Improving the signal processing path for online heartbeat estimation by using Ultra Wideband radar

Petter Moe Kvalvaag

June 14, 2009

Abstract

A number of companies, research establishments and universities worldwide are currently researching the use of Ultra Wideband radar technology to conduct non-contact and non-invasive measurements of heartbeat.

Both cardiac and respiratory activity cause a visible and measurable motion to the chest wall. When aiming a radar to a subject's chest, this motion can be recorded and processed to obtain the rates of respiration and heartbeat.

This thesis looks into the processing of chest wall displacement data to improve heart rate measurements. At the first stage of the signal processing, the low frequency, high amplitude respiration signal component must be separated from the much smaller heartbeat component. Traditionally, linear Finite Impulse Response (FIR) highpass filters have been used to remove the respiration component. FIR filters have a fixed corner frequency. If the respiration rate is higher than this fixed frequency, FIR filtering can not work. In addition, FIR filters must be quite long for sufficient stopband attenuation, and thus they introduce a time delay to the system. Alternative signal processing techniques are based on cancellation of respiration harmonics, which eliminates the system delay and works at any respiration frequency. One such method from the literature was investigated, and this thesis presents a modified version that works better with actual radar data.

After separation, the heart rate must be determined. Traditionally, frequency estimation of a sliding window has been used. A different method from the literature is based on adaptive filtering. It transforms the heartbeat signal into a train of unit pulses, each occurring at the start of a heartbeat. Instant heartbeat information is thus gathered, as opposed to the traditional method which yields the *mean* heart rate during the time window.

Some system results were obtained for both constructed test signals and actual radar data. Generally FIR filtering performed better than harmonic cancellation, but the harmonic canceller outperformed FIR filtering when the respiration frequency tended towards the filter's corner frequency. Feasibility of a system that gives instant heart rate measurements in real-time was proven. The improvements on traditional systems are elimination of time delay, enabling separation at high respiration frequencies and giving the instant heart rate rather than an avereraged one.

PREFACE

The idea of this work came to life during my summer internship at the Norwegian Defence Research Establishment. Initially we planned that under mentoring of professor John Håkon Husøy at the University of Stavanger and professor Svein-Erik Hamran from the Norwegian Defence Research Establishment at Kjeller, the thesis would take shape.

Immediately after the work started I obtained contact with Mr. Terje Hauan, Chief Executive Officer at MicroImpulse AS. MicroImpulse is a Stavanger-based company researching the use of Ultra Wideband radar for non-contact, non-invasive sensing of cardiopulmonary activity. We decided that liaisoning with a Stavanger-based company is much easier than travelling to Kjeller, and it was possible to do the same exercises with MicroImpulse's equipment and data.

Thus my mentoring has primarily come from MicroImpulse's Russian daughter company, NanoPulse, who are responsible for developing the hardware and software. Thanks are directed to Chief Technology Officer at NanoPulse, Mikhail Numerov, for providing data and explanations of their systems and the necessary software. Thanks to John Håkon Husøy for helping out with the details of the mathematics and the format of the report. Thanks also to Svein-Erik Hamran for introducing me to this very interesting subject. Finally, I'd like to thank my fellow students for their valuable inputs and conversations on the topic, and for creating a great environment in which to work.

Stavanger, 15. June 2009

Petter Moe Kvalvaag

NOTATION

The following describes the notations used in this thesis:

Scalars are written with upper- or lowercase characters in normal fonts

Matrices are written with bold uppercase letters, e.g. $\mathbf{X}$

Vectors are written with bold lowercase letters, e.g. $\mathbf{x}$

Transpose of a vector or a matrix is indicated by ^{T}, e.g. $\mathbf{X}^T$

Gradient of a multidimensional function with respect to $\mathbf{x}$ is $\nabla_{\mathbf{x}}$.

Complex conjugates of matrices and scalars are indicated by *, e.g. x^*

Inverted complex conjugate matrices are indicated by $^{-*}$, e.g. $\mathbf{A}^{-*}$

Absolute value of a scalar is written $|\cdot|$

P-norm is denoted $\|\cdot\|_p$, and p is always specified although only the *euclidian norm*, or 2-norm, has been used.

Argument minimizer/maximizer. The argument a that minimizes $f(a)$ is denoted $\arg\min_a f(a)$.

Cost functions, usually the sum of squared residuals, are always called $S(\mathbf{x})$ where $\mathbf{x}$ is a vector of parameters.

Big-O notation is used about algorithmic cost and approximation errors. In the context of algorithmic cost, $O(N^3)$ means that the number of calculations in the algorithm has the same *order of magnitude* as the number of parameters N raised to the third power. When used in the context of approximation errors, e.g. $O(\|\mathbf{x}\|_2^2)$, it means that the approximation error has the same *order of magnitude* as the two-norm of $\mathbf{x}$ squared.

ABBREVIATIONS

The following table lists a number of common abbreviations used in this thesis alphabetically.

Abbreviation	Meaning
AM	Amplitude Modulation
Bpm.	Context specific. *Breaths pr. minute* for respiration or *beats pr. minute* for heart rates
FIR	Finite Impulse Response
FM	Frequency Modulation
H&J	Hooke and Jeeves
IIR	Infinite Impulse Response
MIR	Micro Impulse Radar
SNR	Signal-to-Noise Ratio
UWB	Ultra Wideband

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Background

Since the 1970's researchers have been exploring the concept of using microwave radars to measure respiration and heartbeat in humans and animals. As humans breathe the lungs will expand and contract, causing a periodically repeating movement in the chest wall. Similarly, heartbeats will cause a periodic motion to the chest wall.

It is hypothesized that if this motion can be measured, a time series containing successive measurements for a period of time can be processed to gather information about the subject's cardiac and respiratory activity.

To explore the concept further, imagine aiming a radar at a patient's chest. There will be a measurable distance $x(t)$ between the patient's chest and the radar antenna cf. FIG. 1.1.

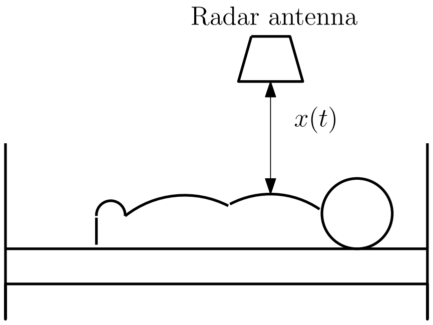
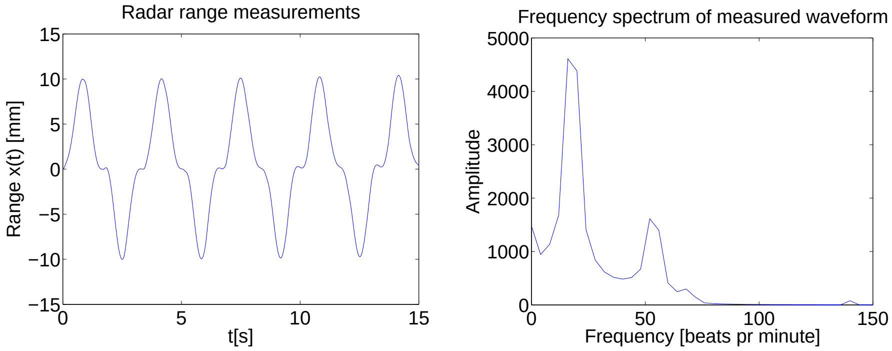


FIG. 1.1: Concept sketch, patient in bed with radar based vital signs monitor.

The chest wall motion caused by cardiac and respiratory activity manifests itself as miniscule variations in this distance $x(t)$. By using high resolution range detection, a sampled discrete time series $x(n)$ containing the waveform of the chest motion can be gathered. As an example, the subject could be breathing at a (typical) rate of 18 cycles pr. minute, and his/her heart beat at a rate of 50 beats pr minute. Further, assume that all chest wall motion is sinusoidal (it is not, this is addressed later). If the radar was capable of sampling $x(t)$ each 0.1s, then that time series would look like FIG. 1.2(a). From this data the frequency spectrum could be gathered by applying the Discrete Fourier Transform, yielding FIG. 1.2(b).



(a) Measured waveform $x(n)$ (b) Estimated frequency spectrum

FIG. 1.2: Radar measurements and frequency estimates of $x(t)$

Modelling the motion as a sum of sinusoids is not very accurate. A more realistic model is introduced in SECTION 4.2.

1.2 Today's system

Today's system uses a combination of signal processing techniques to obtain the patient's heartrate. In the first stage, a linear highpass Finite Impulse Response (FIR) filter is applied to remove the low frequency respiration signal component. After that, frequency estimation is conducted by using the autocorrelation function of a sliding window of the remaining heartbeat signal.

FIR filtering has two key disadvantages. Firstly, a long filter is required for sufficient stopband attenuation. This introduces a signal delay, and thus the patient's instantaneous heartrate cannot be obtained. In many situations however, a few seconds delay is completely acceptable. Secondly, and worse, is the fact that the FIR filter has a constant corner frequency, somewhere in between the high end of "normal" respiration frequencies and the low end of "normal" heartbeat frequencies. Naturally these parameters are bound to vary with the patient's age, physique and especially the monitoring situation.

The autocorrelation method for calculating the heartrate is also flawed in some ways. For an accurate frequency estimate, we have by experiment established that a window of at least 7 seconds is required (for 60 beats pr. minute). Essentially that means that at any time, it is the patient's *average heart rate between two to nine seconds ago* (taking into account a two second delay from the filter) that is measured. Because of the problems inherent to the methods presently used, new signal processing strategies must be devised.

1.3 Problem formulation

This thesis focuses on an alternative method for the separation of heart and respiration. A primary goal is to overcome the problem of high respiration frequencies not being cancelled. A secondary goal is to eliminate the time delay of the filter. As a supplement to the elimination of time delay, instant measurement of the heart rate is also sought. This would allow measurement of the *heart rate variability*, that is, the beat-for-beat variations in the heart rate. All these issues are geared towards making the heart rate measurements more *robust* and *instantaneous*.

1.4 Report contents

Chapter 2 provides the reader with some more background to the problem, including an overview of different radar technologies that have been applied in creating similar systems. Chapter 3 describes the alternative signal processing methods that have been investigated in this thesis. Chapter 4 moves on to describe how the data used in the thesis were obtained. Chapter 5 presents the results that were obtained, and these are discussed in Chapter 6. Finally a conclusion can be found in Chapter 7.

1.5 Recommended previous knowledge

To fully enjoy and endure reading this thesis, it is recommended that the reader has a sound background within signal processing. An interest in linear algebra and linear and nonlinear optimiziation techniques will also make the thesis more readable.

CHAPTER 2

AN OVERVIEW OF MICROWAVE SENSING OF CARDIOPULMONARY ACTIVITY

2.1 Introduction

Already in Lin [1975] the idea of using a microwave device to measure respiration was conceived. Since then the concept has been expanded to cover heart rate measurements, and several methods and approaches have been developed and tested. However, a major commercial breakthrough of such a system is yet to be claimed. The system variations have primarily been in the choices of range measurement device, thus some of the major radar classes along with a brief description will be presented in the next sections.

While reading this chapter, it should be kept in mind that the problem formulation worked on in this thesis is *completely transparent* to which type of radar is used. To understand the work conducted, it is only necessary to acknowledge that *successive measurements of the chest wall displacement* are obtained, one way or the other. This chapter merely provides an insight into which technologies are available and used today.

2.2 Introduction to radar systems

Radar is an acronym for RAdio Detection And Ranging. It can be used to detect the range, size and velocity of nearby objects that reflect microwaves or radio waves. Since the first devices made in the early 1900's the technology has matured into precise detection and ranging tools, for a wide array of applications.

Skolnik [2008] provides both a comprehensive introduction and detailed information on a variety of radar technologies, and serves as a reference to large parts of this chapter.

2.3 Doppler radar implementations

The term *doppler radar* refers to all radars applying the doppler effect to determine the radial velocity of a detected object.

2.3.1 Description of the doppler effect

The doppler effect describes what happens when any type of wave is transmitted or reflected from a moving object. A common example is the audible frequency shift occurring when an ambulance approaches quickly. Once the ambulance drives by the pitch of the siren is clearly lower than it was when it was approaching. The same effect occurs when sound or electromagentic radiation is reflected from a moving object cf. FIG. 2.1. This is due to sound waves being "squeezed" during the approach and "dragged out" when the ambulance drives away.

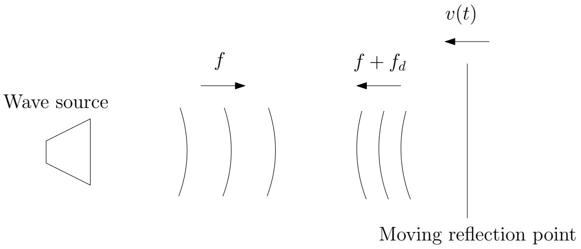


FIG. 2.1: Doppler effect in moving object.

The frequency shift f_d can be described mathematically, in the case of reflection from a moving object this is defined as

$$f_d = \frac{2v(t)}{\lambda}, \quad \lambda = \frac{c}{f} \tag{2.1}$$

where c denotes the speed of light and f is the frequency of the source wave. The velocity $v(t)$ is defined such that objects moving towards the source have a positive velocity.

2.3.2 Radar application of the doppler effect

The doppler effect is applied in radars to determine the velocity of the detected object. Continuous Wave (CW) systems transmit one frequency continuously. By continuously monitoring the *difference between transmitted and received frequencies*, f_d, equation (2.1) is used to obtain the velocity of the reflecting object.

2.3.3 Heart rate measurements using doppler radar

Most of the work conducted in the field of microwave cardiopulmonary sensing has been based on various types of doppler radars. The doppler radar only yields information about *moving targets*, but since periodic motion is measured, displacement $x(t)$ relates to velocity $v(t)$ such that

$$x(t) = \frac{d}{dt}v(t) \tag{2.2}$$

and in the case of sinusoidal motion

$$\frac{d}{dt}\sin(t) = \cos(t) \tag{2.3}$$

and the velocity measurement is essentially a phase shifted measure of the displacement.

A complete description of such a system is described in Droitcour [2006]. Other recent work is found in Massagram et al. [2007] and Morgan and Zierdt [2009].

2.4 Ultra Wideband radar implementations

2.4.1 Introduction

The term Ultra Wideband (UWB) refers to a frequency spectrum that has been allocated by the U.S. Federal Communications Commission for unlicenced use. Contrary to conventional narrowband technologies such as FM and AM, the point of Ultra Wideband technology is to *spread the spectrum* such that the emission at each individual frequency is much weaker, allowing the total transmitted power to be the same. One can imagine smearing the Dirac's delta-like impulse of narrowband radio out over a much broader spectrum. The interested reader will confer to Appendix D for a motivation for using the Ultra Wideband concept, from an information theoretical viewpoint. For a thorough introduction to Ultra Wideband radio technology, see Siwiak and McKeown [2004].

2.4.2 Definitions and regulations

By definition, electromagnetic radiation covering a spectrum broader than 500 MHz *or* has a fractional bandwidth of more than 0.2, is considered to be Ultra Wideband. The fractional bandwidth is defined as

$$BW_f = 2\frac{f_h - f_l}{f_h + f_l} = \frac{f_h - f_l}{f_c} \tag{2.4}$$

where f_h and f_l are the upper and lower frequencies at the *10 dB point*, that is, the frequency at which the spectral emissions are 10 dB lower than the highest frequency of the spectrum. f_c is the center frequency defined as $\frac{1}{2}(f_h + f_l)$. Any such spectrum is allowed between 3.1 GHz and 10.6 GHz.

The reason that it has been possible to allocate such a broad band to UWB radios is that the band actually overlaps bands allocated to other users. Thus there is a restriction on the *maximum allowable spectral density*, and this is defined such that transmitted power at any given frequency is so low that other devices on that frequency will consider the transmission as noise.

2.4.3 Micro Impulse Radar

There are many types of Ultra Wideband radar implementations. A common implementation is the one presented in Staderini [2008], which is known as Micro Impulse Radar (MIR). This type of UWB radar transmits ultra short pulses in time, typically in the order of hundreds of picoseconds. As an example, a pulse of 300 ps duration and its corresponding spectrum is found in Fig. 2.2. The 10 dB point is here around 5 GHz and thus the pulse more than satifies the requirements for being called Ultra Wideband. Due to the frequency response of the antenna, this pulse changes characteristics and is typically shifted up in frequency so that it meets the 3.1-10.6 GHz bandlimit.

Pulses that short in time cannot readily be sampled. It would simply require too high a sample rate for a sensible implementation. This turns the concept of *range detection* upside down. As will be explained, the radar is actually forced to *assume* a range, and then *check* if there is an object present there.

The amplitude of the reflection is sampled at the time *one expects* an echo. To understand the concept better, cf. to Fig. 2.3 (from Staderini [2008]). A pulse is transmitted, and a copy of the pulse is delayed through a high resolution *delay line*. This delay line controls the *expected distance* of the reflector.

If the radar controller has guessed the correct distance to a target, the sampler is going to record a high amplitude sample. If there is no reflecting object at the guessed distance, the sampler is going to record a low amplitude sample. The samples are accumulated in a low pass filter to eliminate noise from other units transmitting at interfering frequencies.

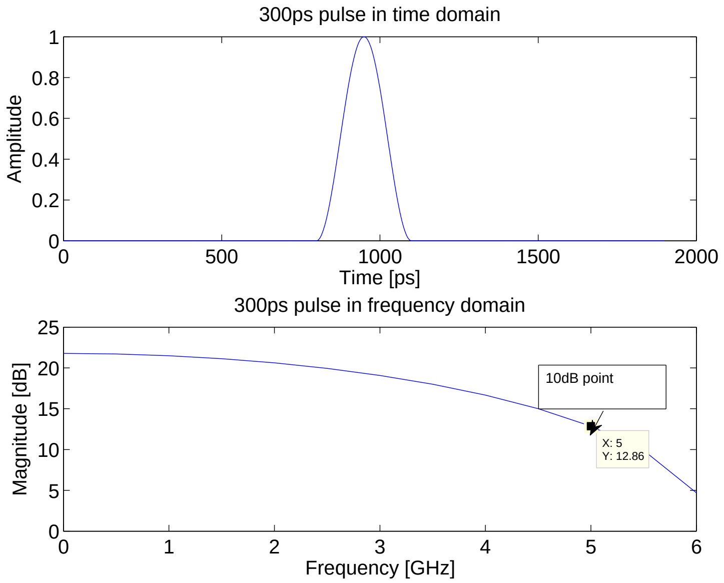


FIG. 2.2: Time and frequency domain represenation of a typical Ultra Wideband pulse.

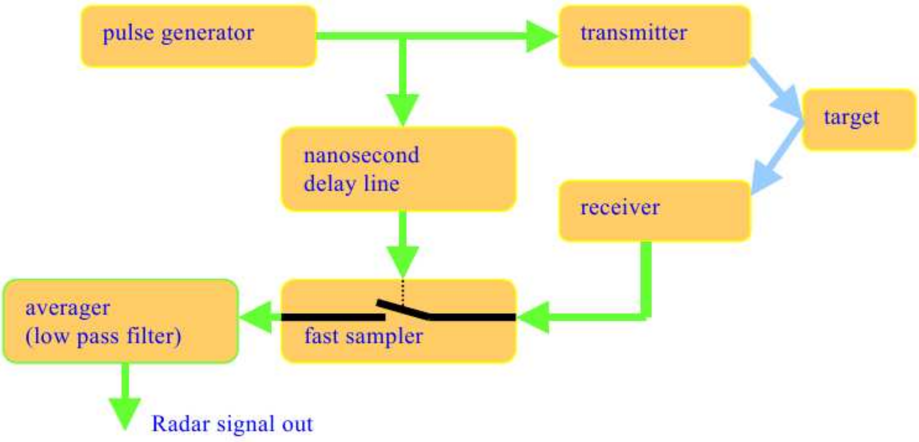


FIG. 2.3: Block diagram of Ultra Wideband radar Staderini [2008].

When using this type of radar, one actually needs to *sweep* over all possible distances to locate the target. After an initial sweep the radar has found its operating range. Then it can start sweeping over the expected motion of the chest wall to gain a time series containing successive measurements of the distance between the antenna and the chest wall.

This type of radar was used in cardiopulmonary sensing in Michahelles et al. [2004].

2.4.4 Ultra Wideband radar, NanoPulse type

The Ultra Wideband radar developed by NanoPulse has been used to gather the data used in this thesis. In order to satisfy the spectrum requirements of Ultra Wideband, a 2 ns long pulse has been *mixed up in frequency* with a sinusoid at 7.4 GHz. The mixer is an analog multiplicator, and thus the transmitted pulse and corresponding spectrum will look something like FIG. 2.4.

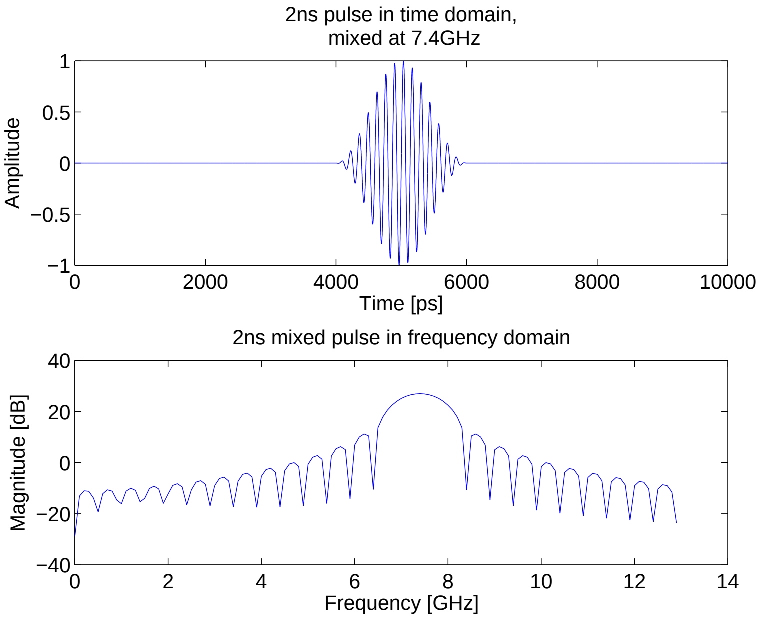


FIG. 2.4: Ultra Wideband pulse satisfying spectral constraints

At the receiving end the pulse is mixed down again, and a low-pass filter is used to regain the 2 ns long pulse. Another low-pass filter with a very low corner frequency (20 Hz) is used to accumulate pulses. Then finding the distance to the target is a matter of estimating the *time delay* of the received pulse.

2.5 Summary

A brief introduction to several possible chest wall displacement measurement techniques were presented. All microwave radar implementations have the advantage that they can “see through” clothes and, depending on the frequency, other obstacles like rocks and walls. Thus the technology can be used in a number of scenarios like life detection under earthquake rubble or through-wall sensing of persons and their movement, respiration and heart rate.

All technologies face challenges of getting sufficient resolution for detecting the miniscule motion caused by the heartbeat. In that respect, laser measurements of the chest wall displacement can be conducted with great accuracy and resolution. However, a laser cannot see through any obstacles, and thus many of the microwave radar advantages are forfeit.

CHAPTER 3

SIGNAL PROCESSING TECHNIQUES FOR RADAR SENSING OF THE HEARTBEAT

3.1 Introduction

This chapter describes some of the methods used subsequently. Most of the practical work in this thesis deals with extracting the correct heart rate from noise-laden radar range measurements from the chest wall.

At the first stage in the processing, the challenge is to separate the heart signal from the respiration signal. The simplest way to do this is by linear filtering. A different method based on cancellation of respiration harmonics is also investigated. The latter is based on the work in Morgan and Zierdt [2009], but a slightly different approach was made.

For calculating the heart rate two methods will be investigated. One is based on the short-time autocorrelation function (STAF), and the other takes a different approach by using adaptive filtering.

3.2 Separating heartbeat from respiration using linear filtering

3.2.1 Introduction

Linear filtering is a commonly used signal processing technique to remove unwanted signal components. The technique is described in most literature on digital signal processing, e.g. Proakis and Manolakis [2006]. Two major classes of filters are finite impulse response (FIR) and infinite impulse response (IIR) filters. All linear digital filters can be described by the difference equation

$$\begin{aligned} y(n) &= a_1 y(n-1) + a_2 y(n-2) + \cdots + y_p(n-p) \\ &\quad + b_0 x(n) + b_1 x(n-1) + \cdots + b_q x(n-q) \end{aligned} \tag{3.1}$$

that desribes the output $y(n)$ given the input $x(n)$, for a filter with p poles and q zeros. The transfer function is described in the z plane as

$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots b_q z^{-q}}{1 + a_1 z^{-1} + \cdots a_p z^{-p}} \tag{3.2}$$

The transfer function's zeros are determined by the b parameters and are often referred to as the *moving average* parameters. The poles are determined by the a parameters and are often referred to as the *autoregressive* parameters. An *all-pole* filter contains only autoregressive components, and is thus a pure IIR filter. An *all-zero* filter contains only moving average parameters, and is thus a FIR filter. Filters containing both parameter types are IIR filters, due to the autoregressive part.

In the case of separating heartbeat signals from respiration signals, we mainly want to look at the *frequency response* of the transfer function. This indicates how frequency components are attenuated or amplified through the filter. Cf. FIG. 3.1 for an example transfer function of a lowpass FIR filter with the normalized corner frequency 0.2 and 8 filter coefficients.

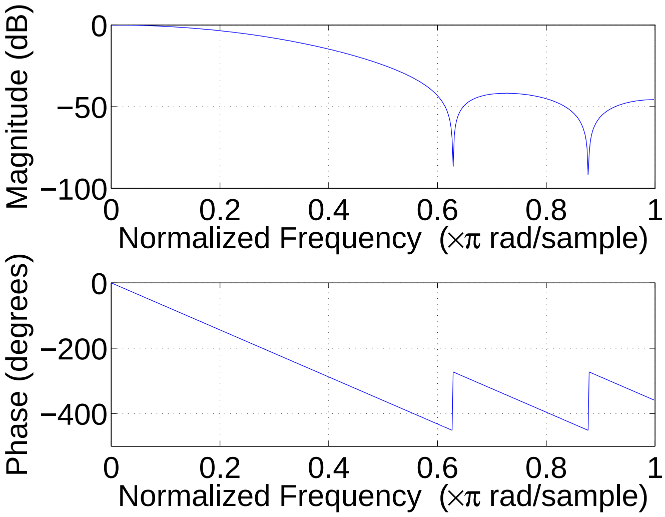


FIG. 3.1: Example FIR low-pass filter.

3.2.2 Designing a filter for removing respiration signal components

When designing a suitable filter for removing the respiration component, a few key properties must be considered.

It was decided to use an FIR filter, because *linear phase* is required. Linear phase is hard to obtain when using IIR filters. The filter's phase $\theta(\omega)$ is related to the *group delay* $D(\omega)$ such that

$$D(\omega) \equiv \frac{d}{d\omega}\theta(\omega) \tag{3.3}$$

and consequently linear phase results in a constant group delay.

A constant group delay ensures that all frequencies are delayed equally in time, and if this is not the case we risk a distortion of the heart waveform.

Generally we want as much stopband attenuation as possible. One possibility is to let the acceptable time delay decide how much stopband attenuation we get. A higher order filter will yield more stopband attenuation, but consequently the group delay through the filter increases.

The acceptable time delay will vary with the intended use of the system. However, it was found that there is not much improvement on the filtered signal when going beyond 400 filter coefficients. This results in a group delay of 200 samples, or 2 seconds at sample rate $f_s = 100$ Hz.

With this background it was decided to use a 400 coefficients long FIR filter where the impulse response has been windowed with the Kaiser window using $\beta = 3$. The corner frequency f_c was chosen to be 0.65 Hz, which is somewhere in between the lower end of expected heart rates(50-90 bpm. = 0.83-1.5 Hz) and the upper end of expected respiration rates (13-24 bpm. = 0.22-0.4 Hz). This filter's frequency response is found in FIG. 3.2.

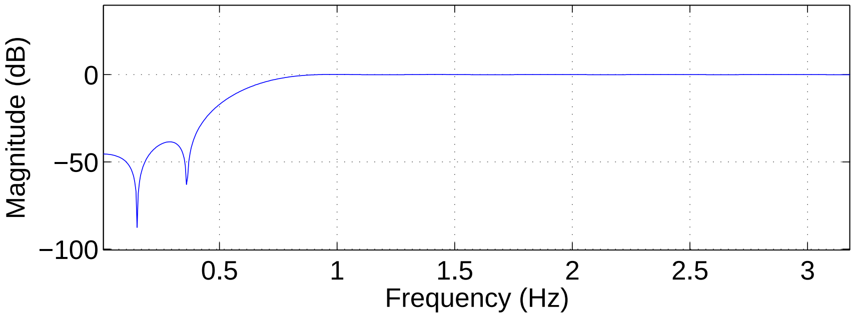


FIG. 3.2: Frequency response of filter for separating heartbeat signal components from respiration signal components. 400 tap FIR filter with Kaiser windowed impulse response, $\beta = 3$, corner frequency 0.65 Hz.

3.2.3 Removing out-of-band noise

A 20 coefficients long FIR filter has also been applied for removing out-of-band noise. Cutting off at 10 Hz generally yields better heartbeat detection. The filter's frequency response is found in Fig. 3.3.

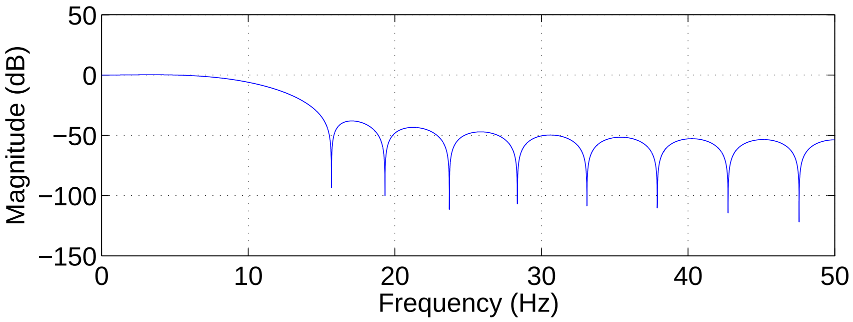


Fig. 3.3: Frequency response of filter for removing out-of-band noise. 20 tap FIR filter with Kaiser windowed impulse response, $\beta = 3$, corner frequency 10Hz

3.3 Separating heartbeat from respiration using subtraction of harmonic components

3.3.1 Introduction

One problem with separation of heart and respiration using linear filtering is that both respiration and heartbeat frequencies vary from subject to subject and from situation to situation, and they are by no means stationary. For example, during patient hyperventilation the respiration frequency could tend towards the frequency of the heartbeat, rendering linear filtering methods useless.

A new signal processing method for separation is presented in Morgan and Zierdt [2009]. The idea is to model the time series $x(n) = s(n) + M(n)$ where $s(n)$ is the desired heart signal and $M(n)$ is a periodic component stemming from respiration that we wish to remove. Further, $M(n)$ consists of L harmonic components of the fundamental respiration frequency, and a constant DC term that applies if the data are not zero-mean.

The algorithm will first try to estimate the fundamental respiration frequency, hereafter called f_0, by using an autocorrelation technique. Then an optimal set of $2L+1$ weigths x_l consisting of the *amplitudes* and *phases* of the L harmonic components and a DC term x_{2L+1} is found. Several approaches were tested, but the Gauss-Newton method for nonlinear optimization was best suited for the task. The resulting weights are used to reconstruct an estimate of $M(n)$. The cost function is defined as the sum of the squared residuals $r(n) = y(n) - M(n)$, and this can be seen as a datafitting problem.

3.3.2 Defining a suitable model

As stated in the literature, e.g. Morgan and Zierdt [2009], the chest wall motion arising from respiration can be modelled as the sum of sinusoids with the fundamental frequency ω_0 and a number of harmonic components with frequencies $l\omega_0$. A model M could be formulated thus

$$M(\mathbf{x}, n, \omega_0) = \sum_{l=1}^{L} x_{2l-1} \cos(l\omega_0 n + x_{2l}) + x_{2L+1} \tag{3.4}$$

where $\mathbf{x}$ is the parameter vector containing $\begin{bmatrix} x_1 & x_2 & \cdots & x_{2L+1} \end{bmatrix}^T$ and ω_0 is the normalized fundamental respiration frequency $\frac{2\pi f_0}{f_s}$. x_i is an amplitude component for odd i and a phase component for even i, and a constant DC term for $i = L + 1$.

3.3.3 Defining an optimization problem

With the i'th observed sample contained in y_i, the residual is defined

$$r_i = y_i - M(\mathbf{x}, i, \omega_0) \tag{3.5}$$

At sample m, the optimal $\mathbf{x}$ that minimizes the sum of squared residuals in N samples from the past is sought, and a cost function is defined thus

$$S(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|_2^2 = \frac{1}{2} \sum_{i=m-N+1}^{m} (r_i(\mathbf{x}))^2 \tag{3.6}$$

3.3.4 Choosing a solution

A number of methods exist in the literature for solving such least-squares problems. The fact that the residual model is nonlinear in $\mathbf{x}$ [1] constrains us to use a *nonlinear least squares* method.

The Gauss-Newton method described in APPENDIX C has proved to solve the problem quickly and reliably. The Gauss-Newton method is an iterative algorithm that solves the cost function in equation (3.6) thus

1. Solve

$$\mathbf{J}^T \mathbf{J} \mathbf{h}_{gn} = -\mathbf{J}^T \mathbf{r} \tag{3.7}$$

[1] A linear model of the residuals can be written as the linear combination $\mathbf{r} = \mathbf{y} - \mathbf{A}\mathbf{x}$, which is not the case due to the phase component of the sinusoid, cf. equation (3.4).

2. Update $\mathbf{x}$

$$\mathbf{x} = \mathbf{x} + \mathbf{h}_{gn} \tag{3.8}$$

$\mathbf{J}(\mathbf{x})$ is the Jacobian matrix defined as

$$\mathbf{J}(\mathbf{x}) = [N \times n] = \begin{bmatrix} \frac{\partial r_1}{\partial x_1} & \frac{\partial r_1}{\partial x_2} & \cdots & \frac{\partial r_1}{\partial x_n} \\ \frac{\partial r_2}{\partial x_1} & \frac{\partial r_2}{\partial x_2} & \cdots & \frac{\partial r_2}{\partial x_n} \\ \vdots & & & \vdots \\ \frac{\partial r_N}{\partial x_1} & \frac{\partial r_N}{\partial x_2} & \cdots & \frac{\partial r_N}{\partial x_n} \end{bmatrix} \tag{3.9}$$

for N samples and n parameters in $\mathbf{x}$.

It can be shown (e.g. in Appendix A of Madsen et al. [2004]) that if the columns of $\mathbf{J}$ are *linearly independent*, and the system is critically determined or overdetermined ($N \geq n$), then $\mathbf{A} = \mathbf{J}^T\mathbf{J}$ is positive definite. Thus $\mathbf{h}_{gn}$ has a unique solution that is found by solving equation (3.7).

3.3.5 Comparison with standard solution

The solution found in Morgan and Zierdt [2009] theoretically has a few advantages over the Gauss-Newton method. These lie in the model and consequently the cost function formulation used:

$$M(\mathbf{x}, n, \omega_0) = \sum_{l=1}^{L} A_l \cos(l\omega_0 n + \phi_l) \tag{3.10}$$

$$= \mathbf{Re}\left\{\sum_{l=1}^{L} A_l e^{j\phi_l} e^{j\omega_0 n}\right\} \tag{3.11}$$

$$= \mathbf{Re}\left\{\sum_{l=1}^{L} x_l e^{j\omega_o n}\right\} \tag{3.12}$$

The DC term, call it x_{DC}, is not included in the model and is thus not optimized over. It is rather determined by

$$x_{DC} = \frac{1}{N} \sum_{i=m-N+1}^{m} y(i) - M(\mathbf{x}, i, \omega_0) \tag{3.13}$$

which is the mean value of the deviation between the measured time series and the model. This is intuitively the DC term that minimizes the cost function.

The cost function can now be formulated

$$S(\mathbf{x}) = \sum_{i=m-N+1}^{N} [y_i - M(\mathbf{x}, i, \omega_0) - x_{DC}]^2 \tag{3.14}$$

and a solution to this problem is

$$\mathbf{x} = 2(\mathbf{R} - \mathbf{C}\mathbf{R}^{-*}\mathbf{C}^{*})^{-1}(\mathbf{r} - \mathbf{C}\mathbf{R}^{-*}\mathbf{r}^{*}) \tag{3.15}$$

R and **C** are complex and hermitian $[L \times L]$ matrices, and **r** is an $[L \times 1]$ correlation vector. The details are given in Appendix E.

Indeed this is an elegant solution that requires the inversion of an $[L \times L]$ matrix rather that the inversion of a $[(2L+1) \times (2L+1)]$ matrix every sample. In addition, since it is a solution by standard equation, convergence is immediate. For harmonic cancellation of signals of more stationary nature like the test signals produced in this thesis, this solution works very well.

Despite the theoretical advantages, the method failed to work well on the actual data gathered in this thesis. The problem seems to be that the solution is unstable when using short windows. This causes the optimal parameters to vary so much from sample to sample, that the datafit is a jigsaw-shaped signal. The jigsaw-shape is naturally also reflected in the residuals, jeopardizing good heart rate estimates. It is necessary to use short data windows because people usually do not breathe in equal cycles for prolonged periods of time. Another problem with long windows is the need for an exact estimate of ω_0. When optimizing the amplitudes and phases of harmonics with a non-variable estimated fundamental frequency, it is clear that, when using long windows, the estimated frequency must be very exact to get a good datafit.

3.3.6 Cost of solution

Solving for the Gauss-Newton step in equation (3.7) can be quite costly. Matrix inversion typically costs in the order of $O(N^3)$ which means that we want to keep $\mathbf{A} = \mathbf{J}^T\mathbf{J}$ small. This is however feasible, since using for example $L = 3$ harmonic components gives $2L + 1 = 7$ parameters to solve for, and thus $\mathbf{A} = [7 \times 7]$. $7^3 = 343$, and with sample rate $f_s = 100$ Hz, the matrix inversions of the algorithm require 34 300 flops (floating point operations pr. second). Although this is feasible for a standard computer, the algorithm should be made more efficient for e.g. embedded implementations. A possible amendment is using the Cholesky factorization procedure

$$\begin{aligned}
\mathbf{A} &= \mathbf{C}^T\mathbf{C} && \text{|| Cholesky factors}\\
\mathbf{z}_{gn} &= [\mathbf{C}^T]^{-1}(-\mathbf{J}\mathbf{r}) && \text{|| Intermediate step}\\
\mathbf{h}_{gn} &= [\mathbf{C}]^{-1}\mathbf{z}_{gn} && \text{|| Final step}
\end{aligned}$$

where **C** is upper triangular, the intermediate and final steps are solved by backward and forward substitution (see e.g. C.H. Edwards and Penney [1988] or Madsen et al. [2004] for details on solving systems of linear equation).

Bacward and forward substitution are much cheaper than matrix inversion ($O(N^2)$), but Cholesky factorization is quite expensive. One algorithm (Madsen et al. [2004], appendix A) costs $O(\frac{1}{3}N^3)$, so savings are marginal especially for small N. Therefore it is advisable to develop updating formulas for $[\mathbf{J}^T\mathbf{J}]^{-1}$ and $-\mathbf{J}^T\mathbf{r}$, similar to what is done in recursive least squares filtering cf. Appendix B.

On a final note, Matlab R2008a processes 400 seconds of data in about 14 seconds on a standard AMD Turion64 X2 laptop computer when inverting directly (using the slash operator[2]), and about 13 seconds using Cholesky factorization. However, using the Cholesky factorization algorithm found in Madsen et al. [2004], appendix A ($O(\frac{1}{3}N^3)$), causes a dramatic increase runtime to about 22 seconds. Apparently it is hard to beat Matlab's implementations of the latest and greatest algorithms. This should absolutely be in mind when attempting to program the algorithm in a different programming language.

3.3.7 Applying the Gauss-Newton method in harmonic cancellation

The Jacobian matrix consists of rows $\mathbf{J}_i$ containing the elements of the gradient of the residual vector. For three harmonic components, the i'th row is thus

$$\mathbf{J}_i = \begin{bmatrix} \frac{\partial r_i}{\partial x_1} & \frac{\partial r_i}{\partial x_2} & \cdots & \frac{\partial r_i}{\partial x_7} \end{bmatrix} \tag{3.16}$$

$$\tag{3.17}$$

$$\begin{aligned} \mathbf{J}_i = [\nabla_{\mathbf{x}}[y_i - (x_1\cos(\omega_0 n + x_2) + x_3\cos(2\omega_0 n + x_4) + \\ x_5\cos(3\omega_0 n + x_6) + x_7)]]^T \end{aligned} \tag{3.18}$$

$$= \begin{bmatrix} -\cos(\omega_0 i + x_2) \\ x_1\sin(\omega_0 i + x_2) \\ -\cos(2\omega_0 i + x_4) \\ x_3\sin(2\omega_0 i + x_4) \\ -\cos(3\omega_0 i + x_6) \\ x_5\sin(3\omega_0 i + x_6) \\ -1 \end{bmatrix}^T \tag{3.19}$$

Using a sliding window type algorithm, the Jacobian matrix is calculated for the last N samples of the data. This is typically 3-5 seconds, corresponding to 300-500 samples, which is by experiment the ideal window size (cf. Section 5.4.2). The Gauss-Newton step is calculated at every new sample

[2] The Matlab slash operator computes the system of linear equations by Gaussian elimination, instead of inverting the matrix directly.

m, and the parameter vector is updated $\mathbf{x}_m = \mathbf{x}_{m-1} + \mathbf{h}_{gn,m}$. The m'th fitted sample is then $M(\mathbf{x}_m, m, \omega_0)$.

3.3.8 Estimating the fundamental frequency

Since the harmonic components of respiration are of much higher magnitude than the harmonic components of the heartbeat, it is reasonable to assume that the respiration frequency can be found by calculating the highest absolute value of the frequency spectrum of the input signal.

Several signal processing methods are available for doing this. Morgan and Zierdt [2009] suggest to calculate the average power spectral density using Welch's averaged, modified periodogram method. This method has not yielded satisfactory results on the signals used in this thesis. Thus the fundamental frequency estimation has been implemented by finding a peak in the autocorrelation function, as explained in the next section.

3.3.9 Maximizing the autocorrelation function

In Appendix A it is shown that the first local peak in the autocorrelation function $r_{xx}(n)$, to the right of the zero lag peak[3] occurs at time lag T where the fundamental frequency $f_0 = \frac{1}{T}$. To find the first peak in the autocorrelation function a direct search algorithm was implemented. By using a Hooke & Jeeves (H&J) minimization method from *linear programming*, on the objective function $-r_{xx}(n)$, the nearest maximum from any starting guess can be found. Fig. 3.4 shows a typical autocorrelation function with a red cross at the desired local functional maximizer.

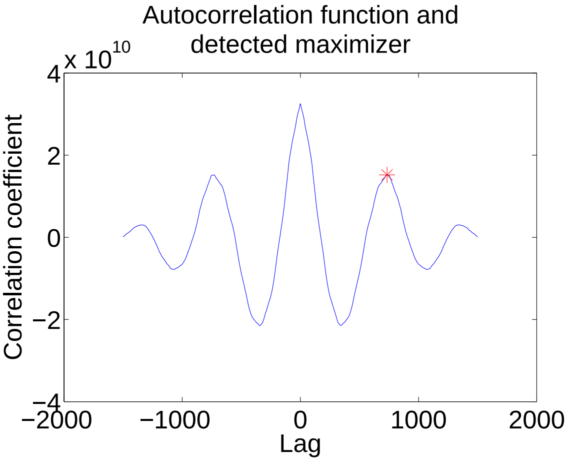


Fig. 3.4: Finding the local maximizer of the autocorrelation function.

[3] The autocorrelation function will always be largest at the zero lag, unless it is scaled.

One-dimensional H&J minimization is done by beginning at a starting guess, looking Δ indexes to the left and right, to see if the function has a lower value there. If it has, then move to that point and repeat. If it has not, then halve Δ and repeat. Repeat until $\Delta \leq \Delta_{min}$. This way there is no need to evaluate the autocorrelation function at every possible time lag.

Algorithm 3.1 One-dimensional Hooke & Jeeves functional minimizer

```
Require: Cost function $f$, $n_0$, $\Delta$, $\Delta_{min}$
  $n := n_0$
  $\hat{n} := \arg\min\{f(n-\Delta), f(n), f(n+\Delta)\};$
  stop := 0
  while !stop do
    if $f(\hat{n}) < f(n)$ then
      $z := \hat{n}$
    else
      $z := n$
      $\Delta := \frac{\Delta}{2}$
    end if
    $\hat{n} := \arg\min\{f(z-\Delta), f(z), f(z+\Delta)\};$
    if $\Delta \leq \Delta_{min}$ then
      stop := 1
    end if
  end while
```

In devising a method for finding the first peak to the right of the zero-lag peak, the following algorithm has yielded robust results. The H&J minimization function is used as a tool to locate the nearest local peak from any starting point. First the nearest peak to an expected starting guesss is found. Now there are two possible scenarios:

- Guess is too close to zero, H&J minimization finds the *zero lag peak*. Solution: Double the guess.
- Guess is too far away from desired peak. In this case H&J iterates to the *second* peak in the autocorrelation function. Solution: Halve the guess.

To find the desired peak, algorithm ALGORITHM 3.2 is used. It runs through H&J minimizer starting in the given starting point.

Initially it runs through the H&J minimizer and check for a zero lag hit. If it occurs it doubles the starting guess. If it occurs again it triples the starting guess, and so on. Once it hits something that is not at the zero lag, it can be assumed that this is the peak that was searched for. However, in the case that the zero lag was never found, it is necessary to check that there is no other minimizer *between* the current minimizer and the zero lag.

Algorithm 3.2 Autocorrelation peak detection

```
stop := 0
k := 1
startlag := 300                                   // First starting guess
while !stop do
   maxcorr, lag := hookejeeves( @r, k * startlag, 20, 1 )
   if lag > 20 && (maxcorr < 1) then
      stop := 1;
   end if
   k := k+1;
end while
stop := 0;
while !stop do
   maxcorr, lagcheck := hookejeeves( @r, lag/2, 20, 1 );
   if (lagcheck < 3) || ((lagcheck > 0.95*lag) && (lagcheck < 1.05*lag)) ||
   (maxcorr > 1) then
      stop := 1;
   else
      lag := lagcheck;
   end if
end while
```

Therefore it checks what H&J outputs when starting at half the current minimizer. If this returns either the current minimizer or the zero lag, the correct peak has been found. If not, there is a minimizer inbetween. In this case, start H&J minimizer at half the lag of the latest peak, and repeat.

This algorithm might seem elaborate, but it has given the best results in the experiments. In this algorithm's defence, it eliminates the need for calculating the autocorrelation function at all possible time lags. Using a local peak detection algorithm would require this. An alternative implementation was local peak detection directly on the time series, but this strategy resulted in higher variance in the frequency estimate and consequently worse performance for the Gauss-Newton algorithm.

3.3.10 Handling the f_0 update

Every time f_0 is updated, the parameters change so much that the algorithm needs some iterations to converge again. Since nonlinear optimization is used, immediate convergence cannot be expected. Therefore the model value is computed from old values of $\mathbf{x}$ and ω_0 a few samples after the update. Meanwhile the algorithm converges to values that are valid for the updated f_0. If this mechanism is not used, the residuals will have ripples at f_0 updating incidents making heartrate estimation more difficult.

3.3.11 Summary of algorithm

The final algorithm is of the sliding window type. With each new sample the Gauss-Newton step $\mathbf{h}_{gn}$ is solved for N samples from the past, and $\mathbf{x}$ is updated. Then the model estimate $M(\mathbf{x}, i, \omega_0)$ and residual $r_i = y_i - M(\mathbf{x}, i, \omega_0)$ are calculated. Then the residual is stored in an ouput vector containing the heart signal. f_0 is updated periodically. When updating f_0 the optimal $\mathbf{x}$ will change quite much, and therefore a mechanism that uses the old $\mathbf{x}$ and ω_0 for a given number of samples after an f_0 update is implemented. Cf. ALGORITHM 3.3 for an overview.

Algorithm 3.3 Harmonic cancelling

```
Require: input,$\mathbf{x}_0$,$f_s$, $\omega_{0,initial}$, win
  f0update := 10                                    // Update f0 each 10s
  holdlength := 10                    // Use old $\mathbf{x}$ 10 samples after f0 update
  $\mathbf{x} := \mathbf{x}_0$
  $\omega_0 := \omega_{0,initial}$
  f0update := round( f0update * $f_s$ )
  for $i$ = win to length( input ) do
    if mod( i, f0update ) then
      $\mathbf{i}$ := ($i$-f0update+1):$i$
      $\mathbf{y}$ := input($\mathbf{i}$)
      $old\omega_0 := \omega_0$
      $\mathbf{oldx} := \mathbf{x}$
      $\omega_0$ := updatef0( $\mathbf{y}$, $f_s$ );
      hold := holdlength
    end if
    $\mathbf{i}$ := ($i$-win+1):$i$
    $\mathbf{y}$ := input($\mathbf{i}$)
    $\mathbf{J}(\mathbf{x})_{ij} := \frac{\partial r_i}{\partial x_j}$
    $\mathbf{r} := \mathbf{y} - M(\mathbf{x}, \mathbf{i}, \omega_0)$
    Solve $\mathbf{J}^T\mathbf{J}\mathbf{h}_{gn} = -\mathbf{J}^T\mathbf{r}$
    $\mathbf{x} := \mathbf{x} + \mathbf{h}_{gn}$
    if hold $>$ 0 then
      fit($i$) := $y_i - M(\mathbf{oldx}, i, old\omega_0)$
      hold := hold - 1
    else
      fit($i$) := $y_i - M(\mathbf{x}, i, \omega_0)$
    end if
  end for
```

3.4 Heart rate estimation using autocorrelation

3.4.1 Introduction

This method is based on calculating the autocorrelation function of a time window of the estimated heart signal. The method is proven useful on a digital signal processor for doppler radar sensing of vital signs in Lohman et al. [2001], and it is also the method that is currently used by MicroImpulse.

3.4.2 Windowing

For heart rate measurements, frequency estimation of a time window of suitable length is conducted. A too long time window reduces the time resolution of the estimated heart rate, because the average rate within that window is found. A too short window decreases the frequency resolution of the heart rate. This is the limitation that most time-frequency distribution face. In fact, time-frequency distributions that have high resolution in both time and frequency do not exist. The curious reader is encouraged to check Schniter [2005] for an introduction of the *time-frequency uncertainty principle*, which provides the theoretical background for the last statement.

A sliding window algorithm is implemented in this thesis. By experiment it was found that at least 7 seconds long windows were required for accurate rate calulations. This is the tradeoff that was hinted by the time-frequency uncertainty principle. High resolution of the heart rate estimate comes at the expense of time resolution. The window will slide one sample each iteration.

Further, Lohman et al. [2001] argues that applying a Hanning window (cf. Fig. 3.5) will prevent spectral leakage, and indeed this has proven necessary to obtain accurate results.

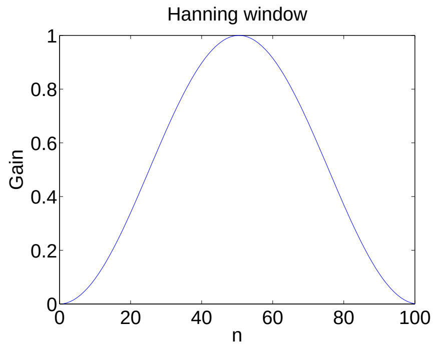


Fig. 3.5: The Hanning window

3.4.3 Spectral esimation

The spectral esimation on this autocorrelation function is the exact same problem as the spectral estimation on the respiration signal, described in Section 3.3.9. Therefore it is solved in the same way, except that the starting guess is shifted down to a period of 1 second.

3.5 Heart rate estimation using adaptive model estimation

3.5.1 Introduction

Methods such as the autocorrelation method described in the previous section suffer from low time resolution. The heart rate estimate at any time is the average heart rate over the time period of the window function. Byrne et al. [1986] therefore suggests that the heart measurements can be modeled as an all-pole filter excited by a train of unit pulses $v(n)$ and white noise $w(n)$, where $g(n) = v(n) + w(n)$;

$$x(n) = a_1 x(n-1) + a_2 x(n-2) + \cdots + a_p x(n-p) + g(n) \tag{3.20}$$

In the model each heartbeat in $x(n)$ is initiated by the impulse. Varying amplitudes of the heartbeat corresponds to varying amplitudes of the impulse. This implies that the impulse response of the a_n coefficients corresponds to one heartbeat.

If the a coefficients were known, one could deconvolve the measured time series $x(n)$ with the all-pole filter, and get a train of impulses. At the arrival of each impulse, the start of a new heartbeat could be assumed. By measuring the time between each impulse, an estimate of the instant pulse frequency could be be obtained.

3.5.2 Adaptive filtering

Usually there is no *a priori* knowledge of the coefficients of the all-pole filter. By using least squares estimation, the coefficients can be estimated by comparing the estimate the predicted $\hat{x}(n)$ to the measured $x(n)$, where

$$\hat{x}(n) = a_1 x(n-1) + a_2 x(n-2) + \cdots + a_p x(n-p) + g(n) \tag{3.21}$$

and the prediction error $\varepsilon(n) = \hat{x}(n) - x(n) = g(n)$. Adaptive filtering techniques can be used to find the parameters a_n that minimize the prediction error. Three online parameter estimation methods were used in Byrne et al. [1986]; a first order predictor, recursive least squares (RLS) and normalized least squares lattice. In this thesis the system is implemented with the RLS filter.

3.5.3 Recursive least squares filtering

A well known class of adaptive filters is the recursive least squares (RLS) filter. For a p'th order filter, consider the regressor vector

$$\mathbf{x}_n = \begin{bmatrix} x(n-1) & x(n-2) & \cdots & x(n-p) \end{bmatrix}^T \tag{3.22}$$

and the parameter vector

$$\mathbf{a}_n = \begin{bmatrix} a_1 & a_2 & \cdots & a_p \end{bmatrix}^T \tag{3.23}$$

Now the model can be written

$$x(n) = \mathbf{x}_n^T \mathbf{a} + g(n) \tag{3.24}$$

At time n, the $\hat{\mathbf{a}}$ that minimizes the sum of squared prediction errors is estimated

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{a}} S_n(\mathbf{a}) \tag{3.25}$$

where

$$S_n(\mathbf{a}) = \sum_{k=p+1}^{n} (x(k) - \mathbf{x}_k^T \mathbf{a})^2 \tag{3.26}$$

In Appendix B it is shown that the optimal $\hat{\mathbf{a}}$ is found recursively by updating a gain matrix $\mathbf{K}$ and the inverse of the autocorrelation matrix $\mathbf{P}$ such that

$$\mathbf{P}_n = \mathbf{P}_{n-1} - \frac{\mathbf{P}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{P}_{n-1}}{1 + \mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n} \tag{3.27}$$

$$\mathbf{K}_n = \frac{\mathbf{P}_{n-1}\mathbf{x}_n}{1 + \mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n} \tag{3.28}$$

and using the error

$$\varepsilon_n(\hat{\mathbf{a}}_{n-1}) = x(n) - \mathbf{x}_n^T \hat{\mathbf{a}}_{n-1} \tag{3.29}$$

we get the updating formula for $\hat{\mathbf{a}}$

$$\hat{\mathbf{a}}_n = \hat{\mathbf{a}}_{n-1} + \mathbf{K}_n \varepsilon_n \left(\hat{\mathbf{a}}_{n-1}\right) \tag{3.30}$$

A complete description is found in Appendix B, confer also to any literature on time series analysis or adaptive filtering, e.g. Madsen [2007] or Haykin [2002].

3.5.4 Defining a heartbeat detection variable

Once the optimal coefficients $\hat{\mathbf{a}}$ are found, the prediction error is used to determine if the model was excited by a random variable or by a unit pulse. This is true because the error is equal to the model excitation (cf. equation (3.21)).

As long as there is no impulse on the input, the error $\varepsilon(n)$ is expected to have the same magnitude as the (assumed) white noise on the input. Therefore the error signal itself can serve as a detection variable.

However, the detection variable can be improved by introducing a variable f that checks if the present regressor $\mathbf{x}$ behaves according to its expectation

$$f(n) = \mathbf{x}_n^T \mathbf{P}_{n-1} \mathbf{x}_n \tag{3.31}$$

This variable is already calculated as a byproduct of the RLS algorithm, cf. equation (3.27). Generally $f(n)$ is zero if the regressor vector behaves according to its expectation, and gives a positive fluctuation when it does not.

A log-likelihood variable $De(n)$ and a variable $D(n)$ of proportional magnitude can be defined such that

$$\begin{aligned} De(n) &= \log\left(\frac{f(n)}{f(n-1)}\right) && (3.32) \\ &= \log(f(n)) - \log(f(n-1)) && (3.33) \end{aligned}$$

which has a magnitude proportional to

$$D(n) = f(n) - f(n-1) \tag{3.34}$$

Finally the heartbeat detector variable $d(n)$ is defined

$$d(n) = D(n)\varepsilon(n) \tag{3.35}$$

This variable is then expected to be close to zero at all times, except for a few samples around the start of a heartbeat. In this manner the beat-for-beat variations can be observed online and in real-time.

3.5.5 Calculating the heart rate

For calculating the heart rate, the interdistance between the pulses needs to be estimated. By continuously comparing the detector variable squared by the expected detector variable squared (that is, the variance of the detector variable), a decision is made whether the detector variable is large enough for a heartbeat to have occured. The variance of the heartbeat detector is tracked by using a forgetting factor $\lambda \in [0, 1]$ such that a variable u proportional to the variance is updated thus

$$
\begin{aligned}
u_n &= y_n^2 + \lambda u_{n-1} && (3.36)\\
&= y_n^2 + \lambda(y_{n-1}^2 + \lambda u_{n-2}) && (3.37)\\
&= \sum_{k=0}^{n} \lambda^k y_{n-k}^2 && (3.38)
\end{aligned}
$$

Since variance is defined as $\frac{\sum_{k=0}^{n-1} y_k^2}{n}$, u must be divided by the sum of its weights to find the variance s. The sum of the weights is

$$\sum_{k=0}^{\infty} \lambda^k = \frac{1}{1-\lambda} \qquad (3.39)$$

which is valid for $|\lambda| < 1$ (Weisstein [2009]). Therefore the variance s is updated by dividing each y by the sum of the weights, that is, multiplying by $1 - \lambda$.

$$s_n = (1-\lambda)y_n^2 + \lambda s_{n-1} \qquad (3.40)$$

Now that the updating formula for the variance is established, one can simply define detection by querying if the heartbeat detector squared is significantly larger than the variance. If it is, then the start of a heartbeat is assumed. Since the detector variable is expected to fluctuate more after such a detection, a few samples are simply skipped before starting the search for a new large fluctuation of the detector variable. The interdistance between detected impulses is then used to calculate the heart rate.

3.5.6 Summary of RLS heartbeat detector

Given the time series $x(n)$ containing the heartbeat signal extracted from the radar range data, a method to transform this time series into a heartbeat detector variable was explained. The detector variable can be used to detect each heartbeat, and contrary to the autocorrelation method this yields an instant estimate of the heartrate.

The method models the time series as an all-pole filter whose coefficients are estimated by the RLS algorithm for adaptive filtering. The all-pole filter is assumed to be excited by white noise and a train of impulses. The impulses are detected by using statistics describing how likely it is that the last model excitation was from white noise, compared to how likely it is that the last model excitation was an impulse of greater magnitude than the variance of the input.

CHAPTER 4

ACQUISITION OF RADAR DATA AND CONSTRUCTED TEST SIGNALS

4.1 Introduction

Two sources of signals have been tested in this thesis. On one hand a test signal was constructed, so that the theoretical qualities of signal processing methods can be assessed. On the other hand, some actual radar data provided by MicroImpulse AS were used. This chapter describes how the data were produced.

4.2 Test signal

The test signal used is equal to that used in Morgan and Zierdt [2009]. The respiration and heartbeat components of this signal are described in the following sections.

4.2.1 Heartbeat component

As mentioned earlier in this thesis, a pure sinusoid does not really resemble the shape of the heartbeat. Morgan and Zierdt [2009] used a test signal that looks more like the heartbeats observed in real radar data. The heartbeat component is modeled as a train of decaying exponentials filtered through a second order Butterworth filter with corner frequency ω_0 and decay time constant τ. It is shown in Appendix C of Morgan and Zierdt [2009] that a formula for this pulse shape is

$$x_H(t) = e^{\frac{-t}{\tau}} + \left[\left(\frac{\sqrt{2}}{\omega_0 \tau} - 1 \right) \sin\left(\frac{\omega_0 t}{\sqrt{2}} \right) - \cos\left(\frac{\omega_0 t}{\sqrt{2}} \right) \right] e^{\frac{-\omega_0 t}{\sqrt{2}}} \tag{4.1}$$

An example of this waveform sampled at $f_s = 100$ Hz is depicted in Fig. 4.1.

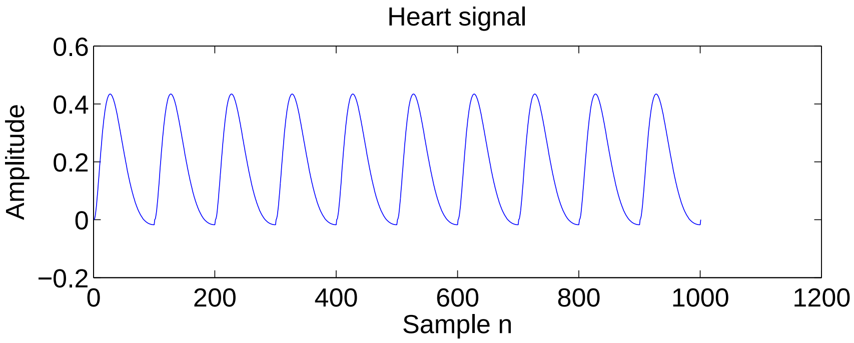


Fig. 4.1: Heart component of test signal.

4.2.2 Respiration component

A sine wave raised to the third power was used to mimic a respiration signal component. The formula is thus

$$x_R(t) = \sin^3(\pi f_R t) \tag{4.2}$$

Such a waveform is depicted in Fig. 4.2.

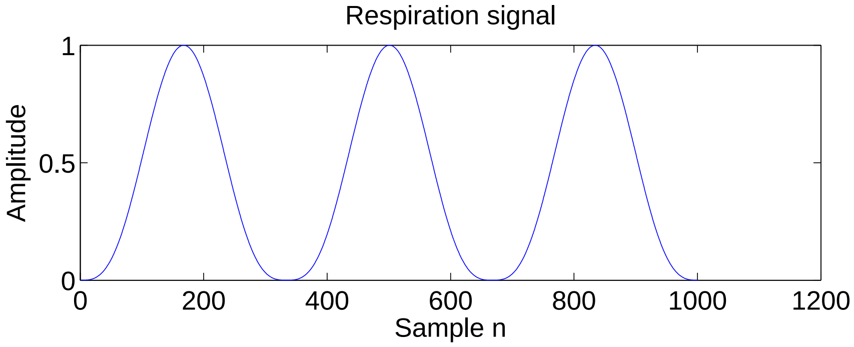


Fig. 4.2: Respiration component of test signal.

The difference in amplitude of the heart and respiration components vary significantly from subject to subject and from situation to situation.

Around 10 times difference is a reasonable figure, hence the test signal $x(t) = x_H(t) + 5x_R(t)$ was used (since the peak heartbeat value is $x_{H,peak} \simeq 0.43$). The combined waveform is depicted in FIG. 4.3, and both are sampled at $f_s = 100$ Hz.

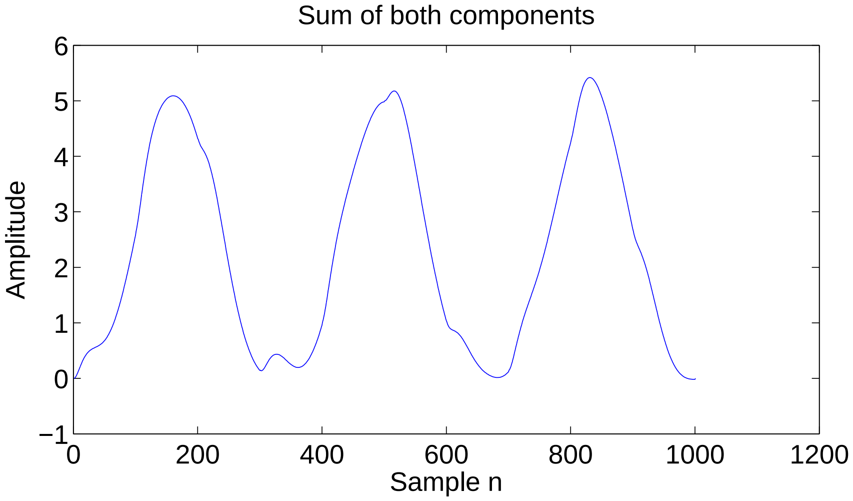


FIG. 4.3: Combined test signal

4.3 Radar signals

The MicroImpulse AS proprietary software UltraScope provides a textfile with consecutive radar range data. The data in this thesis were produced by NanoPulse, and were gathered from a 27 year old male.

CHAPTER 5

RESULTS AND OBSERVATIONS OF HEART RATE DETECTION ALGORITHMS

5.1 Introduction

In this chapter the results from the experiments are presented. First the theoretical capabilities of the methods will be analyzed by looking at the test signal, and then results from real radar data is presented.

5.2 Separation of heart and respiration

5.2.1 FIR filtering method

First we present the base case using FIR filtering. This will give an impression of the separation capability of the system as it is implemented today.

FIG. 5.1 shows the separation of the heartbeat signal using a 400 tap FIR filter whose impulse response was windowed with the Kaiser window, $\beta = 3$. As will be shown later, this filter configuration was found to be a good compromise between stopband attenuation and group delay.

The number of coefficients was determined on the background of filtering the test signal and comparing the output to a time delayed version of the heart component of the test signal. The error was measured as the signal-to-noise ratio (SNR)

$$10 \log_{10} \frac{\|\mathbf{y}\|_2^2}{\|\mathbf{y} - \hat{\mathbf{y}}\|_2^2} \tag{5.1}$$

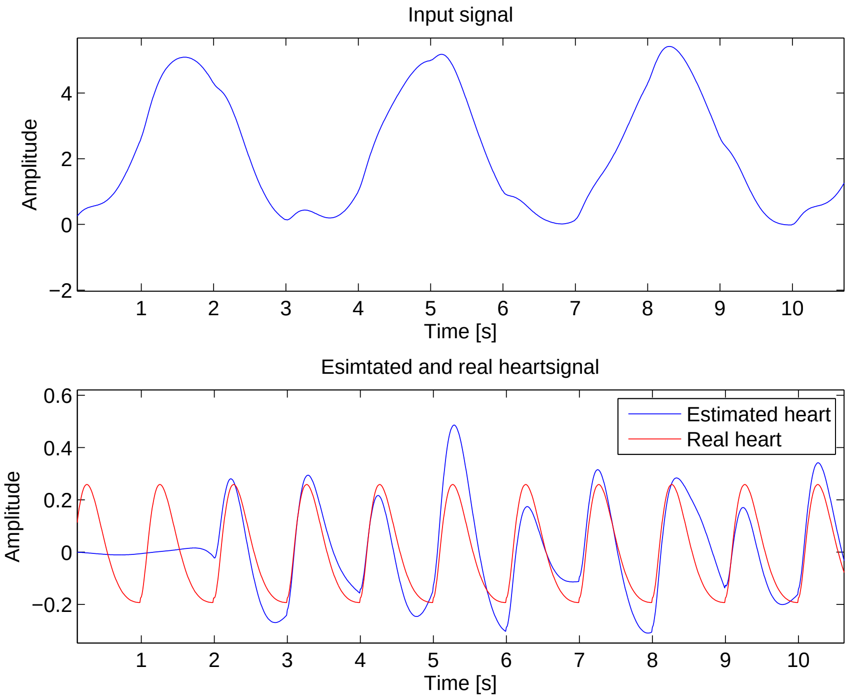


Fig. 5.1: Separation of the test signal using a 400 tap FIR filter, Kaiser windowed with $\beta = 3$. The two second group delay is visible at the start of the signal.

where $\mathbf{y}$ is the heart component of the test signal and $\hat{\mathbf{y}}$ is the filter output. The heart component is delayed with the group delay, which for this type of filter is half the number of filter coefficients. The SNR is plotted versus the number of filter coefficients in Fig. 5.2.

As expected the signal gets better and better with higher filter orders. After around 300 filter coefficients however, the payoff for increasing the number is much lower, and considering the increase in group delay this seems to be a reasonable area to stop.

At a respiration frequency of 35 breaths pr. minute, the filter is not working and cannot even reach a positive SNR (i.e. error power less than signal power). The corner frequency was 0.65 Hz corresponding to 39 breaths pr. minute. Generally this frequency should not be increased because it may affect the lower end of the heartbeat frequency. However, a person breathing at such a frequency is likely to have a faster heartbeat as well, so it could be advisable to adapt the filter's corner frequency to the situation or to the measured respiration frequency.

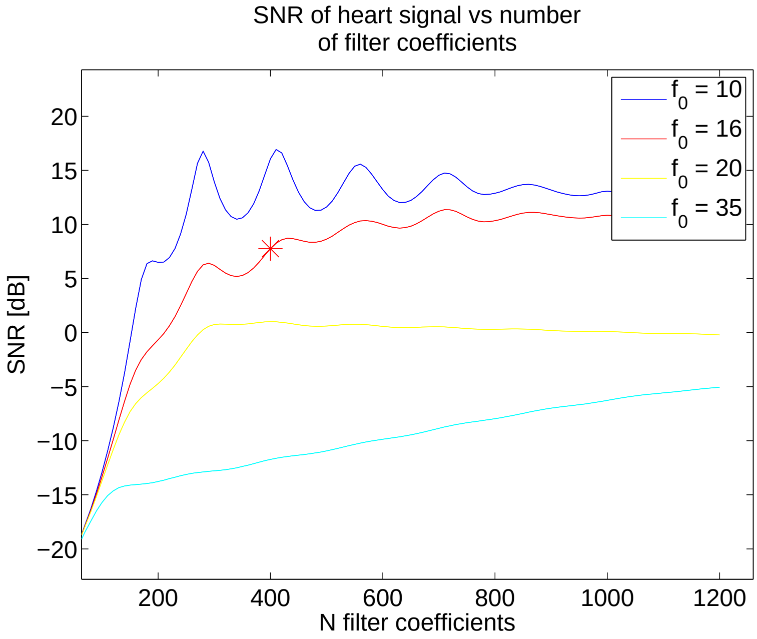


FIG. 5.2: Mean squared error vs. the number of filter coefficients, for different values of the fundamental respiration frequency f_0. After around 300 coefficients the payoff for increasing the number decreases. 400 coefficients were chosen based on this curve.

5.2.2 Harmonic cancelling

The same exercise was done with the harmonic canceller. FIG. 5.3 presents the signal-to-noise ratio for different lengths of the *sliding window* that was described in SECTION 3.3.11.

FIG. 5.3 has a few interesting points:

1. $f_0 = 20$ Hz yields a flat SNR around 0 for any window length.
2. The lower the respiration frequency is, the longer window is needed for good results.

The reason for point 1 is that the heartbeat frequency in this test was 60 bpm. That is *exactly* the third harmonic of the respiration frequency of 20 breaths pr. minute. Hence the harmonic canceller cancels almost everything, so the residual has very low power. Calculating the SNR yields thus

$$\frac{\|\mathbf{y}\|_2^2}{\|\mathbf{y}-\hat{\mathbf{y}}\|_2^2} \simeq \frac{\|\mathbf{y}\|_2^2}{\|\mathbf{y}\|_2^2} = 1 \tag{5.2}$$

since $\hat{\mathbf{y}}$ has been almost completely cancelled.

The reason for point 2 is that a lower respiration frequency yields a respiration cycle that lasts longer in time. To explain this, imagine fitting

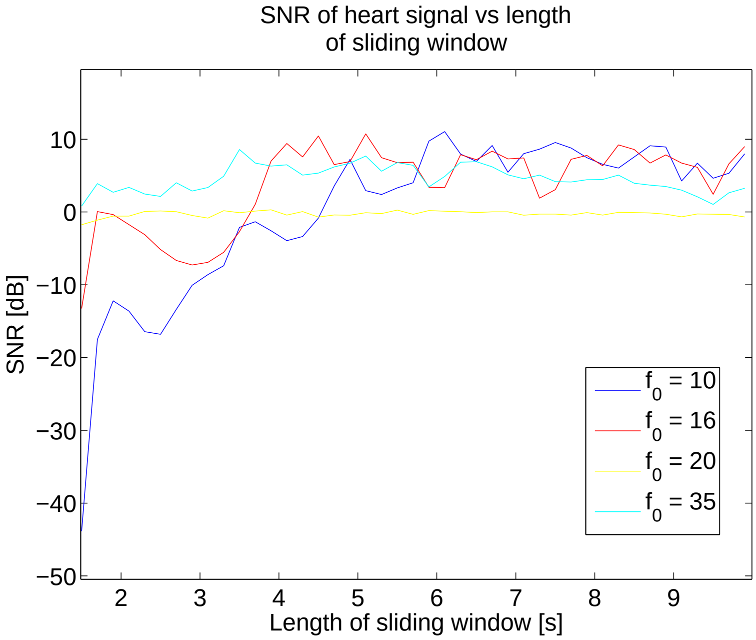


FIG. 5.3: SNR vs. the length of the sliding window in the harmonic canceller, for different values of the fundamental respiration frequency f_0.

some data to a pure sinusoid. If those data do not cover a whole cycle, it is intuitive that this datafitting problem becomes less determined as the data cover less of a whole cycle. Indeed if we only had a small fraction of a cycle, then the columns of the Jacobian matrix would become near linear combinations of one another, yielding a badly conditioned system of linear equations. The problem is not to be confused with having *too few samples* of data, it is the *fraction of the sinusiodal cycle* that dictates how determined the system is.

The reason for dwelling so much with this issue is because when dealing with real signals, the respiration component does not have a stationary behavior. A shorter window allows for a quicker reaction to a change in the system, and generally results in a more robust separation.

At $f_0 = 35$ Hz it is seen that the SNR is decreasing slightly with a longer window size. The reason for this is that the f_0 estimate is not accurate enough. When optimizing over a window of many sinusoidal cycles, a very exact frequency estimate is needed to find parameters that are valid at the end of the window (which is where they are inserted into the model and the residual is calculated). There are many ways to refine the frequency estimate, but real respiration signals are not stationary enough for this to yield any profit.

5.2.3 Filtering versus harmonic cancelling

FIR filtering under the right condtitions tends to give a better signal-to-noise ratio on the heart signal. At up to 20 breaths pr. minute, FIR filtering yields an SNR about 9 dB at a reasonable number of filter coefficients. Using harmonic cancelling, the SNR generally jumps between 5 and 10 dB. As expected, the harmonic canceller works better at higher frequencies than FIR filtering. Filtering has the following advantages over the harmonic canceller:

- Stable and predictable
- Does not cancel the heartbeat signal if the heart beats at a multiple of the respiration frequency
- Easy/cheap to implement

Similarly, the harmonic canceller has the following advantages over FIR filtering:

- No signal delay, allows for a more *online* system
- Works at any respiration frequency, except when the heart beats at a multiple of the respiration frequency

5.3 Heart rate detection

In this section the autocorrelation method for calculating the heartrate is compared to the RLS heartbeat detector. Both methods are presented separately, then the advantages and disadvantages of each method are presented.

5.3.1 Autocorrelation method

FIG. 5.4 shows the mean squared error of the heartrate estimate. A vector containing all heartrate estimates is compared to a vector containing the real heart rates from a generated testsignal. In this case the heartrate was 60 bpm. throughout a vector of 50 seconds. The error (estimated minus real heartrate) is squared and divided by the total number of compared samples.

When using the biased autocorrelation function there is evidence of errors in the spectral estimate. It generally performs worse at all window lengths, and never reaches zero errors. For an accurate spectral estimate, a window of at least 7 seconds is needed when using the unbiased autocorrelation function.

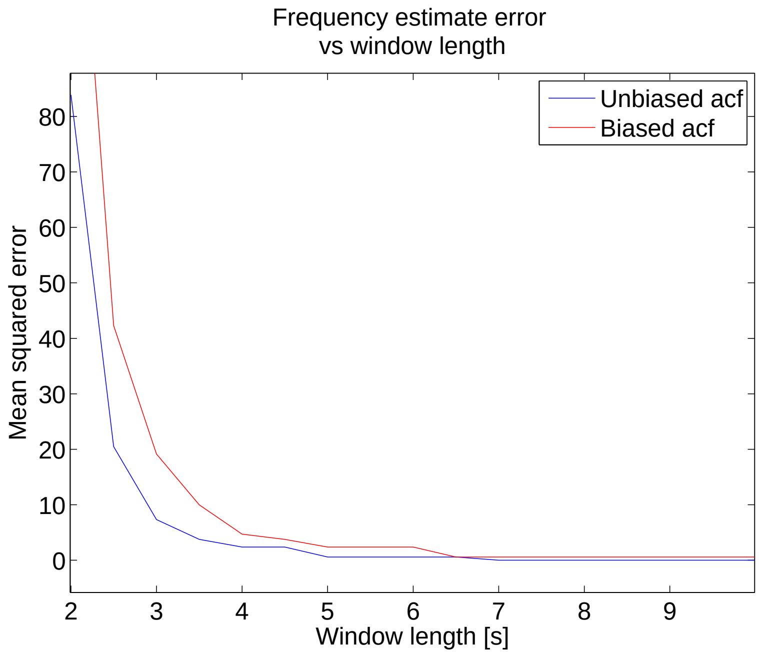


FIG. 5.4: Comparison of heartrate estimates using the biased and unbiased autocorrelation function, for different window lengths.

5.3.2 RLS heartbeat detector

When running the RLS heartbeat detector on the constructed heartbeat signal, the output produced is found in FIG. 5.5.

As expected there is a large deviation when the system changes at the transition between two heartbeats. Tracking of the interdistance between pulses yields the heartrate versus time plot in FIG. 5.6.

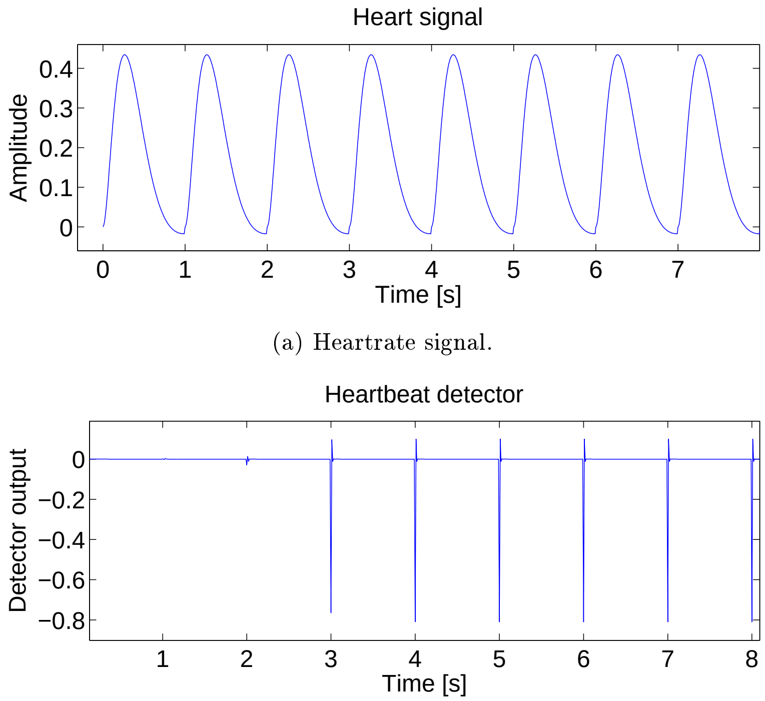
Using this method generates very accurate results, and it is a lot faster to execute than the autocorrelation method.

5.4 System results

5.4.1 FIR filtering

FIG. 5.7 shows system results for FIR filtering. The results are generally pleasing, where both the autocorrelation method and the RLS detector work well. The autocorrelation method produces some ripples and is not as correct as it was when applied to the clean heartbeat component. The errors are however negligibly small.

The RLS heartbeat detector however has some rather large unwanted errors to it. Fixing these errors comes down to finding a better way of deciding if the present sample is the start of a heartbeat. When looking at



(a) Heartrate signal.

(b) RLS heartrate detector.

FIG. 5.5: RLS heartrate detector compared to input signal, 3 filter coefficients and forgetting factor 0.95.

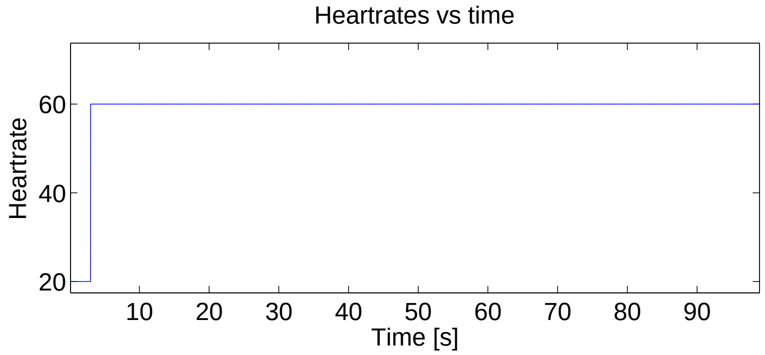


FIG. 5.6: Detected heart rates as a function of time.

the detector output, it should not be too difficult to eliminate these errors, as the heartbeats are easily detected by the eye. As stated earlier this thesis does not concern itself much with this problem, but indeed there is a lot of work that could be done in that field.

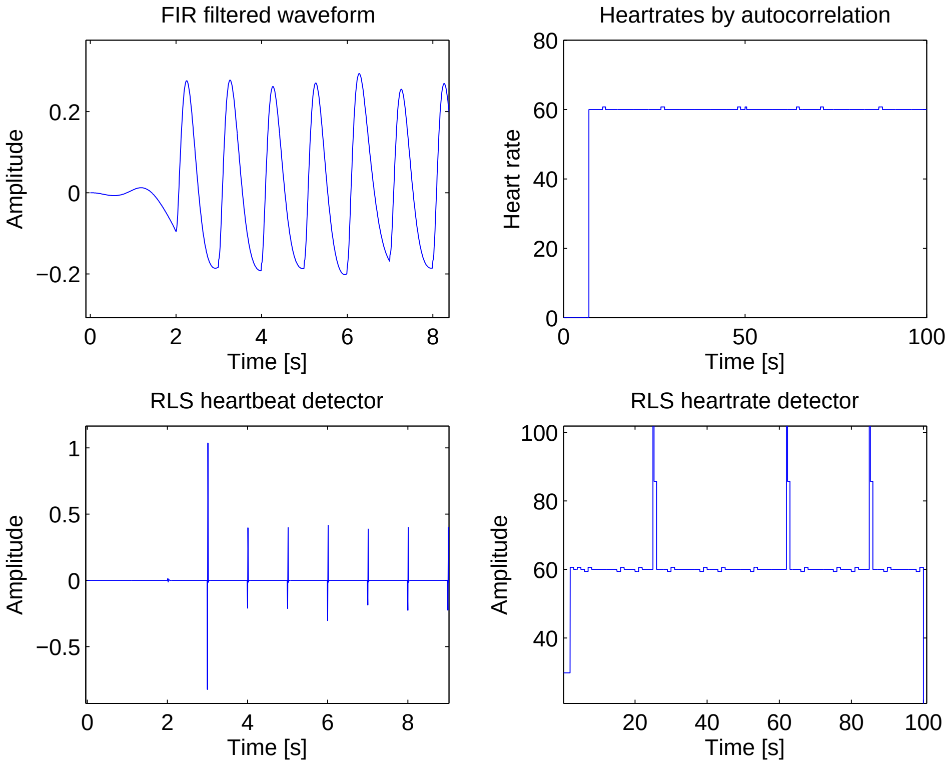


Fig. 5.7: System results using FIR filtering. The filter's group delay of 2 seconds is visible. The heart rate detector does not work during this period. Both heart rate detectors find the correct heart rate of 60 bpm. throughout the time series. The RLS detector struggles with the heartbeat detection logic at times, resulting in large deviations.

5.4.2 Harmonic cancelling

Fig. 5.8 shows system results for the harmonic canceller. In the autocorrelation based heartrate detector, the results are similar to those obtained when using FIR filtering. During the first few seconds however, it yields some large deviations due to the fact that the harmonic canceller waits for a whole window to be obtained before it starts cancelling the harmonics. After the initial convergence period, results are satisfactory.

For the RLS heartrate detector, the results are very good. Indeed it estimates the rate correctly thoughout the whole time series, except for during the convergence period.

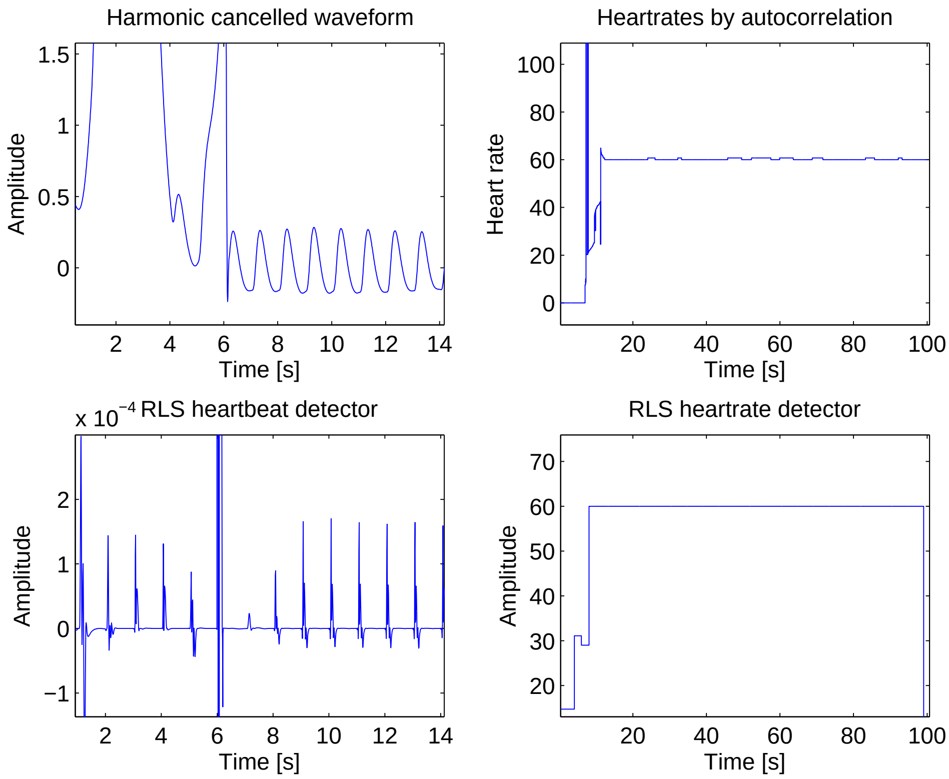


Fig. 5.8: System results using harmonic cancelling. No cancellation is done before the first 6 seconds of data have arrived. Both heartbeat detectors find the correct heart rate. The RLS detector even works before separation is started.

5.5 System results on real signals

This section describes some system results obtained from 50 seconds of test data from `Male_27y.usvs`. An excerpt of these data is found in Fig. 5.9. First the base case using FIR filtering is presented, then the performance of the harmonic canceller is investigated.

5.5.1 FIR filtering

Fig. 5.10 shows the total system performance for separation and heartbeat detection. Both the autocorrelation method and the RLS method are capable of reliably detecting the heart rate. The RLS heart rate detector seems to struggle at times, this is because the algorithm is not robust enough to deal with real signals (in fact, it was barely robust enough to deal with contstructed signals). Those results are included to show that when it works, the rates detected are close to the rates detected using the autocorrelation

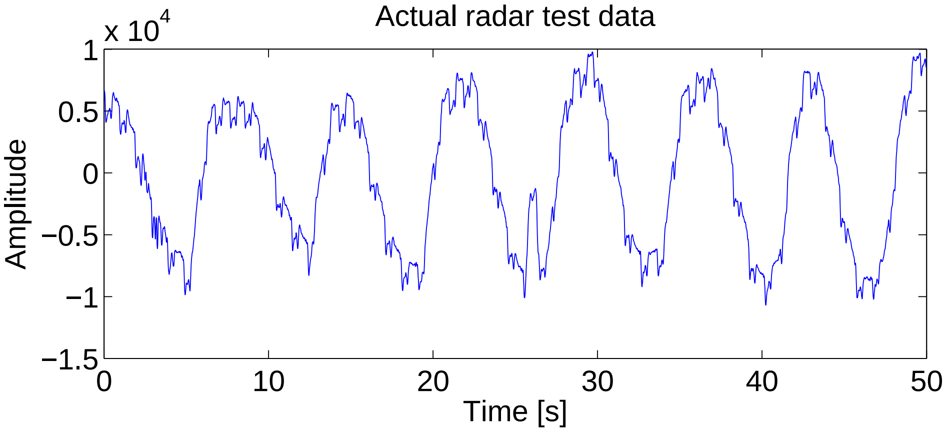


FIG. 5.9: Real radar test data.

method. Since there is no reference ECG signal, this is the only indication that the detected rates are correct. The fact that both methods yield similar results increases the credibility of both methods.

5.5.2 Harmonic cancelling

FIG. 5.11(a) shows the results of the datafit. After about 26 seconds an artifact in the data can be seen. Artifacts of this kind can occur frequently, and thus the system must be able to deal with this. A 6 seconds long sliding window was used in this example. Given that this kind of artifact is not concerned with in the model, the canceller copes quite well. It quickly returns to normal operation after the artifact, and even tries to cancel some of it (although lagging slightly behind).

When looking at the residual after around 26 seconds, there is a distinct and periodic shape that does not stem from the heartbeat. The periodicity is unfortunate, because when using the autocorrelation function the periodicity of the larger component is found, cf. FIG. 5.12. In this case the heartbeat detector actually detects the frequency that was introduced by the artifacts in the data, which is about 30 bpm. Naturally, this is not acceptable.

The RLS heartbeat detector is much less sensitive to these data artifacts. The detector variable at the time of the artifact is found in FIG. 5.13. Certainly the detector variable is a bit “muddled“ at some points, but for the most part heartbeat impulses are clearly indicated and should be possible to detect. However, the rate detection algorithm suggested in this thesis is not sophisticated enough, and this is an area where much more research should be done.

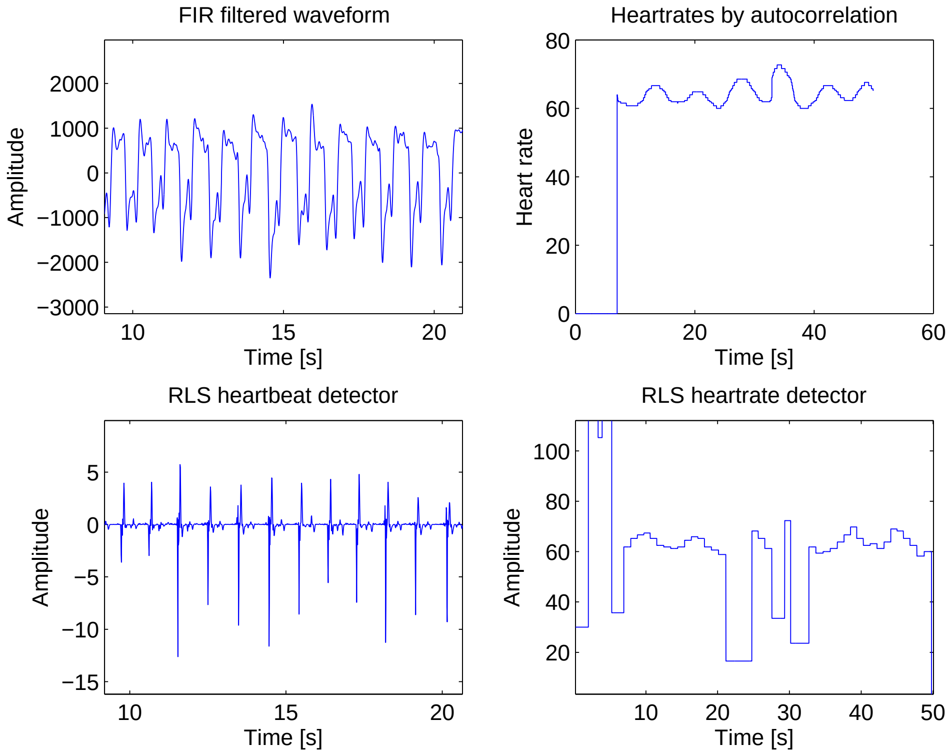
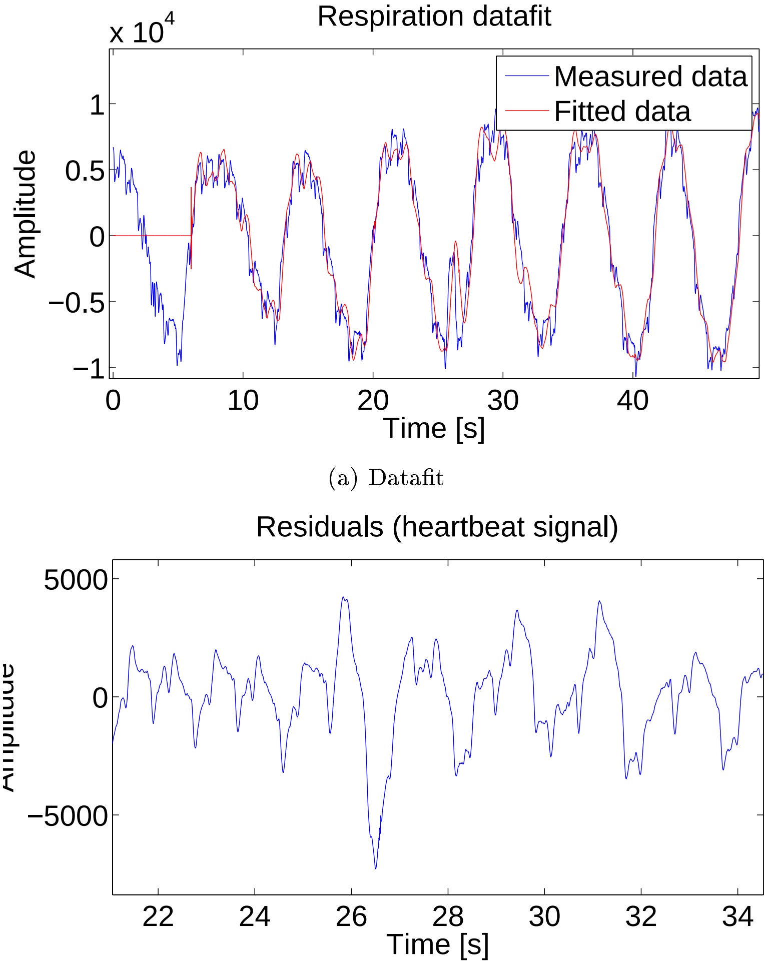


Fig. 5.10: System results using FIR filtering on a real test signal. The autocorrelation method for detecting the heart rates is much more robust than the RLS heart rate detector. However, the RLS heartbeat detector yields sharply defined peaks at the start of each heartbeat.



(a) Datafit

(b) Residuals

Fig. 5.11: Datafit and residuals of 50 seconds of data from Male_27y.usvs. The residuals around the artifact at about 26 seconds are shown. Until that point the heartbeats are clearly visible in the time series. After around 26 seconds however, a new higher amplitude and lower frequency periodic component manifests itself in the residuals. Sliding window was 6 seconds.

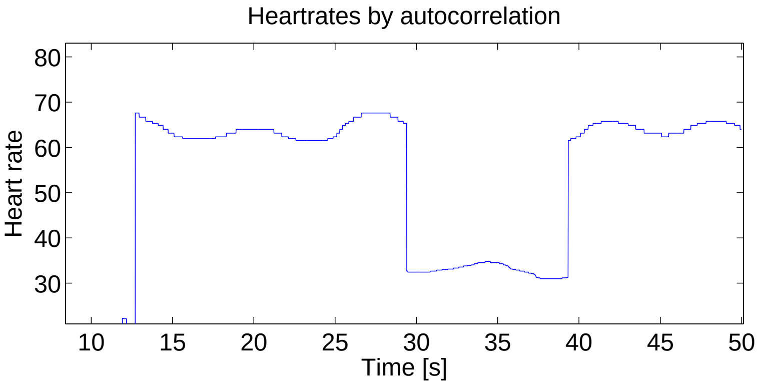


FIG. 5.12: Estimated heartbeat frequency by the autocorrelation method. The autocorrelation function has detected the larger amplitude periodic component as introduced by the harmonic canceller during an artifact in the measurement data, cf FIG. 5.11. Sliding window was 7 seconds.

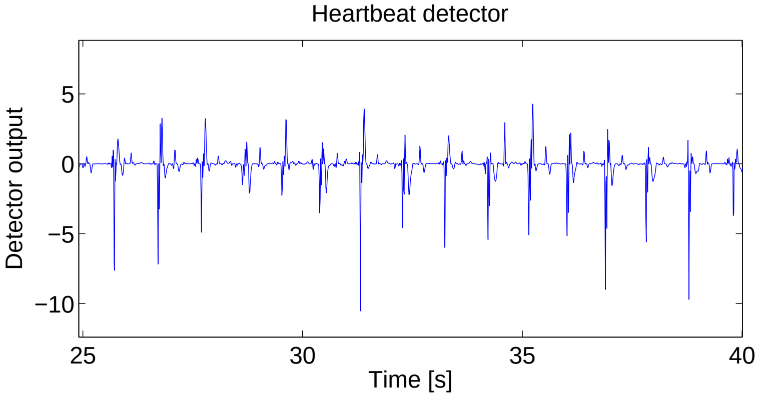


FIG. 5.13: RLS heartbeat detector during the period when the autocorrelation heartbeat detector failed. Here the heartbeats are clearly visible to the eye, and they should be detectable by a robust heartbeat detector.

CHAPTER 6

DISCUSSION

6.1 Heart rate detection results

When drawing conclusions on the results, it is essential to remember that the lack of empirical data prohibits drawing a proper conclusion of what works and what does not. However, the results obtained yield some *indications* of how a new heartbeat detection strategy could work.

6.1.1 FIR filtering vs harmonic cancelling

The observations obtained indicate a better and more stable heart rate/-heartbeat detection for both the autocorrelation method and the RLS method when using FIR filtering. However, the real test signal was gathered from a calm person breathing at about 9 breaths pr. minute. As proven using the constructed test signal, separation deteriorates dramatically when the respiration rate increases to more than 30 breaths pr. minute.

It is this kind of problems the new separation method is trying to overcome. The harmonic canceller will theoretically be able to handle any respiration frequency, except when the heartbeat frequency is an exact multiple of the respiration frequency. Thanks to heart rate variability and the fact that people seldom respire at the exact same frequency for prolonged time periods, this scenario is unlikely to cause a big problem.

In situations optimal for FIR filtering, harmonic cancelling may not perform *as good* as FIR filtering, but maybe *good enough*. To find out, a large set of empirical data with a reference ECG signal is needed. If it is not good enough, perhaps a hybrid method could be devised. That is, when the respiration frequency is too high, switch to harmonic cancelling.

If the filtering delay is not acceptable, harmonic cancelling is as far as the author knows the only alternative.

6.1.2 Combination of methods

It should be noted that harmonic cancelling introduced some periodic artifacts of higher amplitude than the heart signal, when the radar signal deviated from the model. This caused the autocorrelation method to fail, but the RLS method performed better.

This *indicates* that it is better to use RLS heartbeat detection in combination with harmonic cancelling. This fits neatly with the fact that both methods are geared towards an online and instant record of the heart rate. On the other hand, a combination of FIR filtering and autocorrelation heart rate detection should be used in situations where instant heart rate information is not crucial. If a robust RLS heart rate detector was devised (not achieved in this thesis), this could be used with FIR filtering to get the instantaneous heart rate with a 2 second time delay.

6.2 Some thoughts on the optimization solution

This section is devoted to some unsubstantiated, but interesting observations and thoughts from the author. The fact that nonlinear optimization works better for a problem that can be posed as a linear optimization problem is indeed strange, and should be discussed.

A possible reason for this may be that immediate convergence for every sample *is not desired*. The Newton-type standard equation solves the optimization too fast and too correctly. Because of measurement noise and possibly the heartbeat signal, the system solution from sample to sample *differs too much* (for short windows). Hence the model calculation for each sample also differs enough to make the fit jigsawing. This possible explanation fits well with the fact that long windows were needed to make the Morgan and Zierdt [2009] method work well. Using longer windows lets the optimization procedure pay less heed to the heartbeat signal and measurement noise.

Perhaps more likely, it could be a result of the conditioning of the system. The parameter vector is not updated on the basis of the current parameter vector, but calculated "from scratch" from the current data window at every sample. When using short windows, the system could tend towards bad conditioning, and this would cause a numeric solution to be less accurate. Thus the sample-by-sample variations of the parameter vector could be high enough to cause jigsawing on the datafit.

On the contrary, the Gauss-Newton method updates the current parameter vector by a step, hence no recalculation is ever done. This allows for softer steps of the parameter vector, and slower convergence. Thanks to that, there are no sudden jumps in the parameter vector, and the datafit is a smooth curve.

6.3 Directions for further research

6.3.1 RLS heart rate detector

The most pressing issue is to improve the algorithm for deciding whether the current sample is the start of a heartbeat. Given the impulses on the heartbeat detector that are observed by the eye, such an algorithm should not be too hard to devise.

6.3.2 Optimization problem in harmonic canceller

Once the heart rate detector is in place, different optimization strategies should be investigated to see which gives the best results compared to a reference ECG signal.

Among nonlinear optimization strategies, methods such as the Levenberg-Marquardt method and Powell's Dog Leg (Madsen et al. [2004]) methods should be implemented and compared.

6.3.3 Window length in harmonic canceller

An algorithm for adaptively changing the window length should be implemented. For instance, when the patient stops breathing for a second or suddenly changes respiration frequency, the window should be as short as possible to allow for a quick reaction to the system change. During longer periods of constant breathing, the window should be longer to get a "more determined" system of linear equations to solve for.

6.3.4 f_0 update interval

As it was discussed earlier, the f_0 update creates some convergence issues. Therefore, one should really only update f_0 *when necessary.* As an example, an algorithm could track the residual power, and a sudden increase could trigger an f_0 update. As long as the algorithm works fine with the current f_0, there is really no need to change it.

CHAPTER 7

CONCLUSION

A method to make heart rate measurements as robust and instantaneous as possible was sought. In terms of robustness, the harmonic canceller proved to work well regardless of the respiration frequency, whereas the FIR filtering method would stop working at a respiration frequency of more than 30 breaths per minute. The problem of the heartbeat occurring at a frequency multiple of the respiration frequency should be investigated more thoroughly.

The harmonic cancelling algorithm found in the literature was not suited for online implementation. It required too long time windows to work properly, and thus did not have the ability to react to frequently occurring unstationary parts of the respiration signal component. Therefore an alternative method which could handle shorter windows was devised.

In terms of measuring the instantaneous heart rate, the RLS heartbeat detector was able to instantly recognize the start of a heartbeat. Most importantly, the combination of harmonic cancelling and the RLS heartbeat detector was able to yield an online measurement of the heart rate with no time delay and little constraints on the subject's respiration rate.

However, FIR filtering performed better than harmonic cancelling at lower frequencies. It yielded stronger RLS heartbeat detector output, and is preferred over harmonic cancelling if the 2 second time delay is acceptable *and* the respiration frequency is sufficiently low.

Finally, further research must be conducted on the RLS heart rate detection algorithm, as the one used in this thesis was not very robust.

APPENDIX A

AUTOCORRELATION FUNCTION IN FREQUENCY ESTIMATION

A.1 Introduction

The autocorrelation function is often useful in frequency estimation, and it was used both to estimate the respiration and heartbeat frequencies. The following sections explain the concept, and outline how it has been implemented in this thesis. A more detailed explanation on the topic can be found in most books on digital signal processing, time series analysis or statistics, e.g. Proakis and Manolakis [2006] or Madsen [2007].

The autocorrelation function is the "self-correlation" function that estimates how a signal correlates with a *timeshifted version of itself.* Mathematically this is estimated by

$$r_{\mathrm{xx}}(k) = \frac{\sum_{n=1}^{N} x_n x_{n-k}}{N} \tag{A.1}$$

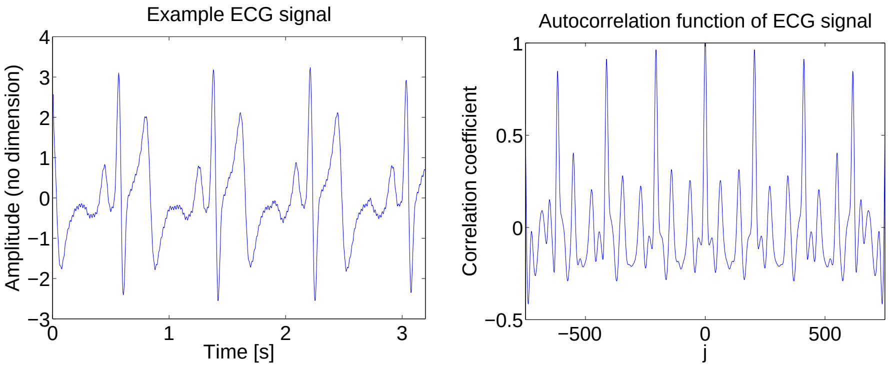
where N is the number of samples.

When applying this to a signal contained in a vector $\mathbf{x}$, define a $[1 \times k]$ vector $\mathbf{k}$ that contains k zeros. Then a pointwise multiplication of $[\mathbf{x} \quad \mathbf{k}]$ and $[\mathbf{k} \quad \mathbf{x}]$ is conducted to obtain the autocorrelation function. This essentially means that the larger values of k, the fewer samples are multiplied with nonzero values. This is the inherent *bias* when estimating the autocorrelation function. An amendment to this is to use the *unbiased* autocorrelation function, that *scales* the correlation coefficient by the absolute value of k:

$$r_{\mathrm{xx}}(k) = \frac{\sum_{n=1}^{N} x_n x_{n-k}}{N - |k|} \tag{A.2}$$

A.2 Frequency estimation

It is well known that the autocorrelation function of a periodic signal also is a periodic signal with the same frequency. This is because a periodic signal contains *repeating patterns*. Thus, the the correlation between a signal and a timeshifted version of itself is highest at the timeshift corresponding to one period. FIG. A.1(b) shows the autocorrelation function for a few seconds of an ECG signal, which has a distinct and repeating shape that is ideal for detecting the correlation.



(a) Example ECG signal

(b) Estimated autocorrelation function

FIG. A.1: Example ECG signal with estimated autocorrelation function

For actually estimating the frequency, one can find the first maximizer to the right of the zero-lag peak. That time lag corresponds to one period of the signal. The frequency $f = \frac{1}{T}$ where T is the period can be found in a discrete system samled at f_s by

$$f = \frac{1}{\arg\max_n r_{xx}(n) \cdot f_s} \tag{A.3}$$

where $\arg\max_n r_{xx}(n)$ is the *local maximizer* around the first peak of the autocorrelation function.

APPENDIX B

RECURSIVE LEAST SQUARES FILTER

The Recursive Least Squares (RLS) filter is described in most literature on time series analysis or adaptive filtering, e.g. Haykin [2002] or Madsen [2007].

B.1 Developing the recursive least squares filter

The RLS filter is a recursive (on-line) parameter estimation technique. Consider the regressor vector

$$\mathbf{x}_n = \begin{bmatrix} x(n-1) & x(n-2) & \cdots & x(n-p) \end{bmatrix}^T \tag{B.1}$$

and the parameter vector

$$\mathbf{a}_n = \begin{bmatrix} a_1 & a_2 & \cdots & a_p \end{bmatrix}^T \tag{B.2}$$

Now the model can be written as

$$x(n) = \mathbf{x}_n^T \mathbf{a} + g(n) \tag{B.3}$$

where $g(n)$ is gaussian noise.

At time n, an optimal $\hat{\mathbf{a}}$ that minimizes the sum of squared prediction errors is sought, formally

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{a}} S_n(\mathbf{a}) \tag{B.4}$$

where

$$S_n(\mathbf{a}) = \sum_{k=p+1}^{n} (x(k) - \mathbf{x}_k^T \mathbf{a})^2 \tag{B.5}$$

The solution of equation (B.4) is found by writing the model (B.3) for all samples in the range $[p+1, n]$, where n is the present sample of the time series. Introduce the variables

$$\begin{aligned}
\mathbf{y} &= \begin{bmatrix} x(p+1) & x(p+2) & \cdots & x(n) \end{bmatrix}^T \\
\mathbf{X} &= \begin{bmatrix} \mathbf{x}_{p+1} & \mathbf{x}_{p+2} & \cdots & \mathbf{x}_n \end{bmatrix}^T \\
\mathbf{g} &= \begin{bmatrix} g(p+1) & g(p+2) & \cdots & g(n) \end{bmatrix}^T
\end{aligned}$$

The compact form of the model (B.3) for all observations can now be written as

$$\mathbf{y} = \mathbf{X}\mathbf{a} + \mathbf{g} \tag{B.6}$$

The sum of squared errors at the last sample in the time series (cf. equation (B.5)):

$$S_n(\mathbf{a}) = (\mathbf{y} - \mathbf{X}\mathbf{a})^T (\mathbf{y} - \mathbf{X}\mathbf{a}) \tag{B.7}$$

To find the $\hat{\mathbf{a}}$ that minimizes $S_n(\mathbf{a})$, the point where its derivative equals 0 can be found:

$$S_n(\hat{\mathbf{a}}) = (\mathbf{y} - \mathbf{X}\hat{\mathbf{a}})^T (\mathbf{y} - \mathbf{X}\hat{\mathbf{a}}) \tag{B.8}$$

$$\nabla_{\mathbf{a}} S_n(\hat{\mathbf{a}}) = -2\mathbf{X}^T (\mathbf{y} - \mathbf{X}\hat{\mathbf{a}}) = 0 \tag{B.9}$$

$$\Rightarrow (\mathbf{X}^T\mathbf{X})\hat{\mathbf{a}} = \mathbf{X}^T\mathbf{y} \tag{B.10}$$

and consequently

$$\hat{\mathbf{a}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{B.11}$$

Let us call $\mathbf{R}_n = \mathbf{X}^T\mathbf{X}$ and $\mathbf{h}_n = \mathbf{X}^T\mathbf{y}$. Then

$$\begin{aligned}
\mathbf{R}_n &= \mathbf{X}^T\mathbf{X} \\
&= \begin{bmatrix} \mathbf{x}_{p+1} & \mathbf{x}_{p+2} & \cdots & \mathbf{x}_n \end{bmatrix} \begin{bmatrix} \mathbf{x}_{p+1}^T \\ \mathbf{x}_{p+2}^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix} \\
&= \sum_{s=p+1}^{n} \mathbf{x}_s \mathbf{x}_s^T
\end{aligned}$$

Recursive updating of $\mathbf{R}_n$ is thus

$$\mathbf{R}_n = \mathbf{R}_{n-1} + \mathbf{x}_n\mathbf{x}_n^T \tag{B.12}$$

In the same manner,

$$\begin{aligned}
\mathbf{h}_n &= \mathbf{X}^T\mathbf{y} \\
&= \begin{bmatrix}\mathbf{x}_{p+1} & \mathbf{x}_{p+2} & \cdots & \mathbf{x}_n\end{bmatrix} \begin{bmatrix} x(p+1) \\ x(p+2) \\ \vdots \\ x(n) \end{bmatrix} \\
&= \sum_{s=p+1}^{n} \mathbf{x}_s x(s)
\end{aligned}$$

and $\mathbf{h}_n$ can be updated recursively;

$$\mathbf{h}_n = \mathbf{h}_{n-1} + \mathbf{x}_n x(n) \tag{B.13}$$

With these definitions, equation (B.11) can be written as

$$\hat{\mathbf{a}} = \mathbf{R}_n^{-1}\mathbf{h}_n = \mathbf{R}_n^{-1}\left[\mathbf{h}_{n-1} + \mathbf{x}_n x(n)\right]$$

According to equation (B.10)

$$\mathbf{h}_{n-1} = \mathbf{R}_{n-1}\hat{\mathbf{a}}_{n-1} \tag{B.14}$$

and

$$\begin{aligned}
\hat{\mathbf{a}}_n &= \mathbf{R}_n^{-1}\left[\mathbf{R}_{n-1}\hat{\mathbf{a}}_{n-1} + \mathbf{x}_n x(n)\right] \\
&= \mathbf{R}_n^{-1}\left[\mathbf{R}_n\hat{\mathbf{a}}_{n-1} - \mathbf{x}_n\mathbf{x}_n^T\hat{\mathbf{a}}_{n-1} + \mathbf{x}_n x(n)\right] \\
&= \hat{\mathbf{a}}_{n-1} + \mathbf{R}_n^{-1}\mathbf{x}_n\left[x(n) - \mathbf{x}_n^T\hat{\mathbf{a}}_{n-1}\right]
\end{aligned}$$

The RLS algorithm for recursive estimation of filter coeffients is thus developed:

$$\hat{\mathbf{a}}_n = \hat{\mathbf{a}}_{n-1} + \mathbf{R}_n^{-1}\mathbf{x}_n\left[x(n) - \mathbf{x}_n^T\hat{\mathbf{a}}_{n-1}\right] \tag{B.15}$$

$$\mathbf{R}_n = \mathbf{R}_{n-1} + \mathbf{x}_n\mathbf{x}_n^T \tag{B.16}$$

The matrix inversion in this formulation of the algorithm prevents it from being implemented efficiently. The next section will discuss some improvements that will allow a faster execution.

B.2 Efficient implementation of the RLS algorithm

To avoid inversion of $\mathbf{R}_n$ in each step, introduce

$$\mathbf{P}_n = \mathbf{R}_n^{-1} \tag{B.17}$$

A special case of the *matrix inversion lemma* known as the *Sherman-Morrison formula* Bartlett [1951] states that

$$\left[\mathbf{A} + \mathbf{u}\mathbf{v}^T\right]^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{u}\left[1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u}^T\right]^{-1}\mathbf{v}^T\mathbf{A}^{-1} \tag{B.18}$$

and using $\mathbf{A} = \mathbf{R}_{t-1}$ and $\mathbf{u} = \mathbf{v} = \mathbf{x}_n$, obtain

$$\mathbf{P}_n = \mathbf{P}_{n-1} - \frac{\mathbf{P}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{P}_{n-1}}{1 + \mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n} \tag{B.19}$$

Further, introduce

$$\begin{aligned}
\mathbf{K}_n &= \mathbf{P}_n\mathbf{x}_n = \mathbf{P}_{n-1}\mathbf{x}_n - \frac{\mathbf{P}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n}{1 + \mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n} \\
&= \frac{\mathbf{P}_{n-1}\mathbf{x}_n\left[1 + \mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n\right] - \left[\mathbf{P}_{n-1}\mathbf{x}_n\mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n\right]}{1 + \mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n} \\
&= \frac{\mathbf{P}_{n-1}\mathbf{x}_n}{1 + \mathbf{x}_n^T\mathbf{P}_{n-1}\mathbf{x}_n}
\end{aligned}$$

and then equation (B.15) can be written as

$$\hat{\mathbf{a}}_n = \hat{\mathbf{a}}_{n-1} + \mathbf{K}_n\left[x(n) - \mathbf{x}_n^T\hat{\mathbf{a}}_{n-1}\right] \tag{B.20}$$

Finally, introduc the prediction error

$$\varepsilon_n(\hat{\mathbf{a}}_{n-1}) = x(n) - n^T\hat{\mathbf{a}}_{n-1} \tag{B.21}$$

and equation (B.15) is written compactly

$$\hat{\mathbf{a}}_n = \hat{\mathbf{a}}_{n-1} + \mathbf{K}_n\varepsilon_n\left(\hat{\mathbf{a}}_{n-1}\right) \tag{B.22}$$

APPENDIX C

GAUSS-NEWTON METHOD

The Gauss-Newton method seeks a parameter vector $\mathbf{x}_{min}$ that minimizes the sum of squared residuals

$$r_i(\mathbf{x}) = y_i - M(\mathbf{x}, t_i, params) \tag{C.1}$$

where each y_i is gathered from some dataset and M is an approximate model of the behaviour of y_i at time t_i with the variable parameters contained in $\mathbf{x}$ and a constant set of parameters contained in *params*.

The cost function $S(\mathbf{x})$ is the sum of squared residuals

$$S(\mathbf{x}) = \frac{1}{2}\|\mathbf{r}(\mathbf{x})\|_2^2 = \frac{1}{2}\sum_{i=1}^{N}(r_i(\mathbf{x}))^2 \tag{C.2}$$

scaled by 0.5 for convenience.

The Gauss-Newton method is the basis for a class of methods for solving nonlinear systems. For example it forms the basis for Levenberg-Marquardt method and the Powell's Dog Leg methods. The Gauss-Newton method is the simplest of these methods and it has provided promising results on the data in this study.

C.1 Developing the Gauss-Newton method

The following theorem provides approximations to the value of r_i in the neighborhood of $\mathbf{x}$, defined by $\mathbf{x} + \mathbf{h}$.

Theorem C.1.1 (1st order Taylor expansion)**.** *If $f(\mathbf{x})$ has continuous partial derivatives of second order, then,*

$$f(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + \mathbf{h}^T \nabla f(\mathbf{x}) + O(\|\mathbf{h}\|_2^2) \tag{C.3}$$

The Taylor series expansion will contain an infinite number of terms, but to develop the Gauss-Newton method one needs to stop at the first term and accept that there will be an approximation error of magnitude dictated by $\mathbf{h}$. The Big-O notation means that the approximation error has the same *order of magnitude* as $\|\mathbf{h}\|_2^2$.

This theorem can be used on all values of $\mathbf{r}$, and consequently

$$\mathbf{r}(\mathbf{x}+\mathbf{h}) = \mathbf{r}(\mathbf{x}) + \mathbf{J}(\mathbf{x})\mathbf{h} + O(\|\mathbf{h}\|_2^2) \tag{C.4}$$

where $\mathbf{J}(\mathbf{x})$ is the Jacobian matrix (for $\mathbf{x} \in \mathbb{R}^n$) for N samples

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial r_1}{\partial x_1} & \frac{\partial r_1}{\partial x_2} & \cdots & \frac{\partial r_1}{\partial x_n} \\ \frac{\partial r_2}{\partial x_1} & \frac{\partial r_2}{\partial x_2} & \cdots & \frac{\partial r_2}{\partial x_n} \\ \vdots & & & \vdots \\ \frac{\partial r_N}{\partial x_1} & \frac{\partial r_N}{\partial x_2} & \cdots & \frac{\partial r_N}{\partial x_n} \end{bmatrix} \tag{C.5}$$

Define an approximate $\ell(\mathbf{h})$ and let equation (C.4) show that

$$\mathbf{r}(\mathbf{x}+\mathbf{h}) \simeq \ell(\mathbf{h}) \equiv \mathbf{r}(\mathbf{x}) + \mathbf{J}(\mathbf{x})\mathbf{h} \tag{C.6}$$

Further, inserting into the cost function from equation (C.2), and defining another approximation $L(\mathbf{h})$ yields

$$\begin{aligned} S(\mathbf{x}+\mathbf{h}) \simeq L(\mathbf{h}) &\equiv \frac{1}{2}\ell(\mathbf{h})^T\ell(\mathbf{h}) && \text{(C.7)} \\ &= \frac{1}{2}\mathbf{r}^T\mathbf{r} + \mathbf{h}^T\mathbf{J}^T\mathbf{r} + \frac{1}{2}\mathbf{h}^T\mathbf{J}^T\mathbf{J}\mathbf{h} && \text{(C.8)} \\ &= S(\mathbf{x}) + \mathbf{h}^T\mathbf{J}^T\mathbf{r} + \frac{1}{2}\mathbf{h}^T\mathbf{J}^T\mathbf{J}\mathbf{h} && \text{(C.9)} \end{aligned}$$

where $\mathbf{r} = \mathbf{r}(\mathbf{x})$ and $\mathbf{J} = \mathbf{J}(\mathbf{x})$.

If standing at an arbitrary starting guess $\mathbf{x}$, wanting to perturb the argument by $\mathbf{h}$ towards $\mathbf{x}_{min}$, then the perturbation step $\mathbf{h}$ can be found by minimizing $L(\mathbf{h})$ (cf. equation (C.7)). Indeed this is the *Gauss-Newton step*

$$\mathbf{h}_{gn} = \arg\min_{\mathbf{h}} L(\mathbf{h}) \tag{C.10}$$

Again, this can be solved by differentiating and equating to zero. Using vector differentiation rules found in e.g. Appendix B of Haykin [2002], the gradient of $L(\mathbf{h})$ is

$$\nabla_{\mathbf{h}} L(\mathbf{h}) = \mathbf{J}^T\mathbf{r} + \mathbf{J}^T\mathbf{J}\mathbf{h} \tag{C.11}$$

Equating to zero yields

$$\mathbf{J}^T\mathbf{J}\mathbf{h}_{gn} = -\mathbf{J}^T\mathbf{r} \tag{C.12}$$

and updating $\mathbf{x}$

$$\mathbf{x} = \mathbf{x} + \mathbf{h}_{gn} \tag{C.13}$$

Remember, the cost function is only an approximation to the desired cost function, and cf. equation (C.4), the larger the step is the larger the approximation error. That means if the starting guess is far from the solution, some iterations are needed for convergence, or in the worst case there will be no convergence at all. In the case of convergence problems, a *line search* algorithm should be implemented. When using the line search algorithm, updating $\mathbf{x}$ is

$$\mathbf{x} = \mathbf{x} + \alpha\mathbf{h}_{gn} \tag{C.14}$$

The α parameter is determined by the line search algorithm. Basically, it will investigate the cost function for different values of α to determine how far in the descent direction to perturb $\mathbf{x}$. Since $\mathbf{h}_{gn}$ is a guaranteed descent direction, convergence is guaranteed when using line search given that $\mathbf{J}(\mathbf{x})$ has full rank. It has not been necessary to implement line search in our algorithm.

APPENDIX D

ULTRA WIDEBAND EVOLUTION

D.1 Ultra Wideband from an information theoretical viewpoint

After the era of communicating with morse code and the telegraph, the first widespread use of electromagnetic waves for communication was the Amplitude Modulation (AM) radio. In AM the signal is modulated by changing the amplitude, always transmitting at the same frequency. Hence AM is the most narrow band technology. Later Frequency Modulation (FM) was invented, introducing "wideband" radio. By modulating the carrier frequency corresponding to the signal's amplitude, a wider band is used. This modulation was immune to noise caused by natural amplitude variations. Together with the possibility of using a wider spectrum, this allowed FM to yield much better signal-to-noise ratios.

The introduction of information theory in the late 1940's redefined the relationship between power density, noise and information capacity. In this new paradigm, it was known that the more an information signal is spread in bandwidth in a way that resembles background noise, the more information it is capable of holding. To another coexisting signal that is similarly spread, this information is regarded as background noise. Since, under some specific conditions, signal energy can be detected more efficiently than noise energy, both signals can coexist. As a result, higher bandwidth allows more signals to coexist and more total information can be conveyed. Until that point radio transmission had mostly been based on transmitting a signal with high power and low bandwidth, whereas now one could alternatively transmit a signal with low power density and high bandwidth.

This discovery led to a technique called "spread-spectrum" modulation. Spread-spectrum occupies a wide frequency band allowing many nodes to communicate in the same band. Before transmitting, the output is modulated by a certain code. Only one of the nodes on the network will receive the message, while the signal appears to other nodes as noise. This greatly improved the capacity of for instance cellular systems. In 1995 we saw commercial deployment of Code Division Multiple Access (CDMA) multiplexing in cellular networks. CDMA is the main competitor against GSM, the latter which uses more a more traditional combination of Time Division Multiple Access and Frequency Division Multiple Access. In 1999 the 3G standard emerged, which is based on CDMA technology. Today we see a significant growth in devices applying spread-spectrum techniques in personal communications. Most of these coexist well with other devices utilizing the same frequency band, thanks to the coding scheme.

The next step in developing wideband techniques was to go even wider. Research was concentrated towards developing short-impulse signaling between antennas. It is known from fourier theory that as a time-domain pulse grows shorter and shorter in time, approaching the diracs delta, the spectrum grows wider and wider, approaching the flat spectrum of white noise. From this research emerged the technology called “impulse radio", which we refer to today as Ultra Wideband radar. In 2002 the Federal Communications Commission (FCC) made available a band spanning from 3.1GHz to 10.6GHz for commercial communications development in the United States. It is this technology that the radar in this project is based on.

APPENDIX E

HARMONIC CANCELLER, MORGAN AND ZIERDT TYPE

This chapter describes the type of harmonic canceller described in Morgan and Zierdt [2009]. The notation is mostly kept from the original artice, so the variables may not be named as expected.

Most importantly, the residuals are called $y(n)$, the range data are called $x(n)$, the model is called $\hat{v}(n)$ and the parameter vector is called $\mathbf{w}$.

E.1 The optimization problem

The model for the unwanted harmonic components is

$$v(n) = \sum_{l=1}^{L} A_l \cos(l\omega_0 n + \phi_l) \tag{E.1}$$

$$= \mathbf{Re}\left\{\sum_{l=1}^{L} A_l e^{j\phi_l} e^{j\omega_0 n}\right\} \tag{E.2}$$

$$= \mathbf{Re}\left\{\sum_{l=1}^{L} h_l e^{j\omega_o n}\right\} \tag{E.3}$$

where $h_l = A_l e^{j\phi_l}$ and $\omega_0 = 2\pi f_0$. Define an estimate $\hat{v}(n)$ of $v(n)$;

$$\hat{v}(n) = \mathbf{Re}\left\{\sum_{l=1}^{L} w_l e^{j\hat{\omega}_0 n}\right\} \tag{E.4}$$

where w_l is an approximation of h_l and $\hat{\omega}_0$ is an approximation of ω_0. The system output is thus

$$y(n) = x(n) - \hat{v}(n) - w_0 \tag{E.5}$$

where w_0 here is the DC term.

Using $\mathbf{w} = \begin{bmatrix} w_0 & w_1 & \cdots & w_L \end{bmatrix}^T$ [1] we can define a cost function as the mean of the squared residual errors,

$$S(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} y^2(n) \tag{E.6}$$

Finally, the optimization problem is formally defined as

$$\mathbf{w}_{\min} = \arg\min_{\mathbf{w}} S(\mathbf{w}) \tag{E.7}$$

E.2 Rewriting the cost function

Equation (E.6) cannot readily be minimized, so we need to do some adjustments to it. Inserting equation (E.5) into equation (E.6) yields

$$S(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} [x(n) - \hat{v}(n) - w_0]^2 \tag{E.8}$$

From equation (E.8) it is seen that the w_0 that minimizes the cost function is the mean of the differences $x(n) - \hat{v}(n)$ for all n

$$w_{0,\min} = \frac{1}{N} \sum_{n=1}^{N} (x(n) - \hat{v}(n))$$

The cost function can now be expanded to

[1] For convenience of notation, the vector $\mathbf{w}$ contains $w0$ now, but later $w0$ will be replaced by the mean of the input signal, and thus determininistic with respect to the optimization procedure

$$S(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\left[x(n) - \hat{v}(n) - \frac{1}{N}\sum_{m=1}^{N}[x(m) - \hat{v}(m)]\right]^2 \tag{E.9}$$

$$= \frac{1}{N}\sum_{n=1}^{N}\left[\underbrace{x(n) - \frac{1}{N}\sum_{m=1}^{N}[x(m)]}_{\tilde{x}(n)} + \frac{1}{N}\sum_{m=1}^{N}[\hat{v}(m)] - \hat{v}(n)\right]^2 \tag{E.10}$$

$$= \frac{1}{N}\sum_{n=1}^{N}\left[\tilde{x}(n) - \mathbf{Re}\left\{\sum_{l=1}^{L}\left[w_l e^{jl\hat{\omega}_0 n} - \frac{1}{N}\sum_{m=1}^{N} w_l e^{jl\hat{\omega}_0 m}\right]\right\}\right]^2 \tag{E.11}$$

$$= \frac{1}{N}\sum_{n=1}^{N}\left[\tilde{x}(n) - \mathbf{Re}\left\{\sum_{l=1}^{L} w_l \left[\underbrace{e^{jl\hat{\omega}_0 n} - \frac{1}{N}\sum_{m=1}^{N} e^{jl\hat{\omega}_0 m}}_{\tilde{e}_l(n)}\right]\right\}\right]^2 \tag{E.12}$$

With these definitions, the cost function can be written as

$$S(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\left[\tilde{x}(n) - \mathbf{Re}\left\{\sum_{l=1}^{L} w_l \tilde{e}_l(n)\right\}\right]^2 \tag{E.13}$$

$$\tilde{x}(n) = x(n) - \frac{1}{N}\sum_{m=1}^{N}[x(m)] \tag{E.14}$$

$$\tilde{e}_l(n) = e^{jl\hat{\omega}_0 n} - \frac{1}{N}\sum_{m=1}^{N} e^{jl\hat{\omega}_0 m} \tag{E.15}$$

where $\tilde{x}(n)$ and $\tilde{e}_l(n)$ are zero-mean over the data block.

Expanding equation (E.13) yields;

$$S(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\left[\tilde{x}(n) - \mathbf{Re}\left\{\sum_{l=1}^{L} w_l \tilde{e}_l(n)\right\}\right]^2 \tag{E.16}$$

$$= \frac{1}{N}\sum_{n=1}^{N}\left[\tilde{x}^2(n) - 2\mathbf{Re}\left\{\sum_{l=1}^{L} w_l \tilde{x}(n)\tilde{e}_l(n)\right\} + \mathbf{Re}\left\{\sum_{l=1}^{L} w_l \tilde{e}_l(n)\right\}^2\right] \tag{E.17}$$

Define the input power $P_{\tilde{x}} = \frac{1}{N}\sum_{n=1}^{N}\tilde{x}^2(n)$ and a correlation vector $\mathbf{r}$ with elements $r_l = \frac{1}{N}\sum_{n=1}^{N}\tilde{x}(n)\tilde{e}_l^*(n)$, and we can write

$$S(\mathbf{w}) = P_{\tilde{x}} - 2\mathbf{Re}\left\{\mathbf{r}^H\mathbf{w}\right\} + \frac{1}{N}\sum_{n=1}^{N}\mathbf{Re}\left\{\sum_{l=1}^{L} w_l\tilde{e}_l(n)\right\}^2 \tag{E.18}$$

The last term can be written as the triple sum;

$$\frac{1}{N}\sum_{n=1}^{N}\mathbf{Re}\left\{\sum_{l=1}^{L} w_l\tilde{e}_l(n)\right\}^2 = \frac{1}{N}\sum_{n=1}^{N}\mathbf{Re}\left\{\sum_{l=1}^{L}\sum_{k=1}^{L}[w_l\tilde{e}_l(n)\tilde{e}_k(n)w_k]\right\} \tag{E.19}$$

$$= \mathbf{Re}\left\{\sum_{l=1}^{L}\sum_{k=1}^{L}[w_l\frac{1}{N}\sum_{n=1}^{N}[\tilde{e}_l(n)\tilde{e}_k(n)]w_k]\right\} \tag{E.20}$$

By splitting w_l and $\tilde{e}_l$ into their real and complex parts $w_{R,l} + jw_{I,l}$ and $\tilde{e}_{R,l} + j\tilde{e}_{I,l}$ and expanding, it can be shown that

$$\begin{aligned}&\mathbf{Re}\left\{\sum_{l=1}^{L}\sum_{k=1}^{L}[w_l\frac{1}{N}\sum_{n=1}^{N}[\tilde{e}_l(n)\tilde{e}_k(n)]w_k]\right\} = \\ &\frac{1}{2}\left[\sum_{l=1}^{L}\sum_{k=1}^{L}[w_l^*\frac{1}{N}\sum_{n=1}^{N}[\tilde{e}_l(n)\tilde{e}_k^*(n)]w_k\right] + \\ &\frac{1}{2}\mathbf{Re}\left\{\sum_{l=1}^{L}\sum_{k=1}^{L}[w_l\frac{1}{N}\sum_{n=1}^{N}[\tilde{e}_l^*(n)\tilde{e}_k^*(n)]w_k\right\}\end{aligned} \tag{E.21}$$

and by introducing the complex and hermitian correlation matrixes $\mathbf{R}$ and $\mathbf{C}$ defined by their components

$$R_{lk} = \frac{1}{N}\sum_{n=1}^{N}\tilde{e}_l(n)\tilde{e}_k^*(n) \tag{E.22}$$

$$C_{lk} = \frac{1}{N}\sum_{n=1}^{N}\tilde{e}_l^*(n)\tilde{e}_k^*(n) \tag{E.23}$$

equation (E.18) can be written as

$$S(\mathbf{w}) = P_{\tilde{x}} - 2\mathbf{Re}\left\{\mathbf{r}^H\mathbf{w}\right\} + \frac{1}{2}\mathbf{w}^H\mathbf{R}\mathbf{w} + \frac{1}{2}\mathbf{Re}\left\{\mathbf{w}^T\mathbf{C}^*\mathbf{w}\right\} \tag{E.24}$$

The optimal $\mathbf{w}$ can be found by differentiating S with respect to $\mathbf{w}^*$ and equating to zero. Rules for conjugate vector differentiation can be found in Appendix B of Haykin [2002].

$$\nabla_{\mathbf{w}^*} S(\mathbf{w}) = -\mathbf{r} + \frac{1}{2}\mathbf{R}\mathbf{w} + \frac{1}{2}\mathbf{C}\mathbf{w}^* = 0 \tag{E.25}$$

To remove $\mathbf{w}^*$ we can premultiply the conjugate of equation (E.25) with $\mathbf{C}\mathbf{R}^{-*}$:

$$\mathbf{C}\mathbf{R}^{-*} \left(\nabla_{\mathbf{w}^*} S(\mathbf{w})\right)^* = -\mathbf{C}\mathbf{R}^{-*}\mathbf{r}^* + \frac{1}{2}\mathbf{C}\mathbf{w}^* + \frac{1}{2}\mathbf{C}\mathbf{R}^{-*}\mathbf{C}\mathbf{w}^* = 0 \tag{E.26}$$

Subtracting equation (E.26) from equation (E.25) yields

$$\begin{aligned} -\mathbf{r} + \frac{1}{2}\mathbf{R}\mathbf{w} + \frac{1}{2}\mathbf{C}\mathbf{w}^* - \left(-\mathbf{C}\mathbf{R}^{-*}\mathbf{r}^* + \frac{1}{2}\mathbf{C}\mathbf{w}^* + \frac{1}{2}\mathbf{C}\mathbf{R}^{-*}\mathbf{C}^*\mathbf{w}\right) &= 0 \\ \frac{1}{2}\mathbf{w}(\mathbf{R} - \mathbf{C}\mathbf{R}^{-*}\mathbf{C}^*) &= \mathbf{r} - \mathbf{C}\mathbf{R}^{-*}\mathbf{r}^* \\ \mathbf{w} &= 2(\mathbf{R} - \mathbf{C}\mathbf{R}^{-*}\mathbf{C}^*)^{-1}(\mathbf{r} - \mathbf{C}\mathbf{R}^{-*}\mathbf{r}^*) \end{aligned} \tag{E.27}$$

APPENDIX F

MATLAB CODE

This appendix lists the code used to produce the results in this thesis, in the order that they are used.

F.1 Functions for separating heart and respiration signal components

The two functions used for extracting the heart signal components are listed. Listing F.1 lists the FIR filtering method and Listing F.2 lists the harmonic canceller.

Listing F.1: FIR filtering extraction of the heartbeat signal

```
function [heart, snr] = firseparate( inputsignal, fs, N, beta, plotvar, ...
    varargin )
%FIRSEPARATE Cancel the respiration component of radar data using FIR
% filtering.
%
% Input variables:
% inputsignal:  Input signal
% fs:           Sampling frequency
% N:            Desired number of filter coefficients
% beta:         Beta factor for the Kaiser window
% plotvar:      'plot' for plotting results
% varargin:     For testing purposes, input real heart component here,
%               and get the signal-to-noise ratio in the snr output
%               variable

hpfreq = 0.65;

% Create filter
normhpfreq = 2*hpfreq/fs;
h = fir1( N, normhpfreq, 'high', kaiser( N+1, beta ) );

% Filter inputsignal
heart = filter( h, 1, inputsignal );
```

```
% Plot results
if nargin > 5
realheart = varargin{1};
realheart = realheart - mean( realheart );
realheart = circshift( realheart, [0 round( N/2 )] );
t = (0:(length( inputsignal ) -1)) ./ fs;
if strcmp( plotvar, 'plot' )
    subplot( 2, 1, 1 ); set( gca, 'Fontsize', 12 );
    plot( t, inputsignal );
    title( 'Input␣signal' );
    xlabel( 'Time␣[s]' );
    ylabel( 'Amplitude' );
    subplot( 2, 1, 2 ); set( gca, 'Fontsize', 12 );
    plot( t, heart ); hold on;
    plot( t, realheart, 'r' );
    title( 'Esimtated␣and␣real␣heartsignal' );
    xlabel( 'Time␣[s]' );
    ylabel( 'Amplitude' );
    legend( 'Estimated␣heart', 'Real␣heart' );
end
snr = norm( realheart )^2 / norm( heart - realheart)^2;
disp( ['SNR␣=␣' num2str( snr )] );
end

end
```

Listing F.2: Harmonic cancelling of the respiration signal component

```
function [residuals, fitvect, snr] = gnfit( input, fs, x0, w, winlength,...
    varargin )
%GNFIT Fit radar range data to a model of harmonic components.
% [residuals, fitvect, snr] = gnfit( input, fs, x0, w,winlength,varargin )
% outputs the residuals, fitted respiration signal and the signal-to-noise
% ratio given that the varargin{1} is the real heart signal.
% Use input signal 'input', sample rate 'fs', an initial parameter vector
% starting guess 'x0'(nonzero!), initial respiration frequency guess 'w'
% and the window length 'winlength'. Use varargin{2} to control the length
% of the window used to estimate the fundamental respiration frequency.

if nargin > 6
    f0win = varargin{2};
else
    f0win = 20;
end
convsamps = 15;
f0update = 10;

winint = round( winlength * fs );
f0winint = round( f0win * fs );

input = input(:);
fitvect = zeros( length( input ), 1 );

x = x0;

holdvar = 0;

for i = winint:length( input )
    if (~mod( i, f0winint ))
        oldx = x;
        oldw = w;
        t = ((i-f0winint+1):i)';
        f0sig = (input(t)-mean(input(t)));
        w = findw0( f0sig, fs );
        disp( ['New w0 = ' num2str(w)] );
        holdvar = convsamps;
    end
    t = ((i-winint+1):i)';
    y = input(t);
    J = [-cos(w*t+x(2)) x(1)*sin(w*t+x(2)) -cos(2*w*t+x(4)) ...
        x(3)*sin(2*w*t+x(4)) -cos(3*w*t+x(6)) x(5)*sin(3*w*t+x(6)) ...
        -1 * ones( winint, 1 ) ];
    fit = x(1)*cos(w*t+x(2))+x(3)*cos(2*w*t+x(4))+x(5)*cos(3*w*t+x(6))+x(7);
    r = y - fit;
    if holdvar > 0
        holdvar = holdvar - 1;
        thisfit = oldx(1)*cos(oldw*i+oldx(2))+oldx(3)*...
            cos(2*oldw*i+oldx(4))+oldx(5)*cos(3*oldw*i+oldx(6))+oldx(7);
        fitvect(i) = thisfit;
    else
        fitvect(i) = fit(end);
    end
    if 0   % Solve hgn with direct slash operator or cholesky
        hgn = (J'*J)\((-J')*r);
    else
        L = chol( J'*J );
        zgn = L' \ ((-J')*r);
        hgn = L \ zgn;
    end
    x = x + hgn;

end
residuals = input - fitvect;
t = (0:(length( input ) - 1))./fs;
figure; set( gca, 'FontSize', 20 );
plot( t, input );
hold on;
plot( t, fitvect, 'r' );
legend( 'Measured data', 'Fitted data' );
title( 'Respiration datafit' );
xlabel( 'Time [s]' );
ylabel( 'Amplitude' );
figure; set( gca, 'FontSize', 20 );
plot( t, residuals );
title( 'Residuals (heartbeat signal)' );
xlabel( 'Time [s]' );
ylabel( 'Amplitude' );

if nargin > 5
    realheart = varargin{1};
    realheart = realheart - mean( realheart );
    errors = residuals(winint:end) - realheart(winint:end);
    snr = norm( realheart(winint:end) )^2 / norm( errors )^2;
    disp( ['SNR = ' num2str( snr )] );
    figure;
    plot( t, residuals ); hold on;
    plot( t, realheart, 'r' );
    title( 'Esimtated and real heartsignal' );
    xlabel( 'Time [s]' );
    ylabel( 'Amplitude' );
    legend( 'Estimated heart', 'Real heart' );
end
end
```

```
function [w0] = findw0( argin , fs )
startlag = 300;
stop = 0;
k = 1;
while ~stop
    [maxcorr , lag] = hookejeeves( @corr , k*startlag , 20, 1, 50, argin )
    if lag > 20 && (maxcorr < 1)
        stop = 1;
    end
    k = k+1;
end
stop = 0;
while ~stop
    [maxcorr , lagcheck] = hookejeeves( @corr , lag/2 , ...
        20, 1, 50, argin );
    if (lagcheck < 3) || ((lagcheck > 0.95*lag) && ...
            (lagcheck < 1.05*lag)) || (maxcorr > 1)
        stop = 1;
    else
        lag = lagcheck;
    end
end
f0 = fs/lag;
w0 = f0/fs*2*pi;

end
```

F.2 Functions for estimating the heart rate

Three methods are listed. Heart rate estimation by the autocorrelation function is listed in Listing F.3, heartbeat detector by RLS filtering in Listing F.4 and the RLS heartrate estimator in Listing F.5

Listing F.3: Heart rate detection by the autocorrelation function

```
function [heartrate] = findheartrate( input, fs , varargin )
% FINDHEARTRATE Calculate heart rates by using the autocorrelation method
% [heartrate] = findheartrate( input , fs , varargin ) returns a vector
% containing heart rates , on the basis of input and the sample rate fs .
% Variable arguments:
% 1: Choice of window type , e.g. @rectwin or @hamming. Default is hanning.
% 2: 'unbiased' for unbiased autocorrelation function (default), and
%    'biased' for the biased autocorrelation function.
% 3: The length of the sliding window in seconds.

if nargin > 2
    winfunc = varargin{1};
else
    winfunc = @hanning;
end
if nargin > 3
    corrtype = varargin{2};
else
    corrtype = 'unbiased';
end

if nargin > 4
    windowsize = varargin{3};
else
    windowsize = 8;
```

```
end

% Initialize variables
N = round( windowsize*fs );
windowvec = window( winfunc, N );
input = input(:);

%Enter windowing loop

for i = N:length( input )

    win = input((i-N+1):(i));
    win = win .* windowvec;

    %Find peaks in autocorrelation

    startlag = 100;
    stop = 0;
    k = 1;
    while ~stop
        [maxcorr, lag, output] = hookejeeves( @corr, k*startlag, 10, ...
            1, 20, win, corrtype );
        if lag > 20 && (maxcorr < 1)
            stop = 1;
        end
        k = k+1;
    end
    stop = 0;
    while ~stop
        [maxcorr, lagcheck, output] = hookejeeves( @corr, lag/2 , ...
            20, 1, 50, win );
        if (lagcheck < 3) || ((lagcheck > 0.95*lag) && ...
                (lagcheck < 1.05*lag)) || (maxcorr > 1)
            stop = 1;
        else
            lag = lagcheck;
        end
    end
    heartrate(i) = fs/(lag)* 60;
end

end
```

Listing F.4: Heartbeat detector by RLS model estimation

```
function detector = rlsheartrate( input, p, b, fs );
%RLSHEARTRATE Calculates the heartbeat signal into a train of impulses.
% detector = rlsheartrate( input, p, b, fs ) returns the train of impulses,
% each occurring at the start of a heartbeat. Use p coefficients in the RLS
% filter, forgetting factor b and sample rate fs.

heartcurve = input(:);
t = (0:(length( input ) - 1))./fs;

A = zeros( p,1 );
P = 10*eye( p );
Pinv = inv( P );
detector = zeros( length( heartcurve ), 1 );
es = zeros( length( heartcurve ), 1 );
Ds = zeros( length( heartcurve ), 1 );
test = Ds;
outputsignal = heartcurve;
```

```
f = 0;
D=0;
for i = (p):(length( heartcurve ) - 1)
    X = outputsignal((i-p+1):i);
    est = A' * X;
    e = heartcurve(i+1) - est ;
    oldP = P;
    oldf = f;
    f = X'*oldP*X;
    K = oldP*X/(b+f);
    P = (oldP/b - K*X'*oldP/b);
    A = A + K * e;
    es(i) = e;
    Ds(i) = f - oldf;
    test(i) = f;
    detector(i) = Ds(i)*e;
end
%subplot( 2,1,1 );
plot( t, input ); set( gca, 'FontSize', 16 );
title( 'Heart␣signal' );
xlabel( 'Time␣[s]' );
ylabel( 'Amplitude' );
figure; set( gca, 'FontSize', 16 );
plot( t, detector );
title( 'Heartbeat␣detector' );
xlabel( 'Time␣[s]' );
ylabel( 'Detector␣output' );

end
```

Listing F.5: Heart rate detection from RLS heartbeat detector

```
function [heartrate] = findheartraterls( input, fs )
%FINDHEARTRATERLS Find the heartrates based on the RLS heartbeat detector
% [heartrate] = findheartraterls( input, fs ) returns the heartrates on the
% basis of the input (output of rlsheartrate.m) and the sample rate.

ff = 0.95;
s = 0.001;
i = 1;
stop = 0;
thishit = 1;
heartrate = zeros( 1, length( input ) );
while i < length( input )
    if (input(i)^2) > (5 * 1/(1-ff) * s)
        lasthit = thishit;
        thishit = i;
        hr = fs/(thishit - lasthit) * 60;
        heartrate(lasthit:thishit) = hr;
        i = i + 30;
    else
        s = ff * s + (1-ff) * input(i)^2;
        i = i + 1;
    end
end
t = (0:(length( input ) - 1)) ./ fs;
plot( t, heartrate ); set( gca, 'FontSize', 16 );
title( 'Heartrates␣vs␣time' );
xlabel( 'Time␣[s]' );
ylabel( 'Heartrate' );
```

F.3 Auxiliary functions

The functions `corr.m` (Listing F.6) and `hookejeeves.m` (Listing F.7) were used to calculate the correlation function and finding the Hooke & Jeeves functional minimizer. The Hooke & Jeeves functional minimizer was readily available at the start of this thesis, as it had been written as an assignment in a previous course in the author's studies.

Listing F.6: Autocorrelation function at given time lag

```
function [corrval] = corr( lag, input, varargin )
%CORR Calculates the autocorrelation coefficient
% [corrval] = corr( lag, input, varargin ) returns the correlation
% coefficient at time lag 'lag', of the input window 'input'.
% Use varargin for 'unbiased' or 'biased' correlation. Default is
% unbiased.

if nargin > 2
    option = varargin{1};
else
    option = 'unbiased';
end
lag = round( lag );
if lag < 0
    lag = -lag;
end
inputvector = [input; zeros( lag, 1 )];
corrvector = [zeros( lag, 1 ); input];
switch( option )
    case 'biased'
        corrval = -sum(inputvector.*corrvector);
    case 'unbiased'
        corrval = -sum(inputvector.*corrvector) / ...
            (length( input ) - abs( lag ));
end
end
```

Listing F.7: Hooke & Jeeves functional minimizer

```
function [ F,x_hat, output ] = hookejeeves ( fun,x,delta,deltamin,maxiter,varargin )
%HOOKEJEEVES Pattern-search method based multidimensional minimizer fuction
%   [ F, X_HAT, OUTPUT ] = HOOKEJEVES ( FUN, X, DELTA, DELTAMIN, MAXITER, VARARGIN )
%   starts at X in function handle FUN and explores in the direction DELTA
%   and -DELTA, to find a better point X_HAT with lower function value
%   than X. It will repeat the procedure until it no longer finds a better
%   point. Then it will try to decrease the step length DELTA. The function
%   stops on DELTA < DELTAMIN or number of iterations > MAXITER. VARARGIN
%   can be used to pass extra arguments to the function, if needed. X and
%   DELTA must be specified as a vector or scalar, and have identical
%   dimensions. Delta can also be specified as a vector or scalar, but this
%   does not need to be the same dimension as X and DELTA, as it is the
%   2-norm of DELTA and DELTAMIN that are compared.
%
%   F is the returned function value in the minimized point. X_HAT is the
%   best coordinates found at the time of the function ending. OUTPUT gives
%   the following output:
%
%   OUTPUT.DELTA is the step length at the end of the function
%   OUTPUT.EVAL is the number of function evaluations during the run
%   OUTPUT.ITER is the number of iterations the function used.
%
```

```
%   Example:
%   [ F,x_hat, output ] = hookejeeves ( @sin, 0, 0.1, 1e-6, 200 )
%   Returns the closest minimizer of sin(x) starting in x=1.5 and initially
%   stepping 0.1. The lowest allowable DELTA is 1e-6 and the maximum number
%   of iterations is 200.

eval=0;           % Keeps track of the number of function evaluations
[x_hat f_xhat exploreeval]=explore(fun,x,delta,varargin{:});
eval=eval+exploreeval;
stop=0;           % Stop=1 is stopping criteria
iter=0;           % Keeps track of the number of iterations
while stop==0
    iter=iter+1;
    eval=eval+1;
    f_x=feval(fun, round(x), varargin{:});
    if f_xhat<f_x;
        z=x_hat+(x_hat-x);
        x=x_hat;
    else
        z=x;
        delta=delta./2;
    end
    if(iter>maxiter || norm(delta) < norm(deltamin))
        stop=1; % Stop on maximum number of iterations or
                % maximum number of function evaluations
    end
    [x_hat f_xhat exploreeval]=explore(fun,z,delta,varargin{:});
    eval=eval+exploreeval;
end
% Set the output variables
F=feval(fun,x_hat,varargin{:});
output.iterations=iter;
output.eval=eval;
output.delta=delta;
end

function [x_h,f_xhat,exploreeval] = explore(fun,x,delta,varargin)
x_h=x;
I=eye(numel(x_h));
exploreeval=1;        % Keeps track of the number of function evaluations
                      % in the explore function
f_xhat=feval(fun, round(x_h), varargin{:});
for j=1:numel(x_h)
    Deltaj=delta(j)*I(j,:);
    f_xhatm=feval(fun, round(x_h-Deltaj), varargin{:});
    exploreeval=exploreeval+1;
    if (f_xhatm<f_xhat)
        x_h=x_h-Deltaj;
        f_xhat=f_xhatm;
    else
        f_xhatp=feval(fun, round(x_h+Deltaj), varargin{:});
        exploreeval=exploreeval+1;
        if(f_xhatp<f_xhat)
            x_h=x_h+Deltaj;
            f_xhat=f_xhatp;
        end
    end
end

end
```

LIST OF FIGURES

BIBLIOGRAPHY

M. S. Bartlett. An inverse matrix adjustment arising in discriminant analysis. In *The Annals of Mathematical Statistics*, volume 22, pages 107–111, 1951.

W.J. Byrne, R. Zapp, P. Flynn, and M. Siegel. Adaptive filter processing in microwave remote heart monitors. In *IEEE Transactions on Biomedical Engineering*, volume 33, pages 717–722, 1986.

Jr. C.H. Edwards and David E. Penney. *Elementary Linear Algebra*. Prentice Hall, 1988.

Amy Droitcour. *Non-contact measurement of heart and respiration rates with a single-chip microwave doppler radar*. PhD thesis, Stanford University, June 2006.

Simon Haykin. *Adaptive Filter Theory*. Prentice Hall, 4th edition, 2002.

James C. Lin. Noninvasive microwave measurement of respiration. In *Proceedings of the IEEE*, page 1530, October 1975.

B. Lohman, O. Boric-Lubecke, V.M. Lubecke, P.W. Ong, and M. M. Sondhi. A digital signal processor for doppler radar sensing of vital signs. In *Engineering in Medicine and Biology Society, Proceedings of the 23rd Annual International Conference of the IEEE*, volume 4, pages 3359–3362, 2001.

Henrik Madsen. *Time Series Analysis*. Chapman and Hall, 2007.

K. Madsen, H.B. Nielsen, and O. Tingleff. Methods for non-linear least squares problems. `http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3215/pdf/imm3215.pdf`, 2004.

Wansuree Massagram, Noah M. Hafner, Byung-Kwan Park, Victor M. Lubecke, Anders Host-Madsen, and Olga Boric-Lubecke. Feasibility of heart rate variability measurement from quadrature doppler radar using arctangent demodulation with dc offset compensation. In *Proceedings of the 29th Annual International Conference of the IEEE EMBS*, August 2007.

Florian Michahelles, Ramon Wicki, and Bernt Schiele. Less contact: Heart-rate detection without even touching the user. In *Eighth International Symposium on Wearable Computers*, volume 1, pages 4–7, November 2004.

Dennis R. Morgan and Michael G. Zierdt. Novel signal processing techniques for doppler radar cardiopulmonary sensing. In *Signal Processing*, volume 89, pages 45–66, 2009.

John G. Proakis and Dimitris K Manolakis. *Digital Signal Processing*. Prentice Hall, 4th edition, 2006.

Phil Schniter. Time-frequency uncertainty principle. From Connexions web, `http://cnx.org/content/m10416/2.18/`, October 2005.

Kazimierz Siwiak and Debra McKeown. *Ultra Wideband Radio Technology*. Wiley, 2004.

Merrill Skolnik, editor. *Radar Handbook*. McGraw Hill, 3rd edition, 2008.

Enrico M. Staderini. Everything you always wanted to know about uwb radar...: a practical introduction to the ultra wideband technology. *Online Symposium of Electronics Engineers*, 2008.

Eric W. Weisstein. Geometric series. From MathWorld - A Wolfram Web Resource, `http://mathworld.wolfram.com/GeometricSeries.html`, 2009.