprecise because they solely rely on the patients' recall ability, which can be poor. More objective measurements are increasingly possible using accelerometers. They also have the advantage of being passively recorded and rely less on patients' active input. Gait patterns can be picked up by algorithms, meaning walking and gait might be used to identify changes in mobility in people living with RMDs and act as a marker of disease progression in contrast to self-reported data. They can also be used to assess the effectiveness of treatment interventions.

Analysing walking and gait data collected by accelerometers can result in measurements of disease severity that are cheaper, more accessible, more ecological and less time consuming for the clinician. The use of devices integrated into watches would be a more comfortable way of measurement for patients due to the potential familiarity. However, the ability to interpret wrist-collected data and its usefulness for accessing musculoskeletal patients' gait outside of the laboratory still needs to be assessed.

Developing an understanding of which gait parameters differentiate RMDs from the general population, or identifying differences between or within RMDs, can lead to future applications for self-management, clinical care and research (such as early diagnosis, support symptom tracking and personalised digital physical activity interventions), providing an objective surrogate measure of disease severity and progression without relying on self-reported measures or clinically assessed functional mobility assessments, supporting clinical care as well as acting as a digital

biomarker for research. The inclusion of accelerometers in consumer devices with increasing market penetration makes this a particularly exciting and expanding opportunity.

We aim to investigate if RMDs and their painful symptomatic affects the walking behaviour and walking ability of patients negatively, either through a decrease in mobility or through pain avoidance. Comparing RMD patients with controls, we would expect to see shorter and less frequent engagement in walking behaviour and negatively impacted gait parameters (i.e. slower, shorter and more variable steps).

8.3. Methods

8.3.1. Estimating Participant Groups

The UK BioBank collected 103,712 accelerometer datasets from the 10th of May 2013 till the 3rd of December 2015 (see Figure 81). A previous study determined 96,600 of these datasets are valid for data analysis (Doherty, et al., 2017). Of the 502,649 UK BioBank participants, 5,657 reported Rheumatoid arthritis (2,849 with medical treatment) (Siebert, et al., 2016), 30,727 cases of Osteoarthritis can be found (10,083 with a hospital diagnosis) (Zengini, et al., 2017) and 2,432 cases of gout (1,861 confirmed by biomarker) (Cadzow, Merriman, & Dalbeth, 2017) Alone for these RMDs we can estimate to find around 1,100 self-reported RA, 5,900 self-reported OA and 2,200 of gout cases in the accelerometer data (~ 550 medically treated RA cases; ~ 1,900 hospital diagnosis of OA; ~ 1,700 biomarker identifications of gout).

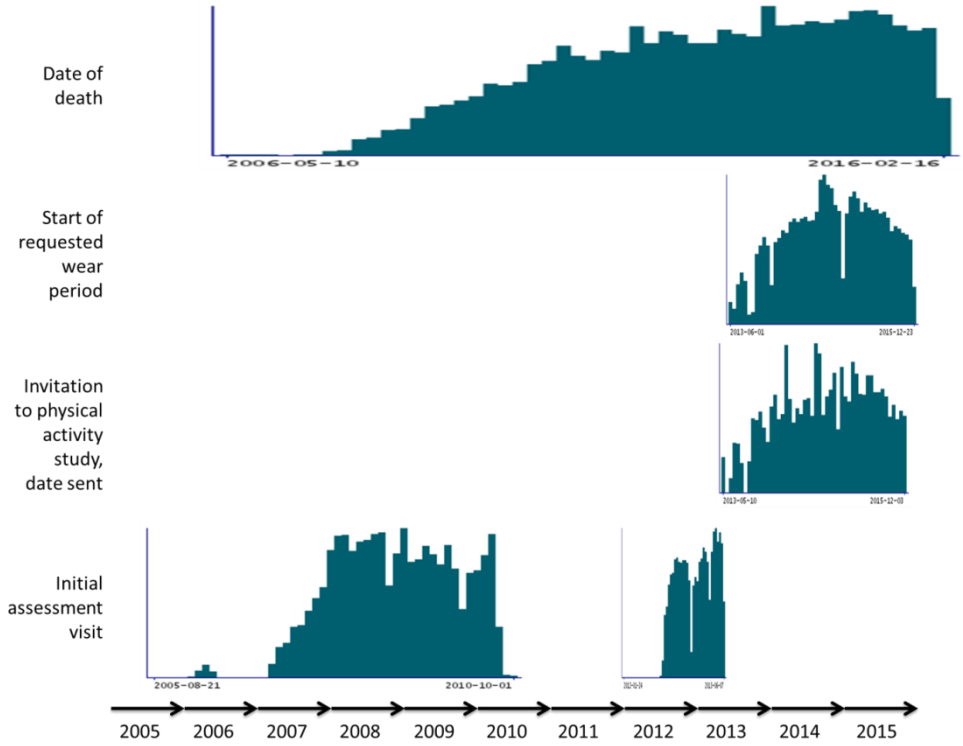


Figure 86 - ***Data Collection Timeline in the UK BioBank*** *(UK BioBank, n.d.).*

8.3.2. UK BioBank Application and Wrangling Process

We applied for the dataset in February 2019 and our application was approved in May 2019. However, signing the Material Transfer Agreement (MTA) caused some problems. Since one of our supervisors (Dr Little) was located at Aston University, we needed an MTA for both the University of Manchester and Aston University. We had problems securing a secondary MTA contact for Aston University. After we finally secured one in September, Dr Little moved to Birmingham University and the problem repeated. We resolved the issue in the end by taking him off the application as a collaborator in January 2020. Consequently, we were not able to involve him when examining the UK BioBank data.

Wrangling the dataset to extract the participants' RDM groups and to select a control group proved difficult for us. The author's supervisor, Dr Mark Lunt extracted the patient RMD group information. We started using the Stata do-file created by Dr Lunt in July 2020 and used it to extract the IDs of participants with RDM conditions and assign them a control group.

We then ran into some technical issues with downloading the dataset. The accelerometer data files are around 250MB each and we identified 17,544 participants for our dataset. This results in a required space of 4,386,000 MB of storage for the raw data alone. Proper conduct would require the data to be saved in raw format and saved after pre-processing, in addition to the two output files of our GDA and SPE. We looked at around 1 h of data from the SRDS and it was 1,474 KB raw and 9,879 KB after pre-processing. For the selected participants, that would mean around 30TB of storage space of pre-processed data alone. Since we only obtained 9 TB of storage data, we decided to enable the GDA code to load the raw accelerometer data, pre-process it, run the GDA, save only the identified gait data and delete the raw data file.

When first attempting to download the data we ran into some administrative issues. Overall it took us till the end of February 2021 to secure enough secure space in the Incline server within the University of Manchester and to start successful downloading attempts of the accelerometer dataset. First attempts to analyse the accelerometer dataset were taken in May 2021 and in June 2021 the GDA algorithm was finished and adjusted to run on the UK BioBank accelerometer dataset. The SPE was run at the end of October 2021.

In summary, while both the support team of the UK BioBank and the University of Manchester IT team were very helpful and access had clearly been optimised for easy handling, solving the challenge of downloading and wrangling such large datasets will necessarily cause unexpected delays since all research project face unique problems. We remember the advice that was given to us when we first decided to use the UK BioBank dataset: “It will take at least two years to figure everything out.”

8.3.3. Participant Selection

Wrangling of the participant selection was done in STATA. Dr Mark Lunt reported the following about the participant group identification:

“The first step is to identify individuals with various types of RMDs, using the methodology of Cook et al (2018)*. Patients answering “yes” to the question ‘Has a doctor ever told you that you have had any other serious medical conditions or disabilities?’ were then presented with a list of possible conditions: those who selected Osteoarthritis, Rheumatoid Arthritis, Gout, Psoriatic Arthritis, Ankylosing Spondylitis or Systemic Lupus erythematosus were assigned to the corresponding patient groups. Those who did not select any of the above conditions were included in the pool of potential controls. Expected sizes for each of these groups were: OA (595 - 1081), RA (306 - 1087), Gout (952 - 2229), PsA (93 - 166), AS (134 - 241) and SLE (60 - 107). Two controls were matched by age, gender and BMI to each subject with a musculoskeletal condition.”*

From the resulting dataset, we first removed participants that requested to withdraw from the UK BioBank study during the time of our analysis. In addition, we had to remove duplicates from the data. This occurred when participants went to assessment centres multiple times and reassessment results were saved as a new line of data. In these events, we kept the data from the initial assessment. Then we looked at the RMD participants’ sex, year of birth and Townsend score and matched them to controls of the same sex and year of birth and who had the closest Townsend score to them. Each RMD and Control Pair was assigned a matching number.

We identified a total of 17,422 Participants (8,711 RMDs and Controls each). The OA group consisted of 6,459 Participants, the RA group contained 833, the Gout group had 1,121 participants and the “other conditions” group was made up of 570 participants.

We used Matlab to divide the participants into 31 groups with 562 participants each and to write them in the appropriate format into a txt-file, which is required for batched loading of the participant data in the UK BioBank. Loading the groups into 31 different folders helped with keeping the output files navigable since the computer would not freeze when you opened the output folder which in

turn ensured that data loading, processing and subsequent deletion of the data could be done in steps and that we were able to process multiple folders at the same time cutting down considerably on the time needed to process the full dataset. After each data handling step and algorithm application step we made sure that all participants' IDs were successfully downloaded and processed. Overall, initially downloading the data was the most time-intensive process.

8.3.4. Accelerometer Data Analysis

The sensor used for the UK BioBank accelerometer dataset collected was the AX3 Axtivity sensor, a research-grade deceive. Data was collected continuously from participants for 7 days, at 100 Hz with an acceleration range of ±8 g (Brage, et al., 2016).

For the Methods of the GDA and SPEs please see Chapters 5.3 and Chapter 6.3. To recap, the weighted mean of all gait episodes of each participant was taken for each step parameter. The Step parameters analysed were:

- Episode length (in min)
- Acceleration Range (in m/sec^2) of the Gait Cycle
- Step-Rate (in steps/min)
- Symmetry of Step Time (in sec)
- Variance of Acceleration Rage (in m/sec^2)

We extracted the descriptive statistic of each participant group (RMD OA, RMD RA, RMD Gout, other RMDs, OA Control, RA Control, Gout Control and other RMD Control). Afterwards, we applied a paired t-test to each of the RMD groups and their controls for each selected step parameter.

8.4. Results

Table 18 depicts the descriptive statistics of the various participant groups identified within the UK BioBank database for our UK BioBank dataset. There is not much difference in the average age of the participants, most of them being born around 1948. The RA and other RMD groups were two years older than the whole group on average. The STD of the groups ranged from 6 to 8 years. The Townsend scores were also quite similar between groups with the OA and gout group having an average Townsend score of around -1.75 with an STD of around 2.8 and the RA and other RMD groups having an average Townsend score of around -0.6 and an STD of around 2.9. Notable is the difference between males and females within the group with around twice as many females being present in the OA and RA groups while mostly males made up the gout group.

	All Participants		RMD		Controll	
	year of birth	Townsend	year of birth	Townsend	year of birth	Townsend
N	17422	17422	8711	8711	8711	8711
mean	1948.49	-1.70	1948.49	-1.70	1948.49	-1.70
std	6.54	2.82	6.54	2.82	6.54	2.82
95%CI upper	1948.39	-1.74	1948.35	-1.76	1948.35	-1.76
95%CI lower	1948.59	-1.66	1948.63	-1.64	1948.63	-1.64
Sex female	10264	7158	5132	3579	5132	3579

	OA RDM		OA Controll		RA RMD		RA Controll	
	year of birth	Townsend	year of birth	Townsend	year of birth	Townsend	year of birth	Townsend
N	6459	6459	6459	6459	833	833	833	833
mean	1947.92	-1.72	1947.92	-1.72	1949.96	-1.43	1949.96	-1.44
std	6.16	2.82	6.16	2.82	6.98	2.90	6.98	2.90
95%CI upper	1947.77	-1.79	1947.77	-1.78	1949.49	-1.63	1949.49	-1.63
95%CI lower	1948.07	-1.65	1948.07	-1.65	1950.44	-1.24	1950.44	-1.24
Sex female	4291	2168	4291	2168	572	261	572	261

	Gout RMD		Gout Control		other RMD		other controll	
	year of birth	Townsend	year of birth	Townsend	year of birth	Townsend	year of birth	Townsend
N	1121	1121	1121	1121	570	570	570	570
mean	1948.81	-1.78	1948.81	-1.78	1951.98	-1.61	1951.98	-1.61
std	6.71	2.79	6.71	2.79	8.01	2.92	8.01	2.92
95%CI upper	1948.42	-1.94	1948.42	-1.94	1951.32	-1.85	1951.32	-1.85
95%CI lower	1949.21	-1.62	1949.21	-1.62	1952.64	-1.37	1952.64	-1.37
Sex female	65	1056	65	1056	315	255	315	255

Table 18 - Descriptive statistics of the Participant Groups.

8.4.1. Comparing Patients with Matched Controls

Figure 87 (left) depicts the mean and STD of the step parameters between patients and control groups. At first glance, all RMD groups seem to have shorter gait episodes, lower acceleration ranges, lower step rates and worse symmetry of step time. However, while most participants show an increased variance of acceleration range over the control group, the gout group variance was starkly reduced compared to the control group. In general, we can see that the STD overlaps considerably between RMD and Control, but also between different RMD groups. Looking at the right graph that contains the standard error (see Figure 87, right), we can see that the extracted means of the sample populations vary considerably between groups.

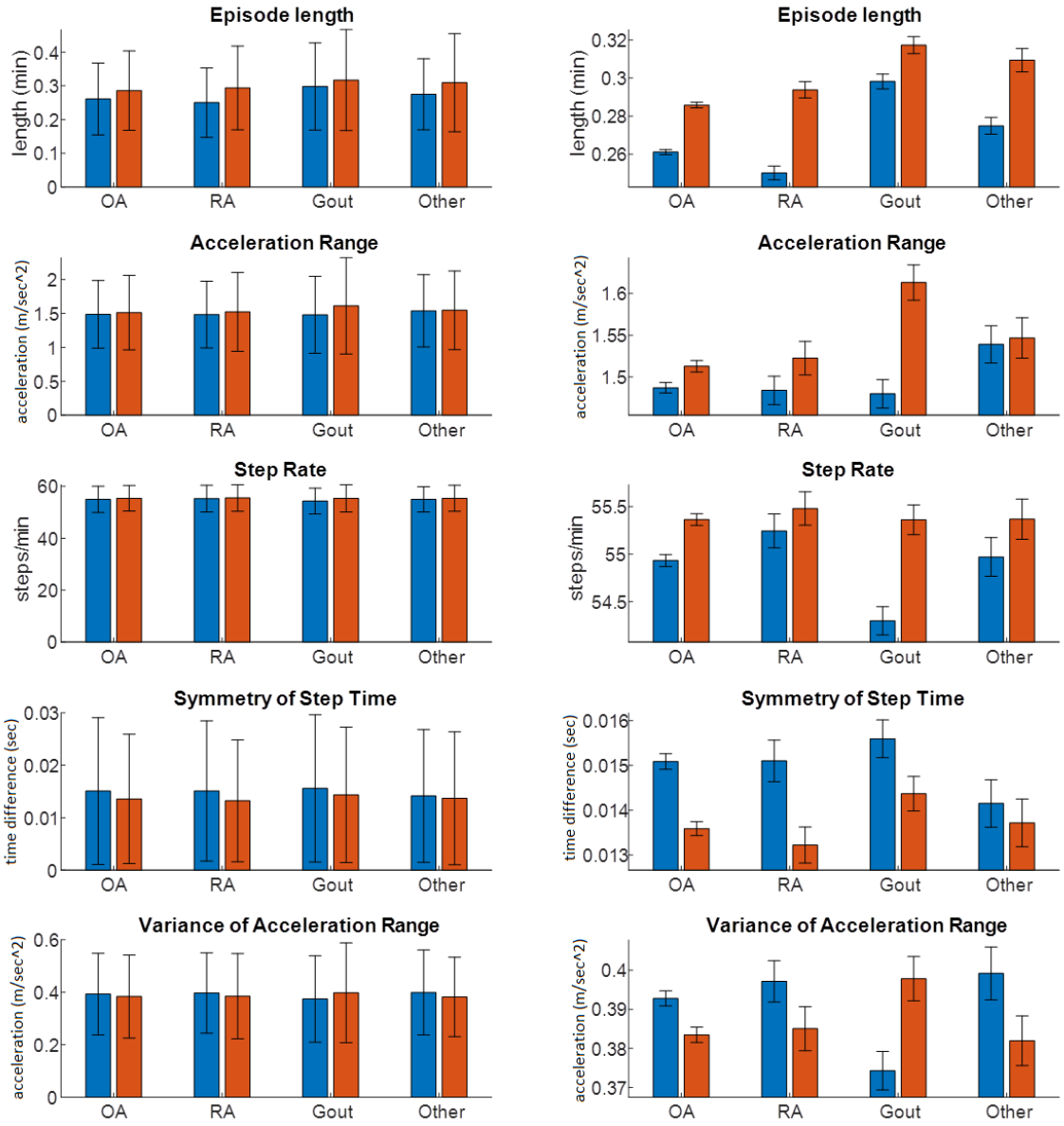


Figure 87 - ***Comparing Step Parameters of Patients with matched Controls****. Left: Mean and STD; Right: Zoomed in Mean with standard error.*

Looking at the right side of Figure 87 and the Table 19 containing the t-test results and the participants' groups' descriptive values for the various step parameters, we can make the following observations:

The episode length of all patient groups was notably different to the controls, with RA displaying the largest difference. Interestingly the gout patient group's episode length was longer than even the average OA and RA control groups. The paired t-test showed significant differences for the OA, RA and other RMD groups, but not for the gout group. The RA and other RMD groups even showed a moderate effect size. All identified means lie between 0.24 and 0.29 min.

The acceleration range significantly differed between the OA and gout groups, but not between the RA and other RMD groups. The acceleration range of the gouts control group is also much larger than any other group and displays the strongest difference from their patient group. All identified means lie between 1.46 and 1.58 m/sec^2.

Similar observations to those of the acceleration range can be made when looking at the step rate. Most notable is the very slow step rate of the gout patient group. The OA and gout groups are significantly different from each other. All identified means lie between 54.48 and 55.89 steps/sec.

The patient groups for OA and RA displayed significantly more asymmetric walking with RA having the highest difference of symmetry compared to their control. All identified means lie between 0.0152 and 0.0167 sec.

At last, the OA and gout groups varied significantly in the variance of acceleration magnitude. Interestingly, while the OA patient group showed a significant increase in variance, the gout patient group displayed the lowest variance of all participant groups. All identified means lie between 0.38 and 0.41 m/sec^2.

Overall the difference between the step parameters for patient and control groups was small, but compared to matched controls often significant. For the OA group, all step parameters tested showed a significant decrease in functionality for the patient group. Only the RA patient group showed a significant reduction in episode length and in step time symmetry. For the gout patient group, the acceleration range and step rate were significantly decreased. Furthermore, they showed a significant decrease in acceleration range variance. For the other RMDs group, we could only find a significant decrease in episode length and a significant increase in acceleration rate variance.

Table 19 – Descriptive Statistics and Results of paired t-tests between Patient and Control Groups. P. = Patient Group, C. = Control Group, md = mean difference

		Descriptive Statistics	**Paired t-test**
		Episode length (in min)	
OA	P.	M = 0.24,STD = 0.11, 95CI = [0.24 - 0.24]	t(12916) = -10.15
N = 6459	C.	M = 0.26,STD = 0.12, 95CI = [0.26 - 0.26]	p < .001
	md	M = 0.02,STD = 0.16, 95CI = [0.02 - 0.02]	d = 0.18
RA	P.	M = 0.23,STD = 0.10, 95CI = [0.22 - 0.24]	t(1664) = -6.24
N = 833	C.	M = 0.27,STD = 0.13, 95CI = [0.26 - 0.28]	p < .001
	md	M = 0.04,STD = 0.16, 95CI = [0.03 - 0.05]	d = 0.31
Gout	P.	M = 0.28,STD = 0.15, 95CI = [0.27 - 0.29]	t(2240) = -0.78
N = 1121	C.	M = 0.28,STD = 0.14, 95CI = [0.27 - 0.29]	p = .434
	md	M = 0.00,STD = 0.21, 95CI = [-0.01 - 0.01]	d = 0.03
other	P.	M = 0.26,STD = 0.12, 95CI = [0.25 - 0.27]	t(1138) = -3.43
N = 570	C.	M = 0.29,STD = 0.16, 95CI = [0.28 - 0.30]	p < .001
	md	M = 0.03,STD = 0.20, 95CI = [0.01 - 0.05]	d = 0.21
		Acceleration Range (of Gait in m/sec^2)	
OA	P.	M = 1.48,STD = 0.53, 95CI = [1.47 - 1.49]	t(12916) = -2.00
N = 6459	C.	M = 1.50,STD = 0.56, 95CI = [1.49 - 1.51]	p < .050
	md	M = 0.02,STD = 0.77, 95CI = [0.00 - 0.04]	d = 0.04
RA	P.	M = 1.47,STD = 0.51, 95CI = [1.44 - 1.50]	t(1664) = -1.06
N = 833	C.	M = 1.49,STD = 0.57, 95CI = [1.45 - 1.53]	p = .291
	md	M = 0.03,STD = 0.78, 95CI = [-0.02 - 0.08]	d = 0.05
Gout	P.	M = 1.46,STD = 0.61, 95CI = [1.42 - 1.50]	t(2240) = -4.28
N = 1121	C.	M = 1.58,STD = 0.69, 95CI = [1.54 - 1.62]	p < .001
	md	M = 0.12,STD = 0.93, 95CI = [0.07 - 0.17]	d = 0.18
other	P.	M = 1.54,STD = 0.59, 95CI = [1.49 - 1.59]	t(1138) = 0.23
N = 570	C.	M = 1.53,STD = 0.67, 95CI = [1.47 - 1.59]	p = .817
	md	M = -0.01,STD = 0.85, 95CI = [-0.08 - 0.06]	d = -0.01
		Step Rate (steps/min)	

OA	P.	M = 55.64,STD = 5.75, 95CI = [55.50 - 55.78]	t(12916) = -2.54
N = 6459	C.	M = 55.89,STD = 5.52, 95CI = [55.76 - 56.02]	p < .050
	md	M = 0.25,STD = 7.85, 95CI = [0.06 - 0.44]	d = 0.04
RA	P.	M = 55.71,STD = 5.72, 95CI = [55.32 - 56.10]	t(1664) = -0.35
N = 833	C.	M = 55.80,STD = 5.56, 95CI = [55.42 - 56.18]	p = .729
	md	M = 0.10,STD = 7.95, 95CI = [-0.44 - 0.64]	d = 0.02
Gout	P.	M = 54.48,STD = 5.63, 95CI = [54.15 - 54.81]	t(2240) = -3.34
N = 1121	C.	M = 55.27,STD = 5.55, 95CI = [54.95 - 55.59]	p < .001
	md	M = 0.79,STD = 7.88, 95CI = [0.33 - 1.25]	d = 0.14
other	P.	M = 55.51,STD = 5.65, 95CI = [55.05 - 55.97]	t(1138) = -0.24
N = 570	C.	M = 55.59,STD = 5.29, 95CI = [55.16 - 56.02]	p = .807
	md	M = 0.08,STD = 7.63, 95CI = [-0.55 - 0.71]	d = 0.01
		Symmetry of Step Time (in sec): All Values e-02	
OA	P.	M = 1.65,STD = 1.45, 95CI = [1.61 - 1.69]	t(12916) = 5.17
N = 6459	C.	M = 1.52,STD = 1.38, 95CI = [1.49 - 1.55]	p < .001
	md	M = -0.13,STD = 1.99, 95CI = [-0.18 - -0.08]	d = -0.09
RA	P.	M = 1.67,STD = 1.42, 95CI = [1.57 - 1.77]	t(1664) = 2.07
N = 833	C.	M = 1.53,STD = 1.31, 95CI = [1.44 - 1.62]	p < .050
	md	M = -0.14,STD = 1.96, 95CI = [-0.27 - -0.01]	d = -0.10
Gout	P.	M = 1.65,STD = 1.40, 95CI = [1.57 - 1.73]	t(2240) = 1.63
N = 1121	C.	M = 1.56,STD = 1.45, 95CI = [1.48 - 1.64]	p = .103
	md	M = -0.10,STD = 2.00, 95CI = [-0.22 - 0.02]	d = -0.07
other	P.	M = 1.56,STD = 1.30, 95CI = [1.45 - 1.67]	t(1138) = 0.01
N = 570	C.	M = 1.56,STD = 1.45, 95CI = [1.44 - 1.68]	p = .989
	md	M = 0.00,STD = 2.00, 95CI = [-0.16 - 0.16]	d = 0.00
		Variance of Acceleration Range (in m/sec^2)	
OA	P.	M = 0.40,STD = 0.17, 95CI = [0.40 - 0.40]	t(12916) = 3.30
N = 6459	C.	M = 0.39,STD = 0.17, 95CI = [0.39 - 0.39]	p < .001
	md	M = -0.01,STD = 0.24, 95CI = [-0.02 - 0.00]	d = 0.06
RA	P.	M = 0.40,STD = 0.16, 95CI = [0.39 - 0.41]	t(1664) = 0.93
N = 833	C.	M = 0.39,STD = 0.17, 95CI = [0.38 - 0.40]	p = .353
	md	M = -0.01,STD = 0.23, 95CI = [-0.03 - 0.01]	d = -0.05
Gout	P.	M = 0.38,STD = 0.18, 95CI = [0.37 - 0.39]	t(2240) = -3.03
N = 1121	C.	M = 0.40,STD = 0.20, 95CI = [0.39 - 0.41]	p < .010
	md	M = 0.02,STD = 0.27, 95CI = [0.00 - 0.04]	d = 0.13
other	P.	M = 0.41,STD = 0.18, 95CI = [0.40 - 0.42]	t(1138) = 2.03
N = 570	C.	M = 0.39,STD = 0.18, 95CI = [0.38 - 0.40]	p < .050
	md	M = -0.02,STD = 0.25, 95CI = [-0.04 - 0.00]	d = 0.12

8.5. Discussion

Our initial aims were to extract:

1) Micro-measures of Step Parameters (for each walking episode): Gait Cadence & Variability; Stride & Step Time; Step Asymmetry & Irregularity; Amplitude of Heel-Strike

2) Macro-measures of Gait Episodes (changes of gait parameters between gait episodes): Walking & Running distinction; Gait Categories (group episodes by pattern and evaluate their: Quantity (how many different categories); Quality (of each category); Distribution (of time spent in each category); Step Parameters mean & variation (for each category)))

3) Macro-measures of Walking Behaviour: Time of Gait Episode/Category Occurrence (morning, midnight, afternoon, night); Total time spend walking; Walking Episodes length

These aims ended up being a little bit too ambitious for the scope of this thesis and we run out of time to extract these values from the SPE outputs in a constructive manner. Nevertheless, we at

least managed to analyse the proposed micro-measures of the gait parameters. The above list could be used for guiding the addition of further parameter extraction to the SPE code.

For further analysis conventional logistical and multinomial regression to identify individual parameters that differ and machine learning classification methods such as random forests to identify combinations of parameters that differ could be used. Principal components analysis can be used to identify important types of variation within the population. Potential confounders such as obesity self-reported physical activity, medical history and socio-demographic factors could also be explored.

The specific research questions of comparing gait parameters between people with RMDS and healthy controls were conducted, but in the future, it could be extended to a comparison of gait parameters between different RMDs and a comparison of gait parameters within specific RMDs by the presence of co-morbidities or surrogate measures of disease severity.

In comparison to the KOALAP data analysis, we already improved upon the counterintuitive handling of the values of the step symmetry. However, the parameters could potentially be improved in general by a factor ranking a participant's parameters on a scale of 1 to 0 with 1 reflecting the participants' personal best and 0 their personal worst step parameter values might. In hindsight, this would be a more effective way to report the participants' functions and could lead to better results for tracking participants' performance over time.

The gait parameter values extracted for the patient and control groups did generally displays the directions we expected, with was not the case for the KOALAP data. A notable exception is the gout patient group. However, the longer walking episodes of the gout groups in comparison to the other participant groups could be explained by the high ratio of men within the group. This could also explain the high acceleration range of the control group. Furthermore, a reduced step rate, increased asymmetry and reduced variance of the low acceleration range could speak for a careful walking behaviour. It furthermore should be considered, that only one week of data was recorded. In gout patient who tends to experience symptoms in the form of attacks, in comparison to the OA and RA patients who generally experience regular symptoms, this could mean that physical functioning was not impaired by their disease during data collection. Furthermore, since the other RMD patient group consisted of various RMD types with more variations within the symptom expression than the OA, RA and Gout patient groups, we are not surprised, to see a less distinct difference between the patient and control group. Overall, these observations promise a good likelihood to identify differences between step parameters of different RMD patient groups.

9. Conclusion

During the initial literature review, we discovered that there is a lack of research done on the physical functioning of RMD patients' using accelerometer data. While a great deal of research has been done using accelerometers to extract step counts and physical activity measures in the patient's everyday environment, the investigation of physical functioning has so far been contained to laboratory settings. With the rise of access to IMUs within smartphones and smartwatches, eHealth is becoming a rising field of interest for treating patients. While many consumer devices already offer their users tracking of their health data, it is hard to judge how reliable these output measures are for research and clinical purposes. We looked at the step counts recorded by a Fitbit, a very common consumer device, and discovered that it had a clear tendency to overcount steps. This is likely due to users being less likely to complain about overcounted than undercounted steps. We, therefore, developed a gait detection and step parameter extraction algorithm that would enable researchers' access to accelerometer data depicting the participants' gait in a free-living environment and step parameters that are commonly looked at for diagnostic purposes.

The literature review helped us identify step parameters of interest, such as gait symmetry and variance. Our step parameter extraction algorithm successfully outputs the selected parameters of walking episode lengths in minutes, acceleration range of the steps, the step rate, the symmetry of the walking frequency, and variance of the acceleration range. We have to admit that we had much higher ambitions when we started this project. For example, we were not able to distinguish between the stance phase and swing phase of the gait cycle. The symmetry of these phases in particular was often especially successful as a diagnostic tool within the literature review. We have to concede that the acceleration pattern of gait collected at the wrist simply cannot depict the same amount of detail compared to acceleration directly recorded on the legs. This is always the case for gait patterns with free arm swing. In "stiff" arm conditions, we can sometimes see acceleration patterns similar to the ones depicted in the literature review, but walking behaviours under "stiff" arm conditions tend to be more confounded by noise than behaviours under arm "swing" conditions. Nevertheless, we managed to identify some measures of step symmetry successfully.

The algorithm was mostly developed using the SRDS. This dataset comes with limitations; mostly in that it was only recorded using one healthy and young participant. It, therefore, reflects mostly peak walking behaviour. Furthermore, the non-walking behaviour consisted only of the coffee trial. The participant is a very animated speaker. This is reflected in both the Fitbit step counts they managed to accumulate while sitting down and in the common gait detection methods' inability to reliably identify the coffee drinking and talking behaviour as non-gait. While we were recording the dataset we mainly planned to use it to inform the labelling of KOALAP dataset. Only after data collection was done did we decide to use it to actively support the development of the AGDA. We only included trials in which we were able to label walking and non-walking behaviours to the second. Trials that we were not able to label with this accuracy were: 1) the initial trial, which served its purpose more to evaluate the performance; 2) the driving trial, which we judged not to show interesting behaviour patterns, that weren't already served by the coffee trial; and 3) the football trial.

The last trial was recorded quite spontaneously, and when we fail to collect the data properly we weren't motivated to repeat it. Nevertheless, we can justify the exclusion of sports behaviours from the SRDS with our focus on walking behaviours. Since the walking speed thresholds were set quite wide, our algorithm could potentially include running behaviour since one man's jogging is another man's stroll. We decided not to include activity with frequencies higher than what the literature judges to be fast walking speeds. If we were to include higher frequency data, we would more likely end up with non-walking behaviours, such as shaking something, instead of running behaviours. Furthermore, it is more likely for patients to remember and properly document sports behaviours than the patients' everyday walking. Running behaviour that is picked up by the algorithm will nevertheless be useful. According to the literature review, patients' physical functioning can get worse with more demanding activities, which would have a strong impact on the step parameters, making impairment more obvious. On the other hand, if the patient engages in sports while displaying improved physical functioning, this would also reflect more positive on the physical functioning measures.

We hope that the limitations of the SRDS were counteracted by our careful considerations when developing the algorithm steps. In general, we used the data to find where our algorithm was not performing as we had planned. Within the over 4 hours of data, we identified many hurdles to overcome for detecting gait. Furthermore, we examined some gait data output of the KOALAP dataset and also use these observations to exclude non-walking behaviours. Within this thesis, we did not have access to labelled patient data. Labelling the KOALAP data could not be reliably done. We would propose that an accurately labelled dataset recorded from patients and healthy participants with more variance of behaviour could be beneficial to further analyse the accuracy of the AGDA.

While the algorithm's gait detection accuracy was only verified against the SRDS, we were very careful to not set thresholds depending on the dataset itself. Instead, the thresholds we set are commonly used in gait detection methods (such as the standard deviation thresholds) or derived from the literature (such as the walking speed thresholds). While we had set out with the very ambitious goal of identifying every step in the data, in order to have data to extract physical functioning data from, it is important to reliably extract walking data that can be analysed. Therefore, we are sure that strongly impaired walking will probably not be picked up by the algorithm. We decided to exclude gait from the step parameter extraction that was too irregular to be picked up since it would strongly confound the extraction of step parameter measures. We propose instead that the output parameter of episode length is a good measure to identify a patient's ability to sustain walking behaviour acceptable for analysing. The constant interruption of walking behaviour can therefore also be a good indicator of the amount of gait that is too disrupted to measure otherwise. Since we weigh the step parameter output of each episode according to the episode length, the step parameters themselves are a reflection of the patients' gait when they can sustain it.

The algorithm itself is widely applicable to different kinds of datasets. We were able to apply the algorithm to both the dataset collected from a consumer device (KOALAP) at 50 Hz and from a research-grade device (UK BioBank) at 100 Hz. We coded the algorithm to be able to handle

different sampling frequencies without the need to change the inner workings of the code. Nevertheless, datasets need to be collected at a sampling frequency that is suitable to identify gait behaviour. We would not recommend running the code on datasets with sample frequencies lower than 50 Hz. Also, we doubt that step parameters identified from data with sampling frequencies lower than 30 Hz will have the necessary resolution to reflect step parameter measures accurately. With higher sampling frequencies the resolution of the data improves, but the algorithm speed decreases. We, therefore, do not recommend sampling frequencies higher than 100 Hz. An acceleration range of ±8 g is appropriate for the extraction of gait behaviours. While our code can also handle other acceleration ranges, we do not recommend using very low (e.g. ±2 g) or very high (e.g. ±32 g) acceleration ranges, since the data will most likely not be able to reflect gait patterns accurately. Our algorithm relies on 3-axis accelerometer data input, and cannot handle 1-axis or 6-axis data input. The algorithm was developed for data collected from smartwatches at the wrist location, but can also be used to analyse data recorded at other locations of the body. For example, walking behaviours recorded by smartphones would most likely display “stiff” walking behaviours and therefore be picked up by the algorithm. Data collected at the legs could also be analysed, but our code would be excessive for this task. For leg data, algorithms with more simple gait detection methods and more detailed step parameter extraction methods would be more suitable.

Within the UK Biobank dataset, we were able to show that we could find significant differences between patients and matched controls when looking at our extracted step parameters. In general, patients had shorter walking episodes, reflecting a decreased ability to sustain measurable walks. The acceleration range was reduced, reflecting walking with less energy. The step rate was lowered, reflecting a decrease in walk speed. The step frequency difference between the left and right leg (the symmetry of step time) was larger, reflecting irregularity of gait and favouring of one side of the other. The variance of the acceleration range was also generally larger for the patient groups, reflecting irregular gait. While the difference between patient and control groups is quite small, we have to consider that the controls are not necessarily “healthy” individuals. We matched participants of the RMD groups with non-RMD participants. This participant selection does not exclude participants from the control group that might have other conditions that can affect gait, such as Parkinson’s disease. Nevertheless, age and sex are some of the strongest factors affecting step parameters. We, therefore, choose these characteristics as the primary factors for matching participants. The confounding effect of non-RMD medical conditions should be counterweighted by the large size of over 17,000 participants.

Due to the size of the participant pool of the UK BioBank dataset, we decided to only extract one measure for each participant for each step parameter. Furthermore, only one week of behaviours was recorded. We are therefore only offered a small window to detect the participants’ physical functioning. We assume that this had caused some problems when looking at the gout patient group. Since gout is a RMD that causes decreases in physical functioning during “attacks”, we cannot be sure if even a majority of the gout patients had attacks of gout during the time they were measured. From looking at the data, we even assume that gout patients might exhibit more careful walking due even when not experiencing a gout attack. Surprisingly, their variance was very

low, while the symmetry between limbs was strongly affected and the step rate and acceleration range were decreased. We think that this might reflect unconscious and habitual efforts to relieve strain on the affected side of the body even outside of gout attacks. This observation should be handled with care and would require further examinations that are beyond the scope of this thesis. For our goal of using the AGDA for research purposes, the UK Biobank dataset shows that we were able to extract step parameters that clearly differentiate RMD and non-RMD patient groups.

The UK Biobank dataset could not tell us more about the relationship between our step parameter measures and physical functioning beyond comparing RMD patients with non-RMD patients. We used the KOALAP dataset to relate step parameters extracted from passively recorded accelerometer data with actively recorded pain scores, submitted by the participants. Looking at the distribution of the step parameters extracted from OA patients, we can see wide differences in the distribution of the step rate, acceleration range and acceleration range variance between patients. Looking at correlations between step parameters and pain scores, we saw some small to medium correlations for some participants. We found the most reliable and strongest correlations between pain reports during aggravating activity and the acceleration range. Interestingly, the direction of the correlations varied strongly between participants. Also, which step parameter was correlated with which kind of pain report varied widely between participants. This has implications for the goal of developing a diagnostic tool for smartwatches, such as the KOALAP app. For example, episode length and acceleration range might be strongly negatively correlated with daily function, as was the case for user 34, but this will not be the case for every OA patient. Which step parameters and pain reports to focus on needs to be highly individualised to the user. This most likely means a long initial period of data collection for users. It would enable them to identify what gait parameters correlates with their self-reported pain before a potential app could be used for passive tracking. Furthermore, we need to consider that there is a potential issue with a paradoxical weighting of step parameter measures against pain reports. Flares of pain could cause the participants' gait to be too strongly impaired to be picked up while walking behaviour during periods of relief might just hit the threshold for step parameter extraction. When correlated directly with pain reports, this could lead to a bias where inactivity of short episodes of higher quality gait gets overshadowed by higher activity with lower quality gait. We came to this conclusion by comparing the step parameter of episode length with the other step parameters. This problem is therefore a challenge that would need to be addressed before implementing the algorithm as a diagnostic tool. We most likely did not have this problem with the UK BioBank dataset, since we looked at the participants' performance as a whole and not over time.

Another challenge of using consumer devices for research purposes or for the development of diagnostic tools is handling data extraction. It would be prudent to closely work with the developer of the smartwatch software during the development of an app. But even though the KOALAP app was developed in collaboration with Google, we still faced major data-wrangling hurdles. The major problem was the unfriendly way data was stored which led to one second of data being regularly lost. The problem of non-unique observations found in the KOALAP dataset is most likely due to a difference in priorities between research-grade and consumer devices. Consumer devices, such as the Huawei Watch 2, offer us a larger range of possibilities than research-grade devices, such as

the Axtivity. For example, a smartwatch app could both be used for data collection and extraction, initiate the participant to conduct walking behaviours for labelling and request self-report of physical functioning and well-being from patients. The Axtivity, on the other hand, can only record the raw accelerometer data.

Lastly, we should acknowledge that the current state of the AGDA does not necessarily reflect the best coding practices. Data outputs are saved as Matlab files, which would make reading the data difficult for people not possessing the software. The same is true for the variable and processing time documentation. Data should better be served in binary format files, and documentation of runs should be recorded in JavaScript Object Notification format. Also, for the SRDS and the KOALAP dataset pre-processing was already done, and is therefore not included in the code. We would recommend always having the code handle data pre-processing, in order to ensure consistency. Furthermore, the GDA and SPE codes repeat some steps, which could be combined in later versions of the AGDA to improve efficiency. These codes were separated since we ran the GDA code on the KOALAP and UK Biobank datasets while the SPE was developed. The output data of the GDA, therefore, includes some data that we did not require for the SPE and we did not save some features extracted during the GDA that we did end up requiring in the SPE. Within this thesis, we reported the code in a form that yields the same outputs for the KOALAP and UK Biobank datasets. The code in its entirety and future versions of the code will be published on Github (Weihrich, 2022). We did this in order to ensure that the methodology of the data reported in the KOALAP and UK Biobank data analysis is clear and accurate. Corrections of the code between running the KOALAP and UK Biobank mainly included debugging problems with recording run-information documentation, switches between input and output folders and the type of information saved in the output datafiles. We reported the codes with these improved corrections since they did not affect the step parameter output data we extracted for analysis. We did not re-run the GDA on the datasets, since the size of the datasets did not allow for the time. Hence, instead of reporting the gait algorithms' performance time from the KOALAP and UK Biobank, we decided to report performance on the SRDS instead. We did it in both its original form and after randomising the data in five-minute chunks and accounted for the slight difference in the code made in the pre-processing of the KOALAP and UK Biobank dataset. We furthermore justified this decision, arguing that a smartwatch or other research teams would not necessarily have the processing capabilities of the iCSF. After optimisation and adjustment of the code for an app format, we are optimistic the gait data and step parameter extraction could even run within the app itself, due to the comparatively low processing requirements of the algorithm. Nevertheless, it needs to be seen how this would affect memory and battery capacity of the smartwatch, which is a limitation to keep in mind.

In conclusion, we are confident that we have successfully developed the fundamental structure of an algorithm, for gait data and step parameter extraction in free-living conditions from data recorded by smartwatches. However, before this algorithm can be integrated into an app targeted at RMD patients, further work should include optimisation of the algorithm, optimisation of the weighting of step parameters measures, and verification of accuracy and validity on a labelled dataset with a larger and more diverse population and behaviour investigation.

10. References

We would like to note that in 2018 the charity “Arthritis Research UK” merged with “Arthritis Care” into “Versus Arthritis”. Consequently, the old arthritisresearchuk.org website was taken down and its content was transferred to the versusarthritis.org domain. The brochures are therefore no longer accessible through the links.

Aitaterini, P., & Ji, T. (2005). Frequency and velocity of people walking. The Structural Engineer , 84 (3), 36-40.

Aminian, K., Rezakhanlou, K., De Andres, E., Fritsch, C., Leyvraz, P.-F., & Robert, P. (1999). Temporal feature estimation during walking using miniature accelerometers: an analysis of gait improvement after hip arthroplasty. Medical & Biological Engineering & Computing , 37 (6), 686-691.

Andreu-Perez, J., Garcia-Gancedo, L., McKinnell, J., Van der Drift, A., Powell, A., Hamy, V., et al. (2017). Developing fine-grained actigraphies for rheumatoid arthritis patients from a single accelerometer using machine learning. Sensors , 17 (9), 2113.

Arthritis Research UK. (2012). Condition: Osteoarthritis [Brochure]. Retrieved 2017, from Arthritis Research UK: https://www.arthritisresearchuk.org/arthritis-information/conditions/osteoarthritis.aspx

Arthritis Research UK. (2011). Keep Moving [Brochure]. Retrieved from Arthritis Research UK: https://www.arthritisresearchuk.org/~/media/Files/Arthritis-information/Living-with-arthritis/2282-Keep-moving-inc-poster.ashx

Arthritis Research UK. (2017). State of Musculoskeletal Health 2017 [Brochure]. Retrieved from Arthritis Research UK: https://www.arthritisresearchuk.org/~/media/Files/Press-releases/StateOfMusculoskeletalHealth2017.ashx?la=en

Atallah, L., Wiik, A., Lo, B., Cobb, J. P., Amis, A. A., & Yang, G.-Z. (2014). Gait asymmetry detection in older adults using a light ear-worn sensor. Physiological Measurement , 35 (5), N29-N40.

Auvinet, B., Chaleil, D., & Barrey, E. (1999). Accelerometric gait analysis for use in hospital outpatients. Revue du Rhumatisme (English ed.) , 66 (7-9), 389-397.

Axivity Ltd. (2015, February 05). AX3 Data Sheet: 3-Axis Logging Accelerometer, Version 1.3. Retrieved June 20, 2021, from axivity.com: https://axivity.com/files/resources/AX3_Data_Sheet.pdf

Axivity Ltd. (nan). Wrist Band - For Puck Format Sensors. Retrieved 03 25, 2022, from Axivity: https://axivity.com/product/wrist-band

Badawy, R., Raykov, Y., & Little, M. A. (2017, Nov 23). A unified algorithm framework for quality control of sensor data for behavioural clinimetric testing. arXiv preprint arXiv:1711.07557 .

Barden, J. M., Clermont, C. A., Kobsar, D., & Beauchet, O. (2016). Accelerometer-based step regularity is lower in older adults with bilateral knee osteoarthritis. Frontiers in Human Neuroscience , 10, 625.

Barisic, A., Leatherdale, S. T., & Kreiger, N. (2011). Importance of frequency, intensity, time and type (FITT) in physical activity assessment for epidemiological research. Canadian Journal of Public Health , 102 (3), 174-175.

Barrois, R., Gregory, T., Oudre, L., Moreau, T., Truong, C., Aram Pulini, A., et al. (2016). An automated recording method in clinical consultation to rate the limp in lower limb osteoarthritis. PloS ONE , 11 (10), e0164975.

Barrois, R., Oudre, L., Moreau, T., Truong, C., Vayatis, N., Buffat, S., et al. (2015). Quantify osteoarthritis gait at the doctor's office: a simple pelvis accelerometer based method independent from footwear and aging. Computer Methods in Biomechanics and Biomedical Engineering , 18 (S1), 1880-1881.

Beukenhorst, A. L., Howells, K., Cook, L., McBeth, J., O'Neill, T. W., Parkes, M. J., et al. (2020). Engagement and participant experiences with consumer smartwatches for health research: longitudinal, observational feasibility study. JMIR mHealth and uHealth , 8 (1), e14368.

Beukenhorst, A. L., Parkes, M. J., Cook, L., Barnard, R., van der Veer, S. N., Little, M. A., et al. (2019). Collecting symptoms and sensor data with consumer smartwatches (the Knee OsteoArthritis, Linking Activity and Pain Study): protocol for a longitudinal, observational feasibility study. JMIR research protocols , 8 (1), e10238.

Bolink, S. A., Brunton, L. R., van Laarhoven, S., Lipperts, M., Heyligers, I. C., Blom, A. W., et al. (2015). Frontal plane pelvic motion during gait captures hip osteoarthritis related disability. Hip International , 25 (5), 413-419.

Bolink, S. A., van Laarhoven, S. N., Lipperts, M., Heyligers, I. C., & Grimm, B. (2012). Inertial sensor motion analysis of gait, sit–stand transfers and step-up transfers: differentiating knee patients from healthy controls. Physiological Measurement , 33 (11), 1947-1958.

Brage, S., Doherty, A., Hammerla, N., van Hees, V., Granat, M., Jackson, D., et al. (2016, Jan 27). Physical activity monitor (accelerometer) Version 1.0. Retrieved Jul 15, 2020, from UK BioBank: https://biobank.ndph.ox.ac.uk/ukb/ukb/docs/PhysicalActivityMonitor.pdf

Brajdic, A., & Harle, R. (2013). Walk detection and step counting on unconstrained smartphones. Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing .

Brenneman, E. C., & Maly, M. R. (2018). Identifying changes in gait waveforms following a strengthening intervention for women with knee osteoarthritis using principal components analysis. Gait & Posture , 59, 286-291.

Brouwer, G. M., Van Tol, A. W., Bergink, A. P., Belo, J. N., Bernsen, R. M., Reijman, M., et al. (2007). Association between valgus and varus alignment and the development and progression of radiographic osteoarthritis of the knee. Arthritis & Rheumatism , 56 (4), 1204-1211.

Bussmann, J. B., Martens, W. L., Tulen, J. H., Schasfoort, F. C., Van Den Berg-Emons, H. J., & Stam, H. J. (2001). Measuring daily behavior using ambulatory accelerometry: the Activity Monitor. Behavior Research Methods, Instruments, & Computers , 33 (3), 349-356.

Cadzow, M., Merriman, T. R., & Dalbeth, N. (2017). Performance of gout definitions for genetic epidemiological studies: analysis of UK Biobank. Arthritis Research & Therapy , 19 (181), 1-7.

Calliess, T., Bocklage, R., Karkosch, R., Marschollek, M., Windhagen, H., & Schulze, M. (2014). Clinical evaluation of a mobile sensor-based gait analysis method for outcome measurement after knee arthroplasty. Sensors , 14 (9), 15953-15964.

Çankaya, D., Aktı, S., Ünal, Ş. B., & Sezgin, E. A. (2021). Unicompartmental knee arthroplasty results in a better gait pattern than total knee arthroplasty: Gait analysis with a smartphone application. Joint diseases and related surgery , 32 (1), 22.

Chaisson, C. E., Zhang, Y., Sharma, L., Kannel, W., & Felson, D. T. (1999). Grip strength and the risk of developing radiographic hand osteoarthritis: results from the Framingham Study. Arthritis & Rheumatism: Official Journal of the American College of Rheumatology , 42 (1), 33-8.

Christiansen, C. L., Bade, M. J., Paxton, R. J., & Stevens-Lapsley, J. E. (2015). Measuring movement symmetry using tibial-mounted accelerometers for people recovering from total knee arthroplasty. Clinical Biomechanics , 30 (7), 732-737.

Clermont, C. A., & Barden, J. M. (2016). Accelerometer-based determination of gait variability in older adults with knee osteoarthritis. Gait & Posture , 50, 126-130.

Coggon, D., Reading, I., Croft, P., McLaren, M., Barrett, D., & Cooper, C. (2001). Knee osteoarthritis and obesity. International Journal of Obesity & Related Metabolic Disorders , 25 (5), 622-627.

Cole, C. A., Anshari, D., Lambert, V., Thrasher, J., & Valafar, H. (2013). Detecting Smoking Events Using Accelerometer Data Collected via Smartwatch Technology: A Feasibility Study. JMIR Mhealth Uhealth , 5 (12), 1.

Collins, J. J., & Whittle, M. W. (1989). Impulsive forces during walking and their clinical implications. Clinical Biomechanics , 4 (3), 179-187.

Cook, M. J., Bellou, E., Bowes, J., Sergeant, J. C., O'Neill, T. W., Barton, A., et al. (2018). The prevalence of co-morbidities and their impact on physical activity in people with inflammatory rheumatic diseases compared with the general population: results from the UK Biobank. Rheumatology , 57 (12), 2172-2182.

Crawford, D. A., Duwelius, P. J., Sneller, M. A., Morris, M. J., Hurst, J. M., Berend, K. R., et al. (2021). 2021 Mark Coventry Award: Use of a smartphone-based care platform after primary partial and total knee arthroplasty: a prospective randomized controlled trial. The Bone & Joint Journal , 103-B (6 Supple A), 3-12.

Crawford, D. A., Lombardi Jr, A. V., Berend, K. R., Huddleston III, J. I., Peters, C. L., DeHaan, A., et al. (2021). Early outcomes of primary total hip arthroplasty with use of a smartphone-based

care platform: a prospective randomized controlled trial. The Bone & Joint Journal , 103-B (7 Supple B), 91-97.

Dejan. (2015, November 19). How MEMS Accelerometer Gyroscope Magnetometer Work & Arduino Tutorial. Retrieved July 16, 2018, from How To Mechatronics: https://howtomechatronics.com/how-it-works/electrical-engineering/mems-accelerometer-gyrocope-magnetometer-arduino/

D'Errico, J. (2021). MATLAB Central File exchange. Retrieved 04 14, 2020, from movingstd & movingstd2: https://uk.mathworks.com/matlabcentral/fileexchange/9428-movingstd-movingstd2

Dimension Engineering. (n.d.). A beginner's guide to accelerometers. Retrieved January 25, 2018, from Dimension Engineering: https://www.dimensionengineering.com/info/accelerometers

Dixon-Warren, S. J. (2010, December 23). Motion Sensing in the iPone 4: MEMS Accelerometer. Retrieved July 16, 2018, from MEMS Journal: http://www.memsjournal.com/2010/12/motion-sensing-in-the-iphone-4-mems-accelerometer.html

Doherty, A., Jackson, D., Hammerla, N., Plötz, T., Olivier, P., Granat, M. H., et al. (2017). Large scale population assessment of physical activity using wrist worn accelerometers: The UK Biobank Study. PloS ONE , 12 (2), e0169649.

Esfandyari, J., De Nuccio, R., & Gang, X. (2010, November 15). Introduction to MEMS gyroscopes. Retrieved December 8, 2017, from SolidState Technology: http://electroiq.com/blog/2010/11/introduction-to-mems-gyroscopes/

Fitbit. (2018). Fitbit Charge 2: User Manual Version 1.3. Retrieved Juli 20, 2021, from fitbit.com: https://help.fitbit.com/manuals/manual_charge_2_en_US.pdf

Gouttebarge, V., Haruhito, A., & Kerkhoffs, G. M. (2018). Knee osteoarthritis in professional football is related to severe knee injury and knee surgery. Injury Epidemiology , 5 (1), 1-9.

Hammerla, N. (2012). AX3_readFile.m. Retrieved 02 15, 2021, from https://github.com/digitalinteraction/openmovement/blob/master/Software/Analysis/Matlab/AX3_readFile.m

Hanly, S. (2016, March 11). Accelerometers: Taking the Guesswork out of Accelerometer Selection. Retrieved December 8, 2017, from MIDE ENGINEERING SOLUTIONS: https://blog.mide.com/accelerometer-selection

Hartog, D. d., Harlaar, J., & Smit, G. (2021). Stumblemeter: design and validation of a system that detects and classifies stumbles during gait. Sensors , 21 (19), 6636.

He, J., Lippmann, K., Shakoor, N., Ferrigno, C., & Wimmer, M. A. (2019). Unsupervised gait retraining using a wireless pressure-detecting shoe insole. Gait & Posture , 70, 408-413.

Hodt-Billington, C., Helbostad, J. L., Vervaat, W., Rognsvåg, T., & Moe-Nilssen, R. (2011). Changes in gait Symmetry, gait velocity and self-reported function followings total hip replacement. Journal of Rehabilitation Medicine , 43 (9), 787-793.

Hodt-Billington, C., Helbostad, J. L., Vervaat, W., Rognsvåg, T., & Moe-Nilssen, R. (2012). Criteria of gait asymmetry in patients with hip osteoarthritis. Physiotherapy Theory and Practice , 28 (2), 134-141.

Homma, D., Minato, I., Imai, N., Miyasaka, D., Sakaif, Y., Horigome, Y., et al. (2020). Three-dimensional Evaluation of Abnormal Gait in Patients with Hip Osteoarthritis. Acta Medica Okayama , 74 (5), 391-399.

Howe, R. T., & Chang, S. C. (1989). Patent No. U.S. Patent No. 4,851,080. Washington, DC: U.S. Patent and Trademark Office.

Huawei. (n.d.). Huawei Watch 2. Retrieved Mai 25, 2018, from Huawei Consumer: https://consumer.huawei.com/en/wearables/watch2/specs/

Hunter, D. J., & Eckstein, F. (2009). Exercise and osteoarthritis. Journal of Anatomy , 214 (2), 197-207.

Inman, V. T., Ralston, H. J., & Todd, F. (1981). Human walking. Baltimore, USA: Williams & Wilkins.

Jackson, D. G. (2015). github.com. Retrieved 02 19, 2021, from Open Movment GUI Application: https://github.com/digitalinteraction/openmovement/blob/master/Docs/omgui/index.md

John. (2011, August 17). MEMS Accelerometer. Retrieved December 8, 2017, from Instrumentation Today: http://www.instrumentationtoday.com/mems-accelerometer/2011/08/

Jolles, B. M., Duc, C., Coley, B., Aminian, K., Pichonnaz, C., Bassin, J.-P., et al. (2011). Objective evaluation of shoulder function using body-fixed sensors: a new way to detect early treatment failures? Journal of Shoulder and Elbow Surgery , 20 (7), 1074-1081.

Kalane, A. (2018, January 3). Finding the 3rd axis in 3d space. Retrieved July 16, 2018, from StackExchange: https://math.stackexchange.com/questions/2590734/finding-the-3rd-axis-in-3d-space

Kandel, S., Heer, J., Plaisant, C., Kennedy, J., van Ham, F., Henry-Riche, N., et al. (2011). Research directions in data wrangling: Visualizations and transformations for usable and credible data. Information Visualization , 10 (4), 271-288.

Karas, M., Bai, J., Strączkiewicz, M., Harezlak, J., Glynn, N. W., Harris, T., et al. (2018). Accelerometry data in health research: Challenges and opportunities: Review and examples. bioRxiv , 276154.

Khan, H., Walker, P. S., Zuckerman, J. D., Slover, J., Jaffe, F., Karia, R. J., et al. (2013). The potential of accelerometers in the evaluation of stability of total knee arthroplasty. The Journal of Arthroplasty , 28 (3), 459-462.

Kluge, F., Hannink, J., Pasluosta, C., Klucken, J., Gaßner, H., Gelse, K., et al. (2018). Pre-operative sensor-based gait parameters predict functional outcome after total knee arthroplasty. Gait & Posture , 66, 194-200.

Kobsar, D., & Ferber, R. (2018). Wearable sensor data to track subject-specific movement patterns related to clinical outcomes using a machine learning approach. Sensors , 18 (9), 2828.

Kobsar, D., Osis, S. T., Phinyomark, A., Boyd, J. E., & Ferber, R. (2016). Reliability of gait analysis using wearable sensors in patients with knee osteoarthritis. Journal of Biomechanics , 49 (16), 3977-3982.

KOOS. (2012, August). Questionnaires, user's guides and scoring files. Retrieved Mai 15, 2018, from KOOS : http://www.koos.nu/KOOSscoring2012.pdf

Liikavainio, T., Bragge, T., Hakkarainen, M., Karjalainen, P. A., & Arokoski, J. P. (2010). Gait and muscle activation changes in men with knee osteoarthritis. The Knee , 17 (1), 69-76.

Little, M. A. (2018, March 26). E-mail correspondence.

Lyytinen, T., Bragge, T., Hakkarainen, M., Liikavainio, T., Karjalainen, P. A., & Arokoski, J. P. (2016). Repeatability of knee impulsive loading measurements with skin-mounted accelerometers and lower limb surface electromyographic recordings during gait in knee osteoarthritic and asymptomatic individuals. Journal of Musculoskeletal & Neuronal Interactions , 16 (1), 63-74.

Marks, R. (2007). Obesity profiles with knee osteoarthritis: Correlation with pain, disability, disease progression. Obesity , 15 (7), 1867-1874.

Mehta, S. P., Bremer, H., Cyrus, H., Milligan, A., & Oliashirazi, A. (2021). Smartphone goniometer has excellent reliability between novice and experienced physical therapists in assessing knee range of motion. Bodywork and Movement Therapies , 25, 67-74.

Meyns, P., Bruijn, S. M., & Duysens, J. (2013). The how and why of arm swing during human walking. Gait & Posture , 38 (4), 555-562.

Mokkink, L. B., Terwee, C. B., van Lummel, R. C., de Witte, S. J., Wetzels, L., Bouter, L. M., et al. (2005). Construct validity of the DynaPort KneeTest: A comparison with observations of physical therapists. Osteoarthritis and Cartilage , 13 (8), 738-743.

Nedelkovski, D. (2015). MEMS Accelerometer Gyroscope Magnetometer & Arduino. Retrieved December 8, 2017, from How to Mechatronics: http://howtomechatronics.com/how-it-works/electrical-engineering/mems-accelerometer-gyrocope-magnetometer-arduino/

Neumann, D. (2010). Kinesiology of the Musculoskeletal System: Foundations for Rehabilitation (Vol. 2). St. Louis, Missouri, United States: Mosby.

NHS choices. (2016, August 4). Osteoarthritis. Retrieved October 25, 2017, from NHS choices: https://www.nhs.uk/conditions/osteoarthritis/

Nicolson, P. J., Hinman, R. S., Wrigley, T. V., Stratford, P. W., & Bennell, K. L. (2019). Effects of covertly measured home exercise adherence on patient outcomes among older adults with chronic knee pain. journal of orthopaedic & sports physical therapy , 49 (7), 548-556.

Nowell, W. B., Gavigan, K., Kannowski, C. L., Cai, Z., Hunter, T., Venkatachalam, S., et al. (2021). Which patient-reported outcomes do rheumatology patients find important to track digitally? A real-world longitudinal study in ArthritisPower. Arthritis research & therapy , 23 (1), 1-10.

Öberg, T., Karsznia, A., & Öberg, K. (1993). Basic gait parameters: reference data for normal subjects, 10-79 years of age. Journal of Rehabilitation Research and Development , 30 (2), 210-210.

O'Connor, P. (2021). MATLAB Central File Exchange. Retrieved 17 05, from PeakSeek : https://www.mathworks.com/matlabcentral/fileexchange/26581-peakseek

ODPHP. (n.d.). Appendix 1. Translating Scientific Evidence About Total Amount and Intensity of Physical Activity Into Guidelines. . Retrieved November 10, 2017, from Office of Disease Prevention and Health Care (ODPHP): https://health.gov/paguidelines/guidelines/appendix1.aspx

ODPHP. (n.d.). Office of Disease Prevention and Health Care. Retrieved 11 10, 2017, from Appendix 1. Translating Scientific Evidence About Total Amount and Intensity of Physical Activity Into Guidelines.: https://health.gov/paguidelines/guidelines/appendix1.asp

Ogata, K., Yasunaga, M., & Nomiyama, H. (1997). The effect of wedged insoles on the thrust of osteoarthritic knees. International Orthopaedics , 21 (5), 308-312.

Palazzo, C., Nguyen, C., Lefevre-Colau, M.-M., Rannou, F., & Poiraudeau, S. (2016). Risk factors and burden of osteoarthritis. Annals of Physical and Rehabilitation Medicine , 59 (3), 134-138.

Park, J. (2008). Synthesis of natural arm swing motion in human bipedal walking. Journal of Biomechanics , 41 (7), 1417-1426.

Reddy, N. P., Rothschild, B. M., Verrall, E., & Joshi, A. (2001). Noninvasive measurement of acceleration at the knee joint in patients with rheumatoid arthritis and spondyloarthropathy of the knee. Annals of Biomedical Engineering , 29 (12), 1106-1111.

Redell, D. (1998, January 11). Thinking About Accelerometers and Gravity. Retrieved January 23, 2018, from Livermore Unit National Association of Rocketry: http://www.lunar.org/docs/LUNARclips/v5/v5n1/Accelerometers.html

Roos, E. M., Roos, H. P., Lohmander, L. S., Ekdahl, C., & Beynnon, B. D. (1998). Knee Injury and Osteoarthritis Outcome Score (KOOS) - Development of a self-administered outcome measure. Journal of Orthopaedic & Sports Physical Therapy , 28 (2), 88-96.

Rööser, B., Ekbladh, R., & Lidgren, L. (1988). The shock-absorbing effect of soles and insoles. International Orthopaedics , 12 (4), 335-338.

Rouhani, H., Favre, J., Crevoisier, X., & Aminian, K. (2014). A wearable system for multi-segment foot kinetics measurement. Journal of Biomechanics , 47 (7), 1704-1711.

Rouhani, H., Favre, J., Crevoisier, X., & Aminian, K. (2011). Ambulatory measurement of ankle kinetics for clinical applications. Journal of Biomechanics , 44 (15), 2712-2718.

Rouhani, H., Favre, J., Crevoisier, X., & Aminian, K. (2012). Measurement of multi-segment foot joint angles during gait using a wearable system. Journal of Biomechanical Engineering , 134 (6), 061006.

Rouse, M. (2005, September). big-endian and little-endian. Retrieved Juni 7, 2018, from Tech Target: https://searchnetworking.techtarget.com/definition/big-endian-and-little-endian

Rueterbories, J., Spaich, E. G., Larsen, B., & Andersen, O. K. (2010). Methods for gait event detection and analysis in ambulatory systems. Medical Engineering & Physics , 32 (6), 545-552.

Schmid, S., Luder, G., Mueller Mebes, C., Stettler, M., Stutz, U., Ziswiler, H.-R., et al. (2013). Neuromechanical gait adaptations in women with joint hypermobility—An exploratory study. Clinical Biomechanics , 28 (9), 1020-1025.

Scuderi, G. R., Bourne, R. B., Noble, P. C., Benjamin, J. B., Lonner, J. H., & Scott, W. N. (2012). The new knee society knee scoring system. Clinical Orthopaedics and Related Research , 470 (1), 3-19.

Siebert, S., Lyall, D. M., Mackay, D. F., Porter, D., McInnes, I. B., Sattar, N., et al. (2016). Characteristics of rheumatoid arthritis and its association with major comorbid conditions: Cross-sectional study of 502 649 UK Biobank participants. RMD Open , 2 (1), e000267.

Smidt, G. L., Deusinger, R. H., Arora, J., & Albright, J. P. (1977). An automated accelerometry system for gait analysis. Journal of Biomechanics , 10 (5-6), 367-375.

Song, S.-M., & Waldron, K. J. (1987). An analytical approach for gait study and its applications on wave gaits. The International Journal of Robotics Research , 6 (2), 60-71.

Staab, W., Hottowitz, R., Sohns, C., Sohns, J. M., Gilbert, F., Menke, J., et al. (2014). Accelerometer and gyroscope based gait analysis using spectral analysis of patients with osteoarthritis of the knee. Journal of Physical Therapy Science , 26 (7), 997-1002.

Stauber, A., Schüßler, N., Palmdorf, S., Schürholz, N., Bruns, D., Osterbrink, J., et al. (2020). RECOVER-E–a mobile app for patients undergoing total knee or hip replacement: study protocol. BMC musculoskeletal disorders , 21 (1), 1-10.

Straczkiewicz, M., Glynn, N. W., & Harezlak, J. (2019). On placement, location and orientation of wrist-worn tri-axial accelerometers during free-living measurements. Sensors , 19 (9), 2095.

Swift, A. (2002, February 28). Osteoarthritis 3: impact on patients. Retrieved January 26, 2018, from Nursing Times: https://www.nursingtimes.net/clinical-archive/long-term-conditions/osteoarthritis-3-impact-on-patients/5042032.article

TE Connectivity. (2017, September). CHOOSING THE RIGHT. Retrieved December 15, 2017, from Mouser Electronics: https://www.mouser.com/pdfdocs/choosing-the-right-accelerometer-white-paper.pdf

The MathWorks, I. (2018). MATLAB R2018b [computer software]. Natick, USA.

Tomite, T., Saito, H., Kijima, H., Ishikawa, N., Hatakeyama, Y., Tazawa, H., et al. (2021). Evaluation of anteroposterior accelerometric change after bi-cruciate stabilized total knee arthroplasty and posterior stabilized total knee arthroplasty. The Knee , 32, 121-130.

Tomkins-Lane, C. C., & Haig, A. J. (2012). A review of activity monitors as a new technology for objectifying function in lumbar spinal stenosis. Journal of Back and Musculoskeletal Rehabilitation , 25 (3), 177-185.

Turcot, K., Aissaoui, R., Boivin, K., Pelletie, M., Hagemeister, N., & de Guise, J. A. (2008). New accelerometric method to discriminate between asymptomatic subjects and patients with medial knee osteoarthritis during 3-D gait. IEEE Transactions on Biomedical Engineering , 55 (4), 1415-1422.

Turcot, K., Hagemeister, N., de Guise, J. A., & Aissaoui, R. (2011). Evaluation of unipodal stance in knee osteoarthritis patients using knee accelerations and center of pressure. Osteoarthritis and Cartilage , 19 (3), 281-286.

UK BioBank (n.d.). UK BioBank Schowcase: Data-Fields: 40000, 90010, 110006 & 53. Retrieved March 2018, from UK BioBank Schowcase: https://biobank.ndph.ox.ac.uk/showcase/

Van den Dikkenberg, N., Meijer, O. G., Van Der Slikke, R. M., Van Lummel, R. C., Van Dieën, J. H., Pijls, B., et al. (2002). Measuring functional abilities of patients with knee problems: Rationale and construction of the DynaPort knee test. Knee Surgery, Sports Traumatology, Arthroscopy , 10 (4), 204-212.

van der Straaten, R., Wesseling, M., Jonkers, I., Vanwanseele, B., Bruijnes, A. K., Malcorps, J., et al. (2020). Functional movement assessment by means of inertial sensor technology to discriminate between movement behaviour of healthy controls and persons with knee osteoarthritis. Journal of neuroengineering and rehabilitation , 17 (1), 1-12.

Van Hees, V. T., Slootmaker, S. M., De Groot, G., Van Mechelen, W., & Van Lummel, R. C. (2009). Reproducibility of a triaxial seismic accelerometer (DynaPort). Medicine & Science in Sports & Exercise , 41 (4), 810-817.

van Hemert, W. L., Senden, R., Grimm, B., Kester, A. D., van der Linde, M. J., & Heyligers, I. C. (2009). Patella retention versus replacement in total knee arthroplasty; functional and clinimetric aspects. Archives of Orthopaedic and Trauma Surgery , 129 (2), 259-265.

Vangeneugden, J., Verlaan, L., Oomen, P., Liu, W.-Y., Peters, M., Natour, N., et al. (2020). Signatures of knee osteoarthritis in women in the temporal and fractal dynamics of human gait. Clinical Biomechanics , 76, 105016.

Vissers, M. M., Bussmann, J. B., De Groot, I. B., Verhaar, J. A., & Reijman, M. (2011). Walking and chair rising performed in the daily life situation before and after total hip arthroplasty. Osteoarthritis and Cartilage , 19 (9), 1102-1107.

Weihrich, K. S. (2022, April 27). Aston Gait Detection Algorithm. Retrieved from https://github.com/KatySaWeihrich/AGDA

WHO. (n.d.). Physical activity. Retrieved November 2011, 2017, from World Health Organisation (WHO): http://www.who.int/topics/physical_activity/en/

Wouda, F. J., Jaspar, S. L., Harlaar, J., van Beijnum, B.-J. F., & Veltink, P. H. (2021). Foot progression angle estimation using a single foot-worn inertial sensor. Journal of neuroengineering and rehabilitation , 18 (1), 1-10.

Wu, A. R., Simpson, C. S., van Asseldonk, E. H., van der Kooij, H., & Ijspeert, A. J. (2019). Mechanics of very slow human walking. Nature Research: Scientific Reports , 9 (1), 1-10.

Yamada, M., Aoyama, T., Mori, S., Nishiguchi, S., Okamoto, K., Ito, T., et al. (2012). Objective assessment of abnormal gait in patients with rheumatoid arthritis using a smartphone. Rheumatology International , 32 (12), 3869-3874.

Zengini, E., Hatzikotoulas, K., Tachmazidou, I., Steinberg, J., Hartwig, F. P., Southam, L., et al. (2017). The genetic architecture of osteoarthritis: insights from UK Biobank. bioRxiv , 174755.

Zhang, H., & Zhou, Y. (2021). Concept verification of a Remote Automatic Scoring System for Evaluating Knee Function after Total Knee Arthroplasty. The Journal of Knee Surgery , 34 (04), 464-470.

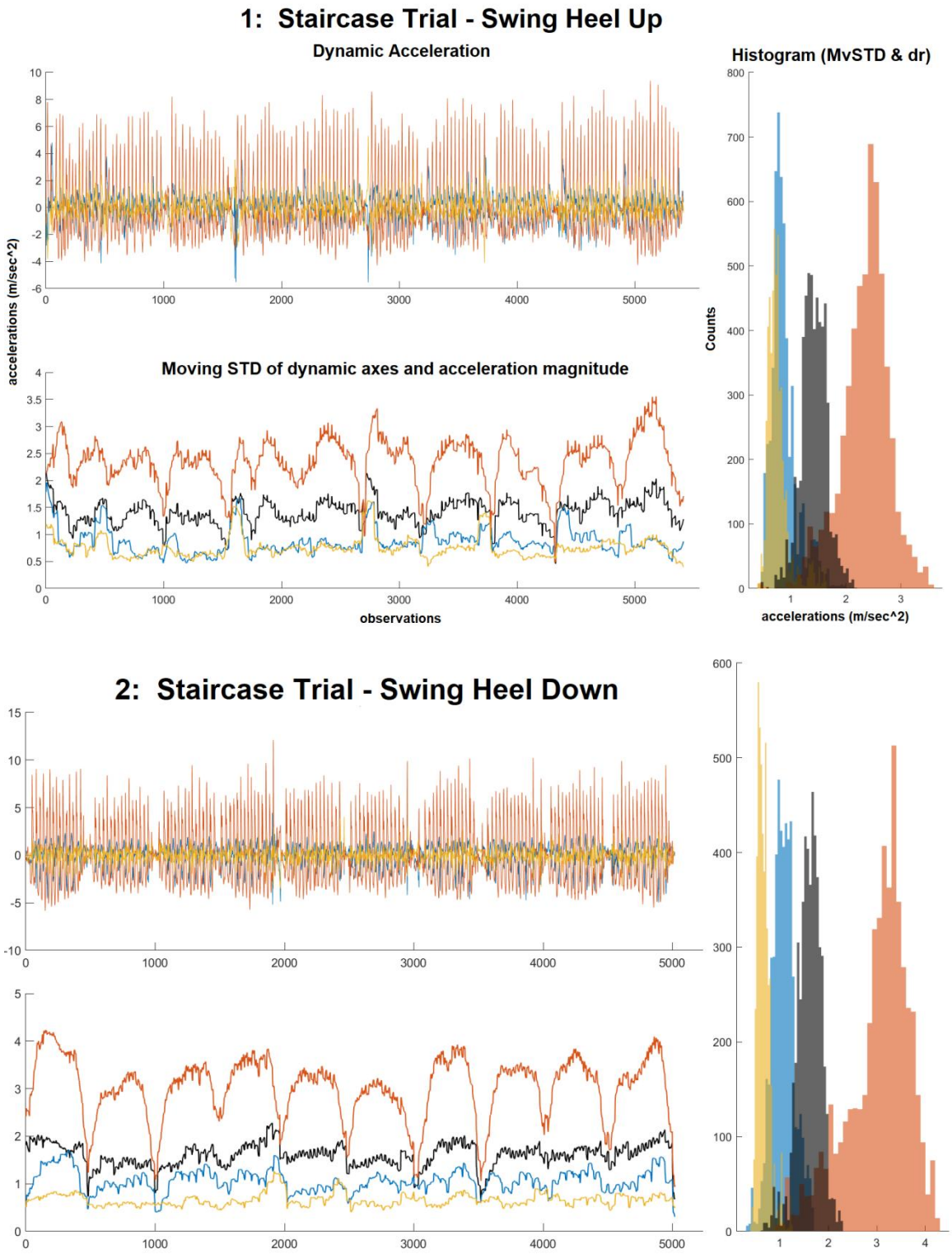
Zhang, H.-H., Yan, S.-H., Fang, C., Guo, X.-Y., & Zhang, K. (2016). Clinical evaluation and gait characteristics before and after total knee arthroplasty based on a portable gait analyzer. Orthopaedic Surgery , 8 (3), 360-366.

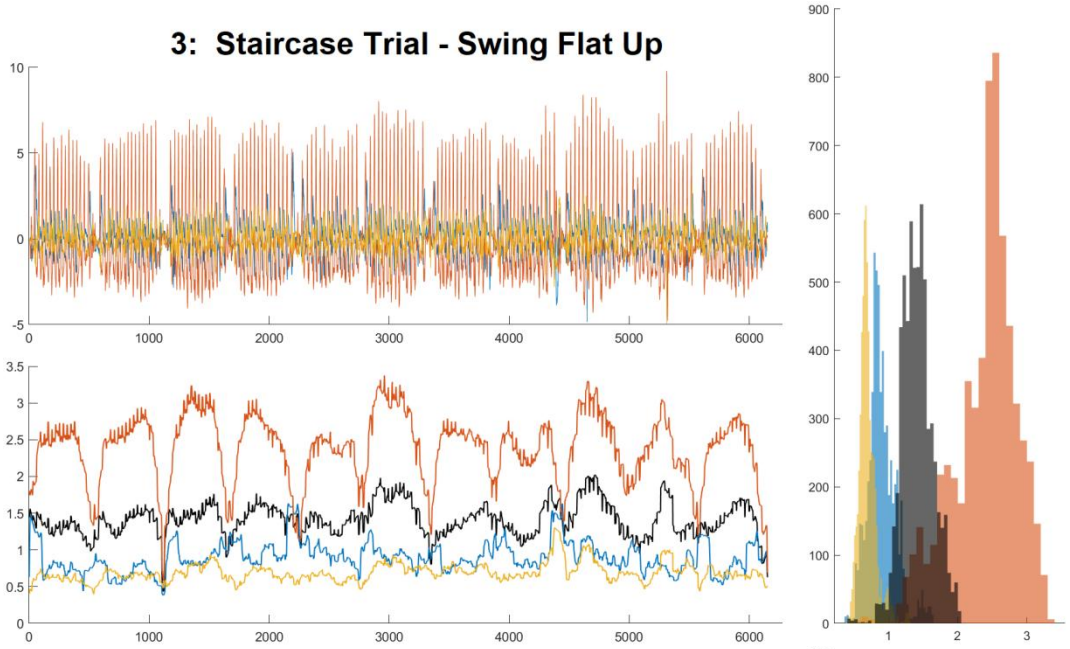
Zhang, Y., & Jordan, J. M. (2010). Epidemiology of osteoarthritis. Clinics in Geriatric Medicine , 26 (3), 355-369.

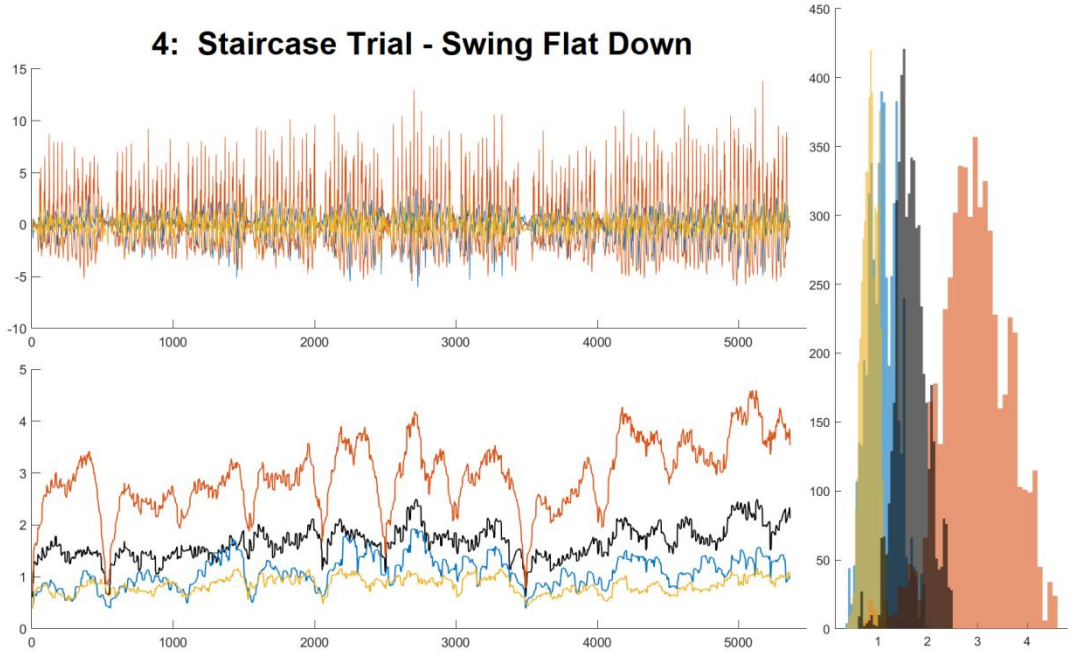
Zijlstra, A., de Bruin, E. D., Bruins, N., & Zijlstra, W. (2008). The step length-frequency relationship in physically active community-dwelling older women. European Journal of Applied Physiology , 104 (3), 427-434.

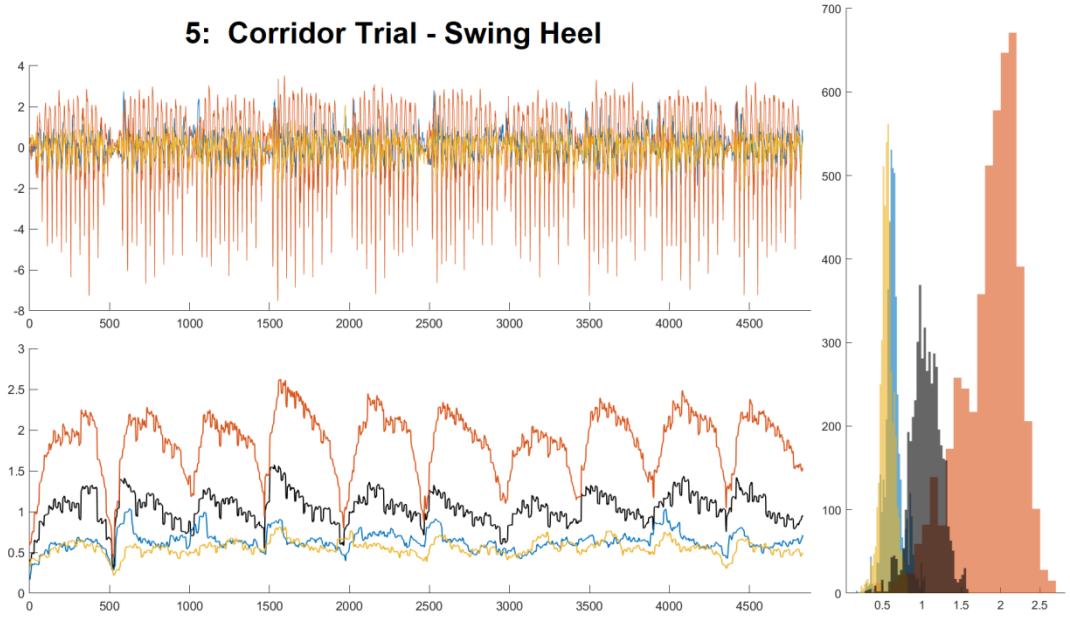
11. Appendix

*Figure 88 - **Overview of the SRDS data by the Updated Labels**. Blue = x-axis, red = y-axis, yellow = z-axis; black = dr. 50 observations = 1 sec.*

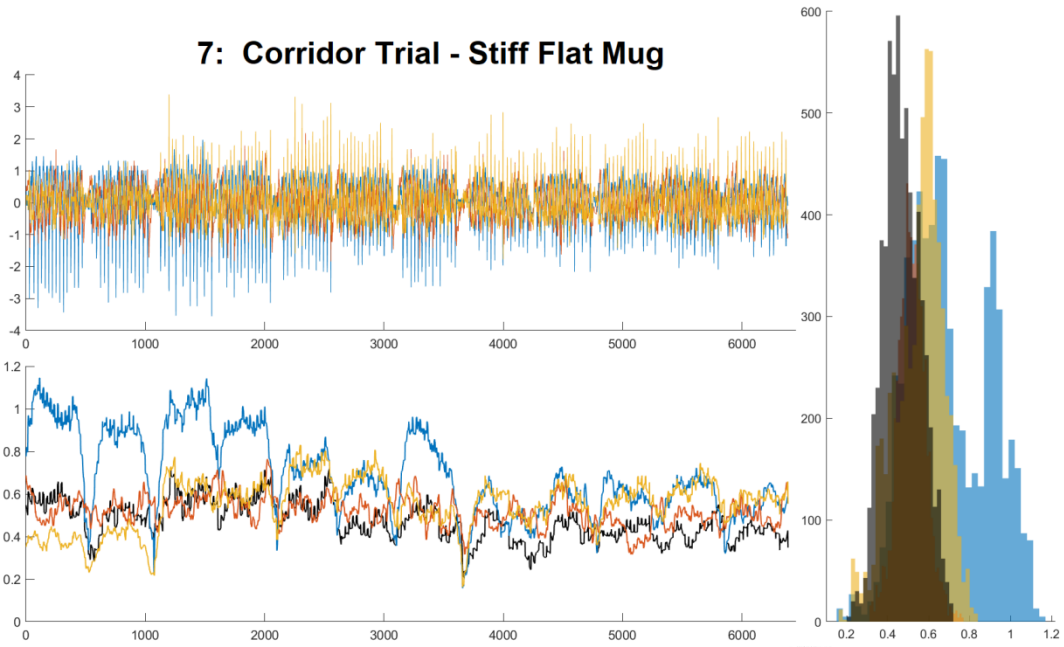



3: Staircase Trial - Swing Flat Up


4: Staircase Trial - Swing Flat Down

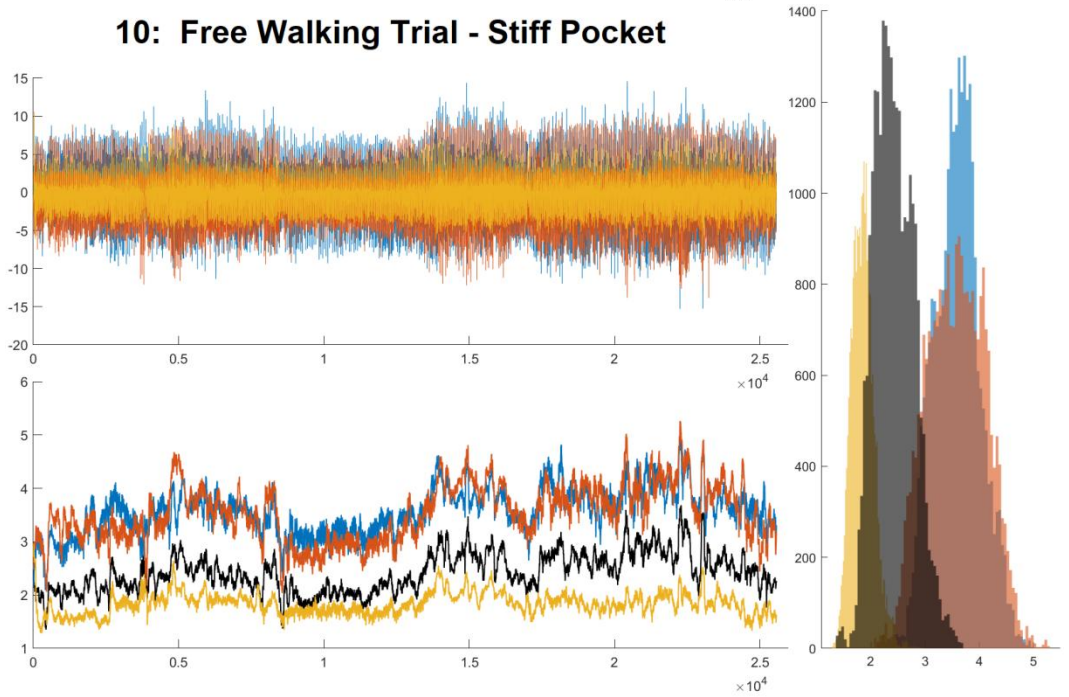

5: Corridor Trial - Swing Heel

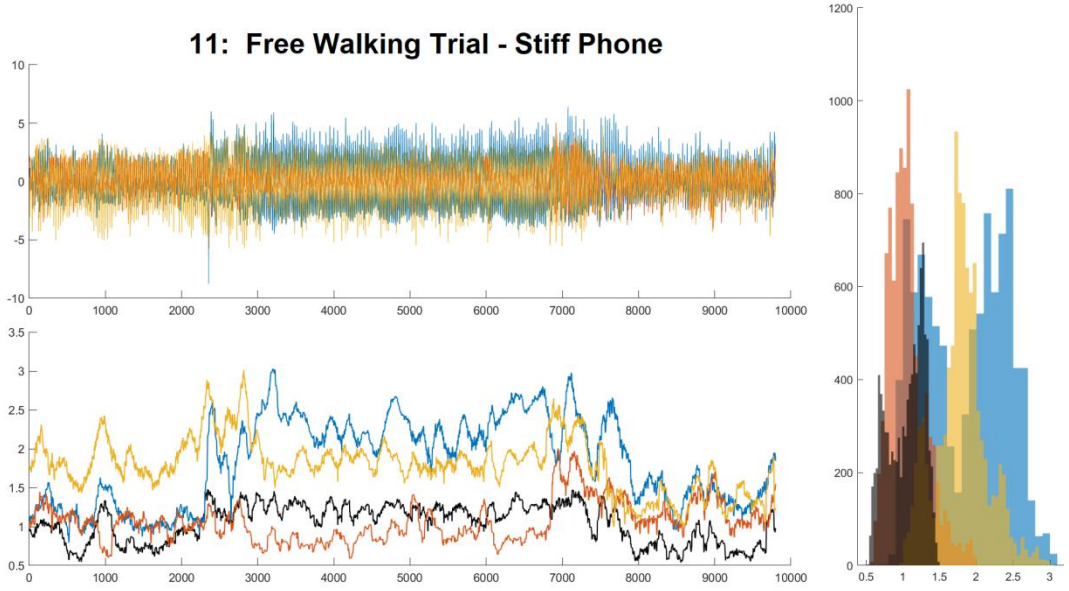

6: Corridor Trial - Swing Flat

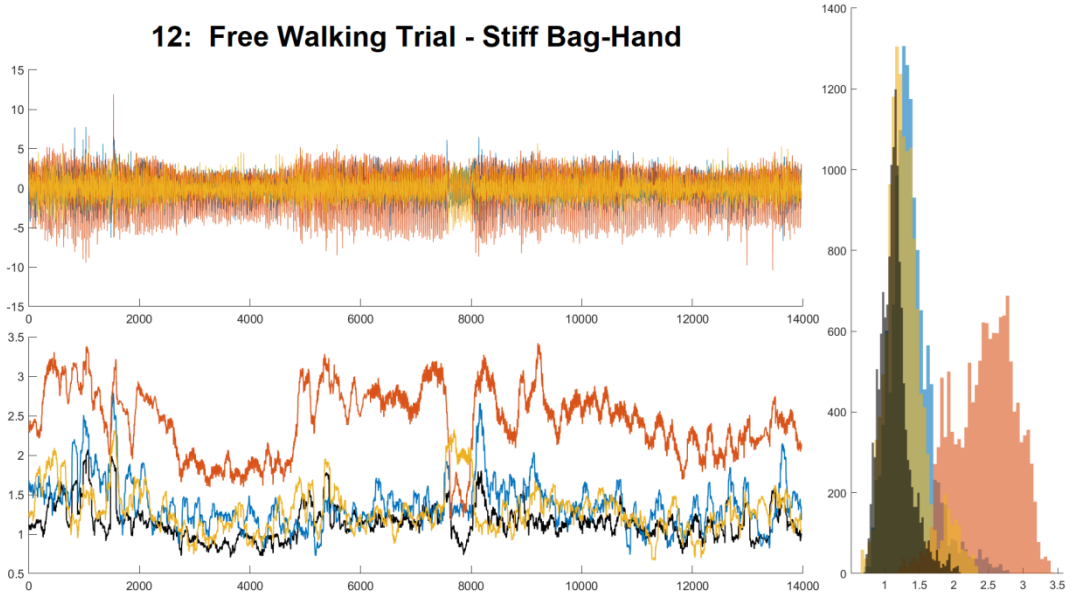

7: Corridor Trial - Stiff Flat Mug

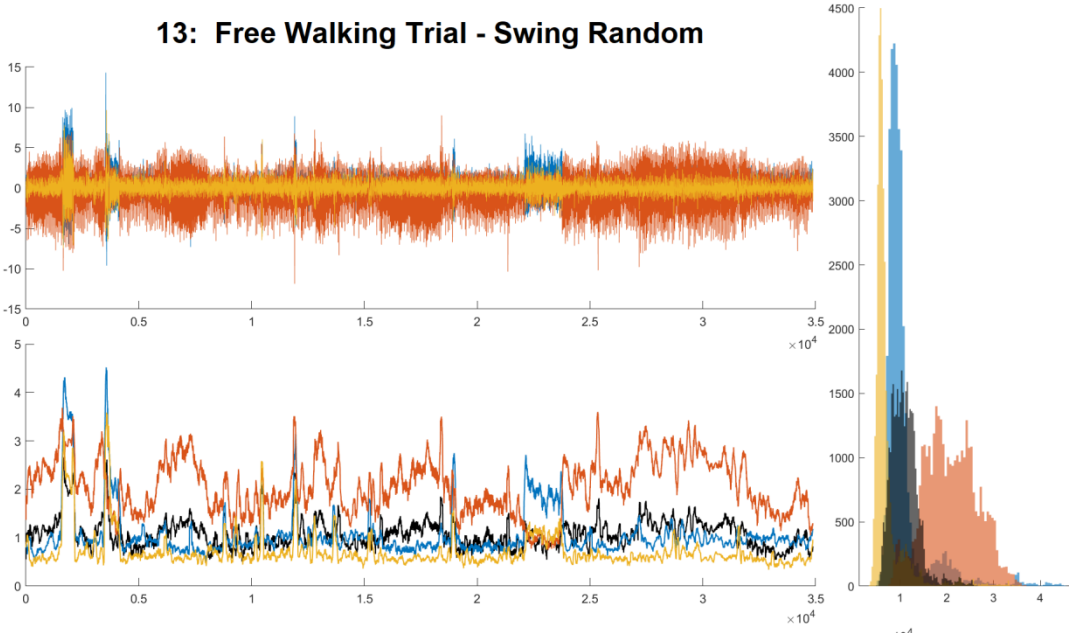

8: Corridor Trial - Stiff Flat Bag-Shoulder


9: Free Walking Trial - Swing Clean


10: Free Walking Trial - Stiff Pocket


11: Free Walking Trial - Stiff Phone


12: Free Walking Trial - Stiff Bag-Hand


13: Free Walking Trial - Swing Random

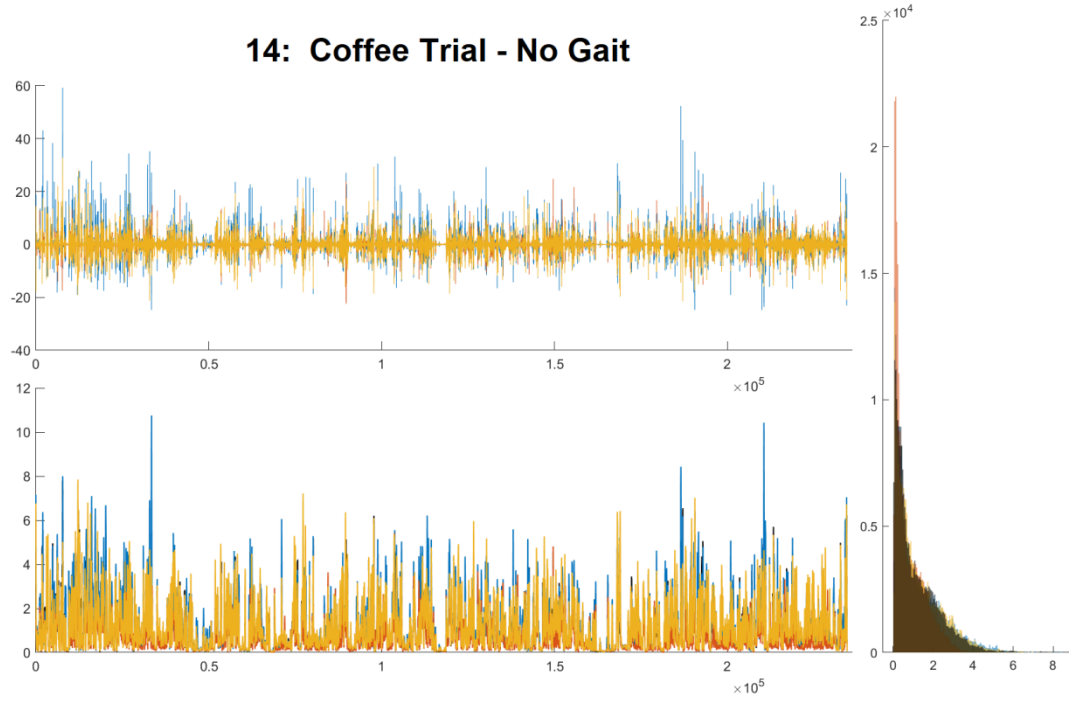
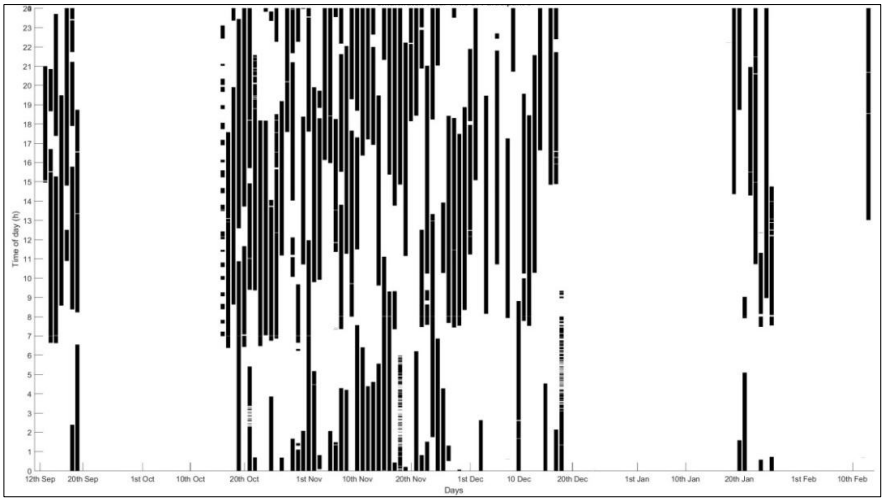
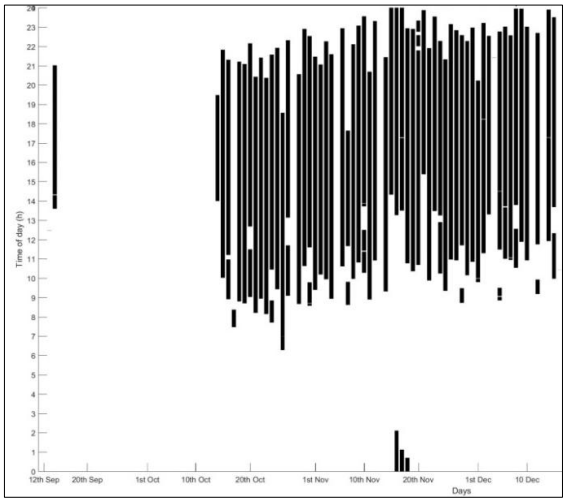

14: Coffee Trial - No Gait

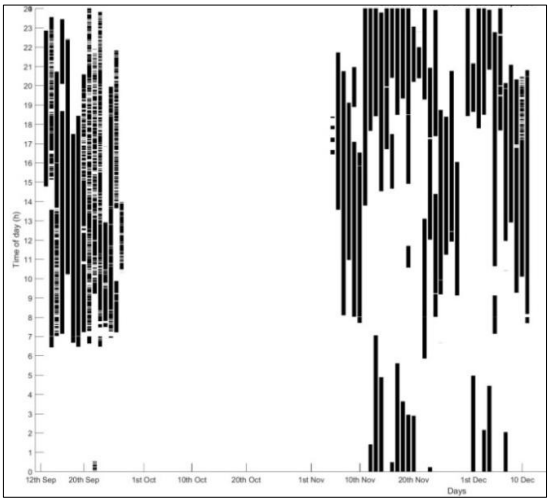
Figure 89 - ***Wear Time by User***



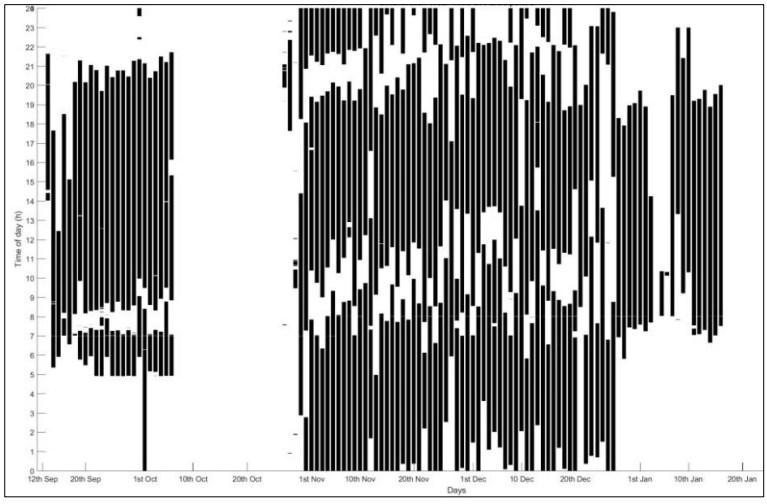
Participant 03



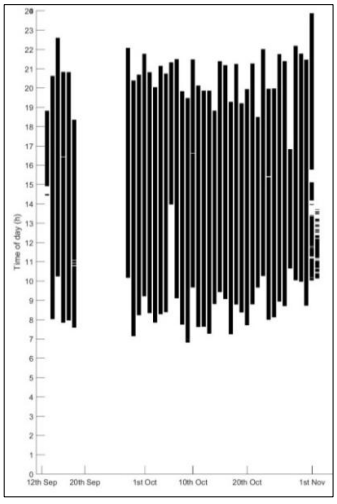
Participant 05



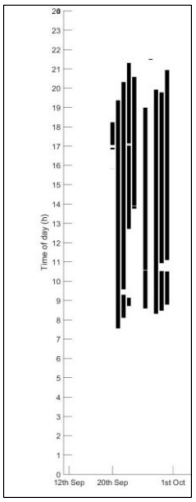
Participant 07



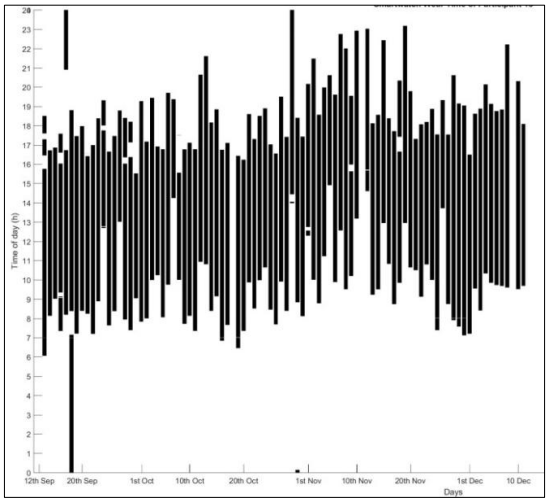
Participant 08



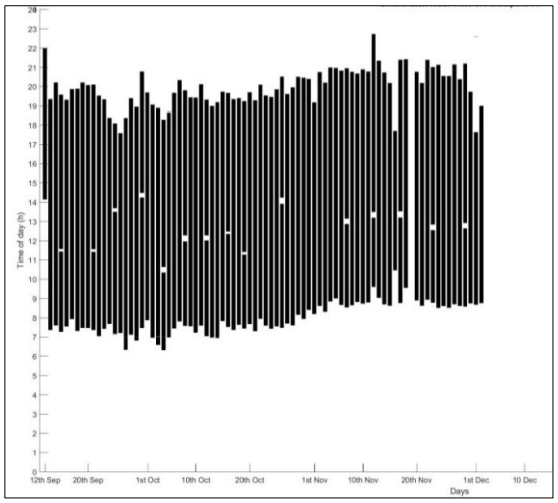
Participant 09



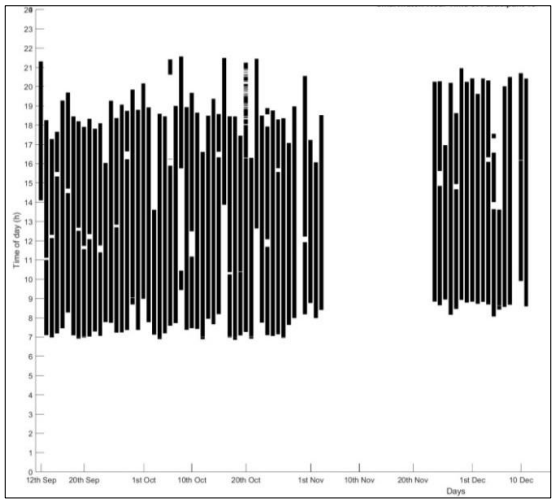
Participant 10



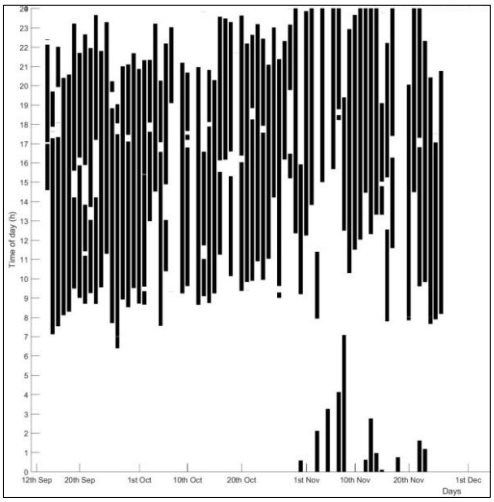
Participant 13



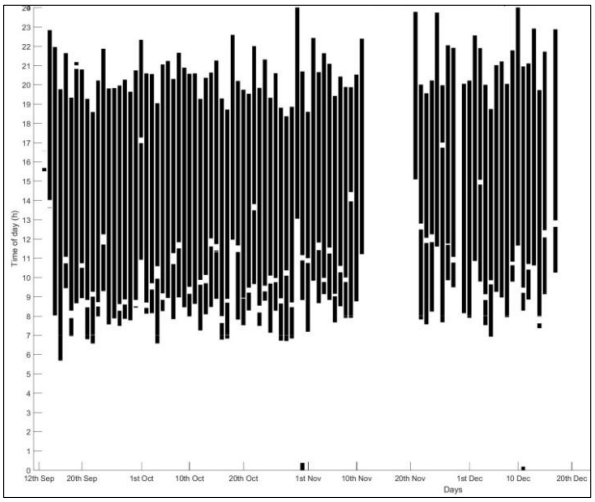
Participant 14



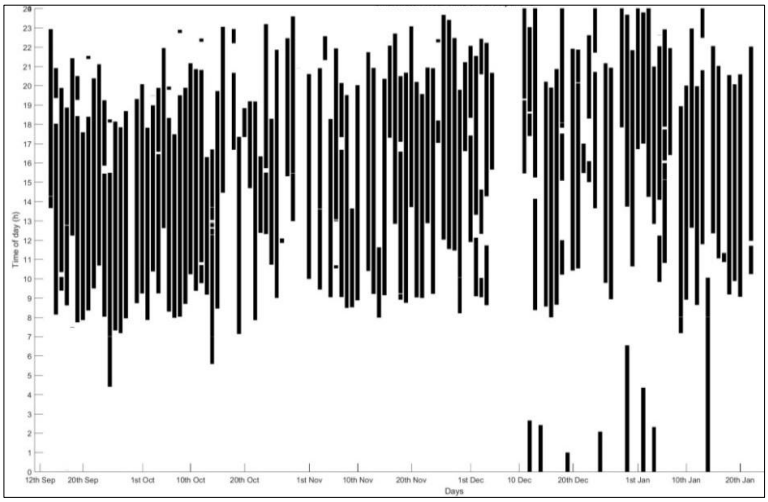
Participant 15



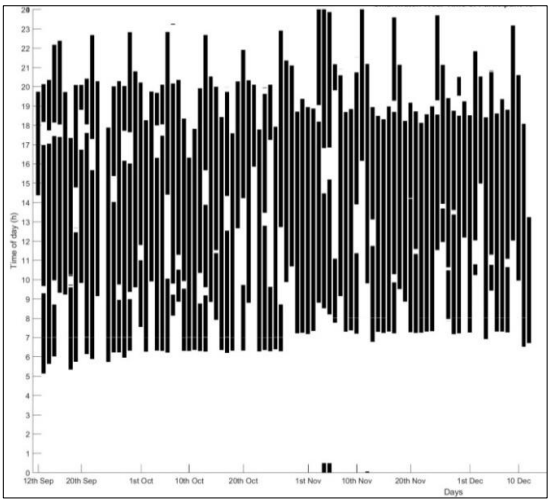
Participant 16



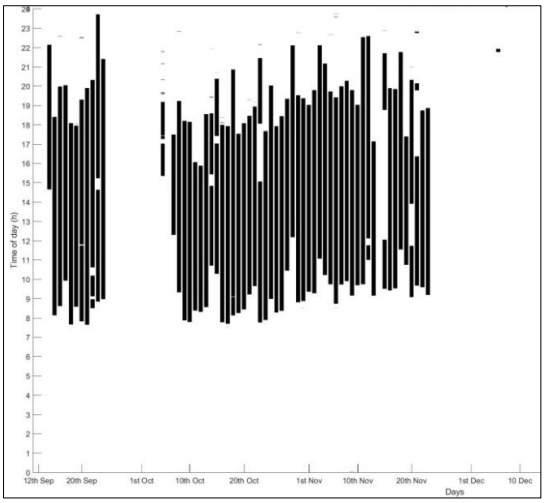
Participant 17



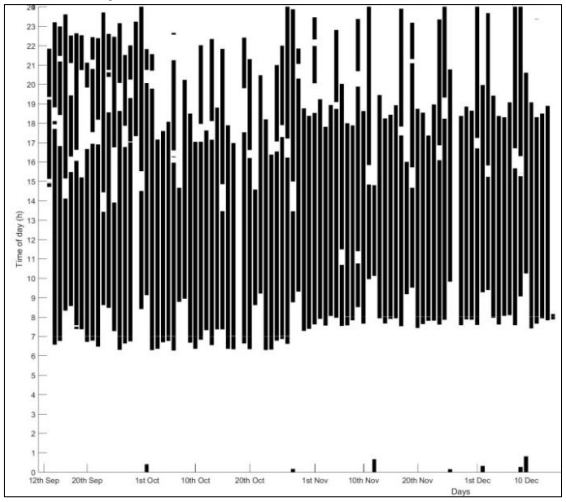
Participant 18



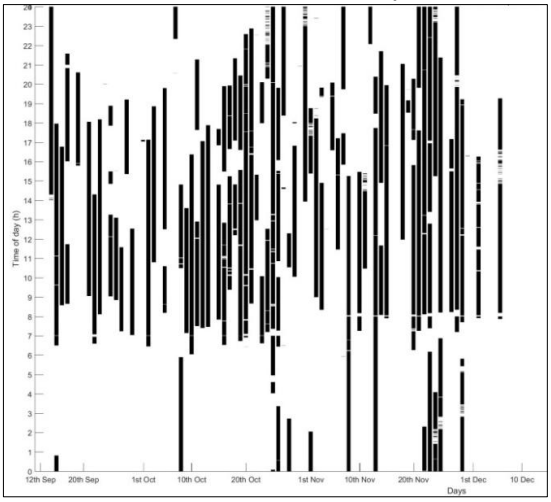
Participant 19



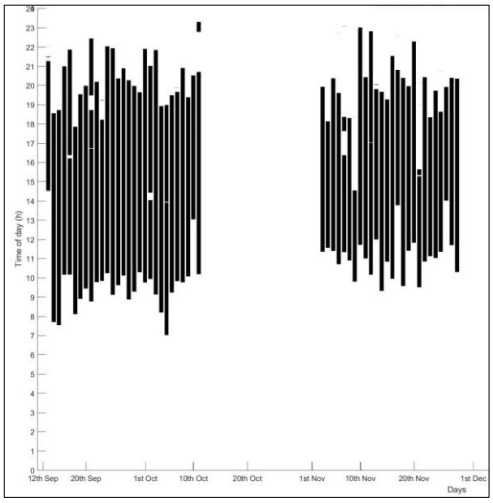
Participant 20



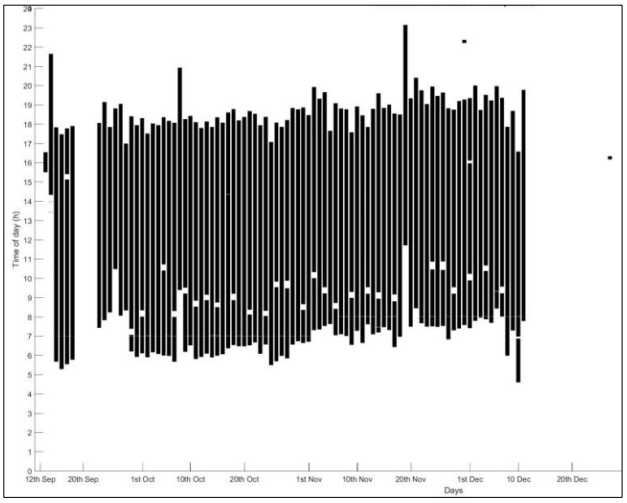
Participant 21



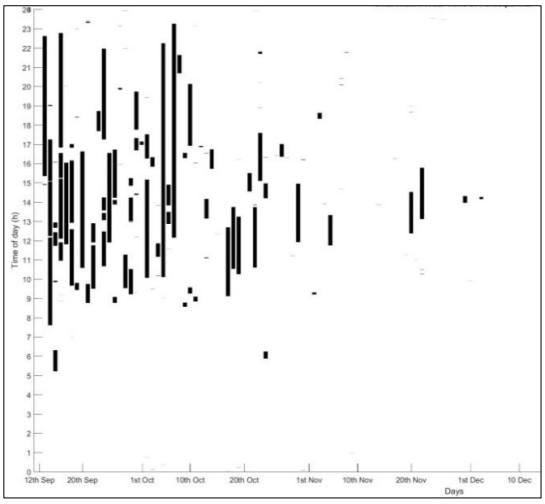
Participant 23



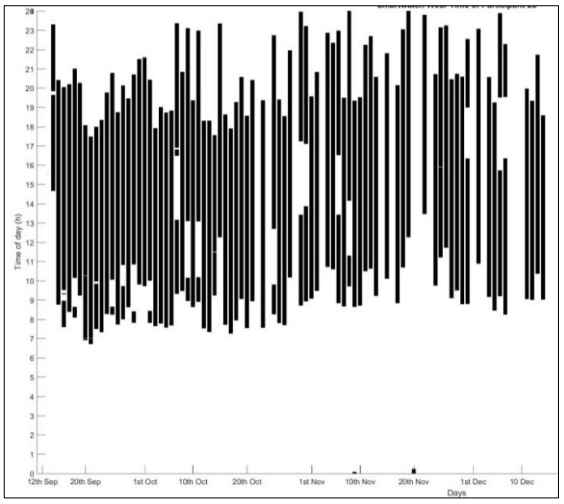
Participant 25



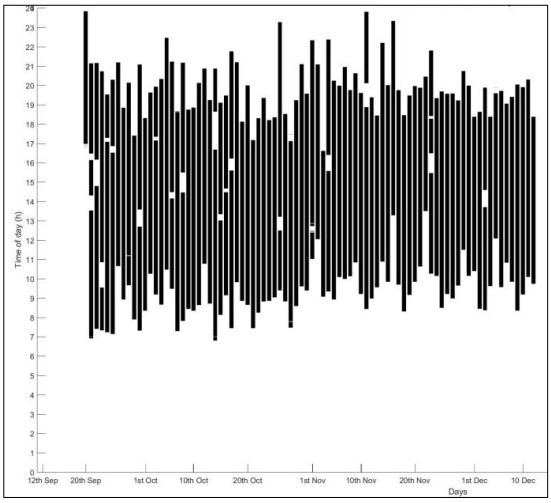
Participant 26



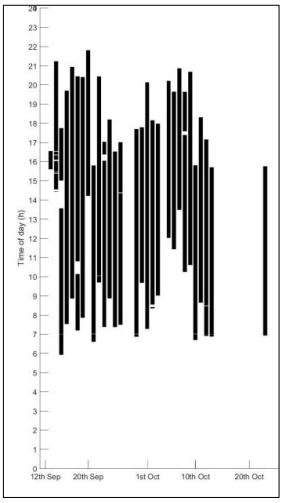
Participant 27



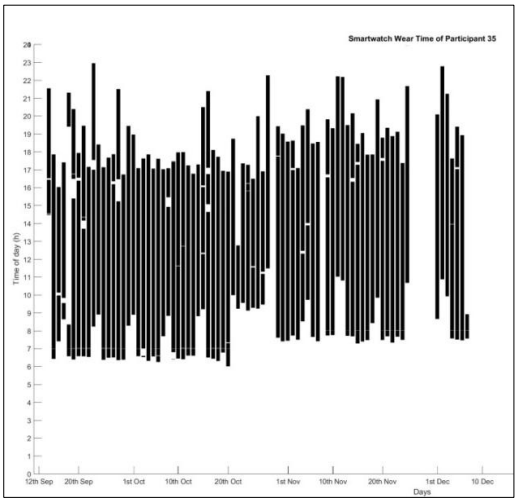
Participant 28



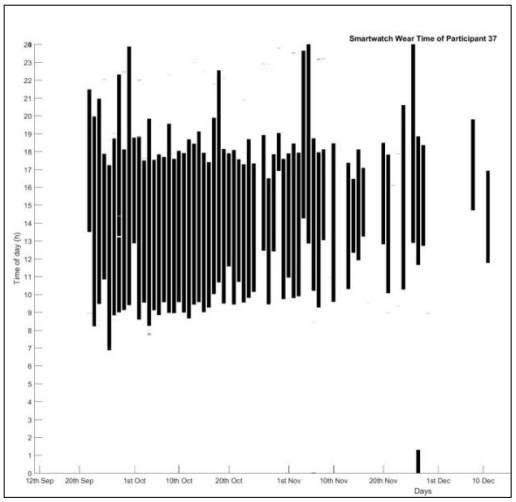
Participant 30



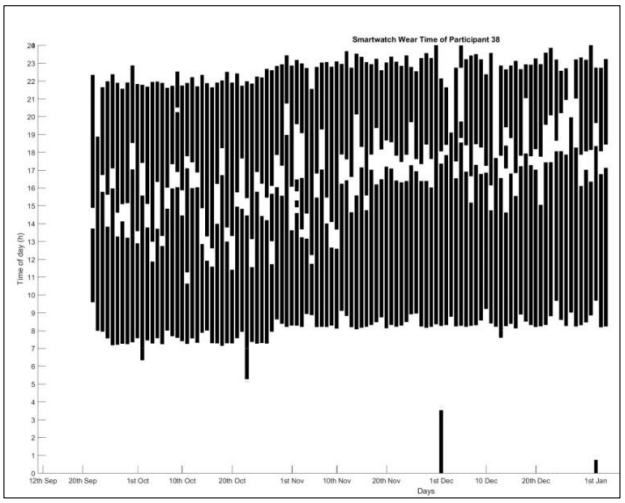
Participant 34



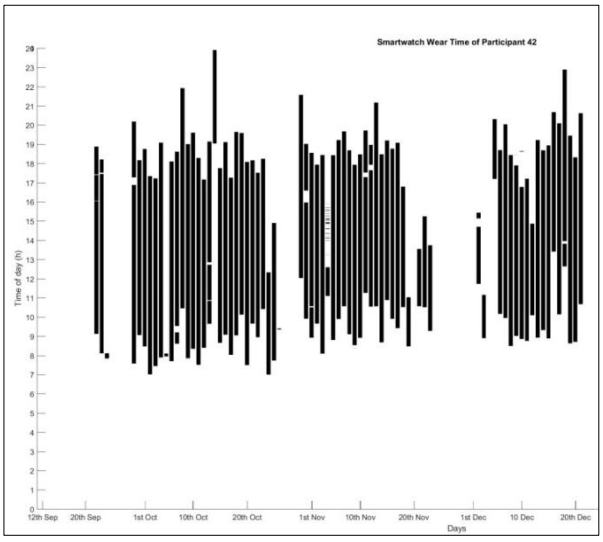
Participant 35



Participant 37



Participant 38



Participant 42

Figure 90 - ***Gyroscope data example and Sample frequency observations*** *of participant 7 for the 19th September.*

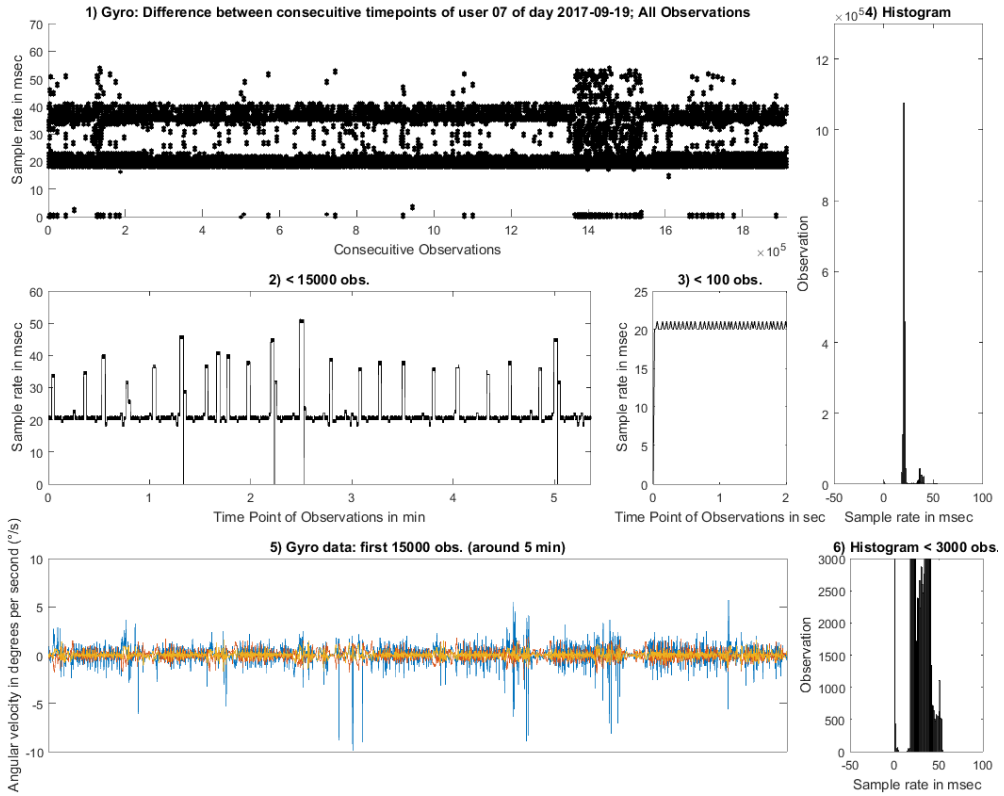


Figure 91 - ***Magnetometer data example and Sample frequency observations*** *of participant 7 for the 19th September.*

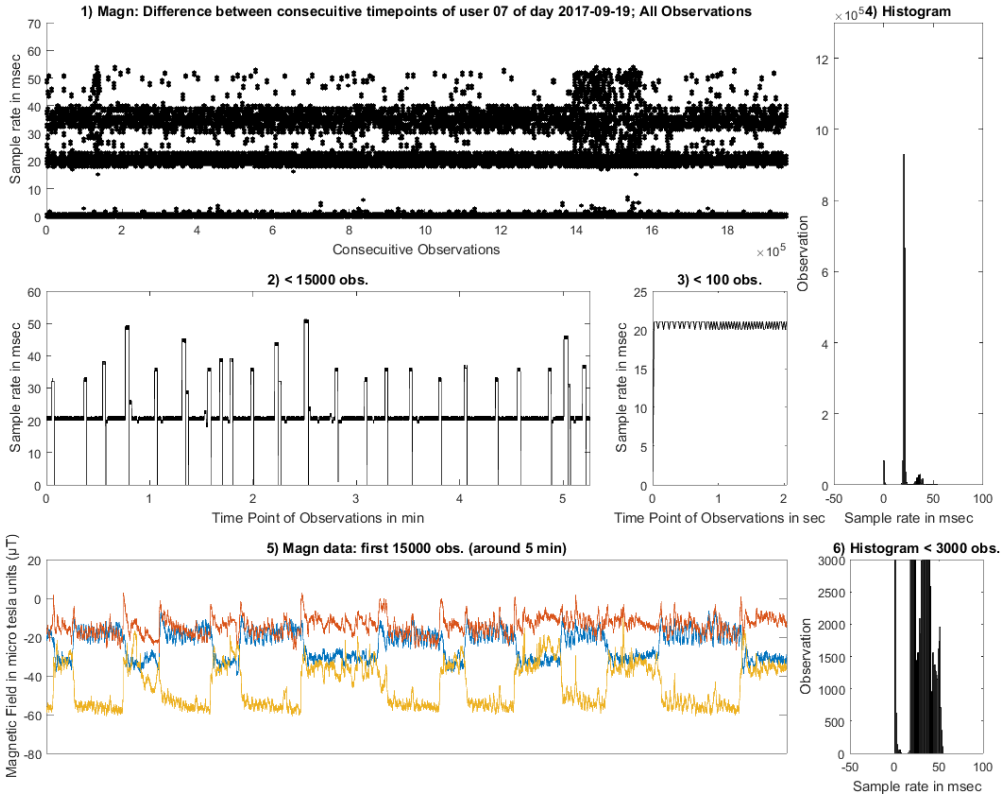


*Figure 92 - **Barometer data example and Sample frequency observations** of participant 7 for the 19th September.*

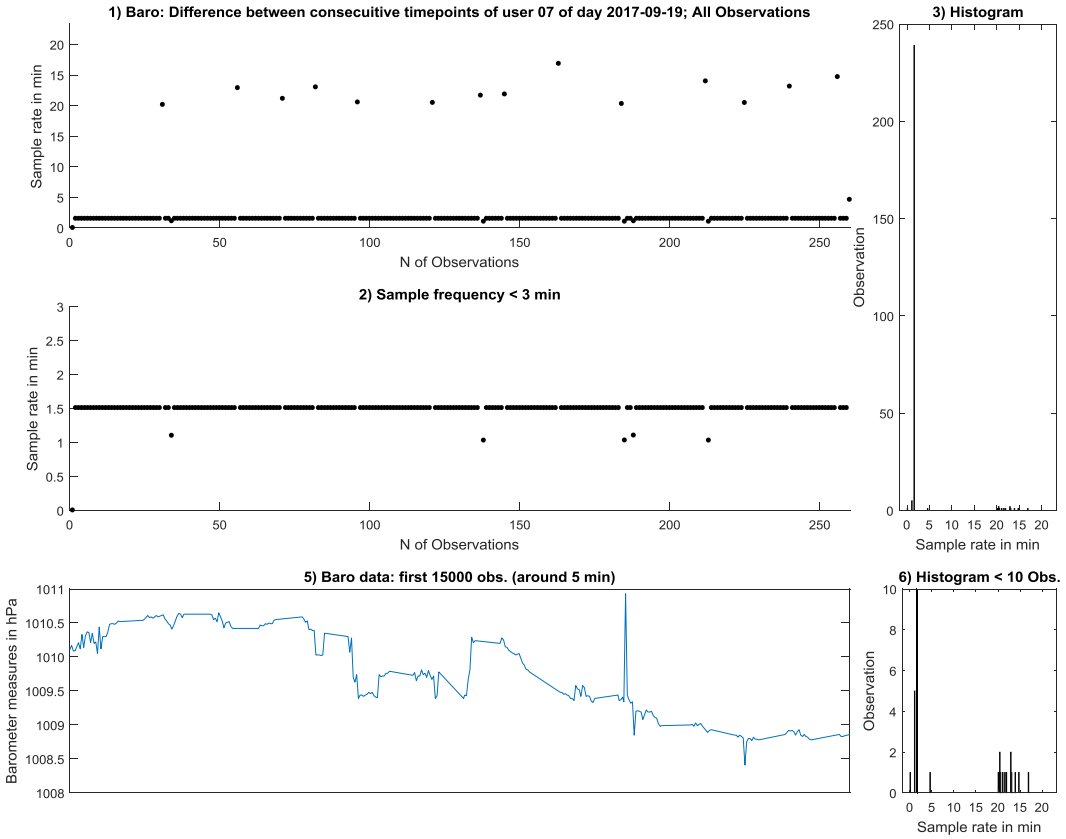


*Figure 93 – **Heart Rate data example and Sample frequency observations** of participant 7 for the 19th September.*

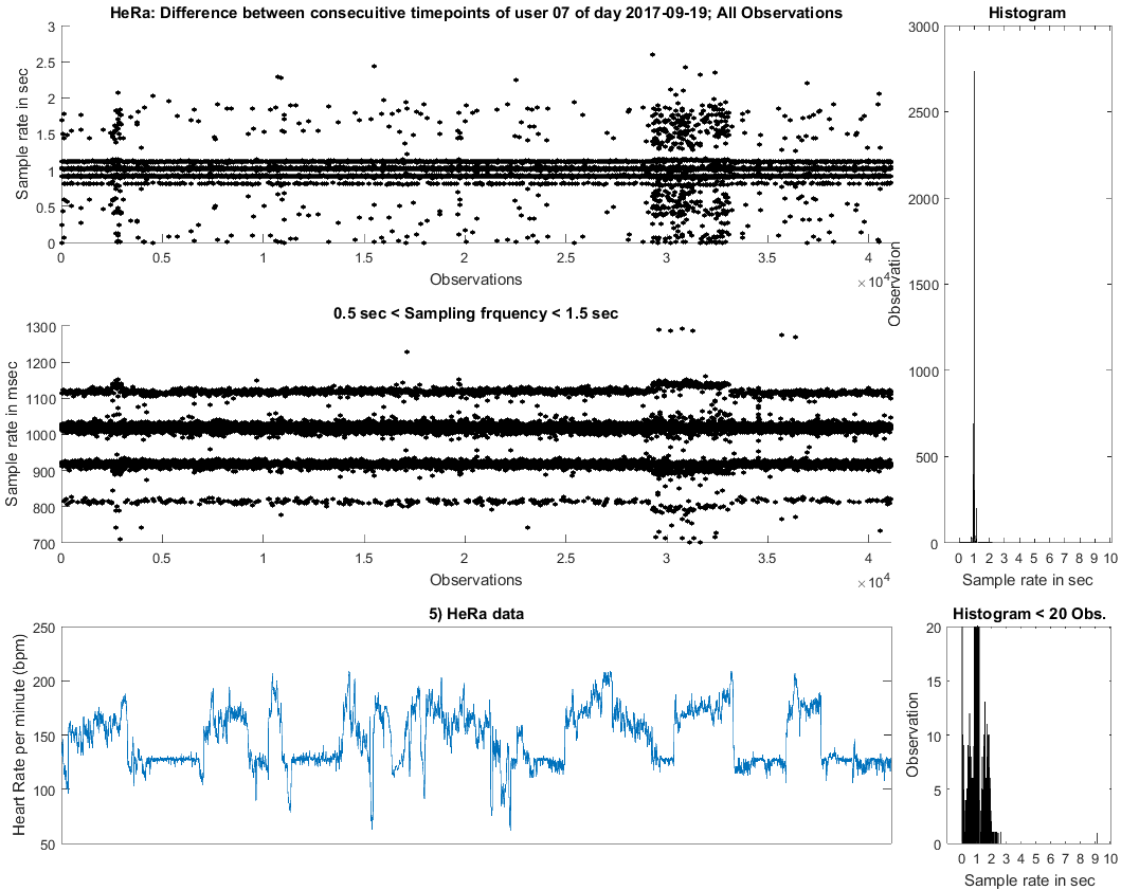
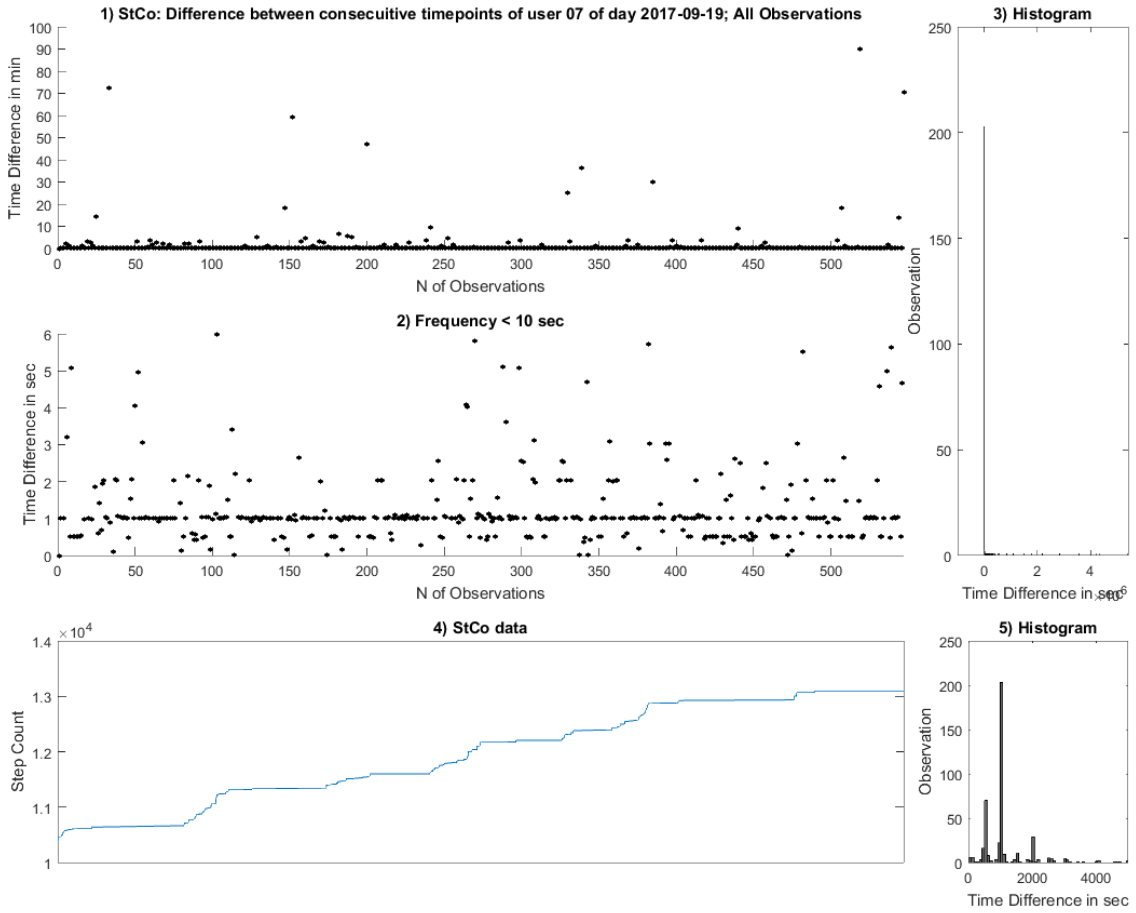


Figure 94 – ***Step count data example and Sample frequency observations*** *of participant 7 for the 19th September.*



The following pages will contain the Aston Gait Detection Algorithm (AGDA) codes, which consist of the two main codes: the Gait Detection Algorithm (GDA) and the Step Parameter Extraction (SPE) codes. Functions will be printed below the codes they belong to. The codes start at the following pages:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%                  AstonGDACode_GaitDetectionAlgorithm.m                   %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Fist Step of The AGDA: Identify Data Representing Walking Behaviours
%     The algorithm is build to read in accelerometer data and Output a
%     vector z that identifies sections of walk (1) and non-walking (0)

%% Input needed:
% Please consider changes to these parts of the code before using it:
% Under "Version" select what type of format your input data has. You can
%       add new formats for new datasets and select them using this the
%       variable "Version_Dataset"
%       If you onlx want to run a subset of your dataset use "x_str" and
%       "x_end" to select that subset. The files will be run in
%       alphabetical order with x_str beeing the first and x_end the last
%       file to be run. "CreateIMG" indicates if the create of figures in
%       the data is wanted. Should be turned of for larger datasets.
%       "iteration_add_on" adds characters to the Output file names.
%       For Biobank dataset select folder to run code on using "folder n".
% Under "folder/file location/name" select the folder the data is located
%       in using variable "CHOICE_dir_basis".
%       use variable "filename_load" to input a string that is shared by
%       all files that you want to analyse.
%       "CHOICE_dir_basis_sup" holds the directory to the FUNCTIONS needed
%       to run the code.
%       "CHOICE_dir_Output" select the folder you want the output data and
%       images to be located in. Use [] to stay in the Input folder.
% Under "Variabes" cange variables that are needed for the code. Please
%       take a:look over them before runing the code on a new dataset.
%       "var_feq" describes the frequency in Hz at which the data was
%       recorded.
% Under "Identify Data files within the Folder" a function named
%       "func_GDA_SelectData" extracts a list of files to be analysed.
%       New "Version_Dataset" have to be adjusted here.

%% List of Functions Needed
% Should be included in the Supplementary folder
% For the Gait Algorithm:
%    func_GDA_SelectData.m
%    func_GDA_LoadData.m
%    func_GDA_DASA.m
%    func_GDA_mSTD.m
%    func_GDA_Epi.m
%    func_GDA_AC.m

% Additional Codes needed from File Exchange:
%    movingstd (https://uk.mathworks.com/matlabcentral/fileexchange/9428
%               movingstd-movingstd2)
%    peakseek  (https://uk.mathworks.com/matlabcentral/fileexchange/26581-
%               peakseek)

%% House-Keeping
clear all       % clear all variables from the current workspace
fclose('all');  % close all open files
close all       % deletes all figures whose handles are not hidden
tic             % set timer

%% Version
CreateIMG = 0;
    % Figures: 1 = on; 0 = off
Version_Dataset = 1;
    % 1 = Selfrecorded Dataset
    % 2 = KOALAP Dataset
    % 3 = BioBank Dataset
    % 4 = dataset 1, but apply preprocessing from Biobank
x_str = 1;  % point to start the data extraction (musst be 1 or higher)
x_end = []; % point to end the data extraction (must be [] or a number > 1)
iteration_add_on = '_V1'; % addition to savefile name
if Version_Dataset == 3
```

```
    folder_n = '31'; % group number of the folder the code should run on
end

fprintf(['Dataset Type used: ' num2str(Version_Dataset)  ' \n'])

%% folder/file location/name
% CHOICE_dir_basis      Location of dataset
% CHOICE_dir_Output     Intended folder for Output data
%                       Lave empty to save data with the input dataset
% filename_load         Part of the filename that is identical within all
%                       target filed of the dataset
% CHOICE_dir_basis_sup  Location of needed additional Codes

if (Version_Dataset == 1) || (Version_Dataset == 4)
    CHOICE_dir_basis = 'D:\PhD_Matlab\Data_SelfRecorded\';
    filename_load = 'StairCWalkCoffeeFWalk';
    CHOICE_dir_Output = 'D:\PhD_Matlab\Max_Output\AGDA_Codes\Output_AGDA\';
CHOICE_dir_basis_sup='D:\PHD_FinalThesisSumission\newCodes\old_new_codes\';
end
if Version_Dataset == 2
    User_List = ['USER_03'; 'USER_05';'USER_07'; 'USER_08';'USER_09'; ...
        'USER_10';'USER_13'; 'USER_14';'USER_15'; 'USER_16';'USER_17'; ...
        'USER_18';'USER_19'; 'USER_20';'USER_21'; 'USER_23';'USER_25'; ...
        'USER_26';'USER_27'; 'USER_28';'USER_30'; 'USER_34'; 'USER_35'; ...
        'USER_37'; 'USER_38';'USER_42'];
    CHOICE_dir_basis = '.../data/extracted/04_CoFi4-Accl/';
    filename_load = 'CoFi4_';
    CHOICE_dir_Output = '.../data/extracted/AstonGDA_Output/';
    CHOICE_dir_basis_sup = '.../scripts/AGDA_Codes/';
end
if Version_Dataset == 3
CHOICE_dir_basis=['.../Data_extracted_UKB/Participant_Group' folder_n '/'];
    filename_load = '_90001_0_0';
    CHOICE_dir_Output_pre = [];
    CHOICE_dir_basis_sup = '.../AGDA_Codes/';
end

% add the folders needed to pathway
addpath(CHOICE_dir_basis_sup)
addpath(CHOICE_dir_basis)

% open the Ouput folder
if ~isempty(CHOICE_dir_Output); cd(CHOICE_dir_basis);
else; cd(CHOICE_dir_Output); end

% update Variable Record
SetUp_Variables.filename_load = filename_load;
SetUp_Variables.CHOICE_dir_basis = CHOICE_dir_basis;
SetUp_Variables.CHOICE_dir_basis_sup = CHOICE_dir_basis_sup;
SetUp_Variables.CHOICE_dir_Output = CHOICE_dir_Output;

%% Variabes
% do not change during the run
% Sample Frequency of the data
if (Version_Dataset==1)||(Version_Dataset==2)||(Version_Dataset==4)
    var_feq = 50; end
if Version_Dataset == 3; var_feq = 100; end

% Determine min and max Step Frequency
slowWalk_StepSec = 0.78;
    slowWalk_MsecStep = 1/slowWalk_StepSec;
    slowWalk_obs = slowWalk_MsecStep*var_feq;
    slowWalk_Hz = slowWalk_StepSec;
fastWalk_StepSec = 3.26;
    fastWalk_MsecStep = 1/fastWalk_StepSec;
    fastWalk_obs = fastWalk_MsecStep*var_feq;
    fastWalk_Hz = fastWalk_StepSec;

% thresholds for episode length.
minEpisode_steps = 5;
```

```
QualityminEpisode_steps = 15;

% std threshold
if (Version_Dataset==1)||(Version_Dataset==2)||(Version_Dataset==4)
    STDthld_standart = 0.6; end
if Version_Dataset == 3; STDthld_standart = 0.3; end
    % note: Biobank uses something different then KOALAP because of the
    % magnitude of general magnitude the data (not techniquly nessesary,
    % but better)

% update Variable Record
if (Version_Dataset == 1)||(Version_Dataset == 2)
    SetUp_Variables.var_feq = var_feq;
    SetUp_Variables.slowWalk_StepSec = slowWalk_StepSec;
    SetUp_Variables.slowWalk_obs = slowWalk_obs;
    SetUp_Variables.fastWalk_StepSec = fastWalk_StepSec;
    SetUp_Variables.fastWalk_obs = fastWalk_obs;
    SetUp_Variables.minEpisode_steps = minEpisode_steps;
    SetUp_Variables.QualityminEpisode_steps = QualityminEpisode_steps;
end

%% save Variable Record
if (Version_Dataset == 2)
    save(['SetUP_Variables' iteration_add_on '.mat'],'SetUp_Variables')
    save(['SetUP_Variables_idv' iteration_add_on '.mat'],'CreateIMG',...
        'Version_Dataset','x_str','x_end','slowWalk_obs','fastWalk_obs',...
        'var_feq','filename_load','CHOICE_dir_basis_sup',...
        'CHOICE_dir_basis','CHOICE_dir_Output',...
        'minEpisode_steps','QualityminEpisode_steps')
end
if (Version_Dataset==3)||(Version_Dataset==4)||(Version_Dataset==1)
    save(['SetUP_Variables_idv' iteration_add_on '.mat'],'CreateIMG',...
        'Version_Dataset','x_str','x_end','slowWalk_obs','fastWalk_obs',...
        'var_feq','filename_load','CHOICE_dir_basis_sup',...
        'CHOICE_dir_basis','CHOICE_dir_Output',...
        'minEpisode_steps','QualityminEpisode_steps','STDthld_standart')
end

%% Prep Record files output
RunTime = []; RunTime.name = [];
if (Version_Dataset == 2) || (Version_Dataset == 1)
    PctReduc = []; PctReduc.name = [];
end
if (Version_Dataset == 3) || (Version_Dataset == 4)
    RunTime.date_num = now;
    RunTime.date_str = datestr(now,'yyyy-mm-dd HH:MM:SS.FFF');
    val_1msec = 1/24/60/60/1000;
end

%%%                              Start Code                                %%%%

%% loop for user folders in KOALAP
n_folderLoop = 1;
if Version_Dataset == 2; n_folderLoop = length(User_List(:,1));
    else; n_folderLoop = 1; end
for i_folderLoop = 1:n_folderLoop

    CHOICE_dir_use = CHOICE_dir_basis;
    if Version_Dataset == 2
        CHOICE_dir_use = [CHOICE_dir_basis User_List(i_folderLoop,:) '/'];
        x_end = [];
    end
    time_fstr = toc;

    %% Identify Data files within the Folder
    [filename_load_pre,filename_load_pos,filename_save_pre] = ...
        func_GDA_SelectData(Version_Dataset,CHOICE_dir_use,filename_load);
    toc_curr = toc;
    fprintf(['     Prep finished after: ' num2str(toc_curr) ' sec' ' \n'])
```

```
    % calculate list with datasize (FS_MB 0 File Size in megabyte)
    if Version_Dataset == 3
        Folder_structure     = dir(CHOICE_dir_use);
        idx_cwafiles = find(not(cellfun('isempty', (strfind(cellstr(...
            char({Folder_structure(1:end).name}')), '.cwa')))));
        FS_MB = ([Folder_structure(idx_cwafiles).bytes]/1000/1000)';
    end

    %% Loop loading data
    if isempty(x_end); x_end = length(filename_load_pre(:,1)) ; end
    if x_end - x_str > 5; CreateIMG = 0; end % failsave: to many files
    for i_file = x_str:x_end
        %% record start time of processing file
        time_fstr = toc;
        if (Version_Dataset==2)||(Version_Dataset==1)||(Version_Dataset==4)
            RunTime(i_file).str = toc;
        end

        %% create Filename to be loaded
        fn_load_pre = filename_load_pre(i_file,:);
        fn_load_pre(isspace(fn_load_pre)) = [];
        fn_load_pos = filename_load_pos(i_file,:);
        fn_load_pos(isspace(fn_load_pos)) = [];
            % note: empty spaces can be created in filename_load_pos when
            %       other documents in the folder have longer filenames

        %% Identif Filename building stones for current round
if Version_Dataset==1;fn_save=['SCwCFw_V1N' num2str(i_file)]; end
if Version_Dataset==2;fn_save=['U' fn_load_pre(9:10) fn_load_pos(2:5)];end
if Version_Dataset==3;fn_save=filename_save_pre(i_file,:); end
if Version_Dataset==4;fn_save=['SCwCFw_V4N' num2str(i_file)]; end

        %% Update Record Files
        RunTime(i_file).name = fn_save;
        if (Version_Dataset == 2) || (Version_Dataset == 1)
        PctReduc(i_file).name = fn_save; end

%% %%%%%%%%%%%%%%%%% Start of Gait Detection Functions %%%%%%%%%%%%%%%%% %%

%% Preperations
% Output in Command Window
if (Version_Dataset == 2) || (Version_Dataset == 1)
    fprintf([fn_save ' ' num2str( i_file) ' of ' ...
    num2str(length(filename_load_pre(:,1))) ' \n']); end
if (Version_Dataset == 3) || (Version_Dataset == 4)
    fprintf([fn_save ' ' num2str( i_file) ' of ' ...
    num2str(x_end) ' in folder ' CHOICE_dir_use(:,57:end-1) ' \n']); end
toc_next = toc;

% Set the whole Dataset as Potential Walking (all z = 1)
z_Base = [];
z_Base_3D = [];

%% 1) extract data
[data_d, data_g, data_t, lgt_TotalData,data_dr,data_Lab,~,~] = ...
    func_GDA_LoadData(fn_load_pre, fn_load_pos,...
    CHOICE_dir_use,Version_Dataset,var_feq);
if lgt_TotalData > var_feq*43200; CreateIMG = 0; end % failsave:large files

% note time
toc_curr = toc-toc_next;
lgt_TD_h = lgt_TotalData/var_feq/60/60;
fprintf(['     DataLoad took ' num2str(toc_curr) 'sec/' ...
     num2str(lgt_TD_h) 'h = ' num2str(toc_curr/lgt_TD_h) 'sec/h' ' \n'])
toc_next = toc;

%% update Records
RunTime(i_file).datalgt_obs = lgt_TotalData;
RunTime(i_file).datalgt_h = lgt_TotalData/var_feq/60/60;
RunTime(i_file).datalgt_d = RunTime(i_file).datalgt_h/24;
```

```
if (Version_Dataset == 2)||(Version_Dataset == 1)||(Version_Dataset == 4)
    RunTime(i_file).pos_extractData = toc;
    RunTime(i_file).extractData = RunTime(i_file).pos_extractData - ...
                                  RunTime(i_file).str;
    PctReduc(i_file).datalgt = lgt_TotalData;
end

%% 2) Identify Dominant Axis
[z01_domAxis_3D,z01_domAxis,data_thldPct] = ...
    func_GDA_DASA(z_Base,z_Base_3D,data_dr,data_d, ...
    var_feq,lgt_TotalData,0,fn_save,[]);
% CreateIMG = 0/1;  % IMG name -> Image Name add-on;  % data_l = []/data_l;

% note time & effectivness
toc_curr = toc-toc_next;
if (Version_Dataset == 2)||(Version_Dataset == 1)||(Version_Dataset == 4)
    lgt1Dpre = lgt_TotalData;    lgt3Dpre = lgt_TotalData*3;
    lgt1Dpos = sum(z01_domAxis); lgt3Dpos = sum(sum(z01_domAxis_3D));
    lgt1Dpre_h = lgt1Dpre/var_feq/60/60;
    fprintf(['     DomAxis took ' num2str(toc_curr) 'sec for ' ....
    num2str(lgt1Dpre_h) 'h = ' num2str(toc_curr/lgt1Dpre_h) 'sec/h' ' \n'])
    RunTime(i_file).pos_domAxis = toc;
    RunTime(i_file).domAxis =  RunTime(i_file).pos_domAxis - ...
                               RunTime(i_file).pos_extractData;
    PctReduc(i_file).domAxis1D = (1-(lgt1Dpos/lgt1Dpre))*100;
    PctReduc(i_file).domAxis3D = (1-(lgt3Dpos/lgt3Dpre))*100;
else; fprintf(['     Dom Axes done ' num2str(toc_curr) ' sec ']); end
toc_next = toc;

%% 3) apply moving STD Threshold
[z02_mSTD,z02_mSTD_3D,output_stdthld] = ...
    func_GDA_mSTD(z01_domAxis,z01_domAxis_3D,data_dr,var_feq,...
    CreateIMG,fn_save,[],i_file,STDthld_standart);
% FigureON = 0/1; % IMG_name -> char(User) % data_Lab = []/data_Lab;
% UserID = i_file % STDthld standart -> variable

% note time & effectivness
toc_curr = toc-toc_next;
if (Version_Dataset == 2)||(Version_Dataset == 1)||(Version_Dataset == 4)
    lgt1Dpre = sum(z01_domAxis); lgt3Dpre = sum(sum(z01_domAxis_3D));
    lgt1Dpos = sum(z02_mSTD); lgt3Dpos = sum(sum(z02_mSTD_3D));
    lgt1Dpre_h = lgt1Dpre/var_feq/60/60;
    fprintf(['     mSTD took ' num2str(toc_curr) 'sec for ' ....
    num2str(lgt1Dpre_h) 'h = ' num2str(toc_curr/lgt1Dpre_h) 'sec/h' ' \n'])
    RunTime(i_file).pos_mSTD = toc;
    RunTime(i_file).mSTD = RunTime(i_file).pos_mSTD - ...
                           RunTime(i_file).pos_domAxis;
    PctReduc(i_file).mSTD1D = (1-(lgt1Dpos/lgt1Dpre))*100;
    PctReduc(i_file).mSTD3D = (1-(lgt3Dpos/lgt3Dpre))*100;
else; fprintf(['     mSTD done ' num2str(toc_curr) ' sec ']); end
if (Version_Dataset == 2) || (Version_Dataset == 3)
    clear z01_domAxis_3D z01_domAxis
end
toc_next = toc;

%% 4) Identify Episodes (start, finish)
% Note: need to check if the mmethode in func_GDA_Epi works
    % tested:               Epi         AC              Sum
    %   ohne peak check     0.02        4.62    sec     4.64
    %   origical(checked)   2.95        1.08    sec     4.03
    %   new (unchecked)     0.18        1.98    sec     2.16

[z03_Episodes_3D,z03_Episodes,epiList_loc_3D] = ...
    func_GDA_Epi(z02_mSTD_3D,data_dr,data_d, ...
    lgt_TotalData,slowWalk_obs, fastWalk_obs, minEpisode_steps);

% note time & effectivness
toc_curr = toc-toc_next;
if (Version_Dataset == 2)||(Version_Dataset == 1)||(Version_Dataset == 4)
    lgt1Dpre = sum(z02_mSTD); lgt3Dpre = sum(sum(z02_mSTD_3D));
```

```
    lgt1Dpos = sum(z03_Episodes); lgt3Dpos = sum(sum(z03_Episodes_3D));
    lgt1Dpre_h = lgt1Dpre/var_feq/60/60;
    fprintf(['     Idf Episodes took ' num2str(toc_curr) 'sec for ' ...
    num2str(lgt1Dpre_h) 'h = ' num2str(toc_curr/lgt1Dpre_h) 'sec/h' ' \n'])
    RunTime(i_file).pos_IdfEpisodes = toc;
    RunTime(i_file).IdfEpisodes = RunTime(i_file).pos_IdfEpisodes - ...
                                  RunTime(i_file).pos_mSTD;
    PctReduc(i_file).IdfEpisodes1D = (1-(lgt1Dpos/lgt1Dpre))*100;
    PctReduc(i_file).IdfEpisodes3D = (1-(lgt3Dpos/lgt3Dpre))*100;
else; fprintf(['     Idf Episodes done ' num2str(toc_curr) ' sec ']); end
if (Version_Dataset == 2) || (Version_Dataset == 3)
    clear z02_mSTD z02_mSTD_3D
end
toc_next = toc;

%% 5) Apply AC
% Note: Output
    % AC_epiList(:,1)   location of the START of the episode in data_d
    % AC_epiList(:,2)   location of the END of the episode in data_d
    % AC_epiList(:,3)   AXIS of the data of the epsiode in data_d
    % AC_epiList(:,4)   Episode LENGTH
    % AC_epiList(:,5)   AC freuqency
    % AC_epiList(:,6)   Quality (prominence of AC frequency)
    % AC_epiList(:,7)   Previous Episode idex
    % Quality_extend.
    %   epi             Reference to Previous Episode idex(AC_epiList(:,7))
    %   spk_pdom
    %   heigth
    %   prom
    %   lgt
    %   locadd
    %   diffhgtprm
    %   diffloc
    %   trendlinear_spk
    %   trendlinear_hgt
    %   trendlinear_prm
    %   GaitCycle
    %   GaitCycle_lst
    %   z Lab

[AC_epiList,z04_AC_3D,z04_AC,Quality_extend] = ...
func_GDA_AC(epiList_loc_3D,data_d,...
    lgt_TotalData, slowWalk_obs, fastWalk_obs,...
    data_Lab,CreateIMG,fn_save);
% data_Lab = data_Lab/[]; % figure_ON = 1/0; % IMG name ->  char(User)

% note time & effectivness
toc_curr = toc-toc_next;
if (Version_Dataset == 2)||(Version_Dataset == 1)||(Version_Dataset == 4)
    lgt1Dpre = sum(z03_Episodes); lgt3Dpre = sum(sum(z03_Episodes_3D));
    lgt1Dpos = sum(z04_AC); lgt3Dpos = sum(sum(z04_AC_3D));
    lgt1Dpre_h = lgt1Dpre/var_feq/60/60;
    fprintf(['     AC took ' num2str(toc_curr) 'sec for ' ...
    num2str(lgt1Dpre_h) 'h = ' num2str(toc_curr/lgt1Dpre_h) 'sec/h' ' \n'])
    RunTime(i_file).pos_applyAC = toc;
    RunTime(i_file).applyAC = RunTime(i_file).pos_applyAC - ...
                              RunTime(i_file).pos_IdfEpisodes;
    PctReduc(i_file).applyAC1D = (1-(lgt1Dpos/lgt1Dpre))*100;
    PctReduc(i_file).applyAC3D = (1-(lgt3Dpos/lgt3Dpre))*100;
else; fprintf(['     AC done ' num2str(toc_curr) ' sec  \n']); end
if (Version_Dataset == 2) || (Version_Dataset == 3)
    clear z03_Episodes_3D z03_Episodes epiList_loc_3D
end
toc_next = toc;

%%  Quality Check
if CreateIMG == 1; IMG_Quality(Quality_extend,fn_save,data_Lab); end

%% 6) Exract Gait data
% Biobank Only, in Koalab the data gets combined first
```

```
if (Version_Dataset == 3)||(Version_Dataset == 4)||(Version_Dataset == 1)
% Exclude data that is not potential walking behaviour
    % identify location of walking data
    idx_tue_walk = find(z04_AC == 1);
    % fill data with mesures of interest
    info_ACfreq = nan(3,lgt_TotalData);
    info_ACQual = nan(3,lgt_TotalData);
    ref_iepi  = nan(3,lgt_TotalData);
    for i_epi = 1:length(AC_epiList(:,1))
        info_ACfreq(AC_epiList(i_epi,3),...
        AC_epiList(i_epi,1):AC_epiList(i_epi,2)) = AC_epiList(i_epi,5);
        info_ACQual(AC_epiList(i_epi,3),...
            AC_epiList(i_epi,1):AC_epiList(i_epi,2)) = AC_epiList(i_epi,6);
        ref_iepi(AC_epiList(i_epi,3),...
            AC_epiList(i_epi,1):AC_epiList(i_epi,2)) = AC_epiList(i_epi,7);
    end
    % extract walking data
    walk_d = data_d(:,idx_tue_walk);
    walk_t = data_t(:,idx_tue_walk)-floor(data_t(1,1))+1;
    walk_g = data_g(:,idx_tue_walk);
    walk_info_ACfreq = info_ACfreq(:,idx_tue_walk);
    walk_info_ACQual = info_ACQual(:,idx_tue_walk);
    walk_ref_iepi = ref_iepi(:,idx_tue_walk);
    t_range = [data_t(1,1) data_t(1,end)];

    % note: code to reverse setting the start date to 1
    if 1 == 2; walk_t_org = walk_t+floor(t_range(1,1))-1; end
end

%% %%%%%%%%%%%%%%%%%% End of Gait Detection Functions %%%%%%%%%%%%%%%%%% %%

        %% Save Data
        if (Version_Dataset == 2)
            save(['Results_GD_' fn_save '.mat'],...
            'data_d','data_g','data_t','lgt_TotalData',...
            'data_thldPct','z04_AC_3D','z04_AC','AC_epiList',...
            'Quality_extend')
        end
        if(Version_Dataset==3)||(Version_Dataset==4)||(Version_Dataset==1)
            save(['Results_GD_' fn_save '.mat'],...
            'walk_d','walk_g','walk_t',...
            'lgt_TotalData','t_range',...
            'walk_info_ACfreq','walk_info_ACQual','walk_ref_iepi',...
            'Quality_extend','var_feq','output_stdthld');
        end
        if (Version_Dataset == 1) || (Version_Dataset == 4)
            save(['DataZ_' fn_save '.mat'],'data_Lab',...
                'data_d','data_dr','data_t','data_g','lgt_TotalData', ...
                'data_thldPct','output_stdthld',...
                'z01_domAxis_3D','z01_domAxis','z02_mSTD','z02_mSTD_3D',...
                'z03_Episodes_3D','z03_Episodes','z04_AC_3D','z04_AC',...
                'epiList_loc_3D','AC_epiList')
        end

        %% Housekeeping for the file
        % delete dataset of current file
        clear data_d data_g data_t lgt_TotalData data_dr
        clear data_thldPct output_stdthld
        clear AC_epiList z04_AC_3D z04_AC Quality_extend
        if (Version_Dataset == 2) || (Version_Dataset == 1)
        clear z01_domAxis_3D z01_domAxis z02_mSTD z02_mSTD_3D
        clear z03_Episodes_3D z03_Episodes epiList_loc_3D
        end
        if (Version_Dataset == 3) || (Version_Dataset == 4)
            clear idx_true_walk info_ACfreq info_ACQual ref_iepi
            clear tue_walk_sum idx_tue_walk
            clear walk_d walk_t walk_g walk_info_ACfreq walk_info_ACQual
            clear walk_ref_iepi t_range
        end
```

```
        % lose figures and files that are still open potentially
        close all;
        fclose ('all');

        %% Calculate Runtime & Update records
        if (Version_Dataset==2)||(Version_Dataset==1)||(Version_Dataset==4)
            RunTime(i_file).end = toc;
         RunTime(i_file).str_end = RunTime(i_file).end-RunTime(i_file).str;
            RunTime(i_file).savingData = RunTime(i_file).end - ...
                                         RunTime(i_file).pos_applyAC;
        end

        if (Version_Dataset == 3)
            RunTime(i_file).Fielsize_MB % note: I know it is File not Fiel
            % runtime
            time_fend = toc;
            RunTime(i_file).ProcessingTimeSec = time_fend-time_fstr;
            RunTime(i_file).PTSperH = RunTime(i_file).ProcessingTimeSec/...
                RunTime(i_file).datalgt_h;
           RunTime(i_file).PTSperMB = RunTime(i_file).ProcessingTimeSec/...
                RunTime(i_file).FielSize_MB;

            fprintf(['     total runtime file: ' ...
                num2str(RunTime(i_file).ProcessingTimeSec/60) ' min \n'])
            fprintf(['     processing time: ' ...
                num2str(RunTime(i_file).PTSperH) ' sec/h \n'])

            tleft_sec=sum(FS_MB(i_file:x_end,:))*RunTime(i_file).PTSperMB;
            tleft_h = tleft_sec/60/60; tleft_day = tleft_h/24;

            if tleft_day > 1; fprintf(['Predicted Remaining' ...
             'Time: ' num2str(tleft_day) ' days \n']); end
            if (tleft_h > 1) && (tleft_day <= 1); fprintf([ ...
             'Predicted Remaining Time: ' num2str(tleft_h) ' h \n']); end
            if tleft_h <= 1; fprintf(['Predicted Remaining Time: ' ...
             num2str(tleft_h/60) ' min \n']); end
        end
        if (Version_Dataset==2)||(Version_Dataset==1)||(Version_Dataset==4)
            fprintf(['Runtime folder: ' num2str(toc-time_fstr) ' sec \n'])
        end

        %% save Records
        save(['Results_Runtime' iteration_add_on '.mat'],'RunTime');
        if (Version_Dataset==2)||(Version_Dataset==1)||(Version_Dataset==4)
            save(['Results_PctRedc' iteration_add_on '.mat'],'PctReduc');
        end

        %% Delete file currently running (Biobank only)
        % to save space
        if Version_Dataset == 3
            delete([CHOICE_dir_basis  fn_load_pre fn_load_pos]);
        end

    end % for i_file = x_str:x_end

    %% save Records (for the whole run)
    if (Version_Dataset == 3) || (Version_Dataset == 4)
        RunTime(i_file).date_num = now;
        RunTime(i_file).date_str = datestr(now,'yyyy-mm-dd-HH:MM:SS.FFF');
        save(['Results_Runtime' iteration_add_on '.mat '], 'RunTime');
    end

end % for i_folderLoop = 1:n_folderLoop

%% House-Keeping
clear all;      % clear all variables from the current workspace
fclose('all');  % close all open files
close all;      % deletes all figures whose handles are not hidden
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_GDA_SelectData
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [filename_load_pre,filename_load_pos,filename_save_pre] = ...
    func_GDA_SelectData(Version_Dataset,CHOICE_dir_basis,filename_load)
%% FUNCTION INPUT
% Version_Dataset               1 = Selfrecorded Dataset
%                               2 = Koabab
%                               3 = BioBank
%                               4 = Selfrecorded Dataset in raw cwa format
%                               5 = .... (to be added if needed)
% CHOICE_dir_basis              Location of dataset to analyse
% filename_load                 Part of the filename that is the same for
%                               all datainstences of interest

%% FUNCTION OUTPUT
% filename_load_pre  List of all files (front half, most of the filename)
% filename_load_pos  List of all files (back h, mostly only ".bin" ending)
% filename_save_pre  Change to an alternative name when saving data

%% extract all files in folder of interest
    Folder_structure     = dir(CHOICE_dir_basis);
    char_FilesInFolder = char({Folder_structure(1:end).name}'); % reformat

%% remove files: wrong data format
if (Version_Dataset == 1)||(Version_Dataset == 2)||(Version_Dataset == 4)
    ToKeepIfWith = '.bin';
end
if (Version_Dataset == 3)
    ToKeepIfWith = '.cwa';
end
% identify location in list of files of interest
idx_FilesIntest_format = find(not(cellfun('isempty', ...
    (strfind(cellstr(char_FilesInFolder), ToKeepIfWith)))));
% create new list with files of interest
char_FilesInFolder = char(char_FilesInFolder(idx_FilesIntest_format,:));

%% remove files: wrong file name basis
ToKeepIfWith = filename_load;
idx_FilesIntest_fn = find(not(cellfun('isempty', ...
    (strfind(cellstr(char_FilesInFolder), ToKeepIfWith)))));
char_FilesInFolder = char(char_FilesInFolder(idx_FilesIntest_fn,:));

%% extra step remove files: select files only ones
    % find file types with time (since time should always be there)
% Note: In Biobank the data resides in only one file, in Koalap &
%    Self-recorded the data is devided by cathegory (accelerometer & time)
if (Version_Dataset==1)||(Version_Dataset==4); ToKeepIfWith = '_--t'; end
if Version_Dataset == 2; ToKeepIfWith = '---t'; end
if (Version_Dataset == 1)||(Version_Dataset == 2)||(Version_Dataset == 4)
    idx_FilesIntest_fn = find(not(cellfun('isempty', ...
        (strfind(cellstr(char_FilesInFolder), ToKeepIfWith)))));
   char_FilesInFolder_final=char(char_FilesInFolder(idx_FilesIntest_fn,:));
    % identify loaction of "ToKeepIfWith" and cut out what comes before and
    %   after into the new filename components
    Idx_DataTypeIdentifier = strfind(char_FilesInFolder_final(1,:),...
        ToKeepIfWith) :      strfind(char_FilesInFolder_final(1,:),...
        ToKeepIfWith)+length(ToKeepIfWith)-1;
end

if (Version_Dataset==3);char_FilesInFolder_final = char_FilesInFolder;end
% Note: If there are more conditions they can be added here in the manner
%       displayed above.

%% extract the Naming Convention of Interest
% very dependen on the dataset type
if (Version_Dataset == 1) || (Version_Dataset == 2)
    filename_load_pre = char_FilesInFolder_final(:,...
                                (1:Idx_DataTypeIdentifier(1,1)-1));
```

```
    filename_load_pos = char_FilesInFolder_final(:,...
                                (Idx_DataTypeIdentifier(end,end)+1):end);
end

if Version_Dataset == 3
    filename_load_pre = char_FilesInFolder_final(:,1:end-4);
    filename_load_pos = char_FilesInFolder_final(:,(end-3):end);
end
if Version_Dataset == 4
    filename_load_pre = char_FilesInFolder_final(:,1:21);
    filename_load_pos = char_FilesInFolder_final(:,26:end);
end

%% adjust changes in filename for future Versions:
% not used so far in the codes
if (Version_Dataset == 1) || (Version_Dataset == 4)
    filename_save_pre = [];
    for i_n = 1:length(filename_load_pre(:,1))
        filename_save_pre = [filename_save_pre; ...
            [filename_load_pre(i_n,:) '_V' num2str(i_n)]];
    end
end
% for KOALAP: remove 'T-accl'
if Version_Dataset == 2
    ToRemove = '_T-accl';
    remove_spaces = strfind(filename_load_pre(1,:),ToRemove) : ...
        strfind(filename_load_pre(1,:),ToRemove)+length(ToRemove)-1;
    filename_save_pre = filename_load_pre(:,1:remove_spaces(1,1)-1);
    % add part number 'U42_P029'
    filename_save_pre = [filename_save_pre(:,7)  ...
        filename_save_pre(:,9:10) filename_load_pos(:,1:5)];
end
% for Biobank
if Version_Dataset == 3
    ToRemove = filename_load;
    remove_spaces = strfind(filename_load_pre(1,:),ToRemove) : ...
        strfind(filename_load_pre(1,:),ToRemove)+length(ToRemove)-1;
    filename_save_pre = filename_load_pre(:,1:remove_spaces(1,1)-1);
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_GDA_LoadData
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [data_d,data_g,data_t,lgt_TotalData,data_dr,data_Lab,data_l,...
    data_raw] = func_GDA_LoadData(fn_load_pre, fn_load_pos,...
    CHOICE_dir_basis,Version_Dataset,var_feq)
%% FUNCTION INPUT
% fn_load_pre           List of all files (front half, most of the
%                       filename)
% fn_load_pos           List of all files (back half, mostly only ".bin"
%                       ending)
% Version_Dataset       1 = Selfrecorded Dataset
%                       2 = Koabab
%                       3 = BioBank

%% FUNCTION OUTPUT
% data_d            dynamic data (d = ai - g) (ai = raw accelerometer data)
% data_g            gavitational data
% data_t            time data
% lgt_TotalData     length of the dataset
% data_dr           magnitude of the data (dr = sqrt(sum((data_d').^2,2))')
% data_Lab          Handlabels (self-recorded dataset only)
% data_l            see above, just more detailed

%% load data and preprocessing depending on format of the dataset
%% Version_Dataset 1 - Selfrecorded Dataset
% preprocessing was already done
% additionan Labeled data can be loaded
if (Version_Dataset == 1) || (Version_Dataset == 4)

    fn = [CHOICE_dir_basis fn_load_pre '_--g' fn_load_pos]; % dynamic
    fid_load = fopen(fn,'rb','ieee-be');
    data_g = fread(fid_load, [3 inf], 'float32');

    fn = [CHOICE_dir_basis fn_load_pre '_--t' fn_load_pos]; % time
    fid_load = fopen(fn,'rb','ieee-be');
    data_t = fread(fid_load, [1 inf], 'float64');

    fn = [CHOICE_dir_basis fn_load_pre '_--d' fn_load_pos]; % dynamic
    fid_load = fopen(fn,'rb','ieee-be');
    data_d = fread(fid_load, [3 inf], 'float32');

    fn = [CHOICE_dir_basis fn_load_pre '_Lab' fn_load_pos]; % Lables
        % 0 - Non-Lablelled data (time between trials & triggers)
        % 1 - Resting data (Coffee Trail, non-walking)
        % 2 - Walking data (corridor/stair walking trial, free walking)
    fid_load = fopen(fn,'rb','ieee-be');
    data_Lab = fread(fid_load, [1 inf], 'float32');

    fn = [CHOICE_dir_basis fn_load_pre '_--l' fn_load_pos]; % Lables
    % more detailed
    fid_load = fopen(fn,'rb','ieee-be');
    data_l = fread(fid_load, [1 inf], 'float32');
    data_raw = [];
end

%% Version Dataset 2 - Koalab Dataset
% preprocessing was already done
if Version_Dataset == 2
    fn = [CHOICE_dir_basis fn_load_pre '---g' fn_load_pos]; % dynamic
    fid_load = fopen(fn,'rb','ieee-be');
    data_g = fread(fid_load, [3 inf], 'float32');

    fn = [CHOICE_dir_basis fn_load_pre '---t' fn_load_pos]; % time
    fid_load = fopen(fn,'rb','ieee-be');
    data_t = fread(fid_load, [1 inf], 'float64');

    fn = [CHOICE_dir_basis fn_load_pre '--ai' fn_load_pos]; % accelerometer
    fid_load = fopen(fn,'rb','ieee-be');
```

```
    data_ai = fread(fid_load, [3 inf], 'float32');

    data_d = data_ai - data_g; % dynamic

    data_Lab = [];
    data_l = [];
    data_raw = [];
end

%% Version Dataset 3 - Biobank data
% no preprocessing was done
% different gravity removal from the dataset
if Version_Dataset == 3
    % load data
    data_raw = AX3_readFile([CHOICE_dir_basis fn_load_pre fn_load_pos]);
    fclose('all'); fprintf('Loding done \n')
    data_prep_ai = data_raw.ACC(:,2:4)';
    data_prep_t = data_raw.ACC(:,1)';

    % remove "Non-Unique" values
    [~,ind_nonqnique] = unique(data_prep_t);
    data_prep_t = data_prep_t(:,ind_nonqnique);
    data_prep_ai = data_prep_ai(:,ind_nonqnique);

    % catch and remove to many days
    %   NOTE: file '1228329_90001_0_0.cwa'
    %   started on '2013-01-18 03:37:54.000' and had a single recorded
    %   datapoint till '2015-01-12 10:00:04.000' every few days
    data_t_str = data_prep_t(1,1);
    data_prep_ti = data_prep_t - data_t_str;
    t_days = unique(floor(data_prep_ti));
    idx_days = [];
    for i_days = 1:length(t_days)
        i_idx_days = find(floor(data_prep_ti) == t_days(i_days));
        if length(i_idx_days) < var_feq*60*5 % min of 5 min
            idx_days = [idx_days i_idx_days];
        end
    end
    data_prep_ti(:,idx_days) = [];
    data_prep_ai(:,idx_days) = [];
    % reset point of 0
    if ~isempty(idx_days)
        data_t_str_new = data_prep_ti(1,1)+data_t_str;
        data_prep_ti = data_prep_ti + data_t_str - data_t_str_new;
        data_t_str = data_t_str_new;
    end

    % Preprocessing: Interpolation
    %   plot msec diff between data_prep_t
    %   histogram(diff(data_prep_t)/((1/24/60/60/1000)))
    %       day h  m  s  Fequency
    interpolationON = 1;
    if interpolationON == 1
        srate = (1/24/60/60)/var_feq;
        data_ti = data_prep_ti(1,1):srate:data_prep_ti(end,end);
        data_ai = [];
        for i_axis = 1:3
            data_ai(i_axis,:) = interp1(data_prep_ti,...
                                data_prep_ai(i_axis,:),data_ti,'pchip');
        end
        data_t = data_ti+data_t_str;
    else
        fprintf(['No Interpolation \n'])
        data_t = data_prep_t;
        data_ai = data_prep_ai;
    end

    % identify datalength
    lgt_TotalData = length(data_t(1,:));
```

```
    % Preprocessing: Remove Gravity
    data_g = nan(3,lgt_TotalData);
    n = 2;
    lowPassF = 0.50;
    [b,a] = butter(n,lowPassF/(var_feq*0.5),'low');
    for i_axis = 1:3
        % use filtfilt instead of filter to forward and backward filter
        data_g(i_axis,:) = filtfilt(b,a,data_ai(i_axis,:));
    end

    % clean up
    fprintf('gravity extraction done \n')
    data_l = [];
    data_Lab = [];
end

%% Version Dataset 4 - Self Recorded Dataset - change Preprocessing
if Version_Dataset == 4
    data_ai = data_d + data_g;
    clear data_g
    data_g = nan(3,length(data_ai(1,:)));
    n = 2; lowPassF = 0.50;
    [b,a] = butter(n,lowPassF/(var_feq*0.5),'low');
    for i_axis = 1:3
        % use filtfilt instead of filter to forward and backward filter
        data_g(i_axis,:) = filtfilt(b,a,data_ai(i_axis,:));
    end
end

%% Housekeeping
fclose('all');

%% cut/transform data for analysis (if needed)
% Magnitude of the data
if (Version_Dataset == 3) || (Version_Dataset == 4)
    data_d = data_ai - data_g;
end
data_dr = sqrt(sum((data_d').^2,2))';

%% data properties
lgt_TotalData = length(data_d);

end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_GDA_DASA
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [z_domAxis_3D,z_domAxis,data_thldPct] = func_GDA_DASA(...
    z_Base,z_Base_3D,data_dr,data_d, var_feq,lgt_TotalData,...
    FigureON,IMG_name,data_l,UserID)
%% Purpose
% find locations where an axis moving std is larger then the magnitude of
%   the whole data's moving STD

%% FUNCTION INPUT
% z_Base        vector indicating already excluded data
% data_dr       magnitude of dynamic data:
%               data_dr = sqrt(sum((data_d').^2,2))'
% data_d        dynamic data (d = ai - g)
% var_feq       Sample frequency of the data
% lgt_TotalData Length of the whole dataset

%% FUNCTION OUTPUT
% z_domAxis_3D      labeling as potential gait (1) or non-gait (0) for all
%                   three axis
% z_domAxis         labeling as potential gait (1) or non-gait (0) if one
%                   of the three axis are potential gait
% data_thldPct      percentage difference were the threshold was meet

%% Codes needed from Supplementary
% movingstd         (https://uk.mathworks.com/matlabcentral/fileexchange/
%                   9428-movingstd-movingstd2)

if 1 == 2
    %% Identigy Windowed STD
    % Moving windowed STD (MwSTD)
    data_MwSTD_dr  = movingstd(data_dr,var_feq);
    data_MwSTD_ax1 = movingstd(data_d(1,:),var_feq);
    data_MwSTD_ax2 = movingstd(data_d(2,:),var_feq);
    data_MwSTD_ax3 = movingstd(data_d(3,:),var_feq);

    %% Smooth data
    % Moving windowed Mean (MwMean or f (filter) of the MwSTD)
    windowsize_sec_vector = (1/var_feq)*ones(1,var_feq);
    data_fMwSTD_dr  = filter(windowsize_sec_vector,1,data_MwSTD_dr);
    data_fMwSTD_ax1 = filter(windowsize_sec_vector,1,data_MwSTD_ax1);
    data_fMwSTD_ax2 = filter(windowsize_sec_vector,1,data_MwSTD_ax2);
    data_fMwSTD_ax3 = filter(windowsize_sec_vector,1,data_MwSTD_ax3);

    %% Identify "percentage difference"
    % Percentage Difference (PctDif) of each of the d MwSTD to the dr MwSTD
    data_PctDiffMwSTD_ax1=(data_fMwSTD_ax1-data_fMwSTD_dr)./data_fMwSTD_dr;
    data_PctDiffMwSTD_ax2=(data_fMwSTD_ax2-data_fMwSTD_dr)./data_fMwSTD_dr;
    data_PctDiffMwSTD_ax3=(data_fMwSTD_ax3-data_fMwSTD_dr)./data_fMwSTD_dr;

else
    %% combine steps above (reduce memeory need)
    windowsize_sec_vector = (1/var_feq)*ones(1,var_feq);
    data_fMwSTD_dr   = filter(windowsize_sec_vector,1,...
        movingstd(data_dr,var_feq));
    data_PctDiffMwSTD_ax1 = ((filter(windowsize_sec_vector,1,...
        movingstd(data_d(1,:),var_feq))) - data_fMwSTD_dr)./data_fMwSTD_dr;
    data_PctDiffMwSTD_ax2 = ((filter(windowsize_sec_vector,1,...
        movingstd(data_d(2,:),var_feq))) - data_fMwSTD_dr)./data_fMwSTD_dr;
    data_PctDiffMwSTD_ax3 = ((filter(windowsize_sec_vector,1,...
        movingstd(data_d(3,:),var_feq))) - data_fMwSTD_dr)./data_fMwSTD_dr;
end

%% Determine Percentage Difference Threshold
% identify which axis fMwSTD has the largest distance to the dr fMwSTD
PctDif_max = max([data_PctDiffMwSTD_ax1;data_PctDiffMwSTD_ax2;...
    data_PctDiffMwSTD_ax3]);
```

```
% threshold: the axis is closer to the maximum fMwSTD then to the dr fMwSRD
PctDif_thld = PctDif_max/2;

%% Identify data exceeding Threshold
% Identify axis exceeding the threshold
%   record in the format of a Logical Operations vector (attention: not
%   actually coded as an Logical Operations)
z_domAxis_3D = zeros(3,lgt_TotalData);
z_domAxis_3D(1,data_PctDiffMwSTD_ax1 > PctDif_thld) = 1;
z_domAxis_3D(2,data_PctDiffMwSTD_ax2 > PctDif_thld) = 1;
z_domAxis_3D(3,data_PctDiffMwSTD_ax3 > PctDif_thld) = 1;
z_domAxis_3D(:,sum(z_domAxis_3D) == 0) = 1;

%% Record Results
z_domAxis = sum(z_domAxis_3D);
z_domAxis(:,z_domAxis > 1) = 1;

%% Apply Basline z
if ~isempty(z_Base)
    z_domAxis(:,z_Base == 0) = 0;
end
if ~isempty(z_Base)
    z_domAxis_3D(1,z_Base_3D(1,:) == 0) = 0;
    z_domAxis_3D(2,z_Base_3D(2,:) == 0) = 0;
    z_domAxis_3D(3,z_Base_3D(3,:) == 0) = 0;
end

%% extra: for plot purposes
%   keep a record of the percDiff when the threshold is exceeded
data_thldPct = [data_PctDiffMwSTD_ax1;...
    data_PctDiffMwSTD_ax2;data_PctDiffMwSTD_ax3];
data_thldPct(1,data_PctDiffMwSTD_ax1 < PctDif_thld) = 0;
data_thldPct(2,data_PctDiffMwSTD_ax2 < PctDif_thld) = 0;
data_thldPct(3,data_PctDiffMwSTD_ax3 < PctDif_thld) = 0;

%% Figures 001
if FigureON == 1
%% figure: dispay 10 largest episodes of no interupion OR Sort by Labels
    % determine Plot Window
    plot_datalgt = 1*50*60; % 2min
    n_plot = 100;
    %% identify episodes according to length
    matrix_max = zeros(3,length(data_thldPct));
        matrix_max(1,data_thldPct(1,:) == max(data_thldPct)) = 1;
        matrix_max(2,data_thldPct(2,:) == max(data_thldPct)) = 1;
        matrix_max(3,data_thldPct(3,:) == max(data_thldPct)) = 1;
    mtx_max_cgp = sum(abs(diff(matrix_max')'));
    % identify location of change points
    cgp_idx = [1 find(mtx_max_cgp ~= 0) length(mtx_max_cgp)];
    cgp_lgt = diff(cgp_idx); % identify length of change points
    cgp_sort = sort(cgp_lgt,'descend');
    [~,hit_idx_pre] = ismember(cgp_lgt,cgp_sort(1,1:n_plot));
    hit_idx = find(hit_idx_pre > 0);
    list_plot = [cgp_idx(hit_idx)' cgp_idx(hit_idx+1)'];
    % determine ranges for Labeled data
    if ~isempty(data_l); LabelOn = 2; else; LabelOn = 1; end
    % run plots
    for ix = 1:LabelOn
        % select between episodes by length (ix = 1) or label (ix = 2)
        if ix == 1; nil = length(list_plot(:,1)); end
        if ix == 2; nil = length(unique(data_l)); end
        for il = 1:nil
            % select data
            if ix == 1; i_idx = list_plot(il,1):list_plot(il,2); end
            if ix == 2; i_idx = find(data_l == il); end
            % plot figure
            figure('units','normalized','outerposition',[0 0 1 1],...
                'visible','off'); hold on;
            subplot(3,4,1:3); hold on;
                plot(data_d(:,i_idx)')
```

```
            xlim([floor(length(i_idx)/2)-floor(plot_datalgt/2)  ...
                floor(length(i_idx)/2)+floor(plot_datalgt/2)]);
            title(['Data of Behaviour: l = ' num2str(il)])
        subplot(3,4,5:7); hold on;
            plot(data_MwSTD_ax1(:,i_idx),'Color',[0, 0.4470, 0.7410])
           plot(data_MwSTD_ax2(:,i_idx),'Color',[0.8500,0.3250,0.0980])
           plot(data_MwSTD_ax3(:,i_idx),'Color',[0.9290,0.6940,0.1250])
            plot(data_MwSTD_dr(:,i_idx),'k')
            xlim([floor(length(i_idx)/2)-floor(plot_datalgt/2)  ...
                floor(length(i_idx)/2)+floor(plot_datalgt/2)]);
            title('Moving STD')
        subplot(3,4,9:11); hold on;
            plot(data_thldPct(:,i_idx)')
            plot(z_domAxis(i_idx),'k')
            xlim([floor(length(i_idx)/2)-floor(plot_datalgt/2) ...
                floor(length(i_idx)/2)+floor(plot_datalgt/2)]);
           ylim([0 max(mean(data_thldPct'))+max(std(data_thldPct'))*4])
            title('Percentage Difference')
        subplot(3,4,[4 8 12]); hold on;
            histogram(data_MwSTD_dr(:,i_idx),50,'Normalization',...
                'probability','EdgeColor','none','FaceColor','k')
            histogram(data_MwSTD_ax1(:,i_idx),50,'Normalization',...
                'probability','EdgeColor','none','FaceColor',...
                [0, 0.4470, 0.7410])
            histogram(data_MwSTD_ax2(:,i_idx),50,'Normalization',...
                'probability','EdgeColor','none','FaceColor',...
                [0.8500, 0.3250, 0.0980])
            histogram(data_MwSTD_ax3(:,i_idx),50,'Normalization',...
                'probability','EdgeColor','none','FaceColor',...
                [0.9290, 0.6940, 0.1250])
            title('Histogram (Moving STD)')
        % save data
        if ix == 1; saveas(gcf,[ 'IMG_' IMG_name '_domAx2_Bhv_' ...
                num2str(il,'%05d\n') '.tif']); end
        if ix == 2; saveas(gcf,[ 'IMG_' IMG_name '_domAx2_Lab_' ...
                num2str(il,'%05d\n') '.tif']); end
        close all
end;end;end % if FigureON == 1
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_GDA_mSTD
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [z_mSTD,z_mSTD_3D,output_stdthld] = func_GDA_mSTD(...
    z_Base,z_Base_3D,data_dr,var_feq,...
    FigureON,IMG_name,data_Lab,UserID,STDthld_standart)

%% Purpose
% to remove sections such as sleeping or rest

%% FUNCTION INPUT
% data_dr       magnitude of the data: dr = sqrt(sum((data_d').^2,2))'
% var_feq       Frequency at which the data was recorded

%% FUNCTION OUTPUT
% z_mSTD        vector labeling data as potential gait (1) or rest (0)

%% Codes needed from Supplementary
% movingstd         (https://uk.mathworks.com/matlabcentral/fileexchange/
%                       9428-movingstd-movingstd2)

%% Moving STD
% identify moving STD
Y_STDfilter = movingstd(data_dr,var_feq);

% identify histogram of filtered data
rslt_hgf_01 = histcounts(Y_STDfilter,'BinWidth',0.01);

i_z_1D = z_Base;
i_thld_pass = 0;
figure_data = [];
while_condition = 1;
xxx = 1;
while (while_condition > 0) && (while_condition < 7)
    if FigureON == 1; figure_data(xxx).data_dr = data_dr(:,i_z_1D == 1);end

    %% identify histogram
    i_hg_dr = histcounts(data_dr(:,i_z_1D == 1),'BinWidth',0.01);
    % make sure that the peack is not onwith in first bin and can therefore
    % not be found.
    i_hg_dr(1) = 0;
    % identify gradient

    %% check if theshold should still be applied
    % histogram need to have a remarkable peak
    [ii_amp,ii_loc,ii_wid,ii_prm] = findpeaks(i_hg_dr,'MinPeakHeight',...
        max(i_hg_dr)/2,'MinPeakDistance',STDthld_standart*100);
        % use of Min Heigth to prevent
    % select maximumpeak
    i_fpidx =  find(ii_amp == max(ii_amp));
    i_amp = ii_amp(i_fpidx); i_loc = ii_loc(i_fpidx);
    i_wid = ii_wid(i_fpidx); i_prm = ii_prm(i_fpidx);

    %% identify threshold
    % use: range of wid
    i_thld = i_loc *0.01;
    i_thld_pass = i_loc*2 *0.01;

    %% check threshold
    if (i_thld_pass < STDthld_standart)
        %% record new data to be used (z in 1D)
        i_z_1D(Y_STDfilter < i_thld) = 0;
        output_stdthld = i_thld;
        %% finish While set up
        while_condition = while_condition + 1;
    else
        while_condition = -1;
    end
```

```
    %% interlude
    % record data for plots
    if FigureON == 1
        figure_data(xxx).histogram = i_hg_dr;
        figure_data(xxx).findpeaks = [i_amp;i_loc;i_wid;i_prm];
        figure_data(xxx).findpeaks_all = [ii_amp;ii_loc;ii_wid;ii_prm];
        figure_data(xxx).threshold = i_thld_pass;
        figure_data(xxx).z_new = i_z_1D;
        figure_data(xxx).data_zf = data_dr(:,i_z_1D == 1);
        figure_data(xxx).data_z = data_dr(:,i_z_1D == 1);
        xxx = xxx+1;
    end
end

%% select final 1D
z_mSTD = i_z_1D;

%% describe 3D
z_mSTD_3D = z_Base_3D; z_mSTD_3D(:,i_z_1D == 0) = 0;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_GDA_Epi
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [z_Episodes_3D,z_Episodes,epiList_loc] = func_GDA_Epi(...
    z_Base_3D,data_dr,data_d,lgt_TotalData, ...
    slowWalk_obs, fastWalk_obs, minEpisode_steps)
%% FUNCTION INPUT

%% FUNCTION OUTPUT
% epiList_loc(:,1:2)     str and end loc
% epiList_loc(:,3)       Axis
% epiList_loc(:,4)       length

% Variables
var_minEpi_lgt_slowWalk = round(slowWalk_obs*minEpisode_steps);
var_minEpi_lgt_fastWalk = round(fastWalk_obs*minEpisode_steps);

var_MinPeakDistance = fastWalk_obs;
var_MaxPeakDistance = slowWalk_obs;

%% Identify Change Points
epiList_loc_01 = [];
for i_axis = 1:3
    % find location of changes for each axis
    loc_changePt = find(diff(z_Base_3D(i_axis,:)) ~= 0);
    % identify start or end points of episodes
    loc_cPt_potentialStr = [1,loc_changePt+1];
    loc_cPt_potentialEnd = [loc_changePt,lgt_TotalData];
    % remove episodes that are not potential walking episodes
    loc_str_cPt = loc_cPt_potentialStr(:,...
        z_Base_3D(i_axis,loc_cPt_potentialStr) == 1);
    loc_end_cPt = loc_cPt_potentialEnd(:,...
        z_Base_3D(i_axis,loc_cPt_potentialEnd) == 1);
    % transform results
    loc_cPt_pre = [loc_str_cPt',loc_end_cPt',...
        zeros(length(loc_str_cPt),1),loc_end_cPt'-loc_str_cPt'];
    loc_cPt_pre(:,3) = i_axis;
    % add to results
    epiList_loc_01 = [epiList_loc_01;loc_cPt_pre];
end

%% Exclusion: Minimal Episode length
% if the episode is not long enough to fit "minEpisode_steps" fast steps,
%   exclude the episode
epiList_loc_02_pre = epiList_loc_01(...
    epiList_loc_01(:,4) >= var_minEpi_lgt_fastWalk,:);


%% Exclusion: Minimal Peaks in short Episodes
% if the episode is smaller then "minEpisode_steps" slow steps check if
%   it has enough peaks to show at least 5 steps
% not needed in longer episodes

% Citation:
% Peter O'Connor (2021). PeakSeek (https://www.mathworks.com/
%   matlabcentral/fileexchange/26581-peakseek), MATLAB Central File
%   Exchange. Retrieved April 9, 2021.
excludeNoPeaks_loc = zeros(1,length(epiList_loc_02_pre(:,1)));
min_peaks = minEpisode_steps*2-1;
for i_epi = 1:length(epiList_loc_02_pre(:,1))
    if epiList_loc_02_pre(i_epi,4) <= var_minEpi_lgt_slowWalk
        Y_d = data_d(epiList_loc_02_pre(i_epi,3),...
            epiList_loc_02_pre(i_epi,1):epiList_loc_02_pre(i_epi,2));
        thld_amp_hill = mean(Y_d(:,Y_d>0));
        thld_amp_vall = mean(Y_d(:,Y_d<0));
        [i_n_pks_pos, ~] = peakseek(Y_d,var_MinPeakDistance,thld_amp_hill);
        [i_n_pks_neg, ~] = peakseek(Y_d,var_MinPeakDistance,thld_amp_vall);
        if (length([i_n_pks_pos,i_n_pks_neg]) < min_peaks) || ...
                (length(i_n_pks_pos) < minEpisode_steps) || ...
```

```
                (length(i_n_pks_neg) < minEpisode_steps)
            excludeNoPeaks_loc(1,i_epi) = i_epi;
end;end;end
excludeNoPeaks_loc(:,excludeNoPeaks_loc == 0) = [];
epiList_loc_03 = epiList_loc_02_pre;
epiList_loc_03(excludeNoPeaks_loc,:) = [];

%% Reform Results into z
epiList_loc = epiList_loc_03;

z_Episodes_3D = zeros(3,lgt_TotalData);
for i_res = 1:length(epiList_loc(:,1))
    z_Episodes_3D(epiList_loc(i_res,3),...
        epiList_loc(i_res,1):epiList_loc(i_res,2)) = 1;
end

z_Episodes = sum(z_Episodes_3D);
z_Episodes(:,z_Episodes > 1) = 1;

%% figures;
Figure_On = 0; % need to run mSTD and episode manually first
if Figure_On == 1
    plot_mSTD = z_mSTD_3D; plot_Episodes = z_Episodes_3D;
    plot_d_std = data_d; plot_d_epi = data_d;
    for iplot = 1:3
        plot_mSTD(iplot,z_mSTD_3D(iplot,:) == 0) = nan;
        plot_Episodes(iplot,z_Episodes_3D(iplot,:) == 0) = nan;
        plot_d_std(iplot,z_mSTD_3D(iplot,:) == 0) = nan;
        plot_d_std(iplot,z_Episodes_3D(iplot,:) == 1) = nan;
        plot_d_epi(iplot,z_Episodes_3D(iplot,:) == 0) = nan;
    end

    figure; hold on;
    plot(plot_d_epi')
    plot(plot_d_std','Color',[0.5 0.5 0.5])
    xlim([70000 80000])
    ylim([-8 8])
    title('Comaprison between z-mSTD-3D and z-Episodes-3D')
    ylabel('acceleration (g)'); xlabel('Observations (obs)')
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_GDA_AC
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [AC_epiList,z_AC_3D,z_AC,Quality_extend] = func_GDA_AC(...
    epiList_loc,data_d,lgt_TotalData, slowWalk_obs, fastWalk_obs,...
    data_Lab,FigureON,IMG_name)
%% Purpose
% to remove sections such as sleeping or rest

%% FUNCTION INPUT
% data_dr       magnitude of the data: dr = sqrt(sum((data_d').^2,2))'
% var_feq       Frequency at which the data was recorded
% epiList_loc   Information about episodes from func_GDA_episodes
%   epiList_loc(:,1)    location of the START of the episode in data_d
%   epiList_loc(:,2)    location of the END of the episode in data_d
%   epiList_loc(:,3)    AXIS of the data of the epsiode in data_d
%   epiList_loc(:,4)    Episode LENGTH
%   epiList_loc(:,5)    Place Holder for identifying episodes labeld as
%                       walking (1) or non walking (0) via
%                       "Notes_epiList_loc_zu_data_Lab.m" (can leave empty)
% data_d            dynamic accelerometer data
% lgt_TotalData     length of the whole dataset
% slowWalk_obs      max observations/sampes need for 1 slow step
% fastWalk_obs      min obs/samples needed for 1 fast step

%% FUNCTION OUTPUT
% AC_epiList(:,1)    location of the START of the episode in data_d
% AC_epiList(:,2)    location of the END of the episode in data_d
% AC_epiList(:,3)    AXIS of the data of the epsiode in data_d
% AC_epiList(:,4)    Episode LENGTH
% AC_epiList(:,5)    AC freuqency
% AC_epiList(:,6)    Quality (prominence of AC frequency)
% AC_epiList(:,7)    Previous Episode idex
% z_AC_3D       vector labeling gait (1) & non-gait (0) for 3 axis
% z_AC          vector labeling gait (1) & non-gait (0) overall

%% Notes
% if data_Lab is not present: replace with data_Lab = [];
% if figure shall be computet: figure_ON = 1; else figure_ON = 0;

%% set Variables
stepfactor = 3; % at minimum 2
% identify center of the autocorrelation plot
plotcenter = ceil(slowWalk_obs*stepfactor)+1;

%% prep data output
epiList_z = zeros(1,length(epiList_loc(:,1)));
epiList_AC = nan(1,length(epiList_loc(:,1)));
epiList_Q = nan(1,length(epiList_loc(:,1)));
z_AC_3D = zeros(3,lgt_TotalData);
ExclusionCriteria = [0 0 0 0 0 0 0 0 nan 0];
plot_Lab_z = [];
Quality_extend = []; Quality_extend(length(epiList_loc(:,1))).i_epi = [];

%% run loops
for i_epi = 1:length(epiList_loc(:,1))
    %% prep run
    clear bwpks_list spk_pdom loc_p0 loc_pdom sc_dpks_hgt_pct

i_z = nan; plot_title = '0 - str'; % image

    %% identify data
    Y_d = data_d(epiList_loc(i_epi,3),...
        epiList_loc(i_epi,1):epiList_loc(i_epi,2));

    %% AC simple
    % run a cross correlation
    xc = xcorr(Y_d(1,:),ceil(slowWalk_obs*stepfactor),'coeff');
```

```
    %% Identify peaks
    % select range of interest to be analysed
    xc_half = xc(:,plotcenter:end);
    % identify were where the plot first crosses 0
    loc_xc_first0 = find(xc_half <=0); loc_xc_first0 = loc_xc_first0(1,1);
    % identify were the plot "fizzels out"
    loc_xc_fizend=find(xc_half>mean(xc));loc_xc_fizend=loc_xc_fizend(end);

    % largest peak needs to occure after a valley (the graph needs to
    %   hit 0 before it can hit a peak (else flucturations in the first
    %   part of the graph might be missinterpret as a peak)
    if ~isempty(loc_xc_first0)
        xc_half(:,1:loc_xc_first0) = nan(1,1);
        xc_half(:,loc_xc_fizend:end) = nan(1,1);

    %% apply findpeaks to identify every peak
        [amp_p0,loc_p0,wid_p0,prm_p0] = findpeaks(xc_half);
    else
        amp_p0 = [];
i_z = 0;
plot_title = '1 - exclusion: xc does not reach 0';% image
ExclusionCriteria(1,1) = ExclusionCriteria(1,1)+1;
    end

%% identify dominant peaks
    %% find area of potential domiant peaks
    if isnan(i_z)
        % end of xc_half2 when it fizzels out
        xc_half2 = xc(:,plotcenter:loc_xc_fizend+plotcenter-1);
        % identify where xc_half corsses the 0
        pointsofcontent = unique([1 find(diff(xc_half2 <= 0)) ...
            find(diff(xc_half2 == 0)) find(diff(xc_half2 >= 0)) ...
            length(xc_half2)]);
        pot_pdom_list = [];
        for i_xxx = 1:length(pointsofcontent)-1
            % the first one needs to be the entering of 0
            i_xxx_Y = xc_half2(1,pointsofcontent(i_xxx)+1:...
                               pointsofcontent(i_xxx+1)-1);
            % can not be less then 1/4 of a fastWalk_obs (e.g. i_epi = 46)
            if length(i_xxx_Y) > fastWalk_obs/4
                % if there is negative space discard change
                if ~any(i_xxx_Y < 0)
                    % do not record if it is the first section
                    if pointsofcontent(i_xxx) >= loc_xc_first0
                        % create a list of points of content
                        pot_pdom_list = [pot_pdom_list, ...
                            [pointsofcontent(i_xxx); ...
                            pointsofcontent(i_xxx+1)]];
        end;end;end;end % for i_xxx

% exclusion criteria: at least 2 potential peaks areas need to be found
        pot_pdom_sz = size(pot_pdom_list);
        if pot_pdom_sz(1,2) < 2
i_z = 0;
plot_title = '2a - exclusion: pot_pdom_list < two dominant peaks';
ExclusionCriteria(1,2) = ExclusionCriteria(1,2)+1;
        end
    end % isnan(i_z) % find area of potential domiant peaks

    %% find dominant peaks
    if isnan(i_z)
    % loop the areas identified by pot_pdom_list
        loc_pdom = [];
        for i_fm = 1:length(pot_pdom_list)
            xc_half3 = nan(1,length(xc_half));
            xc_half3(1,pot_pdom_list(1,i_fm):pot_pdom_list(2,i_fm)) =  ...
                xc_half(1,pot_pdom_list(1,i_fm):pot_pdom_list(2,i_fm));
            % identify the location of the maxium peaks in each area
            loc_pdom = [loc_pdom, find( xc_half3 == max(xc_half3))];
        end
```

```
        % potential problem: loc_pdom migth identify different peaks then
        %   loc_p1 -> remove the peaks of loc_pdom that are not in loc_p1
        loc_pdom(~ismember(loc_pdom,loc_p0)) = [];

 % exclusion criteria: at least 2 peaks need to be found
        if length(loc_pdom) < 2
i_z = 0;
plot_title = '2b - exclusion: loc_pdom < two dominant peaks'; % image
ExclusionCriteria(1,3) = ExclusionCriteria(1,3)+1;
        end
    end % isnan(i_z) % find domiant peaks

    %% compair dominant (loc_pdom) and identified peaks (loc_p0)
    if isnan(i_z)
        % map them onto findpeaks (pks_p0) to identify stats
        spk_pdom = [amp_p0(ismember(loc_p0,loc_pdom));... % amplitude (amp)
            loc_pdom;... % location on xc_half (loc)
            wid_p0(ismember(loc_p0,loc_pdom));... % width (wid)
            prm_p0(ismember(loc_p0,loc_pdom))]; % prominence (prm)

        %% The peaks can fizzle out towards the end
        % take the max amp and make the last peak that reaches that
        %   threshold the last peak
if length(spk_pdom(1,:)) > 3 % don't do this if there are less then 3 peaks
    thld_spkdomlgt = find(spk_pdom(1,:)>= max(spk_pdom(1,:))/3,1,'last');
        if thld_spkdomlgt < 3; thld_spkdomlgt = 3; end % keep >= 3 peaks
    if length(spk_pdom(1,:)) > thld_spkdomlgt % remove peaks after the last
        spk_pdom = spk_pdom(:,1:thld_spkdomlgt); end
    % repeat with prominence
    thld_spkdomlgt = find(spk_pdom(4,:) >= max(spk_pdom(4,:))/3,1,'last');
        if thld_spkdomlgt < 3; thld_spkdomlgt = 3; end
    if length(spk_pdom(1,:)) > thld_spkdomlgt
        spk_pdom = spk_pdom(:,1:thld_spkdomlgt); end
end

        %% remove spk_pdom that have an outliying prominance
        % identify the mean
        i_mean = mean(spk_pdom(4,:));
        % take all values above the mean and calculate a new mean and std
        thldmean = mean(spk_pdom(4,spk_pdom(4,:) > i_mean)) - ...
            3*std(spk_pdom(4,spk_pdom(4,:) > i_mean));
        % check that the new threshold is closer to 0 then to the mean
        %   threshold
        if abs(thldmean-mean(spk_pdom(4,spk_pdom(4,:)>i_mean)))>...
                abs(thldmean-0)
            % remove the prominance that do not hit the new threshold
            spk_pdom(:,spk_pdom(4,:) < thldmean ) = [];
        end

% exclusion criteria: at least 2 peaks need to be found
        if length(spk_pdom(1,:)) < 2
i_z = 0;
plot_title = '2c - exclusion: spk_pdom < two dominant peaks'; % image
ExclusionCriteria(1,3) = ExclusionCriteria(1,3)+1;
        end

    %%   identify fequency
        AC_frqu = spk_pdom(2,1);
        epiList_AC(1,i_epi) = AC_frqu;

    %%  identify Quality
        % the percentage size; location of the dominant peak in relation
        %   to the first local heigth of graph
        xc_lower_thld = min(xc_half2(:,1:spk_pdom(2,1)));
        xc_ysz = 1+abs(xc_lower_thld); % distance to lag 0
        sc_dpks_hgt_pct = (spk_pdom(1,:)+abs(xc_lower_thld))./xc_ysz ;

    end % if isnan(i_z)

     %% Exclusion criterial AC
```

```
% 1) AC frequ is to large to find 5 peaks with
    if isnan(i_z)
        if length(Y_d) <(AC_frqu*5)
i_z = 0;
plot_title = '4 - exclusion: Y_d smaller then 5 AC frqu '; % image
ExclusionCriteria(1,4) = ExclusionCriteria(1,4)+1;
        end
    end

% 2) AC frequ needs to be between slowWalk_obs and fastWalk_obs
    if isnan(i_z) % AC_frqu outside of slowWalk and fastWalk obs
        if (AC_frqu < fastWalk_obs) || (AC_frqu > slowWalk_obs)
 i_z = 0;
 plot_title='5 - exclusion: AC_frqu outside of slowWalk and fastWalk obs';
 ExclusionCriteria(1,5) = ExclusionCriteria(1,5)+1;
        end
    end % if isnan(i_z) % AC_frqu outside of slowWalk and fastWalk obs

% 3) dominant peaks need to be homogenous
    if isnan(i_z)
    % Ensure dinstance between the peaks is about the same
    % identify platform width
        % select primary peaks width
        bwpks_wid = spk_pdom(3,1);
        % platform width can not be larger then AC_frqu/2
        if bwpks_wid > AC_frqu/2; bwpks_wid = AC_frqu/2; end
        %   calculate the location of the with around the 1st peak
        accprange = [AC_frqu-floor(bwpks_wid/2) AC_frqu+ceil(bwpks_wid/2)];

     % run a loop around the peaks and platforms
        bwpks_list = [spk_pdom(2,:); ...
            accprange(1),nan(1,length(spk_pdom(2,:))-1);...
            accprange(2),nan(1,length(spk_pdom(2,:))-1); ...
            zeros(1,length(spk_pdom(2,:)))]; bwpks_list(4,1) = 1;
        ibwpks = 2;
        while (ibwpks <= length(bwpks_list(1,:))) && (isnan(i_z))
            % if the next column only has a peak
            if ~isnan(bwpks_list(1,ibwpks-1))
                % create platform for the current peak
                bwpks_list(2,ibwpks) = bwpks_list(1,ibwpks-1)+...
                    AC_frqu-floor(bwpks_wid/2);
                bwpks_list(3,ibwpks) = bwpks_list(1,ibwpks-1)+...
                    AC_frqu+ceil(bwpks_wid/2);
                % check if the current peak is on the platform
                if (bwpks_list(1,ibwpks) >= bwpks_list(2,ibwpks)) && ...
                        (bwpks_list(1,ibwpks) <= bwpks_list(3,ibwpks))
                    bwpks_list(4,ibwpks) = 1;
                else
                % declare episode non-gait if the peak falls before the pf
                    if bwpks_list(1,ibwpks) < bwpks_list(2,ibwpks)
                        bwpks_list(4,ibwpks) = 999;
i_z = 0;
plot_title = '6 - exclusion: Peak falls before the Platform';
ExclusionCriteria(1,6) = ExclusionCriteria(1,6)+1;
                    end
                % if peak comes after the platform, there is still a
                %   potential for a unfound peak -> create a new platform
                %   and test the peak again
                    if bwpks_list(1,ibwpks) > bwpks_list(3,ibwpks)
                    % there is still a potential for a unfound peak
                        % create a new colum before the empty peak
                        bwpks_list=[bwpks_list(:,1:ibwpks) [nan(3,1);0] ...
                            bwpks_list(:,ibwpks+1:end)];
                        % move the peak to the new coluum
                        bwpks_list(1,ibwpks+1) = bwpks_list(1,ibwpks);
                        % delete the peak in th current coluum
                        bwpks_list(1,ibwpks) = nan;
                        % note the start and end of the next platform
                       bwpks_list(2,ibwpks+1)=bwpks_list(2,ibwpks)+AC_frqu;
                       bwpks_list(3,ibwpks+1)=bwpks_list(3,ibwpks)+AC_frqu;
```

```
                    end
                end % (bwpks_list(1,ibwpks) >= bwpks_list(2,ibwpks)) ...
            else % ~isnan(bwpks_list(1,ibwpks-1))
            % if the next column only has a platform
                % if dot lies on the new platform
                if (bwpks_list(1,ibwpks) >= bwpks_list(2,ibwpks)) && ...
                        (bwpks_list(1,ibwpks) <= bwpks_list(3,ibwpks))
                    bwpks_list(4,ibwpks) = 1;
                else
            % if two platform in a row do not have a dot: discard episode
                    bwpks_list(4,ibwpks) = 999;
i_z = 0;
plot_title = '7 - exclusion: two platforms in a row'; % image
ExclusionCriteria(1,7) = ExclusionCriteria(1,7)+1;
                end
            end % ~isnan(bwpks_list(1,ibwpks-1))
            ibwpks = ibwpks+1;
        end % while command

    % check if all dots found platforms
        if sum(bwpks_list(4,:)) == length(bwpks_list(4,:))
i_z = 1;
plot_title = 'A - z = 1: all peaks are on platforms';
ExclusionCriteria(1,10) = ExclusionCriteria(1,10)+1;
        else
            % eine einzelsehende leere platform kann übrig bleiben
i_z = 0;
plot_title = '7 - exclusion: platforms are left empty';
ExclusionCriteria(1,8) = ExclusionCriteria(1,8)+1;

        end
    end % if isnan(i_z) % dominant peaks need to be homogenous


%% Summary/ Decission
    if i_z == 1
        % record data z
        z_AC_3D(epiList_loc(i_epi,3),...
            epiList_loc(i_epi,1):epiList_loc(i_epi,2)) = 1;

        % note z in epi
        epiList_z(1,i_epi) = 1;

        % Quality Controll
        %   note size (prominence) of dominant peak (the larger the better)
        epiList_Q(1,i_epi) = sc_dpks_hgt_pct(1,1);
        %   further info
        Quality_extend(i_epi).i_epi = i_epi;
        Quality_extend(i_epi).spk_pdom = spk_pdom;
        %       all higths
        Quality_extend(i_epi).heigth = sc_dpks_hgt_pct;
        %       all prom
        Quality_extend(i_epi).prom = spk_pdom(4,:);

        %       size:
        Quality_extend(i_epi).lgt = length(spk_pdom(1,:));
        %       noise:  additional peaks detected by loc_p0
        Quality_extend(i_epi).locadd = length(loc_p0(:, loc_p0 <= ...
            max(spk_pdom(2,:))))-length(spk_pdom(2,:));
        %       std:    heigth and prominance between peaks
        Quality_extend(i_epi).diffhgtprm = [std(sc_dpks_hgt_pct) ...
            std(spk_pdom(4,:))];
        %       std:    diffence in distance between locs !!!
        Quality_extend(i_epi).diffloc = std(diff(spk_pdom(2,:)));
        if std(diff(spk_pdom(2,:))) == 0
            Quality_extend(i_epi).diffloc = nan; end
        %       trend:  abfallend - heigth and prominance between peaks
        Quality_extend(i_epi).trendlinear_spk = ...
            polyfit(spk_pdom(2,:),spk_pdom(1,:),1);
        Quality_extend(i_epi).trendlinear_hgt = ...
```

```
            polyfit(spk_pdom(2,:),sc_dpks_hgt_pct,1);
        Quality_extend(i_epi).trendlinear_prm = ...
            polyfit(spk_pdom(2,:),spk_pdom(4,:),1);
        %       extra behaviour: gait cycle instead of steps
                    % secound peaks need to be larger than the previous
        if mod(length(sc_dpks_hgt_pct),2) == 0 % is even
            Quality_extend(i_epi).GaitCycle = ...
                sc_dpks_hgt_pct(2:2:end)-sc_dpks_hgt_pct(1:2:end);
        end
        if mod(length(sc_dpks_hgt_pct),2) == 1 % is odd
            Quality_extend(i_epi).GaitCycle = ...
                sc_dpks_hgt_pct(2:2:end)-sc_dpks_hgt_pct(1:2:end-1);
        end
        Quality_extend(i_epi).GaitCycle_1st = ...
            sc_dpks_hgt_pct(2)-sc_dpks_hgt_pct(1);

        % Lab
        if ~isempty(data_Lab)
            Quality_extend(i_epi).z_Lab = round(mean(...
                data_Lab(:,epiList_loc(i_epi,1):epiList_loc(i_epi,2))),0);
        end
    end % if i_z == 1

%% create Figures
    if FigureON == 1
    if ~isempty(data_Lab);plot_Lab_z = round(mean(data_Lab(:,...
            epiList_loc(i_epi,1):epiList_loc(i_epi,2))),0);
    else; plot_Lab_z = 1; end
    if (i_z == 1) || (plot_Lab_z == 2)
        data_d_nan = data_d;
        if ~isempty(data_Lab)
            data_d_nan(:,data_Lab(:,1:end-1) ~= 2) = nan;
        end
        iplot_str = epiList_loc(i_epi,1);
        iplot_end = epiList_loc(i_epi,2);
        iplot_lgtextra = iplot_end-iplot_str; if iplot_lgtextra > 1*50*60;
            iplot_lgtextra = 1*50*60; end % set to max 1 min
        if iplot_end > lgt_TotalData; iplot_end = lgt_TotalData; end
        close all
        fig_g = figure('units','normalized','outerposition',[0 0 1 1],...
            'visible','off');
        subplot(2,1,1); hold on
            if iplot_end+iplot_lgtextra < lgt_TotalData
                if iplot_str-iplot_lgtextra > 1
plot(iplot_str-iplot_lgtextra:iplot_end+iplot_lgtextra,...
    data_d_nan(:,iplot_str-iplot_lgtextra:iplot_end+iplot_lgtextra)');
                else
plot(1:iplot_end+iplot_lgtextra,data_d_nan(:,1:iplot_end+iplot_lgtextra)');
                end
            else
plot(iplot_str-iplot_lgtextra:lgt_TotalData,data_d_nan(:,...
    iplot_str-iplot_lgtextra:lgt_TotalData)');
            end
            plot(epiList_loc(i_epi,1):epiList_loc(i_epi,2),Y_d,'k')
            title(['In func-GDA-AC: epiList-loc: ' num2str(i_epi) ...
               ' on data-d: ' num2str(iplot_str) ' - ' num2str(iplot_end)])
            ylabel('acceleration (m/sec^2)')
            xlabel('observations')
       subplot(2,1,2); hold on
            area([plotcenter-1+fastWalk_obs plotcenter-1+slowWalk_obs],...
[max(xc) max(xc)],min(xc),'FaceColor',[0.75 0.75 0.75],'EdgeColor','none')
            if exist('spk_pdom','var'); findpeaks(xc,'MinPeakHeight',...
              min(spk_pdom(1,:)),'Annotate','extents'); else; plot(xc); end
            plot(plotcenter:plotcenter+length(xc_half)-1,xc_half)
            ylim([-1 1.1])
            % plot_curr.Legend.Location = 'southoutside';
            if exist('spk_pdom','var') % figure ledgent change
                plot_curr = gca;
plot_curr.Legend.String = {'Area bw Fast & Slow Walk Thlds','xc',...
    'peaks (spk_pdom height)','width','prominence'};
```

```
            end
            if exist('loc_p0','var')
                scatter(loc_p0+plotcenter-1,amp_p0)
plot_curr.Legend.String = {'Area bw Fast & Slow Walk Thlds','xc',...
    'peaks (spk-pdom height)','width','prominence','every Peak'};
            end
            if exist('loc_pdom','var')
                scatter(loc_pdom+plotcenter-1,amp_p0(ismember(loc_p0,...
                    loc_pdom)),'filled')
plot_curr.Legend.String = {'Area bw Fast & Slow Walk Thlds','xc',...
    'peaks (spk-pdom height)','width','prominence',...
    'All Peaks','Dominant Peaks'};
            end
            if exist('spk_pdom','var')
                scatter(spk_pdom(2,:)+plotcenter-1,spk_pdom(1,:),'filled')
plot_curr.Legend.String = {'Area bw Fast & Slow Walk Thlds','xc',...
    'Pks (spk-pdom height)','width','prominence',...
    'All Pks','Dom Pks','Dom Pks after exclusion'};
            end
if exist('bwpks_list','var')
    plot_x_loc = [];
    plot_y_amp = [];
    for iplot_bwpks =  find(~isnan(bwpks_list(2,:)))
        plot_i_str_loc = bwpks_list(2,iplot_bwpks)+plotcenter-1;
        plot_i_end_loc = bwpks_list(3,iplot_bwpks)+plotcenter-1;
        if plot_i_end_loc > length(xc); plot_i_end_loc= length(xc);
            plot_x_loc = [plot_x_loc, plot_i_str_loc,plot_i_end_loc];
            plot_y_amp = [plot_y_amp, plot_i_end_amp, plot_i_end_amp];
        else
            plot_i_nan_loc = bwpks_list(3,iplot_bwpks)+plotcenter+1;
            plot_i_end_amp = max(xc(:,plot_i_str_loc:plot_i_end_loc));
plot_x_loc = [plot_x_loc, plot_i_str_loc,plot_i_end_loc,plot_i_nan_loc];
            plot_y_amp = [plot_y_amp, plot_i_end_amp, plot_i_end_amp, nan];
        end
        plot_i_end_amp = max(xc(:,plot_i_str_loc:plot_i_end_loc));
plot_x_loc = [plot_x_loc, plot_i_str_loc,plot_i_end_loc,plot_i_nan_loc];
        plot_y_amp = [plot_y_amp, plot_i_end_amp, plot_i_end_amp, nan];
    end
    plot(plot_x_loc,plot_y_amp, 'k')
    plot_curr.Legend.String = {'Area bw Fast & Slow Walk Thlds','xc',...
        'Pks (spk-pdom height)','width','prominence',...
     'All Pks','Dom Pks','Dom Pks after exclusion','homogenous Platforms'};
end
            title(plot_title)
            plot_curr.Legend.Visible = 'off';
            ylabel('correlation (r)')
            xlabel('AC slide (obs)')
            if ~isempty(data_Lab)
                if plot_Lab_z == 2
saveas(gcf,[ 'IMG_' IMG_name  '_AC4_' num2str(i_epi,'%05d\n') ...
    '_' num2str(plot_Lab_z) '_z' num2str(i_z) '.tif'])
                else
saveas(gcf,[ 'IMG_' IMG_name  '_AC4_' num2str(i_epi,'%05d\n') '_' ...
    num2str(plot_Lab_z) '_z' num2str(i_z) '_error.tif'])
                end
            else
saveas(gcf,[ 'IMG_' IMG_name  '_AC4_' num2str(i_epi,'%05d\n')  '_z' ...
    num2str(i_z) '.tif'])
            end
            close all
    end % i_z == 1
    if (i_epi == 428) && (i_epi == 334)
        xc_half_cut = xc(:,plotcenter:plotcenter+loc_xc_first0);
        xc_half2(:,1:loc_xc_first0) = nan(1,1);

        figure; hold on; title(['Autocorrelation example epi = ' ...
            num2str(i_epi)]);
        xlabel('shift/lagg of the correlated signal in samples');
        ylabel('correlation');ylim([min(xc)-0.1 1.1])
        plot_x = [1:length(xc)]-ceil(length(xc)/2);
```

```
        plot(plot_x,xc,'b')
        plot([0:length(xc_half_cut)-1],xc_half_cut,'r')
        plot([0:length(xc_half)-1],xc_half,'r')
        plot([0:length(xc_half2)-1],xc_half2,'g')
        plot([plot_x(1,1) plot_x(end,end)],[0 0],'k')
        plot([plot_x(1,1) plot_x(end,end)],[mean(xc) mean(xc)],...
            'Color',[0.5 0.5 0.5])
    end
    end % FigureOn
end % for i_epi = 1:length(epiList_loc(:,1))

%% Clean UP
% create AC_epilist
% combine info
AC_epiList = [epiList_loc(:,1:4), epiList_AC', epiList_Q',...
    (1:length(epiList_loc(:,1)))'];
% remove non walking episodes
AC_epiList(epiList_z == 0,:) = [];
% create 1D z
z_AC = max(z_AC_3D);
% clean Quality
Quality_extend(epiList_z(1:length(Quality_extend)) == 0) = [];
% display Exclusion Criteria used
if FigureON == 1
    EC_Label = [...
'1 -  exclusion: xc does not reach 0                                ''; ...
'2a - exclusion: pot_pdom_list < two dominant peaks                 ''; ...
'2b&c - exclusion: loc_pdom < two dominant peaks                    ''; ...
'4 - exclusion: Y_d smaller then 5 AC frqu                          ''; ...
'5 - exclusion: AC_frqu outside of slowWalk and fastWalk obs        ''; ...
'6 - exclusion: A peak is closer to the previous peak then AC frqu  ''; ...
'7 - exclusion: two platforms in a row                              ''; ...
'7 - exclusion: platforms are left empty                            ''; ...
'                                                                   ''; ...
'A - z = 1: all peaks are on platforms                              '  ];
    save(['Results_AC_EC_' IMG_name '.mat'],'ExclusionCriteria','EC_Label')
end
end % function AC
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% IMG_Quality
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function IMG_Quality(Quality_extend,IMG_name,data_Lab)
%% prep
plot_size = extractfield(Quality_extend,'lgt');
plot_noise = extractfield(Quality_extend,'locadd');
plot_std_hgtprm = extractfield(Quality_extend,'diffhgtprm');
plot_std_hgt = plot_std_hgtprm(1:2:end);
plot_std_prm = plot_std_hgtprm(2:2:end);
plot_std_diffloc = extractfield(Quality_extend,'diffloc');
plot_trend_spk_slope = extractfield(Quality_extend,'trendlinear_spk');
plot_trend_spk_slope = plot_trend_spk_slope(1:2:end);
plot_trend_hgt_slope = extractfield(Quality_extend,'trendlinear_hgt');
plot_trend_hgt_slope = plot_trend_hgt_slope(1:2:end);
plot_trend_prm_slope = extractfield(Quality_extend,'trendlinear_prm');
plot_trend_prm_slope = plot_trend_prm_slope(1:2:end);
plot_bhv_GaitCylce = extractfield(Quality_extend,'GaitCycle_1st');

%% plot Quality
%% with Lab
if ~isempty(data_Lab)
    % prep
    z_ACLab = round(extractfield(Quality_extend,'z_Lab'),0);

    % einzelnd alle
    figure('units','normalized','outerposition',[0 0 1 1],'visible','off');
    sb_y = 3;   sb_x = 3;
     subplot(sb_y,sb_x,1); hold on; title('N of dom Spk')
        histogram(plot_size(:,z_ACLab == 2),'BinWidth',1) % size
        histogram(plot_size(:,z_ACLab ~= 2),'BinWidth',1)
     subplot(sb_y,sb_x,2); hold on; title('N of noise Spk')
        histogram(plot_noise(:,z_ACLab == 2),'BinWidth',1) % noise
        histogram(plot_noise(:,z_ACLab ~= 2),'BinWidth',1)
    subplot(sb_y,sb_x,3); hold on; title('GaitCycle')
        histogram(plot_bhv_GaitCylce(:,z_ACLab == 2))
        histogram(plot_bhv_GaitCylce(:,z_ACLab ~= 2))
    subplot(sb_y,sb_x,4); hold on; title('Std Heigth')
        histogram(plot_std_hgt(:,z_ACLab == 2)) % size
        histogram(plot_std_hgt(:,z_ACLab ~= 2))
    subplot(sb_y,sb_x,5); hold on; title('STD Prom')
        histogram(plot_std_prm(:,z_ACLab == 2))
        histogram(plot_std_prm(:,z_ACLab ~= 2))
    subplot(sb_y,sb_x,6); hold on; title('STD Diff Loc')
        histogram(plot_std_diffloc(:,z_ACLab == 2))
        histogram(plot_std_diffloc(:,z_ACLab ~= 2))
    subplot(sb_y,sb_x,7); hold on; title('Slope Spk')
        histogram(plot_trend_spk_slope(:,z_ACLab == 2),'BinWidth',0.001)
        histogram(plot_trend_spk_slope(:,z_ACLab ~= 2),'BinWidth',0.001)
    subplot(sb_y,sb_x,8); hold on; title('Slope Hgt')
        histogram(plot_trend_hgt_slope(:,z_ACLab == 2),'BinWidth',0.001)
        histogram(plot_trend_hgt_slope(:,z_ACLab ~= 2),'BinWidth',0.001)
    subplot(sb_y,sb_x,9); hold on; title('Slope Prm')
        histogram(plot_trend_prm_slope(:,z_ACLab == 2),'BinWidth',0.001)
        histogram(plot_trend_prm_slope(:,z_ACLab ~= 2),'BinWidth',0.001)
    saveas(gcf,[ 'IMG_' IMG_name  '_AC4_Quality_All.tif']); close all

    % calc
    plot_mag = plot_noise./plot_size;


    % Secelcted
    figure('units','normalized','outerposition',[0 0 1 1],'visible','off');
    sb_y = 3;   sb_x = 2;
    subplot(sb_y,sb_x,1); hold on; title('Comp N dom vs noise ')
scatter(plot_size(:,z_ACLab == 2),plot_noise(:,z_ACLab == 2),'filled');
xlim([0 max(plot_size)+1]); ylim([min(plot_noise)-1 max(plot_noise)+1])
scatter(plot_size(:,z_ACLab ~= 2),plot_noise(:,z_ACLab ~= 2),...
'Linewidth',1.5);   xlabel('N dom'); ylabel('N noise')
        plot([1 max([plot_size plot_noise])],[0 max([plot_size ...
```

```
            plot_noise])*2])
    subplot(sb_y,sb_x,2); hold on; title('Comp std hgt vs prm');
        scatter(plot_std_hgt(:,z_ACLab == 2),...
            plot_std_prm(:,z_ACLab == 2),'filled')
        scatter(plot_std_hgt(:,z_ACLab ~= 2),...
            plot_std_prm(:,z_ACLab ~= 2),'Linewidth',1.5);
        xlabel('std hgt'); ylabel('std prm')
    subplot(sb_y,sb_x,3); hold on; title('Slopes');
        scatter(plot_trend_hgt_slope(:,z_ACLab == 2),....
            plot_trend_prm_slope(:,z_ACLab == 2),'filled')
        scatter(plot_trend_hgt_slope(:,z_ACLab ~= 2),...
            plot_trend_prm_slope(:,z_ACLab ~= 2))
        xlabel('Heigth'); ylabel('Prom');
    subplot(sb_y,sb_x,4); hold on; title('STD Diff Loc')
        histogram(plot_std_diffloc(:,z_ACLab == 2))
        histogram(plot_std_diffloc(:,z_ACLab ~= 2)); ylim([0 10])
     subplot(sb_y,sb_x,5); hold on; title('Slope Hgt vs Gait C');
        scatter(plot_trend_hgt_slope(:,z_ACLab == 2),...
            plot_bhv_GaitCylce(:,z_ACLab == 2),'filled')
        scatter(plot_trend_hgt_slope(:,z_ACLab ~= 2),...
            plot_bhv_GaitCylce(:,z_ACLab ~= 2))
        xlabel('Heigth'); ylabel('Gait C');
        plot([min(plot_trend_hgt_slope) max(plot_trend_hgt_slope)], ...
            [0 0],'k');
        plot([0 0], [min(plot_bhv_GaitCylce) max(plot_bhv_GaitCylce)],'k')
     subplot(sb_y,sb_x,6); hold on; title('Slope Prm vs Gait C');
        scatter(plot_trend_prm_slope(:,z_ACLab == 2),...
            plot_bhv_GaitCylce(:,z_ACLab == 2),'filled')
        scatter(plot_trend_prm_slope(:,z_ACLab ~= 2),...
            plot_bhv_GaitCylce(:,z_ACLab ~= 2))
        xlabel('Heigth'); ylabel('Gait C');
        plot([min(plot_trend_prm_slope) max(plot_trend_prm_slope)],...
            [0 0],'k');
        plot([0 0], [min(plot_bhv_GaitCylce) max(plot_bhv_GaitCylce)],'k')
    saveas(gcf,[ 'IMG_' IMG_name  '_AC4_Quality_Selected.tif']); close all


    figure;  hold on;
        scatter(plot_size(:,z_ACLab == 2),...
            plot_std_diffloc(:,z_ACLab == 2),'filled')
        scatter(plot_size(:,z_ACLab ~= 2),plot_std_diffloc(:,z_ACLab ~= 2))
        xlim([min(plot_size)-1 max(plot_size)+1]);
        ylim([min(plot_std_diffloc)-1 max(plot_std_diffloc)+1])


else
%% without Lab
    figure('units','normalized','outerposition',[0 0 1 1],'visible','off');
    sb_y = 3;   sb_x = 3;
     subplot(sb_y,sb_x,1); hold on; title('N of dom Spk')
        histogram(plot_size,'BinWidth',1)
     subplot(sb_y,sb_x,2); hold on; title('N of noise Spk')
        histogram(plot_noise,'BinWidth',1)
    subplot(sb_y,sb_x,3); hold on; title('GaitCycle')
        histogram(plot_bhv_GaitCylce)
    subplot(sb_y,sb_x,4); hold on; title('Std Heigth')
        histogram(plot_std_hgt)
    subplot(sb_y,sb_x,5); hold on; title('STD Prom')
        histogram(plot_std_prm)
    subplot(sb_y,sb_x,6); hold on; title('STD Diff Loc')
        histogram(plot_std_diffloc)
    subplot(sb_y,sb_x,7); hold on; title('Slope Spk')
        histogram(plot_trend_spk_slope,'BinWidth',0.001) % size
    subplot(sb_y,sb_x,8); hold on; title('Slope Hgt')
        histogram(plot_trend_hgt_slope,'BinWidth',0.001)
    subplot(sb_y,sb_x,9); hold on; title('Slope Prm')
        histogram(plot_trend_prm_slope,'BinWidth',0.001)
    saveas(gcf,[ 'IMG_' IMG_name  '_AC4_Quality_All.tif']); close all

    % calc
```

```
    plot_mag = plot_noise./plot_size;
    figure('units','normalized','outerposition',[0 0 1 1],'visible','off');
    sb_y = 3;   sb_x = 2;
    subplot(sb_y,sb_x,1); hold on; title('Comp N dom vs noise ')
        scatter(plot_size,plot_noise,'Linewidth',1.5);
        xlim([0 max(plot_size)+1]); ylim([min(plot_noise)-1 ...
            max(plot_noise)+1]); xlabel('N dom'); ylabel('N noise')
 plot([1 max([plot_size plot_noise])],[0 max([plot_size plot_noise])*2])
    subplot(sb_y,sb_x,2); hold on; title('Comp std hgt vs prm');
        scatter(plot_std_hgt,plot_std_prm,'filled')
        xlabel('std hgt'); ylabel('std prm')
    subplot(sb_y,sb_x,3); hold on; title('Slopes');
        scatter(plot_trend_hgt_slope,plot_trend_prm_slope,'filled')
        xlabel('Heigth'); ylabel('Prom');
    subplot(sb_y,sb_x,4); hold on; title('STD Diff Loc')
        histogram(plot_std_diffloc);
        ylim([0 10])
     subplot(sb_y,sb_x,5); hold on; title('Slope Hgt vs Gait C');
        scatter(plot_trend_hgt_slope,plot_bhv_GaitCylce,'filled')
        xlabel('Heigth'); ylabel('Gait C');
plot([min(plot_trend_hgt_slope) max(plot_trend_hgt_slope)], [0 0],'k');
plot([0 0], [min(plot_bhv_GaitCylce) max(plot_bhv_GaitCylce)],'k')
     subplot(sb_y,sb_x,6); hold on; title('Slope Prm vs Gait C');
        scatter(plot_trend_prm_slope,plot_bhv_GaitCylce,'filled')
        xlabel('Heigth'); ylabel('Gait C');
plot([min(plot_trend_prm_slope) max(plot_trend_prm_slope)], [0 0],'k');
plot([0 0], [min(plot_bhv_GaitCylce) max(plot_bhv_GaitCylce)],'k')
    saveas(gcf,[ 'IMG_' IMG_name  '_AC4_Quality_Selected.tif']); close all
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% AstonGDACode_StepParameterExtraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% House-Keeping
clear all %  clear all variables from the current workspace
fclose('all'); % close all open files
close all %deletes all figures whose handles are not hidden

workONcode = 1;
CreateIMG = 0; % 1 = on; 0 = off

%% Version
Version_Dataset = 1;
% Selfrecorded Dataset = 1
% KOALAP Dataset = 2
% BioBank Dataset = 3
x_str = 1; % point to start the data extraction (musst be 1 or higher)
x_end = []; % point to end the data extraction (must be [] or a number > 1)

KOALAP_Version = 0;
% KOALAP Version: Not KOALAP = 0
% KOALAP Version: Original = 1
% KOALAP Version: New = 2

fprintf(['Dataset Type used: ' num2str(Version_Dataset)  ' \n'])

%% folder/file location/name
if Version_Dataset == 1
    CHOICE_dir_basis = ['D:\PhD_Matlab\Data_SelfRecorded\'];
    filename_load_structure = 'Results_GD_SCwCFw_';
    filename_variables = 'SetUP_Variables_idv_V1.mat';
    CHOICE_dir_basis_sup = ['...\AstonCode_StepParameters\'];
end
if Version_Dataset == 2
    CHOICE_dir_basis =
['.../data/extracted/AstonGDA_Output/Extracted02_OnlyWalking/'];
    CHOICE_dir_basis_Pain = ['.../data/extracted/PainScores/'];
    CHOICE_dir_output =
['.../data/extracted/AstonGDA_Output/Extracted03_StepParameters/'];
    filename_load_structure = 'UserRslt_GD_';
    filename_load_Pain = 'PainScores_raw';
    CHOICE_dir_basis_sup = ['.../scripts/AGDA_Parameters_Code/'];
    filename_variables = 'SetUP_Variables_idv_01.mat';
    addpath(CHOICE_dir_basis_Pain)
end
if Version_Dataset == 3
    CHOICE_dir_basis = '.../Data_extracted_UKB/';
    CHOICE_dir_output = ['.../Data_extracted_UKB/All_SPE_Output/'];
    filename_load_structure = 'Results_GD_';
    filename_variables = 'SetUP_Variables_idv_01.mat';
    CHOICE_dir_basis_sup = ['.../AGDA_Codes/StepParameterExtraction/'];
end
addpath(CHOICE_dir_basis)
addpath(CHOICE_dir_basis_sup)

%% Folder structure
if Version_Dataset == 3
    Folder_ON = 1;
    Folder_List = ['Participant_Group02';...
        'Participant_Group03';'Participant_Group04';...
        'Participant_Group05';'Participant_Group06';...
        'Participant_Group07';'Participant_Group08';...
        'Participant_Group09';'Participant_Group10';...
        'Participant_Group11';'Participant_Group12';...
        'Participant_Group13';'Participant_Group14';...
        'Participant_Group15';'Participant_Group16';...
        'Participant_Group17';'Participant_Group18';...
        'Participant_Group19';'Participant_Group20';...
        'Participant_Group21';'Participant_Group22';...
```

```
        'Participant_Group23';'Participant_Group24';...
        'Participant_Group25';'Participant_Group26';...
        'Participant_Group27';'Participant_Group28';...
        'Participant_Group29';'Participant_Group30';...
        'Participant_Group31'];
    Folder_List = ['Participant_Group30'];
    folder_xx = 1:length(Folder_List(:,1));
    SLASH = '/';

    %% load participants Ids
    load([CHOICE_dir_basis_sup 'UKBioBank_UserList.mat'])

else
    Folder_ON = 0; Folder_List = ['only one']; folder_xx = 1;
end

%% directory
if workONcode == 1
    cd(CHOICE_dir_basis_sup)
else
    cd(CHOICE_dir_basis)
end
%% Variabes (that do not change during the run)
% note: save in extra folder
minEpisode_steps_run = 5;
minEpisode_steps_eval = 15;
%QualityminEpisode_steps = 15;

% Time Cathegories
for x = 1:4
    if x == 1;   var_TimeOfDay(x).str_char =  '03:00:00.000';
                 var_TimeOfDay(x).end_char =  '08:59:59.999'; end
    if x == 2;   var_TimeOfDay(x).str_char =  '09:00:00.000';
                 var_TimeOfDay(x).end_char =  '14:59:59.999'; end
    if x == 3;   var_TimeOfDay(x).str_char =  '15:00:00.000';
                 var_TimeOfDay(x).end_char =  '20:59:59.999'; end
    if x == 4;   var_TimeOfDay(x).str_char =  '21:00:00.000';
                 var_TimeOfDay(x).end_char =  '02:59:59.999'; end
    var_TimeOfDay(x).str_num = datenum(var_TimeOfDay(x).str_char,...
        'HH:MM:SS.FFF')-datenum('00:00:00.000','HH:MM:SS.FFF');
    var_TimeOfDay(x).end_num = datenum(var_TimeOfDay(x).end_char,...
        'HH:MM:SS.FFF')-datenum('00:00:00.000','HH:MM:SS.FFF');
end
clear x

%% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %% %%%
%%                                 Start Code                              %%%
for folder_x = folder_xx
%% Identify Data files within the Folder
% extract all files in folder of interest
if Version_Dataset == 3
Folder_structure = dir([CHOICE_dir_basis Folder_List(folder_x,:) SLASH]);
else
Folder_structure = dir(CHOICE_dir_basis);
end
char_FilesInFolder = char({Folder_structure(1:end).name}'); % reformat

% remove files: wrong data format
ToKeepIfWith = '.mat';
idx_FilesIntest_format = find(not(cellfun('isempty', ...
    (strfind(cellstr(char_FilesInFolder), ToKeepIfWith)))));
char_FilesInFolder = char(char_FilesInFolder(idx_FilesIntest_format,:));

% remove files: wrong file name
ToKeepIfWith = filename_load_structure;
idx_FilesIntest_format = find(not(cellfun('isempty', ...
    (strfind(cellstr(char_FilesInFolder), ToKeepIfWith)))));
char_FilesInFolder = char(char_FilesInFolder(idx_FilesIntest_format,:));

if Version_Dataset == 1
```

```
    filename_load_pre = char_FilesInFolder(:,1:18);
    filename_load_mid = char_FilesInFolder(:,19:22);
    filename_load_pos = char_FilesInFolder(:,23:26);
end
if Version_Dataset == 2
    filename_load_pre = char_FilesInFolder(:,1:12);
    filename_load_mid = char_FilesInFolder(:,13:15);
    filename_load_pos = char_FilesInFolder(:,16:19);
end
if Version_Dataset == 3 % e.g. 'Results_GD_1000335.mat'
    filename_load_pre = char_FilesInFolder(:,1:11);
    filename_load_mid = char_FilesInFolder(:,12:18);
    filename_load_pos = char_FilesInFolder(:,19:22);
end
clear idx_FilesIntest_format Folder_structure char_FilesInFolder ...
    ToKeepIfWith

%% Load Variables
%   of Selfrecorded Datset
if (Version_Dataset == 1)
  if filename_load_mid(1,2) == '1'; load('SetUP_Variables_idv_V1.mat')
  else; load('SetUP_Variables_idv_V4.mat'); end
end
%   of Biobank dataset
if Version_Dataset==3
    % ignore some variabes that should not be loaded (Directories)
nonvarlist = ['CHOICE_dir_Output$|CHOICE_dir_basis$|' ...
    'CHOICE_dir_basis_sup$|CHOICE_dir_output'];
load([CHOICE_dir_basis Folder_List(folder_x,:) ...
    SLASH filename_variables],'-regexp',['^(?!' nonvarlist ')\w']);
end
%   of KOALAP dataset
if Version_Dataset==2
    var_feq = 50;
    slowWalk_StepSec = 0.78;
        slowWalk_obs = (1/slowWalk_StepSec)*var_feq;
    fastWalk_StepSec = 3.26;
        fastWalk_obs = (1/fastWalk_StepSec)*var_feq;
    % achtungloadinVaribales
    % load([CHOICE_dir_basis filename_variables]);
end

%% prep
if (Version_Dataset==3)||(KOALAP_Version==2);WalkEpi_output_AllUser=[]; end
ProcessingTime = [];
%% Loop loading data
if isempty(x_end); x_end = length(filename_load_pre(:,1)) ; end
for i_file = x_str:x_end
    tic
    fprintf(['User ' filename_load_mid(i_file,:) ' started. File ' ...
        num2str(i_file) ' of ' num2str(x_end) '. Folder: ' ...
                                    Folder_List(folder_x,:)  ' \n']);
    %% Load Data
    % Walking Data - load all
    if Version_Dataset ~= 3
        load([CHOICE_dir_basis filename_load_pre(i_file,:) ...
            filename_load_mid(i_file,:) filename_load_pos(i_file,:)]);
    else
        load([CHOICE_dir_basis Folder_List(folder_x,:) SLASH ...
            filename_load_pre(i_file,:) ...
            filename_load_mid(i_file,:) filename_load_pos(i_file,:)]);
    end
    % Load Pain Data
    %   only KOALAP
    if Version_Dataset == 2
        load([ CHOICE_dir_basis_Pain 'PainData_' ...
            filename_load_mid(i_file,:) filename_load_pos(i_file,:)]);
    end

%% %% %% %% %% %% %% %% %% %% %% Start Code %% %% %% %% %% %% %% %% %% %% %
```

```
    %% prep: redue walk_t
    if Version_Dataset ~= 3
walk_t_days = floor(walk_t);
walk_t_time = walk_t - walk_t_days;
if Version_Dataset ~= 1; Calc_RecordedData_d = unique(floor(t_range_min));
else; Calc_RecordedData_d = unique(floor(t_range)); end
    day_str = min(Calc_RecordedData_d); day_end = max(Calc_RecordedData_d);
walk_t_date = day_str+walk_t_days-1;
    [DayNumber,~] = weekday(walk_t_days);
walk_t_wk = DayNumber-1; walk_t_wk(:,walk_t_wk==0) = 7;
% create original t
walk_t_org = day_str+walk_t-1;
    clear DayNumber day_str day_end Calc_RecordedData_d

    else
walk_t_days = floor(walk_t);
walk_t_time = walk_t - walk_t_days;
day_str = floor(t_range(1));
walk_t_date = day_str+walk_t_days-1;
    [DayNumber,~] = weekday(walk_t_days);
walk_t_wk = DayNumber-1; walk_t_wk(:,walk_t_wk==0) = 7;
% create original t
walk_t_org = day_str+walk_t-1;
    clear DayNumber day_str
    end

%% Code Steps
    %% 1) select axis
    thld_dr_factor = 0.75;
    [walk_d_01_mSTD] = func_SPE_mstd(walk_d,var_feq,thld_dr_factor);

    %% 2): identify sustained axis
    thld_01 = fastWalk_obs*minEpisode_steps_run;
    thld_02 = fastWalk_obs*minEpisode_steps_eval;
    thld_03_base=minEpisode_steps_run; thld_03 = slowWalk_obs*thld_03_base;
    [walk_02_EpiAxis] = func_SPE_removeShortEpi(walk_d_01_mSTD,...
        walk_info_ACfreq,thld_01,thld_02,thld_03_base,thld_03);
    clear thld_01 thld_02 thld_03_base thld_03
    clear walk_d_01_mSTD

    %% 3) create AC
    % t_diff = length at which the AC window gets interupted in sec
    t_diff = 5; min_Steps = minEpisode_steps_run;
    wdw_ac = var_feq*10; wdw_sl = var_feq*1;
    [walk_AC_loc,walk_AC_amp] = func_SPE_ACslide(walk_02_EpiAxis,...
        walk_t_org,var_feq,wdw_ac,wdw_sl,...
        slowWalk_obs,fastWalk_obs,t_diff,min_Steps);
    clear t_diff min_Steps wdw_ac wdw_sl

    %% 4) Select main axis using wdwamp_diff
    factor_wdwamp = 2;
    [DomWalk_d_preGDA,DomWalk_t,DomWalk_axis,DomWalk_AC,~] = ...
        func_SPE_SelectMainAxis(walk_02_EpiAxis,walk_t_org,...
        walk_AC_loc,walk_info_ACfreq,factor_wdwamp,slowWalk_obs);
    clear factor_wdwamp

    %% 5) Identify Steps
    [~,~,~,wdwamp_hill02,wdwamp_vall02,~,~] = ...
        func_AmpOverWindowMax(DomWalk_d_preGDA,DomWalk_AC); %(Y,var_window)
    % format wdwamphill like the rest of the data
    DomWalk_hill_01 = nan(1,length(walk_02_EpiAxis(1,:)));
    DomWalk_hill_01(1,wdwamp_hill02(2,:)) = wdwamp_hill02(1,:);
    DomWalk_vall_01 = nan(1,length(walk_02_EpiAxis(1,:)));
    DomWalk_vall_01(1,wdwamp_vall02(2,:)) = wdwamp_vall02(1,:);
    clear wdwamp_hill02 wdwamp_vall02

    %% 5.b) identify filtered steps
    if 1 == 2
    FigureOn = 0;
    [DomWalk_f] = func_SPE_filter(DomWalk_d_preGDA,DomWalk_AC,var_feq,...
```

```
        FigureOn,DomWalk_hill_01,DomWalk_vall_01);
    end

    %% 6) identify Episodes of gait
    % devide into episodes were the t distance is larger then 2 sec.
    DomWalk_t_calc=DomWalk_t; DomWalk_t_calc(:,isnan(DomWalk_t_calc)) = 0;
    idx_change = find(abs(diff(DomWalk_t_calc)) > (1/24/60/60/var_feq)*2);
    strend_list = [1 idx_change+1 ; idx_change length(DomWalk_t)]';
    strend_list(DomWalk_t_calc(strend_list(:,1)') == 0,:) = [];
    clear idx_change  DomWalk_t_calc

    %% 7) Remove Hill Vall Violations by eigther ading or removing peaks
    DomWalk_hill_02 = DomWalk_hill_01; DomWalk_vall_02 = DomWalk_vall_01;
    FigureOn = 0;
    for i_epi = 1:length(strend_list(:,1))
        % extract data
        istr = strend_list(i_epi,1); iend = strend_list(i_epi,2);
        DomWalk_d_exp = DomWalk_d_preGDA(:,istr:iend);
        DomWalk_hill_pre = DomWalk_hill_01(:,istr:iend);
        DomWalk_vall_pre = DomWalk_vall_01(:,istr:iend);
        % apply function
        [DomWalk_hill_pos,DomWalk_vall_pos] = ...
            func_SPE_HiVaViolation(DomWalk_hill_pre,DomWalk_vall_pre,...
            DomWalk_d_exp,FigureOn);
        % apply HillVall Violations to the whole dataset
        DomWalk_hill_02(:,istr:iend) = DomWalk_hill_pos;
        DomWalk_vall_02(:,istr:iend) = DomWalk_vall_pos;
    end
    clear i_str i_end i_epi DomWalk_d_exp DomWalk_hill_pre ...
        DomWalk_vall_pre DomWalk_hill_pos DomWalk_vall_pos

%% 8) Identify gait and step
    DomWalk_d=nan(1,length(DomWalk_d_preGDA)); WalkEpi = []; FigureOn = 0;
    min_gaitCylces = 5;
    for i_epi =1:length(strend_list(:,1))
        % Extract data
        istr = strend_list(i_epi,1); iend = strend_list(i_epi,2);
%% 8.1) function: identify and clean up for Gait Cycles
        [DomWalk_d_rslt,max_mean, HillList_1st_pos,HillList_2nd_pos,...
            VallList_1st_pos,VallList_2nd_pos,~,~] = ...
        func_SPE_GaitCycleIdf(DomWalk_d_preGDA(:,istr:iend),...
            DomWalk_hill_02(:,istr:iend),DomWalk_vall_02(:,istr:iend),...
            min_gaitCylces,FigureOn,[istr iend]);
% 8.2) record output Structure if there are spikes
        if isempty(HillList_1st_pos); WalkEpi(i_epi).Calc_NSpk = 0; else
%% 8.3) find start and end idx of each gait and step cycle
% Gait Cylce: start point: first positive value before the DomHill
HillList_1st_loc = sort(HillList_1st_pos(2,:));
VallList_2nd_loc = sort(VallList_2nd_pos(2,:));
loclist_Gait_strend = func_SPE_CreateLocListTo0_V2(...
    HillList_1st_loc,VallList_2nd_loc,DomWalk_d_rslt);
%   Step cycle
HillList_1st_loc = sort([HillList_1st_pos(2,:) HillList_2nd_pos(2,:)]);
VallList_2nd_loc = sort([VallList_1st_pos(2,:) VallList_2nd_pos(2,:)]);
loclist_SPE_strend = func_SPE_CreateLocListTo0_V2(...
    HillList_1st_loc,VallList_2nd_loc,DomWalk_d_rslt);

%% 9) create a sturucture for output
% create a final data d
DomWalk_d(:,istr:iend) = DomWalk_d_rslt;
% identifier for data properties
WalkEpi(i_epi).strend = [istr iend]; WalkEpi(i_epi).i_epi_prev = i_epi;
WalkEpi(i_epi).idx_nan = find(isnan(DomWalk_d_rslt));
% use i_d = DomWalk_d(:,istr:iend); i_d(...idx_nan) = nan;
    % index that are nan within episode DomWalk_d(:,i_str:i_end)
WalkEpi(i_epi).orientation = max_mean;
    % orientation: 1 = no changes
    %              2 & 4 = removed first peak
    %              3 & 4 = mirror around axis
    % note: we saved this information for later use but did not had time
```

```
    %       to entangle it further.
WalkEpi(i_epi).axis = median(DomWalk_axis(:,istr:iend));
% identifier for episodes time properties
WalkEpi(i_epi).t.datenum = round(mean(walk_t_date(:,istr:iend)),0);
WalkEpi(i_epi).t.WK = round(mean(walk_t_wk(:,istr:iend)),0);
WalkEpi(i_epi).t.meanTime = mean( walk_t_time(:,istr:iend));

for x = 1:length(var_TimeOfDay)
    if var_TimeOfDay(x).end_num > var_TimeOfDay(x).str_num
        if (WalkEpi(i_epi).t.meanTime >= var_TimeOfDay(x).str_num) && ...
            (WalkEpi(i_epi).t.meanTime <= var_TimeOfDay(x).end_num)
            WalkEpi(i_epi).t.TC = x; end
    else % spezial case: the TC crosses midnigth
        if (WalkEpi(i_epi).t.meanTime >= var_TimeOfDay(x).str_num) || ...
            (WalkEpi(i_epi).t.meanTime <= var_TimeOfDay(x).end_num)
            WalkEpi(i_epi).t.TC = x; end
end;end
% catch if TC does not exist
if ~isfield(WalkEpi(i_epi).t,'TC')
    x_order = [var_TimeOfDay(:).str_num];
    [~,x] = min(abs(x_order-WalkEpi(i_epi).t.meanTime));
    WalkEpi(i_epi).t.TC = x;
end

% output from identified Steps
WalkEpi(i_epi).Calc_NSpk = length(HillList_1st_pos(1,:));
WalkEpi(i_epi).Spk.Hill1st = HillList_1st_pos;
WalkEpi(i_epi).Spk.Hill2nd = HillList_2nd_pos;
WalkEpi(i_epi).Spk.Vall1st = VallList_1st_pos;
WalkEpi(i_epi).Spk.Vall2nd = VallList_2nd_pos;
WalkEpi(i_epi).Spk.locList_Step = loclist_SPE_strend;
WalkEpi(i_epi).Spk.locList_Gait = loclist_Gait_strend;

clear HillList_1st_loc VallList_2nd_loc loclist_Gait_strend ...
    loclist_SPE_strend calc_StepN_TOD  calc_StepN_TOD_day  ...
    calc_StepN_TOD_h
        end
clear istr iend DomWalk_d_rslt max_mean ...
    HillList_1st_pos HillList_2nd_pos VallList_1st_pos VallList_2nd_pos
    end % for i_epi =1:length(strend_list(:,1))
    % remove 0 episodes
    WalkEpi([WalkEpi.Calc_NSpk]<5) = [];
    clear strend_list

%% 10) Step Parameters
% Orignal KOALAP Version were parameters for ach step are recorded
if KOALAP_Version == 1
    [WalkEpi_output] = func_SPE_EP_originalKOALAPversion(...
        WalkEpi_input,DomWalk_d,var_feq);
end

% New version where the mean of each Episode is taken, so that output is
%   standardised
[WalkEpi_output] = func_SPE_EP(WalkEpi,DomWalk_d,var_feq);
clear WalkEpi DomWalk_d

%% save parameters for user file
if Version_Dataset == 1
save(['RSLT_SP_SCwCFw_' filename_load_mid(i_file,:) '.mat'],...
    'WalkEpi_output','var_TimeOfDay','var_feq','t_range',...
    'walk_t_org','lgt_TotalData')
end
if (Version_Dataset == 2) && (KOALAP_Version == 1)
    save([CHOICE_dir_output 'RSLT_SP_' filename_load_mid(i_file,:) ...
        '.mat'],'WalkEpi_output','var_TimeOfDay','var_feq','t_range',...
        'walk_t_org','lgt_TotalData')
end

if (Version_Dataset == 3)
    %% 11) Save data for Participant
```

```
% Data
if i_file == 1; WalkEpi_output_AllUser = WalkEpi_output; else
WalkEpi_output_AllUser(i_file) = WalkEpi_output; end
% user ID
WalkEpi_output_AllUser(i_file).User_Name = filename_load_mid(i_file,:);
% user medical condition
user_id = str2num(filename_load_mid(i_file,:));
idx_user = find(BB_eid == user_id);
WalkEpi_output_AllUser(i_file).User_Condition_Patient = ...
    BB_Group_TruePatient(idx_user);
WalkEpi_output_AllUser(i_file).User_Condition_OA = BB_oa(idx_user);
WalkEpi_output_AllUser(i_file).User_Condition_RA = BB_ra(idx_user);
WalkEpi_output_AllUser(i_file).User_Condition_Gout = BB_gout(idx_user);
WalkEpi_output_AllUser(i_file).User_Condition_other = ...
    BB_other_cond(idx_user);
WalkEpi_output_AllUser(i_file).User_Condition_PairID = ...
    BB_wrngdata_pairidx(idx_user);
end
% version KOALAP
if (KOALAP_Version == 2)
    % remove medial conditions
    WalkEpi_output = rmfield(WalkEpi_output,{'User_Condition_Patient',...
        'User_Condition_OA','User_Condition_RA','User_Condition_Gout',...
        'User_Condition_other','User_Condition_PairID'});
    % add pain scores for each episode
    calc_timeEpi = WalkEpi_output.datenum + WalkEpi_output.meanTime;
        % find closest pain score for each Cathegory
    for p = 1:5
if p == 1; calc_pains_t = Pain_AGG_t; calc_pains_val = Pain_AGG_val; end
if p == 2; calc_pains_t = Pain_DyF_t; calc_pains_val = Pain_DyF_val; end
if p == 3; calc_pains_t = Pain_IMP_t; calc_pains_val = Pain_IMP_val; end
if p == 4; combine_OAMOPM_t = [Pain_OAM_t; Pain_OPM_t];
    combine_OAMOPM_val = [Pain_OAM_val; Pain_OPM_val];
    [calc_pains_t,calc_idx_oamopm] = sort(combine_OAMOPM_t);
    calc_pains_val = combine_OAMOPM_val(calc_idx_oamopm); end
if p == 5; calc_pains_t = Pain_QoL_t; calc_pains_val = Pain_QoL_val; end
        if length(calc_pains_t) > 1
        calc_t_mtx = repmat(calc_timeEpi,[length(calc_pains_t) 1]);
        calc_t_mtx_pain = repmat(calc_pains_t,[1 length(calc_timeEpi)]);
        calc_mtx_min = calc_t_mtx - calc_t_mtx_pain;
        calc_mtx_min(calc_mtx_min < 0) = nan;
        [calc_t_diff,calc_mtx_ind] = min(calc_mtx_min);
        calc_pain_scores = [calc_pains_val(calc_mtx_ind)]';
        else
            calc_pain_scores = nan;
        end
if p == 1;  WalkEpi_output.Pain_Typestr_Agg = Pain_AGG_type_str;
            WalkEpi_output.Pain_Typenum_Agg = Pain_AGG_type_num;
            WalkEpi_output.Pain_Closest_Agg = calc_pain_scores;
            WalkEpi_output.Pain_Timedif_Agg = calc_t_diff; end
if p == 2;  WalkEpi_output.Pain_Closest_DyF = calc_pain_scores;
            WalkEpi_output.Pain_Timedif_DyF = calc_t_diff; end
if p == 3;  WalkEpi_output.Pain_Typestr_IMP = Pain_IMP_type_str;
            WalkEpi_output.Pain_Typenum_IMP = Pain_IMP_type_num;
            WalkEpi_output.Pain_Closest_IMP = calc_pain_scores;
            WalkEpi_output.Pain_Timedif_IMP = calc_t_diff; end
if p == 4;  WalkEpi_output.Pain_Closest_oAMoPM = calc_pain_scores;
            WalkEpi_output.Pain_Timedif_oAMoPM = calc_t_diff; end
if p == 5;  WalkEpi_output.Pain_Closest_QoL = calc_pain_scores;
            WalkEpi_output.Pain_Timedif_QoL = calc_t_diff; end
    end
    % combine data
    if i_file == 1; WalkEpi_output_AllUser = WalkEpi_output; else
        WalkEpi_output_AllUser(i_file) = WalkEpi_output; end
    % add user ID
    WalkEpi_output_AllUser(i_file).User_Name = filename_load_mid(i_file,:);
    clear WalkEpi_output calc_pains_t calc_pains_val calc_timeEpi ...
        combine_OAMOPM_t combine_OAMOPM_val calc_idx_oamopm ...
        calc_t_mtx calc_t_mtx_pain calc_mtx_min calc_t_diff ...
        calc_mtx_ind calc_pain_scores
```

```
end
%% keepin track of codes time
toc
ProcessingTime(i_file) = toc;
time_toc = mean(ProcessingTime)+2*std(ProcessingTime);
time_left = time_toc*(x_end-i_file);
time_left_h =  floor(time_left/60/60);
time_left_min = floor((time_left -  time_left_h*60*60)/60);
time_clock = fix(clock);
time_clock_new = time_clock;
time_clock_new(4) = time_clock_new(4)+time_left_h;
while time_clock_new(4) > 24; time_clock_new(3) = time_clock_new(3)+1;
time_clock_new(4) = time_clock_new(4)-24; end
time_clock_new(5) = time_clock_new(5)+time_left_min;
while time_clock_new(5) > 60; time_clock_new(4) = time_clock_new(4)+1;
time_clock_new(5) = time_clock_new(5)-60; end
fprintf([ 'Time left: ' num2str(time_left_h) ' h  ' ...
    num2str(time_left_min) ' min; Code finished at: '  ...
    num2str(time_clock_new)  '\n'])
end % for i_file = x_str:x_end

%% save folder data
if Version_Dataset == 3
 save([CHOICE_dir_output 'RSLT_SPE_folder_' Folder_List(folder_x,13:19)...
     '.mat'],'WalkEpi_output_AllUser','ProcessingTime')
end
if (KOALAP_Version == 2)
    save([CHOICE_dir_output 'RSLT_SPE_All.mat'],....
        'WalkEpi_output_AllUser','ProcessingTime')
end
end % for folder_x = folder_xx
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_AmpOverWindowMax
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [wdwamp_max, wdwamp_min,wdwamp_diff,...
    wdwamp_hill,wdwamp_vall,...
    wdwamp_center,wdwamp_lgt] = ...
    func_AmpOverWindowMax(Y,var_window)
%% Purpose
% identify the maximum and minimum value in the window around the datapoint
% find peaks in the data at a distance of var_widow
%% FUNCTION INPUT
% Y             Data to analyse (in 1D -> d(1,:)
%               If you want to run this code on 3d data, write a loop for
%               it yourself.
% var_window    frequency at which the peaks are to be expected
%% FUNCTION OUTPUT
% wdwamp_max    max value of datapoints within the window at each datapoint
% wdwamp_min    min value of datapoints within the window at each datapoint
% wdwamp_diff   differentce between min and max
% wdwamp_hill   identified peaks at min distance of window
% wdwamp_vall   identified valleys at min distance of window

%% identify & check data dymensions
% var_type 1: data is 1 dimentional
% var_type 3: data is 3 dymensional
dym_Y = size(Y);
if dym_Y(1,1) ~=1
    fprintf('dimensions of dynmic data are wrong');     TerminateCode;
end
if dym_Y(1,2) <= max(var_window)
    fprintf('dynmic data input to short');              TerminateCode;
end
% var_type_wdw 1: var_window is one value for the whole section
% var_type_wdw 2: var_window is assigned to each Y(1,:).
dym_wdw = size(var_window);
if dym_wdw(1,2) == 1; var_type_wid = 1; else; var_type_wid = 2; end
if (dym_wdw(1,2) ~=1) && (dym_wdw(1,2) ~= dym_Y(1,2)) && (dym_wdw(1,1) ~=1)
    fprintf('frequency input has wrong dymensions');    TerminateCode;
end
if dym_wdw(1,1) ~= 1
    fprintf('frequency input has wrong dymensions');    TerminateCode;
end
%% prepare output
wdwamp_max = nan(1,dym_Y(1,2));
wdwamp_min = nan(1,dym_Y(1,2));
if var_type_wid == 1; var_wdw = var_window; end
%% run loop for each datapoint
for i_amp = 1:dym_Y(1,2)
    %% identify datasection to be evaluated for max aplitude
    if var_type_wid == 2; var_wdw = var_window(:,i_amp); end
    i_wdw_str = i_amp-floor(var_wdw/2);
    i_wdw_end = i_amp+ceil(var_wdw/2);
    if i_wdw_str < 1;           i_wdw_str = 1;           end
    if i_wdw_end > dym_Y(1,2);  i_wdw_end = dym_Y(1,2);  end
    % accound of nans
    if ~isnan(i_wdw_str) && ~isnan(i_wdw_end)
        %% find max & min value in this range
        wdwamp_max(1,i_amp) = max(Y(1,i_wdw_str:i_wdw_end));
        wdwamp_min(1,i_amp) = min(Y(1,i_wdw_str:i_wdw_end));
    end
end
%% identify peaks and valleys (steps)
loc_max = find(Y == wdwamp_max);    amp_max = Y(:,Y == wdwamp_max);
wdwamp_hill = [amp_max;loc_max];
loc_min = find(Y == wdwamp_min);    amp_min = Y(:,Y == wdwamp_min);
wdwamp_vall = [amp_min;loc_min];
%% nessesary clean up
%   find flat spikes (where one plato identified 2 spikes)
%   (is an error of the func_AmpOverWindowMax methode
%   had to be repeared as participant 16 had a flat start of the section
```

```
for i_hiva = 1:2
    % select hill or vall to run
    if i_hiva == 1; wdwamp_cur = wdwamp_hill; end
    if i_hiva == 2; wdwamp_cur = wdwamp_vall; end
    %   identify location of consecuitive peaks
    idx_error = find(diff(wdwamp_cur(2,:)) == 1);
    % identify Sequences
    prep_idxerror = idx_error;
    prep_idxerror(end+1) = -1; % Add a new isolating point end
            % Find indexes of isolating points
    idx_deviderpoints = find(diff(prep_idxerror) ~= 1);
    [~,lgt_dvp] = size(idx_deviderpoints);
    Start_Idx = 1 ; % Set start index
    runloc_str = []; runloc_end = [];
    for i_dvp = 1:lgt_dvp
        End_Idx = idx_deviderpoints(i_dvp); % Set end index
            % Find consecuative sequences
        Sequ = [prep_idxerror(Start_Idx:End_Idx) prep_idxerror(End_Idx)+1];
            % update start index for the next consecuitive sequence
        Start_Idx = End_Idx + 1;
        runloc_str = [runloc_str,Sequ(1)];
        runloc_end = [runloc_end,Sequ(end)];
    end
    % identify spikes to be deleted for each run
    note_delete = [];
    for i_run = 1:length(runloc_str)
        % if the pltau number is uneven take the center
        lgt_run = length(runloc_str(i_run):runloc_end(i_run));
        % remove the surrounding peaks
        %   removes everything that needs to be removed if lgt_run is
        %   uneven
        if  mod(lgt_run,2) == 1
            % remove 1 extra since you add to the start
            n_deleterange = floor(lgt_run/2)-1;                          end
        if  mod(lgt_run,2) == 0
            % remove 2 extra since else the center will be taken away too
            n_deleterange = floor(lgt_run/2)-2;                          end
        if n_deleterange > 0
            note_delete = [note_delete,...
            runloc_str(i_run):runloc_str(i_run)+n_deleterange,...
            runloc_end(i_run)-n_deleterange:runloc_end(i_run)];           end
        % select center if lgt_run is even (simply remove the left value
        if mod(lgt_run,2) == 0
            idx_center_left = lgt_run/2;
           note_delete = [note_delete,runloc_str(i_run)+idx_center_left-1];
        end
    end
    % clean up
    note_delete = unique(note_delete);
    wdwamp_cur(:,note_delete) = [];
    if i_hiva == 1; wdwamp_hill = wdwamp_cur; end
    if i_hiva == 2; wdwamp_vall = wdwamp_cur; end
end
%% Other
% identify the data range
wdwamp_diff = wdwamp_max-wdwamp_min;
% identify center point
wdwamp_center = wdwamp_min + (wdwamp_diff/2);
% identify number of hills and valleys
wdwamp_lgt = [length(wdwamp_hill(1,:)) length(wdwamp_vall(1,:)) ...
              length(wdwamp_hill(1,:))+length(wdwamp_vall(1,:))];
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_GDA_mSTD
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [z_mSTD,z_mSTD_3D,output_stdthld] = func_GDA_mSTD(...
    z_Base,z_Base_3D,data_dr,var_feq,...
    FigureON,IMG_name,data_Lab,UserID,STDthld_standart)
%% Purpose
% to remove sections such as sleeping or rest

%% FUNCTION INPUT
% data_dr       magnitude of the data: dr = sqrt(sum((data_d').^2,2))'
% var_feq       Frequency at which the data was recorded

%% FUNCTION OUTPUT
% z_mSTD        vector labeling data as potential gait (1) or rest (0)

%% Codes needed from Supplementary
% movingstd         (https://uk.mathworks.com/matlabcentral/fileexchange/
%                       9428-movingstd-movingstd2)

%% run code form inside
if 1 == 2
    z_Base = z01_domAxis; z_Base_3D = z01_domAxis_3D;
    FigureON = CreateIMG; IMG_name = fn_save; UserID = i_file;
end

%% Moving STD
% identify moving STD
Y_STDfilter = movingstd(data_dr,var_feq);

% identify histogram of filtered data
rslt_hgf_01 = histcounts(Y_STDfilter,'BinWidth',0.01);

i_z_1D = z_Base;
i_thld_pass = 0;
figure_data = [];
while_condition = 1;
xxx = 1;
while (while_condition > 0) && (while_condition < 7)
    if FigureON == 1; figure_data(xxx).data_dr = data_dr(:,i_z_1D == 1);end

    %% identify histogram
    i_hg_dr = histcounts(data_dr(:,i_z_1D == 1),'BinWidth',0.01);
    % make sure that the peack is not onwith in first bin and can therefore
    % not be found.
    i_hg_dr(1) = 0;
    % identify gradient

    %% check if theshold should still be applied
    % histogram need to have a remarkable peak
    [ii_amp,ii_loc,ii_wid,ii_prm] = findpeaks(i_hg_dr,'MinPeakHeight',...
        max(i_hg_dr)/2,'MinPeakDistance',STDthld_standart*100);
        % use of Min Heigth to prevent
    % select maximumpeak
    i_fpidx =  find(ii_amp == max(ii_amp));
    i_amp = ii_amp(i_fpidx); i_loc = ii_loc(i_fpidx);
    i_wid = ii_wid(i_fpidx); i_prm = ii_prm(i_fpidx);

    %% identify threshold
    % use: range of wid
    i_thld = i_loc *0.01;
    i_thld_pass = i_loc*2 *0.01;

    %% check threshold
    if (i_thld_pass < STDthld_standart)
        %% record new data to be used (z in 1D)
        i_z_1D(Y_STDfilter < i_thld) = 0;
        output_stdthld = i_thld;
        %% finish While set up
        while_condition = while_condition + 1;
```

```
    else; while_condition = -1;  end
    %% interlude
    % record data for plots
    if FigureON == 1
        figure_data(xxx).histogram = i_hg_dr;
        figure_data(xxx).findpeaks = [i_amp;i_loc;i_wid;i_prm];
        figure_data(xxx).findpeaks_all = [ii_amp;ii_loc;ii_wid;ii_prm];
        figure_data(xxx).threshold = i_thld_pass;
        figure_data(xxx).z_new = i_z_1D;
        figure_data(xxx).data_zf = data_dr(:,i_z_1D == 1);
        figure_data(xxx).data_z = data_dr(:,i_z_1D == 1);
        xxx = xxx+1;
    end
end
%% select final 1D
z_mSTD = i_z_1D;
%% describe 3D
z_mSTD_3D = z_Base_3D;
z_mSTD_3D(:,i_z_1D == 0) = 0;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_SPE_removeShortEpi
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [walk_d3] = func_SPE_removeShortEpi(walk_d2,walk_info_ACfreq,...
    thld_01,thld_02,thld_03_base,thld_03)
% thld_01 = fastWalk_obs*5;
% thld_02 = fastWalk_obs*15;
% thld_03_base = 5;
% thld_03 = slowWalk_obs*thld_03_base;
walk_d3 = walk_d2;
walk_nan = ~isnan(walk_d3);
for i_axis = 1:3
    % create a list of all episodes
    strend_list = ...
        [[1 find(diff(walk_nan(i_axis,:)))+1                    ]', ...
        [  find(diff(walk_nan(i_axis,:))) length(walk_d3(1,:))]'];
    if walk_nan(i_axis,1) == 1
        strend_list = strend_list(1:2:end,:);
    end
    if walk_nan(i_axis,1) == 0
        strend_list = strend_list(2:2:end,:);
    end
    % identify list length
    strend_list_lgt = strend_list(:,2)-strend_list(:,1);
    % remove anly smaller then 5 fast step
    idx_del_list = find(strend_list_lgt < thld_01)';
    for i_del = idx_del_list
        walk_d3(i_axis,strend_list(i_del,1):strend_list(i_del,2)) = nan;
    end
    strend_list(idx_del_list,:) = [];
    strend_list_lgt(idx_del_list,:) = [];
    % remove smaller then 15 fast steps if they are surrounded
    idx_del_list = find(strend_list_lgt < thld_02)';
    for i_del = idx_del_list
        i_walknan = walk_nan(:,strend_list(i_del,1):strend_list(i_del,2));
        % check if there is an uninterupted other axis overlying this axis
        sum_walknan = sum(i_walknan');
        if (sum_walknan(1)==sum_walknan(2))||(sum_walknan(1)==...
                sum_walknan(3))||(sum_walknan(3)==sum_walknan(2))
            walk_d3(i_axis,strend_list(i_del,1):strend_list(i_del,2))= nan;
        end
    end
    % remove smaller then 15 slow steps if the episode is not long to
    %   contain 15 stepes taken at the speed of walk_info_ACfreq
    idx_del_list = find(strend_list_lgt < thld_03)';
    for i_del = idx_del_list
        i_walknan = min(walk_info_ACfreq(i_axis,strend_list(i_del,1):...
            strend_list(i_del,2)));
     if isnan(i_walknan)||(strend_list_lgt(i_del) < i_walknan*thld_03_base)
          walk_d3(i_axis,strend_list(i_del,1):strend_list(i_del,2)) = nan;
     end
    end
end % for i_axis = 1:3
end % function
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_SPE_ACslide
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [mtx_loc, mtx_amp] = func_SPE_ACslide(walk_d_AC,walk_t,...
    var_feq,wdw_ac,wdw_sl,slowWalk_obs,fastWalk_obs,t_diff,min_Steps)

%% select methode
max_output = 0; % problem: often selects gait cycle instead of steps cycle
Ist_output = 1; % select 1st

%% set up varibales
var_feq_datenum = 1/24/60/60/var_feq; % time diff between obseration
lgt_walk = length(walk_d_AC(1,:)); % length of episode

%% set up window AC
% prep output for the whole dataset
mtx_sz = ceil(wdw_ac/wdw_sl);
mtx_amp = nan(3,lgt_walk); mtx_loc = nan(3,lgt_walk);
% loop axis
for i_axis = 1:3
    % handle data including nan
    if any(isnan(walk_d_AC(i_axis,:)))
       i_idx = find(~isnan(walk_d_AC(i_axis,:)));
       i_walk_d = walk_d_AC(i_axis,i_idx);
       i_walk_t = walk_t(1,i_idx);
       lgt_walk = length(i_walk_d(1,:)); % new length of the dataset
    else
        i_walk_d = walk_d_AC(i_axis,:);
        i_walk_t = walk_t(1,:);
    end
    % identify location of breaks between episodes (marked START of an epi)
    loc_forcedStop = find(diff(i_walk_t) > var_feq_datenum*t_diff)+1;
    % prep output for the axis data
    i_mtx_amp = nan(mtx_sz,lgt_walk); i_mtx_loc = nan(mtx_sz,lgt_walk);
    i_wdw = 1; n_mtx = 1;

    %% run windows
    while i_wdw  > 0
        % select window
        i_str = (wdw_sl*i_wdw)-wdw_sl+1;
        i_end = (wdw_sl*i_wdw)-wdw_sl+wdw_ac;
        % identify if the end of the dataset has been reached
        if i_end > lgt_walk; i_end = lgt_walk; i_wdw = -10; end
        % identify if episodes are interrupted
        if (any(loc_forcedStop >= i_str)) && (any(loc_forcedStop <= i_end))
            idx_fSL = intersect(find(loc_forcedStop >= i_str+1),...
                find(loc_forcedStop <= i_end-1));
                % +1 and -1 accounts for instences where forcedStopLoc
                %   falls onto the start end of i_str:i_end
            lgt_fSL = length(idx_fSL)+1;
        else
            lgt_fSL = 1; % no interuption takes place
        end

        %% prep AC for the current window/windows
        for i_loop_fSL = 1:lgt_fSL
            % identify new str and end points of the window
            if lgt_fSL == 1
                ii_str = i_str; ii_end = i_end;
            else
                if i_loop_fSL == 1;         ii_str = i_str; else
                    ii_str = loc_forcedStop(idx_fSL(i_loop_fSL-1));
                end
                if i_loop_fSL == lgt_fSL;  ii_end = i_end; else
                    ii_end = loc_forcedStop(idx_fSL(i_loop_fSL))-1;
                end
            end
            % identify if the datasection is long enough to run a window
            if ii_end-ii_str > floor(fastWalk_obs*min_Steps)
                %% run AC
```

```
        i_xc = xcorr(i_walk_d(:,ii_str:ii_end),ceil(slowWalk_obs),'coeff');
        % extract area after xc = 1
        i_xc = i_xc(1,ceil(length(i_xc)/2):end);
        % exclude are that is fater then  fast step
        i_xc(:,1:floor(fastWalk_obs)) = nan; i_xc(:,i_xc < 0) = nan;

                %% Version 1) identify max loc
                if max_output == 1
                    i_pot_loc = peakseek(i_xc);
                    if ~isempty(i_pot_loc)
                        i_pot_amp = i_xc(i_pot_loc);
                        [pot_amp,pot_idx] = max(i_pot_amp);
                        pot_loc = i_pot_loc(pot_idx);
                        % record output
                        if (pot_amp>0.25)&&(pot_loc>floor(fastWalk_obs))...
                                &&(pot_loc<ceil(slowWalk_obs))
                            i_mtx_amp(n_mtx,ii_str:ii_end) = pot_amp;
                            i_mtx_loc(n_mtx,ii_str:ii_end) = pot_loc;
                end;end;end

                %% Version 2) identify first output
                if Ist_output == 1
                    i_pot_loc = peakseek(i_xc);
                    if ~isempty(i_pot_loc)
                        pot_loc = i_pot_loc(1,1);
                        pot_amp = i_xc(pot_loc);
                        if (pot_amp>0)&&(pot_loc>floor(fastWalk_obs)) ...
                                &&(pot_loc<ceil(slowWalk_obs))
                            i_mtx_amp(n_mtx,ii_str:ii_end) = pot_amp;
                            i_mtx_loc(n_mtx,ii_str:ii_end) = pot_loc;
                end;end;end
            end % if ii_end-ii_str > floor(fastWalk_obs*min_Steps)
        end % for i_loop_fSL = 1:lgt_fSL

        %% finish loop
        i_wdw = i_wdw+1; n_mtx = n_mtx+1; if n_mtx > mtx_sz, n_mtx = 1; end
    end % while i_wdw  > 0

    %%  reconstruct output matrix
    if any(isnan(walk_d_AC(i_axis,:)))
        mtx_loc(i_axis,i_idx) = median(i_mtx_loc,'omitnan');
        mtx_amp(i_axis,i_idx) = median(i_mtx_amp,'omitnan');
    else
        mtx_loc(i_axis,:) = median(i_mtx_loc,'omitnan');
        mtx_amp(i_axis,:) = median(i_mtx_amp,'omitnan');
    end
end % for i_axis = 1:3
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_SPE_SelectMainAxis
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [DomWalk_d,DomWalk_t,DomWalk_axis,DomWalk_AC,DomWalk_d_plot] = ...
    func_SPE_SelectMainAxis(walk_d3,walk_t,...
    mtx_loc,walk_info_ACfreq,factor_wdwamp,slowWalk_obs)

    wdwamp_diff = nan(3,length(walk_d3(1,:)));
    for i_axis = 1:3
        [~, ~,i_wdwamp_diff,~,~,~,~] = ...
        func_AmpOverWindowMax(walk_d3(i_axis,:),....
        mtx_loc(i_axis,:)*factor_wdwamp);
        wdwamp_diff(i_axis,:) = i_wdwamp_diff;
    end

    % accound for very quick changes(at least 1step present to change axis)
    [~,DomWalk_axis_pre] = max(wdwamp_diff);
    % change axis only if a min of 1 step is present
idx_dwAxis = [1 find(diff(DomWalk_axis_pre)~= 0) length(DomWalk_axis_pre)];
idx_idx_dwAxis = find(diff(idx_dwAxis) < slowWalk_obs);
% idx_idx_dwAxis(1,1) = 3   idx_dwAxis(3) --> idx_dwAxis(4)-idx_dwAxis(3)->
%                           DomWalk_axis_pre(idx_dwAxis(3:4)) = [2 1] -->
%                           DomWalk_axis_pre(idx_dwAxis(3)+1:idx_dwAxis(4))

    DomWalk_axis = DomWalk_axis_pre;
    for i_dwA = idx_idx_dwAxis
        % identify min AC
        i_minAC = min(min(walk_info_ACfreq(:,idx_dwAxis(i_dwA)+1:...
            idx_dwAxis(i_dwA+1))));
            if isnan(i_minAC); i_minAC = slowWalk_obs; end
        if idx_dwAxis(i_dwA+1)-idx_dwAxis(i_dwA) < i_minAC
% identify surrounding areas --> if there are the same fill space with axis
%   achtung BioBank fix: need to fix a bug were data outside of the
%       datarange gets requested
            fix_end_location = idx_dwAxis(i_dwA+1)+1;
            if fix_end_location > length(DomWalk_axis_pre);
                fix_end_location = length(DomWalk_axis_pre); end
            if DomWalk_axis_pre(idx_dwAxis(i_dwA)) == ...
                    DomWalk_axis_pre(fix_end_location)
                DomWalk_axis(idx_dwAxis(i_dwA)+1:idx_dwAxis(i_dwA+1)) = ...
                    DomWalk_axis_pre(idx_dwAxis(i_dwA));
    end;end;end

    % apply axis to create Dominant 1D plot
    DomWalk_d = nan(1,length(walk_d3(1,:)));
    DomWalk_AC = nan(1,length(walk_d3(1,:)));
    DomWalk_d_plot = nan(3,length(walk_d3(1,:)));
    for i_axis = 1:3
DomWalk_d(:,DomWalk_axis ==i_axis) = walk_d3(i_axis,DomWalk_axis ==i_axis);
DomWalk_AC(:,DomWalk_axis ==i_axis) =mtx_loc(i_axis,DomWalk_axis ==i_axis);
DomWalk_d_plot(i_axis,DomWalk_axis==i_axis)=...
    walk_d3(i_axis,DomWalk_axis==i_axis);
    end
    DomWalk_t = walk_t; DomWalk_t(1,isnan(DomWalk_d)) = nan;
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_SPE_HiVaViolation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [DomWalk_hill_exp,DomWalk_vall_exp] = ...
    func_SPE_HiVaViolation(DomWalk_hill_exp,DomWalk_vall_exp,...
    DomWalk_d_exp,FigureON)

DomWalk_hill_plot = DomWalk_hill_exp;
DomWalk_vall_plot = DomWalk_vall_exp;

%% check for valley hill violations within the episode
DomWalk_obs = 1:length(DomWalk_hill_exp);
list_error_hivaViolation = sortrows([[DomWalk_obs(:,~isnan(...
    DomWalk_hill_exp)); ones(1,length(find(~isnan(DomWalk_hill_exp))))],...
    [DomWalk_obs(:,~isnan(DomWalk_vall_exp)); zeros(1,length(find(...
    ~isnan(DomWalk_vall_exp))))]]',1)';
idx_hivaVio = find(diff(list_error_hivaViolation(2,:)) == 0);

%% check if violations happend
if ~isempty(idx_hivaVio)
    %% identify average hill/vall values
error_amp_mstd_hill = [mean(DomWalk_hill_exp(:,~isnan(DomWalk_hill_exp...
    ))),std(DomWalk_hill_exp(:,~isnan(DomWalk_hill_exp)))];
error_amp_mstd_vall = [mean(DomWalk_vall_exp(:,~isnan(DomWalk_vall_exp...
    ))),std(DomWalk_vall_exp(:,~isnan(DomWalk_vall_exp)))];
error_amp_rang_hill = [error_amp_mstd_hill(1,1)-error_amp_mstd_hill(1,2)...
    , error_amp_mstd_hill(1,1)+error_amp_mstd_hill(1,2)];
error_amp_rang_vall = [error_amp_mstd_vall(1,1)-error_amp_mstd_vall(1,2)...
    , error_amp_mstd_vall(1,1)+error_amp_mstd_vall(1,2)];
error_loc_mstd_hill = [mean(diff(DomWalk_obs(:,~isnan(DomWalk_hill_exp...
    )))), std(diff(DomWalk_obs(:,~isnan(DomWalk_hill_exp))))];
error_loc_mstd_vall = [mean(diff(DomWalk_obs(:,~isnan(DomWalk_vall_exp...
    )))), std(diff(DomWalk_obs(:,~isnan(DomWalk_vall_exp))))];
error_loc_rang_hill = [error_loc_mstd_hill(1,1)-error_loc_mstd_hill(1,2)...
    , error_loc_mstd_hill(1,1)+error_loc_mstd_hill(1,2)];
error_loc_rang_vall = [error_loc_mstd_vall(1,1)-error_loc_mstd_vall(1,2)...
    , error_loc_mstd_vall(1,1)+error_loc_mstd_vall(1,2)];

    %% loop violations
    for i_err = idx_hivaVio
    % set up
        potential_deletion = 0; potential_addition = 0;
        sure_deletion = 0; sure_addition = 0;
        i_list_err = list_error_hivaViolation(:,i_err:i_err+1);
        i_d = DomWalk_d_exp(:,i_list_err(1,1):i_list_err(1,2));

    %% decide if there is a valley or hill problem & extract info
        if (i_list_err(2,1)==0)&&(i_list_err(2,2)==0)% 2 valleys present
            error_rang_loc = error_loc_rang_vall;
            error_rang_loc_flip = error_loc_rang_hill;
            error_mean_amp = error_amp_mstd_vall;
            error_mean_amp_flip = error_amp_mstd_hill;
            error_rang_amp = error_amp_rang_hill;
            error_DomWalk = DomWalk_vall_exp;
            error_DomWalk_flip = DomWalk_hill_exp;
            [error_amp_val,error_amp_loc] = max(i_d);
        else % 2 hills present
            error_rang_loc = error_loc_rang_hill;
            error_rang_loc_flip = error_loc_rang_vall;
            error_mean_amp = error_amp_mstd_hill;
            error_mean_amp_flip = error_amp_mstd_vall;
            error_rang_amp = error_amp_rang_vall;
            error_DomWalk = DomWalk_hill_exp;
            error_DomWalk_flip = DomWalk_vall_exp;
            [error_amp_val,error_amp_loc] = min(i_d);
        end
        error_loc_val = i_list_err(1,2)-i_list_err(1,1);

    %% identify addition or deletion criteria
%  note: all descrbtions from the point of vie of two consecuitive valleys
```

```
%  potential addition: max(data) between valleys is closer mean hill
%                      then mean valley
        distancetoVall = abs(diff([error_amp_val error_mean_amp(1,1)]));
      distancetoHill = abs(diff([error_amp_val error_mean_amp_flip(1,1)]));
        %   IF the found potential peak amplitude is closer to the othere
        %       hills then to its surrounding valley
        if distancetoHill < distancetoVall
            potential_addition = potential_addition+1;
            % make it a sure peak if it is really close
            if distancetoVall/(distancetoVall+distancetoHill) > 0.75
                sure_addition = 1;
            end
        %   check IF the surounding peaks have space for another peak
            idx_surr = [i_err-1 i_err+2];
            if (idx_surr(1,1) > 0) && (idx_surr(1,2) <= length(...
                list_error_hivaViolation(1,:)))% make sure no errors happen
                i_loc_diff = list_error_hivaViolation(1,idx_surr(1,2))-...
                    list_error_hivaViolation(1,idx_surr(1,1));
                if i_loc_diff > error_rang_loc_flip(1,2)
                        potential_addition = potential_addition+1;
                else;   sure_deletion = 1;
                end
            else;       potential_deletion = potential_deletion+1;
            end
        else;           sure_deletion = 1;
        end % if distancetoHill < distancetoVall

    %% apply deletion of addition
        if sure_deletion == 1
           [~,iii_idx_delete] = min(abs(error_DomWalk(:,i_list_err(1,:))));
            error_DomWalk(:,i_list_err(1,iii_idx_delete)) = nan;
        else
            if sure_addition == 1
error_DomWalk_flip(:,i_list_err(1,1)+error_amp_loc-1) = error_amp_val;
            else
                if potential_addition >  potential_deletion % add peak
error_DomWalk_flip(:,i_list_err(1,1)+error_amp_loc-1) = error_amp_val;
                else
[~,iii_idx_delete] = min(abs(error_DomWalk(:,i_list_err(1,:))));
error_DomWalk(:,i_list_err(1,iii_idx_delete)) = nan;
                end
            end
        end

    %% Apply the corrections to the data
        if (i_list_err(2,1)==0) && (i_list_err(2,2 ==0)% 2 valleys present
            DomWalk_vall_exp = error_DomWalk;
            DomWalk_hill_exp = error_DomWalk_flip;
        else
            DomWalk_hill_exp = error_DomWalk;
            DomWalk_vall_exp = error_DomWalk_flip;
        end
    end % for i_err = idx_hivaVio
end % if ~isempty(idx_hivaVio)
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_SPE_GaitCycleIdf
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [DomWalk_d_rslt,max_mean,...
    HillList_1st_pos,HillList_2nd_pos,...
    VallList_1st_pos,VallList_2nd_pos,...
    DomWalk_hill_rslt,DomWalk_vall_rslt] = ...
    func_SPE_GaitCycleIdf(DomWalk_d_exp,DomWalk_hill_pos,...
    DomWalk_vall_pos,min_gaitCylces,FigureOn,i_strend)

% prep
idx_exp = 1:length(DomWalk_d_exp);

 %% 1)  create List
    Hill_List_pre = [DomWalk_hill_pos(~isnan(DomWalk_hill_pos));...
        idx_exp(~isnan(DomWalk_hill_pos))];
    Vall_List_pre = [DomWalk_vall_pos(~isnan(DomWalk_vall_pos));...
        idx_exp(~isnan(DomWalk_vall_pos))];
    %% 2) Devide gait/step
    HillList_1st_preturn = Hill_List_pre(:,1:2:end);
    HillList_2nd_preturn = Hill_List_pre(:,2:2:end);
    VallList_1st_preturn = Vall_List_pre(:,1:2:end);
    VallList_2nd_preturn = Vall_List_pre(:,2:2:end);

    %% ignore episodes without steps
if ~any([length(HillList_1st_preturn),length(HillList_2nd_preturn),...
length(VallList_1st_preturn),length(VallList_2nd_preturn)]<min_gaitCylces)

 %% 3) identify which hill/vall has the higthest mean amplitude (dominance)
    % identify mean an variance
    Calc_MSTD = [ ...
    mean(HillList_1st_preturn(1,:)),std(HillList_1st_preturn(1,:));...
    mean(HillList_2nd_preturn(1,:)),std(HillList_2nd_preturn(1,:));...
    mean(VallList_1st_preturn(1,:))*-1,std(VallList_1st_preturn(1,:));...
    mean(VallList_2nd_preturn(1,:))*-1,std(VallList_2nd_preturn(1,:))];
    % identify who has the higest mean
    [~,max_mean] = max(Calc_MSTD); max_mean = max_mean(1,1);
    %%% PRETURN to PRESORT:
    % turn epiosde data around 0 g so that the dominant spikes are positive
    if (max_mean == 3) || (max_mean == 4)
    DomWalk_d_calc = DomWalk_d_exp*-1;
    DomWalk_hill_calc = DomWalk_vall_pos;
    DomWalk_hill_calc(1,:) = DomWalk_hill_calc(1,:)*-1;
    DomWalk_vall_calc = DomWalk_hill_pos;
    DomWalk_vall_calc(1,:) = DomWalk_vall_calc(1,:)*-1;
    HillList_1st_presor = VallList_1st_preturn;
    HillList_1st_presor(1,:) = HillList_1st_presor(1,:)*-1;
    HillList_2nd_presor = VallList_2nd_preturn;
    HillList_2nd_presor(1,:) = HillList_2nd_presor(1,:)*-1;
    VallList_1st_presor = HillList_1st_preturn;
    VallList_1st_presor(1,:) = VallList_1st_presor(1,:)*-1;
    VallList_2nd_presor = HillList_2nd_preturn;
    VallList_2nd_presor(1,:) = VallList_2nd_presor(1,:)*-1;
    else
    DomWalk_d_calc = DomWalk_d_exp;
    DomWalk_hill_calc = DomWalk_hill_pos;
    DomWalk_vall_calc = DomWalk_vall_pos;
    HillList_1st_presor = HillList_1st_preturn;
    HillList_2nd_presor = HillList_2nd_preturn;
    VallList_1st_presor = VallList_1st_preturn;
    VallList_2nd_presor = VallList_2nd_preturn;
    end

    %%% PRESORT to PREORDER:
    % identify if the first or secound spike in the order is the
    %   dominant one
    if (max_mean == 2) || (max_mean == 4)
    HillList_1st_preorder = HillList_2nd_presor;
    HillList_2nd_preorder = HillList_1st_presor;
    VallList_2nd_preorder = VallList_2nd_presor;
```

```
    VallList_1st_preorder = VallList_1st_presor;
    else
    HillList_1st_preorder = HillList_1st_presor;
    HillList_2nd_preorder = HillList_2nd_presor;
    VallList_1st_preorder = VallList_1st_presor;
    VallList_2nd_preorder = VallList_2nd_presor;
    end

    %%% PREORDER to Final
    % Ensure that the pattern is Hill 1st, Vall 1st, Hill 2nd, Vall 2nd
    % identify loc of first Hill is the first dot
    loc1st_hill1st = HillList_1st_preorder(2,1);
    if any(loc1st_hill1st > [HillList_2nd_preorder(2,1), ...
            VallList_1st_preorder(2,1),VallList_2nd_preorder(2,1)])
    % identify the dots closest to hill1st
    loc_vall1st = VallList_1st_preorder(2,VallList_1st_preorder(2,:) > ...
        loc1st_hill1st); loc_vall1st = loc_vall1st(1,1);
    loc_vall2nd = VallList_2nd_preorder(2,VallList_2nd_preorder(2,:) > ...
        loc1st_hill1st); loc_vall2nd = loc_vall2nd(1,1);
    % identify order
    if loc_vall1st > loc_vall2nd % turn valleys
    HillList_1st_pos = HillList_1st_preorder;
    HillList_2nd_pos = HillList_2nd_preorder;
    VallList_1st_pos = VallList_2nd_preorder;
    VallList_2nd_pos = VallList_1st_preorder;
    else % order was already correct
    HillList_1st_pos = HillList_1st_preorder;
    HillList_2nd_pos = HillList_2nd_preorder;
    VallList_1st_pos = VallList_1st_preorder;
    VallList_2nd_pos = VallList_2nd_preorder;
    end
    else % the order was already correct
    HillList_1st_pos = HillList_1st_preorder;
    HillList_2nd_pos = HillList_2nd_preorder;
    VallList_1st_pos = VallList_1st_preorder;
    VallList_2nd_pos = VallList_2nd_preorder;
    end

    %%% Remove Fizzel
    % identify dots before the first peak
idx_vall1st=find(VallList_1st_pos(2,VallList_1st_pos(2,:)<loc1st_hill1st));
idx_vall2nd=find(VallList_2nd_pos(2,VallList_2nd_pos(2,:)<loc1st_hill1st));
idx_hill2nd=find(HillList_2nd_pos(2,HillList_2nd_pos(2,:)<loc1st_hill1st));
    % remove dots before first peak
    if ~isempty(idx_vall1st); VallList_1st_pos(:,idx_vall1st) = []; end
    if ~isempty(idx_vall2nd); VallList_2nd_pos(:,idx_vall2nd) = []; end
    if ~isempty(idx_hill2nd); HillList_2nd_pos(:,idx_hill2nd) = []; end
    % identify length of steps
    min_lgt_List = min([length(HillList_1st_pos(2,:)), ...
        length(HillList_2nd_pos(2,:)), length(VallList_2nd_pos(2,:)), ...
        length(VallList_2nd_pos(2,:))]);
    HillList_1st_pos = HillList_1st_pos(:,1:min_lgt_List);
    HillList_2nd_pos = HillList_2nd_pos(:,1:min_lgt_List);
    VallList_1st_pos = VallList_1st_pos(:,1:min_lgt_List);
    VallList_2nd_pos = VallList_2nd_pos(:,1:min_lgt_List);

%% clean DomWalk_d_calc to str/end at the first and last 0 point with isnan
    DomWalk_d_rslt = DomWalk_d_calc;
    d_pre = DomWalk_d_calc(1,1:HillList_1st_pos(2,1));
    d_pos = DomWalk_d_calc(1,VallList_2nd_pos(2,end):end);
    idx_dstr = find((d_pre > 0) == 0);
    idx_dend = find((d_pos < 0) == 0);
    if ~isempty(idx_dstr)
        idx_dstr = idx_dstr(end);
        DomWalk_d_rslt(:,1:idx_dstr-1) = nan;
    end
    if ~isempty(idx_dend)
        idx_dend = idx_dend(1)+VallList_2nd_pos(2,end)-1;
        DomWalk_d_rslt(:,idx_dend+1:end) = nan;
    end
```

```
    %% preformat DomWalk_vall
    DomWalk_hill_rslt = nan(1,length(DomWalk_d_rslt));
    DomWalk_hill_rslt(HillList_1st_pos(2,:)) = HillList_1st_pos(1,:);
    DomWalk_hill_rslt(HillList_2nd_pos(2,:)) = HillList_2nd_pos(1,:);
    DomWalk_vall_rslt = nan(1,length(DomWalk_d_rslt));
    DomWalk_vall_rslt(VallList_1st_pos(2,:)) = VallList_1st_pos(1,:);
    DomWalk_vall_rslt(VallList_2nd_pos(2,:)) = VallList_2nd_pos(1,:);


    if FigureOn == 1
        figure('visible','off'); hold on;
        subplot(2,2,1); hold on; title('Preturn: Original Dataset')
            plot(DomWalk_d_exp);
scatter(HillList_1st_preturn(2,:),HillList_1st_preturn(1,:),'k','filled')
scatter(HillList_2nd_preturn(2,:),HillList_2nd_preturn(1,:),'r','filled')
scatter(VallList_1st_preturn(2,:),VallList_1st_preturn(1,:),'k','filled')
scatter(VallList_2nd_preturn(2,:),VallList_2nd_preturn(1,:),'r','filled')
        subplot(2,2,2); hold on; title('Presort: after selecting axis ')
            plot(DomWalk_d_calc);
scatter(HillList_1st_presor(2,:),HillList_1st_presor(1,:),'k','filled')
scatter(HillList_2nd_presor(2,:),HillList_2nd_presor(1,:),'r','filled')
scatter(VallList_1st_presor(2,:),VallList_1st_presor(1,:),'k','filled')
scatter(VallList_2nd_presor(2,:),VallList_2nd_presor(1,:),'r','filled')
        subplot(2,2,3); hold on; title('Preorder: after selecting spike')
            plot(DomWalk_d_calc);
scatter(HillList_1st_preorder(2,:),HillList_1st_preorder(1,:),'k','filled')
scatter(HillList_2nd_preorder(2,:),HillList_2nd_preorder(1,:),'r','filled')
scatter(VallList_1st_preorder(2,:),VallList_1st_preorder(1,:),'k','filled')
scatter(VallList_2nd_preorder(2,:),VallList_2nd_preorder(1,:),'r','filled')
        subplot(2,2,4); hold on; title('POS: after clean up')
            plot(DomWalk_d_rslt);
scatter(HillList_1st_pos(2,:),HillList_1st_pos(1,:),'k','filled')
scatter(HillList_2nd_pos(2,:),HillList_2nd_pos(1,:),'r','filled')
scatter(VallList_1st_pos(2,:),VallList_1st_pos(1,:),'k','filled')
scatter(VallList_2nd_pos(2,:),VallList_2nd_pos(1,:),'r','filled')
        suptitle([  num2str(i_strend(1)) ' to ' num2str(i_strend(2))  ...
            '; MaxMean: ' num2str(max_mean)])
        digetnum = [ '%0' num2str(numel(num2str(6))) 'd\n'];
        saveas(gcf,[ 'IMG_GaitCylcleIdf_' num2str(i_strend(1),digetnum)...
            '_' num2str(i_strend(2),digetnum)  '_MaxMean' ...
            num2str(max_mean) '.tif']); close all
    end
else
    DomWalk_d_rslt = nan(1,length(DomWalk_d_exp(1,:)));
    HillList_1st_pos = [];
    HillList_2nd_pos = [];
    VallList_1st_pos = [];
    VallList_2nd_pos = [];
    DomWalk_hill_rslt = nan(1,length(DomWalk_d_exp(1,:)));
    DomWalk_vall_rslt = nan(1,length(DomWalk_d_exp(1,:)));
    max_mean = [];
end % if ~any([length(HillList_1st_preturn),...] < min_gaitCylces)
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_SPE_CreateLocListTo0_V2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function loclist_Gait_strend = func_SPE_CreateLocListTo0_V2(...
    HillList_1st_loc,ValList_2nd_loc,DomWalk_d_rslt)
loclist_Gait_strend = nan(length(HillList_1st_loc),2);
for i = 1:length(HillList_1st_loc)
    %% identify start of gait episode
    i_d = DomWalk_d_rslt(:,1:HillList_1st_loc(1,i));
    if i > 1; i_d(1:ValList_2nd_loc(1,i-1)-1) = nan; end
    i_str_neg = find(i_d < 0);    i_str_pos = find(i_d > 0);
    %% decission
    Methode_02 = 0; Methode_03 = 0;
    % the graph does not intersect 0 select the
    if (isempty(i_str_neg)) || (isempty(i_str_pos))
% the graph did not intersect 0 (e.g. dom hill is neg) find center between
%   Hill and previous valley use the value of the center as the new 0 value
        if i > 1; Methode_02 = 1; else
            % there is no previous sub Valley
            %   use the first possible value that is not nan
            Methode_03 = 1;                                              end
    else
% the graph intersect 0; find the last time the graph intersect 0 before
%   reaching the dominant hill
        i_str = i_str_pos(:,i_str_pos>i_str_neg(end));
        % problem:  there is a more positive peak between the previous
            %           vall and the current hill
        if ~isempty(i_str)
            i_str = i_str(1,1); % Methode 1
        else
            % repeat methode from above
            if i > 1; Methode_02 = 1; else; Methode_03 = 1; end
        end
    end
% select centre between prev vall and current hill as changepoint
    if Methode_02 == 1
        i_val = DomWalk_d_rslt(:, ValList_2nd_loc(1,i-1));
        i_hil = DomWalk_d_rslt(:,HillList_1st_loc(1,i));
        new_0 = i_hil + (i_val-i_hil)/2;
        i_str_neg = find(i_d < new_0);
        i_str_pos = find(i_d > new_0);
        i_str = i_str_pos(:,i_str_pos>i_str_neg(end));
        if ~isempty(i_str)
            i_str = i_str(1,1);                                      else
% problem: the current hill located below the the previous valley and the
%   graph is tehrefore declining and not incrreasing.
%   Is a massive error but does not happen often,  use the center loc
%   instead of passing a value on the y-axis
            i_str = ceil(mean(find(~isnan(i_d))));                 end;end
    if Methode_03 == 1 % select fist non nan value (if i == 1)
        i_str = find(~isnan(i_d)); i_str = i_str(1,1);               end
    loclist_Gait_strend(i,1) = i_str;

    %% identify end of the episode
    if i > 1; loclist_Gait_strend(i-1,2) = i_str-1; end
    if i == length(HillList_1st_loc)
        i_d =  DomWalk_d_rslt(:,ValList_2nd_loc(end):end);
        i_end = find(i_d > 0);
        if ~isnan(i_end)
            loclist_Gait_strend(i,2) = ValList_2nd_loc(end)+i_end-2;
        else % find last usable datapoint
            i_end = find(~isnan(i_d));
            loclist_Gait_strend(i,2) = ValList_2nd_loc(end)+i_end(end)-1;
    end;end
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% func_SPE_EP
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [WalkEpi_output,VariableNameList] = ....
    func_SPE_EP(WalkEpi_input,DomWalk_d,var_feq)

VariableNameList = {'Epis_allGait_AccRange', 'Epis_allStep_AccForce', ...
'Epis_allStep_AccRange', 'Epis_allStep_TimeSec', 'Epis_allStep_Velocity',...
'Epis_chng0PerSpike', 'Epis_domStep_AccForce','Epis_domStep_AccRange',...
'Epis_domStep_TimeSec', 'Epis_domStep_Velocity', 'Epis_Epi_min',  ...
'Epis_Gait_TimeSec', 'Epis_Gait_Velocity', 'Epis_NSpk', 'Epis_SPE_RateMin',...
'Epis_subStep_AccForce','Epis_subStep_AccRange','Epis_subStep_TimeSec', ...
'Epis_subStep_Velocity', 'Epis_Symmetry_AccForce', 'Epis_Symmetry_AccRange',...
'Epis_Symmetry_TimeSec', 'Epis_Symmetry_Velocity', 'Epis_Variance_AccForce',  ...
'Epis_Variance_AccRange','Epis_Variance_TimeSec','Epis_Variance_Velocity'};

%% prep output
lgt_epi = length(WalkEpi_input);

% maintain Structure
WalkEpi_output.User_Name = []; WalkEpi_output.User_Condition_Patient = [];
WalkEpi_output.User_Condition_OA = []; WalkEpi_output.User_Condition_RA = [];
WalkEpi_output.User_Condition_Gout = [];WalkEpi_output.User_Condition_other = [];
WalkEpi_output.User_Condition_PairID = [];

WalkEpi_output.orientation = nan(1,lgt_epi);
WalkEpi_output.axis  = nan(1,lgt_epi);
WalkEpi_output.datenum = nan(1,lgt_epi); WalkEpi_output.WK = nan(1,lgt_epi);
WalkEpi_output.meanTime = nan(1,lgt_epi); WalkEpi_output.TC = nan(1,lgt_epi);

WalkEpi_output.Epis_NSpk = nan(1,lgt_epi);
WalkEpi_output.Epis_chng0PerSpike = nan(1,lgt_epi);
WalkEpi_output.Epis_Epi_min = nan(1,lgt_epi);
WalkEpi_output.Epis_SPE_RateMin = nan(1,lgt_epi);

WalkEpi_output.Epis_allStep_AccForce = nan(1,lgt_epi);
WalkEpi_output.Epis_allStep_AccRange = nan(1,lgt_epi);
WalkEpi_output.Epis_allStep_TimeSec = nan(1,lgt_epi);
WalkEpi_output.Epis_allStep_Velocity = nan(1,lgt_epi);
WalkEpi_output.Epis_allGait_AccRange = nan(1,lgt_epi);
WalkEpi_output.Epis_Gait_TimeSec = nan(1,lgt_epi);
WalkEpi_output.Epis_Gait_Velocity = nan(1,lgt_epi);

WalkEpi_output.Epis_domStep_AccForce = nan(1,lgt_epi);
WalkEpi_output.Epis_domStep_AccRange = nan(1,lgt_epi);
WalkEpi_output.Epis_domStep_TimeSec = nan(1,lgt_epi);
WalkEpi_output.Epis_domStep_Velocity = nan(1,lgt_epi);
WalkEpi_output.Epis_subStep_AccForce = nan(1,lgt_epi);
WalkEpi_output.Epis_subStep_AccRange = nan(1,lgt_epi);
WalkEpi_output.Epis_subStep_TimeSec = nan(1,lgt_epi);
WalkEpi_output.Epis_subStep_Velocity = nan(1,lgt_epi);

WalkEpi_output.Epis_Symmetry_AccForce = nan(1,lgt_epi);
WalkEpi_output.Epis_Symmetry_AccRange = nan(1,lgt_epi);
WalkEpi_output.Epis_Symmetry_TimeSec = nan(1,lgt_epi);
WalkEpi_output.Epis_Symmetry_Velocity = nan(1,lgt_epi);
WalkEpi_output.Epis_Variance_AccForce = nan(1,lgt_epi);
WalkEpi_output.Epis_Variance_AccRange = nan(1,lgt_epi);
WalkEpi_output.Epis_Variance_TimeSec = nan(1,lgt_epi);
WalkEpi_output.Epis_Variance_Velocity = nan(1,lgt_epi);

for i_epi = 1:lgt_epi
%% Extract data
i_str=WalkEpi_input(i_epi).strend(1);i_end=WalkEpi_input(i_epi).strend(2);
iepi_d=DomWalk_d(:,i_str:i_end);iepi_d(:,WalkEpi_input(i_epi).idx_nan)=nan;
loclist_SPE_strend = WalkEpi_input(i_epi).Spk.locList_Step;
loclist_Gait_strend = WalkEpi_input(i_epi).Spk.locList_Gait;

% output
WalkEpi_output.orientation(1,i_epi) = WalkEpi_input(i_epi).orientation;
```

```
WalkEpi_output.axis(1,i_epi)  = WalkEpi_input(i_epi).axis;
WalkEpi_output.datenum(1,i_epi) = WalkEpi_input(i_epi).t.datenum;
WalkEpi_output.WK(1,i_epi) = WalkEpi_input(i_epi).t.WK;
WalkEpi_output.meanTime(1,i_epi) = WalkEpi_input(i_epi).t.meanTime;
WalkEpi_output.TC(1,i_epi) = WalkEpi_input(i_epi).t.TC;
WalkEpi_output.Epis_NSpk(1,i_epi) = WalkEpi_input(i_epi).Calc_NSpk;

    %% Acceleration measures
    %% Acceleration Force
    % size of acceleation Hills (positive accelearion peaks
WalkEpi_input(i_epi).Calc_domStep_AccForce = ...
    WalkEpi_input(i_epi).Spk.Hill1st(1,:);
WalkEpi_input(i_epi).Calc_subStep_AccForce = ...
    WalkEpi_input(i_epi).Spk.Hill2nd(1,:);
WalkEpi_input(i_epi).Calc_allStep_AccForce = [...
    WalkEpi_input(i_epi).Spk.Hill1st(1,:), ...
    WalkEpi_input(i_epi).Spk.Hill2nd(1,:)];
% output
WalkEpi_output.Epis_allStep_AccForce(1,i_epi) = mean(...
    WalkEpi_input(i_epi).Calc_allStep_AccForce);
WalkEpi_output.Epis_domStep_AccForce(1,i_epi) = mean(...
    WalkEpi_input(i_epi).Calc_domStep_AccForce);
WalkEpi_output.Epis_subStep_AccForce(1,i_epi) = mean(...
    WalkEpi_input(i_epi).Calc_subStep_AccForce);

    %% Acceleration Range
    % steps amplitude: diff between consecuitive hill and vall
WalkEpi_input(i_epi).Calc_domStep_AccRange = ....
    WalkEpi_input(i_epi).Spk.Hill1st(1,:) - ...
    WalkEpi_input(i_epi).Spk.Vall1st(1,:);
WalkEpi_input(i_epi).Calc_subStep_AccRange = ...
    WalkEpi_input(i_epi).Spk.Hill2nd(1,:) - ...
    WalkEpi_input(i_epi).Spk.Vall2nd(1,:);
WalkEpi_input(i_epi).Calc_allStep_AccRange = ...
    [WalkEpi_input(i_epi).Calc_domStep_AccRange, ...
    WalkEpi_input(i_epi).Calc_subStep_AccRange];
    % gait ampltiude: diff bw min and max value of a gaitcycle
    Calc_allGait_Amp = nan(1,length(loclist_Gait_strend));
    for i_g = 1:length(loclist_Gait_strend)
i_gd = iepi_d(:,loclist_Gait_strend(i_g,1):loclist_Gait_strend(i_g,2));
    Calc_allGait_Amp(1,i_g) = max(i_gd)-min(i_gd);
    end
% output
WalkEpi_output.Epis_allStep_AccRange(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_allStep_AccRange);
WalkEpi_output.Epis_allGait_AccRange(1,i_epi) = mean(Calc_allGait_Amp);
WalkEpi_output.Epis_domStep_AccRange(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_domStep_AccRange);
WalkEpi_output.Epis_subStep_AccRange(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_subStep_AccRange);

    %% Velocity (g/sec)
    % area under the curve, but we added the negative and positive
    % Velocity together
    Calc_SPE_Volume = nan(1,length(loclist_SPE_strend));
    for i_g = 1:length(loclist_SPE_strend)
i_gd = iepi_d(:,loclist_SPE_strend(i_g,1):loclist_SPE_strend(i_g,2));
Calc_SPE_Volume(1,i_g) = polyarea([1 1:length(i_gd) length(i_gd) 1]/50,...
    [0 abs(i_gd) 0 0]);
    end
WalkEpi_input(i_epi).Calc_allStep_Velocity = Calc_SPE_Volume;
WalkEpi_input(i_epi).Calc_domStep_Velocity = Calc_SPE_Volume(1:2:end);
WalkEpi_input(i_epi).Calc_subStep_Velocity = Calc_SPE_Volume(2:2:end);
    % gait volume
    Calc_Gait_Volume = nan(1,length(loclist_Gait_strend));
    for i_g = 1:length(loclist_Gait_strend)
i_gd = iepi_d(:,loclist_Gait_strend(i_g,1):loclist_Gait_strend(i_g,2));
Calc_Gait_Volume(1,i_g) = polyarea([1 1:length(i_gd) length(i_gd) ...
        1]/50,[0 abs(i_gd) 0 0]);
    end
```

```
% output
WalkEpi_output.Epis_allStep_Velocity(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_allStep_Velocity);
WalkEpi_output.Epis_Gait_Velocity(1,i_epi) = mean(Calc_Gait_Volume);
WalkEpi_output.Epis_domStep_Velocity(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_domStep_Velocity);
WalkEpi_output.Epis_subStep_Velocity(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_subStep_Velocity);

%% Temporal Descriptors
%% Gait cycle time/Stride and Step Time
WalkEpi_input(i_epi).Calc_allStep_TimeSec = ...
    ((loclist_SPE_strend(:,2)-loclist_SPE_strend(:,1))')/var_feq;
WalkEpi_input(i_epi).Calc_domStep_TimeSec = ...
    WalkEpi_input(i_epi).Calc_allStep_TimeSec(1:2:end);
WalkEpi_input(i_epi).Calc_subStep_TimeSec = ...
    WalkEpi_input(i_epi).Calc_allStep_TimeSec(2:2:end);

% output
WalkEpi_output.Epis_allStep_TimeSec(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_allStep_TimeSec);
WalkEpi_output.Epis_Gait_TimeSec(1,i_epi) = ...
    mean(((loclist_Gait_strend(:,2)-loclist_Gait_strend(:,1))')/var_feq);
WalkEpi_output.Epis_domStep_TimeSec(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_domStep_TimeSec);
WalkEpi_output.Epis_subStep_TimeSec(1,i_epi) = ...
    mean(WalkEpi_input(i_epi).Calc_subStep_TimeSec);

%% Cadence/Frequency/Step rate
WalkEpi_input(i_epi).Calc_Epi_min = (WalkEpi_input(i_epi).strend(2) - ...
    WalkEpi_input(i_epi).strend(1))/var_feq/60;
% output
WalkEpi_output.Epis_Epi_min(1,i_epi) = WalkEpi_input(i_epi).Calc_Epi_min;
WalkEpi_output.Epis_SPE_RateMin(1,i_epi) = ...
    WalkEpi_input(i_epi).Calc_NSpk/WalkEpi_input(i_epi).Calc_Epi_min;

%% Gait Quality Mesures
%% Symmetry
% output
WalkEpi_output.Epis_Symmetry_AccForce(1,i_epi) = ...
    mean(diff([WalkEpi_input(i_epi).Calc_subStep_AccForce; ...
    WalkEpi_input(i_epi).Calc_domStep_AccForce]));
WalkEpi_output.Epis_Symmetry_AccRange(1,i_epi) = ...
    mean(diff([WalkEpi_input(i_epi).Calc_subStep_AccRange; ...
    WalkEpi_input(i_epi).Calc_domStep_AccRange]));
WalkEpi_output.Epis_Symmetry_TimeSec(1,i_epi) = ...
    mean(diff([WalkEpi_input(i_epi).Calc_subStep_TimeSec; ...
    WalkEpi_input(i_epi).Calc_domStep_TimeSec]));
WalkEpi_output.Epis_Symmetry_Velocity(1,i_epi) = ...
    mean(diff([WalkEpi_input(i_epi).Calc_subStep_Velocity; ...
    WalkEpi_input(i_epi).Calc_domStep_Velocity]));

%% Variance
% output
WalkEpi_output.Epis_Variance_AccForce(1,i_epi) = ...
    std(WalkEpi_input(i_epi).Calc_allStep_AccForce);
WalkEpi_output.Epis_Variance_AccRange(1,i_epi) = ...
    std(WalkEpi_input(i_epi).Calc_allStep_AccRange);
WalkEpi_output.Epis_Variance_TimeSec(1,i_epi) = ...
    std(WalkEpi_input(i_epi).Calc_allStep_TimeSec);
WalkEpi_output.Epis_Variance_Velocity(1,i_epi) = ...
    std(WalkEpi_input(i_epi).Calc_allStep_Velocity);

%% Graph passes 0 per Spike
% for every spike (hill vall 1st 2nd) chould be one change in orientation
i_spikes = WalkEpi_input(i_epi).Calc_NSpk*2;
i_chgn_d = (iepi_d < 0);
chng_d = length(find(diff(i_chgn_d) ~= 0));
% output
WalkEpi_output.Epis_chng0PerSpike(1,i_epi) = chng_d/i_spikes; end; end
```