Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science

The Use of Data Science for Sports Analysis Purposes

A thesis submitted to the
Delft Institute of Applied Mathematics
In partial fulfillment of the requirements

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

Delft Institute of Applied Mathematics
Specialisation Stochastics

MARIEKE DE VRIES
4280717

Delft, the Netherlands
October 2019

MSc thesis APPLIED MATHEMATICS

"The Use of Data Science for Sports Analysis Purposes"

MAAIKE MARIEKE DE VRIES

Delft University of Technology

Supervisor

Dr. J. Söhl

Other thesis committee members

Prof. dr. ir. G. Jongbloed

Dr.ir. G.F. Nane

October, 2019

Delft

Abstract

This thesis is dedicated to the application of data science to sports data. The research for this thesis is part of a bigger project on injury prevention and sport performance called Citius Altius Sanius (CAS). Two data sets from two different projects within CAS are analysed, with two different goals; one focusses on sports injury prevention in soccer, the other on performance prediction in baseball.

First we analyse a data set on acceleration during exercise from project P6, generated while testing a prototype of wearable sensor trousers during soccer drills. The aim of P6 is to design special leg wear with wearable sensors in order to gain more knowledge on hamstring injuries. Therefore, an algorithm needs to be developed to identify the intensity of certain movements using sensor data. Features were extracted from the acceleration data in order to classify the intensity. Four methods are then tested on the data, of which the decision tree seems to produce the best results. Analysis showed that this model seemed to be able to predict low intensity well (99.1% accuracy), although it struggles significantly more with medium and high intensity exercise (75.5%).

The second data set covered the growth in throwing speed of a group of young baseball athletes between the ages of 12 and 18. The aim of the research was to identify a common growth curve for throwing speed of pitchers during adolescence and provide personalised growth curve models. A mixed effects or multilevel design was chosen to model the growth in throwing speed, due to its ability to model the hierarchical nature of the longitudinal data. After analysing the data set and covariates, we found we could reduce the number of predictors, and thus the cost of collecting data. Furthermore, it is possible to predict throwing speed on a personal level using only age and one measurement on the predictors and throwing speed, although predictions are improved when more measurements are available.

The results of this research can be implemented in the projects, although some complications and opportunities for improvement still exist. Recommendations for future research have therefore been discussed.

Preface

Before you lies a Master's thesis on the use of data science for sports analysis purposes. It combines three of my favourite things: data science, applications of maths to real-life problems and charts with fun colours. I hope you have as much fun reading this as I have had creating the graphs.

This is one of the most demanding projects I have ever undertaken, only surpassed by a two hour kayak trip in Belgium; I have a terribly bad back, and lost my friends within ten minutes of paddling. It has turned me off kayaking for life.

Whereas on the kayak trip I was left to my own devices, I fortunately have had a lot of assistance during this project. I would not have been able to write this preface if it were not for a lot of people that helped and guided me throughout the years.

First of all my gratitude goes out to Dr. Jakob Söhl for his supervision and support during this project. I would like to thank Dr. ir. Tina Nane and Prof. dr. ir. Geurt Jongbloed for participating in my thesis committee. I would also like to thank Prof. dr. Frank Redig for his patience and supervision during my previous project, Dr. ir. Martin van Gijzen and Leonie Boortman for their counsel and lending a sympathetic ear, and Prof. dr. ir. Geurt Jongbloed again for always being willing to guide me on my way of becoming a statistician.

Furthermore, I am grateful for the support of my friends and family during the good times and the not-so-good times. Lastly, I could not forget the people to whom I am most grateful: my parents, without whose support, counsel and love I would surely have gone mad by now.

Marieke de Vries

Delft, October 8, 2019

Contents

List of Figures

List of Tables

1 Introduction

Over the last couple of years, interest in data collection and analysis has increased across various disciplines such as business, management and government [19]. Given this rise in interest across the board, it seems to be expected that sports organisations followed suit. However, despite how new this fascination with data seems to be, the reality is that sport and data analysis have been interlinked for far longer than most people realise.

Perhaps the most famous use of data analysis in sports is the 2011 movie *Moneyball* [48], in which an accountant uses a statistical approach for scouting and analysing players for his baseball team, and leads them to victory despite their small budget. In baseball, such analyses are so widely used that they have their own name: sabermetrics. But, despite how recent the increase in interest may seem, sabermetrics goes back to the creation of the box score by Henry Chadwick in 1858 [38].

While baseball lends itself well to analysing due to the structure of the game, much better than most team sports, this does not mean that the use of data analysis is restricted to baseball alone. Even in more chaotic team sports such as soccer, calculating ball possession, the percentage of successful passes and the number of attempted shots at the goal are standard for most professional games. Additionally, during games players are often tracked across the field to quantify their activity in terms of distance.

Data science has more to offer to sports than just game analytics, however. More and more research is targeted to improving performance; for example in the field of cycling, where teams decide their participation in cycling competitions on the basis of data analysis [5], or the inclusion of sensors on skates for competitive speed skating, in order to give real-time feedback to an athlete and improve their form [49]. New research not only focusses on performance improvement, but injury prevention as well; recent studies have shown that the analysis of data collected by sensors, questionnaires and tests, can help identify risk factors or athletes with an elevated risk for injury [53, 47].

This research focusses on how data science can aid sports professionals in these last two categories; the research conducted for this thesis is part of a bigger project within the Citius Altius Sanius (CAS) program, which aims to reduce sports injuries and improve performance in a number of different sports.

1.1 Aim of research

In recent years, there have been millions of injuries sustained in the Netherlands due to sports activities - roughly 4.5 million each year. Some quite extreme: in 2016, 121.000 people required emergency care due to sports injuries. In total, they cost 5 million euros in direct medical costs. "Half of these injuries could potentially be prevented through effective support and self-management" [11]. The aim of the CAS project is thus to "stimulate people at all performance levels to engage in and sustain physical activity through sports and fitness, improve their performance and prevent injuries by providing informative and motivating information using advanced sensor and data science techniques." [33].

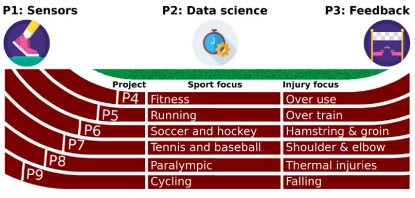


Figure 1.1: Structure of the CAS project. The three fundamental research lines (sensors, data science and feedback) are combined with the six applied research lines [11].

The CAS project consists of nine projects, as is depicted
in Figure 1.1. CAS focusses on six different sports and types of injury, all of which are contained within one project. Within each project, the use of data science is one of the fundamental research lines, alongside sensors and feedback. Here data science is applied in order to "relate the load to injury mechanisms, and provide an individual training advice to stimulate the athlete and prevent injuries, or return to sports and exercise quicker." [33].

The CAS research project started in April 2018 [12]. In the third quarter of 2018, the first prototypes had been built and data had been generated, after which some projects contacted the Statistics department of the Delft Institute of Applied Mathematics in order to gain more insight into the generated data. This thesis will cover the analysis of some of this data.

The research done for this thesis will be connected with two projects within CAS: project P6, which aims to reduce hamstring injuries in soccer and field hockey by employing smart sensor shorts, and project P7, which focusses on shoulder and elbow injuries in tennis and baseball due to bad coordination.

The data supplied by project P6 was generated by a prototype of the sensor shorts that is being developed by Industrial Design Engineering. In order to gain more insight into hamstring injuries, more data on muscle load during exercise needs to be generated and analysed. We therefore will aim to develop a method that is able to detect medium and high intensity activity. The data supplied by project P7 originates from project Fastball, a previous research, and was provided for analysis. The data covers growth in throwing speed in youth baseball pitchers. The aim of the data analysis is to develop a method for predicting throwing speed over time.

1.2 Structure of thesis

As mentioned previously, two separate subjects within the CAS project are covered in this thesis: injury prevention in soccer and performance prediction in baseball. In Chapter 2 these projects will be introduced in detail.

After this the thesis is seperated into two parts: first we will deal with the soccer project, where we will look at sensor data in order to estimate intensity as a proxy for muscle load. Chapter 3 gives a background on some of the literature dealing with classification in sports using sensor data. The next two chapters are dedicated to the exploration and classification of the data itself, where Chapter 4 goes into how measures for classification were extracted and Chapter 5 provides some background on the methods used for classification and their results.

The next chapters are dedicated to the research that has been conducted for throwing speed prediction for youth in baseball. For this research, mixed effects models were used, and Chapter 6 gives a concise overview on the literature in this area. Chapters 7 through 9 focus on the data and prediction itself. Chapter 7 deals with the data, Chapter 8 explores the different models that can be fitted to this data and Chapter 9 provides the final model. Chapter 10 reflects on this research and discusses problems and recommendations for future research.

2 Introduction to the CAS projects

The research conducted for this thesis is part of a bigger project within the Citius Altius Sanius (CAS) program, which aims to reduce sports injuries and improve performance for athletes in a number of different sports. There are nine different projects within the CAS program, most of which Delft University of Technology is involved in. Out of all those projects, this thesis will be dealing with data from the projects P6, where the focus is on team sports such as soccer, and P7, which focusses on coordinative sports such as baseball. Although the aims of these projects are very similar, the data and accompanying analyses are not. In this chapter we will look at the different projects, discuss the related data sets, and specify what the aim of the analysis is for each individual project.

2.1 Project P6

Project P6 focusses on soccer and field hockey, and more specifically on hamstring injuries. These injuries are often due to physical overload during matches, but little is known about the factors that contribute to such overload. In order to gain more knowledge about the stress put on the hamstring when playing soccer, a special leg wear with wearable sensors is being developed for use during exercising. The data that was supplied for analysis was generated from an early prototype of such a garment.

2.1.1 Overview of project P6

The study consists of five participants who performed six soccer-related exercises consecutively. These soccer exercises vary in difficulty and intensity, and the movement of the participants is recorded by a set of six sensors that are placed on the body.

There is one global sensor called the LPM sensor, which measures, among other things, the global location and acceleration. This sensor is placed on the back of the person performing the exercise. A selection of local sensors is used to calculate the acceleration at precise points on the body. In this case, the sensors are placed on the pelvis area (middle lower back), the thigh area on both sides and the leg or shin area. These local sensors give higher precision of the intensity of an exercise [44] when compared to the global LPM system.

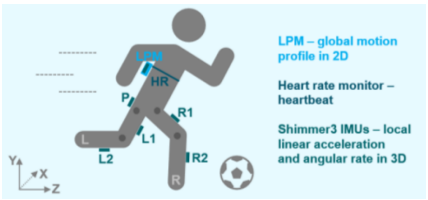


Figure 2.1: Placement of the sensors [44]

Not much is known about the impact that certain exercises have on specific muscles. There are multiple hypotheses about why athletes are injured while playing sports. One of the hypotheses asserts that sports injuries are related to muscle fatigue, caused by playing at higher intensity over a longer period. The other hypothesis places more emphasis on peaks of high intensity. Although both hypotheses focus on moments of high intensity, the first pays more attention to the total length of the intense period, while the latter focuses more on the frequency of these periods. The aim of the research conducted for project P6 is thus to use the local sensors to gain a better understanding of the impact of certain movements and exercises.

2.1.2 Overview of data for project P6

The data from these local sensors are generated by an accelerometer, a gyroscope and a magnetometer with a frequency of 200 Hz. Of these three instruments, only the accelerometer was useful for analysis purposes, as the other two were not properly set up to deal with the intense movement of the sensors during the exercises. The accelerometer describes the localised acceleration in a three-dimensional space.

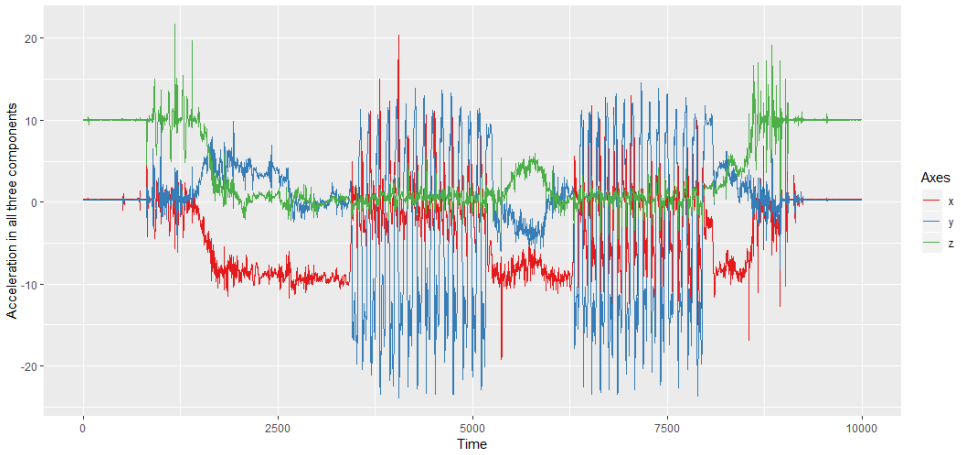


Figure 2.2: Example of acceleration data on left thigh over a time period of roughly 50 seconds, acceleration in m/s^2, time in 5 millisecond (ms)

High peaks correspond with high acceleration - sudden speeding up during an exercise, or kicking the ball. A dip in the acceleration is reflective of a significant decrease in speed or a sudden stop in movement. One way to summarise this data [41] is by calculating the size of the vector (x, y, z) at time t by

$$a_1(t) = \sqrt{x^2(t) + y^2(t) + z^2(t)}, \tag{2.1}$$

which gives a comprehensive overview of the acceleration in all three components. This result is plotted below for the same time period as that given in Figure 2.2.

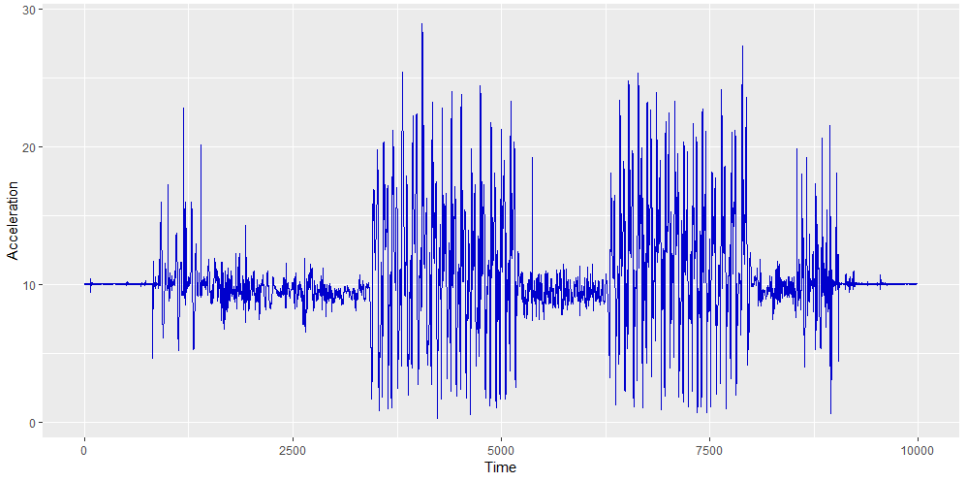


Figure 2.3: Acceleration data on left thigh over 50 seconds, size of acceleration in m/s^2, time in 5 millisecond (ms)

Figure 2.3 illustrates the first problem with the data: the acceleration is not centered around zero. This is because gravity needs to be taken into account for the acceleration, which exerts a constant force on the sensors. The data is therefore centered around 9.81 m/s^2, the standard acceleration due to gravity. Usually this gravitational force can be filtered out of the data as it only works in one of the three directions in which the sensor measures. However, due to the rotation of the sensor during the exercise, the gyroscope is needed to properly analyse the acceleration in the three directions. In the current data set, the gyroscope did not measure precisely enough; therefore the gravitational force cannot be removed before summarising the data.

2.1.3 Mathematical research

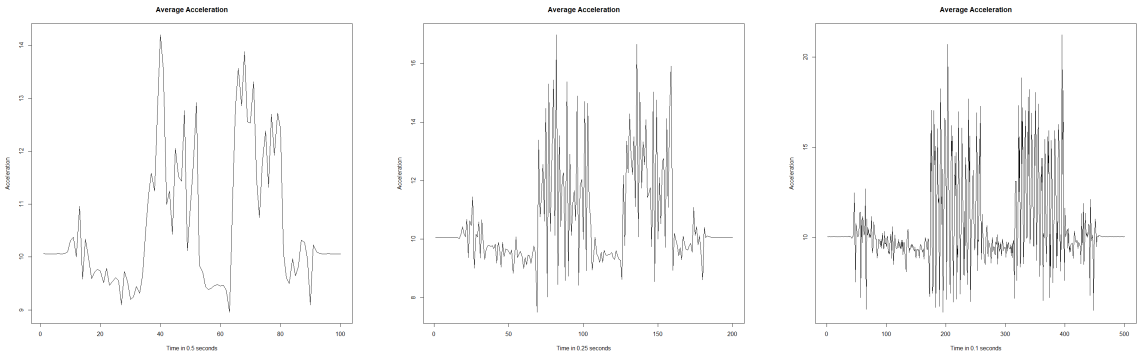
Two aspects of the data analysis are of interest:

- Is it possible to predict the intensity of an exercise given the acceleration data from the localised sensors?
- Is it possible to give a real-time summary of the generated data?

2.1.3.1 Prediction of intensity

As discussed above, there are two hypotheses about the occurrence of injuries when exercising. In order to predict intensity, we will first need to construct a method that converts acceleration data into intensity data by classification. As the sensors generate much data, it is necessary to reduce the information obtained from the sensors to be more manageable. The first step would be to find a suitable features that can be extracted from the data.

One way to summarize the data is by averaging the acceleration over time steps Δt. Choosing a time step needs to be done carefully, as a large time step could flatten out the data in such a way that moments of high intensity would no longer be visible. Below the data from Figure 2.3 is summarized using different time steps.



(a) $\Delta t = 0.5s$ (b) $\Delta t = 0.25s$ (c) $\Delta t = 0.1s$

Figure 2.4: Averaged acceleration on the left thigh over 50 seconds

While Figure 2.4c shows perhaps too many peaks to properly analyse the movement, Figure 2.4a leaves out the highest peak visible in Figure 2.4c. Finding a proper time step is therefore a delicate balance between visibility and not removing too much detail from the data. In Rogers et al. [41] several other methods are discussed for analysing and summarizing data gathered from accelerometers, and Bonomi et al. [4] discuss possible features that can be extracted from

the data. These methods, and others, will be considered in an effort to find a suitable method to summarise the data. This will then be used to classify the intensity of the exercises.

2.1.3.2 Variable selection

In order to give real-time feedback to coaches or medical staff, it is useful to keep the amount of information in the feedback as low as possible. However, the feedback does need to be comprehensive enough to be able to make good decisions about whether to continue a certain exercise or to substitute a player during a game.

There are two ways in which the data can be displayed: directly from the sensor (e.g. the raw acceleration data) or summarised. For the summarised measurements, the results of the classification research could be used. However, as this is not a direct measure, there might be a delay in the feedback. Therefore, it can be useful to look at the many variables that are directly available and make a selection.

2.2 Project P7

Project P7 focusses on coordinative sports such as baseball and tennis. With these sports there are relatively many upper extremity injuries and so again sensors are deployed to gain more knowledge about how and why such injuries occur. However, unlike the soccer data, the data that was supplied for analysis by the members of the project team was not generated as part of the CAS project. Instead, the data comes from a longitudinal study on baseball performance that was conducted to gain more knowledge about growth patterns in throwing speeds and the effect of injury on these growth patterns. Therefore, both the data and the way in which it can be analysed differs significantly from project P6.

2.2.1 Overview of project Fastball

Project Fastball is a longitudinal study on the throwing speed of pitchers of the Dutch Youth Baseball team. Every six months over a period of three years, all pitchers in the under-18 group were asked to perform a series of physical tests and were interviewed to determine whether they had sustained injuries during the last six months. In total, this resulted in six separate measurements of a group of 125 pitchers, although for many of these pitchers less than six measurements were available. This is mostly due to the nature of the study: some pitchers entered the study later due to not being old or good enough at the start of the study, while yet others withdrew from the study before it had ended due to being too old for the under-18 group, or not performing well enough to continue for the national team.

2.2.2 Overview of the data for project Fastball

The main interest behind this research is the ball throwing speed. The athletes were asked to pitch ten times while the speed of the ball was measured. The resulting average was then calculated; this measure became our main response variable. Alongside these results, the following variables were also measured:

- Length and weight of the athlete
- The force of the external rotation (ER) of the throwing arm
- The force of the internal rotation (IR) of the throwing arm
- Range of motion of the external rotation (ER) of the shoulder

- Range of motion of the internal rotation (IR) of the shoulder
- Whether the athletes were injured within the last 6 months in the shoulder
- Whether the athletes were injured within the last 6 months in the elbow

The age of the participants, along with the year when they started pitching, is also known.

2.2.3 Mathematical research

The main research question for project Fastball is whether we can fit a model to predict throwing speed using the variables measured during the study. However, as discussed in Section 2.2.1, there is the issue of missing data to take into account. This problem needs to be considered first, in order to produce a useful model later.

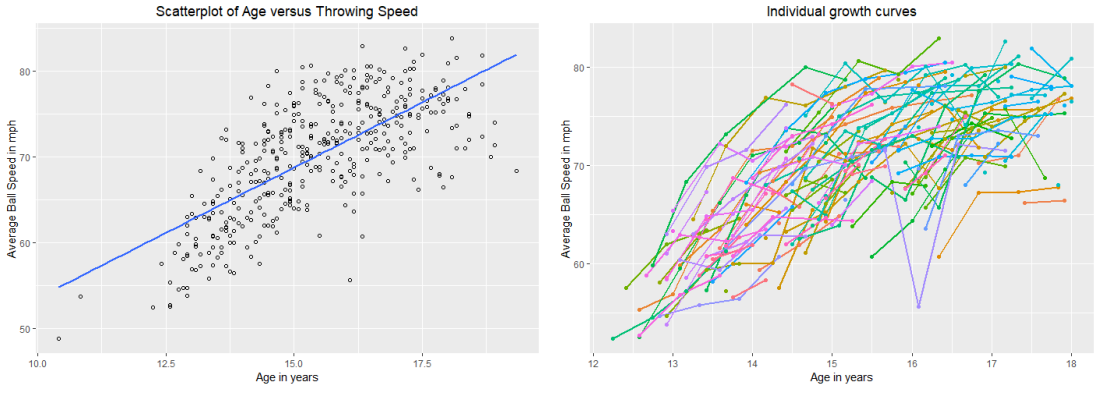
2.2.3.1 Missing data

As discussed in Section 2.2.1, the data is missing in most cases due to the athletes being too old or too young to participate in the study. In some cases the pitchers did not perform well enough to stay in the national team, and were thus missing from the study as well. In both cases, it can be argued that the missing data is *missing not at random* (MNAR). In cases where age plays a role, this is quite obvious. However, due to the nature of the study, pitchers who perform poorly on the team will also perform poorly in the strength tests.

Besides the observations of the participants that seem to be MNAR in our data set, there are also missing values such as height and weight measurements or ball speeds while yet other measurements are recorded. These values seem to be missing at random, although this is hard to verify as the reason for the missingness is unknown.

2.2.3.2 Function estimation

After exploring the data, it seems that there is a clear relationship between age and throwing speed. Figure 2.5a shows a scatter plot where the throwing speed is plotted against the age, and Figure 2.5b shows the growth curve for the individual pitchers:



(a) Scatter plot with linear regression line (b) Growth curves for individual pitchers

Figure 2.5: Relationship between pitchers' age and throwing speed in mph

As is visible in Figure 2.5a, the linear regression line is not a good fit for the data. The scatterplot instead suggests a non-linear trend. We will first try to estimate the relationship between age and speed, and then we shall try to incorporate the other measures such as length, weight, and

internal or external rotation. If we look again at the scatter plot in Figure 2.5a but plot a smoothed fit curve instead of a linear regression line, the result is as follows:

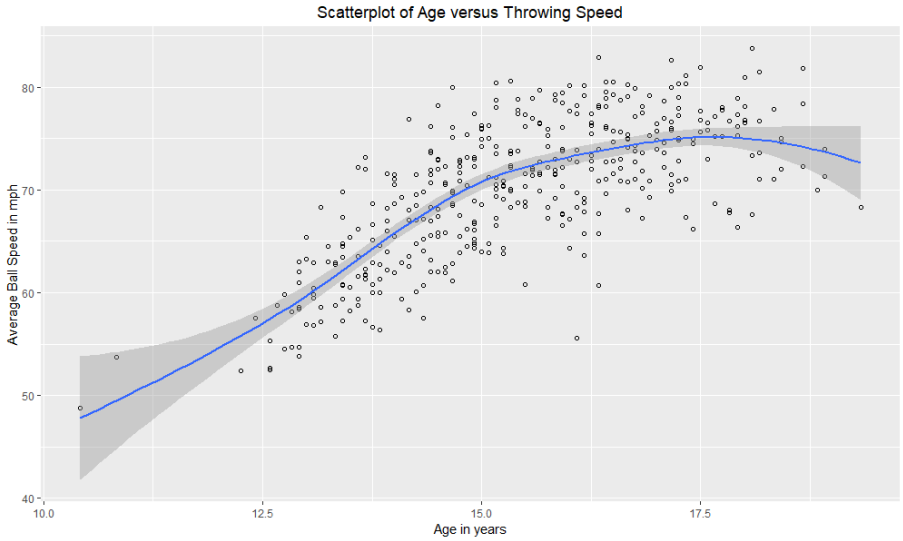


Figure 2.6: Scatter plot of throwing speed and age with Local Polynomial Regression Fitting (loess) line

This plot includes data on pitchers that are outside of the scope of our research, because they are either too young (i.e. younger than twelve) or too old to be a part of the national under-18 team. Even when excluding the observations for pitchers that were older than 18 or younger than 12, the same trend is visible.

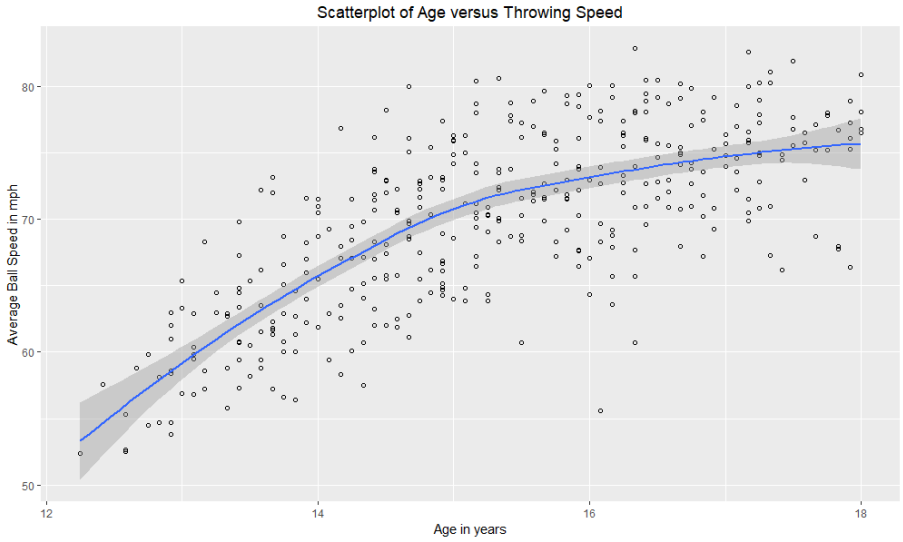


Figure 2.7: Scatter plot of throwing speed and age between 12 and 18 with Local Polynomial Regression Fitting (loess) line

It is clear that a linear model might not be sufficient for predicting throwing speed. We therefore propose using models that are specifically made to deal with longitudinal data, such as mixed

effects or multilevel models. Multilevel models are an effective way to model data that is hierarchically structured: they combine the information at different levels and thus handle data at multiple levels more effectively. Chapter 6 delves into the theory and literature behind this model.

In our case, we can structure the data in such a way that we get a linear or non-linear 2-level model, meaning it models the data by using a hierarchy with two levels. This would mean the individual observations of each pitcher would be on the lower level, and the general growth curve per individual on the second level. Using this method, we hope to obtain a model that uses both the individual growth and the general shape of the growth curves in a meaningful way.

Soccer

3 Classification of intensity during sports using accelerometers

Injuries are, unfortunately, an unavoidable side effect of sports, although some athletes are more at risk than others depending on the sport. Sports where high-speed sprinting or multidirectional acceleration are common, such as soccer, have for example a high prevalence of hamstring injuries [15]. Injury prevention is beneficial for multiple reasons. First and foremost for health reasons; injuries can be debilitating and would preferably be avoided by the athlete. Furthermore, they have long-lasting effects: most athletes never make a full recovery, meaning they lose some of the functional capabilities they had pre-injury. Injuries can in general have a remaining negative effect on health-related quality of life [53].

Another reason is economical in nature: injuries in professional sports lead to absence and increased costs. In soccer specifically, more than 30% of all injuries and a quarter of injury absence are caused by muscle injury; 90% of those injuries occur in the muscles of the lower extremities, the vast majority of which occur in the hamstring. On average a player sustains 0.6 muscle injuries per season; for a professional soccer team of 25 players, 80 football days are lost to injuries each season [14]. Financially, the costs of an injured player for elite soccer teams averages about half a million euros per month [13].

Only a small number of injuries in soccer occur due to contact or foul play: roughly 5% [14]. The other 95% of injuries are therefore somewhat preventable, and are worth investigating to reduce their prevalence.

3.1 Why measure intensity?

Of all soccer injuries, two-thirds of them occur due to acute indirect trauma, the others have a gradual onset [14]. Fatigue could be an important factor for muscle strains, considering injuries are predominantly sustained at the end of training sessions and matches [15, 55]. Furthermore, the explosive nature of some of the movements also cause muscle strain and result in injuries [44].

In both cases, understanding the exact amount and intensity of muscle load can aid athletes, trainers and physical therapists in injury prevention. Polglaze et al. [36] stress the utility of understanding the distribution of player load, and therefore energetic demands, during matches in order to "provide appropriate conditioning programs tailored to the needs of the sport and the physiological status of the individual player". Furthermore, the information could be used to detect fatigue and determine substitution schedules.

3.2 Why classify intensity via sensors?

There are many ways to classify intensity, although not all are effective. Methods such as tracking speed, for example by GPS, are not appropriate for classification of team sport activity when the sport is characterized by continual changes in speed and direction. Even when looking at personalised speed thresholds over general methods, this method fails to take acceleration into account, which is more demanding [36]. Furthermore, Schotel [44] shows that methods which collect data on a global level are less accurate compared to methods that use local data from specific parts of the body, and often underestimate the experienced load.

Research in the field of sensor-based classification of soccer related activities is sparse: to date, only Schuldhaus et al.'s work [45] on a sensor-based algorithm for shot/pass classification has been developed. However, a meta-review by Chambers et al. [6] in 2015 showed that detection

of sport-specific movemements by using wearable microsensors is not only possible but also effective for many different sports. They concluded that sensors are a useful tool and are capable of "quantifying sporting demands that other monitoring technologies may not detect".

Furthermore, sensor-based recognition methods do not require additional cameras or sensors to be installed, have more freedom in how and where an athlete can be monitored (both indoors and outdoors) and can detect smaller movements if necessary [57]. As noted by Schuldhaus et al. [45], sensors thus offer a lower-cost alternative to current methods such as video or computerized technology, which are unavailable to most teams.

3.3 How is sensor-data classified?

Although many studies on activity recognition by sensors in the past focussed on single accelerometers, the popularity of multiple motion sensors is growing. Classification of sensor data is usually based on two parts: extracting features for classification at the sensor level, and a classifying method at the "server level" [57].

As for the first step in activity recognition, there are three ways in which features are usually extracted from the sensor data [57]. Firstly, features can be extracted based on statistics of the acceleration, such as the maximum and minimum or the variance. Another popular method focusses on the frequency domain and is based on fixed filter banks such as fast Fourier transform (FFT), which is an algorithm that computes the discrete Fourier transform of a sequence, and wavelets. Lastly, methods such as principal or independent component analysis (PCA/ICA) can be used as well [31]. The signal from the sensor is then classified using features of the acceleration in a set time frame.

There are many classification methods used when classifying accelerometer data: decision trees, neural networks, k-nearest neighbours, Bayesian classifiers and hidden Markov models, to name a few [4]. For example k-nearest neighbours has been popular due to the simplicity of the algorithm, although more complex methods such as hidden Markov models and decision trees have gained favour as well [57].

3.4 Goal of the research

Although much research on the use of accelerometers in activity detection has been performed already, most of it is focussed on the detection of specific sports-related activities. There is a lack in literature on the use of sensors for classifying intensity, especially in the field of team sports such as soccer. Therefore this research aims to develop a method suitable for intensity recognition during soccer using sensors.

4 Classification of accelerometer data

The sensor data that is analysed in this research was collected by Schotel [44] to compare the measurements from global sensors to local sensors, and show that local sensors are more accurate in predicting experienced load than global sensors. Our analysis will therefore focus on the data from the local sensors, placed on the shins, thighs and middle lower back.

4.1 A brief overview of the data

The acceleration data is briefly described in Section 2.1.2. In this section we will delve a bit deeper into the specifics and explain some of the difficulties with analysing the data.

4.1.1 Experiment set-up

The data was generated by five participants, each participating in ten separate drills which are described in Figure 4.1. After each drill the participant was asked to rate the intensity of the exercise on a scale of 1-10. The level of fitness was not equal amongst participants, but all were in good health. The drills are mostly ascending in intensity, which is reflected by the increasing heart rate and experienced intensity amongst participants.

Videos of the drills were recording alongside sensors, although those were not available for this research due to the rules regarding the privacy of the participants. The data from the wearable sensors is collected and converted to a Matlab file, containing the values in the magnetometer and gyroscope in each axis and a time stamp. It also contains the acceleration in all three axes, twice: once measured with a wide range, and once with a low noise (but smaller range). For this research, the wide range acceleration was used, despite the fact that there was more noise. This is because the low noise acceleration could not fully capture the shifts in acceleration.

See Appendix A for a full overview of the experimental protocol [44].

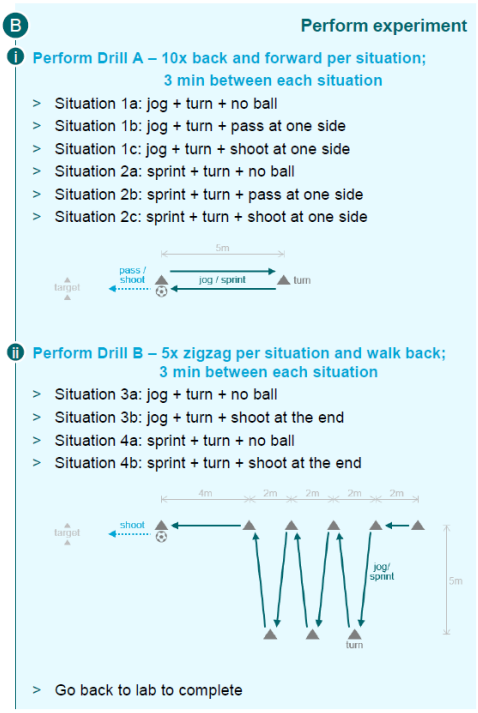


Figure 4.1: Experiment drill protocol

4.1.2 Extraction of the acceleration signal

The sensors used in the research measure acceleration in three dimensions: x (sideways or medial-lateral), y (forward or anterior-posterior) and z (vertical). Due to gravity, there is a constant force working in the z direction, and so the acceleration in the z direction in this sensor is 9.81 m^2/s in rest. However, once the sensor rotates, the direction in which the sensor experiences gravity also changes. This is due to the way in which the sensor measures the axis: they are not constant in relation to the surface, but relative to the sensor itself. This means that the z-axis is no longer perpendicular to the earths surface.

Usually this shift in direction for the axis can be solved in the pre-processing stage of the data analysis by using the gyroscope and magnetometer, which measure the internal rotation. However, in this case the gyroscope was not properly adjusted before the drills were performed, therefore the option of separating axes was not available to us. This is unfortunate, because the separate axes are now meaningless and thus the information that can be learned from the sensors is diminished.

In order to make sense of the acceleration data, we combine the three streams into one; we do this by computing the size of the acceleration vector, meaning

$$a(t) = \sqrt{x^2(t) + y^2(t) + z^2(t)}. \tag{4.1}$$

This measure is used by some sporting microtechnology companies to describe player load [6]. It is however not ideal: the resulting vector is positive, and therefore it is hard to separate acceleration from deceleration, which both have a different impact on the muscle load. However, it does remove the issue of separating the directions within the acceleration.

4.1.3 The data

Each participant was continuously recorded from start to finish, including the time it took to fasten the sensors to the clothing. For each participant the recording took between 70 and 85 minutes. When we calculate this acceleration over the duration of the experiment, we find the following patterns:

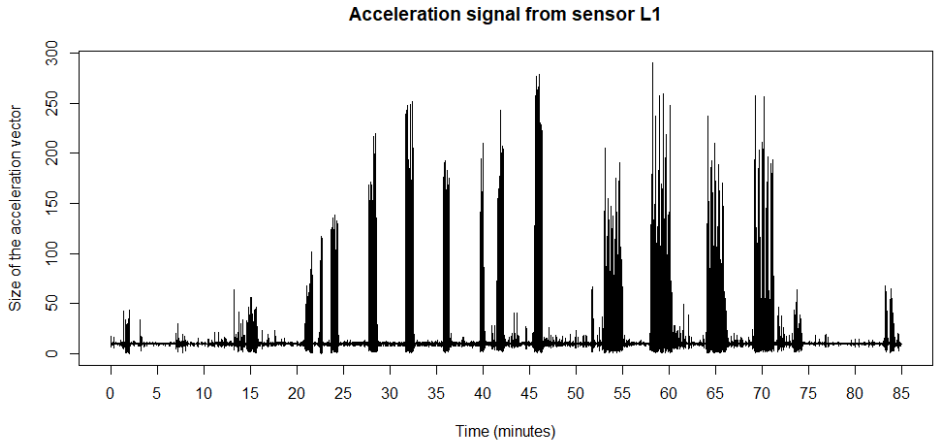


Figure 4.2: Size of acceleration on the left thigh during the drills for participant 1

The moments where the participant performed drills are mostly easy to discern by looking at the data. Especially the later exercises that occur between minute 50 and 75, drills 3a through 4b, are easy to see. However, the data between minute 15 and 45 is somewhat harder to interpret; instead of six distinct peaks of activity, we are left with some that are easy to recognise, such as the peaks between 25 and 35 minutes, and some that are harder to label. In these cases it would have been possible to match the data to the video recordings in order to clear up confusion, but as mentioned before, due to lack of access we were unable to do so. In general, moments of activity are easy to separate from moments of rest.

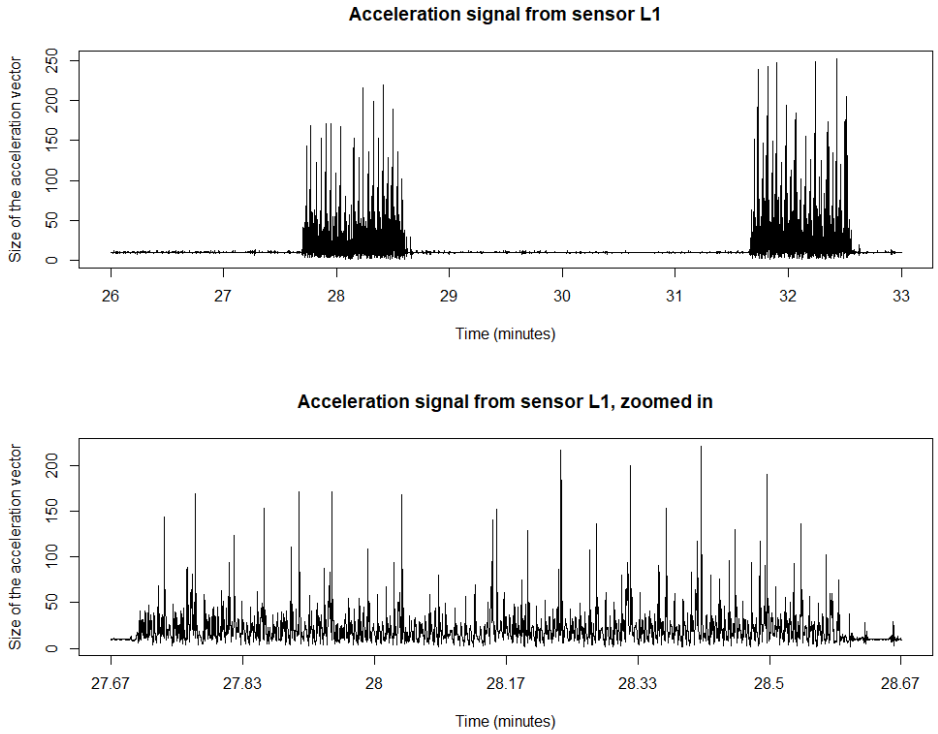


Figure 4.3: Size of acceleration on the left thigh during part of the drills

Figure 4.3 shows the acceleration during 7 minutes of the experiment. In these seven minutes, two drills were performed: drill 1b, where participants jog between pylons and pass at one side, and drill 1c, where participants performed the same drill but shot at one side instead of passing. Drill 1b is also shown in more detail in the bottom figure. Figure 4.3 shows acceleration from the dominant leg, as the athlete in question is left-footed. You can clearly see spikes where the participant turns or passes, both of which have an increased muscle load when compared to regular jogging.

4.1.4 Differences in sensors

Although Figure 4.3 shows distinct peaks in the data when the participant makes a pass, this is not the case for all the sensors. Depending on the dominant leg of the participant, the read-out from the sensors on the legs will differ; as the dominant leg kicks a ball, it will experience more muscle load than the non-dominant leg. This means that kicks and shots are more pronounced in the sensors of the dominant leg. To illustrate this, we plotted the same drill as in Figure 4.3, drill 1b, for the left shin, right shin and pelvis sensors.

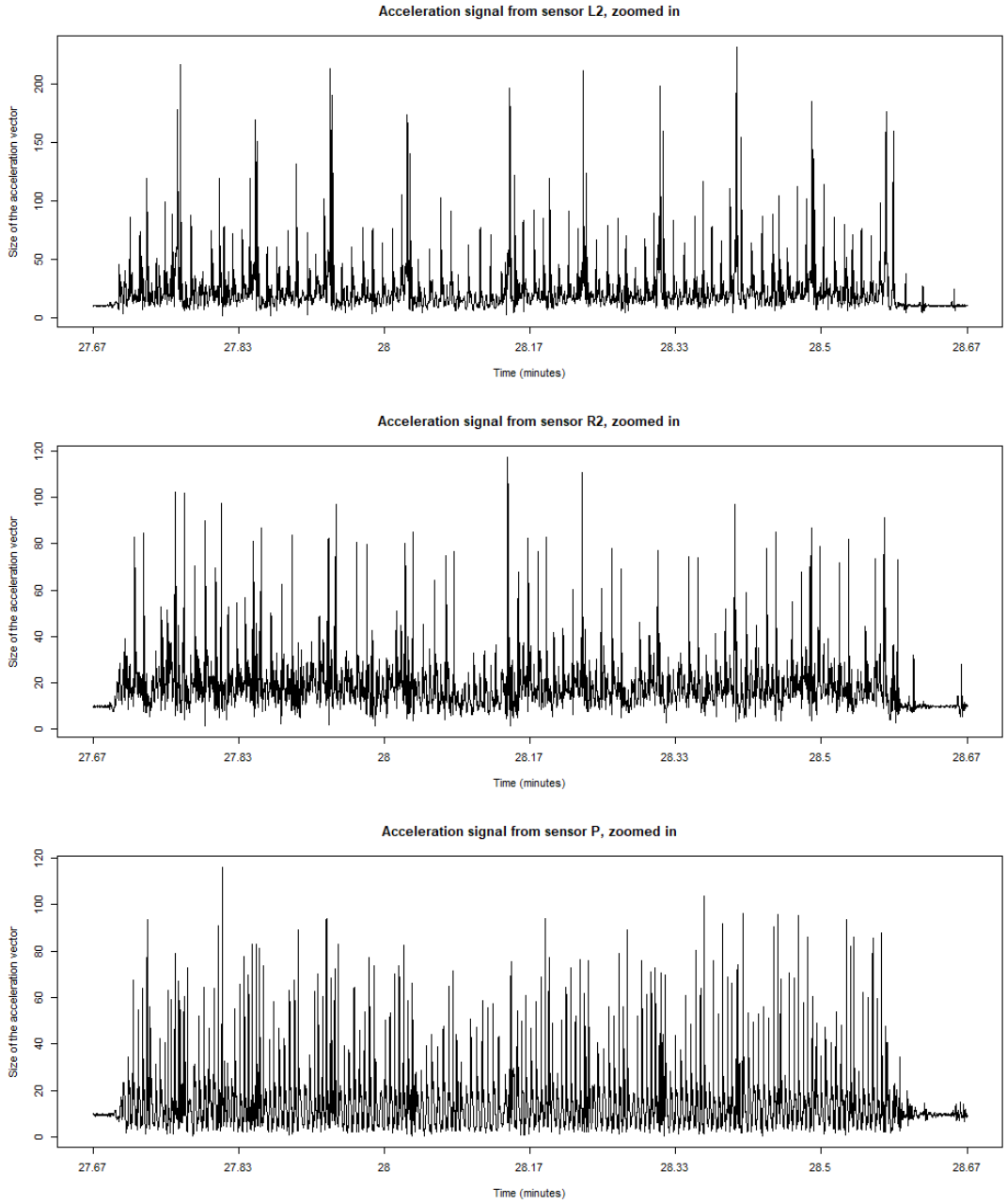


Figure 4.4: Size of acceleration on the left shin, right shin and pelvis during part of the drills

Figure 4.4 shows a clear difference between sensors. Whereas the difference between passing and jogging is clearly visible for the sensor data from the left (and thus dominant) shin, as the plot shows ten high peaks around 200 while the remainder of the peaks stay below 150 or 100, this difference becomes less clear for the right shin. Although most peaks for the L2 sensor are also visible in the plot for the R2 sensor, these peaks are less distinct and are surrounded by peaks of similar height. Also note the axes for both plots: the peaks from the L2 sensor reach a value of 200, while the acceleration in the R2 sensor mostly stays below 100. Passes are therefore harder to extract from the acceleration data.4

Furthermore, whereas separating peaks from passes and jogging was difficult for the R2 sensor,

it seems to be impossible for the one on the lower back; not only do the peaks that are visible for sensor L2 not show up for the P sensor, but there are barely any distinct patterns to discern.

4.2 Measures used for classification

In order to use a classification method, we first need to extract features from our data. For this research, the extracted features were only time-based. We chose a segment size of one second for feature extraction, corresponding to 200 data points per segment.

4.2.1 Previous research on activity detection

The features mentioned in this section are based on the paper by Bonomi, Goris, Yin and Westerterp [4], where accelerometers were employed to classify physical activity by type, duration and intensity. Here a number of features in the time domain are suggested for classification: the average, the standard deviation σ, the peak-to-peak distance a^{PP} and the cross-correlation R between axes and in the same axis. In this paper, the researchers were able to separate the acceleration in the axes and thus extract these features from the distinct axes. Unfortunately our data did not allow for separation of the axes, and as such the features were thus calculated for the size of the acceleration as in Equation (4.1).

The average and standard deviation are calculated the same as usual: a segment of the data is taken, and the values in this segment are used to compute the features.

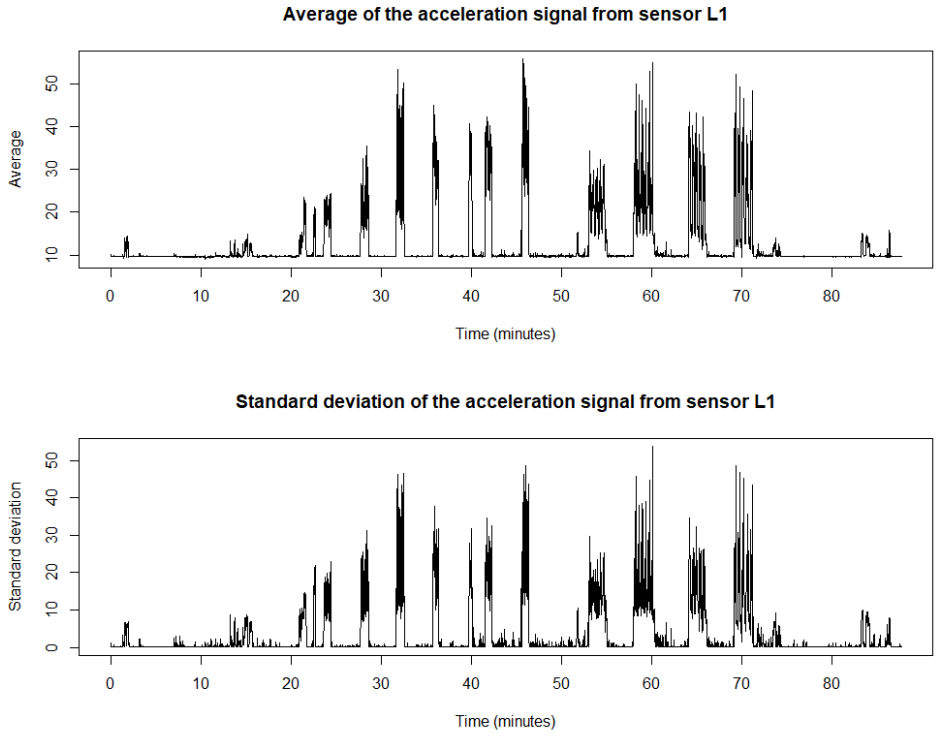
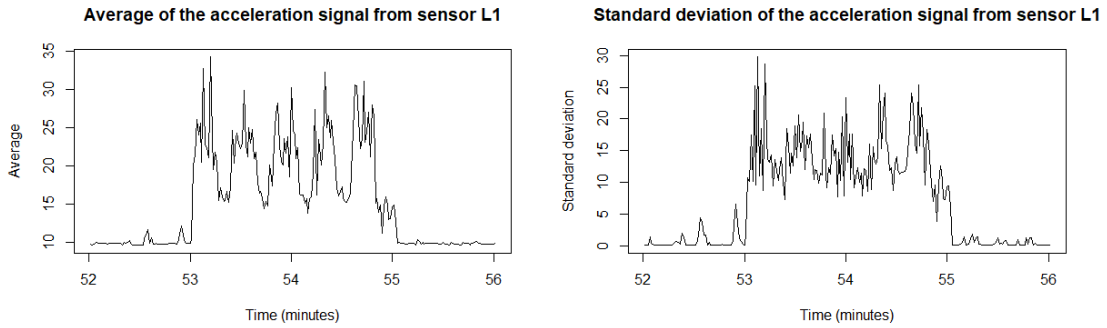


Figure 4.5: Average and standard deviation for the acceleration on the left thigh, calculated per second

Both features look quite similar; they are high when activity is happening and low when nothing

is happening. However, when zooming in on for example drill 3a, we see the following.



(a) Average of the acceleration (b) Standard deviation of the acceleration

Figure 4.6: Average and standard deviation of acceleration measured on the left thigh during drill 3a

Drill 3a consisted of zigzagging between a number of pylons while jogging, and then walking back to the beginning once they had reached all the pylons. In Figure 4.6a, the difference in the average between jogging between pylons and walking is clearly visible, while the standard deviation in Figure 4.6b has more difficulty separating these phases.

The peak to peak distance is a feature that calculates the average distance between peaks in the acceleration. As visible in Figure 4.7, the peak-to-peak distance increases as intensity increases, although there does not seem to be a difference between lower and higher intensity drills, nor is there much separation between drills and moments of rest.

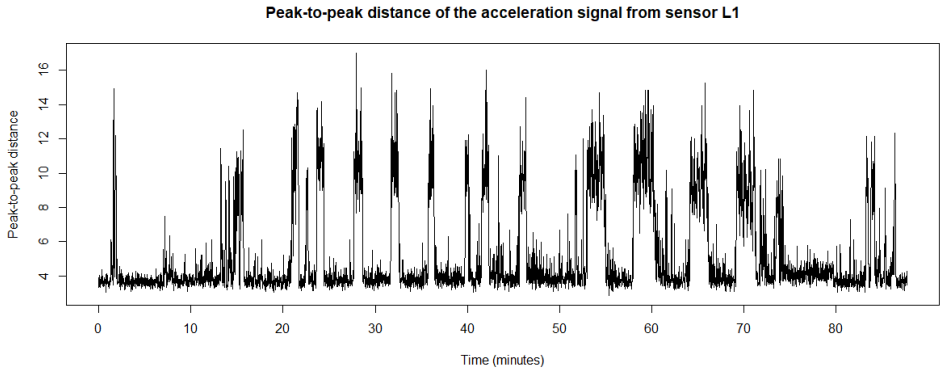


Figure 4.7: Peak-to-peak distance for the acceleration on the left thigh, calculated per second

The cross-correlation R is a feature that represents either a measure on the similarity between to axes when applied to multiple sensor axes, or the similarity in acceleration over two subsequent time intervals in the same axis. When applying to our data, the latter is the case. Given two subsequent segments of the same axis α and β of size N, i the shift between the two segments and j an index that covers the full length of the overlap between α and β, we can define:

$$r_{\alpha\beta}(i) = \begin{cases} \sum_{j=0}^{N-i-1} \alpha_{i+j}\beta_j, & i \geq 0 \\ r_{\alpha\beta}(-i), & i < 0 \end{cases} \tag{4.2}$$

We then define $R_{\alpha\beta} = \max(r_{\alpha\beta})$. The cross-correlation of a vector with a delayed copy of itself is also sometimes called the autocorrelation. Furthermore, the maximum is always obtained when α and β perfectly overlap; that is, $|r_{\alpha\beta}(i)| \leq r_{\alpha\beta}(0)$ due to the rearrangement inequality, stating that for any sequence $x_1 \leq \cdots \leq x_n$ and $y_1 \leq \cdots \leq y_n$ and permutation $x_{\sigma(1)}, \ldots, x_{\sigma(n)}$, we have

$$x_{\sigma(1)}y_1 + \cdots + x_{\sigma(n)}y_n \leq x_1y_1 + \cdots + x_ny_n. \tag{4.3}$$

Therefore, the cross-correlation will be equal to what is known in signal processing as the energy E_s of the segment size:

$$E_s = \langle x(t), x(t) \rangle = \int_{-\infty}^{\infty} |x(t)|^2 \mathrm{d}t. \tag{4.4}$$

where $x(t)$ is our acceleration vector. In the case of a discrete signal such as ours, this integral is replaced by a summation.

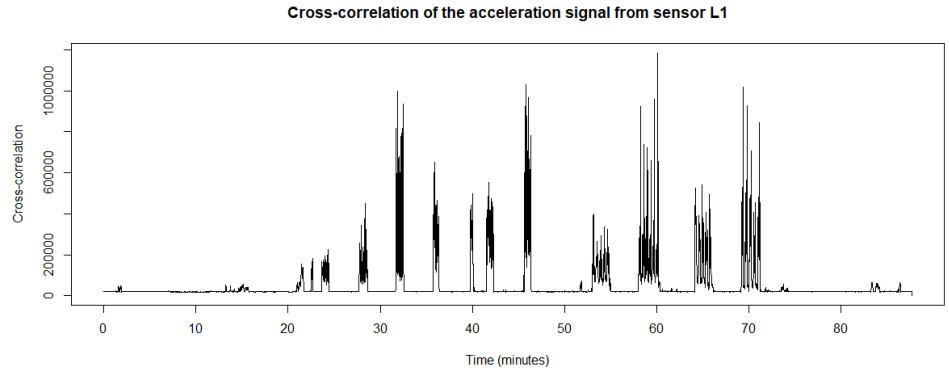


Figure 4.8: Cross-correlation for the acceleration on the left thigh, calculated per second

It seems the cross-correlation is especially high for drills in which participants had to shoot.

4.2.2 Previous research on original data set

In the research of Schotel [44], two measures were found to be the most effective for predicting the perceived intensity of a drill: method 12, which looks at the number of accelerations over a time frame, and method 15, which focusses on the number of peaks.

Additionally, these methods take into consideration how high the peaks and accelerations are: it divides the intensity into zones, based on the maximum magnitude of acceleration that the player was capable of during the exercise. The acceleration is normalised and the zones are divided accordingly: 0-10% for little to no activity (zone 1), 10-40% for low intensity (zone 2),

40-70% for medium intensity (zone 3) and 70-100% for high intensity (zone 4). This divide is visible in Figure 4.9.

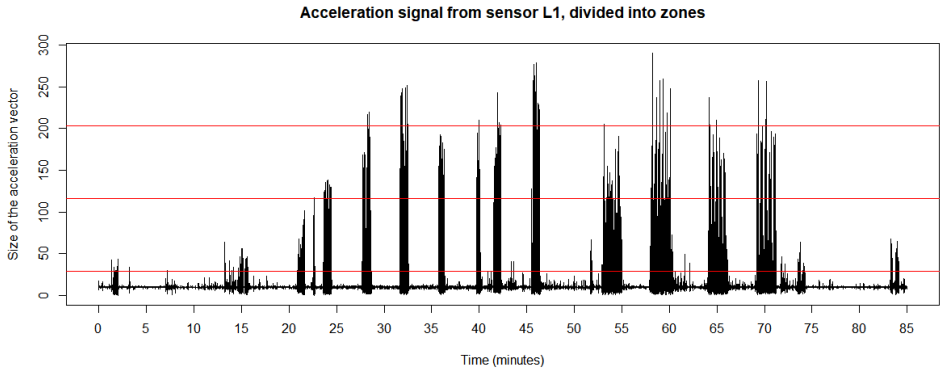


Figure 4.9: Zones for the acceleration on the left thigh

If an acceleration occurs between the 10 and 40 percentage lines, it will be counted for the low intensity zone. These calculations are performed per second, giving the number of accelerations A_j or peaks P_j per zone j. Before adding them, we multiply by a weight factor W_j for each zone j,

$$m_{12} = \sum_{j=1}^{4} A_j \cdot W_j \quad \text{and} \quad m_{15} = \sum_{j=1}^{4} P_j \cdot W_j \tag{4.5}$$

The weight factors for these were 0:1:4:7; if a peak or acceleration occurred in the lower 10% of the acceleration, it was not counted, whereas a peak in the top 30% counts sevenfold.

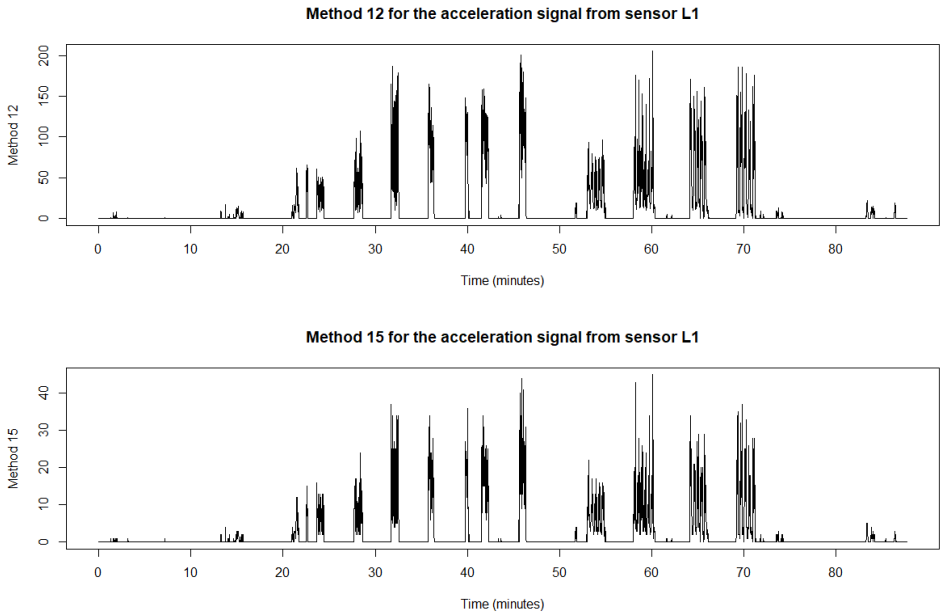


Figure 4.10: Method 12 and 15 for the acceleration on the left thigh, calculated per second

This measure was included as one of the features used for classification. They are quite similar, except in scale: method 15 has a range of 0-40, while method 12 has a range of 200. Otherwise they seem to correspond well to the moments of activity. When accounting for scale by normalising these features and calculating the difference between these methods, there is no obvious difference between the two. As can be seen in Figure 4.11, the only clear difference is that most often method 12 seems to have relatively higher values compared to method 15. This could be due to how this measures detect intensity: for every recorded peak in the data set, multiple accelerations occur.

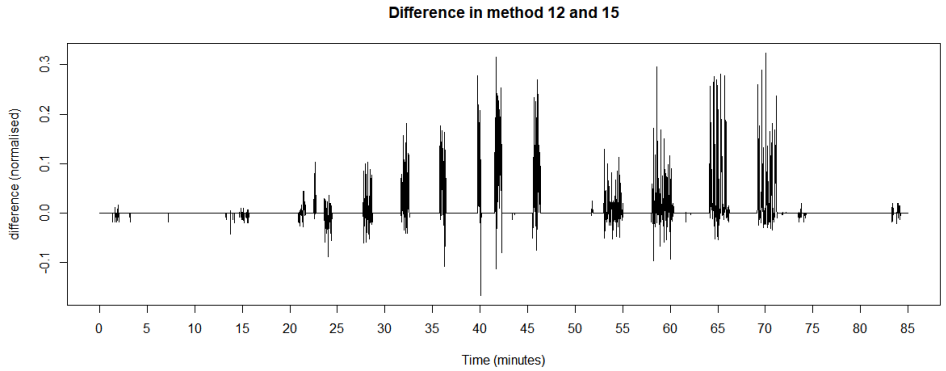


Figure 4.11: Relative difference between method 12 and 15 for the acceleration on the left thigh, calculated per second. Positive values indicate method 12 had a relatively higher value.

One thing to note is that these methods assume that during recording, at least one acceleration was made at high or maximum capacity. Because the methods rely on normalising the data set, a data set in which drills were performed at at most medium intensity could generate the same numeric values as one where the participants performed at the best of their abilities, despite a difference in muscle load.

5 Classification of intensity

The aim of this research is to design an algorithm than can classify activity into three intensity classes: low, medium and high intensity. Drill 1a is a good example of an exercise at medium intensity: the players were asked to jog back and forth between posts for a set number of minutes. Similarly, drill 4a is a good example of a high intensity exercise, where players are asked to zig zag at high speed between pylons.

Some exercises included passes or shots after finishing a drill, increasing the load for the player. However, the labelling of the data did not take this into account: the exercises where the participants jogged were labelled as medium intensity, those where they had to sprint were labelled as high intensity, and the remainder of the data was labelled low intensity exercise.

The length of the acceleration vector was computed from the accelerometer data, and the measures as discussed in Section 4.2 were then calculated on a time interval of 1 second. The sensor data of four out of five test subjects was then labelled and resulted in 86151 data points for low intensity, 7427 for medium and 4951 for high intensity exercise. In total the data set contained roughly 100.000 data points, 87% of which was classified as low intensity exercise. As we will discuss in Section 5.2, this abundance of low intensity data and the characteristics of this data means it is easier to classify than the other two categories.

5.1 Classification methods

In order to classify the data, we will need a method that can order data points into classes based on a training data set. Three methods will be considered for classification: k-nearest neighbours, decision trees and naive Bayes. The theory behind each method will be explained in separate sections. To give a better impression of the error rates we will discuss in Section 5.2, a trivial classifier will be discussed for comparison.

5.1.1 Trivial classifier

The difference between low intensity and medium or high intensity is not very difficult to discern. We could find a value for which we classify everything above as medium intensity exercise and below as low intensity exercise. Because the distinction between medium and high intensity exercise is more difficult to model, we ignore this and just use the two classes. If we look at this classifier, we find that the maximum accuracy obtainable is 93.5%. As can be seen in Table 5.1, this is achieved by setting a bound on the value of method 15 applied to the data.

For each feature described in Section 4.2, a function was defined that classified every data point with a value above a threshold as medium, below as low, and then computed the classification accuracy. This function was than optimised over the range of the measure.

	standard dev.	**Average**	**Peak-to-peak**	**Cross-corr.**	**Method 12**	**Method 15**
Threshold	7.3929	14.576	9.0020	54109	16.10	3.5667
Accuracy	0.9267	0.9298	0.9020	0.9334	0.9333	**0.9350**
Low int.	0.9873	0.9927	0.9744	0.9916	0.9917	0.9920
Medium int.	0.8422	0.8807	**0.6633**	0.8806	0.8788	0.8982
Activity	0.4442	0.4355	0.3198	0.4562	0.4556	0.4610

Table 5.1: Classification accuracy for trivial classifier

After this optimum was computed and the data was classified, we found the classification accuracy for all classes; 'Accuracy' shows the correct classification rate for all the data, 'Low int.' and 'Medium int.' the classification rate for low intensity and medium intensity data respectively, and 'Activity' shows the classification rate for medium and high intensity data combined.

Due to the abundance of low intensity data, especially compared to medium and high intensity data, this classifier finds its optimum at a point where almost all low intensity data is classified correctly; this can be seen in Table 5.1, where the accuracy for low intensity data remains between 97% and 99%, while the accuracy for medium intensity stays below 90%. However, as we will see in Section 5.2, most methods still achieved a relatively low error rate when considering only medium intensity data. Due to the fact that no high intensity exercise was correctly labelled, the overall accuracy when conditioning on high and medium intensity data still is quite poor.

5.1.2 K-Nearest Neighbours

K-nearest neighbours (knn in short) is a classification method that relies on a certain metric, usually Euclidean, to determine the k closest points to our data point in the training set and takes the average of the response variable of these points. In case of qualitative response variables, this is similar to a majority vote. This method is a natural result in our search for methods that are as accurate as possible. When considering $X \in \mathbb{R}^p$ as a real-valued input vector and $Y \in \mathbb{R}$ as a response factor, with a joint distribution $Pr(X, Y)$, we can use a loss function to determine which prediction function f is more accurate. Most commonly the squared error loss,

$$L(Y, f(X)) = (Y - f(X))^2, \tag{5.1}$$

is used to penalise errors in prediction. This gives us a criterion for finding a good prediction for Y: we would like to minimize the expected prediction error

$$EPE(f) = \mathbb{E}(Y - f(X))^2 = \int [y - f(x)]^2 Pr(dx, dy). \tag{5.2}$$

This function is minimised by the conditional mean $f(x) = \mathbb{E}(Y|X = x)$. K-nearest neighbours approximates this solution by

$$\hat{f}(x) = Average(y_i | x_i \in N_k(x)), \tag{5.3}$$

where $N_k(x)$ is the neighbourhood consisting of the k closest points to x in the training set and y_i is the response for x_i. The k-nearest neighbours classifier uses two methods to approximate the conditional mean: the expectation is approximated by averaging over the sample data, and conditioning on $X = x$ is relaxed to conditioning on a certain region in proximity to the target x. Theoretically, provided the joint probability distribution adheres some regularity conditions, as $N, k \to \infty$ such that $k/N \to 0$, the k-nearest neighbour function converges to the true solution $\mathbb{E}(Y|X = x)$ [24, Section 2.4].

Although this sounds very promising, k-nearest neighbours does have some downsides: it can be unstable at times, and suffers from the curse of dimensionality [24, Section 2.5]. Fortunately, although we are dealing with a vast amount of data, our data is not high dimensional and thus some of the problems with k-nearest neighbours, such as the increase of the metric size of the k-nearest neighbourhood, will not have a huge impact on the classification accuracy. As for

stability, this method performs well when there is a large sample size available for training; considering the large amount of data, stability should be largely accounted for.

5.1.3 Decision trees

Tree-based methods for regression and classification use splitting rules to segment the data into a smaller number of regions in order to predict outcomes, thus earning the name *decision trees*. In order to grow a decision tree, the first step is to find suitable regions for stratification: this is done by recursive binary splitting, a top-down, greedy algorithm that splits the predictor space into sections $R_1, \dots, R_n$ trying to minimise either the RSS for regression trees, or an alternative to the classification error rate for classification trees.

An intuitive measure for the classification error rate would be the misclassification error. We first define $\hat{p}_{mj}$ as the proportion of training observations in the m-th region R_m that are from the j-th class; that is, given region R_m with N_m observations and class j, we define

$$\hat{p}_{mj} = \frac{1}{N_m} \sum_{x_i \in R_m} \mathbb{1}(y_i = j). \tag{5.4}$$

Then the misclassification error would be the quantity:

$$E = 1 - \hat{p}_{mj}. \tag{5.5}$$

However in practice this is not preferable for multiple reasons. In the case of classification trees, most often either the Gini index or cross-entropy is used [25, Section 8.1]. The former is a measure of total variance across the J classes, defined by

$$G = \sum_{j=1}^{J} \hat{p}_{mj}(1 - \hat{p}_{mj}). \tag{5.6}$$

This measure is low whenever the $\hat{p}_{mj}$'s are either close to zero or to one, so the Gini index is a good criterion for splitting nodes. Alternatively, cross-entropy can be used as a criteria, denoted by

$$D = -\sum_{j=1}^{J} \hat{p}_{mj} \log(\hat{p}_{mj}). \tag{5.7}$$

Similarly to the Gini index, cross-entropy takes on lower values when the $\hat{p}_{mj}$'s are close to either zero or one. Both methods are numerically similar; in the case of two classes, and p the proportion of observations in the second class, the measures are $E = 1 - \max(p, 1-p)$, $G = 2p(1-p)$ and $D = -p\log(p) - (1-p)\log(1-p)$. In this case both the Gini index and the cross-entropy are differentiable to p, and hence "more amenable to numerical optimization" [24, Section 9.2]. The misclassification error does not have this property, nor is it as sensitive to changes in the node probabilities as the Gini index or cross-entropy. Therefore, this method is not very suitable for growing the decision tree.

Once a measure has been chosen, the recursive binary splitting algorithm defines for any i and s the half-planes

$$R_1(i,s) = \{X|X_i < s\} \quad \text{and} \quad R_2(i,s) = \{X|X_i \geq s\} \tag{5.8}$$

and seeks the value of i and s for which the value of the chosen measure on these planes is lowest. Once it has found this optimum, it creates a split in the predictor space and repeats the process for the newly created space of subsets, starting either in R_1 or R_2. This continues until the predictor space has been sufficiently stratified, which is at a certain stopping criterion. The result of this recursive binary splitting approach is a tree with separately defined subsets of the predictor space, all of which cover a part of the training data. Classification is then based on which class has the majority vote within each region R_j.

One of the strengths of this method is the interpretability of the model. However, it does have its weaknesses. Often the segmentation created by the recursive binary splitting is very suitable for the training data, but might be too complex for the test set. Pruning is used to reduce the chance of overfitting the data: it uses the same criteria as mentioned before, including the misclassification error. If overfitting is a problem, it could reduce the number of nodes in the tree.

Another issue with decision trees is their high variance. Hastie, Tibshiran, and Friedman (2009) note that "often a small change in the data can result in a very different series of splits, making interpretation somewhat precarious. The major reason for this instability is the hierarchical nature of the process: the effect of an error in the top split is propagated down to all of the splits below it." [24, Section 9.2].

5.1.4 Naive Bayes

Suppose we have a classification problem with J classes and want to quantify the accuracy of our estimate $\hat{f}$. Previously we used a quadratic loss function to motivate the k-nearest neighbours method, but we could use the test error rate as another measure for accuracy. This last measure is minimized on average by the Bayes classifier, which assigns test data to the class j for which the conditional probability $Pr(Y = j|X = x_0)$ is largest [25, Section 2.2.3].

In practice this conditional probability is unknown but can be approximated by several methods. One of those is naive Bayes, a method that uses nonparametric kernel density estimates for classification. It thanks its name to Bayes' theorem and the naive assumption at its core: it assumes that for a class j, the predictors X_k are independent, thus reducing the probability density function to

$$f_j(X) = \prod_{k=1}^{p} f_{jk}(X_k) \tag{5.9}$$

Despite its name and rather naive assumption, this method is popular and often performs quite well. The assumption simplifies the estimation by reducing the difficult task of estimating a joint density to estimating individual class-conditional marginal densities, which can be done by using one-dimensional kernel density estimates [24, Section 6.6].

Given the premise of J classes and class priors $\hat{\pi}_j$ (for example the sample proportions) for each class $j \in \{1, \ldots, J\}$, we can fit nonparametric density estimates $\hat{f}_j(X)$ for each class j seperately. By using Bayes' theorem and our assumption of independence, we can compute the posterior by

$$\hat{Pr}(Y = j|X = x_0) = \frac{\hat{\pi}_j \hat{f}_j(x_0)}{\sum_{m=1}^{J} \hat{\pi}_m \hat{f}_m(x_0)} = \frac{\hat{\pi}_j \prod_{k=1}^{p} f_{jk}(x_{k,0})}{\sum_{m=1}^{J} \hat{\pi}_m \prod_{k=1}^{p} f_{mk}(x_{k,0})}. \tag{5.10}$$

This estimate is then used for classifying, assigning classes depending on which class j maximises the posterior. The naive Bayes method is favourable when dealing with high dimensional data.

5.2 Accuracy

All three aforementioned methods have their advantages and disadvantages, both theoretically and practically. To know which method performs the best with our data, the easiest method is to run all methods and compare the error rates. The methods were run both on the normal data set and a data set where the values were normalised per value and vector. Furthermore, the methods were tested by using 5-fold cross validation and the accuracy was averaged over these five trials.

The results are visible in Table 5.2. We computed not only the total percentage of correctly classified data, but to give a broader picture of how these methods performed, the accuracy for high, medium and both high and medium (activity) data was recorded, along with the time it took to run the 5-fold cross-validation.

	Accuracy	**Activity**	**High int.**	**Medium int.**	**Time (s)**
Decision Tree	0.9620	0.7556	0.7062	0.7886	16.2
Decision Tree [N]	0.9632	0.7482	0.7210	0.7663	
Pruned DT	0.9618	0.7546	0.7021	0.7893	13.9
Pruned DT [N]	0.9628	0.7495	0.7273	0.7646	
Naive Bayes	0.9379	0.7677	**0.6060**	**0.8755**	37.7
Naive Bayes [N]	0.9419	0.7562	0.6290	0.8408	
KNN	0.9525	**0.6798**	0.6019	0.7317	480.7
KNN [N]	0.9641	0.7432	0.7216	0.7579	

Table 5.2: Classification methods and their accuracy using time windows of one second, specified by type of intensity classified. [N] indicates normalised data was used. Time indicates the time needed to compute run the classification code in `R` for the regular and normalized data set, in seconds.

As can be seen in Table 5.2, the overall accuracy of all the methods are comparable: all methods perform between 93.7% and 96.4%. Another commonality is that the accuracy drops roughly 20% when not taking low intensity-acitivities into account. This last measure gives a better impression of how accurate our method is, as there is a relatively large amount of low intensity activity recorded and thus skewing the results. Furthermore, this activity is very easy to classify, whereas the difference between medium and high intensity exercise is more difficult to distinguish.

5.3 Final result

After comparing all methods there is not one method that outperforms the others by a large margin; both the decision tree models and the k-nearest neighbours have an overall accuracy of 96% and are roughly 75% accurate when classifying sensor data that was recorded when the players were performing an exercise. Naive Bayes is the highest performing method when only considering moments of activity, but this is not the case when considering only high activity exercise: here the model performs as poor as k-nearest neighbours on the unnormalised data.

In light of all this information, a decision tree seems preferable to the other methods due to its accuracy and speed. Of all three methods, this method is also easiest to interpret and apply to new data sets. The final classification tree is shown in Figure 5.1. It should be noted that pruning did not affect the decision tree.

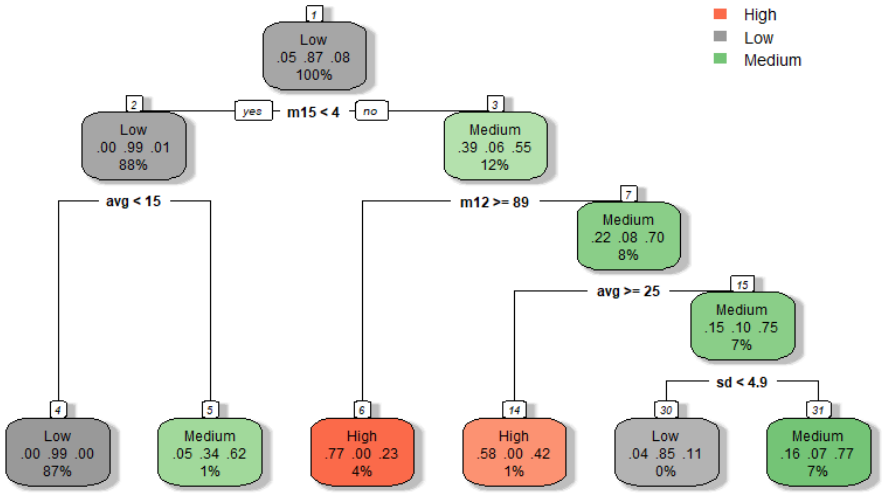


Figure 5.1: Final decision tree, pruned. The data percentages visible in the nodes are not in ascending order but are ordered High - Low - Medium intensity.

As expected, lower values for the classifying measures such as average and standard deviation gives cause for a lower intensity classification. One curious thing to note is the classification accuracy for the different nodes in the decision tree: while the percentage of low intensity data in nodes 4 or 30, where the sensor data is classified as low intensity, is pretty high, these percentages decrease when we look at high and medium classification such as nodes 5 and 14. When we look at node 4 specifically, we see that of the 87% of the full data set that falls in this category and is classified as low intensity, 99% actually is low intensity data. As for node 14, which covers roughly 1% of the data and classifies it as high intensity, only 58% - barely half - of this data is actually high intensity data. When we look more closely to the error rates of this method, we can see this effect.

Error rates

The error rate that is most important to us is that of the labelling of medium and high intensity exercise to low intensity exercise and vice versa; only 3.9% of the medium and high intensity exercise is labelled low intensity exercise, and conversely only 0.9% of low intensity exercise is classified as medium intensity exercise. The full confusion matrix is given in Table 5.3 below.

	Predicted values			
	Low	**Medium**	**High**	*Total*
Low	85.416	735	0	86.151
Medium	299	5739	1389	7427
High	178	1108	3665	4951
Total	85.893	7582	5054	**98.529**

Table 5.3: Confusion matrix for the decision tree in Figure 5.1

As noted before, this method has some difficulty with classifying high and medium intensity correctly. This can be due to several factors, such as incorrect labelling, the inclusion of shots and passes in medium intensity drills (thus containing high intensity moments in data that is labelled as medium intensity), the general nature of the exercises and data or the classification method itself. Nevertheless in terms of overall activity recognition it outperforms the trivial classifier as mentioned in Section 5.1.1, thus proving its value.

5.3.1 Excluding normalising measures

As discussed in Section 5.3, the final classification model makes use of two methods that require a complete data set, preferably including a moment where the participant performed at max intensity, in order to calculate a "normalised" data set. Both method 12 and 15 are designed to work on data sets that are divided into acceleration zones, in our case 0-10% for no activity, 10-40% for low activity, 40-70% for medium activity and 70-100% for high activity. There are two problems with the use of these measures for classification:

1. Classification on-the-go is more complicated because it requires a recalculation of both measures during exercise as maximum acceleration might increase over time, causing a shift in the aforementioned acceleration zones. This also means that when the sensors have just started recording, most data would be classified as medium or high intensity even though not much activity has happened yet. It is clear that for real-time feedback, the classification can be unreliable.

2. Normalising the data means that consistently low output would be on the same level as consistently high output. That is, when a person chooses not to sprint but to jog during the entire exercise and there are no activity bursts that would increase the maximum acceleration, the top 70-100% of the acceleration of this participant would be set equal to that of a person that had sprinted multiple times during the exercise.

To work around those two problems, we can choose to disregard those two methods and only use the 'invariant' measures such as average and peak-to-peak distance for classification. The results of classifying with this reduced data set can be seen below.

	Accuracy	Activity	High int.	Medium int.	Time (s)
Decision Tree	0.9553	0.7179	0.5915	0.8027	12.5
Pruned DT	0.9558	0.7059	0.5958	0.7798	12.0
Naive Bayes	0.9401	0.7251	**0.4711**	**0.8945**	34.5
KNN	0.9525	**0.6791**	0.6005	0.7316	392.3

Table 5.4: Classification methods for non-normalised data and their accuracy using time windows of one second, specified by type of intensity classified. Time indicates the time needed to compute run the classification code in `R` for the regular and normalized data set, in seconds.

There are a few similarities and a few differences between what we see when classifying with the reduced data set, and the previous results in Table 5.2. We can see that again the decision tree and naive Bayes outperform the k-nearest neighbour method. Another similarity is the discrepancy between the classification of high intensity data and medium intensity data for the naive Bayes method.

Unfortunately the exclusion of some of the measures has lead to an increase in the classification error rate for activity data, specifically high intensity data. It seems these measures were helpful in differentiating high and medium intensity data, and without them especially high intensity data is more often incorrectly classified. Curiously, the k-nearest neighbours method now has the lowest misclassification error for high intensity data. All things considered however, the decision tree seems again to be most suited for our data, and so our final model is as follows:

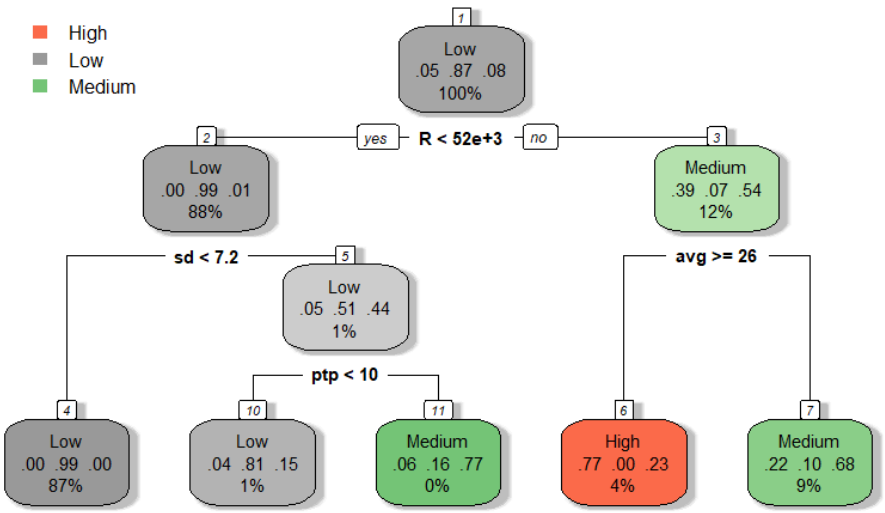


Figure 5.2: Decision tree for reduced data set. As with Figure 5.1, the order in the nodes is high - low - medium intensity

As we can see in Figure 5.2, when classifying with a reduced data set, all four remaining measures are used. Similarly to the tree in Figure 5.1, the nodes 4 and 10 that are classified as low intensity also have the lowest misclassification error, while the medium and high intensity nodes have a lower percentage of data correctly classified. The confusion matrix of this model confirms this as well.

	Predicted values			
	Low	**Medium**	**High**	*Total*
Low	85.230	921	0	86.151
Medium	412	6147	868	7427
High	173	1876	2902	4951
Total	85.815	8944	3770	**98.529**

Table 5.5: Confusion matrix for the decision tree in Figure 5.2

Again only a small percentage of the data is incorrectly classified between low and medium/high intensity data: 1.1% of low intensity data is labelled as medium intensity, and 4.7% of medium or high intensity data is labelled as low intensity (3.5% when only considering high intensity data). Sadly the correct classification rate for high intensity data stays roughly 59%, although medium intensity exercise is now recognised much better with an accuracy of 83%.

Although classification with less predictors is possible and more feasible for offering direct feedback, if a complete data set (including moments of high activity) is available, it seems the inclusion of measures such as method 12 and 15 would be beneficial for correct classification of the data.

Baseball

6 Growth curves for longitudinal data: a literature review

The modelling of growth curves has been a topic of interest among scientists of many disciplines for a long time. As such, there is a vast amount of literature on the topic, ranging from philosophical ideas about growth curves to papers specifying how to compute certain parameters for certain models in a range of programming languages. Growth curves are applicable in many fields, such as the growth of livestock in agriculture [54], the concentration of drugs in blood or levels of cholesterol in medicine [27], the performance of students in schools in educational studies [28]; they can be found in most social and biological sciences.

A natural way of defining growth curve models is to link it to two distinct questions about change [43]:

1. How does growth behave over time for an individual?
2. Is there a significant difference in how growth behaves for different subjects? Are there predictors that explain differences among the change trajectories between individuals?

Both questions are related to fitting growth curves to a data set, but where the first question is related to the within-person level, the second focusses on the between-persons level. While you can answer both questions separately, the simultaneous modelling of these concepts makes growth curves vastly more appealing.

6.1 History of growth curve modelling

For many years there have not been many successful methods for the modelling of systematic change over time. As noted in the short overview by Bollen and Curran [3], "although philosophical discussion of change have been traced back as far as Aristotle (Zeger and Harlow, 1987), the earliest development of statistical analysis of data over time appears to be early in the nineteenth century" (p. 10). Here the focus of the research laid mostly on the modelling of growth curves for groups and populations, rather than the individual, such as estimating a trajectory for characterising continuous mortality rates [22], population growth [50, 51] or anatomical growth [39]. This focus on group-level trajectories remained all through the 1930s, where progress was made in the diversification of applications of these models. Not until the paper by Wishart [54] in 1938, modelling weight gain in bacon pigs, did the interest shift from group to individual trajectories.

Two historically popular methods for analysing longitudinal data were analysis of variance (ANOVA) and multivariate analysis of variance (MANOVA); as Everitt [16] points out, both of these methods have their issues when applied to longitudinal data. One requirement for ANOVA is the homogeniety of variance over time, which is unlikely for this kind of data as observations usually have some form of autocorrelation and variance often increases over time.

In a paper by Potthof and Roy [37] a generalisation of the MANOVA model has been proposed for dealing with growth curves by adding a post-matrix. However, these models increase considerably in size as the number of predictors or treatment effects and the number of groups grow. MANOVA is more flexible in the definition of covariance structures, but "only by including too many parameters; MANOVA falls down on economy" [16]. As late as the seventies there has therefore been a push back to the methods used to model growth curves, as noted by Cronbach and Furby [9] in their paper on modelling growth: "investigators who ask questions regarding gain scores would ordinarily be better advised to frame their questions in other ways." (p. 80).

6.2 Modern methods for growth curve modelling

This attitude towards growth curves shifted after the rise in popularity of other methods that were found useful for modelling change. Currently there are two general methods for fitting growth models to longitudinal data: multilevel (mixed effects) methods and structural equation models [10].

While random coefficient models had been introduced as early as 1919 by Fisher [17], it would take some time until mixed effects models - models consisting of both fixed and random effects - would gain popularity. However, as Pinheiro and Bates [34] note in their book on mixed-effects models, their current popularity "is explained by the flexibility they offer in modeling the within-group correlation often present in grouped data, by the handling of balanced and unbalanced data in a unified framework, and by the availability of reliable and efficient software for fitting them".

Similarly, the first person to propose using latent variables for trajectory modelling was Baker in 1954. In his paper he modelled change by reducing a data set containing 20 repeated measures to four latent factors, each representing different stages of growth [1]. This would in 1984 form the basis for the latent curve modelling using structural equation modelling (SEM) [3], more than a century after the first serious attempts at statistical modelling of change.

This last method aims to model change by assuming a latent, unobserved, trajectory underlying the observed growth. Structural equation models are a more general approach to modelling and testing structural relationships between observed variables, including regression, confirmatory factor and path models [46]. Within this framework, latent curve modelling specifies these relationships more thoroughly: it assumes the existence of some latent variable that can be inferred by the relationships between the observed variables, which form the components of the underlying trajectory [3].

As Curran, Obeidat and Losardo [10] note in their review paper on growth curve modelling, for many data sets and analyses the multilevel and SEM approach are "numerically identical". However, due to the way the observed measures are modelled, there is a clear difference for our analysis. The SEM approach incorporates these measures as indicators on a number of latent factors, which would be increasingly difficult to model for our data as the times at which the participants have been measured, and the number of measurements, differ significantly between participants. We therefore chose to focus on multilevel models, which models this structure more forgivingly.

6.3 Theoretical mixed models and their computation

As with growth curve modelling, there is simply too much literature to properly review in a few pages. We will therefore only give a short overview of the papers that were most instrumental for the analysis in this thesis.

Laird and Ware [27] give a generalized linear model for data with multiple measurements per research unit, which includes growth models as a special case, by dividing the model into two stages. For the first, they introduce population parameters, individual effects and within-person variation. With $\boldsymbol{y_i}$ the variable for which to predict, the model is given by

$$\begin{aligned} \boldsymbol{y_i} &= \boldsymbol{X_i}\boldsymbol{\beta} + \boldsymbol{Z_i}\boldsymbol{b_i} + \boldsymbol{\varepsilon_i}, \quad i = 1, \cdots, M, \\ \boldsymbol{b_i} &\sim N(\boldsymbol{0}, \boldsymbol{\Sigma}), \quad \boldsymbol{\varepsilon_i} \sim N(\boldsymbol{0}, \boldsymbol{R_i}), \end{aligned} \tag{6.1}$$

Where $\boldsymbol{\beta}$ a vector of unknown population parameters and $\boldsymbol{X_i}$ the design matrix linking $\boldsymbol{\beta}$ to $\boldsymbol{y_i}$; furthermore, they define $\boldsymbol{b_i}$ as a vector of unknown individual effects and $\boldsymbol{Z_i}$ as the associated design matrix. Afterwards the between-person variation is included by defining a distribution for the $\boldsymbol{b_i}$. Additionally, $\boldsymbol{R_i}$ is a positive-definite covariance matrix depending on i in size but not in value.

This models can be extended to nonlinear models; Goldstein [21] states the general model can be written as the sum of a mixed-effects linear and nonlinear component. In his paper, he proposes the following model:

$$\boldsymbol{y} = f(\boldsymbol{X_1}\boldsymbol{\beta} + \boldsymbol{Z_u}\boldsymbol{u}) + \boldsymbol{X_2}\boldsymbol{\gamma} + \boldsymbol{Z_e}\boldsymbol{e}, \tag{6.2}$$

with f a nonlinear function, $\boldsymbol{X_1}$ and $\boldsymbol{X_2}$ design matrices for the fixed coefficients $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, $\boldsymbol{e}$ and $\boldsymbol{u}$ sets of random variables with zero means and $\boldsymbol{Z_e}$ and $\boldsymbol{Z_u}$ their corresponding design matrices. This model is then solved by first applying linearisation of f before using standard procedures for the linear multilevel model. Conversely, Lindstrom and Bates [29] propose the nonlinear model

$$y_{ij} = f(\boldsymbol{\phi_i}, \boldsymbol{x_{ij}}) + e_{ij}, \tag{6.3}$$

where f is a nonlinear function of the predictor vector $\boldsymbol{x_{ij}}$ and a parameter vector $\boldsymbol{\phi_i}$ which can be written as a combination of fixed and random effects; they are solved by a combination of least squares estimators for nonlinear fixed effects models and maximum likelihood estimators for linear mixed effects models.

There is a vast amount of literature on mixed effects models and the estimation of their parameters. Methods for computing maximum likelihood estimates for random coefficient models are reviewed in a paper by Harville [23] and later extended by Goldstein [20], where an outline is given for how mixed effects models can be specified for hierarchical models and how this hierarchy can be utilised in estimation. Furthermore, an iterative generalized least squares estimation procedure is given for the computation for the maximum likelihood in the normal case.

On the other hand, Laird and Ware [27] discuss a "unified approach to fitting these models", by a combination of empirical Bayes and maximum likelihood estimation of model parameters and using the EM algorithm. This method is challenged by Lindstrom and Bates [30] by their Netwon-Rhapson method, which is shown to be preferable to the EM algorithm in most cases.

Two packages are currently most common in the modelling of mixed effects models in `R`, namely `nlme` [35] and `lme4` [2]. Both methods have some benefits in certain specific situations, but are otherwise interchangeable. For this research, the package `nlme` is used to model the growth curves for pitching speed. The computational methods for fitting linear mixed effects models fit by the `nlme` package "follow the general framework" of the aforementioned paper by Lindstrom and Bates [30], and use the model formulation of Laird and Ware [27, 35].

6.4 Applications to baseball

There have been instances of the use of growth curves in research to assess longitudinal data in sports. Latent growth curves are used in the research by Conroy and Coatsworth [8] to determine the effect of coach training on fear of failure in youth swimmers. As for mixed effects models,

the research of Roring and Charness [42], researching the effect of aging on the performance of elite chess players, is one of the few that applied a multilevel model to longitudinal data.

Baseball games are quite straightforward to analyse, giving rise to many fan and professional websites that track and analyse Major League games. In 2015 a new pitcher performance metric has been constructed, namely the Deserved Run Average (DRA), which incorporated mixed models in the design [26]. In academic literature however, mixed effects models in baseball research are not very popular; although some applications can be found, such as the use of multilevel models in assessing rate of injury depending on the use of protective equipment [32], research on performance growth curves is particularly hard to find.

This research thus has two main objectives. The first is to find a model that accurately describes the growth in throwing speed of young athletes, which will aid baseball professionals in recruiting and training during the ages of 12 years old to 18 years old. The second is to examine whether growth curves analysis via mixed-effects models can be applied to data on performance in baseball.

7 A look at the data from project Fastball

In this chapter we will focus on the data from project Fastball. Before predicting our model for the growth curve, we will first examine our data.

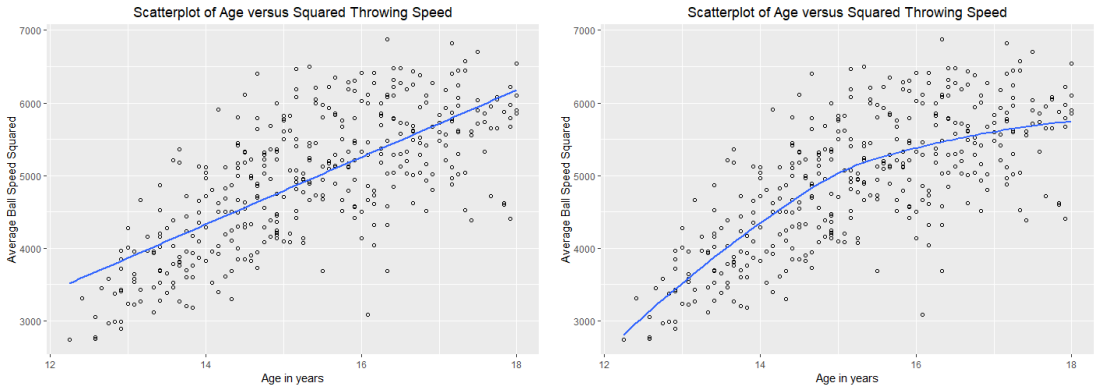
7.1 Overview of data and preparation

In total there are 412 observations in the data set. Because we will model the throwing speed for youth between 12 and 18, we remove 21 observations for pitchers who are outside of this range. We also remove observations for which no ball throwing speeds were measured. The remaining data set contains 391 observations, all of which the ball throwing speeds and age are measured. Of the other variables, height and weight are missing in 5 cases, force of internal rotation in 6 cases, force of external rotation in 7 cases, ROM of the external rotation 5 times and ROM of the internal rotation 3 times.

The remaining data seems to suggest a growth curve similar to a logistic curve, so we first transform the data by squaring the throwing speed. This squared speed is also related to the force needed for throwing by the kinetic energy of the ball:

$$E_{kin} = \frac{1}{2}mv^2 \tag{7.1}$$

Here m is the mass of the ball, and v the speed at which the ball is thrown. Plotting the age against the squared ball throwing speed gives us the following results:



(a) Scatter plot with linear regression line (b) Scatter plot with loess regression line

Figure 7.1: Relationship between pitchers' age and squared throwing speed

The transformation seems to have improved the linearity of the data set, as is illustrated in Figure 7.1a. However, this linear regression line does not seem to be a perfect fit to the data; we therefore computed the loess regression curve and observed that the data set does not follow a linear trend. From Figure 7.1b it does seem as if the growth curve can be partitioned into two different sections; between the ages of twelve and fifteen, the pitchers seem to increase in speed quite a lot as they grow older. When they pass the age of fifteen, however, this trend seems to slow down. This same behaviour is seen when plotting the height of the players against their age. We will investigate this in Section 7.3.

As for outliers, only a few data points seem to differ significantly from the rest. Specifically around sixteen years some pitchers seem to drop in performance.

The most obvious and outlying of these data points is quite visible in Figure 2.5b, as the pitcher is performing as expected at the age of fifteen but drops dramatically at the age of sixteen, after which he performs as expected again. When examining the outlier, it seems that during the six months prior to this measurement he had been injured in both his elbow and shoulder. His performance is therefore not unusual given the circumstances, but the observation is nonetheless quite outlying. Removing this observation however does not change the general structure of the data; when plotting the data and adding the loess line, there is still a clear divide between growth before and after the age of fifteen.

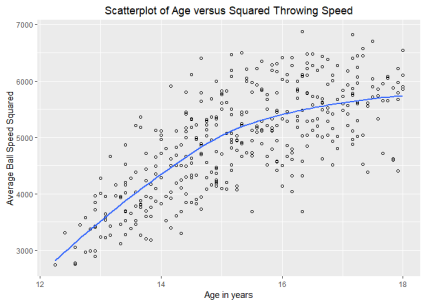
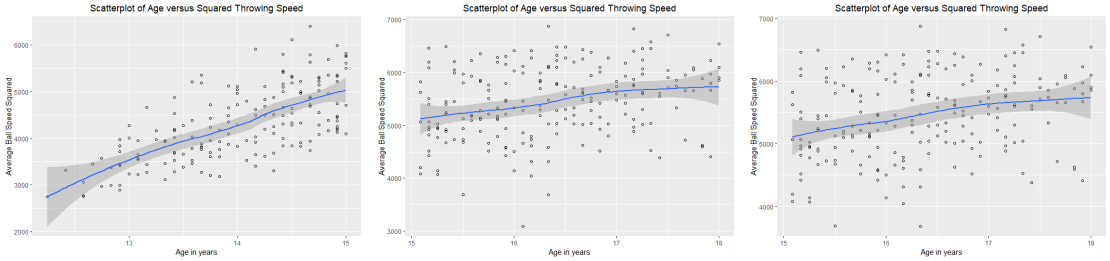


Figure 7.2: Scatter plot with loess regression line after removing the outlier

7.2 Size and spread of data

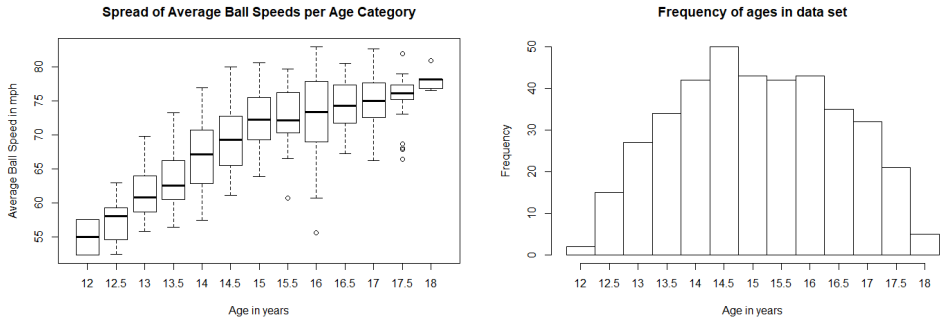
After inspection of the data it seems fair to assume that the observations can best be described by two functions that differ depending on age. If we take the age of 15 to be the turning point for this function, we can model the data after separating it into two sets. The results are as follows:



(a) Under 15 (b) Over 15 (c) Over 15, removed outlier

Figure 7.3: Relationship between pitchers' age and squared throwing speed, separated by age

There seems to be a wide spread in both categories, but especially for the category between 15 and 18 there seems to be a lot of variation between the different players.



(a) Spread of Ball speeds based on age (b) Frequencies of age

Figure 7.4: Spread of Age and Ball speeds

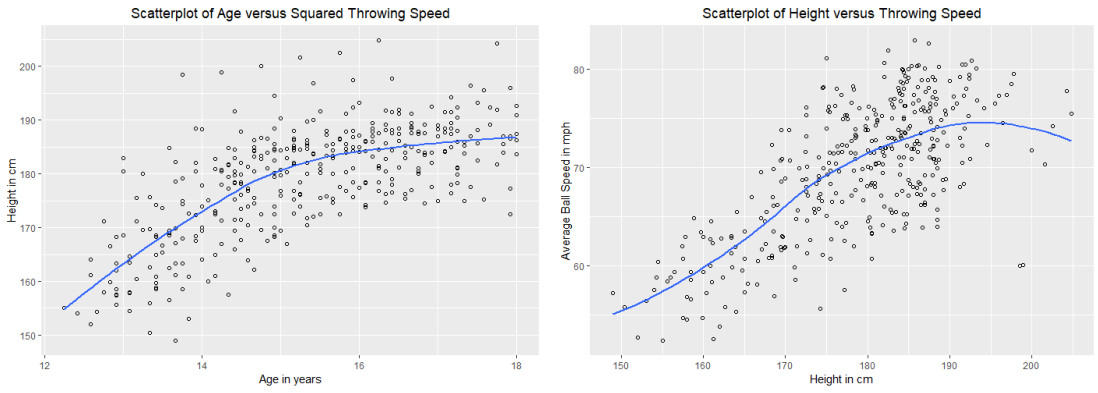
Here we see that there are very few players who have been measured between 17.5 to 18 years old, and relatively many athletes who have been measured between 13.5 and 16 years of age. When examining the standard deviations of the average ball speeds, we see that there is more variation between the throwing speeds for players ages between 13.5 and 16 years old.

Age	max	min	mean	median	sd
12	155	154	154.5	154.50	0.707
12.5	171.1	152	160.6	159.90	5.32
13	183	150.4	165.9	165	8.44
13.5	198.5	149	170.1	168.75	10.3
14	198.9	157.5	175.3	175.30	8.83
14.5	200	162.2	179.0	180.30	7.67
15	201.6	167	181.6	182.50	7.10
15.5	202.5	172.5	183.6	184.1	6.32
16	204.8	173.8	184.4	184.75	6.44
16.5	192.5	174.5	184.8	185	4.84
17	196.5	175	185.4	186.5	5.58
17.5	204.2	172.5	187.1	187	7.65
18	192.7	183.7	188.2	187.4	3.62

Table 7.1: Means and variation of player height by age, ages divided in groups of 6 months

7.3 Effect of length on player performance

As mentioned in Section 7.1, there seems to be a certain age at which the pitchers' speed does not seem to increase as much as previously. This same trend is visible when looking at the height of the pitchers when they age.



(a) Scatter plot of pitchers' age and height (b) Scatter plot of height and throwing speed

Figure 7.5: Scatterplots of relationship between height and age and of height and throwing speed

One curious thing to note is that the loess curve goes down in Figure 7.5b as the players reach a height of 195cm or more. This is a bit unexpected, but most likely due to the two data points in the right lower corner, pitching 60 miles per hour despite their large height. Both these points are from the same player, who is incidentally also the cause of the outliers from Figure 7.5a at the ages 13.5 through 16. The two outliers in Figure 7.5b are thus caused by a young player

that has a low speed in relation to his height, but not in relation to his age.

The age at where growth slows down in Figure 7.5a seems to be related to the age at which the curve in Figure 7.1b changes. Perhaps this is not unexpected as this age coincides with a period in which males no longer seem to grow as rapidly; however, it could also mean that the length of the pitcher is more influential on the throwing speed than at first expected.

To research this hypothesis we shall look at the effect of height on average ball speed. We can compare the lengths of the players to the sample medians, and determine whether the taller players also perform better at pitching.

(a) Blue indicates above median length, red below median.

(b) Percentage deviation from mean length for age group, lighter curves are taller

Figure 7.6: Growth curves for throwing ball speed, differentiated by length of player.

Although it is hard to see from Figure 7.6, the figure does suggest that taller players throw faster on average than shorter players. It has to be noted that not every pitcher can consistently be catagorised as tall or short. Some players had an early or late growth spurt, meaning that they were catagorised as either short or tall but did not remain in this group throughout their career. However, when looking at the group means, we nonetheless see a significant difference.

Age	Shorter	Taller
12.0	**57.55556**	52.40000
12.5	56.78889	**57.64286**
13.0	59.80667	**63.62500**
13.5	60.67712	**66.38268**
14.0	66.11481	**68.10270**
14.5	67.54933	**70.98956**
15.0	70.15635	**74.04497**
15.5	71.45000	**74.14392**
16.0	70.92910	**74.72235**
16.5	**74.75556**	74.03513
17.0	**75.60625**	74.58100
17.5	74.33662	**76.00000**
18.0	77.66667	**78.70000**

Table 7.2: Group means for throwing speeds separated by age and length

Table 7.2 shows again that for most age groups, the taller players do outperform the shorter players. It has to be noted that for age group 12.0 there are only two observations, and thus the discrepancy between this age group and the others can be explained by randomness. However, the discrepancy at age 16.5 and 17.0 is quite curious, and cannot be explained by a low sample size. We can also look at the joint influence of height and age with a three dimensional scatter plot.

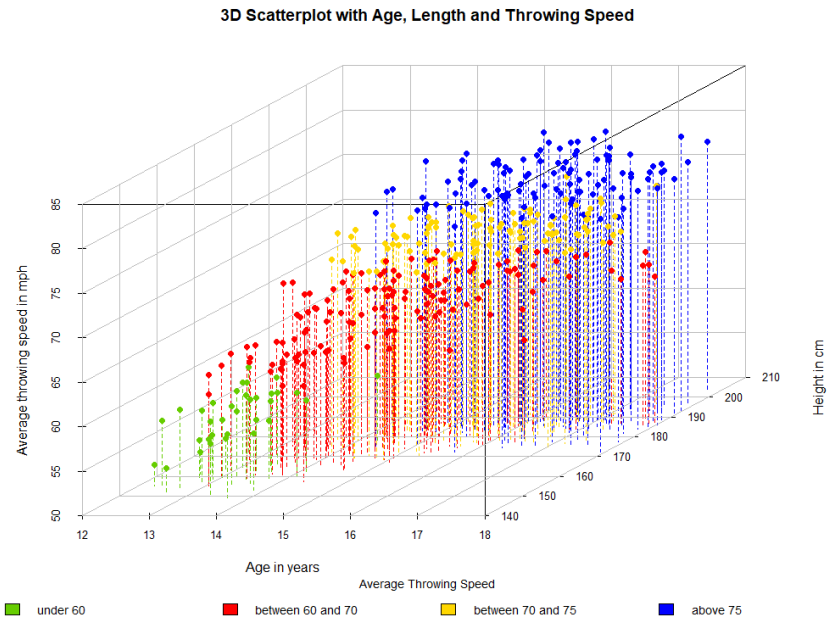


Figure 7.7: 3D scatter plot of height, age and throwing speeds, divided by throwing speed.

Again we can conclude from looking at Figure 7.7 that both the age and height of the pitcher are correlated with throwing speeds.

8 Growth curve modelling

In the social sciences, growth curve modelling is a popular tool for explaining change over the course of time. Despite their popularity, growth curves are not straightforward to model, especially when dealing with hierarchical data. There are many interesting ways of dealing with growth curves, although not all methods are applicable to the data set from project Fastball. In this chapter, we will consider two methods that work especially well for our data, and compare them with a simple linear regression.

8.1 Linear multivariate regression model

The simplest and easiest model is a linear regression model, which we can use to benchmark our other models. Here we treat the observations of the individual pitchers as equal amongst all other observations, meaning that the model just incorporate the non-nested data points and does not factor in which participant generated which data.

One important thing to note is that for our data set, the typical requirements for regression are not satisfied. We do not have independent observations, although we will treat them as such. Therefore, the error terms will not be independent, which is a standard assumption for these kinds of models. Furthermore, because we no longer take into account what participant generated which data, there is a bias towards participants that were able to be measured more than once.

As observed in Figure 7.1a, the linearity of the data set seems to improve by squaring the average ball speed (ABS), so we shall first transform the data. Then we simply generate a multivariate regression model in `R`. For the results of these fits, please see Section B.1.

$$Y_{ij} = \boldsymbol{X_{ij}}^T \boldsymbol{\beta_j} + \varepsilon_{ij}, \tag{8.1}$$

Where Y_{ij} is the squared throwing speed and X_{ij} includes all possible covariates that have been measured. The results of this regression are summarised below.

	Estimate	**Std. Error**	**t-statistic**	**p-value**
(Intercept)	-605.878	729.930	-0.830	0.4070
Age[1]	213.945	28.560	7.491	**5.10E-13**
Height	19.328	4.497	4.298	**2.21E-05**
Weight	9.407	3.442	2.733	**0.0066**
Range of Motion (IR)	-10.863	2.429	-4.474	**1.02E-05**
Range of Motion (ER)	1.737	1.564	1.111	0.2675
Force (IR)	2.272	1.070	2.123	**0.0344**
Force (ER)	4.030	1.162	3.469	**0.0006**

Table 8.1: Coefficient estimates with p-values

The adjusted R^2 of this model is 0.67, based on 377 observations. The AIC is 5808.828, BIC is 5844.218 and the MSE is 274,418. We see that all but one of our predictors is statistically

[1]Since we have pitchers aged between 12 and 18, we first recenter the data by subtracting 12

significant. However, after further investigation it seems that more assumptions for regression have been violated.

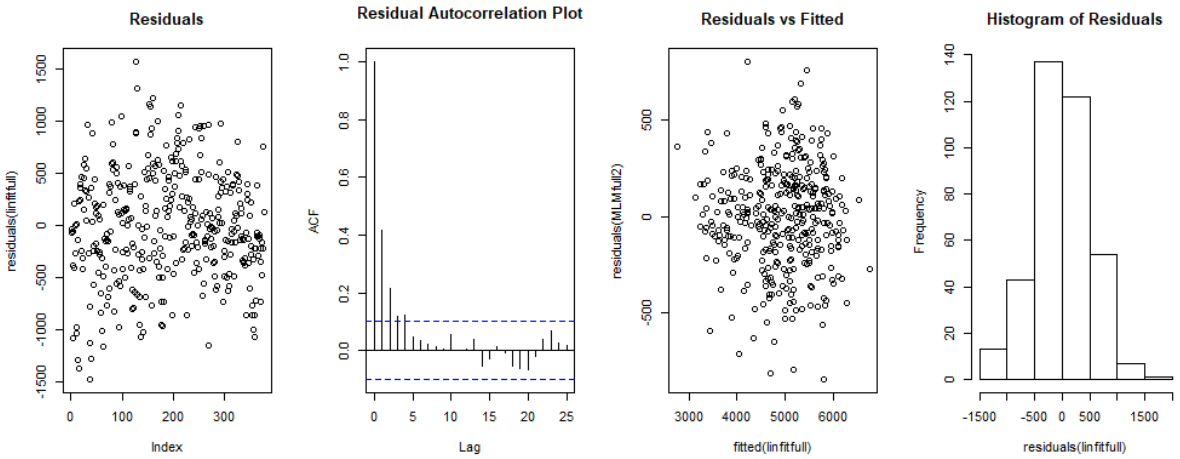


Figure 8.1: Residuals, autocorrelation between residuals, residuals versus fitted values and distribution of residuals for Model (8.1)

Figure 8.1 shows some clear issues with the model: not only are the residuals autocorrelated, but their absolute value also seem to increase as the prediction increases. This can be explained by the high variance of players at higher ages, as noted in Section 7.2 and visible in Figure 7.3c. While the younger players do not have a large spread in performance, the older players differ much in average ball speed. When fitting a linear regression model, this naturally results in a larger spread for higher fitted values. We can also assess the validity of using a linear model by plotting the component + residual plot:

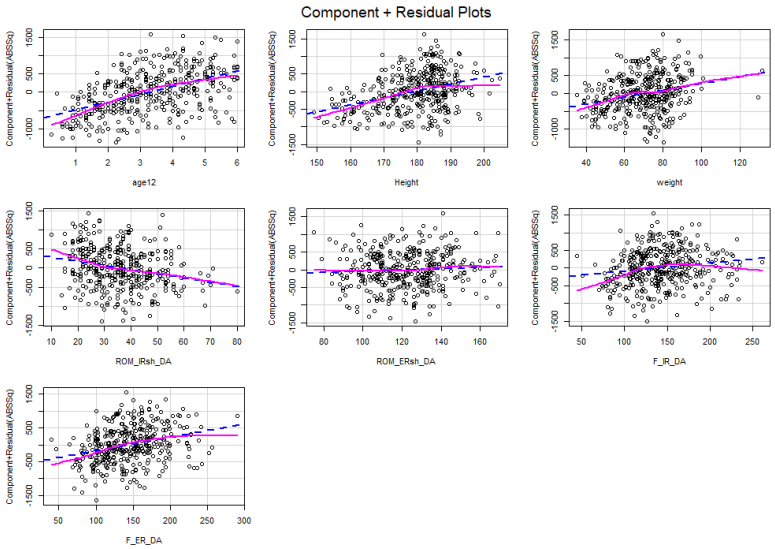


Figure 8.2: Component+residual plots for the linear regression Model 8.1

We see here that the assumption of a linear trend between the response variable and the covariates seems unrealistic for most covariates. There are several methods of dealing with these

kinds of relationships between variables. Instead of using a nonlinear model, we can try adding a squared age term to improve the fit of the linear model. The resulting model is

$$Y_{ij} = \boldsymbol{X'_{ij}}^T \boldsymbol{\beta_j} + \varepsilon_{ij}, \tag{8.2}$$

where X'_{ij} now includes the squared age term to improve model fit. We again fit this model in `R` and find that the adjusted R^2 remains 0.68, but the MSE of this model is reduced to 264,750.

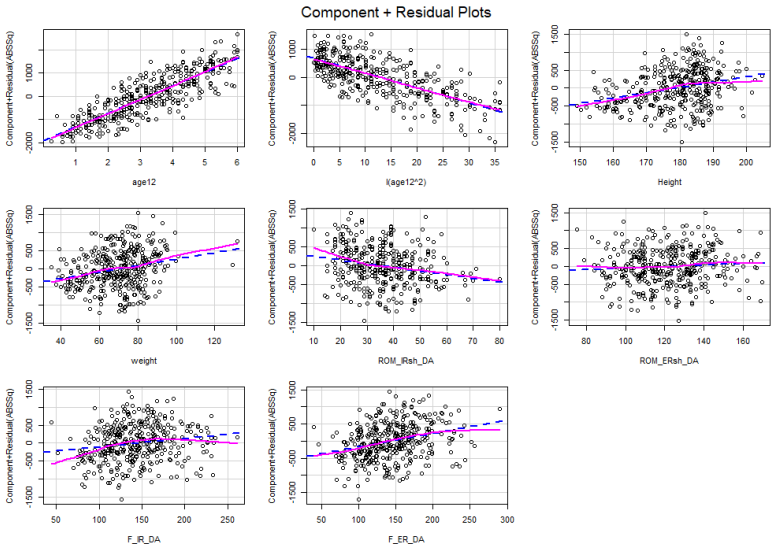


Figure 8.3: Component+residual plots for the linear regression Model (8.2)

The component + residual plot also improved for the age term: it seems adding a squared age term sufficiently improves the linearity for these covariates. Nevertheless, neither the non-linearity problems with many of the other covariates have been fixed, nor the problem concerning autocorrelation and heteroscedasticity, as is visible below:

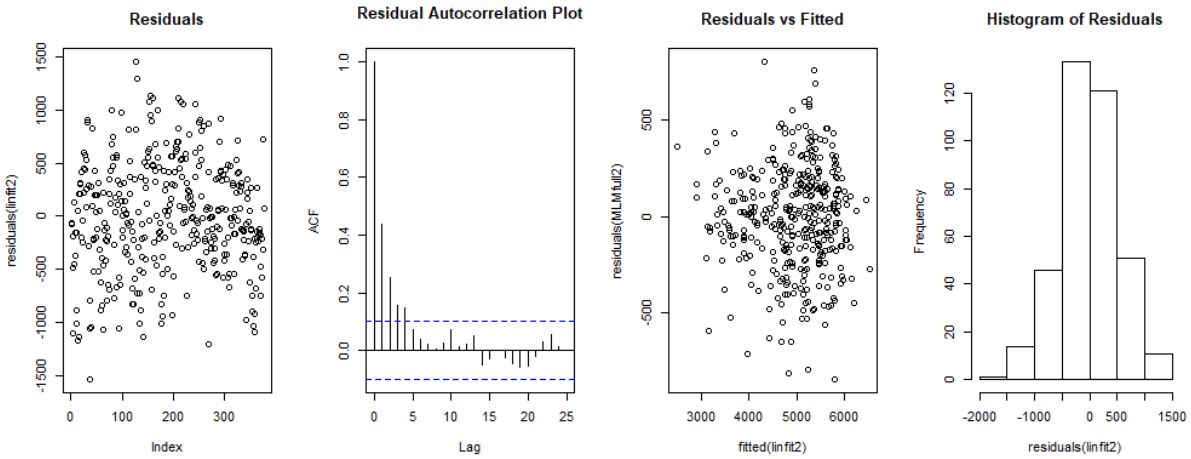


Figure 8.4: Residuals, autocorrelation between residuals, residuals versus fitted values and distribution of residuals for Model (8.2)

Most of these issues can be addressed: we could use a weighted-least-squares regression to deal with the non constant error variance, we could switch to a generalized least squares estimation to allow for autocorrelated errors or we could transform the response variable or covariates to improve the linearity between the ball speed and the predictors [18]. However, as stated above, even with these adaptations we still violate multiple assumptions for linear regression.

8.1.1 Separate age dependent modelling

Although by squaring the ABS and adding a squared age term we improve the linearity, we do not fix the real problem with the data. Around the age of 15, we see a clear shift in the growth curve of the pitchers. The slope of the regression line, and therefore the predicted growth, seems to slow down after this age. Because of this change point, quadratic transformations for improving the linearity will never yield the desired effect. Therefore we will try modelling the two growth curves separately as mentioned in Section 7: one between 12 and 15 years old, the other between 15 and 18 years old.

Oddly enough, although for the pitchers between 12 and 15 years old the R^2 of the simple regression model stays roughly the same (0.66), for those between 15 and 18 years old the regression model does not seem to predict ABS well: the R^2 drops to 0.29. This could be explained by the high spread between players of ABS at those ages, as can be seen in Figure 7.3c.

We can thus conclude that a linear model is not sufficient for modelling our data. We will therefore focus on multilevel models as a more suitable structure for the data set.

8.2 Multilevel model

Multilevel models are a way to model data that is hierarchically structured. They combine the information at different levels and thus handle data at multiple levels more effectively. In our case, we can structure the data in such a way that we get a (non-linear) 2-level model, with the individual observations of each pitcher on the lower level, and the general growth curve per individual on the second level. Using this method, we hope to obtain a model that uses both the individual growth and the general shape of the growth curves in a meaningful way. A great book on modelling multilevel and mixed effects models is written by Pinheiro and Bates [34], whom also authored the `R` package we will be using throughout this thesis. We use the package `nlme` [35]; for the results of the multilevel modelling, please see Appendix B.

8.2.1 Mixed models based on age

We start with a model similar to a multivariate regression model. This model is also called the random intercepts model, as only the intercept is considered a random effect in this model. The model is defined as following:

$$\begin{aligned} Y_{ij} &= \beta_{0j} + \beta_1 a_{ij} + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \end{aligned} \tag{8.3}$$

where Y_{ij} is the throwing speed of individual j at measurement i and a_{ij} the age, and the error terms δ_{ij} and ε_{0j} are normally distributed with mean zero but with different variance. This model resembles a standard regression model, except for the addition of a random error term

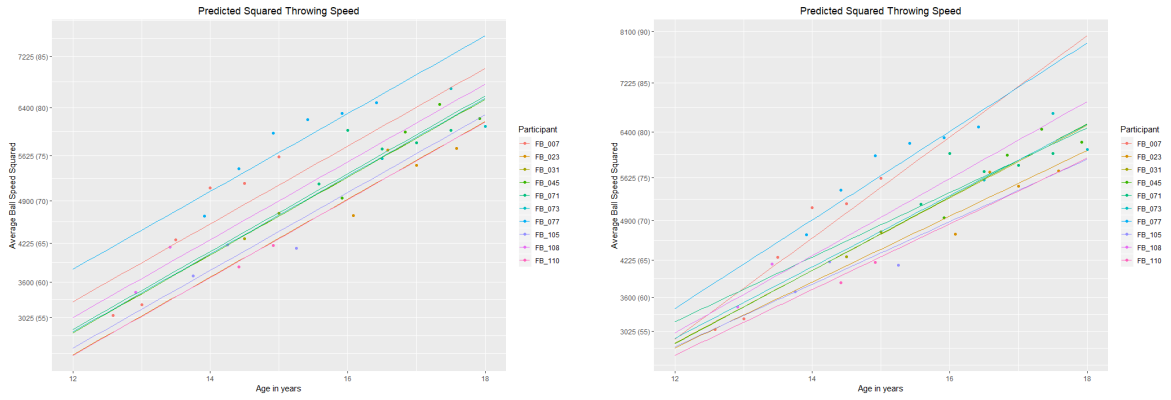
ε_{0j} for β_{0j} which makes it a random intercept. This means that the rate at which an individual grows is equal amongst the population, but every individual starts at a different "base" speed. See Figure 8.5a for a visual representation of this model.

This model supposedly does a better job of modeling the ball speed than our fixed-effects multivariate regression model, and when we look at the mean squared error this seems to hold true: the MSE is reduced from 264,750 to 87,191. This is a vast improvement, but adding more flexibility to the model might improve the fit even more. We do so by adding a random slope to the model: where β_1 used to be fixed for all j, we can make this effect a random effect as well and bring more flexibility to the model:

$$\begin{aligned} Y_{ij} &= \beta_{0j} + \beta_{1j} a_{ij} + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \\ \beta_{1j} &= \gamma_{10} + \nu_{1j}. \end{aligned} \tag{8.4}$$

Here both the initial speed and the rate at which individuals increase in speed are random, and take on different values for each individual. This added flexibility seems to improve the model fit as well, reducing the MSE from 87,191 to 71,500. As the standard deviation for the squared ball speed itself is 921, this prediction seems quite accurate.

We randomly choose ten participants to help visualize the model predictions and compare them with the data. The results are visible below:



(a) Random intercept Model (8.3)

(b) Random intercept and slope Model (8.4)

Figure 8.5: Results for fit of squared average ball speed for 10 of the 114 participants

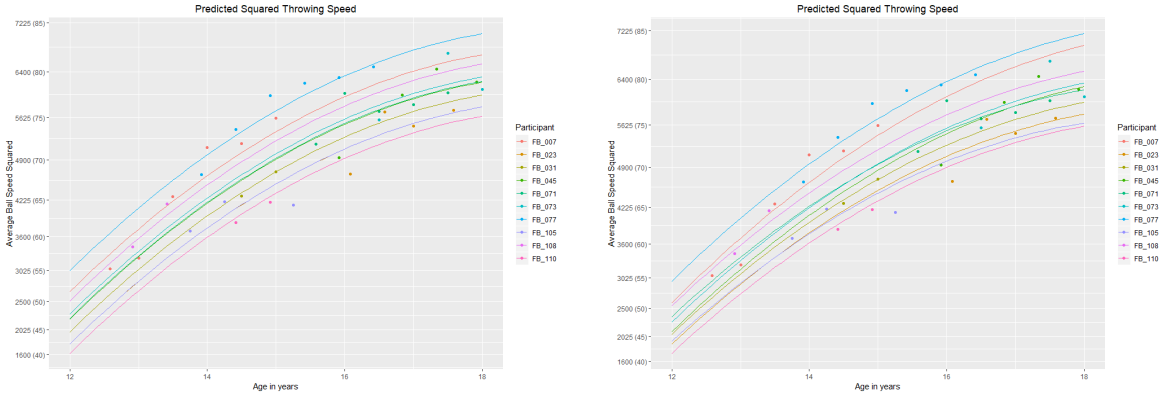
As stated in Section 8.1, a linear model does not seem to fit well with our observations. When we only include age as a predictor variable, we can improve the fit by including quadratic or higher order terms of age; that is, we include the squared age as a separate term:

$$\begin{aligned} Y_{ij} &= \beta_{0j} + \beta_1 a_{ij} + \beta_2 a_{ij}^2 + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \end{aligned} \tag{8.5}$$

We can also include a random slope like in Model (8.4), which might again add some flexibility to the model.

$$
\begin{aligned}
Y_{ij} &= \beta_{0j} + \beta_{1j} a_{ij} + \beta_2 a_{ij}^2 + \delta_{ij}, \\
\beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \\
\beta_{1j} &= \gamma_{10} + \nu_{1j}.
\end{aligned} \tag{8.6}
$$

The log-likelihood of both models is comparable, suggesting that the addition of random slopes is not leading us to a better fit. However, the MSE is reduced from 74,704 for Model (8.5) to 68,833 for Model (8.6). In fact, because the model is now more complex without increasing much in predictive power, we can consider this model as worse than Model (8.5). We will see later that this intuition holds true when considering model selection criteria; for a more detailed comparison between these and other models, see Section 8.3. Below the fit for both models is again shown for the ten participants.

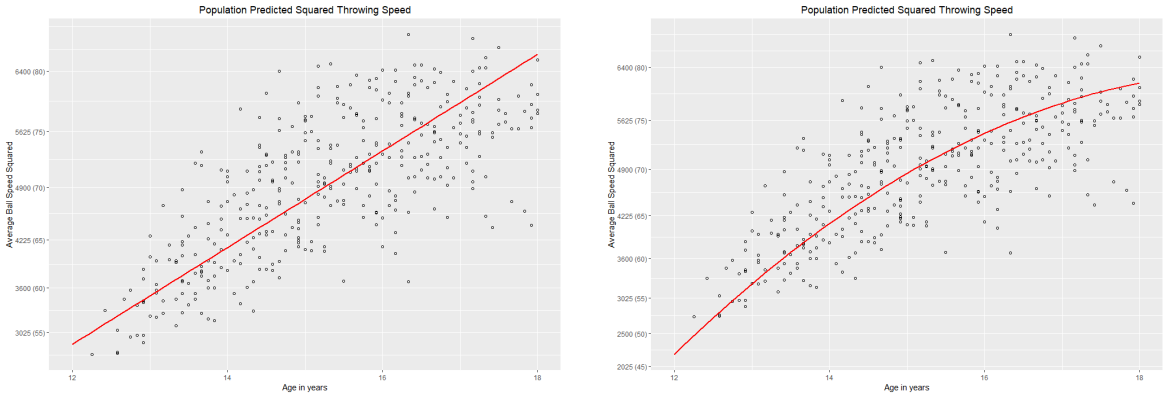


(a) Random intercept Model (8.5)

(b) Random intercept and slope Model (8.6)

Figure 8.6: Results for fit of ABS for 10 of the 114 participants, adding the squared age as a predictor.

Although Figure 8.6b shows some more flexibility in the fit compared to Figure 8.6a, this difference is not very noticeable. It seems including a random slope is therefore not worth the added complexity. However, overall the addition of a squared age term seems to improve the model fit, both for individual predictions as for population predictions, as can be seen below.



(a) Population predictions of the Model (8.4)

(b) Population prediction for Model (8.6)

Figure 8.7: Population predictions of the Models (8.4) and (8.6)

As stated above, it is obvious that the linear model that only incorporates age is not sufficient for prediction the ball speed well. Adding the squared age seems to improve this fit. However, we have more data available than just the age of the pitcher, and we can see if incorporating these covariates improves our basic model.

8.2.2 Including other covariates

The first model is formulated as follows

$$\begin{aligned} Y_{ij} &= \beta_{0j} + \beta_{1j} a_{ij} + \boldsymbol{X_{ij}}^T \boldsymbol{\beta_j} + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \\ \beta_{1j} &= \gamma_{10} + \nu_{1j}, \end{aligned} \tag{8.7}$$

where Y_{ij} is the throwing speed of individual j at measurement i, $\boldsymbol{X_{ij}}^T \boldsymbol{\beta_j}$ is the usual linear regression with fixed effects β_j for the covariates in X_{ij}, namely age, height, weight, force of internal and external rotation and the internal and external range of motion. The results of this fit can be found in Section B.3.

The complete model also includes the quadratic age term, but not the random slope; both of these increase the model complexity, but while the quadratic age term significantly increases the goodness of fit, the random slope does not and is thus not included. The full model is thus:

$$\begin{aligned} Y_{ij} &= \beta_{0j} + \boldsymbol{X'_{ij}}^T \boldsymbol{\beta_j} + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \end{aligned} \tag{8.8}$$

where the only change is that X'_{ij} now includes the age and quadratic age term as well.

8.3 Comparing models

One way to compare models is by applying a model-selection criteria. A common group of criteria is the general family of penalized model-fit statistics for regression models fit by maximum likelihood, taking the form

$$-2\log(L(\hat{\boldsymbol{\theta}}_j)) + cs_j, \tag{8.9}$$

where $L(\hat{\boldsymbol{\theta}}_j)$ is the maximized likelihood under model M_j with the fitted model parameters $\hat{\boldsymbol{\theta}}_j$ and s_j regression coefficients, and c is a constant. The first term $-2\log(L(\hat{\boldsymbol{\theta}}_j))$ is the residual deviance under the model; when dealing with a linear model with normal errors, this reduces to the sum of squares.

The c in Equation (8.9) is a constant that differs for each statistic. Two statistics are used most commonly: the Akaike information criterion (AIC) and Bayesian information criterion (BIC). They are of the form:

$$AIC_j = -2\log(L(\hat{\boldsymbol{\theta}}_j)) + 2s_j, \tag{8.10}$$

$$BIC_j = -2\log(L(\hat{\boldsymbol{\theta}}_j)) + s_j \log(n), \tag{8.11}$$

with n the number of observations. The overall magnitude of these criteria does not say much about the models, but the most information can be gained from the differences between models. Smaller values are better; the model with the smallest AIC and BIC value is most supported by the data [18, Section 22.1]. If we compare our models using the Akaike information criterion and Bayesian information criterion, we find the following:

	AIC	**BIC**
Model (8.2)	5797.306	5836.628
Model (8.3)	5732.322	5748.051
Model (8.4)	5716.741	5740.334
Model (8.5)	5684.056	5703.717
Model (8.6)	5686.106	5713.631
Model (8.7)	5646.827	5697.946
Model (8.8)	**5645.460**	**5689.715**

Table 8.2: AIC and BIC values for the models in Section 8.1 and 8.2

Again, for the resulting fits of these models, see Appendix B. While Model (8.5) is significantly better than Models (8.3) and (8.4) - as expected - it also seems to perform better compared to Model (8.6). While we might expect a model with more complexity to model the true growth better, this is not always the case. This is due to the model relying more on the training data when increasing the complexity; this reduces the bias but increases the variance, and can lead to overfitting [24, Section 7.2]. This same behaviour is visible when excluding the random slope term in the full model; while the AIC value stays roughly the same, the BIC value for the less complex model is lower (and thus better).

As mentioned above, the model with the lowest AIC score is most preferable. We therefore choose Model (8.8) to be our model for predicting the ball speed. Before using this model, we can quickly check the residuals to see whether or not our model assumptions are violated:

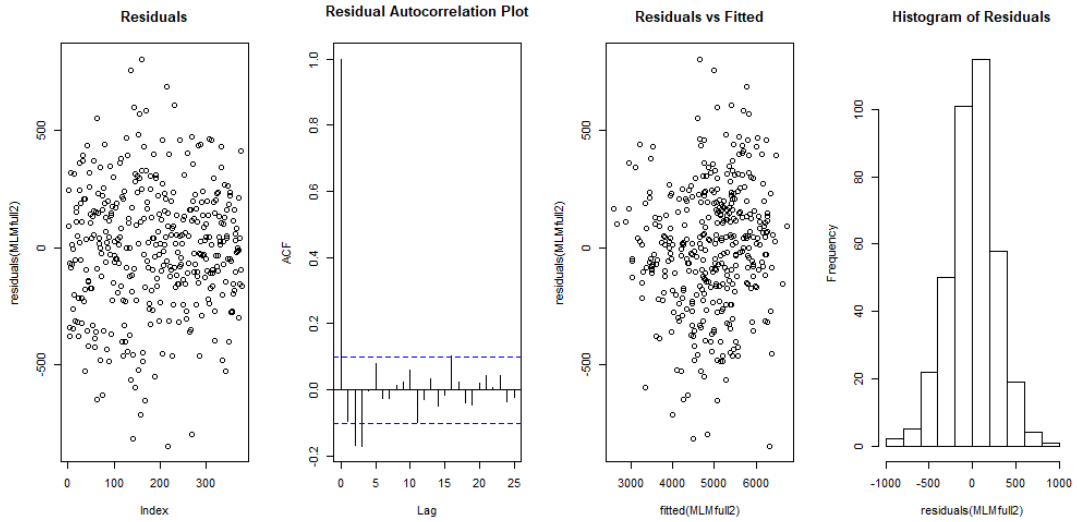


Figure 8.8: Residuals, autocorrelation between residuals, residuals versus fitted values and distribution of residuals

None of these plots suggest that we have any serious issues regarding assumptions on the residuals, although the "Residuals vs Fitted" plot could suggest some heteroscedasticity similar to the linear regression model we fit in Section 8.1.

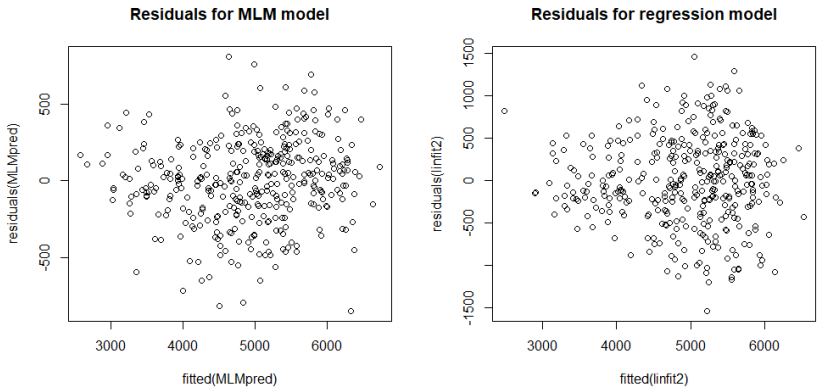


Figure 8.9: Comparison of heteroscedasticity for the linear regression (8.1) and multilevel (8.8) model

By visual inspection, the heteroscedasticity seems to have decreased slightly from regression Model (8.2). To test whether the heteroscedasticity has indeed been reduced, we perform the Levene's test [7] on a discretized version of our model. We divide the data into nine bins of the squared ball speed of size 500 and use these for the Levene's test; the resulting $F_{(8,368)}$ test statistic is 0.512 with corresponding p-value 0.8477, indicating that heteroscedasticity is no longer an issue for this model.The biggest improvement has been made in regards to the autocorrelation of the residuals: whereas for Model (8.1) the residuals were clearly autocorrelated, this no longer seems to be a big issue.

8.4 Correlation and collinearity

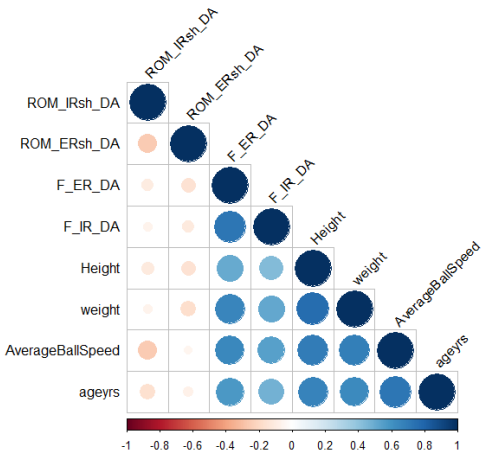


Figure 8.10: Correlation plot for the covariates in our data set

Due to the nature of our data, correlation and collinearity are issues to take into account. One quick method of assessing the levels of correlation between the covariates is to plot the correlation matrix; this is shown in Figure 8.10, which has been generated with the package `corrplot`[52].

Most of these variables are naturally quite correlated due to the fact that they are age-dependent; this is quite obvious for height and weight, but the same is true for the force or range of motion of the athlete's rotation. We can take this effect into account by calculating the partial correlation for the variables with age as the dependent variable. The partial correlation $\rho_{XY \cdot Z}$ between X and Y given Z is given by the correlation of the residuals e_X and

e_Y when computing the linear regression of X and Y with Z. Computing the partial correlation given age gives us the following table:

	Weight	**Height**	**ROM**$_{IR}$	**ROM**$_{ER}$	**F**$_{IR}$	**F**$_{ER}$	**ABS**
Weight	1	0.6018	0.412	0.2105	0.1901	-0.1464	-0.0115
Height	0.6018	1	0.4236	0.4626	0.3158	-0.1694	0.0564
ROM$_{IR}$	0.412	0.4236	1	0.3994	0.3379	0.0044	-0.2032
ROM$_{ER}$	0.2105	0.4626	0.3994	1	0.624	-0.1313	-0.0057
F$_{IR}$	0.1901	0.3158	0.3379	0.624	1	-0.0858	0.0218
F$_{ER}$	-0.1464	-0.1694	0.0044	-0.1313	-0.0858	1	-0.2747
ABS	-0.0115	0.0564	-0.2032	-0.0057	0.0218	-0.2747	1

Table 8.3: Partial correlation between covariates when accounting for age

A better way of assessing the collinearity of our data is assessing the variation inflation factor for our models. This factor indicates the level of collinearity between factors, where a value of 1 indicates an absence of collinearity. In practice, a value over 5 or 10 is considered problematic [25]. When assessing the VIF values, we see that for Model (8.8) the only values of concern are those for the age and squared age term: this is of course no surprise, and not something we can easily remedy if we still want to include the squared age term.

Age	**Age**^{2}	**Height**	**Weight**	**ROM**$_{IR}$	**ROM**$_{ER}$	**F**$_{IR}$	**F**$_{ER}$
23.084486	16.051197	3.795912	3.150706	2.316651	1.631548	1.088724	1.208339

Table 8.4: VIF-values for full multilevel Model (8.8)

If we compare the results of the multilevel model with Model (8.1), the simplest regression model we created in Section 8.1, we see that for this model the VIF value for age is much lower, indicating that the high VIF value is most likely due to the inclusion of the squared age term.

Age	**Height**	**Weight**	**ROM**$_{IR}$	**ROM**$_{ER}$	**F**$_{IR}$	**F**$_{ER}$
2.113969	2.848399	3.259818	1.124998	1.130245	2.125191	2.855665

Table 8.5: VIF-values for simple regression Model (8.1)

We could however always try to reduce the multicollinearity of the model by excluding some of the covariates that are highly correlated; for example, the range of motion for the external rotation is quite correlated to the force of the internal rotation after accounting for age the model to

$$\begin{aligned} Y_{ij} &= \beta_{0j} + \boldsymbol{Z_{ij}}^T \boldsymbol{\beta_j} + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \end{aligned} \tag{8.12}$$

where Z_{ij} contains the age, squared age, height, weight and force and range of motion of the external rotation. The AIC and BIC values remain roughly the same: AIC goes from 5646.460 to 5646.455, and BIC is reduced from 5689.715 to 5681.846 due to the reduction in parameters.

The MSE is also reduced by 1,111. Not much is improved for the multicollinearity of the model, as can be seen in Table 8.6, but this model might be useful for prediction purposes, which we will discuss more thoroughly in Section 9.3.

Age	**Age**^{2}	**Height**	**weight**	$\mathbf{F}_{ER}$	$\mathbf{ROM}_{ER}$
22.240035	15.627946	3.836361	3.163571	1.828059	1.031460

Table 8.6: VIF-values for reduced multilevel Model (8.12)

9 Predicting player quality

Whereas in Chapter 8 we have tried to model the true growth curve related to the quality of the player, we are now interested in predicting the quality of a player over time. This is mainly useful for scouting future players when they are young; if we have a reliable way to predict their success, we can make a better decision on whether we include this player for the national team.

9.1 Issues with prediction

The main problem for using multilevel or mixed effects models is that the random effects cannot be estimated when no response variable has been recorded. Therefore, the throwing speed cannot be estimated on the level of the individual and instead the population estimates are used. Our mixed model then effectively reduces to a fixed effects model. This means that in order to obtain an accurate prediction of the throwing speed of a pitcher at a later age, we first need to measure his performance at least once, including all the covariates such as force and range of motion. If we have one or two of these data points, we can then create a new mixed model for the player.

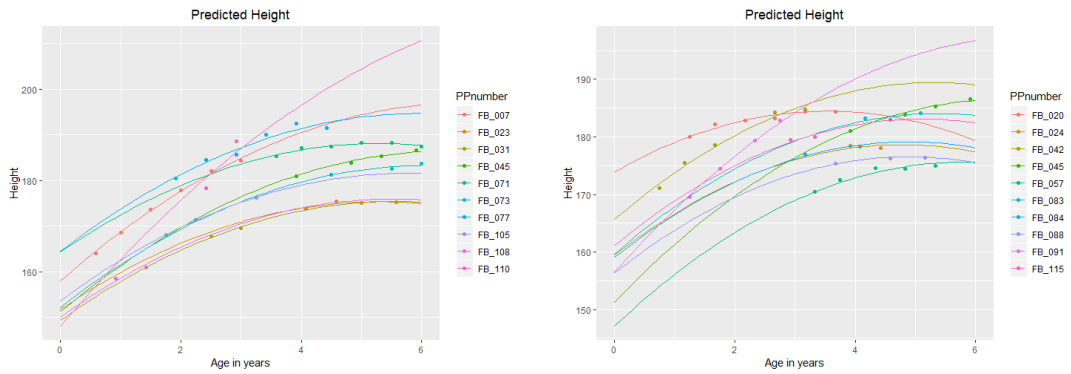
This however leads to another problem. If we want to use the model decided in Section 8.3, we will need to know how tall and heavy the player is at a later age, along with harder to predict measurements such as the range of motion. One method for dealing with, in a sense, "missing data" is to use imputation to compute the missing variables.

9.2 Imputation

We start off by modelling the height of the players based on age alone. As age is the only known covariate when modelling the ball speed at a later age, we cannot include other predictors to improve our fit. We first consider a mixed effects model like the ones discussed in Section 8.2.1:

$$\begin{aligned} H_{ij} &= \beta_{0j} + \beta_{1j} a_{ij} + \beta_2 a_{ij}^2 + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \\ \beta_{1j} &= \gamma_{10} + \nu_{1j}. \end{aligned} \tag{9.1}$$

However, this leads to problems as visiualised below:



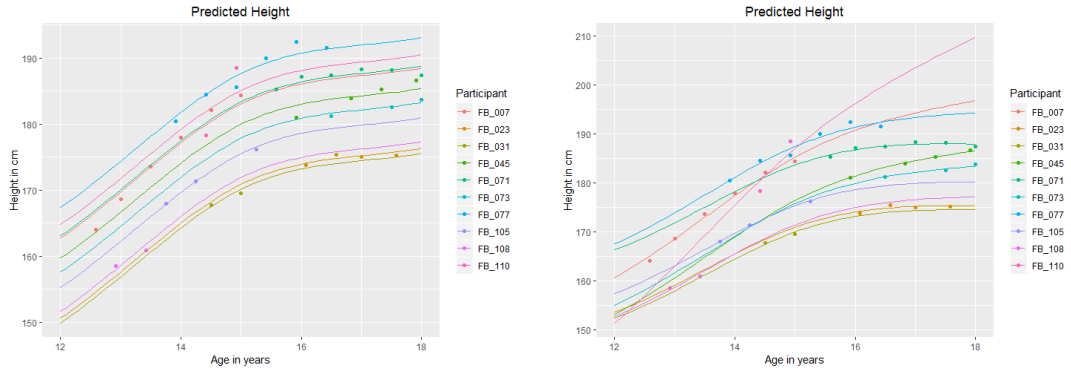
(a) Model for standard subset of players

(b) Model for different subset of players

Figure 9.1: Results for fit of height for players using Model (9.1) for different subsets of players. The horizontal axis shows the age from 12 years old.

As visible in Figure 9.1a and 9.1b, this method has several issues. Most prominently, if there is only data from a very young age, this model seems to overestimate the height in the future. This is most visible in Figure 9.1a for participant 110 (the pink line), for whom the model predicts a height of 210 cm. When taking a different sample from the participants and plotting the model fit, some other issues become visible; this is most visible for participant 20 (red line), whose predicted height decreases after age 16 and who is predicted to be as tall at age 18 as he was at age 13. It should be obvious that this model does not predict length well.

For modelling the length we try a different approach: we use a spline instead of adding a quadratic term. For modelling this in `R`, we use the package `splines` which is part of the `R` base [40]. We still use the mixed effects framework to allow for differences between players, and incorporate both a random intercept and a random slope.



(a) Model fit for height prediction using splines and random intercept

(b) Model fit for height prediction using splines and random slope and intercept

Figure 9.2: Results for fit of height for players using splines in the linear mixed effects model, for different subsets of players.

While the model visualised in Figure 9.2b follows the data more nicely and is preferred when looking at the AIC and BIC values, the model in Figure 9.2a is a better representation of the type of model needed. This is because the end goal for this research is to predict throwing speed for older athletes at a younger age; while both models fit a nice curve to the growth data of individuals, it seems the model for Figure 9.2b is more prone to overestimation when the only recorded data is at a young age. This is because before the age of 15 the speed at which boys grow in length is significantly higher than the speed after the age of 15, and thus when only data before then is collected, the model assumes this growth continues after 15, which is usually not the case.

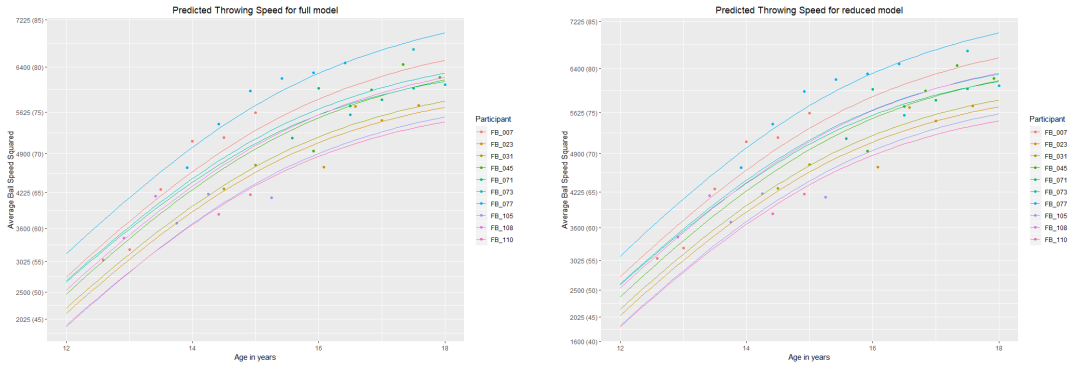
We therefore have found a suitable mixed effects model for the height, and continue modelling the covariates using only age and height as predictors. The models are as follows:

$$
\begin{aligned}
\text{Weight:} \quad W_{ij} &= \beta_{0j} + \beta_{1j} a_{ij} + \beta_2 H_{ij} + \delta_{ij}, && (9.2)\\
\beta_{0j} &= \gamma_{00} + \varepsilon_{0j},\\
\beta_{1j} &= \gamma_{10} + \nu_{1j}.\\
\text{F\_IR\_DA:} \quad (F_{IR})_{ij} &= \beta_{0j} + \beta_1 a_{ij} + \beta_2 H_{ij} + \delta_{ij}, && (9.3)\\
\beta_{0j} &= \gamma_{00} + \varepsilon_{0j},\\
\text{F\_ER\_DA:} \quad (F_{ER})_{ij} &= \beta_{0j} + \beta_1 a_{ij} + \beta_2 H_{ij} + \delta_{ij}, && (9.4)\\
\beta_{0j} &= \gamma_{00} + \varepsilon_{0j},\\
\text{ROM\_IRsh\_DA:} \quad (ROM_{IR})_{ij} &= \beta_{0j} + \beta_1 a_{ij} + \beta_2 H_{ij} + \delta_{ij}, && (9.5)\\
\beta_{0j} &= \gamma_{00} + \varepsilon_{0j},\\
\text{ROM\_ERsh\_DA:} \quad (ROM_{ER})_{ij} &= \beta_{0j} + \beta_1 a_{ij} + \beta_2 H_{ij} + \delta_{ij}, && (9.6)\\
\beta_{0j} &= \gamma_{00} + \varepsilon_{0j},
\end{aligned}
$$

Where H_{ij} is the height of player i at moment j. We tried to simplify the models wherever possible; as you can see, most models do not include a random slope because it either did not significantly improve the model, or actively decreased the AIC or BIC value for the fit. These models will be used to predict future values for the covariates needed to compute the ball speed at age 18.

9.3 Reducing the number of predictors

As we could see in Section 8.4, we could reduce the amount of predictors by excluding the force and range of motion of the internal rotation, without losing much prediction power. As can be seen from Figure 9.3, there does not seem to be a dramatic difference in how these models fit the squared ball speeds. To see if this still holds in the case of prediction, we will compare the predictions using cross validation.



(a) Model fit for ABSSq, full model

(b) Model fit for ABSSq, reduced model

Figure 9.3: Fitted squared ball speeds for different subsets of players, using the full and reduced model.

We use 2-fold cross validation at first: we divide the data in a training set and test set, and predict for the test set by including the first measurement for this participant in the training set so we can make predictions on the level of the individual. We do not use imputation but simply

use the observations we have in our data set. If we now compare the MSE, we find on average 143,479 for the full model and 140,619 for the reduced model, indicating that the reduced model again provides more accurate predictions. However, an added benefit to reducing the amount of predictors is that when using imputation, there is less error in the predicted covariates that can negatively impact the prediction for the ball speed.

To test this hypothesis, we use only the first measured observation for each pitcher to predict future ball speeds. The models we have created in Section 9.2 are fitted using the full data set but excluding all observations for the pitcher except the first measurement, and these models will in turn fit the covariates at a later age for the pitcher. This same method is used for fitting a model for throwing speed. The results of this leave-one-out cross-validation are similar to the 2-fold cross validation we performed previously: the MSE for the full model is 153,927 but that of the reduced model is 148,495.

While the value of some of the predictors in our model are quite cheap to come by, such as height and weight, the force and range of motion have to be measured using an expensive machine. Due to the cost of measuring these predictors and the fact that including these measurements do not seem to improve our fit or predictions, it seems justified to use the reduced model for predicting the ball speeds.

9.4 Final prediction

For the code used to model the individual predictions, please see Appendix C (Section C.2). The models we have created in Section 9.2 have made it possible to predict the throwing speed of pitchers at age 18 even when we do not know what physical characteristics the pitchers will have at that age. The only requirement is that the pitchers have been measured at least once, although the accuracy of the prediction should be improved when more data on the growth is available. Figure 9.4 shows these predictions using all available data; all data was used to fit the models (9.2) through (9.6), the final model (9.7) and the model for player height, after which the imputed values for the variables in the final model are used to predict speed.

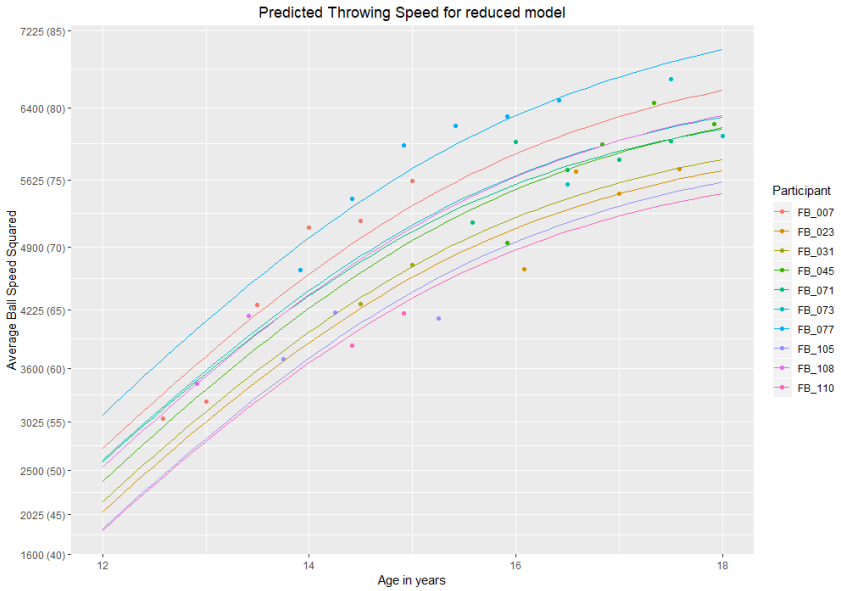


Figure 9.4: Fitted throwing speed using imputation for covariate prediction, individual level

The more data available, the better this model will be able to predict the true growth. When there is no data available for the pitcher the model can only predict on population level, which is shown below.

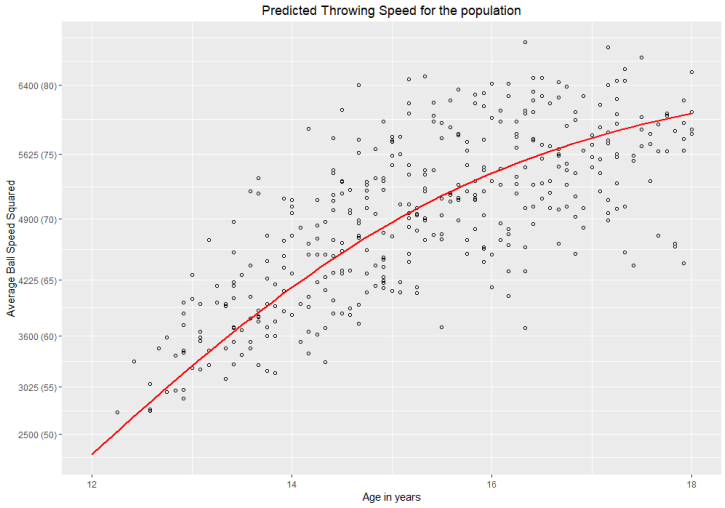


Figure 9.5: Fitted throwing speed using imputation for covariate prediction, population level

The final model has become

$$\begin{aligned} Y_{ij} &= \beta_{0j} + \beta_1 a_{ij} + \beta_2 a_{ij}^2 + \beta_3 H_{ij} + \beta_4 W_{ij} + \\ &+ \beta_6 (F_{ER})_{ij} + \beta_7 (ROM_{ER})_{ij} + \delta_{ij}, \\ \beta_{0j} &= \gamma_{00} + \varepsilon_{0j}, \end{aligned} \tag{9.7}$$

where the covariates can be imputed using the models in Section 9.2. The result of this fit can be found in Appendix B.4.

9.5 Remark on the simplicity of the model

The model we have used in Section 9.4 is a relatively simple multi-level or mixed-effects model. We have restricted ourselves to the use of linear models, while the data that was collected for the research is typically quite suited for non-linear models. However, while a more complex model would have been a better fit of the data, the interpretation of the final model is much more comprehensible than those of most non-linear mixed-effects models. Considering that this model will be used by sports professionals, sometimes with little knowledge of mathematics, this trade-off seemed reasonable.

10 Conclusion, discussion and recommendations for future research

This researched aimed to give insight into the value of data science for sports professionals, and fill gaps in current research, by analysing data sets from two different sources and with two different goals. There are two research questions to answer within this thesis, and so they will be answered and discussed seperately.

10.1 Classifying intensity during soccer practice

We set out to find whether it was possible to produce an algorithm that would be able to give insight into the intensity of an exercise by classifying sensor data. By extracting features from the data we were able to classify intensity using a decision tree, which had the highest overall accuracy of all methods. The decision tree had a classification accuracy of 96.2%, but had more difficulty classifying medium and high intensity than low intensity, resulting in an accuracy of 75% when considering medium and high intensity data.

10.1.1 Discussion and recommendations for future research

In many studies the three axes are are used separately to aid classification, but unfortunately due to the restriction on the gyroscope, we were unable to separate the axes properly. Therefore, only the length of the vector is used for classification, due to which a lot of information is lost. In coming research, additional attention should be paid to the settings of the sensors in order to be able to separate the acceleration in the axes, and therefore gaining more useful information for proper classification. This could increase the accuracy in the classification model as well.

Classification was based on single sensor input, rather than a combination of sensors. Classifying based on multiple sensors might prove to be more effective, although the precise relationship between the sensors has to be determined. Furthermore, there are more methods for classification that could have been considered, but were excluded from current research due to time restrictions. Popular methods are for example neural networks or hidden Markov chains, both of which could offer different perspectives than the ones offered in this thesis.

After-the-fact labelling of the data is both inaccurate and time intensive, especially considering there was no availability to the recorded exercise to check whether a drill was performed correctly. Perhaps part of the inaccuracy when classifying between medium and high intensity is born from this problem, although it is difficult to find how much of the inaccuracy in labelling bled through to the error rates in classification. In future research, it would be beneficial to the accuracy of labelling for the person working with and labelling the data to have access to all recordings.

Furthermore, defining clear moments for specific activities to occur during the experiment can be helpful for data analysis, as this creates a clear sample of the activity that is supposed to be detected from the data. In the case of soccer, an example of this would be for shoot/pass classification to ask the participant to shoot a number of times without doing anything else, thus creating a clear sample of what a recording of a shot looks like.

Lastly, more research has to be conducted in the field of human movement science to better understand and predict muscle load. Although the algorithm in this research was able to detect high and medium intensity activity, it needs more information on the impact of these activities on the muscles to give worthwhile feedback for injury prevention. Two of the methods used for classification have been developed by Schotel [44] as part of her research. However, as mentioned

in her research as well, the percentage zones in which the data is divided and the weightfactors used for these methods are currently a bit arbitrary. More research will hopefully reveal better boundaries for the intensity zones and corresponding weights to identify the true difference between different zones.

10.2 Predicting ball throwing speed for youth baseball pitchers

The aim of the research for project Fastball was to develop a method to predict throwing speed for pitchers in the selection of the under-18 team. More specifically, the research aimed to determine whether it was possible to model a growth curve to the performance of the pitchers during their development.

We concluded that mixed effects models seemed to be well suited for the given data and research question. We were able to develop a method that could predict throwing speed of pitchers between the ages of 12 years old and 18 years old, using multilevel models. Future predictions are made possible by unit imputation of all variables except age.

10.2.1 Discussion and recommendations for future research

Given at least one observation, it is possible to predict the throwing speed of any pitcher using imputation. We first modelled the height by using a combination of splines and multilevel models. This was due to the nature of the relationship between age and height: change in height is nonlinear over time, and growth slows after a certain age. Using splines is therefore a better way to model this changing relationship than a linear relationship alone.

Imputation using height was chosen because the recorded properties have a nonlinear relationship with age, similarly to the nonlinear relationship of height. Using both height and age would be a better predictor of the other covariates than just age alone if we want to fit the relationship using only linear models. However, if relaxing the linearity constraint, splines, for example, could have been used to better approximate the true relationship between age and other covariates, thus eliminating the need for height as a predictor and reducing the potential error in these predictions.

One interesting research topic that has not been covered in this thesis is the modelling of the performance of pitchers relative to their peers. In theory one could model the performance in terms of quartiles, similar to how height predictions are made, where besides or instead of the prediction for the ball speed in absolute terms, we predict whether the player will be in the top 25% pitchers at a later age. For this question the use of quartile regression could be appropriate.

The data set from project Fastball contained data on injuries sustained by the athletes, but unfortunately this was mostly missing and therefore very difficult to include in the analysis. Although this information was never included in the analysis and modelling of the data, the inclusion of injuries would possibly be beneficial for modelling. Not only does this give more information on the players, but we know that injuries have quite an impact on performance; including these predictors can help make sense of unusual data and could possibly help to predict performance after recovering from injury.

References

[1] Baker, G.A. (1954). Factor analysis of relative growth. *Growth, 18*(3), 137-143.

[2] Bates, D., Maechler, M., Bolker, B. & Walker, S. (2015). Fitting Linear Mixed-Effects Models Using lme4. *Journal of Statistical Software, 67*(1), 1-48. doi:10.18637/jss.v067.i01.

[3] Bollen, K.A. & Curran, P.J. (2006). *Latent Curve Models: A Structural Equation Perspective*. Hoboken, New Jersey: John Wiley & Sons

[4] Bonomi, A., Goris, A., Yin, B. & Westerterp, K. (2009). Detection of Type, Duration, and Intensity of Physical Activity Using an Accelerometer. *Medicine and science in sports and exercise, 41*, 1770-7. 10.1249/MSS.0b013e3181a24536.

[5] Boxhoorn, D. (2019, januari 3). Soms slaan we door naar de kant van de getalletjes [Sometimes we go too far to the side of the numbers]. *NRC*. Retrieved from https://www.nrc.nl/nieuws/2019/01/03/soms-slaan-we-door-naar-de-kant-van-de-getalletjes-a3127845

[6] Chambers, R., Gabbett, T. J., Cole, M. H., & Beard, A. (2015). The Use of Wearable Microsensors to Quantify Sport-Specific Movements.*Sports Medicine, 45*(7), 1065–1081. doi: 10.1007/s40279-015-0332-9

[7] Conover, W.J., Johnson, M.E. & Johnson, M.M. (1981). A Comparative Study of Tests for Homogeneity of Variances, with Applications to the Outer Continental Shelf Bidding Data. *Technometrics, 23*(4), 351-361, doi:10.1080/00401706.1981.10487680

[8] Conroy, D. E., & Coatsworth, J. D. (2004). The effects of coach training on fear of failure in youth swimmers: A latent growth curve analysis from a randomized, controlled trial. *Journal of Applied Developmental Psychology, 25*(2), 193–214. doi:10.1016/j.appdev.2004.02.007

[9] Cronbach, L.J. & Furby, L. (1970). How we should measure "change" - or should we? *Psychological Bulletin, 74*(1), 68-80. Retrieved from: https://www.gwern.net/docs/dnb/1970-cronbach.pdf

[10] Curran, P. J., Obeidat, K., & Losardo, D. (2010). Twelve Frequently Asked Questions About Growth Curve Modeling. *Journal of cognition and development: official journal of the Cognitive Development Society, 11*(2), 121–136. doi:10.1080/15248371003699969

[11] Delft University of Technology. (november 20, 2017). *Six million euros in research funding to develop technology that prevents sports injuries.* Retrieved from https://www.tudelft.nl/en/2017/tu-delft/six-million-euros-in-research-funding-to-develop-technology-that-prevents-sports-injuries/

[12] Delft University of Technology. (n.d.) *Citius Altius Sanius*. Retrieved on september 27, 2019, from https://www.tudelft.nl/io/onderzoek/research-labs/emerging-materials-lab/citius-altius-sanius/

[13] Ekstrand, J. (2013). Keeping your top players on the pitch: The key to football medicine at a professional level. *British Journal of Sports Medicine, 47*. 723-724. doi: 10.1136/bjsports-2013-092771.

[14] Ekstrand, J., Hägglund, M. & Waldén, M. (2011). Epidemiology of Muscle Injuries in Professional Football (Soccer). *The American Journal of Sports Medicine, 39*(6), 1226–1232. doi: 10.1177/0363546510395879

[15] Ekstrand, J., Healy, J.C., Waldén, M., Lee, J.C., English, B. & Hägglund, M. (2012). Hamstring muscle injuries in professional football: the correlation of MRI findings with return to play. *British Association of Sport and Excercise Medicine, 46*(2), 112-117. doi:10.1136/bjsports-2011-090155

[16] Everitt, B.S. (2005). Longitunal Data Analysis. In B.S. Everitt, & D.C. Howell (Eds.), *Encyclopedia of Statistics in Behavioral Science* (Vol. 2, pp. 1098-1101). Chichester, England: John Wiley & Sons, Ltd.

[17] Fisher, R.A. (1919). XV.—The Correlation between Relatives on the Supposition of Mendelian Inheritance. *Transactions of the Royal Society of Edinburgh, 52*(2), 399-433. doi:10.1017/s0080456800012163

[18] Fox, J. (2008). *Applied regression analysis and generalized linear models* (2nd edition). Los Angeles: SAGE.

[19] Frizzo-Barker, J., Chow-White, P. A., Mozafari, M., & Ha, D. (2016). An empirical study of the rise of big data in business scholarship. *International Journal of Information Management, 36*(3), 403–413. doi:10.1016/j.ijinfomgt.2016.01.006

[20] Goldstein, H. (1986). Multilevel Mixed Linear Model Analysis Using Iterative Generalized Least Squares. *Biometrika, 73*(1), 43-56. doi:10.2307/2336270

[21] Goldstein, H. (1991). Nonlinear Multilevel Models, with an Application to Discrete Response Data. *Biometrika, 78*(1), 45-51. doi:10.2307/2336894

[22] Gompertz, B. (1820). XVII. A sketch of an analysis and notation applicable to the estimation of the value of life contingencies. *Philosophical Transactions of the Royal Society, 110*, 214–294. doi:10.1098/rstl.1820.0018

[23] Harville, D. (1977). Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems. *Journal of the American Statistical Association, 72*(358), 320-338. doi:10.2307/2286796

[24] Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd edition). New York: Springer.

[25] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R* (6th edition). New York: Springer.

[26] Judge, J. & Baseball Prospectus Stats Team (2015, April 29). *Prospectus Feature: DRA: An In-Depth Discussion.* Retrieved from https://www.baseballprospectus.com/news/article/26196/prospectus-feature-dra-an-in-depth-discussion/

[27] Laird, N., & Ware, J. (1982). Random-Effects Models for Longitudinal Data. *Biometrics, 38*(4), 963-974. doi:10.2307/2529876

[28] Leeuw, J. de (2005). Linear Multilevel Models. In B.S. Everitt, & D.C. Howell (Eds.), *Encyclopedia of Statistics in Behavioral Science* (Vol. 2, pp. 772-779). Chichester, England: John Wiley & Sons, Ltd.

[29] Lindstrom, M.J., & Bates, D.M. (1990). Nonlinear Mixed Effects Models for Repeated Measures Data. *Biometrics, 46*(3), 673-687. doi:10.2307/2532087

[30] Lindstrom, M.J., & Bates, D.M. (1988). Newton-Raphson and EM Algorithms for Linear Mixed-Effects Models for Repeated-Measures Data. *Journal of the American Statistical Association, 83*(404), 1014-1022. doi:10.2307/2290128

[31] Mantyjarvi, J., Himberg, J., & Seppanen, T. (2001). Recognizing human motion with multiple acceleration sensors. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 747–752. doi:10.1109/ICSMC.2001.973004

[32] Marshall, S.W., Mueller, F.O., Kirby, D.P. & Yang, J. (2003) Evaluation of Safety Balls and Faceguards for Prevention of Injuries in Youth Baseball. *The Journal of the American Medical Association (JAMA), 289*(5), 568–574. doi:10.1001/jama.289.5.568

[33] NWO (2017). *Citius Altius Sanius: injury-free exercise for everyone. Fase 3: Programmavoorstel, P16-28*. Den Haag.

[34] Pinheiro, J.C., & Bates, D.M. (2000). *Mixed-Effects Models in S and S-PLUS*. New York: Springer.

[35] Pinheiro, J.C., Bates, D.M., DebRoy, S., Sarkar, D., R Core Team (2018). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-137, https://CRAN.R-project.org/package=nlme.

[36] Polglaze, T., Hogan, C., Dawson, B., Buttfield, A., Osgnach, C., Lester, L., & Peeling, P. (2018). Classification of Intensity in Team Sport Activity. *Medicine & Science in Sports & Exercise, 50*(7), 1487–1494. doi: 10.1249/MSS.0000000000001575

[37] Potthoff, R., & Roy, S. (1964). A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems. *Biometrika, 51*(3/4), 313-326. doi:10.2307/2334137

[38] Puerzer, R. J. (2002). From Scientific Baseball to Sabermetrics: Professional Baseball as a Reflection of Engineering and Management in Society. *NINE: A Journal of Baseball History and Culture, 11*(1), 34-48. University of Nebraska Press. Retrieved September 7, 2019, from Project MUSE database.

[39] Quetelet, L.A.J. (1835). Sur l'homme et le développement de ses facultés ou essai de physique sociale. In E. Boyd, B. S. Savara, & J. F. Schilke (Eds.) (1980), *Origins of the Study of Human Growth* (pp. 317–332). Portland, OR: University of Oregon Health Sciences Center Foundation.

[40] R Core Team (2018). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

[41] Rogers, M.J., Hrovat, K., McPherson, K., Moskowitz, M.E., & Reckart, T. (1997). *Accelerometer data analysis and presentation techniques.* Washington, D.C.: National Aeronautics and Space Administration. Retrieved January, 2019, from https://ntrs.nasa.gov/search.jsp?R=19970034695

[42] Roring, R. W., & Charness, N. (2007). A multilevel model analysis of expertise in chess across the life span. *Psychology and Aging, 22*(2), 291-299. doi:10.1037/0882-7974.22.2.291

[43] Singer, J.D. & Willett, J.B. (2005). Growth Curve Modeling. In B.S. Everitt, & D.C. Howell (Eds.), *Encyclopedia of Statistics in Behavioral Science* (Vol. 2, pp. 772-779). Chichester, England: John Wiley & Sons, Ltd.

[44] Schotel, R. (2019) *Monitoring local muscle load in football: Use leg acceleration, processed with a big data analysis approach, as an indication of the local muscle load to accurately represent the players' experienced load* (Master's thesis). Retrieved June, 2019, from TU Delft Repository. http://resolver.tudelft.nl/uuid:43b78565-ff82-42b9-944f-b5af5021b525

[45] Schuldhaus, D., Zwick, C., Koerger, H., Dorschky, E., Kirk, R. & Eskofier, B. (2015) Inertial Sensor-Based Approach for Shot/Pass Classification During a Soccer Match. *Proc. 21st ACM KDD Workshop on Large-Scale Sports Analytics* (pp. 1-4). Sydney, Australia.

[46] Schumacker, R. E., & Lomax, R. G. (2004). *A Beginner's Guide to Structural Equation Modeling* (2nd edition). Mahwah, New Jersey: Lawrence Erlbaum Associates.

[47] Sikka, R. S., Baer, M., Raja, A., Stuart, M., & Tompkins, M. (2019). Analytics in Sports Medicine. *The Journal of Bone and Joint Surgery, 101*(3), 276–283. doi:10.2106/jbjs.17.01601

[48] Sony Pictures Entertainment. (2011). *Moneyball.*

[49] Verhagen, L. (2019, januari 18). Geen sport ontkomt nog aan de datadrift [No sport can escape the data fever]. *de Volkskrant.* Retrieved from https://www.volkskrant.nl/wetenschap/geen-sport-ontkomt-nog-aan-datadrift˜be933517/

[50] Verhulst, P.F. (1845). Recherches mathématiques sur la loi d'accroissement de la population [Mathematical research on the law of population growth]. *Nouveaux Mémoires de l'Académie Royale des Sciences et Belles-Lettres de Bruxelles, 18*, 1–38. Retrieved from https://eudml.org/doc/182533

[51] Verhulst, P.F. (1847). Deuxième mémoire sur la loi d'accroissement de la population [Second memoir on the law of population growth]. *Mémoires de l'Académie Royale des Sciences, des Lettres et des Beaux-Arts de Belgique, 20*, 1–32. Retrieved from https://eudml.org/doc/178976

[52] Wei, T. & Simko, V. (2017). *R package "corrplot": Visualization of a Correlation Matrix* (Version 0.84). Available from https://github.com/taiyun/corrplot

[53] Wilkerson, G. B., Gupta, A., & Colston, M. A. (2018). Mitigating Sports Injury Risks Using Internet of Things and Analytics Approaches. *Risk Analysis, 38*(7), 1348–1360. doi: 10.1111/risa.12984

[54] Wishart, J. (1938). Growth-Rate Determinations in Nutrition Studies with the Bacon Pig, and Their Analysis. *Biometrika, 30*(1/2), 16-28. doi:10.2307/2332221

[55] Woods, C., Hawkins, R. D., Maltby, S., Hulse, M., Thomas, A. & Hodson, A. (2004). The Football Association Medical Research Programme: an audit of injuries in professional football—analysis of hamstring injuries. *British Journal of Sports Medicine, 38*(1), 36-41. doi:10.1136/bjsm.2002.002352

[56] Zeger, S.L. & Harlow, S.D. (1987). Mathematical models from laws of growth to tools for biologic analysis: fifty years of "Growth". *Growth, 51*(1), 1-21. PMID: 3305180

[57] Yang, A.Y., Jafari, R., Sastry, S.S. & Bajcsy, R. (2009). Distributed recognition of human actions using wearable motion sensor networks. *Journal of Ambient Intelligence and Smart Environments, 1*, 103-115. doi:10.3233/AIS-2009-0016

A Experiment protocol data collection project P6 [44]

Experiment protocol (1/2)

0 PRE-PREPARATION

A Experiment set-up and protocol

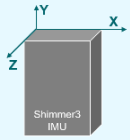
- > Get approval of ethics committee TU Delft – fill in ethics checklist
- > Make information letter participants
- > Make informed consent
- > Figure out LPM system
- > Figure out Shimmer3 IMUs and ConsensysBasic
- > Conduct Shimmer3 IMU and ConsensysBasic trial
- > Design drills based on hypotheses
- > Make protocol
- > Make information participant form and drill questionnaire
- > Make sensor legging size M
- > Conduct experiment trial and update experiment protocol with the results
- > Send information to participants

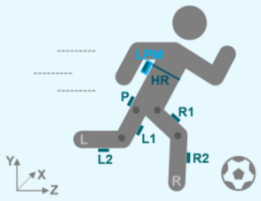
B Equipment

- > Invite 5 participants – with sport clothing
- > Reserve pitch
- > 1x LPM system and 1x shirt
- > 1x Heart rate monitor
- > 6x Shimmer3 IMUs and 1x legging
- > Cones and balls
- > Tape measure size S and L
- > Safety pins
- > Duct tape
- > Print all forms
- > Stopwatch
- > Laptop

C Shimmer orientation and location

- > Define origin of internal coordinate system Shimmers
- > Define orientation and location of Shimmers
  - E90F = P – middle of lower back
  - E90D = R1 – middle of right upper leg
  - E914 = R2 – middle of right lower leg
  - E8E2 = L1 – middle of left upper leg
  - E8D0 = L2 – middle of left lower leg
  - *96EF = X – extra Shimmer at LPM sensor*





LPM – global motion profile in 2D

Heart rate monitor – heartbeat

Shimmer3 IMUs – local linear acceleration and angular rate in 3D

1 PREPARATION

A Preparation before participant arrives

i Collect all equipment

- > Collect Shimmer3 IMUs – from TU Delft
- > Collect LPM sensor #054, heart rate belt #054, and shirt
- > Collect 11 cones and 10 balls

ii Prepare drills

- > See drills and dimensions in section *EXPERIMENT 2A/B*
- > Take cones, balls, and large tape measure to the pitch

iii Prepare Shimmer sensors for logging – repeat 6x

- > Connect Base to power socket and to laptop via USB
- > Start *ConsensysBASIC v1.5.0* and select *Manage Devices*
- > Switch on power of Shimmers and dock in Base
- > Click on *Reset Base* and click on *Reset Shimmers*
- > Select *1 Shimmer* at the time in graphic – repeat 6x
- > Check in device list for each Shimmer – repeat 6x:
  - Firmware version SDLog v0.19.0 – for logging data to SD card
  - SD Card memory empty – if not: click on *Clear SD*
  - Battery life near 100%
- > Click on *Configure* for each Shimmer – repeat 6x
  - Set a trial name: *ExpReal#* – use participant number
  - Select *undock/dock* as start/stop logging method
  - Choose *Shimmer Name* – use name of Shimmer location
  - Choose *Sampling Rate of 199.8 Hz*
  - Select *Low-Noise Accelerometer with ± 2g*
  - Select *Wide-Range Accelerometer with ±16g*
  - Select *Gyroscope with ±2000dps*
  - Select *Magnetometer with ±4.7Ga*
  - Set Shimmer to factory default calibration by clicking on *Reset*
  - Click on *Write Config* to write settings and save the configuration to the selected Shimmer
  - Click on *Done* when configuration is completed
- > Check configuration for each Shimmer – repeat 6x
  - Click on *Configure*
  - Check configuration
  - Click on *Back* and *OK* if the configuration is correct

B Preparation if participant has arrived

i Conduct participant formalities

- > Explain the experiment and goal to the participant, and answer any questions the participant may have
- > Participant and researcher sign *informed consent*
- > Participant puts on legging, heart rate belt, and shirt

ii Start Shimmer sensors to capture data

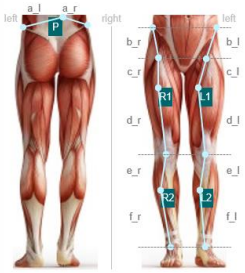
- > Undock Shimmers from Base to start data capturing – data logging will start almost immediately, you must log data for at least one minute to ensure a data file is created
- > Do not power off
- > Watch led behaviour before proceeding – the green LED will turn on and off at one second intervals when capturing data
- > Create a mark in Shimmer data by turning 15x around z-axis

iii Attach all sensors to clothing

- > Tape Shimmer X to LPM sensor and attach LPM sensor to shirt
- > Place Shimmer P, R1, R2, L1, and L2 in legging, use safety pins to keep the sensors in place – pay attention to location and orientation

iv Measure sensor distance

- > Fill in *information participant*
- > Measure length between top right corner of Shimmer X and P with small tape measure
- > Measure length between middle of bending line and top right corner of Shimmer with small tape measure – see figure



v Start LPM system to capture data

- > Click on *imoServer* at desktop
- > Select *Revalidatie Campus* at Prepare in Measurement Selection field and click on *Activate*
- > Check box of *Cam 1* and *Cam 2* of number 054
- > Open *Inmotio Client* at desktop
- > Click on *Live/Record Start* and click on *OK* to start

Sources:
http://www.shimmersensing.com/images/uploads/docs/Consensys_User_Guide_rev1.4a.pdf
https://www.pinterest.co.uk/pin/431361089021515 20/ (seen on 14.03.2018)

Experiment protocol (2/2)

2 EXPERIMENT

A Pre-experiment

i Instructions on how to perform the experiment

> General:
- The location of the drills on the pitch are fixed for all experiments
- The last cone is placed at the penalty spot – start drill from here
- After each situation the experienced load will be asked and noted
- The duration per situation will be timed and noted – use timer
- 3 min between situations and 5 min between drills – use timer
- Perform the drills in a normal and clear manner
- Don't play with the balls or make lots of movements in between the situations, just try to stand relaxed

> Jog/sprint
- Jog: around 60% of your max speed
- Sprint: 100% of your max speed

> Turn
- Turn with one leg at the location of the cone – it doesn't matter which leg and you don't have to go around the cone

> Pass/shoot
- The target is placed at the goal line – more towards the left if you are right-footed and visa versa
- The ball will replace the last cone and a new ball will be put down by the researcher during the situations involving a ball
- The researcher will collect the balls in between the situations
- Pass or shoot the ball directly with the inside of your foot – without any small touch before passing or shooting
- Aim at the target – it doesn't matter if you miss the target

ii Go to pitch to perform experiment

> Create a mark in data by (i) walk out of the medical centre to the right corner flag and jog to the right side of the goal, and (ii) stand still for 30s and (iii) jumping 15x on the penalty spot facing the goal

> Perform experiment

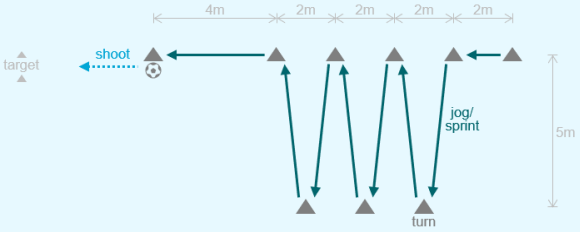
B Perform experiment

i Perform Drill A – 10x back and forward per situation; 3 min between each situation

> Situation 1a: jog + turn + no ball
> Situation 1b: jog + turn + pass at one side
> Situation 1c: jog + turn + shoot at one side
> Situation 2a: sprint + turn + no ball
> Situation 2b: sprint + turn + pass at one side
> Situation 2c: sprint + turn + shoot at one side

ii Perform Drill B – 5x zigzag per situation and walk back; 3 min between each situation

> Situation 3a: jog + turn + no ball
> Situation 3b: jog + turn + shoot at the end
> Situation 4a: sprint + turn + no ball
> Situation 4b: sprint + turn + shoot at the end



> Go back to lab to complete

3 COMPLETION

A Completion before participant leaves

i Stop LPM system with capturing data

> Click in Inmotio Client on *Live/Record Stop* to stop
> Click on *Yes* to save
> Save as *ExpReal#_date* in *ExperimentsRozemarijn* folder at desktop

ii Detach all sensors from clothing

> Detach LPM sensor off shirt and Shimmer X off LPM sensor
> Take Shimmer P, R1, R2, L1, and L2 out legging
> Participant takes off legging, heart rate belt, and shirt

iii Stop Shimmer sensors to capture data

> Create a mark in Shimmer data by turning 15x around z-axis
> Dock Shimmers into Base to stop data capturing

B Completion after participant leaves

i Import data from each Shimmer – repeat 6x

> Scanning SD Cards – one chance:
- Select *1 Shimmer* in graphic
- Click on *Import* and click on *Next* when scanning is completed

> Configuring import sessions:
- Select *ExpReal#*
- Click on >> to add data as new session to the list
- Click on *Next* to continue to the next stage and click on *Yes* to proceed

> Importing session
- Data selected for import is now being imported into the Consensys database
- Click on *Done* when import is completed
- Go to *Manage Data* and check if import is successful – check configuration and time

ii Put all the equipment away – if done

> Undock Shimmers out Base and switch off power
> Bring LPM sensor and heart rate belt back, place LPM sensor in charger, and close all programs at computer
> Take off the cones and balls from the pitch and put away

C Manage and export data

This stage can be done at another/later moment

i Export data LPM – .csv file

> Export and safe LPM data at a sampling rate of 200 Hz and safe video recordings – ask Rosanne

ii Export data per Shimmer – .mat file – repeat 6x

> Click on *Manage Data*
> Select data: *ExpReal#* – repeat 6x
> Select format:
- File Format: *.mat*
- Timestamp Format: *unix*
- Data Format: *calibrated*

> Click on *Export* to export the selected data to a file in the requested format
> Select *ExpReal#_date* folder in *Data Processing and Analysing* folder, and click on *Save*
> Click on *Open Path* when export is completed to navigate to the file(s)
> Click on *Done* in Consensys

iii Process and analyse data in MATLAB

Sources:
http://www.shimmersensing.com/images/uploads/docs/Consensys_User_Guide_rev1.4a.pdf
https://www.pinterest.co.uk/pin/431361089021515 20/ (seen on 14.03.2018)

B Summary outputs for models in R

Outputs for the `summary()` command in R for the models described in Section 8.

B.1 Models from Section 8.1

```
Summary output full linear model

Call:
lm(formula = ABSSq ~ age12 + Height + weight + ROM_IRsh_DA +
    ROM_ERsh_DA + F_IR_DA + F_ER_DA, data = mdatfull)

Residuals:
     Min       1Q   Median       3Q      Max
-1475.69  -320.07   -22.19   373.75  1566.09

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -605.878    729.930  -0.830 0.407047
age12        213.945     28.560   7.491 5.10e-13 ***
Height        19.328      4.497   4.298 2.21e-05 ***
weight         9.407      3.442   2.733 0.006577 **
ROM_IRsh_DA  -10.863      2.428  -4.474 1.02e-05 ***
ROM_ERsh_DA    1.737      1.564   1.111 0.267502
F_IR_DA        2.272      1.070   2.123 0.034386 *
F_ER_DA        4.030      1.162   3.469 0.000585 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 529.5 on 369 degrees of freedom
Multiple R-squared:  0.6754,   Adjusted R-squared:  0.6693
F-statistic: 109.7 on 7 and 369 DF,  p-value: < 2.2e-16

-----------
> AIC(linfitfull)
[1] 5808.828
> BIC(linfitfull)
[1] 5844.218
> logLik(linfitfull)
'log Lik.' -2895.414 (df=9)
```

Listing 1: R output for full linear regression model 8.1

```
Summary output full linear model with added quadratic term

Call:
lm(formula = ABSSq ~ age12 + I(age12^2) + Height + weight + ROM_IRsh_DA +
    ROM_ERsh_DA + F_IR_DA + F_ER_DA, data = mdatfull)

Residuals:
     Min       1Q   Median       3Q      Max
-1539.23  -304.30   -14.66   348.01  1455.12

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -408.954    719.936  -0.568 0.570353
age12        589.903    106.332   5.548 5.54e-08 ***
I(age12^2)   -52.822     14.409  -3.666 0.000283 ***
Height        14.803      4.592   3.223 0.001380 **
weight         9.069      3.387   2.678 0.007741 **
ROM_IRsh_DA   -9.764      2.407  -4.057 6.07e-05 ***
ROM_ERsh_DA    2.096      1.541   1.360 0.174639
F_IR_DA        2.352      1.053   2.234 0.026082 *
F_ER_DA        3.888      1.143   3.400 0.000747 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 520.8 on 368 degrees of freedom
Multiple R-squared:  0.6868,  Adjusted R-squared:  0.68
F-statistic: 100.9 on 8 and 368 DF,  p-value: < 2.2e-16

-----------
> AIC(linfit2)
[1] 5797.306
> BIC(linfit2)
[1] 5836.628
> logLik(linfit2)
'log Lik.' -2888.653 (df=10)
```

Listing 2: R output for full linear regression model with quadratic term 8.2

```
Summary output full linear model with robust regression

Call: rlm(formula = ABSSq ~ age12 + I(age12^2) + Height + weight +
    ROM_IRsh_DA + ROM_ERsh_DA + F_IR_DA + F_ER_DA, data = mdatfull)
Residuals:
     Min       1Q   Median       3Q      Max
-1558.87  -311.04   -19.45   342.13  1477.25

Coefficients:
            Value     Std. Error t value
(Intercept) -391.1858  744.8241    -0.5252
age12        571.0948  110.0083     5.1914
I(age12^2)   -48.9223   14.9071    -3.2818
Height        15.2866    4.7513     3.2174
weight         8.9144    3.5038     2.5442
ROM_IRsh_DA  -10.6996    2.4899    -4.2973
ROM_ERsh_DA    1.8414    1.5944     1.1549
F_IR_DA        2.4486    1.0890     2.2485
F_ER_DA        3.7131    1.1829     3.1390

Residual standard error: 493.8 on 368 degrees of freedom

------
> AIC(linfitrobust)
[1] 5797.78
> BIC(linfitrobust)
[1] 5837.103
> logLik(linfitrobust)
'log Lik.' -2888.89 (df=10)
```

Listing 3: R output for full linear model with robust regression

B.2 Models from Section 8.2.1

```
Summary output random intercept model
Linear mixed-effects model fit by maximum likelihood
 Data: mdatfull
       AIC      BIC    logLik
  5732.322 5748.051 -2862.161

Random effects:
 Formula: ~1 | PPnumber
        (Intercept) Residual
StdDev:    565.7517 344.2977

Fixed effects: ABSSq ~ age12
                Value Std.Error  DF  t-value p-value
(Intercept) 2815.6098  95.72632 262 29.41312       0
age12        625.0249  23.14173 262 27.00856       0
 Correlation:
      (Intr)
age12 -0.805
```

```
Standardized Within-Group Residuals:
          Min            Q1           Med            Q3           Max
-3.0288733256 -0.4749101075  0.0003281178  0.5800414826  2.5212278890

Number of Observations: 377
Number of Groups: 114
```

Listing 4: R output lme for random intercepts Model 8.3

```
Summary output random slope + intercept model
Linear mixed-effects model fit by maximum likelihood
 Data: mdatfull
       AIC      BIC    logLik
  5716.741 5740.334 -2852.37

Random effects:
 Formula: ~age12 | PPnumber
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 418.0214 (Intr)
age12       136.9277 -0.131
Residual    320.1146

Fixed effects: ABSSq ~ age12
               Value Std.Error  DF  t-value p-value
(Intercept) 2876.568  83.63092 262 34.39599       0
age12        623.716  26.14496 262 23.85607       0
 Correlation:
      (Intr)
age12 -0.764

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-3.11941883 -0.49360438 -0.02935689  0.58261613  2.75242381

Number of Observations: 377
Number of Groups: 114
```

Listing 5: R output lme for random slope + intercepts Model 8.4

```
Summary output random intercept model with quadratic term
Linear mixed-effects model fit by maximum likelihood
 Data: mdatfull
       AIC      BIC    logLik
  5684.056 5703.717 -2837.028

Random effects:
 Formula: ~1 | PPnumber
        (Intercept) Residual
StdDev:    542.8396 319.1574

Fixed effects: ABSSq ~ age12 + I(age12^2)
                Value Std.Error  DF  t-value p-value
(Intercept) 2128.6245 129.22617 261 16.47208       0
age12       1138.9636  73.11967 261 15.57671       0
I(age12^2)   -78.2817  10.70337 261 -7.31374       0
 Correlation:
           (Intr) age12
age12      -0.848
I(age12^2)  0.715 -0.955

Standardized Within-Group Residuals:
         Min           Q1          Med           Q3          Max
-3.096436806 -0.472437780  0.003862812  0.554295518  2.424305213

Number of Observations: 377
Number of Groups: 114
```

Listing 6: R output lme for random intercepts Model 8.5

```
Summary output random slope + intercept model with quadratic term
Linear mixed-effects model fit by maximum likelihood
 Data: mdatfull
       AIC      BIC    logLik
  5686.106 5713.631 -2836.053

Random effects:
 Formula: ~age12 | PPnumber
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 524.19015 (Intr)
age12        79.65155 -0.182
Residual    311.41585

Fixed effects: ABSSq ~ age12 + I(age12^2)
               Value Std.Error  DF   t-value p-value
(Intercept) 2198.804 131.67381 261 16.698870       0
age12       1101.564  75.97826 261 14.498416       0
I(age12^2)   -73.451  11.20124 261 -6.557401       0
 Correlation:
           (Intr) age12
age12      -0.864
I(age12^2)  0.736 -0.953

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-3.07552562 -0.47787289 -0.01282994  0.56207161  2.46141379

Number of Observations: 377
Number of Groups: 114
```

Listing 7: R output lme for random slope + intercepts Model 8.6

B.3 Models from Section 8.2.2

```
Summary output random slope + intercept model with all covariates
Linear mixed-effects model fit by maximum likelihood
 Data: mdatfull
       AIC      BIC    logLik
  5646.827 5697.946 -2810.413

Random effects:
 Formula: ~1 + age12 | PPnumber
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 378.83634 (Intr)
age12        77.98112 -0.052
Residual    305.02800

Fixed effects: ABSSq ~ age12 + I(age12^2) + Height + weight + F_ER_DA + F_IR_DA +
    ROM_ERsh_DA + ROM_IRsh_DA
                 Value Std.Error  DF   t-value p-value
(Intercept) -1076.7515  960.8698 255 -1.120601  0.2635
age12         694.6591   97.5934 255  7.117893  0.0000
I(age12^2)    -46.6887   12.3143 255 -3.791428  0.0002
Height         14.0441    6.5380 255  2.148079  0.0326
weight         14.2990    4.5580 255  3.137107  0.0019
F_ER_DA         2.0854    0.9681 255  2.154147  0.0322
F_IR_DA         0.5830    0.7943 255  0.734035  0.4636
ROM_ERsh_DA     4.1448    1.2151 255  3.411038  0.0008
ROM_IRsh_DA    -2.9996    1.7231 255 -1.740766  0.0829
 Correlation:
            (Intr) age12  I(12^2 Height weight F_ER_D F_IR_D ROM_ER
age12        0.440
I(age12^2)  -0.307 -0.925
Height      -0.954 -0.491  0.390
weight       0.357 -0.073 -0.004 -0.524
```

```
F_ER_DA      -0.125 -0.077 -0.062  0.118 -0.223
F_IR_DA       0.008 -0.052  0.065 -0.036 -0.037 -0.467
ROM_ERsh_DA -0.203  0.033 -0.069  0.004  0.079  0.007  0.035
ROM_IRsh_DA -0.117  0.170 -0.099 -0.002 -0.054 -0.017  0.014  0.222

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.67165031 -0.49533417  0.05103827  0.55287874  2.63864604

Number of Observations: 377
Number of Groups: 114
```

Listing 8: R output lme for full linear multilevel model with random slope + intercept 8.7

```
Summary output random intercept model with all covariates
Linear mixed-effects model fit by maximum likelihood
 Data: mdatfull
      AIC      BIC   logLik
  5646.46 5689.715 -2812.23

Random effects:
 Formula: ~1 | PPnumber
        (Intercept) Residual
StdDev:    454.6857 310.8786

Fixed effects: ABSSq ~ age12 + I(age12^2) + Height + weight + F_ER_DA + F_IR_DA +
    ROM_ERsh_DA + ROM_IRsh_DA
                 Value Std.Error  DF   t-value p-value
(Intercept) -1089.5219  989.5151 255 -1.101066  0.2719
age12         767.4708   98.3744 255  7.801531  0.0000
I(age12^2)    -55.5793   12.0038 255 -4.630134  0.0000
Height         13.4290    6.6859 255  2.008548  0.0456
weight         14.7417    4.4958 255  3.279016  0.0012
F_ER_DA         1.9303    0.9599 255  2.010972  0.0454
F_IR_DA         0.4640    0.7960 255  0.583003  0.5604
ROM_ERsh_DA     4.1549    1.2312 255  3.374682  0.0009
ROM_IRsh_DA    -3.3446    1.7366 255 -1.925977  0.0552
 Correlation:
            (Intr) age12  I(12^2 Height weight F_ER_D F_IR_D ROM_ER
age12        0.474
I(age12^2)  -0.340 -0.930
Height      -0.954 -0.526  0.422
weight       0.320 -0.059 -0.018 -0.493
F_ER_DA     -0.127 -0.105 -0.028  0.125 -0.229
F_IR_DA      0.015 -0.055  0.065 -0.046 -0.021 -0.459
ROM_ERsh_DA -0.194  0.029 -0.070 -0.004  0.097  0.007  0.032
ROM_IRsh_DA -0.111  0.193 -0.132 -0.015 -0.034 -0.016  0.015  0.229

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.71847862 -0.50110586  0.05088229  0.55129946  2.58888907

Number of Observations: 377
Number of Groups: 114
```

Listing 9: R output lme for full linear multilevel model with random intercepts 8.8

B.4 Final model in Section 9.4

```
Linear mixed-effects model fit by maximum likelihood
 Data: mdatfull
       AIC      BIC    logLik
  5632.774 5668.14 -2807.387

Random effects:
 Formula: ~1 | PPnumber
        (Intercept) Residual
```

```
StdDev:    471.4562 309.7278

Fixed effects: ABSSq ~ age12 + I(age12^2) + Height + weight + F_ER_DA + ROM_ERsh_DA
                 Value Std.Error  DF   t-value p-value
(Intercept) -1340.4690 1000.7887 256 -1.339413  0.1816
age12         808.8602   96.9645 256  8.341818  0.0000
I(age12^2)    -58.7318   11.8981 256 -4.936223  0.0000
Height         13.5507    6.7961 256  1.993881  0.0472
weight         14.6869    4.5727 256  3.211900  0.0015
F_ER_DA         2.0834    0.8556 256  2.434951  0.0156
ROM_ERsh_DA     4.6447    1.1961 256  3.883322  0.0001
 Correlation:
            (Intr) age12  I(12^2 Height weight F_ER_D
age12        0.520
I(age12^2)  -0.371 -0.929
Height      -0.962 -0.544  0.436
weight       0.314 -0.058 -0.021 -0.492
F_ER_DA     -0.137 -0.147  0.000  0.117 -0.266
ROM_ERsh_DA -0.171 -0.015 -0.043  0.000  0.110  0.026

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-2.6810722 -0.5283475  0.0248107  0.5954327  2.4189089

Number of Observations: 376
Number of Groups: 114
```

Listing 10: R output lme for final prediction model 9.7

C Codes in R

C.1 Code for labelling and classifying acceleration data

```
#########################
# set up data analysis #
#########################

# set working directory containing data sets
setwd("~/Applied Mathematics MSc/Master Thesis/Thesis Sport/Data/Data CSV")

# calculate intensity vector, square root of squared acceleration in X, Y, Z direction
measurefunct<-function(sensor,ex){
  docname<-paste("Ex",ex,"_",sensor,".csv",sep="")
  dat<-read.csv(docname)
  colnames(dat)<-c("Accel_LN_X","Accel_LN_Y","Accel_LN_Z","Accel_WR_X","Accel_WR_Y",
                   "Accel_WR_Z","Gyro_X","Gyro_Y","Gyro_Z","Mag_X","Mag_Y","Mag_Z","
                       Timestamp")
  return(sqrt(dat$Accel_WR_X^2+dat$Accel_WR_Y^2+dat$Accel_WR_Z^2))
}

###############################
# choose levels for variables #
###############################

#choose sensor and number of experiment
sens<-"R1"
exp<-2

# [3] is this the first time you are running the data labelling?
#T if there is no labelled data yet, F otherwise
first<-F

#find acceleration vector and set number of seconds used for classification
accel<-measurefunct(sens,exp)

secs = 1 #set number of seconds for classification
setframe<-seq(from=1,to=length(accel),by=200*secs) #set number of intervals of size secs

# [1] if too few activities are detected, lower the percentage lowlevel; increase it if
    too much is detected
lowlevel<-0.06

# [2] assign intensity levels per detected period of activity
#first label the activities per detected period manually, according to the performed
    exercises
highact<-c(8:10,15:24) #this vector contains the periods labelled high intensity,
    counting from "left to right" (chronological)
medact<-c(1:7,11:14) #this vector contains the periods labelled high intensity, counting
    from "left to right" (chronological)

#set the number of folds for the k-fold crossvalidation
k<-5

####################
# set up libraries #
####################

{
  library(ggplot2)
  library(pracma)
  library(rpart)
  library(caret)
  library(dplyr)
  library(class)
  library(party)
  library(e1071)
  library(gmodels)
}
```

```
####################
# set up functions #
####################

#Functions needed for analysis and classification
{
#calculate cross-correlation according
#to (Bonomi, Goris, Yin & Westerterp, 2009)
rab<-function(i,N,a,b){
  if(i>=0){
    sumterm<-1
    for(j in 1:(N-i)){
      sumterm<-sumterm+a[i+j]*b[j]
    }
    return(sumterm-1)
  }
  if(i<0){
    k=-i
    sumterm<-1
    for(j in 1:(N-k)){
      sumterm<-sumterm+a[k+j]*b[j]
    }
    return(sumterm-1)
  }
}

#find maximum value for cross-correlation
racc<-function(acc,N,frame){
  setframe<-seq(from=1,to=length(acc),by=200*secs)
  rval<-c()
  d<-acc[setframe[frame]:(setframe[frame]+200*secs)]
  S<-length(d[!is.na(d)])
  for(i in 0:(S-1)){
    rval<-c(rval,rab(i,S,acc[setframe[frame]:(setframe[frame]+200*secs)],acc[(setframe[
        frame]+i):(setframe[frame]+200*secs+i)]))
  }
  return(max(rval))
}

#compute method 15 according to thesis by Schotel, 2019
method15<-function(acc,weights){
  Zone<-c(10,40,70,100)
  intdf<-data.frame(zone1=c(),zone2=c(),zone3=c())
  pks<-findpeaks(acc)[,1]
  ZP<-c()
  for(j in 2:4){
    numpeak<-pks[Zone[j-1]<pks & pks <=Zone[j]]
    ZP<-c(ZP,length(numpeak))
  }
  return(weights[1]*ZP[1]+weights[2]*ZP[2]+weights[3]*ZP[3])
}

#compute method 12 according to thesis by Schotel, 2019
method12<-function(acc,weights){
  Zone<-c(10,40,70,100)
  ZP<-c()
  for(j in 2:4){
    numpeak<-acc[Zone[j-1]<acc & acc <=Zone[j]]
    ZP<-c(ZP,length(numpeak))
  }
  return(weights[1]*ZP[1]+weights[2]*ZP[2]+weights[3]*ZP[3])
}

#compute standard deviation, average, peak to peak distance,
#cross-correlation, method 12, 15.
ClassMeasures<-function(acc,secs){
  setframe<-seq(from=1,to=length(acc),by=200*secs)
  #speed<-c() #compute 'speed'
```

```
  #for(i in 1:length(setframe)){
  #  d <- acc[setframe[i]:(setframe[i]+200*secs)]
  #  d <- d[!is.na(d)]
  #  speed<-c(speed,trapz(abs(d-9.81)))
  #}
  sd<-c() # compute standard deviation of acceleration
  for(i in 1:length(setframe)){
    sd<-c(sd,sd(na.omit(acc[setframe[i]:(setframe[i]+200*secs)])))
  }
  avg<-c() # compute average of acceleration
  for(i in 1:length(setframe)){
    avg<-c(avg,mean(na.omit(acc[setframe[i]:(setframe[i]+200*secs)])))
  }

  app<-c() # compute peak-to-peak distance
  np<-c() # compute number of peaks
  for(i in 1:length(setframe)){
    w<-acc[setframe[i]:(setframe[i]+200*secs)]
    peaks<-findpeaks(w)[,2]
    peakh<-findpeaks(w)[,1]
    peakdis<-c()
    if(length(peaks)>=2){
      for(j in 2:length(peaks)){
        peakdis<-c(peakdis,peaks[j]-peaks[j-1])
      }
    }
    else{
      peakdis<-4
    }
    app<-c(app,mean(peakdis))
    np<-c(np,length(subset(peakh,peakh>=200)))
  }

  Racc<-c()
  for(i in 1:length(setframe)){
    Racc<-c(Racc,racc(acc,200*secs,i))
  }

  accnorm<-acc/max(acc)*100
  m12<-c()
  m15<-c()
  for(i in 1:length(setframe)){
    m12<-c(m12,method12(na.omit(accnorm[setframe[i]:(setframe[i]+200*secs)]),c(1,4,7)))
    m15<-c(m15,method15(accnorm[setframe[i]:(setframe[i]+200*secs)],c(1,4,7)))
  }
  return(data.frame(sd,avg,ptp=app,R=Racc,Shots=np,m12,m15))
}

#find frames for which there is significant activity
#percent = 0.1 for secs=1, 0.15 for secs is 5 usually works
act<-function(df,percent){
  #classdf<-ClassMeasures(acc,secs)
  maxR<-max(df$R)
  return(which(df$R>=percent*maxR)*(200*secs))
}

#compute the start and end of moment of activity, given list of moments of activity
frame<-function(list){
  retlist<-c()
  for(i in 1:length(list)){
    #make new list for every item in list (num), containing all items within certain
        distance of num
    num<-list[i]
    newlist<-list-num
    framedis<-1000*secs

    #distance depends on the number of seconds used for classification, per second the
        reach is increased by 1000 frames (roughly 5 seconds)
    surlist<-newlist[newlist %in% c(-framedis:-1,1:framedis)]
```

```
    poslist<-surlist[surlist %in% 1:framedis]
    neglist<-surlist[surlist %in% -framedis:-1]

    #check if there is any activity within the framedistance
    if(length(surlist)>0){
      if(isempty(neglist)){
        retlist<-c(retlist,num-200*secs)
        #if there are no items in list within frame distance before num, include num as
            begin time and add margin
      }
      if(isempty(poslist)){
        retlist<-c(retlist,num+200*secs)
        #if there are no items in list within frame distance after num, include num as
            end time and add margin
      }
    }
  }
  return(retlist) #returns list of begin and end times for activity
}

framewide<-function(list){
  retlist<-c()
  for(i in 1:length(list)){
    #make new list for every item in list (num), containing all items within certain
        distance of num
    num<-list[i]
    newlist<-list-num
    framedis<-1000*secs

    #distance depends on the number of seconds used for classification, per second the
        reach is increased by 1000 frames (roughly 5 seconds)
    surlist<-newlist[newlist %in% c(-framedis:-200,200:framedis)]
    poslist<-surlist[surlist %in% 200:framedis]
    neglist<-surlist[surlist %in% -framedis:-200]

    #check if there is any activity within the framedistance
    if(length(surlist)>0){
      if(isempty(neglist)){
        retlist<-c(retlist,num-200*secs)
        #if there are no items in list within frame distance before num, include num as
            begin time and add margin
      }
      if(isempty(poslist)){
        retlist<-c(retlist,num+200*secs)
        #if there are no items in list within frame distance after num, include num as
            end time and add margin
      }
    }
  }
  return(retlist) #returns list of begin and end times for activity
}

#classify intensity of acceleration by comparing begin/end times of activity with time
    frame
timetype<-function(frames,time,sprintdet,jogdet){
  type<-c()
  sprint<-c()
  jog<-c()
  for(j in 1:(length(sprintdet)+length(jogdet))){
    if(j %in% jogdet){
      jog<-c(jog,seq(time[2*j-1],time[2*j]))
    }
    if(j %in% sprintdet){
      sprint<-c(sprint,seq(time[2*j-1],time[2*j]))
    }
  }
  for(i in 1:length(frames)){
    if(frames[i] %in% sprint){
      type<-c(type,"High")
```

```
    }
    else if(frames[i] %in% jog){
      type<-c(type,"Medium")
    }
    else{
      type<-c(type,"Low")
    }
  }
  return(type)
}

#convert classification by levels to numerical values
numconv<-function(char){
  if(char %in% c("Low")){
    return(1)
  }
  if(char %in% c("Medium")){
    return(2)
  }
  if(char %in% c("High")){
    return(3)
  }
}

#normalise vector
normalise <- function(x) {
  if(max(x)-min(x)>0){
    return ((x - min(x)) / (max(x) - min(x)))
  }
  else{
    return(x)
  }}

#Count function that counts the number of non-empty entries in a vector
counta<-function(x){
  count<-0
  for(i in 1:length(x)){
    if(x[i]>0){
      count<-count+1
    }
  }
  return(count)
}

#compute all relevant accuracies
#works only for this dataset
classaccuracy<-function(pred,labels){
  return(c(overall_acc=(table(pred,labels)[1,1]+table(pred,labels)[2,2]+table(pred,labels
      )[3,3])/sum(table(pred,labels)[,]),
           int_acc=(table(pred,labels)[1,1]+table(pred,labels)[3,3])/(sum(table(pred,
               labels)[1,])+sum(table(pred,labels)[3,])),
           high_acc=table(pred,labels)[1,1]/sum(table(pred,labels)[1,]),med_acc=table(
               pred,labels)[3,3]/sum(table(pred,labels)[3,])))
}


#compare classification methods. Crosspercent is the percentage of data used for
    crossvalidation
#Takes an even percentage of data from all three intensity classes for training. Options
    for classification method
#are "decision tree", "decision tree pruned" and "naive Bayes"
classcomp<-function(k,dataset,datasetn,classmet){
  list<-split(dataset, sample(1:k, nrow(dataset), replace=T))
  listn<-split(datasetn, sample(1:k, nrow(datasetn), replace=T))
  if(classmet %in% c("decision tree")){
    acc<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
        rep(NA,k),stringsAsFactors=F)
    accn<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
        rep(NA,k),stringsAsFactors=F)
```

```
  for(i in 1:k){
    train<-seq(1:k)[seq(1:k)!=i]
    traindf<-list[[train[1]]]
    traindf_n<-listn[[train[1]]]
    train<-train[-1]
    for(j in train){
      traindf<-rbind(traindf,list[[j]])
      traindf_n<-rbind(traindf_n,listn[[j]])
    }
    testdf<-list[[i]]
    testdf_n<-listn[[i]]

    fit<-rpart(type ~ sd + avg + ptp + R + m15 + m12, method="class", data=traindf)
    pred<-predict(fit,testdf,type="class")
    acc[i,]<-classaccuracy(testdf$type,pred)

    fitn<-rpart(type ~ sd + avg + ptp + R + m15 + m12, method="class", data=traindf_n)
    predn<-predict(fitn,testdf_n,type="class")
    accn[i,]<-classaccuracy(testdf_n$type,predn)
  }
  predacc<-colMeans(acc)
  predaccn<-colMeans(accn)
  classmatrix<-rbind(predacc,predaccn)
  rownames(classmatrix)<-c("DT","DT norm")
  return(round(classmatrix,4))
}
else if(classmet %in% c("decision tree pruned")){
  acc<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
      rep(NA,k),stringsAsFactors=F)
  accn<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
      rep(NA,k),stringsAsFactors=F)
  for(i in 1:k){
    train<-seq(1:k)[seq(1:k)!=i]
    traindf<-list[[train[1]]]
    traindf_n<-listn[[train[1]]]
    train<-train[-1]
    for(j in train){
      traindf<-rbind(traindf,list[[j]])
      traindf_n<-rbind(traindf_n,listn[[j]])
    }
    testdf<-list[[i]]
    testdf_n<-listn[[i]]

    fit<-rpart(type ~ sd + avg + ptp + R + m15 + m12, method="class", data=traindf)
    pfit<- prune(fit, cp= fit$cptable[which.min(fit$cptable[,"xerror"]),"CP"])
    pred<-predict(pfit,testdf,type="class")
    acc[i,]<-classaccuracy(testdf$type,pred)

    fitn<-rpart(type ~ sd + avg + ptp + R + m15 + m12, method="class", data=traindf_n)
    pfitn<- prune(fitn, cp= fit$cptable[which.min(fit$cptable[,"xerror"]),"CP"])
    predn<-predict(pfitn,testdf_n,type="class")
    accn[i,]<-classaccuracy(testdf_n$type,predn)
  }
  predacc<-colMeans(acc)
  predaccn<-colMeans(accn)
  classmatrix<-rbind(predacc,predaccn)
  rownames(classmatrix)<-c("Pruned DT","Pruned DT, norm")
  return(round(classmatrix,4))
}
else if(classmet %in% c("naive Bayes")){
  acc<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
      rep(NA,k),stringsAsFactors=F)
  accn<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
      rep(NA,k),stringsAsFactors=F)
  for(i in 1:k){
    train<-seq(1:k)[seq(1:k)!=i]
    traindf<-list[[train[1]]]
    traindf_n<-listn[[train[1]]]
    train<-train[-1]
```

```
      for(j in train){
        traindf<-rbind(traindf,list[[j]])
        traindf_n<-rbind(traindf_n,listn[[j]])
      }
      testdf<-list[[i]]
      testdf_n<-listn[[i]]

      fit<-naiveBayes(type ~ sd + avg + ptp + R + m15 + m12, method="class",data=traindf)
      pred<-predict(fit,testdf,type="class")
      acc[i,]<-classaccuracy(testdf$type,pred)

      fitn<-naiveBayes(type ~ sd + avg + ptp + R + m15 + m12, method="class", data=
          traindf_n)
      predn<-predict(fitn,testdf_n,type="class")
      accn[i,]<-classaccuracy(testdf_n$type,predn)
    }
    predacc<-colMeans(acc)
    predaccn<-colMeans(accn)
    classmatrix<-rbind(predacc,predaccn)
    rownames(classmatrix)<-c("NB","NB norm")
    return(round(classmatrix,4))
  }
  else if(classmet %in% c("knn")){
    acc<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
        rep(NA,k),stringsAsFactors=F)
    accn<-data.frame(overall_acc=rep(NA,k),int_acc=rep(NA,k),high_acc=rep(NA,k),med_acc=
        rep(NA,k),stringsAsFactors=F)
    for(i in 1:k){
      train<-seq(1:k)[seq(1:k)!=i]
      traindf<-list[[train[1]]]
      traindf_n<-listn[[train[1]]]
      train<-train[-1]
      for(j in train){
        traindf<-rbind(traindf,list[[j]])
        traindf_n<-rbind(traindf_n,listn[[j]])
      }
      testdf<-list[[i]]
      testdf_n<-listn[[i]]

      pred<-knn(train=traindf[,1:7],test=testdf[,1:7],cl=traindf$type,k=floor(sqrt(nrow(
          dataset))))
      acc[i,]<-classaccuracy(testdf$type,pred)

      predn<-knn(train=traindf_n[,1:7],test=testdf_n[,1:7],cl=traindf_n$type,k=floor(sqrt
          (nrow(datasetn))))
      accn[i,]<-classaccuracy(testdf_n$type,predn)

    #classmatrix<-data.frame(overall_acc=rep(NA,1),int_acc=rep(NA,1),high_acc=rep(NA,1),
        med_acc=rep(NA,1),stringsAsFactors=F)
    #classmatrix[1,]<-predaccn
    }
    predacc<-colMeans(acc)
    predaccn<-colMeans(accn)
    classmatrix<-rbind(predacc,predaccn)
    rownames(classmatrix)<-c("knn","knn, norm")
    return(round(classmatrix,4))
  }
  else{
    return("invalid classification method. Try: 'naive Bayes' or 'decision tree pruned'."
        )
  }
}

acc_comp_tot<-function(k,dataset,datasetn){
  methods<-c("decision tree","naive Bayes","decision tree pruned","knn")
  classmatrix<-data.frame(overall_acc=rep(NA,7),int_acc=rep(NA,7),high_acc=rep(NA,7),med_
      acc=rep(NA,7),time=rep(NA,7),stringsAsFactors=F)
  for(n in methods){
    start.time<-Sys.time()
```

```
    row<-classcomp(k,dataset,datasetn,n)
    end.time<-Sys.time()
    row<-cbind(row,time=as.numeric(difftime(end.time,start.time,units="secs")))
    classmatrix<-rbind(classmatrix,row)
    }
  return(na.omit(classmatrix))
}
}

#################
# analyse data #
#################

# [1] Use activity function and check whether activity detection was correct
#a plot will be generated where red lines indicate the start and end times for the
    detected activity periods
#if too few activities are detected, lower the percentage in act; increase it if too much
     is detected
df<-ClassMeasures(accel,secs)
plot(df$R,type="l",main=paste("Cross-correlation Acceleration data sensor ",exp,"-",sens,
    sep=""),ylab="Acceleration norm")
abline(v=frame(act(df,lowlevel))/(200*secs),col="red")

#If the correct level is found, save the begin and endtimes
times<-frame(act(df,lowlevel))

# [2] now label the activities according to the intensity, on a scale 1-3
acttype<-timetype(setframe,times,highact,medact)

#check whether activity labelling went well
plot(1+2*df$R/max(df$R),type="l",main=paste("Cross correlation of the acceleration",sens)
    )
lines(sapply(acttype,numconv),col="red")

#add labelling to data frame with measures, create normalised and regular data frame
dfull<-data.frame(df,type=acttype,sensor=paste("Ex",exp,sens,sep=""))
dfulln<-data.frame(sapply(df,normalise),type=acttype,sensor=paste("Ex",exp,sens,sep=""))

#add now labelled data frame to larger dataset containing previous data frames
# [3] run first two lines if this is the first iteration, otherwise run the latter two
if(first == T){
  dftotal<-dfull
  dftotaln<-dfulln
}
if(first == F){
  dftotal<-rbind(dftotal,dfull)
  dftotaln<-rbind(dftotaln,dfulln)
}
```

C.2 Code for predicting throwing speed

```
library(car)
library(lattice)
library(nlme)
library(lme4)
library(ggplot2)
require(splines)

setwd("~/.../Fastball")
mdat<-read.csv("Fastball_len.csv",sep=";")

##############
# data prep #
##############

mdat$ageyrs<-mdat$Age_in_months/12
mdat<-mdat[complete.cases(mdat[,6]),]
mdat<-mdat[!(mdat$Age_in_months>216 | mdat$Age_in_months<144),]

mdat<-mdat[-c(297),] #remove one outlier
mdat$ABSSq<-mdat$AverageBallSpeed^2/100
mdat$ABSSq<-mdat$AverageBallSpeed^2/100

mdatana<-mdat[,c(1,4,5,6,17,18,19,20,23,26,27,28)]
mdatana$age12<-mdatana$ageyrs-12

mdatfull <- na.omit(mdatana)

###################
# lme covariates #
###################

FitHeight<-lme(Height ~ bs(age12,knots=c(3)),random=~1|PPnumber, data = mdatfull,method="
    ML")
FitWeight<-lme(weight ~ Height + age12, random = ~1+age12|PPnumber,data=mdatfull,method="
    ML")
FitF_IR<-lme(F_IR_DA ~ Height + age12, random = ~1|PPnumber,data=mdatfull,method="ML")
FitF_ER<-lme(F_ER_DA ~ Height + age12, random = ~1|PPnumber,data=mdatfull,method="ML")
FitROM_IR<-lme(ROM_IRsh_DA~ Height + age12, random = ~1|PPnumber,data=mdatfull,method="ML
    ")
FitROM_ER<-lme(ROM_ERsh_DA ~ Height + age12, random = ~1|PPnumber,data=mdatfull,method="
    ML")

MLMfull<-lme(ABSSq ~ age12+I(age12^2)+Height+weight+F_ER_DA+F_IR_DA+ROM_ERsh_DA+ROM_IRsh_
    DA,
                     random = ~ 1 + age12 | PPnumber, data=mdatfull, na.action=na.omit,
                         method="ML")
MLMfull2<-lme(ABSSq ~ age12+I(age12^2)+Height+weight+F_ER_DA+ROM_ERsh_DA,
              random = ~ 1  | PPnumber, data=mdatfull, na.action=na.omit, method="ML")

predageSq<-function(fit,age,ID){
  h<-predict(FitHeight,data.frame(age12=c(age),PPnumber=c(ID)),level=1)
  dat<-data.frame(age12=c(age),Height=c(h),PPnumber=c(ID))
  w<-predict(FitWeight,dat,level=1)
  fir<-predict(FitF_IR,dat,level=1)
  fer<-predict(FitF_ER,dat,level=1)
  rir<-predict(FitROM_IR,dat,level=1)
  rer<-predict(FitROM_ER,dat,level=1)

  newdat<-data.frame(age12=c(age),weight=c(w),Height=c(h),PPnumber=c(ID),ROM_IRsh_DA=c(
      rir),ROM_ERsh_DA=c(rer),F_IR_DA=c(fir),F_ER_DA=c(fer))
  speed<-predict(fit,newdat,level=1)
  return.dat<-cbind(newdat,ABSSq1=speed)
  return(speed)
}

set.seed(1)
plotPPnumber<-sample(unique(mdatfull$PPnumber),10,replace=F)
PPindex<-which(mdatfull$PPnumber %in% plotPPnumber)
```

```
mdatplot<-mdatfull[PPindex,]

predframe<-with(mdatplot,expand.grid(PPnumber=unique(as.character(PPnumber)),age12=seq
    (0,6,0.1)))
predSp<-c()
for(i in 1:nrow(predframe)){
  sp<-predageSq(MLMsqrtfull,predframe[i,2],as.character(predframe[i,1]))
  predSp<-c(predSp,sp)
}
predframe$AverageBallSpeed<-predSp

ggplot(mdatplot,aes(age12,AverageBallSpeed,colour=PPnumber))+ggtitle("Predicted Throwing
    Speed") +
  xlab("Age in years") + ylab("Throwing speed in mph") + theme(plot.title = element_text(
      hjust = 0.5)) + labs(col="Participant") +
  geom_point()+ geom_line(data=predframe) + scale_x_continuous(breaks=c(0,2,4,6),labels=c
      ("12", "14", "16","18"))

avgspeed<-function(fit,age){
  h<-predict(FitHeight,data.frame(age12=c(age)),level=0)
  dat<-data.frame(age12=c(age),Height=c(h))
  w<-predict(FitWeight,dat,level=0)
  fir<-predict(FitF_IR,dat,level=0)
  fer<-predict(FitF_ER,dat,level=0)
  rir<-predict(FitROM_IR,dat,level=0)
  rer<-predict(FitROM_ER,dat,level=0)

  newdat<-data.frame(age12=c(age),weight=c(w),Height=c(h),ROM_IRsh_DA=c(rir),ROM_ERsh_DA=
      c(rer),F_IR_DA=c(fir),F_ER_DA=c(fer))
  speed<-predict(fit,newdat,level=0)
  return.dat<-cbind(newdat,ABSSq1=speed)
  return(speed)
}

avgframe<-data.frame(age12=c(seq(0,6,0.1)))
avgSp<-c()
for(i in 1:nrow(avgframe)){
  sp<-avgspeed(MLMsqrtfull,avgframe[i,1])
  avgSp<-c(avgSp,sp)
}
avgframe$AverageBallSpeed<-avgSp

ggplot(mdatana,aes(age12,AverageBallSpeed),pch=17)+ggtitle("Predicted Throwing Speed") +
  xlab("Age in years") + ylab("Throwing speed in mph") + theme(plot.title = element_text(
      hjust = 0.5)) + labs(col="Participant") +
  geom_point(shape=1)+ geom_line(data=avgframe,col="red",lwd=1) + scale_x_continuous(
      breaks=c(0,2,4,6),labels=c("12", "14", "16","18"))
```