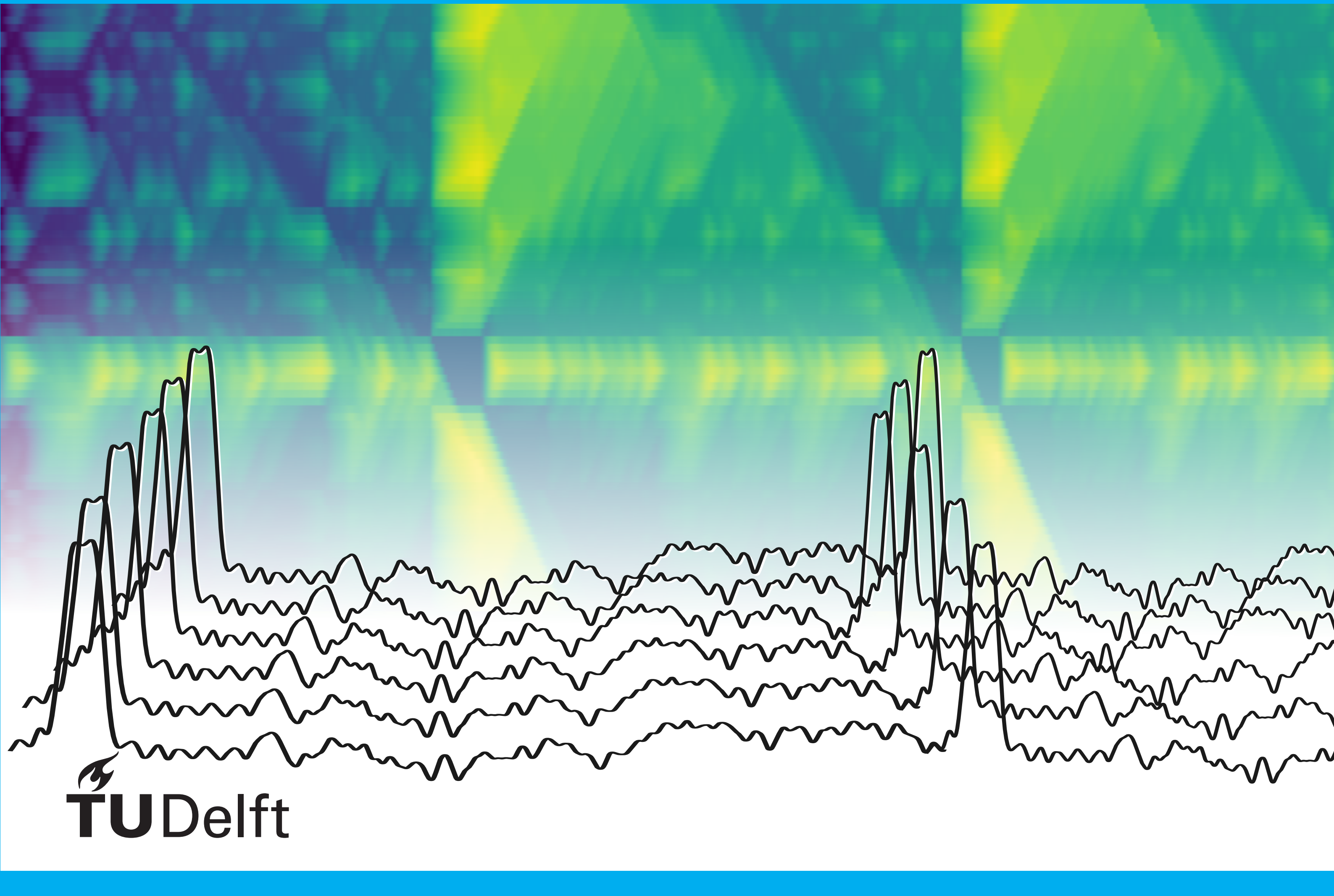
Phase estimation

of recurring patterns in nonstationary signals

R.A. van der Vlist

Msc. Thesis



Phase estimation of recurring patterns in nonstationary signals

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Electrical Engineering

by

R.A. van der Vlist B.Sc.
born in Bodegraven, The Netherlands

This work was performed in:

Circuits and Systems Group
Department of Microelectronics & Computer Engineering
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology
Department of
Microelectronics & Computer Engineering

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"Phase estimation of recurring patterns in nonstationary signals"** by **R.A. van der Vlist B.Sc.** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: September 27th, 2018

Chairman: ______________________
dr.ir. R. Heusdens

Advisor: ______________________
dr.ir. C.H. Taal

Committee Members: ______________________
dr.ir. J.H. Weber

____

Abstract

A phase estimation algorithm is presented to estimate the phase of a recurring pattern in a nonstationary signal. The signal is modelled by a template signal that represents one revolution of the recurring pattern, and that the frequency of this pattern can change at any time with no assumptions about local stationarity. The algorithm uses a constraint maximum likelihood estimator (MLE) to estimate the phase of the recurring pattern in the time series. Using the dynamic programming techniques from the dynamic time warping (DTW) algorithm, the solution is found in an efficient manner. The algorithm is applied to the digitization of meter readings from analog consumption meters.

As of today, analog consumption meters are still widely used to measure the consumption of gas, electricity and water. Often, smart home appliance use a simple reflective photosensor located on a rotating part of the meter to obtain information about the state of the consumption meter. The algorithm presented in this thesis accurately estimates the phase of the repeating pattern that occurs in the sensor observation when the meter rotates. Using this estimate, the signal of the photosensor can be converted to an estimate of the total resource consumption and consumption rate.

The algorithm improves in accuracy over conventional methods based on peak detection, and is shown to work in cases where the peak detection methods fails. Examples of this are signals where there is no distinctive peak in the signal or a signal where the recurring pattern is reversed. Furthermore, a template compression scheme is proposed that is used to decrease the computational complexity of the algorithm. Different time series compression methods are applied to the algorithm and evaluated on their performance.

Preface

This report was written in conclusion of my Master Electrical Engineering - Signals and Systems at the Technical University of Delft, and the research was conducted at Quby.

Foremost I would like to express my gratitude to the people that helped me make this project what it is today. I am grateful to the people at Quby for offering me a place in the company, and it was great to work with so many research- and quality minded people. My thanks go especially to Cees Taal for both his scientific and practical feedback, for the useful meetings and his guidance in the project. Thanks to Richard Heusdens as well for the bi-weekly meetings and for taking the effort to travel to Quby numerous times. The meetings helped me to keep focus in the right direction and to reflect on the things that I had overlooked. I would like to thank my thesis committee for reading this report and investing into this project. My thanks go to Menno van der Reek for all his research and code which I could utilize, for being supportive in the process and for all the nice lunch walks. I also would like to thank Erni Durdevic for letting us measure his rare mirrorless gas meter and for the delicious pasta. Finally, a word of thanks to Bart van Ginkel for helping me formalise some steps of the proof and the notation.

I feel grateful to have been in an environment with so many skilled and talented people, from whom I have learned a lot. I am glad for all the knowledge and insight I have been able to gather during my studies, which feels as a strong foundation that I can build upon in the rest of my career.
It has always fascinated me how many devices and services we have around us everyday, all with their own story of the countless hours it took to design and engineer them. Technology has vastly changed over the past decade, enabling countless new applications for signal processing and changing the way we live our lives and spend our time. It is exciting to see the technology change, and I hope to be a part of the engineering behind the world that surrounds us, and to make it a change for the better. I feel blessed for my time in Delft and for all the people that I have come to know, both inside and outside the university. My time as a student has been a great journey with experiences that will stay with me for a long time.

Rik van der Vlist

Delft, September 2018

Contents

1 Introduction

Managing the increased demand for energy and reducing its impact on the environment is one of the biggest challenges of the twenty-first century. A decreasing stock of fossil fuel sources along with increased awareness of the environmental effects of high energy consumption have created an urge for more efficient resource consumption. A simple first step in the process towards resource efficiency is to gain more insight into energy usage, and to reflect on the current efficiency of resource consumption.

Insight in energy usage is key to raising awareness. Particularly on a customer level, it can be difficult to gain insight into the usage of resources such as electricity, gas or water. At many times, the only insight for consumers is the yearly bill, which only presents the total sum of the consumption, and does not indicate any sources of waste or shows a breakdown of the cost. Providing end users with insight about their household consumption can reduce waste, but also help raise general awareness about how we can reduce consumption. Studies from the Europian Union [1] have shown that users that were provided with personal consumption insights had savings up to 9%.

Recently, numerous solutions have been developed that help customers in this process. This is becoming increasingly easy because many analogue consumption meters are being replaced by smart meters: digital meters that can automatically report usage to the provider or to other endpoint devices in the home of the user. Smart home appliances that aim at monitoring and providing insight into resource consumption obtain their information by interfacing with a smart meter. The appliance converts the data from the meter into graphs or advice to the end user.

The adoption of smart meters is unfortunately a slow process, since the meter can only be installed when the customer agrees to the installation and because due to high costs, replacement is spread out over multiple years. Recent data from Netbeheer Nederland has shown that the adoption of smart meters was around 50% in March 2018 [1]. When no smart meter is present, a method is needed to convert the readings from the analogue meter to a digital format. Therefore, a simple sensor is usually in place that digitises the readings from analogue consumption meters.

Estimating the state of an analogue consumption meter can be a challenging task. In this work, the signals from a typical reflective photosensor are analysed, and it is shown how the observed signals of the meter correlate to the state of the consumption meter. An algorithm is presented that efficiently and accurately estimates the state of the consumption meter from the observed signals, and that works for a variety of meter types. Using a number of simulations, the performance of the algorithm is evaluated, and compared to the conventional techniques.

[1] Data received in personal communication with Netbeheer Nederland (11-04-18) and shared with permission.

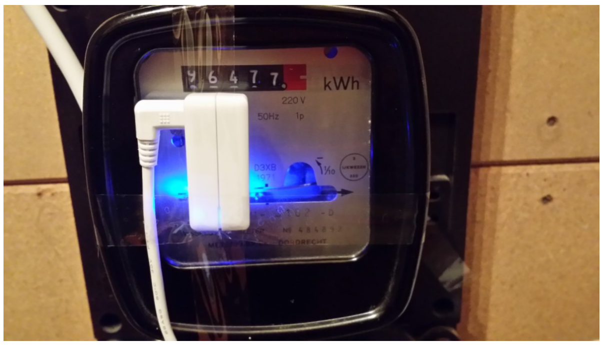


Figure 1.1: A picture of the photosensor that measures the reflectivity of a rotating part of the consumption meter.

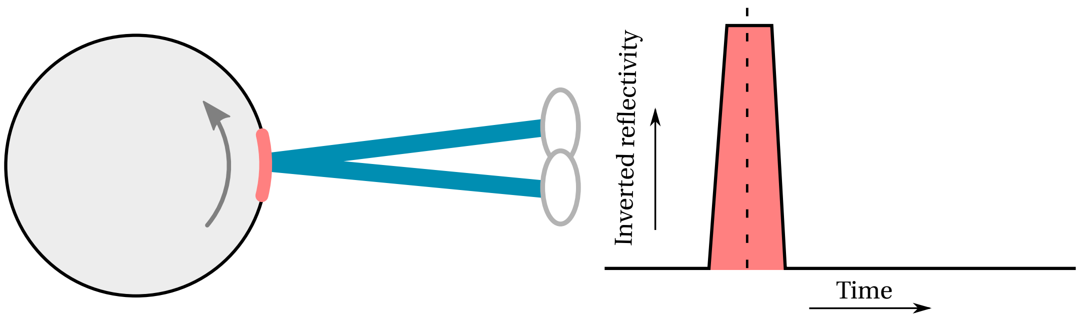


Figure 1.2: On the left, a schematic view is shown of the blue light emitted by the sensor and the reflection on the disc of the electricity meter. On the right side, a sketch of the observed reflection is shown. The peak generated by the marking on the disc can be used to track the position of the disc.

1.1. Optical sensing of analogue meters

A photograph of a small piece of hardware sensing an electricity meter is shown in Figure 1.1 and the working principle of this sensor is depicted in Figure 1.2. The photosensor is suited for analogue meters that have a part that rotates with a frequency proportionally to the consumption. In the case of an electricity meter, a spinning disc below the counter indicates the usage. When the sensor is placed over the edge of this disc, the marking on the disc generates a peak in the observed reflection, which is used to track the meter position. Each meter has a predefined *C value* that indicates how many revolutions the disc makes per kWh. The total consumption can then be found from the total number of revolutions. By measuring the speed of the spinning disc the current consumption rate can be calculated. Generally, the speed of the spinning disc is estimated by measuring the time it takes to complete a revolution.
Not all consumption meters have a spinning disc such as shown in Figures 1.1 and 1.2, but generally all meters have some part that rotates with a frequency proportional to the consumption, such as a small pointer or a mechanical counter.

1.2. Optical signal properties

Most consumption meters have an area with high contrast on one of the rotating parts. Figure 1.4 shows a number of examples of meters where the contrast area is indicated with a blue circle. Almost all meters have either a small mirror or a red or black marking on the disc, which creates a positive or a negative peak in the reflection measured by the photosensor. This feature enables the device that uses the photosensor to count the number of revolutions with a simple peak detection algorithm. A few examples of recordings from the photosensor are shown in Figure 1.3. From left to right recorded reflections for an electricity meter, a gas meter and a water meter are shown. Peaks with high values indicate absorption, whereas peaks with low value are reflections. Some other visual features from the meters can also be recognised in the recordings. When comparing the recorded observation from the water meter in Figure 1.3 with the picture of a water

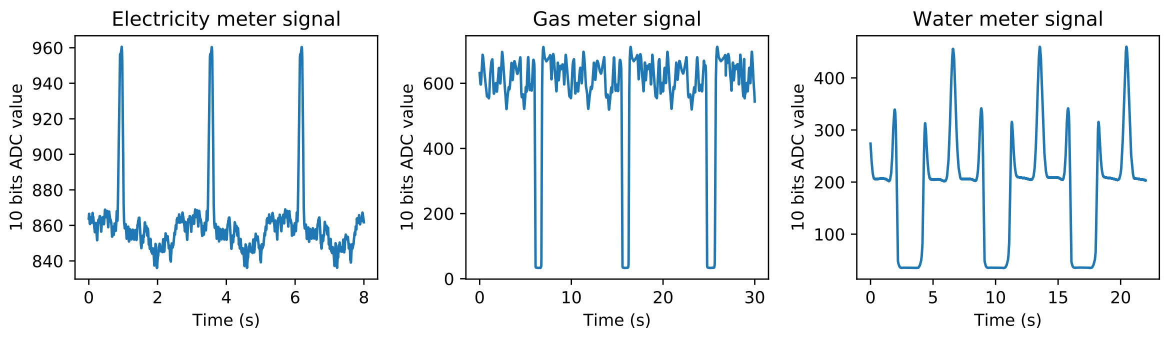


Figure 1.3: Recorded reflections for different types of meters. A higher value indicates less reflection, or higher absorption of transmitted light. From left to right the figures show full rotations each for an electricity meter, a gas meter and a water meter.

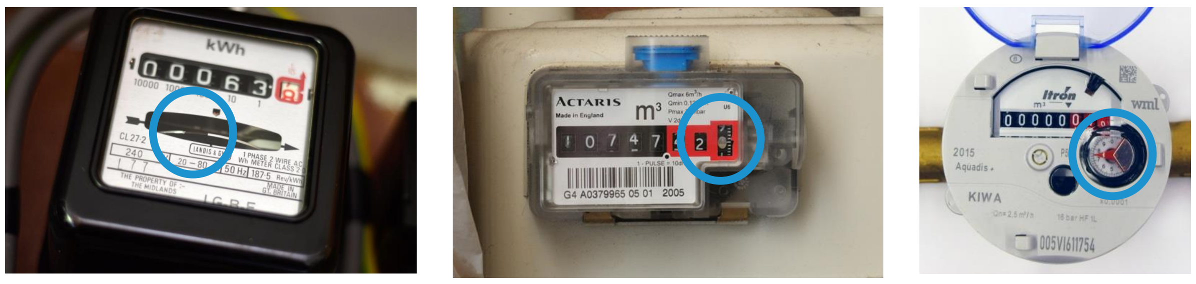


Figure 1.4: From the left to the right an electricity meter, gas meter and water meter are shown. The blue circles indicate a high contrast area of a rotating part of the consumption meter. The high contrast area can either be a mirror that causes a peak in the reflected light, or a black or red marking that causes a decrease in reflected light.

meter in Figure 1.4, the signal peaks can be linked to visual features: for the water meter we see a short peak for the orange pointer on the indicator, and a larger trough for the mirror on the opposite side of the indicator. Similarly, upon closer examination ten small peaks can be discovered in the gas meter signal for each of the digits of the mechanical counter. The electricity meter signal clearly shows a peak from the marking on the disc, but interestingly there are other recurring features in the signal as well. Most likely, those features are generated by impurities on the disc or because the disc is not perfectly round, causing a variation in distance between the disc edge and the sensor.

1.3. Problem definition

As described in the previous signals, many consumption meter types have a marking or a mirror on of the rotating parts, that generates a peak in the signal observed by the photosensor. Yet all of the signals in Figure 1.3 also have other clear repetitive features next to the main signal peaks. Inspection of the observed signals of the photosensor shows that those features very little with time and are unique to each meter and meter sensor. Those features can potentially be used in a meter phase estimation algorithm.
It is common practice in the industry to only use the extreme values of the observed signal for a tracking algorithm. Although this works in most cases, there are a number of practical limitations to this approach. With a peak detection algorithm, the number of revolutions cannot accurately be counted when there is no high contrast area on the meter, or when there are multiple peaks in the signal that are equal in amplitude. This means that the sensor only works for a limited range of consumption meters. A test on a set of over 500 different consumption meters encountered in the Netherlands has shown more than 15 % of the meter types proved not compatible with a peak detection algorithm[2].
Secondly, the peak detection algorithm only uses one specific feature of the signal, and therefore the resolution is limited to an estimation of the number of revolutions. When the resource consumption is very low, the frequency of the meter can drop to a very low level, such that a single revolution can take up to tens of min-

[2]Internal research of Quby B.V., 2012.

utes. This means that the consumption estimate cannot be updated during this time window. As was shown in the previous section, the signal has more features that can be utilised to increase the time resolution of the estimate.
A third problem specific to monitoring electricity meters is that they might turn backwards. When the end user has solar panels installed, the production from the solar panels might result in a negative electricity consumption as seen by the meter. A peak detection algorithm will not be able to distinguish between the backwards and forwards motion, since the algorithm only tracks two states of the signal (a peak is present or no peak is present), and the state pattern will be the same for forward and backwards motion.

The observation that there are many signal features present in the signal not used by the peak detection algorithm, combined with the present issues described in the previous section, lead to the following objectives for a new signal processing algorithm for the photosensor:

Objective

Design an algorithm that can estimate the phase of a rotating object, given the observations of the reflections of that object over time. The algorithm must have the following properties:

1. **Adaptable:** The algorithm must be applicable to a wide range of meters with rotating parts, with as little preconditions as possible.
2. **High resolution:** The algorithm must have an update rate that uses all the features present in the signal and updates the phase accordingly.
3. **Able to detect direction:** The algorithm should detect in which direction the meter is turning.
4. **Computationally efficient:** The algorithm should be able to run in real-time on an embedded microcontroller.

1.4. Scientific contributions

This thesis presents a generic framework for estimating the phase of a signal that has repeating patterns but is not stationary. During the research, the following contributions were made:

- A maximum-likelihood-based phase estimation algorithm is proposed that accurately estimates the state of the consumption meter over time based on the recordings of a reflective photosensor.
- The dynamic programming methods of the dynamic time warping (DTW) algorithm are adapted to solve the maximum-likelihood problem.
- The performance of a number of different time series compression methods is evaluated in the context of the phase estimation algorithm.
- A generic time series phase estimation algorithm is proposed for estimating the phase of nonstationary signals with repeating patterns.

1.5. Thesis outline

In Chapter 2, an outline for the phase estimation algorithm is presented, and commonly used techniques for tracking time series data will be described. In Chapter 3, the contributions of this thesis will be presented. The contributions included an adaptation of existing algorithms to work with streaming time series and different methods to compress the incoming data. In Chapter 4 the performance of the different algorithms will be examined and compared with the state of the art. Chapter 5 will reflect on the results and provide a motivation of the results where possible. Finally, conclusions and suggestions for future work can be found in Chapter 6.

2 Background

The analysis and tracking of time series data has applications in many different fields. The field of time series analysis has been steadily growing over the past decade, and there are numerous applications where a changing frequency over time is tracked, bearing similarity to the problem in this thesis. Similar applications include (amongst others) the tracking of electric signals from the heart (ECG) or brain (EEG) [2, 3], the detection of patterns in financial data [4], the alignment of speech and audio recordings [5] or tracking of music performances [6].

Many extensive reviews on time series data mining have been published in the past five years, detailing the different problems in the field of time series data mining, and commonly used techniques [7–9]. Researchers also assessed the numerical performance of common algorithms on a number of different datasets [10, 11]. Two main problems from the field of time series data mining [9] that are relevant to the work of this thesis are:

1. Similarity measurements: How can the similarity between a pair or group of time series be measured?
2. Representation methods: How can the time series be compressed, whilst maintaining the information of interest?

The first problem, defining similarity measures for time series, is related to the problem of detecting and estimation the phase of a repeating pattern in the recorded signal of the photosensor. The relationship between those two problems is that the signal of the photosensor consists of a repetitive pattern that is repeated at different frequencies, related to the rate of consumption. The second problem is the search for a suitable representation method for time series data. A suitable representation method might help to compress the data and transform it into a domain that provides more efficient estimation. In the context of this thesis, transformation of the data or data compression could potentially help in reducing the computation time of the phase estimation algorithm, without losing too much accuracy. In the rest of this chapter, an overview of the state of the art literature is presented, and a brief explanation of some of the common algorithms is presented.

2.1. Algorithm outline

To structure the literature review (Chapter 2) and the methods section (Chapter 3), a general structure for the meter phase estimation algorithm is provided in this section. Each block in Figure 2.1 will be examined in a separate section. A short motivation is provided in this section, and a more extensive presentation of the motivation and signal model can be found in Chapter 3.

From the example signals presented in Figure 1.3, we see that the signals consist of a repetitive pattern, which will be referred to as the *template*, that changes in frequency based on the rotational speed of the meter.

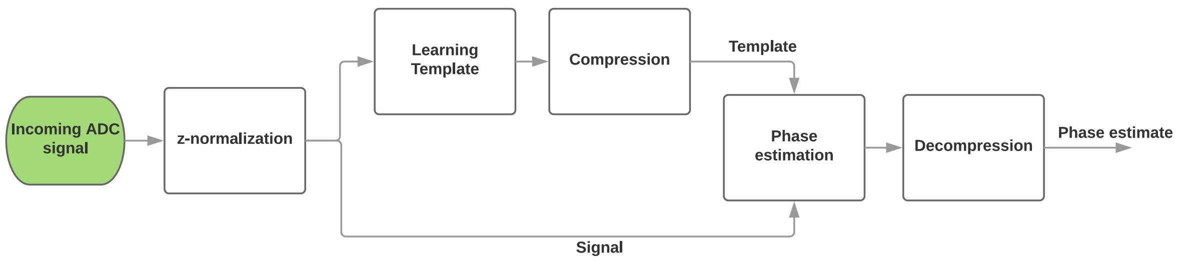


Figure 2.1: The algorithm consists of a learning phase where the template is recorded and an estimation phase where the signal phase is estimated using the template. Both the template and incoming signal can be compressed to reduce complexity.

Therefore, we can interpret the objective as estimating the relative phase of this template signal in the observed signal.
In addition to the repetitive pattern, each signal that was shown in Figure 1.3 has a different vertical offset and a different scale of the template, which can vary over time based on the ambient light. This time-varying difference in scale and offset motivates the introduction of a normalisation step before the actual phase estimation algorithm, so that a representative comparison between the recorded template and the observed signal can be assured.
A schematic view of the workflow of the algorithm is shown in Figure 2.1, and a short explanation of each of the blocks is provided below.

- **ADC signal:** The input of the complete algorithm is a stream of samples from the analogue to digital converter (ADC) connected to the photosensor. The signal is discrete in time and value.
- **Normalisation:** The signal amplitude and mean are expected to vary over time due to changing ambient light conditions. The normalisation step updates the estimate of the signal mean and variance during stationary periods and uses the estimate to normalise the observed signal.
- **Learning the signal template:** The template signal holds the expected reflection from the rotating object for one full rotation. The learning step detects when the signal is stationary, and calculates the template from an average of multiple periods during those segments.
- **Compression:** To reduce algorithmic complexity, the number of samples of the template is reduced by a compression step.
- **Phase estimation:** Given a compressed input signal and a compressed signal template, the phase of the input signal is estimated. The output of the algorithm is the meter position over time.

2.2. Frequency estimation for stationary signals

Estimating the frequency and phase of a signal is a problem that has been studied for a long time. When the assumption can be made that a signal is wide-sense stationary (WSS), the autocorrelation and Fourier transform are very useful to estimate the period of the signal [12, p.13,81-85]. As will be shown in Section 3.1, the observations of the photosensor are not stationary. However, the stationary estimation techniques will be used to learn the signal template during regions of the signal that are locally stationary.

Autocorrelation

When the time series signal is WSS, the autocorrelation is linked to the main period T of the signal as:

$$r_x(T) = r_x(0)\,, \tag{2.1}$$

where r_x is the autocorrelation function of the signal x. Since the autocorrelation also satisfies $r_x(0) \geq r_x(k)$, the period of the signal could theoretically be found by searching for the lag k where Equation (2.1) holds with equality. In practice this relation cannot hold, since the autocorrelation function from an observed signal must be estimated from the samples and is bound to be a mere approximation of the true autocorrelation function. Even though equality will not hold, the autocorrelation will still have a local maximum when $k = T$. Therefore, the autocorrelation can be estimated from the first local maxima after $r_x(0)$ [13, 14]:

$$\hat{T} = \underset{k \geq k_{min}}{\arg\max}\ r_x(k), \tag{2.2}$$

where k_{min} is defined such that the maximum is not part of the first peak. Under some circumstances the autocorrelation provides a good estimate of the period, but it is less effective when the assumptions for a WSS process are violated. When considering a process that is the sum of two WSS processes (for example a periodic and a seasonal effect), the autocorrelation will have peaks at the periods of both processes, but the highest peak in the signal will occur when the two periods collide. An example application of the autocorrelation method is pitch detection in audio signals [15].

Short-time Fourier transform

A closely related technique to the autocorrelation method is the periodogram [12, p. 394]. Similarly, the maximum value of the periodogram can be used to find the main frequency of the observed signal. From the main frequency, the main period can be computed. The advantage of the periodogram is that it will clearly separate the two periods described previously, and show only one peak for each frequency in the signal. When the signal has a shape that is not a sine wave, some harmonics will be present, but generally the main frequency of the signal will give the highest peak. A disadvantage of the periodogram is that when an observation is made with a limited number of samples, the resolution of the frequency estimate can be poor. For improved accuracy and resolution, these two methods can also be combined [16], where a threshold is applied on the periodogram to find the main period, and the autocorrelation is used to improve the resolution of the estimate.

For signals that are not WSS, often the assumption of local stationarity can still be valid. Especially in the field of audio signal processing, signals often are assumed to be stationary within short time windows of approximately 20 ms [17, p.16]. Within those windows, the same techniques as described before can be used to estimate the main period of a signal. Those techniques are called the short time Fourier transform (STFT) and the short time autocorrelation, and have been used for many decades [18].
Since the frequency of the audio signal generally varies slowly with respect to the window length, the periodograms or autocorrelations of the different frames can be combined when estimating the period, using the information of the previous frame to improve the estimate of the current frame. The STFT can be combined with other techniques such as peak filtering method and the so-called *synchrosqueezing transform* [19] to obtain a smooth estimate of the frequency over time. This technique is amongst others used to track heart rates from ECG measurements [3]. One requirement for such an approach is that the signal should approximately be stationary within each window, and that the window must be long enough to detect the main frequency accurately. This is a major drawback when the signal period changes faster than one period of the signal or faster than the window size, because in the latter case the periodicity cannot be detected by the Fourier transform or the autocorrelation. This is the case with the application of this work, since the consumption meter can make a single rotation and the stop again, or can have a frequency that changes within one rotation if the consumption rate is increasing.

2.3. Nonstationary frequency estimation

2.3.1. Dynamic Time Warping

A widespread technique in the field of time series analysis is called *dynamic time warping* (DTW). It was formalised in the late seventies by Sakoe and Chiba [20] and has been broadly adopted since. The goal of DTW is to provide a similarity measure between two time series, which we will call the input signal and the template signal. In many applications, DTW is used as an elastic similarity measure, that can be combined with a nearest-neighbour scheme to find a template from a library of templates that is most similar to the observation. The DTW algorithm provides an alternative to the Euclidean distance measure. When signal

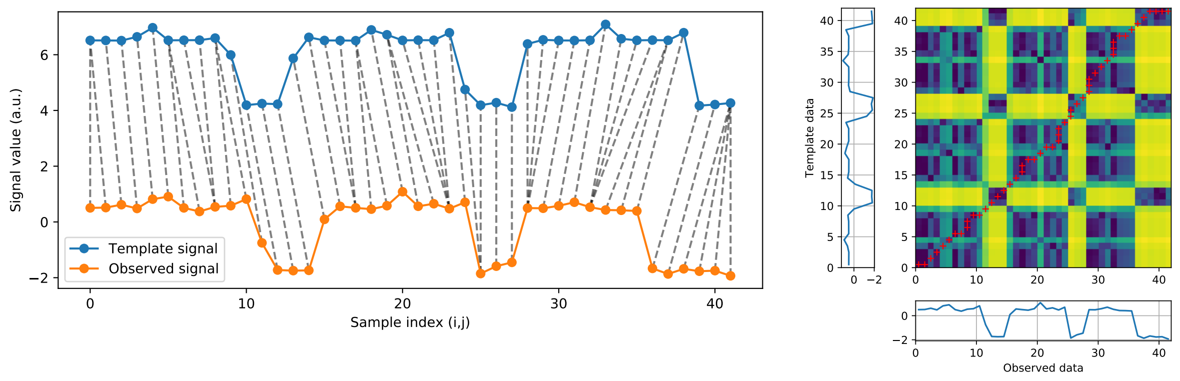


Figure 2.2: On the left the dashed lines represent the optimal alignment or 'warp' between the reference template and the input signal. The matrix to the right shows the cost for all possible warpings where a brighter colour indicates higher cost, and the path that is drawn through it minimises the warping cost and is the optimal warping path. In the left figure, a vertical offset has been added for improved visibility.

features are not synchronous in the template and the observed signal, using the Euclidean distance measure to directly compare the samples of the observations can result in high errors. DTW compensates for temporal misalignment by 'warping' the signals so that the aligned signals have minimal error. This alignment between the two observed signals is of particular interest since it can be used to solve the phase estimation problem, as is shown in Chapter 3.

In this section, a full explanation of the DTW algorithm will be provided, since many of the core principles of the algorithm will be used in the maximum-likelihood based estimator that will be presented in Chapter 3. The DTW algorithm works by considering all samples of the input signal and all samples of the template signal and aligning them such that the difference between the aligned samples of the two signals is minimised. When the signals are of different frequencies, multiple samples of one signal can be linked to a single sample of the other signal. This alignment is called a *warping function* or *warping path* [20].

Let $\boldsymbol{x} = [[x[0], x[1], \dots x[N-1]]$, and $\boldsymbol{y} = [y[0], y[1], \dots, y[M-1]]$ be the respective input signal and template signal. The input and template signal vectors are denoted as $\boldsymbol{x}$ and $\boldsymbol{y}$. A two-dimensional grid of size MN is formed with the elements of the two series along the respective axes. Each node (i, j) on this grid represents a possible alignment between the samples $x[i]$ and $y[j]$, and is associated with a cost $d(i, j)$ that defines an error measure for the two samples. Therefore, this grid is commonly referred to as the *cost matrix*. The warping path is an ordered sequence of K nodes from a starting node (i_0, j_0) to a final node (i_{K-1}, j_{K-1}) [21, p. 482] of the form

$$(i_0, j_0), (i_1, j_1), \dots, (i_k, j_k), \dots (i_{K-1}, j_{K-1}).$$

A visual representation of a warping path is shown on the left of Figure 2.2. Here, two time series are shown that exhibit a similar pattern but deviate locally. For visibility, the signals are plotted with a vertical offset. A dashed line between the signals indicates a warping node (i, j) that matches a sample between the signals $\boldsymbol{x}$ and $\boldsymbol{y}$. Thus, the DTW algorithm corrects for any temporal misalignment by matching each sample $x[i]$ to a sample $y[j]$ based on the nodes (i_k, j_k) of the warping path. Each node in this path is associated with a cost for each node. The cost in a node depends on the values of the samples $x[i]$ and $y[j]$, but will be denoted using only the sample indices i and j for notational convenience:

$$d(i, j) = d(x[i], y[j]), \tag{2.3}$$

where a common choice for the error function $d(i, j)$ is the squared error, or the absolute value.

Example 2.1. *On the right side of Figure 2.2 the cost matrix of the DTW algorithm is shown. Each element of this matrix shows the cost of $d(i, j)$ for the respective node, where a brighter colour indicates a higher cost. For example, observe the low-valued region in both signals around sample index 9-15. This region has a low matching cost when it is matched with the corresponding lower region of the other signal, but has a high cost when it is matched with one of the higher regions in the signal, e.g. the area with indices 19-25. This effect can* *be seen in the cost matrix on the right, where a band-like structure appears the cost matrix at the indices 9-15 corresponding with the lower region. The alignment of $i \in [10-16]$ with $j \in [9-15]$ shows as an area of low cost (darker colours), whereas the area the samples from $i \in [10-16]$ with indices j corresponding with the higher regions show as a high-cost area (brighter colours).*

The goal of the DTW algorithm is to find a contiguous path through the cost matrix that minimises the total cost of the nodes along that path. In the figure, the optimal warping nodes are represented as dashed lines on the left of the figure, and as a set of markers on the right in the cost matrix. Both representations show the same warping path, that aligns the signals $\boldsymbol{x}$ and $\boldsymbol{y}$.

The total cost of a warping path depends on the cost of the nodes in the path and the transitions between succeeding nodes. We will define the transition $(\Delta i_k, \Delta j_k)$ between two nodes in the path as the first-order difference between the elements of the two contiguous nodes:

$$(\Delta i_k, \Delta j_k) = \left(i_k - i_{k-1},\ j_k - j_{k-1}\right).$$

For $k = 0$, we have $\Delta i_0 = 1$ and $\Delta j_0 = 1$. The corresponding transition cost is denoted as $\alpha(\Delta i_k, \Delta j_k)$. The transition cost usually accounts for the number of samples that have been matched between two nodes. The different options for transition costs and motivation behind the choice of transition cost are discussed further on in this section.
The total cost for the warping path is a function of all the nodes in the path, and will be denoted with D. For ease of notation, we will denote the path from the starting (i_0, j_0) to a node (i_k, j_k) as a function of only the final node:

$$D((i_0, j_0)\,(i_1, j_1), \dots (i_k, j_k)) = D(i_k, j_k).$$

The warping cost for the path from (i_0, j_0) to (i_{K-1}, j_{K-1}) is computed as the sum of the cost for each node in the path multiplied by the transition costs:

$$D(i_{K-1}, j_{K-1}) = \sum_{k=0}^{K-1} \alpha(\Delta i_k, \Delta j_k)\, d(i_k, j_k). \tag{2.4}$$

The optimal warping path D^* is the warping path from a node (i_0, j_0) to a node (i_k, j_k) that satisfies the constraints imposed on the path and minimises the total path cost:

$$D^*(i_k, j_k) = \min_{(i_0, j_0)\dots(i_k, j_k)} D(i_k, j_k). \tag{2.5}$$

Common constraints for the warping path will be explained in the next section.

Global and local constraints

In the traditional variant of DTW, a number of constraints are imposed on the warping function, to assure that the warping path is a contiguous path and that the solution of the optimisation problem can be found efficiently. The following constraints are part of the DTW algorithm [20]:

1. Local constraints: The transitions between different nodes in the warping is generally limited by the following constraints:

   (a) Monotonicity: The warping between both samples must be increasing in time:

   $$i_k \geq i_{k-1} \quad \text{and} \quad j_k \geq j_{k-1}. \tag{2.6}$$

   (b) Contiguity condition: To ensure that all samples of both signals are matched, a constraint is placed on the maximum difference between two successive nodes, so that every index i and j occur in at least one of the nodes, and all the samples are matched. This is enforced by the constraint

   $$i_k - i_{k-1} \leq 1 \quad \text{and} \quad j_k - j_{k-1} \leq 1. \tag{2.7}$$

   The two constraints above result in a set of transitions s that is restricted to $(\Delta i_k, \Delta j_k) \in s$, where $s = \{(0,1), (1,0), (1,1)\}$. Other common choices for s are presented in [20] and [22].

2. Boundary conditions: It is assumed that the start and end points in both signals are matching pairs:

$$(i_0, j_0) = (0,0)\,, \qquad \text{(Starting node)} \tag{2.8}$$

$$(i_{K-1}, i_{K-1}) = (M{-}1, N{-}1)\,. \qquad \text{(End-point constraint)} \tag{2.9}$$

3. Maximum allowed warp: To speed up the algorithm, the number of possible warping paths can be reduced by bounding the maximum difference between the index of the signal indices i and j [20] such that

$$|i_k - j_k| \leq r\,. \tag{2.10}$$

where $|\cdot|$ denotes the absolute value operator. This constraint is called the *bounding window*, since it bounds the maximum difference between the indices i and j. A variety of different windows have been proposed. [22], [21, p.495]. Note that this is not a constraint on the transitions or steps between consecutive nodes, but a limit on the cost matrix itself. Coordinates that do not satisfy this constraint are not considered altogether.

Matching cost and transition costs

Both the definition of the cost function $d(i, j)$ and the transition cost can be different depending on implementation. In most cases, a squared error term is used for the cost function:

$$d(i, j) := \left\| x[i] - y[j] \right\|^2\,.$$

Traditionally, the transition weights are based on the number of samples that have been matched. When the time series are aligned to each other using the warping path, each increment of the index i or the index j corresponds with aligning a new time sample. Therefore, the transition weight α depends on the change of indices between consecutive nodes:

$$\alpha(\Delta i_k, \Delta j_k) := \Delta i_k + \Delta j_k\,. \tag{2.11}$$

The latter weighting scheme is known as *symmetric* weighting, since transition weights of a vertical step and a horizontal step combined are equal to a diagonal step, and therefore it is unbiased towards different paths in the cost matrix. The other weighting scheme of Equation (2.13) is not named in literature, but will be referred to as the *unweighted* weighting scheme. When considering all possible paths between two nodes, the unweighted scheme will have lower costs for paths with more diagonal steps.
An alternative for the weighting scheme is *asymmetric weighting* [20]. The asymmetric weighting function is defined as

$$\alpha(\Delta i_k, \Delta j_k) := \Delta i_k\,. \tag{2.12}$$

The reason for the name 'asymmetric' is that it violates the symmetric property of the DTW distance. Normally, given to signals $\boldsymbol{x}$ and $\boldsymbol{y}$, we have

$$DTW(\boldsymbol{x}, \boldsymbol{y}) = DTW(\boldsymbol{y}, \boldsymbol{x})\,,$$

where $DTW(\cdot)$ denotes the minimum path cost for the path with boundary constraints $(i_0, j_0) = (0,0)$ and $(i_{K-1}, j_{K-1}) = (N, M)$. It is easily verified that for the symmetric weighting scheme, the problem does not changed when the signals $\boldsymbol{x}$ and $\boldsymbol{y}$ are swapped. For the asymmetric weighting scheme that is defined above, the cost will however be different.
Finally, in some cases the DTW algorithm is used with unit weights:

$$\alpha(\Delta i_k, \Delta j_k) := 1\,. \tag{2.13}$$

In many papers DTW is formulated with this unweighted scheme [23–26].

Example 2.2. *When the conditions on monotonicity (Equation* (2.6)*) and contiguity (Equation* (2.7)*) described above are applied, the number of possible steps is limited to the following set:*

$$(i_k - i_{k-1}\,,\; j_k - j_{k-1}) = (\Delta i_k, \Delta j_k) \in s = \begin{cases} (1,0)\,, \\ (0,1)\,, \\ (1,1)\,. \end{cases}$$

When the symmetric weighting scheme is applied, the transition weights are as follows:

$$\alpha(\Delta i_k, \Delta j_k) = \begin{cases} 1 & \textit{if } (\Delta i_k, \Delta j_k) = (1,0), \\ 1 & \textit{if } (\Delta i_k, \Delta j_k) = (0,1), \\ 2 & \textit{if } (\Delta i_k, \Delta j_k) = (1,1). \end{cases}$$

The unweighted scheme would have $\alpha = 1$ for each of the steps, including the diagonal step.
Now consider two signals $\boldsymbol{x}$ of length N and $\boldsymbol{y}$ of length M:

$$\boldsymbol{x} = [1,2,8,3,4],$$
$$\boldsymbol{y} = [1,3,4,7,4].$$

From the signal, the cost matrix can be computed using $d(i,j) = (x[i] - y[j])^2$. The cost matrix show the value of $d(i,j)$ for all values of i and j, where the bottom left corner represents $(i,j) = (0,0)$ and the top right corner corresponds with (N,M):

$$d(i,j) = \begin{bmatrix} 9 & 4 & 16 & 1 & 0 \\ 36 & 25 & 1 & 16 & 9 \\ 9 & 4 & 16 & 1 & 0 \\ 4 & 1 & 25 & 0 & 1 \\ 0 & 1 & 49 & 4 & 9 \end{bmatrix}.$$

The next step in the DTW algorithm is to find an optimal warping path from the cost matrix defined above. The next section will explain how this path can be found, and in Example 2.3 the optimal path costs are computed.

Dynamic programming algorithms and Bellman's optimality principle

In the previous section, the concept of a warping path and a cost matrix was introduced, with the warping path as the path that minimises the total warping cost $D(i,j)$ and satisfies the global and local constraints. This section will describe how this optimal warping path can be found in an efficient manner.

Although the number of possible paths through the cost matrix is extremely large[1], the optimal path can be computed without evaluating all possible paths. Let the optimal path from a starting node (i_0, j_0) to any node (i_{K-1}, j_{K-1}) be defined as

$$(i_0, j_0) \xrightarrow{opt.} (i_{K-1}, j_{K-1}). \tag{2.14}$$

Bellman's optimality principle [21, p.484](first described in [28]) states that any subsequence of the optimal path is also an optimal path itself. In other words, when the optimal path from (i_0, j_0) to (i_{K-1}, j_{K-1}) passes through an intermediate node (i,j), the optimal path must be a concatenation of the optimal path from the starting node to the intermediate node, and the optimal path from the intermediate node to the endpoint:

$$(i_0, j_0) \xrightarrow[(i,j)]{opt.} (i_{K-1}, j_{K-1}) = (i_0, j_0) \xrightarrow{opt.} (i,j) \oplus (i,j) \xrightarrow{opt.} (i_{K-1}, j_{K-1}), \tag{2.15}$$

where $\oplus$ denotes the concatenation of the two paths. In the context of the DTW algorithm, this principle can be used to grow the optimal path iteratively. As described previously, the number of possible preceding states is limited by the local constraints. If the optimal path from the starting node to all allowed preceding states is known, Bellman's optimality principle can be used to state the optimal path to the current node is the concatenation of the optimal path from the starting point to one of the predecessors and the path from that predecessor to the current node:

$$D^*(i_k, j_k) = \min_{(i_{k-1}, j_{k-1})} D^*(i_{k-1}, j_{k-1}) + \alpha(\Delta i_k, \Delta j_k) d(i_k, j_k). \tag{2.16}$$

This approach to finding the optimal path is known as *dynamic programming*. Dynamic programming is a technique for problems that can be separated in smaller subproblems, where the result of one problem

[1] In fact, the number of possible paths is roughly proportional to 10^N for a $N \times N$ matrix. For a 100×100 matrix, the number of possible paths is $3.54 \cdot 10^{74}$ [27, p. 81].

is extended iteratively to find the solution to full problem [21, p.484]. Using this recursive definition of the optimal path cost, the optimal path from the origin to any point in the matrix can be found by extending the previously known optimal paths, without having to search all possible paths. The optimal path from the origin to any point in the matrix can then be found in $\mathcal{O}(MN)$ computations. During this procedure, a matrix is constructed that stores all the optimal path costs $D(i, j)$. This matrix is called the *accumulated cost matrix*[29, p.72].

Example 2.3. *Continuing from the signals and the constraints defined in Example 2.2, recall the cost matrix that was found in that example:*

$$d(i,j) = \begin{bmatrix} 9 & 4 & 16 & 1 & 0 \\ 36 & 25 & 1 & 16 & 9 \\ 9 & 4 & 16 & 1 & 0 \\ 4 & 1 & 25 & 0 & 1 \\ 0 & 1 & 49 & 4 & 9 \end{bmatrix}.$$

Now, we will find the optimal path cost from the origin to a point (i, j) *in the matrix using the relation of Equation* (2.16)*. We will initialise the algorithm at node* $(0,0)$*, and set the cost of that node to zero. Next, we realise that for the node* $(1,0)$*, the only allowed predecessor is the node* $(0,0)$*, since the cost function is not defined for the other possible predecessors,* $(0,-1)$ *and* $(1,-1)$*. Therefore, the minimisation of Equation* (2.16) *reduces to*

$$D^*(1,0) = D^*(0,0) + d(1,0) = 0 + 1\,.$$

Similarly, for the node $(0,1)$ *there is also only one allowed predecessor and we find* $D^*(0,1) = 4$*. Now for the node* $(1,1)$*, the set of possible predecessors as defined by monotonicity and contiguity constraints is* $\{(1,0),(0,0),(0,1)\}$*. Evaluating Equation* (2.16) *results in*

$$\begin{aligned} D^*(1,1) &= \min_{(\Delta i_k, \Delta j_k) \in (1,0),(0,0),(0,1)} D^*(i_{k-1}, j_{k-1}) + \alpha(\Delta i_k, \Delta j_k) d(i_k, j_k)\,, \\ &= \min\{(1 + 1\cdot 1),\,(0 + 2\cdot 1),\,(4+1)\}\,, \\ &= 2\,. \end{aligned}$$

Using the obtained results, the optimal cost for to the node $(2,0)$*, can be computed, and then the cost for the node* $(2,1)$*. This process can be repeated the cost to any node in the grid is known. The optimal cost* D^* *for every value* (i, j) *is listed below.*

$$D^*(i,j) = \begin{bmatrix} 58 & 35 & 24 & 10 & 10 \\ 49 & 31 & 8 & 24 & 32 \\ 13 & 6 & 22 & 23 & 23 \\ 4 & 2 & 27 & 27 & 28 \\ 0 & 1 & 50 & 54 & 63 \end{bmatrix}$$

Backtracking

Once the accumulated cost matrix is known, the lowest possible path cost from the starting node to any endpoint can be found directly. The final step of the algorithm is to recover the path that results in the optimal cost from the accumulated cost matrix. This process is called *backtracking* [30]. Equation (2.16) showed the recursive property of the optimal DTW path. This same property can be used to recover the optimal path. The algorithm is initialised with a selected endpoint and operates on the accumulated cost matrix. In each iteration of the algorithm, a new node is added to the optimal path until the starting node is reached. The node that is added to the path is selected from the possible predecessors of the current node. The node from this set with the lowest cost is added to the path, and selected as the new current node. This process is repeated until the starting node is reached. The optimal path links the indices of the template to the indices of the input signal. From this path, the phase estimate of the input signal over time can be found by finding the template index at each time step or index of the input signal. Pseudocode for the backtracking procedure is described in Algorithm 1. Here, the full warping path $((i_0, j_0), (i_1, j_1) \ldots (i_{K-1}, j_{K-1})$ is denoted by $\mathcal{W}$.

Example 2.4. *Continuing from the accumulated cost matrix that was found in Example 2.3, we now apply the backtracking procedure to find the optimal warping path. We initialise the procedure with the endpoint*

constraint, and set $(i_{K-1}, j_{K-1}) = (4,4)$.
Next, we find the predecessor of this node with the lowest cost. In this case the candidates are the nodes $(4,3),(3,3) and (3,4)$. *From those nodes, the node with the lowest cost is node* $(3,4)$ *with cost* 10. *Thus, the node* $(3,4)$ *is added to the list. For this node, the predecessors are* $(2,4),(2,3)$ *and* $(3,3)$. *This time, node* $(3,3)$ *has lowest cost with cost* 8. *This process is continued until the origin is reached, and the complete optimal path is found as*

$$((0,0),(1,1),(1,2),(2,3),(3,4),(4,4)) \ . \tag{2.17}$$

Algorithm 1 backtracking the optimal path from the optimal cost matrix.

```
1: procedure BACKTRACKDTW(endpoint $(i_{K-1}, j_{K-1})$, optimal cost matrix $D^*$)
2:     Initialise $(i,j) = (i_{K-1}, j_{K-1})$.
3:     Initialise $\mathscr{W}^* = (i_{K-1}, j_{K-1})$ .
4:     while $(i,j) \neq (0,0)$ do
5:         Update $(i,j) = \min_{(i,j)}\{D^*(i-1,j), D^*(i,j-1), D^*(i-1,j-1)\}$ .
6:         Append $\mathscr{W}^*$ with $(i,j)$ .
7:     end while
8:     Reverse order of $\mathscr{W}^*$ .
9: end procedure
```

Further literature on dynamic time warping

Because DTW directly matches the values of the signal, it is sensitive to gain and vertical offset errors. A number of variations have been proposed that slightly modify the algorithm. Some of the adaptations include Derivative Dynamic Time Warping [23] where DTW is applied to the derivative of the signal, Cubic Spline Interpolation Time Warping [31], or Two-Dimensional Warping [32], where the template is allowed to be warped both in time and amplitude. Generally, the input and template sequence are assumed to be aligned at the beginning and the end, but the algorithm can also be applied to repetitive signals for streaming data analysis [33] by using a windowing method. To deal with the quadratic time complexity of the algorithm, different windowing techniques (similar to the bounding window in Equation (2.10)) were proposed and compared along with the introduction of the algorithm [20], and more recent research shows that the type of constraint on the warping path also impacts overall performance when DTW is used to search for similar sequences[34].

A paper of particular interest is the work of Makira et al. [14], where they propose a technique called 'self-DTW' to estimate the phase of a quasi-periodic signal. The problem setting is roughly equivalent to that of the work presented in this thesis, where the phase of a repetitive but nonstationary signal is tracked. In the paper, a cost matrix is constructed where the observed signal used both as a template signal and as an input signal. Because the input and template are the same, there is a perfect match along the main diagonal. The path-finding technique used to find a warping path for the off-diagonal warping paths, which match the signal of one period to the same signal in the next period. By first obtaining a rough estimate of the period of the signal, the self-DTW algorithm can be used to find the warping path between different periods of the signal itself. An optimisation problem is then defined that combines the information from all the different inter-period matches, and combines it into one phase estimate for the entire signal. However, the technique has a very high time complexity due to the quadratic programming problem that needs to be solved in addition to multiple iterations of the DTW algorithm. Furthermore, it can only operate on signals that are *quasi-periodic*, i.e. that exhibit periodic behaviour, with a period that varies over time within a certain range.

2.3.2. Particle filters

Particle filtering (PF) or Sequential Monte Carlo (SMC) [35] is a general filtering technique that can be applied to a wide range of models, without requiring linearity of the model. In the filtering model, a number of 'particles' is generated that represent different system states. Next, an estimate for the next state of the particles is derived from the system model. The system output is simulated for all the particles using the system model, and the predicted output of the particles is compared to the observed output. The particles are then weighted by their likelihood given the observed output. After this, new samples are generated based on the weights of the previous samples, with higher weights having a higher probability of being reselected. After this, the next state is computed and the process is repeated.

Since the particles with low likelihood are less likely to be re-sampled on every succeeding iteration of the algorithm, the particles will in most cases converge to the true state of the system. The mean of all particles is used as an estimate for the system state. It is however not always guaranteed that the particles will converge to the optimal solution [36]. In the meter phase estimation problem, the particles could represent different positions and speeds for the consumption meter, and a signal as will be presented in Section 3.1 could be used to predict the output of the states.

2.3.3. Kalman filtering

Kalman filtering is a technique for noisy signals with a known model. The Kalman filter is a minimum mean-squared error (MMSE) estimator that uses the information of a dynamic system model to estimate the model parameters. The estimator is optimal under the assumption of a linear model and Gaussian noise [37, p. 442]. A Kalman filter uses the system model to predict the next state of the system using the statistical model, and combines the estimate of the model with the observed measurement. Given a system state $\boldsymbol{\theta}$ and a transition matrix $\boldsymbol{A}$, the Kalman model is

$$\begin{aligned} \boldsymbol{\theta}[i] &= \boldsymbol{A}\boldsymbol{\theta}[i-1]\,, \\ \boldsymbol{x}[i] &= \boldsymbol{H}\boldsymbol{\theta}[i]\,, \end{aligned} \tag{2.18}$$

where $\boldsymbol{H}$ is the observation matrix that relates the state $\boldsymbol{\theta}$ to an observation $\boldsymbol{x}$.

The extended Kalman filter

For nonlinear models, the extended Kalman filter (EKF)[38, p. 400] was developed. In the EKF, the linear system model of the Kalman filter is extended to a nonlinear model:

$$\boldsymbol{\theta}[i] = f(\boldsymbol{\theta}[i-1])\,, \tag{2.19}$$

$$\boldsymbol{x}[i] = g(\boldsymbol{\theta}[i]) + \boldsymbol{\epsilon}[i]\,, \tag{2.20}$$

where f and g can be any nonlinear function, and replace the matrices $\boldsymbol{A}$ and $\boldsymbol{H}$ of Equation (2.18). The nonlinear model is used for computing the predictions $\hat{\boldsymbol{\theta}}[i|i-1]$ and $\hat{\boldsymbol{x}}[i|i-1]$. The Kalman gain $\boldsymbol{K}$ is computed using first order approximations of the nonlinear functions. The state estimate $\hat{\boldsymbol{\theta}}[i]$ is computed using the approximate Kalman gain that was derived from a linearisation of the model. Because the Kalman gain is now an approximation, the optimality properties are lost. Furthermore, it is difficult to estimate the performance of the algorithm on beforehand since the linearisation depends on the current state of the system[37, p.451]. Another method to adapt the Kalman filter to nonlinear model is the unscented Kalman filter (UKF) [39], which predict the signal means and covariance with higher accuracy than the linearised model.

In the phase estimation problem, the position and frequency of the meter could be modelled by a system state $\boldsymbol{\theta}$, and the observed reflection could be modelled as a nonlinear function $g(\boldsymbol{\theta})$ that maps the state of the consumption meter to the expected reflection. Some recent publications use the EKF and UKF for nonstationary frequency estimation of harmonic signals [40, 41].

2.4. Signal compression

For most of the algorithms presented in the previous section, the computational complexity is high. For the DTW algorithm, the complexity of the algorithm is in the order $\mathcal{O}(MN)$, where M is the number of samples in the template signal and N the number of samples in the observed signal. For the Viterbi algorithm with M states and M transition probabilities, the complexity is as high as $\mathcal{O}(M^2N)$ [42, p. 340]. For those algorithms, the complexity of the algorithm can be reduced by reducing the number of samples in the model M or the samples in the observation N.

One way of reducing the number of samples to process is by decimating the signals, but as we show in Section 4.1 that method might not be the most efficient manner. Therefore, this section will discuss various methods that can be used to compress the model or the observation in an efficient manner.

2.4.1. An overview of compression methods

There are many different representation methods for time series signals. Each representation method compresses the signal in a different way, all with different assumptions on the data model. Informally, many representation methods aim to capture the 'shape' of the signals, or some specific feature of the signal. An extensive comparison table of different representation methods is provided in [7]. Representation methods can be divided in two categories: *data adaptive* methods and *non-data adaptive* methods. Data adaptive methods 'adapt' to the data by minimizing some kind of reconstruction error, usually in an iterative manner. Non-data adaptive methods perform an operation that does not change with respect to the data. An example of this would be computing the first order difference of the data: the operations to compute the derivative are the same, regardless of the data.

A simple and fast method to reduce the number of data points is called Piecewise Aggregate Approximation (PAA), which segments the data into a fixed number of segments, and then computes the average over each segment [43]. Similarly, a piecewise linear approximation technique (PLA) [44] has been proposed that approximates the series with line segments instead of constant segments. In extension to PAA, the piecewise aggregate approximation (APCA) technique with variable-length segments [45] was proposed. In addition to this method, an optimal piecewise segmentation method called Parsimonious Temporal Aggregation (PTA) [46] has been proposed, but at the cost of also having a higher computational complexity. Next to the piecewise linear and aggregate approximations, a number of techniques have been proposed that focus on finding intuitive patterns, such as peaks and troughs [47], shapes [48, 49] or 'perceptually important points'[50].

Other popular techniques are non-data adaptive spectral decompositions such as the discrete Fourier transform [51] or the wavelet transform, which can take multiple types of 'mother waveforms' to convert the signal to a time-frequency representation [52]. The most commonly used waveform for wavelet decompositions is the Haar wavelet, but other waveforms such as the Chebyshev polynomials have also been used for time series analysis [53].

2.4.2. Approximate Piecewise Constant Approximation

The adaptive piecewise constant approximation technique (APCA) is a lossy compression method that approximates the time series with a list of constant segments of variable length. At its core, the compression scheme is based on the Haar wavelet. Because of the unitary property, the energy of the signal is preserved in the wavelet domain. [21, p.375]. In the wavelet domain, only the first M coefficients with the highest energy are considered. Since the wavelet transform is a unitary transform, the squared error introduced in the wavelet domain is the same as the squared error in the time domain. Therefore, when discarding L coefficients, choosing the coefficients with the lowest absolute value will result in the smallest squared error in both time and wavelet domain. When the coefficients are discarded, the signal is converted back to the time domain.
Since some coefficients in the wavelet domain will have overlapping transitions in the time domain and others not, the time domain approximation can have anywhere between M and $3M$ segments. The next step of the algorithm is to merge the segments that give minimal rise in error. After that, the mean of the segments is recomputed. The merge of segments is repeated until a representation with M segments is obtained. After this, the result is translated back to the time domain. Some of the segments are merged until the desired number of segments is obtained. To decrease the error with the original signal, the means of the segments are recomputed. APCA is not an optimal technique, but it can generate good approximations with low computational complexity [45].

2.4.3. Compressed Sensing and Total Variation minimisation

Compressed sensing is a recently developed technique [54] that allows for a near perfect reconstruction of an observed signal on a certain type of signals, even when the signal is sampled at a rate that is below the Nyquist rate. The main assumption for compressed sensing to work, is that the noise-free source signal is sparse when it is transformed to some other domain [55]. A widespread example of Compressed Sensing (CS) is the improvement in MRI imaging time [56] with little to no quality loss. In this example, the imaging time can be reduced by severe undersampling in the so-called *k-space*, thereby reducing the number of measurements needed. When the signal is assumed to be sparse under some transformation, such as under the wavelet transform, a reconstruction can be made by finding a sparse representation in the transformed domain that matches with the observed signal when the inverse transform is applied. This sparse representation can be found using ℓ_1-minimisation.

A subfield of compressed sensing is *total variation minimisation* (TV minimisation). In TV minimisation, it is assumed that the signal can be approximated by a signal with a sparse derivative. [57] This means that the source signal has the property that it is flat for a relatively large part of the time, and sometimes has a non-zero derivative and changes level. Total variation minimisation has been used to de-noise time series that are generated according to such a model, and was shown to preserve the edges where the source signal changed level [58]. The TV minimisation problem can be formulated as

$$\min_{\tilde{\boldsymbol{y}}} \left\| \boldsymbol{y} - \tilde{\boldsymbol{y}} \right\|_2 + \lambda \left\| \nabla \tilde{\boldsymbol{y}} \right\|_1 , \tag{2.21}$$

where $\boldsymbol{y}$ represents the original signal, $\tilde{\boldsymbol{y}}$ the approximation, and $\nabla\tilde{\boldsymbol{y}}$ represents the first order difference of the vector $\tilde{\boldsymbol{y}}$. The first term minimises the squared error between the original template $\boldsymbol{y}$ and the approximation $\tilde{\boldsymbol{y}}$. The second term induces sparsity in the first order difference of the template approximation with an ℓ_1-norm [59, p. 334]. λ is a weighting parameter that determines the relationship between the energy of the derivative and the reconstruction error. Since the ℓ_1-norm induces sparsity, the elements of the first order difference that are close to zero are forced to zero. Because the derivative of the signal is sparse, the signal itself will consist of constant segments. This technique could be used to approximate the signal template with an approximation that consists of constant segments.

2.4.4. Fisher-based sample selection

All of the previous compression methods aimed to find a compressed approximation $\tilde{\boldsymbol{y}}$ that can be reconstructed from fewer samples than the original signal $\boldsymbol{y}$, and that minimises the error $d(\boldsymbol{y}, \tilde{\boldsymbol{y}})$ between the original and the compressed signal.
As an alternative to this approach, a sample selection criterion can be formulated that does not aim to reproduce the original signal, but that aims at retaining the samples of the signal that contain the most information with regards to the estimation task. In [60], a sample selection design technique is presented that selects the samples such that maximises the *Fisher information* of the selected samples. The Fisher information [37, p. 34] is a measure of information that indicates how well a parameter of a stochastic model can be estimated. The Fisher information is denoted as $I(\theta)$. Consider a probability density function $p(X;\theta)$ of a stochastic variable X that depends on the parameter θ. If we wish to make an estimate $\hat{\theta}$ of the true model parameter, there is a direct relationship to the minimal achievable variance of the estimator and the Fisher information:

$$\text{var}(\hat{\theta}) \geq \frac{1}{I(\theta)} . \tag{2.22}$$

This relationship is called the *Cramér-Rao lower bound* (CRLB). The Fisher information is defined as

$$I(\theta) = -E_x \left[\frac{\partial^2 \ln p_X(x \mid \theta)}{\partial \theta^2} \right] . \tag{2.23}$$

Example 2.5. *In some models, the Fisher information is a function of time. Consider the case of a sinusoid in white noise with known frequency but unknown amplitude. When estimating the amplitude of the signal, the amplitude can be estimated with greater accuracy at the peaks of the sinusoid than near the zero-crossings.*

This intuitive reasoning is confirmed when the Fisher information of this model is computed. Consider the model

$$x[i] = A\sin(\omega i) + \epsilon[i], \qquad \epsilon \sim \mathcal{N}(0,\sigma). \tag{2.24}$$

When we compute the Fisher information for the parameter A, we find that it is indeed maximised at the peak of the sinusoid:

$$I(i;A) = \frac{\sin(\omega i)^2}{\sigma^2}. \tag{2.25}$$

In the work of Swärd et al. [60], the principle of Example 2.5 is used to optimally select the samples in time (or in some other sampling space) that minimise the CRLB of the estimator. Consider a model with a vector of parameters $\boldsymbol{\theta}$ and a vector with observations in time $\boldsymbol{x}$. When there are multiple parameters, the *Fisher information matrix* (FIM) $\boldsymbol{I}$ is defined as

$$[\boldsymbol{I}(i;\boldsymbol{\theta})]_{kl} = -E_x\left[\frac{\partial^2 p_X(x \mid \boldsymbol{\theta})}{\partial\theta_k\partial\theta_l}\right]. \tag{2.26}$$

The Cramér-Rao lower bound for a single parameter is found on the diagonal of the inverse of the Fisher information matrix:

$$\mathrm{var}(\hat{\theta}_k) \geq [\boldsymbol{I}(i;\boldsymbol{\theta})^{-1}]_{kk}. \tag{2.27}$$

Using this, the sample selection problem can be formulated as

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \mathrm{tr}\left(\left(\sum_{i=0}^{N-1} w[i]\boldsymbol{I}(i,\boldsymbol{\theta})\right)^{-1}\right), \\ s.t. \quad & \|\boldsymbol{w}\|_1 \leq \lambda, \\ & w[i] \in \{0,1\}, \quad i = 0,1,\dots N-1. \end{aligned} \tag{2.28}$$

where $\boldsymbol{w}$ is a vector that selects which samples are used, and tr$(\cdot)$ denotes the trace operator. The ℓ_1 norm in the constraint of the optimisation problem limits the number of samples that can be selected by the binary selection vector w. Assuming independence of the samples over time, the total Fisher information is found as the sum of the individual observations. Combining this information with the definition from Equation (2.27) and given the fact that the trace operator sums all diagonal elements, we can interpret the objective of Equation (2.28) as the sum of the CRLBs for each estimator $\hat{\theta}_k$, given the sample selection provided by $\boldsymbol{w}$. The goal of the optimisation problem is to select $\boldsymbol{w}$ such that the variance of the estimators is minimised.

A downside of this approach is that since the Fisher information often depends on the parameters themselves, an estimate of the parameters is needed of the parameters. Using the estimate, the samples selection procedure is run. When the parameters are close to the original estimate, the selected samples will still be near the minimum of the new CRLB. A second difficulty is that the optimisation procedure has a high computational complexity.

3 Phase estimation algorithm

The main algorithm for estimating the phase of the consumption meter over time will be presented in this chapter. First, a model is presented that relates the observed signal from the photosensor to the phase of the meter. After that, the maximum likelihood estimator is derived from the model, and it is shown that the dynamic programming techniques of the DTW algorithm can be used to efficiently compute the maximum likelihood estimate.
Next, different compression methods are applied to the signals to reduce the complexity of the algorithm. In Chapter 4, the performance of the algorithm and of the different compression methods is evaluated.

3.1. Statistical model

A number of different observed signals from the photosensor were shown in Figure 1.2 of the introduction. From the figures, it becomes clear that there is a repetitive pattern in the signal that correlates with the state of the consumption meter. In this section, a signal model is introduced, that is used to motivate the choice of algorithms in the rest of the chapter.

When the meter rotates, a difference in reflected light is observed because the rotating part of the meter has varying reflective properties for different positions. The reflection observed by the photosensor forms a time series that contains encoded information about the position of the rotating part. The variations in the reflective properties are caused by markings on the rotating part of the meter or by irregularities on the surface. We will call this observation $\boldsymbol{x}$, and presented a model for how the position or phase of the rotating part θ relates to the measured reflection in this section. Furthermore, the outline for the algorithm that tracks the phase of the rotating part is presented, with the objective to use the phase estimate to estimate the state of the consumption meter.
One important assumption behind the model is that the consumption meter has static reflective properties. This means that each element of the object will give the same reflection pattern each time that it passes by the sensor, and that the reflective properties do not change over time. Thus, the true reflection of the object can be assumed to be a deterministic function of the object angle θ, which can be approximated by a template vector $\boldsymbol{y}$. Figure 3.1 shows a typical template signal for an electricity meter. Here, the reflection template $\boldsymbol{y}$ is shown as a function of the object phase θ. Once this template signal is known, the values of the template can be used to estimate the state of the meter. In addition to the position of the meter, the observed light of the sensor is also influenced by the ambient light and the light emitted by the LED of the photosensor itself, which will be modelled by a gain factor a and can have a vertical offset b. Finally, it is assumed that the measurement is contaminated by zero-mean i.i.d. Gaussian noise $\epsilon[i]$ with standard deviation σ. For each time instance i, an the unfiltered observation $x_{raw}[i]$ can now be modelled as follows:

$$x_{raw}[i] = ay[\theta[i]] + b + \epsilon[i], \quad \epsilon \in \mathcal{N}(0,\sigma). \tag{3.1}$$

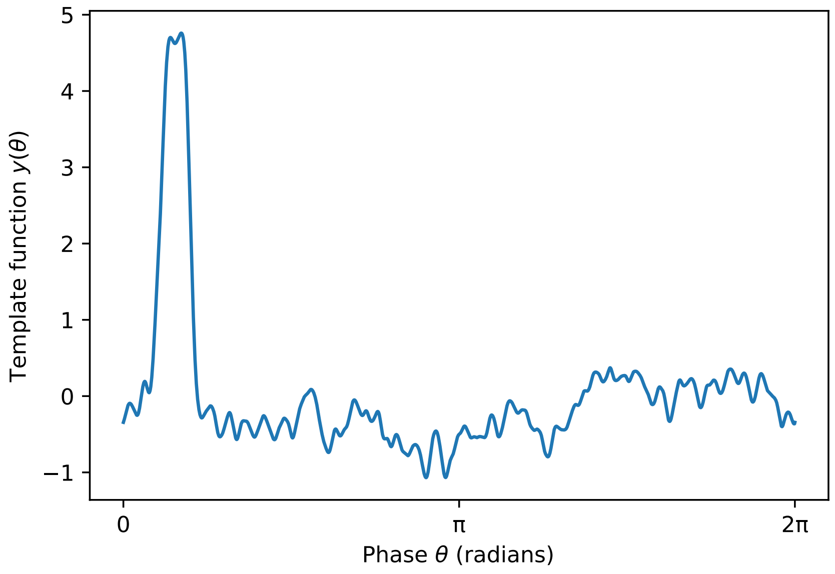


Figure 3.1: The observed reflections by the sensor for different values of θ.

The first step in the algorithm outline is normalisation of the signal. Normalisation of the signal is a nontrivial process, since estimate a and b, $y[\theta[i]]$ must be known. since the gain a and offset b can vary with time as the lighting conditions change. In Section 3.2, a normalisation procedure of the signal is discussed, although the main work of this thesis will focus on the estimation of the phase from the normalised signal rather than on the normalisation procedure. After normalisation, the observation model reduces to

$$x[i] = \frac{x_{raw}[i] - b}{a} = y[\theta[i]] + \epsilon[i], \quad \epsilon \in \mathcal{N}(0, \sigma). \tag{3.2}$$

Figure 3.2 visually demonstrate this model, and shows how the observation $\boldsymbol{x}$ is related to the latent variable $\boldsymbol{\theta}$.

In the rest of this chapter, the algorithms that aim at estimating the phase $\boldsymbol{\theta}$ will assume the normalised observation model of Equation (3.2). Using this observation model, the probability density function of the observations depends on the phase $\theta[i]$ and is defined as

$$\mathrm{p}_X(x \mid \theta[i]) \sim \mathcal{N}(y[\theta[i]], \sigma), \tag{3.3}$$

$$\mathrm{p}_X(x \mid \theta[i]) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{\left(x - y[\theta[i]]\right)^2}{2\sigma^2}\right), \tag{3.4}$$

with corresponding log likelihood

$$\ln \mathrm{p}_X(x \mid \theta[i]) = \ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{\left(x - y[\theta[i]]\right)^2}{2\sigma^2}, \tag{3.5}$$

where X is a stochastic variable, and x is a realisation of that variable.

3.2. Signal normalisation

To compensate for possible errors due to the gain a and offset b as described in Section 3.1, a normalisation procedure is applied before the phase estimation algorithm. When a and b are estimated, the transformation of Equation (3.2) can be used to make the signal zero mean and unit variance. Without loss of generality, we can assume the signal template $\boldsymbol{y}$ is zero mean and unit variance. Using this assumption, the gain and offset

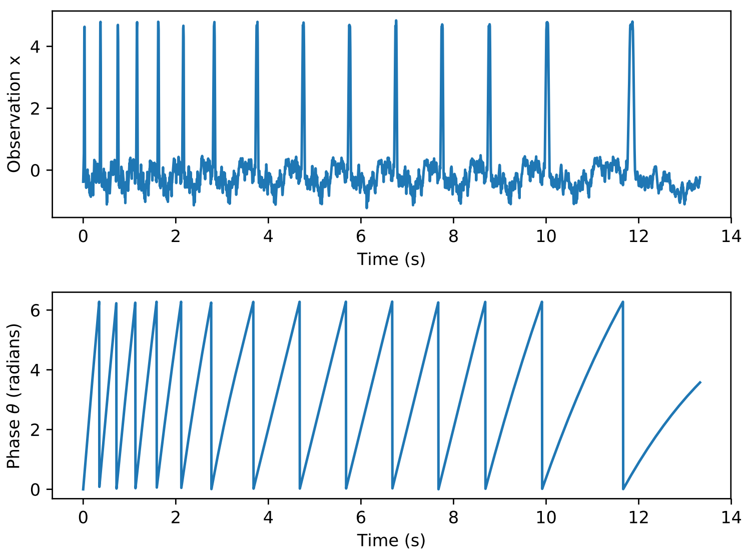


Figure 3.2: An illustration of the signal model of Equation (3.2) using the template signal of Figure 3.1. On the top, the noisy observation $\boldsymbol{x}$ is shown, while the bottom plot shows the latent phase vector $\boldsymbol{\theta}$.

parameters can be estimated as the mean and variance of the signal during stationary periods of the signal. Using the autocorrelation-based technique to detect stationary as described in the next section, a segment of the observation can be extracted that contains an integer number of periods of revolutions, and has an approximately constant speed. Over this segment, a and b can be estimated as

$$\hat{a} = \sqrt{\mathrm{var}(\boldsymbol{x}_{raw} - \hat{b}) - \sigma^2}, \quad \hat{b} = \mathrm{mean}(\boldsymbol{x}_{raw}), \tag{3.6}$$

where mean() and var() denote the sample mean and sample standard deviation respectively. Although the estimator for a is not the MVU estimator, the estimator for $\hat{a}$ is consistent, and the estimator for $\hat{b}$ is unbiased under the assumption that the contribution of $\boldsymbol{y}[\boldsymbol{\theta}]$ is zero mean and unit variance over the sampled segment. The MVU estimator for a and b is derived in Appendix A, but cannot be used in practice since it relies on $\boldsymbol{y}[\boldsymbol{\theta}]$ to be known over the estimated segment.

3.3. Template estimation

3.3.1. Properties of the pattern template

In order to estimate the phase $\boldsymbol{\theta}$ of the meter, an estimate of the reflective function $\boldsymbol{y}$ is needed, which we will approximate with the template vector $\boldsymbol{y}$. Inspection of different readings from the photosensor showed that the signal template varied based on the type and model of the meter that it is attached to, but also based on the placement of the sensor and age of the meter. Figure 1.3 showed some examples of recordings for an electricity meter, a water meter and a gas meter. For this reason, the template needs to be estimated as part of the algorithm, and it is not possible to use a predefined template based on the meter type.

One important observation from the signals in Figure 3.1 is that template $\boldsymbol{y}$ is almost always *surjective* when viewed as a function of θ: there will be angles θ_m and θ_n such that $y[\theta_m] = y[\theta_n]$. This implies that when a certain reflection $x = y[\theta]$ is observed, there will be ambiguity in the estimate of x, since there will be multiple θ that satisfy $x = y[\theta]$. Therefore, even in the noise-free case, the phase cannot uniquely be recovered from a single observation. To summarise, the following assumptions are made about $y[\theta]$:

- The template $\boldsymbol{y}$ is different for every set-up and must be estimated from the signal itself in a learning period.

- The template function is surjective, and therefore θ cannot uniquely be determined from a single observation.

In order to obtain an estimate, it is assumed that there will be moments throughout the day where the observed signal is locally stationary, i.e. the derivative of θ is constant for multiple periods. With a measure for signal stationarity we can then estimate the signal period using techniques for stationary frequency estimation, and extract an average period of the observed signal which will be used as the estimated template.

3.3.2. Detecting periodicity

The autocorrelation function is a measure for how well the signal correlates with a shifted version of itself. It can be used to estimate the fundamental period of a stationary signal, but can also be used to measure whether a signal exhibits periodic behaviour [13, 14]. A consistent estimator for autocovariance function c_x of the normalized signal $\boldsymbol{x}$ is [61, p.189]

$$c_x[k] = \frac{1}{N-k} \sum_{i=0}^{N-1-k} x[i]x[i+k]. \tag{3.7}$$

where k is the lag between the template and the shifted template. The autocorrelation is defined as the autocovariance normalised by the signal energy:

$$r_x[k] = c_x[k] \;/\; c_x[0]. \tag{3.8}$$

here the fact that $c_x[0]$ represents the signal energy is used. After the normalisation step, we have $r_x[0] = 1$. We will use the property that the autocorrelation will only be 1 when the shifted signal matches the original signal. When the signal is too much contaminated by noise or when it is nonstationary, the normalised autocorrelation will be less than one. In this way, a threshold can be set to detect when the signal is approximately periodic. The algorithm will continuously compute the autocorrelation over a fixed window and check if the first peak after $r_x[0]$ exceeds the threshold. If this is the case, the signal is assumed locally stationary and the template can be extracted from the estimation window.

3.3.3. Estimating the template

When the aforementioned threshold is exceeded, the argument of the maximum value of the autocorrelation can be used as an estimate for the period of the signal:

$$\hat{T} = \underset{k \geq k_{min}}{\arg\max} \quad r_x[k]. \tag{3.9}$$

And the estimate for the reflective function is found by averaging over multiple periods of the signal:

$$\hat{y}[i] = \left\lfloor \frac{N}{\hat{T}} \right\rfloor \sum_{i=0}^{\lfloor N/\hat{T} \rfloor} x[n + i\hat{T}]. \tag{3.10}$$

Here $\lfloor \cdot \rfloor$ denotes the floor operator, and N is the number of samples that is used to estimate the signal template.

3.4. Nonstationary phase estimation

Using the statistical model from the first section of this chapter, we can restate the phase estimation problem for the normalised signal model as

Phase estimation problem

Given a signal template $\boldsymbol{y} = [y[0], y[1], \ldots, y[M-1]]$ and a series of observations $\boldsymbol{x} = [x[0], x[1], \ldots, x[N-1]]$ contaminated by measurement noise $\epsilon[i] \sim \mathcal{N}(0, \sigma)$ and generated by the model

$$x[i] = y[\theta[i]] + \epsilon[i] \,, \tag{3.11}$$

find an estimate of the phase vector $\boldsymbol{\theta} = [\theta[0], \theta[1], \ldots, \theta[N-1]]$ with $\theta[i] \in (0, 1, \ldots M-1)$.

In this section, a constrained maximum likelihood estimator (MLE) is proposed that solves the phase estimation problem. After that, the equivalence between the ML estimator and the DTW algorithm is shown, and an adapted version of the DTW algorithm is used to solve the MLE problem.

3.4.1. Maximum likelihood estimation for a single sample

A general estimator for stochastic models is the maximum likelihood estimator (MLE) [37, p. 157]. It is defined as the estimator that maximises the likelihood of the model parameters given an observation of the model. In our case, this means that the probability density function that was derived previously should be maximised with respect to $\boldsymbol{\theta}$. Using the concavity of the log function, this maximizing the likelihood is equivalent to maximizing the log-likelihood. The estimate for a single sample $x[i]$ can be found using the model of Equation (3.2):

$$\hat{\theta}_{MLE} \quad = \quad \max_{\theta} \ln p_X(x \mid \theta) \quad = \quad \max_{\theta} \left[-\frac{1}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} (x - y[\theta])^2 \right] . \tag{3.12}$$

The first maximisation term consists of a term that depends only on σ^2, and σ is assumed to be constant. The second term is a product of σ^2 and a term dependent on θ, and therefore it is equivalent to *minimise* over the squared error between the observed sample and the function $y(\theta[i])$:

$$\hat{\theta}_{MLE}[i] = \quad \min_{\theta} \left(x[i] - y[\theta] \right)^2 , \tag{3.13}$$

which is the least squares estimator. Thus, when the phase is estimated from a single sample, the most likely phase θ is the phase where the template $y[\theta]$ yields a reflection that is the most similar to the observed reflection. Since the reflection is usually not a one-to-one function, there might be multiple θ that have a high likelihood, or low squared error.
To illustrate this, consider the signal in Figure 3.1. Here, an observed reflection with value $x = 5$ only matches the template $y[\theta]$ near the high peak, and will result in an unambiguous estimate of the phase θ. For different values of the observed reflection however, there might be multiple candidates for the phase θ. Concluding, a single sample will not allow us to estimate the phase without ambiguity.

Maximum likelihood estimation for vector observations

Even though we cannot accurately estimate the phase from a single sample, the samples can be combined to obtain a more accurate estimate of the phase over time. When a vector of observations $\boldsymbol{x} = [x[0], x[1], \ldots, x[N-1]]$ is made, the model can be extended to estimate a vector $\boldsymbol{\theta} = [\theta[0], \theta[1], \ldots, \theta[N-1]]$ that contains the phase estimates over time. Since the measurement noise is assumed to be uncorrelated, it can be assumed that the observations $x[i]$ are independent, and that the joint pdf is the product of the individual pdfs:

$$p_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\theta}) = \prod_{i=0}^{N-1} p_X(x[i] \mid \theta[i]) \,. \tag{3.14}$$

When computing the maximum log-likelihood estimate for the vector $\boldsymbol{\theta}$, similar steps can be taken as for the case with a single observations, and the maximum likelihood estimator for $\boldsymbol{\theta}$ can be found as:

$$\hat{\boldsymbol{\theta}}_{MLE} = \quad \min_{\boldsymbol{\theta}} \sum_{i=0}^{N-1} \left(x[i] - y[\theta[i]] \right)^2 . \tag{3.15}$$

As can be seen from the equation above, for each time step i one value of $\theta[i]$ minimises the error with one sample from the observation $x[i]$. This result implies that the most straightforward way to estimate the phase over time $\boldsymbol{\theta}$ is to estimate $\theta[i]$ for each sample $x[i]$ independently. Although this might be the most simple approach, this approach does not take into account the fact that the $\theta[i]$ over time are correlated. Therefore, the next step in finding an estimator for $\boldsymbol{\theta}$ is to define an assumption about how different $\theta[i]$ are correlated.

3.4.2. Proposed method

Since the measurements are based on a physical model, we know that the meter cannot rotate faster than the maximum consumption rate for a household. A limit on the maximum frequency of the meter implies a limited slope for the phase as a function over time. Therefore, we will constrain our maximum likelihood estimator by limiting the difference between two consecutive phase estimates $\theta[i]$ to be smaller than some value c, dependent on the maximum known speed. For the simplicity of the algorithm, we will set this constant to one, and control the constraint by the number of samples in the template and the sampling frequency. Formally, the constrained maximum likelihood estimator for the phase $\boldsymbol{\theta}$ can now be formulated as

$$\begin{aligned} \hat{\boldsymbol{\theta}}_{MLE} = \quad & \min_{\boldsymbol{\theta}} \quad \sum_{i=0}^{N-1} \left(x[i] - y[\theta[i]]\right)^2 , \\ & \text{s.t.} \quad -1 \leq \theta[i] - \theta[i-1] \leq 1 . \end{aligned} \tag{3.16}$$

By setting the maximum slope of the estimate to one, the maximum meter frequency that can be tracked equals the sampling frequency divided by the number of samples in the template. When it is known that the consumption meter rotates only in on direction, the constraint can be adjusted accordingly, and the formulation becomes

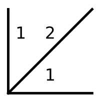
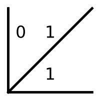
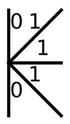
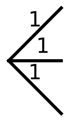
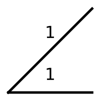
$$\begin{aligned} \hat{\boldsymbol{\theta}}_{MLE} = \quad & \min_{\boldsymbol{\theta}} \quad \sum_{i=0}^{N-1} \left(x[i] - y[\theta[i]]\right)^2 , \\ & \text{s.t.} \quad 0 \leq \theta[i] - \theta[i-1] \leq 1 . \end{aligned} \tag{3.17}$$

In the next section, the relationship between the costs of the DTW warping path and the squared error of the *alignment costs* obtained from the algorithm is shown. This result will be used in Section 3.4.4 to show how the DTW algorithm can be used to solve the MLE problem, and how for different constraints an equivalent DTW and MLE can be found. An overview of the relationship between the constraints of the DTW problems and the constraints of the MLE problems is shown in Table 3.1. Pseudocode for the proposed algorithm is provided in Algorithm 2

Algorithm 2 The algorithm that finds the ML estimate described in Equation (3.17). The algorithm is an adapted form of dynamic time warping and uses the recursive update step from Equation (2.16). Further adaptations with regards to the traditional DTW algorithm are described in Sections 3.4.4 and 3.4.5.

```
1: procedure PHASE ESTIMATION($\boldsymbol{x}$, N, $\boldsymbol{y}$, M)
2:     Initialise $D^*(0,j) = d(0,j) \quad \forall j \in (0,1,\dots,M-1)$
3:     for $i$ in $1,2,\dots N-1$ do
4:         for $j$ in $0,1,\dots M-1$ do
5:             $s = \{(j+q) \mod M : q \in (-1,0,1)\}$ \\Find the possible predecessors of $j$
6:             $D^*(i,j) = \min_{D(i-1,r)} \{D(i-1,r) : r \in s\} + d(i,j)$
7:         end for
8:     end for
9:     Find the optimal endpoint $j^*_{K-1} = \arg\min_j \{D(N-1,j) : j \in (0,1,\dots M-1)\}$ \\Find predecessor with
   minimal cost
10:    $\mathcal{W}^* = $ BACKTRACE($(N-1, j^*_{K-1})$, $D^*$)
11:    $\theta_x = \min_j \{j \in (0,1,\dots,M-1) : (i,j) \in \mathcal{W}\}$
12: end procedure
```

Table 3.1: This table shows the relationships between the constraints and objective of the MLE and the constraints on the DTW algorithm.

MLE problem	**DTW constraints**	**Visual constraints**	**Comments**
$\min_{\boldsymbol{\theta}_x,\boldsymbol{\theta}_y} \; d(\boldsymbol{x}[\boldsymbol{\theta}_y],\boldsymbol{y}) + d(\boldsymbol{x},\boldsymbol{y}[\boldsymbol{\theta}_x])$ s.t. $\theta_x[i] \geq \theta_x[i-1]$, $\theta_y[j] \geq \theta_y[j-1]$	$\alpha(\Delta i_k,\Delta j_k) = \Delta i_k + \Delta j_k$ $0 \leq \Delta i_k \leq 1$ $0 \leq \Delta j_k \leq 1$		Symmetric DTW [20]
$\min_{\boldsymbol{\theta}_x} \; d(\boldsymbol{x},\boldsymbol{y}[\boldsymbol{\theta}_x])$ s.t. $\theta_x[i] \geq \theta_x[i-1]$	$\alpha(\Delta i_k,\Delta j_k) = \Delta i_k$ $0 \leq \Delta i_k \leq 1$ $0 \leq \Delta j_k \leq 1$		Asymmetric DTW [20]
$\min_{\boldsymbol{\theta}_x} \; d(\boldsymbol{x},\boldsymbol{y}[\boldsymbol{\theta}_x])$	$\alpha(\Delta i_k,\Delta j_k) = \Delta i_k$ $\Delta i_k = 1$		Unconstrained MLE
$\min_{\boldsymbol{\theta}_x} \; d(\boldsymbol{x},\boldsymbol{y}[\boldsymbol{\theta}_x])$ s.t. $-1 \leq \theta_x[i] - \theta_x[i-1] \leq 1$	$\alpha(\Delta i_k,\Delta j_k) = \Delta i_k$ $\Delta i_k = 1$ $-1 \leq \Delta j_k \leq 1$		Proposed algorithm
$\min_{\boldsymbol{\theta}_x} \; d(\boldsymbol{x},\boldsymbol{y}[\boldsymbol{\theta}_x])$ s.t. $\theta_x[i] - \theta_x[i-1] \leq 1$	$\alpha(\Delta i_k,\Delta j_k) = \Delta i_k$ $\Delta i_k = 1$ $\Delta j_k \leq 1$		Proposed algorithm, forward rotation only

3.4.3. Relationship between the dynamic time warping algorithm and the alignment costs

Moving away from the maximum likelihood estimator for a moment, a proof is given in this section that shows that there is a direct relationship between the error of the alignment of two signals and the path cost of the DTW algorithm. The alignment that will be presented in Equation (3.21) closely resemble the cost the the MLE minimises of Equation (3.16). Later on, this proof is used to adopt the DTW algorithm to the MLE problem.
The original goal of the DTW algorithm as described in Section 2.3.1 is to find a measure of similarity for two time series that are not synchronous in time. The similarity is measured by finding a warping path that maps indices of one signal to indices of the other signal. This mapping can in turn be used to align the signals in time. This section will show that there is a direct relationship between the squared error of the aligned signals and the cost of the warping path. An example of such an alignment is shown in Figure 3.3. In fact, it will be shown that for some DTW problems, an equivalent maximum-likelihood problem can be formulated, and that under the right constraints a form of the DTW algorithm can be used to find the ML estimator for θ as formulated in Equation (3.16). Table 3.1 shows the relationships between the MLE problem definitions and the DTW constraints.

To show this, we will first introduce some terminology to aid the proof of equivalence. Recall from Section 2.3.1 that the dynamic time warping algorithm (DTW) aligns the samples of two signals $x[i]$, $i = 1,2,\ldots,N$, and $y[j]$, $j = 1,2,\ldots,M$ by searching for a *warping path*. The warping path is defined as a list of nodes $(i_0, j_0),\ldots(i_{K-1}, j_{K-1}) = \mathscr{W}$ of length K, where each node is a pair of indices (i, j). The total cost of a warping path is defined as

$$D(i_{K-1}, j_{K-1}) := \sum_{k=0}^{K-1} \alpha(\Delta i_k,\Delta j_k)\, d(i_k, j_k)\,, \tag{2.4 revisited}$$

where $\alpha(\Delta i_k,\Delta j_k)$ and $d(i_k, j_k)$ represent the transitions cost and warping cost as defined in Section 2.3.1. Next to this, we have $\Delta i_k = i_k - i_{k-1}$ and $\Delta j_k = j_k - j_{k-1}$. The objective of DTW is to find the optimal path $\mathscr{W}^*$ from the starting node (i_0, j_0) to the endpoint (i_{K-1}, j_{K-1}) that minimises the warping cost.
Since the warping path consists of index pairs that matches the indices i with the indices j, we can define a *phase function* that finds a corresponding j for each i, and vice versa. The phase function that aligns j to i

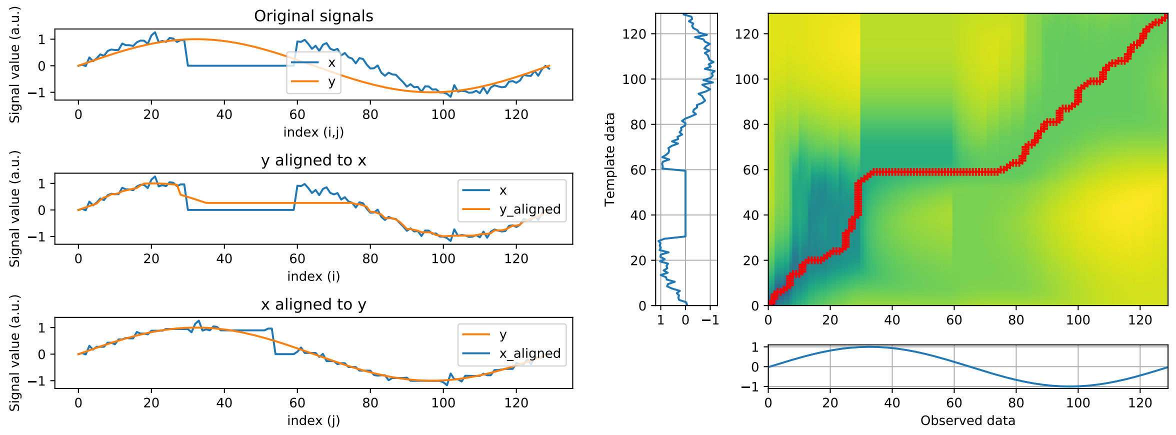


Figure 3.3: On the top left, two signals $\boldsymbol{x}$ and $\boldsymbol{y}$ are shown. Below the signals, the alignment of $\boldsymbol{x}$ and $\tilde{\boldsymbol{y}}$ and the alignment $\boldsymbol{y}$ and $\tilde{\boldsymbol{x}}$ are shown. Both alignments are created using the phase functions that were derived from the warping path $\mathscr{W}^*$ and accumulated cost matrix D^*, as shown on the right.

will be referred to as θ_x, and the function that aligns i to j is referred to as θ_y. This phase function θ_x can then be used to align the samples $y[j]$ to the samples $x[i]$. An example of such an alignment is shown in Figure 3.3. We define the phase functions with respect to x and with respect to y as

$$\theta_x[i] := \min_j \{ j \in (1, \ldots, M-1) \colon (i, j) \in \mathscr{W} \}, \tag{3.18}$$

$$\theta_y[j] := \min_i \{ i \in (1, \ldots, N-1) \colon (i, j) \in \mathscr{W} \}. \tag{3.19}$$

Using this phase function, one signal can be aligned to the other signal by indexing the signal with the phase function. This will be denoted as

$$\boldsymbol{y}[\boldsymbol{\theta}_x] = \left[y[\theta_x[0]], y[\theta_x[1]], \ldots y[\theta_x[N-1]] \right]. \tag{3.20}$$

Now we define the error between the aligned signals as

$$d(\boldsymbol{x}, \boldsymbol{y}[\boldsymbol{\theta}_x]) = \sum_{i=0}^{N-1} (x[i] - y[\theta_x[i]])^2, \tag{3.21}$$

and

$$d(\boldsymbol{x}[\boldsymbol{\theta}_y], \boldsymbol{y}) = \sum_{j=0}^{M-1} (x[\theta_y[j]] - y[j])^2.$$

Given the definition of the warping path cost from Equation (2.4) and the error of the aligned signal in Equation (3.21), we will now show that under the right constraints, both costs are in fact equal.

Theorem 3.1. *Under the constraints*

$$0 \leq \Delta i_k \leq 1, \quad 0 \leq \Delta j_k \leq 1, \tag{3.22}$$

$$\alpha(\Delta i_k, \Delta j_k) := \Delta i_k + \Delta j_k. \tag{3.23}$$

the total costs over all nodes of the warping path as defined by Equation (2.4) *are equal to the error between the alignments of the signals* $\boldsymbol{x}$ *and* $\boldsymbol{y}$*:*

$$D(i_k, j_k) = d(\boldsymbol{x}, \boldsymbol{y}[\boldsymbol{\theta}_x]) + d(\boldsymbol{x}[\boldsymbol{\theta}_y], \boldsymbol{y}), \tag{3.24}$$

$$\sum_{k=0}^{K-1} \alpha(\Delta i_k, \Delta j_k)\, d(i_k, j_k) = \sum_{i=0}^{N-1} (x[i] - y[\theta_x[i]])^2 + \sum_{j=0}^{M-1} (x[\theta_y[j]] - y[j])^2, \tag{3.25}$$

for any warping path that satisfies the constraints, and thus also for the optimal warping path.

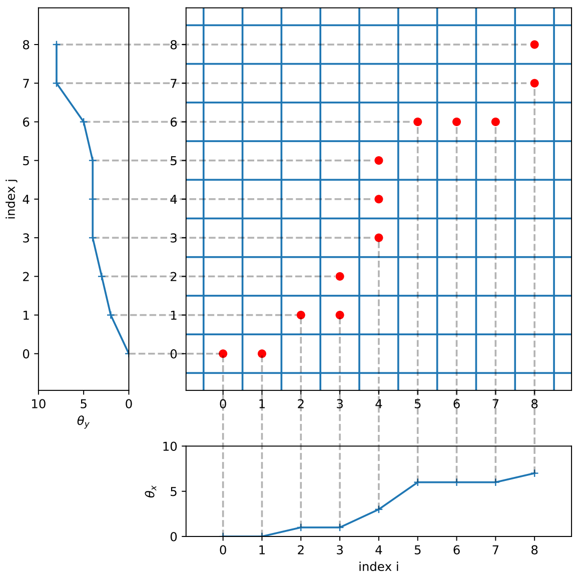


Figure 3.4: A visual representation of the relationship between the warping path and the phase function as described in Equation (3.18). The dashed lines indicate how every first node that occurs at an index i or j corresponds with an element of the phase function.

Proof. Starting from the definition of the warping cost

$$D(i_{K-1}, j_{K-1}) = \sum_{k=0}^{K-1} d(i_k, j_k)\alpha(\Delta i_k, \Delta j_k)\,, \tag{3.26}$$

we will substitute the above definition of the transition cost $\alpha(\Delta i_k, \Delta j_k)$.

$$D(i_{K-1}, j_{K-1}) = \sum_{k=0}^{K-1} d(i_k, j_k)(\Delta i_k + \Delta j_k)\,, \tag{3.27}$$

$$= \sum_{k=0}^{K-1} d(i_k, j_k)\Delta i_k + \sum_{k=0}^{K-1} d(i_k, j_k)\Delta j_k\,. \tag{3.28}$$

Now, the first summation over all nodes K can be rewritten as a summation over all columns $p = 1 \ldots N-1$ and all the nodes that are part of that column, i.e. that satisfy $i_k = p$. For the second term, the summation can be rewritten as a summation over the rows q.

$$D(i_{K-1}, j_{K-1}) = \sum_{p=0}^{N-1} \sum_{(p, j_k)\in\mathcal{W}} d(p, j_k)\Delta i_k + \sum_{q=0}^{M-1} \sum_{(i_k, q)\in\mathcal{W}} d(i_k, q)\Delta j_k\,. \tag{3.29}$$

Note that if $(p, j_k), (p, j'_k) \in \mathcal{W}$ for some $j'_k > j_k$, then $(p, j'_k) \in \mathcal{W}$, so $\Delta i_k = p - p = 0$. This implies that the only contribution in $\sum_{(p, j_k)\in\mathcal{W}}$ comes from the first node in the path that satisfies $i_k = p$. Now recall the definition of the phase function from Equation (3.18), and consider the constraint $j_k > j_{k-1}$. It follows that the only node that contributes to the summation has the value of j_k specified by $\theta_x[i_k]$, and we can write the summation as

$$D(i_{K-1}, j_{K-1}) = \sum_{p=0}^{N-1} d(p, \theta_x(p)) + \sum_{q=0}^{M-1} d(\theta_y(q), q)\,, \tag{3.30}$$

$$= d(\boldsymbol{x}, \boldsymbol{y}[\boldsymbol{\theta}_x]) + d(\boldsymbol{x}[\boldsymbol{\theta}_y], \boldsymbol{y})\,. \tag{3.31}$$

From this it follows that under the constraints provided above, the cost of the total warping path cost $D(i, j)$ is always equal to the corresponding alignment costs $d(\boldsymbol{x}, \boldsymbol{y}[\boldsymbol{\theta}_x])$ between the signals and $d(\boldsymbol{x}[\boldsymbol{\theta}_y], \boldsymbol{y})$. From this it follows that equality must also hold for the optimal case. The optimal warping path $\mathcal{W}^*$ that is the solution to

$$D^*(i_{K-1}, j_{K-1}) = \min_{\mathcal{W}} D(\mathcal{W}) = D(\mathcal{W}^*)\,, \tag{3.32}$$

also minimises the sum of the cost of aligning the signal $\boldsymbol{x}$ to $\boldsymbol{y}$ and aligning the signal $\boldsymbol{y}$ to $\boldsymbol{x}$.

$$\min_{\mathscr{W}} d(\boldsymbol{x}, \boldsymbol{y}[\boldsymbol{\theta}_x]) + d(\boldsymbol{x}[\boldsymbol{\theta}_y], \boldsymbol{y}) = d(\boldsymbol{x}, \boldsymbol{y}[\boldsymbol{\theta}_x]) + d(\boldsymbol{x}[\boldsymbol{\theta}_y], \boldsymbol{y}) \mid_{\mathscr{W}=\mathscr{W}^*} . \tag{3.33}$$

□

A visual illustration of this theorem is provided in Figure 3.4. In the figure, a warping path is shown. To the left and at the bottom of the matrix, the phase functions θ_x and θ_y are shown. The phase functions can be seen of a projection of the first node encountered in the warping path for each i and j respectively. A dashed line indicates a connection between the phase function and the warping path. From the figure, it can be seen that for a diagonal step, two lines connect to the node, indicating a contribution to both phase functions and thus to both alignment costs. When there is a horizontal or vertical step, only one dashed line connects to that node, and this step contributes to only one of the phase functions. Thus, we can say that diagonal steps contribute to two phase functions and thus two both terms of Equation (3.21), while orthogonal steps contribute to only one of the terms. Defining orthogonal steps as unit weight and diagonal steps as double cost corresponds with the definition of the transition cost $\alpha(\Delta i_k, \Delta j_k)$ in the theorem. The relationship between the transition costs and the contributions to the phase functions partly motivates the theorem.

The result of Theorem 3.1 is that under the right constraints, solving the DTW problem is equivalent to finding the optimal alignment. When DTW is used with a different weighting scheme, such as the unweighted scheme as described in 2.13, this analogy no longer holds. In many applications, it is desirable to use the symmetric weighting schemes, since most applications use DTW to cluster groups of time series, or to find a reference sequence that is similar to the recorded sequence. In such cases, it is a logical step to optimise for a symmetric cost function such as the one defined in 3.33.
For the meter tracking problem of this thesis, a symmetric warping scheme is questionable. The goal of the phase estimation algorithm is to estimate the phase of the observed signal $\boldsymbol{x}$ with respect to some template vector $\boldsymbol{y}$. This is analogous to aligning the template to the observed signal. The other alignment is however less relevant to the meter phase estimation algorithm. In the next section, this analogy is further explored, and it is shown that using a asymmetric weighting scheme, DTW alignment is analogous to the maximum likelihood estimator.

3.4.4. Solving the maximum likelihood problem using DTW

Now that the relationship between the alignment costs and the step size for the DTW algorithm is shown, it will be shown that the asymmetric DTW algorithm bears great resemblance with the maximum likelihood estimator for $\boldsymbol{\theta}$ as defined in Equation (3.16). The asymmetric DTW algorithm [20] uses the following constraints:

$$\alpha(\Delta i_k, \Delta j_k) := \Delta i_k ,$$
$$0 \le \Delta i_k \le 1 , \quad 0 < \Delta j_k \le 1 .$$

Following the same procedure as in the proof of Theorem 3.1, the summation over all nodes can be expressed as a summation over the indices:

$$D(i_{K-1}, j_{K-1}) = \sum_{k=0}^{K-1} d(i_k, j_k)(\Delta i_k) ,$$

Again, we rewrite the summation over all nodes to a summation over all rows and the nodes that are in that row:

$$D(i_{K-1}, j_{K-1}) = \sum_{p=0}^{N-1} \sum_{(p, j_k) \in \mathscr{W}} d(p, j_k) \Delta i_k . \tag{3.34}$$

The difference with the symmetric case is that since $\alpha(\Delta i_k, \Delta j_k)$ was defined to depend only on Δi_k and not on Δj_k, the summation is no longer split into two terms, but only one term remains. Following the same reasoning as with the symmetric case, we find that the only values that have $\Delta i_k \neq 0$ are the values of (i, j) described by the phase function with respect to i:

$$\begin{aligned} D(i_{K-1}, j_{K-1}) &= \sum_{p=0}^{N-1} d(p, \theta_x(p)), \\ &= \sum_{p=0}^{N-1} d(x[p], y[\theta_x[p]]), \\ &= d(\boldsymbol{x}, \boldsymbol{y}[\boldsymbol{\theta}_x]). \end{aligned} \tag{3.35}$$

Upon closer inspection, we find that this derivation shows that the optimal warping path is the path minimises the error between the samples $x[i]$ and the matched samples $y[\theta_x[i]]$. This problem formulation carries great resemblance with the maximum likelihood estimator for the phase over time as derived in Equation (3.15). The question that needs to addressed, is whether the phase function θ_x derived in this section is the same function as the maximum-likelihood estimator, θ_{MLE}. In the problem formulation of Equation (3.15), the phase estimation is completely unconstrained: at a time index i, any template sample $y[\theta_x[j]]$ can be selected such that $d(x[i], y[\theta[i]])$ is minimised. For the solution of the DTW algorithm, the constraint $0 \leq \Delta j_k \leq 1$, along with the definition of the phase function, result in the implicit constraint on the phase function. This is shown in the second entry of Table 3.1.

$$\theta_x[i_k] \geq \theta_x[i_{k-1}]. \tag{3.36}$$

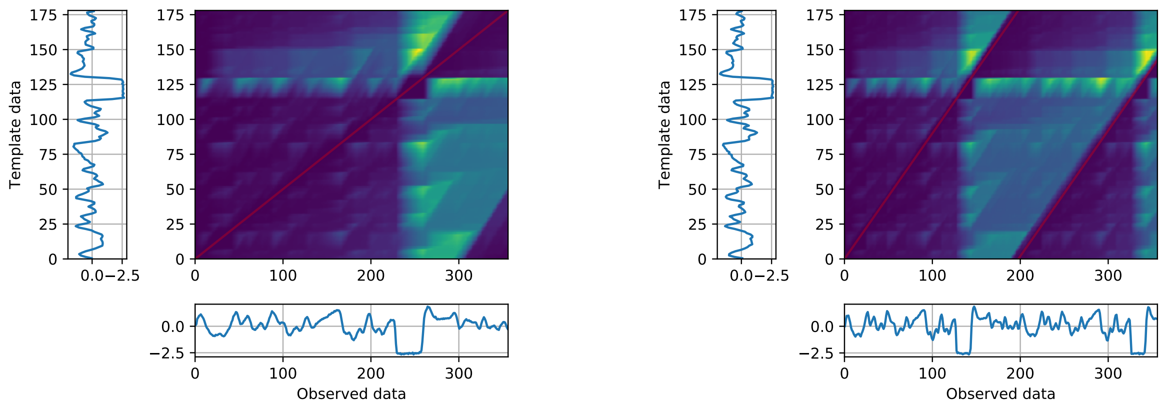
This constraint arises from the DTW problem because of the constraints on the warping path. The constraint that both i and j should be increasing in the path implies that for higher values of i in the path, the value of j must be equal or higher too. Combining this with the definition of the phase function of Equation (3.18) leads to the implicit constraint on the phase function.
This implicit constraint can be avoided by dropping the constraint on Δj_k and setting the constraint on Δi_k as $\Delta i_k = 1$. Under those constraints, the DTW algorithm produces the same result as the unconstrained maximum likelihood estimator. In order to make the problem similar to the constraint estimator of Equation (3.16), we can set the constraint $-1 \leq \Delta j_k \leq 1$.
Concluding, we can say that for most formulations of the DTW algorithm, an equivalent constrained MLE can be formulated, and the other way around. An overview of the different problem formulations is shown in Table 3.1.

3.4.5. Further adaptations to the DTW algorithm

The DTW algorithm is designed to work with two signals with aligned starting and end points. However, in our case, the observed signal can be of any length, and since the meter will continuously rotate and repeat the template pattern, there is no fixed endpoint. Fortunately, the DTW algorithm computes the optimal path cost $D^*(i, j)$ for *any* point (i, j), as was described in Section 2.3.1. In this section, it is described how the information from the accumulated cost matrix D^* can be used to find the most likely endpoint. Furthermore, a windowing technique is presented to adapt the DTW algorithm for real-time phase estimation.

Endpoint constraints

Traditionally, the DTW algorithm is used for aligning two time series that may be of different length, but are still assumed to be aligned at the start and of the end of the signal. For example, in a speech recognition application, the DTW algorithm is used to determine the similarity between a recorded vowel and a sample from a library of known vowels. The vowel starts the beginning of the observation and ends with the final sample, and therefore it is a reasonable assumption that the starting points and end points of the signals are aligned.
For the meter phase estimation problem, it is not known in which state the consumption meter is at the end of the observation $x[N-1]$. Therefore, any value of θ is a candidate endpoint. Now recall that the accumulated cost matrix $D^*(i, j)$ holds the lowest achievable cost for the warping path to any index pair (i, j). This means that for the final time index i_{K-1}, the lowest possible costs for all j can be found from the accumulated cost matrix. If we recall the relationship between the path cost and the maximum likelihood estimator from



(a) accumulated cost matrix D^*, input and template aligned at end of signal.

(b) accumulated cost matrix D^*, input ends in the middle of the template.

Figure 3.5: The accumulated cost matrices for different speed input signals. The red line indicates the phase estimate. On the left, a case is shown where the endpoints of the observations are aligned. On the right, the adaptations for the endpoints are shown, connecting the first and last sample of the template and allowing the estimation to end in any state.

Equation (3.35), it can be seen that selecting the endpoint with the lowest warping cost is equivalent to the alignment with the lowest squared error, which is in turn equivalent to selecting the most likely endpoint. Therefore, we can find the most likely endpoint j_{K-1} as the endpoint that minimises the cost D^*:

$$j_{K-1} = \underset{D^*(N-1,j)}{\arg\min} \{D^*(N-1,j) : j \in (0,1,\ldots,M-1)\}. \tag{3.37}$$

A visual demonstration of this technique is shown in Figure 3.5, where one observation is shown that is aligned at the end of the signal, and another observation that is not aligned at the end of the signal. By selecting the most likely endpoint and applying the backtracking procedure of Algorithm 1 from there, the optimal warping path is found. The optimal warping path is indicated in the figure with a red line.

Starting point

In traditional DTW, the algorithm is initialised by letting $D^*(0,0) = d(0,0)$. Since we have no indication of the starting state of the meter, the algorithm will be initialised with

$$D^*(0,j) = d(0,j) \quad \forall j \in (0,1,\ldots,M-1). \tag{3.38}$$

By backtracking the optimal path from the most likely endpoint (i_{K-1}, j_{K-1}), the optimal warping path $\mathscr{W}^*$ and the phase function $\boldsymbol{\theta}_x$ can be determined.

Repeating the template

To be able to track multiple periods of the signal, the phase estimation algorithm should be able to start at the beginning of the template when the end has been reached. For this reason, we will connect the first state of the template $y[0]$ to the last state of the template $y[M-1]$, so that the first state is reachable from the last state and vice versa.

Real-time phase estimation

In order to apply the phase estimation algorithm to a stream of observations, further adaptations to the algorithm are required. In principle, the algorithm described in the previous section can be applied to two time series of any length. However, in a real-time environment it is not feasible to apply the signal to the entire history of the signal.

To avoid this, an approach is proposed in this section that segments the observations into frames with length W and hop size $h \leq W$. For each window of the observation, an estimate of the phase can be made by applying the DTW algorithm to the input window and the template signal. The estimated phase for each window

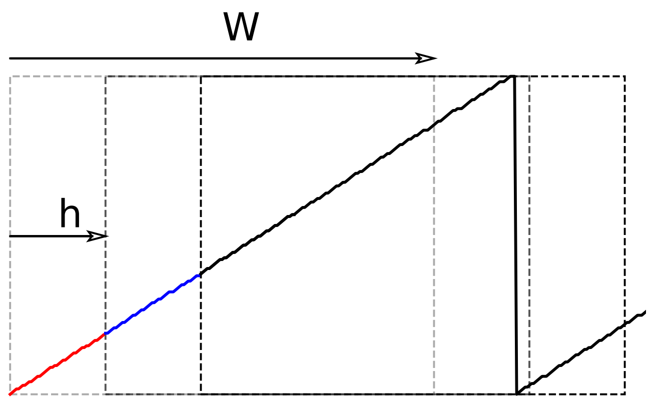


Figure 3.6: A visual illustration of the windowing technique described in Algorithm 3. At each iteration, the window is shifted slightly and the backtracking step is performed over the window of length W, and h new elements are added to the phase estimate

can then be concatenated again, so that a contiguous phase estimate is obtained. This is done by first estimating the minimal warping cost D^* for the first W samples. After this, the backtracking step can be applied to find the phase estimate θ_x. Now the phase estimate for the first h samples is used. After the backtracking step, the matrix D^* can be grown until $W+h^{\text{th}}$ sample. This is illustrated in Figure 3.6. Next, the backtracking step is applied again and the phase estimate is updated until the $2h^{\text{th}}$ sample. To prevent the cost from becoming too high, the minimum value of D^* can be subtracted from all elements after the backtracking step. An overview of this procedure is presented in Algorithm 3. Because the phase is estimated over windows of data, there is a delay between the observation and the phase estimate for that observation. This delay is equal to the length of the window divided by the sampling rate.

Algorithm 3 An windowing technique for continuously estimating the phase of an incoming signal.

```
1:  procedure STREAMING PHASE ESTIMATION ALGORITHM($\boldsymbol{x}$, $\boldsymbol{y}$)
2:      Initialise $D^*(0, j) = d(0, j) \quad \forall j \in (0, 1, \dots, M-1)$
3:      Compute first $W$ rows of $D^*$ using the method from Algorithm 2.
4:      while there are samples left in $\boldsymbol{x}$ do
5:          Find $\theta_x$ using the backtracking procedure.
6:          Add first $h$ elements from $\theta_x$ to the phase estimate.
7:          Discard the first $h$ rows of $D^*$.
8:          Subtract minimum value of $D^*$.
9:          Compute $h$ new rows of $D^*$ using the method from Algorithm 2.
10:     end while
11: end procedure
```

3.4.6. Phase estimation using compressed template signals

The phase estimation algorithm presented previously finds the most likely phase function $\boldsymbol{\theta}$ given the observed signal $\boldsymbol{x}$ and a template $\boldsymbol{y}$. To do this, it considers the squared error of the observed signal with every sample of the template signal. Therefore, the algorithmic complexity scales with the number of observations N per time unit and also with the number of samples M in the template. The complexity of the algorithm is therefore proportional to $\mathcal{O}(MN)$.

To reduce the complexity of the algorithm, either M or N should be decreased. In the next section, a number of nonuniform downsampling techniques are applied to reduce the number of samples M of the template.

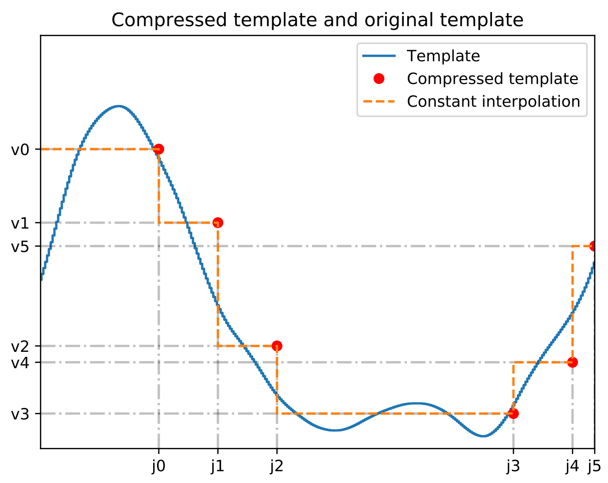


Figure 3.7: Visual representation of a strongly compressed template signal, showing the numbered index-value pairs. The dashed line shows a constant interpolation of the compressed signal.

A nonuniform downsampling scheme for the observed signal is not considered, since compressing the input signal generally is a more computationally intensive problem than applying the phase estimation algorithm to all samples of the observation. The template signal only needs to be compressed once when a new template is learned.

Now consider a template signal $\boldsymbol{y}$. Let $\tilde{\boldsymbol{y}}$ be an approximation of $\boldsymbol{y}$ consisting of L time-value pairs as $((j_0, v_0), (j_1, v_1), \dots (j_{L-1}, v_{L-1}))$, where i denotes the corresponding index of the original template and v the corresponding value for that index. We will denote the vector that holds all the values as $\tilde{\boldsymbol{y}}_v$, and the vector with all the indices as $\tilde{\boldsymbol{y}}_j$. An example of a compressed signal is shown in Figure 3.7. The approximation of the template is not unique, and the next section will discuss different ways of finding such an approximation.

When applying the phase estimation algorithm to a compressed signal, the values of the compressed template $\tilde{\boldsymbol{y}}_v$ can be used as the new template for estimation. The algorithm will then produce a compressed estimate of the phase $\tilde{\boldsymbol{\theta}}_x$, that represents an alignment between the compressed template $\tilde{\boldsymbol{y}}$ and the observed signal $\boldsymbol{x}$. The indices $\tilde{\boldsymbol{y}}_j$ of the compressed signal can in turn be used to obtain an uncompressed phase estimate $\hat{\boldsymbol{\theta}}_x$. This process is summarised in Algorithm 4.

Algorithm 4 Compressed phase estimation procedure.

```
1: procedure CompressedPhaseTracking(observation $\boldsymbol{x}$, template $\boldsymbol{y}$)
2:     $(\tilde{\boldsymbol{y}}_j, \tilde{\boldsymbol{y}}_v) = \text{Compress}(\boldsymbol{y})$
3:     $\tilde{\boldsymbol{\theta}}_x = \text{Track phase}(\boldsymbol{x}, \tilde{\boldsymbol{y}})$
4:     $\hat{\boldsymbol{\theta}} = \text{Uncompress}(\tilde{\boldsymbol{\theta}}_{\boldsymbol{x}}, \tilde{\boldsymbol{y}}_j)$
5: end procedure
```

In the next section, different methods for determining a compressed estimate are explored. Since there are fewer samples in the compressed template, the phase estimate will be less detailed, and might also be more sensitive to errors. Furthermore, the slope constraint that was formulated might be violated, since two adjacent samples of the compressed template can have a bigger time step between them. In Chapter 4, the effect of the compression ratio on the performance of the algorithm is investigated.

3.5. Converting the phase estimate to a consumption rate

Once the estimate of the phase of the meter has been determined using the algorithm described in the previous sections, the estimate needs to be converted to a consumed quantity and corresponding consumption rate. All household electricity meters have a *C-value*. This C-value is a ratio that relates the rotation of the meter to the amount of energy was consumed. A typical value for C is 600 revolutions per kWh, but can be anywhere in the range from 75 to 750. For gas meters and water meters, a similar ratio can be determined that relates the revolutions of the meter to the gas or water usage in litres.

Using this value, the absolute phase that was estimated by the meter module can directly be translated to a consumed quantity. To estimate the consumption rate, the first order difference of the signal can be computed. Since the estimate of the phase can have some local deviation and noise, the derivative of the signal was smoothed over time.

3.6. Compression methods

In this section, a number of different compression methods are presented. For each of the compression schemes a short explanation is provided, and some motivation why it was selected.

Four different methods to approximate the template signal with a lower number of samples are considered. First, the total variation (TV) minimisation and APCA technique are both commonly used to approximate time series with constant segments. The third method is based on the Fisher information of the signal that was derived from the signal model. As a last reference method, a uniform downsampling of the template is considered. The performance of the compression methods will be judged in this section based on the squared error with the original templates, and in Section 4.1 the performance of the compression methods in combination with the phase estimation algorithm is evaluated. The reason that the squared error was chosen as a metric, is that the phase estimation algorithm works by minimizing the error between the aligned template and the observed signal, with the squared error as an error measure. Therefore, it is likely that templates with a lower squared error with the original template also perform better in the phase estimation algorithm.

A visual comparison of the compression methods is shown in Figure 3.8. In the figure, example signals of an electricity meter, a gas meter and a water meter are compressed using the APCA method, TV minimisation, Fisher information sample selection and downsampling. It can be seen from the figure that the APCA method replicates the shape of the signal well. Table 3.2 shows the RMS error that corresponds with the compressed templates shown in Figure 3.8. From the table, it can be seen that the APCA method has a lower reconstruction error than the downsampled signal. The TV minimisation has a higher error than APCA or the downsampled signal, and the Fisher information also has a relatively high RMS. Here it must be noted that for the Fisher information, the only selection criterion was to select the samples, and not to optimise for the reconstruction error. In Section 4.1 it will be shown that there is a relation between the RMS error of the approximation and the performance of the algorithm.

3.6.1. Total variation minimisation and APCA

The TV minimisation approach tries to find a representation of the signal that minimises the squared error between the template and the approximation, and at the same time minimises the number of segments in the approximation. When the number of segments is reduced, the template can be reproduced by only the values and timestamps of those segments, and the complexity of the algorithm is reduced. With the APCA method, a similar method is used where each of the segments is represented only by its value. Both methods have a representation that segments the signal into a desired number of segments, but also aim at keeping the error with the original template as low as possible.

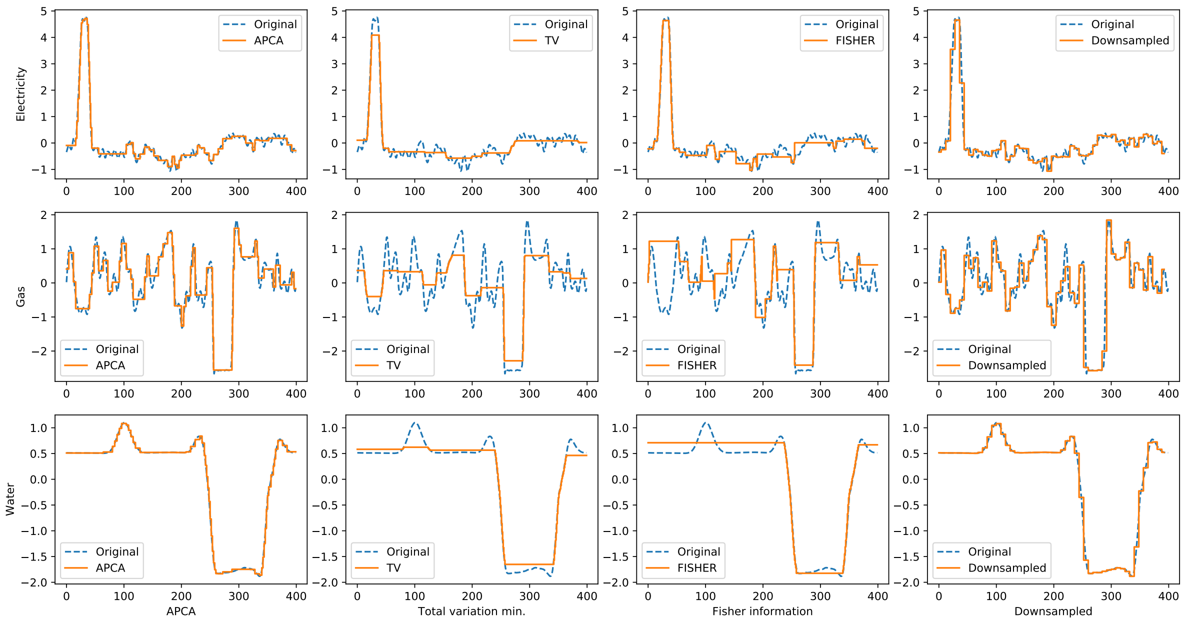


Figure 3.8: Different compression methods result in different approximations of the original template. For three different signals, an approximation is shown using the compression techniques is shown with a constant interpolation of the segments. In the figures, the original template is shown with 400 samples, and the approximations were formed with 50 samples.

3.6.2. Fisher information maximisation

In the text below, the Fisher information method as explained in Section 2.4.4 is applied to the signal model that was found earlier. The Fisher information method selects a number of samples that have the highest local information about the parameter. The timestamps and values of the samples with the highest Fisher information are then used as the new template values for the phase estimation algorithm.

In the work of Swärd et al. [60], an optimal sample selection scheme is derived for a multi-parameter model. In Section 2.4.4, it was explained how the sum of the Cramér-Rao lower bounds of the estimators can be minimised. The CRLBs are found on the diagonal of the inverse of the Fisher information matrix. The optimisation problem selects the samples such that the Fisher information matrix of the selected samples results in the lowest total CRLB. From the phase estimation algorithm, recall the model that was assumed for the observations:

$$\ln \mathrm{p}(X|\theta[i]) = \ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) - \frac{\left(x - y(\theta[i])\right)^2}{2\sigma^2}, \tag{3.5 revisited}$$

For this model, we can use the equations of Section 2.4.4 to compute the Fisher information. The resulting Fisher information is

$$I[y(\theta[i])] = \frac{1}{\sigma^2}\frac{\partial^2 y(\theta[i])}{\partial\theta[i]^2}. \tag{3.39}$$

A similar result is also presented in [37, p.36]. From this result we can conclude that the Fisher information is proportional with the square of the derivative of the template function. Reflecting on this result, we can see that the parts of a template that have the highest derivative indeed provide the greatest change in x when a small deviation in the phase θ occurs. Because of this property, any noise on the observation $x[i]$ will result in minimal local error in $\theta[i]$ where the derivative of $y[\theta]$ is the highest. The main difference between the approach in this thesis and the work of Swärd et al. [60] is that in their work a model with a small number of parameters is used with respect to the number of observed samples. In one of the examples, five parameters of a noisy sinusoid are estimated using 50 samples. In our model, an observation of 50 samples is described by a phase function also consisting of 50 samples.

Table 3.2: RMS error for different combinations of compression methods and meter types

	APCA	**TV**	**Fisher**	**Downsampling**
Electricity	0.16	0.23	0.18	0.30
Gas	0.25	0.43	0.70	0.42
Water	0.05	0.15	0.15	0.09

3.6.3. Uniform downsampling

Another method that should be considered is downsampling the template. Downsampling of the template has the benefit that is trivial to implement, and retains the even spacing over time of the template. First, a low-pass filter is applied to the signal to prevent aliasing effects after downsampling, and then the signal can be downsampled. There are a number of downsides to the downsampling technique. First, some steep transitions might not be preserved by the low-pass filter. Furthermore, when a template has constant segments (for example the water meter example in Figure 1.3), the downsampled signal will have the same resolution for the constant segments with low detail as for parts of the signal with high detail. In other words, since the downsampling technique is non-data-adaptive, it will have the same sampling density for areas with high detail as for areas with low detail.

4 Results

In this chapter, the performance of the proposed algorithm is evaluated through a number of different experiments. The algorithm is compared to a number of conventional frequency tracking algorithms. The algorithm will be evaluated on three main experiments. First, the performance of the phase estimation algorithm is evaluated as a function of the number of samples in the compressed template. Secondly, the performance of the algorithm is compared with three different reference algorithms. Finally, the performance of the algorithm under different signal-to-noise ratios (SNRs) is evaluated.

The performance of the algorithm will be evaluated on two different outputs of the algorithm. First, the estimate of the consumption rate. The consumption rate corresponds with the frequency of the consumption meter, and the root-mean-square (RMS) of the error between the estimate and the actual rate is used as a metric for the performance. For electricity meters, this value is converted to Watts, and for water and gas this value is converted to an error in the flow rate in litres/hour. The second measure of performance is the total accumulated error in the total consumed quantity, which corresponds with the number of revolutions the meter has made over the total window. For electricity, this value is converted to kWh, and for gas and water this value is converted to litres. A linear interpolation of the phase and frequency estimates was made to compute the local deviation and RMSE.

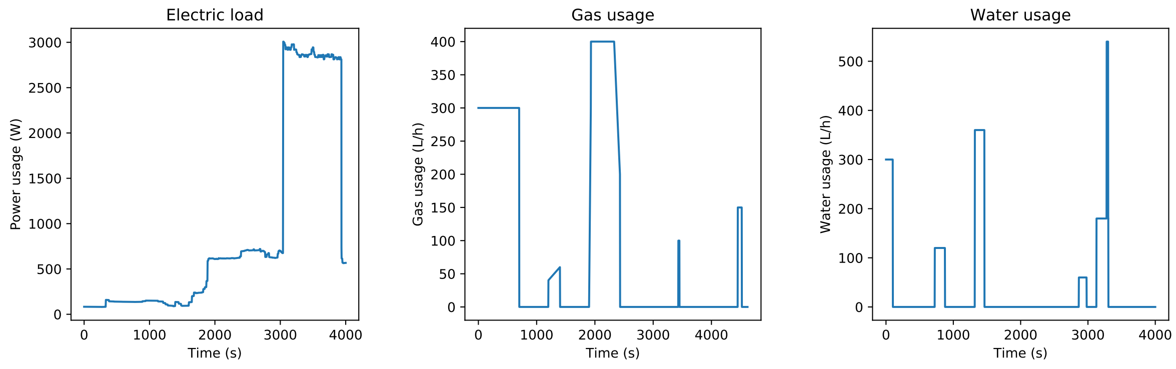


Figure 4.1: The load patterns that were used for the test results shown in Figure 4.3 and Figure 4.2.

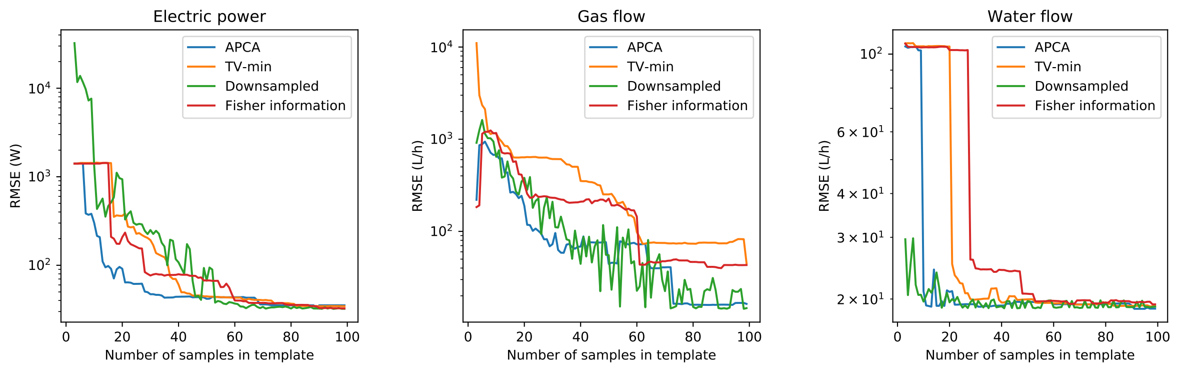


Figure 4.2: Root mean square error for the consumption rate of electricity, gas and water. The consumption rate was computed as the derivative from the estimated phase.

4.1. Performance for different compression rates

In the previous section, different compression methods were presented to reduce the number of samples that are needed to represent the signal template. Now, the performance of those different compression methods is evaluated as a function of the number of samples of the template.
For this test, typical consumption patterns were used for an electricity meter, a gas meter and a water meter, along with typical template patterns for the respective meter types. The templates used in this test are the same templates that were shown in Figure 3.8. The load patterns are shown in Figure 4.1. In this test, only the performance of the phase estimation step is evaluated. This implies that the template signal is known, and that the observed signal is zero mean and unit variance, so that no errors are introduced due to gain or offset errors. The load pattern that was used has a length of approximately 4000 seconds and was simulated at a sampling frequency such that the slope constraint of the algorithm was in the right range for the corresponding meter type, given the C-value of the meter and the maximum consumption for a household. The sampling frequencies that were used 200, 5 and 20 Hz for the electricity, gas and water meter respectively. For the first two tests, a SNR of 20 dB was used.

From the load pattern of Figure 4.1, the frequency of the meter over time was calculated. Then, a noisy observation $\boldsymbol{x}$ was simulated using the statistical model of Equation (3.2). The noisy simulated signal was used as an input to the phase estimation algorithm. Because the observed signals were simulated, a ground-truth reference of the actual consumption rate is available that was used to compute the performance of the phase estimation algorithm. After the estimation step of the phase estimation algorithm, the phase estimate is converted to an estimate of the power over time as described in Section 3.5, using C-values of 600, 100 and 1000 for the electricity, gas and water meter respectively. For the estimate of the consumption rate, the RMSE is shown as a function of the number of samples to represent the template in Figure 4.2. The template is considered 'uncompressed' at 200 samples, since no observable improvement in performance was observed when adding more than 200 samples. Figure 4.3 shows the accumulated error in Wh or litres over the total time window. Comparing the graphs, we can see that the accumulative error converges to zero for all compression methods, which means that when the signal is represented with enough samples, it will accurately estimate the number of revolutions that the meter has made. The phase estimation method might deviate locally, but will not deviate enough to miss a revolution. For example considering the gas meter, there will be a number of samples where the algorithms is able detect the peak the mirror is in front of the sensor, but there are not enough sample in the template to track the effects caused by the digits on the gas meter, causing a local deviation. The effect of those local deviations is reflected in the RMSE of the consumption rate.

There is a clear difference in the performance of the different compression methods. For low sample count, the data-adaptive APCA compression method performs the best. For a slightly higher sample count, the downsampling method performs best. The Fisher information method outperforms the TV minimisation method. As the number of samples increases, the error for all compression methods converges. When looking back at the results of Table 3.2, we can see that there is indeed a correlation between the error of the compressed template and the actual template, and the RMS performance measure presented in this chapter.

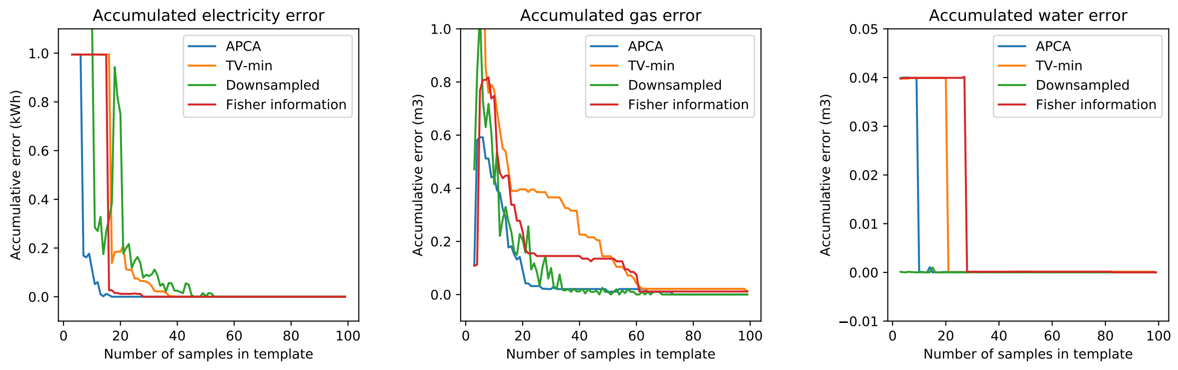


Figure 4.3: Comparison the error at the end of the estimation for different compression schemes. Once there are enough samples to not miss any of the clear peaks in the signal, the accumulated error converges to zero.

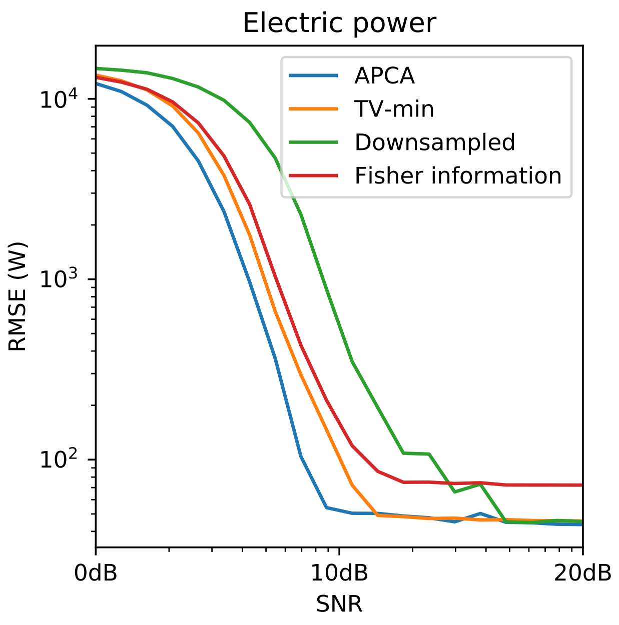


Figure 4.4: The performance of the different compression methods as a function of the signal to noise ratio of the observation for the electricity dataset. The templates were compressed using 50 samples. The SNR for recorded signals was typically between 10 and 100.

Performance for different signal-to-noise ratios

Figure 4.4 shows the result of a simulation where the tracking algorithm was run on the same electricity dataset as before, but with different signal-to-noise ratios (SNRs). The template signal was compressed using the four compression methods with 50 samples. In recorded signals from actual consumption meters, the SNR typically varied between 10 and 20 dB. In this test, the SNR was swept from 0 to 20 dB. As can be seen from Figure 4.4, the downsampled compression method performs poorly for low SNRs. The TV-minimisation and Fisher information method roughly show the same behaviour, and APCA method performs the best.

Other datasets

To test for the objective of detecting a meter with backwards motion, a load curve for a household with solar patterns was generated. A household that has solar panels can deliver power back through the grid when the solar panels generate more energy than the consumption rate of the household, and the meter rotates

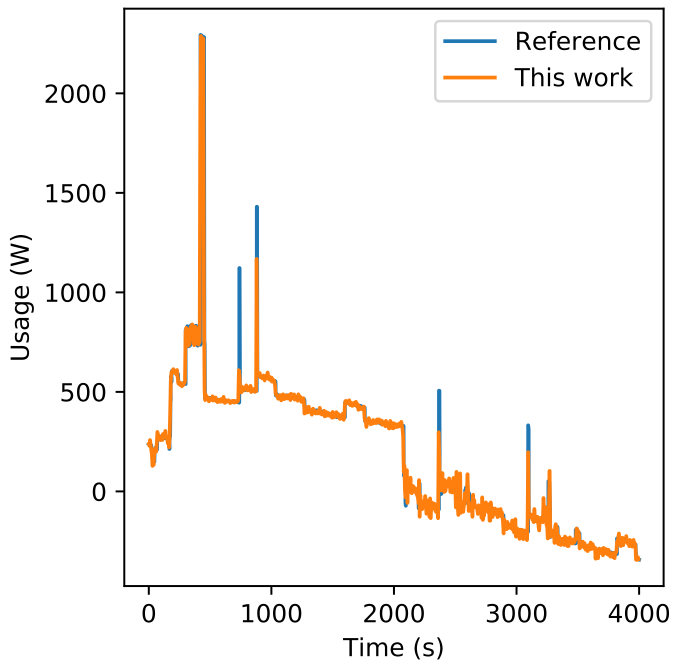


Figure 4.5: An example load curve for a household with solar panels. The power consumption from the household alternates between positive and negative usage, resulting in a meter that rotates backwards and forwards.

in the opposite direction of normal operation. Figure 4.5 shows an example of such a load curve. In the figure, the reference pattern is shown along with the estimate by the phase estimation algorithm and the APCA compressed signal. The accumulated usage error of the algorithm converged to zero as more samples were added to the template, and the RMS error converged to approximately 30 Watts. The curve with respect to the number of samples looked similar to that of the test with the normal electricity pattern, but slightly more samples were needed for the algorithm to direct the direction of the meter.

Next to the problem with solar panels, some older meter types exist that do not have a clear marking on the rotating part of the meter. Figure 4.7 shows an observed signal from a gas meter that does not have a clear peak in the signal. The phase estimation algorithm needed approximately 100 samples for the accumulative error of the phase estimation algorithm to converge to less than one period. This is slightly more than the 50-75 samples that were needed for the other patterns. This is due to the fact that there is more variation in the signal, and more samples are needed to capture the shape of the signal.

Proposed compression method

Based on the results of the simulations using different compression rates shown in Figures 4.2 to 4.4, the APCA compression method is selected as the proposed compression method, since it has the best overall performance. When using 75 samples in the compressed template, we can see that the accumulated error has converged for all three datasets with the APCA method, and that there is little decrease in RMS error if more than 75 samples are added. In the next section, those settings are used for the comparison with the other algorithms.

4.2. Comparison with other algorithms

For comparison, a number of different algorithms were implemented to track the phase of the consumption meters. As a first reference algorithm, an advanced peak detection algorithm was applied, which assumes a mirror or marking on the meter [62]. Secondly, the short time Fourier Transform-based pitch tracking method

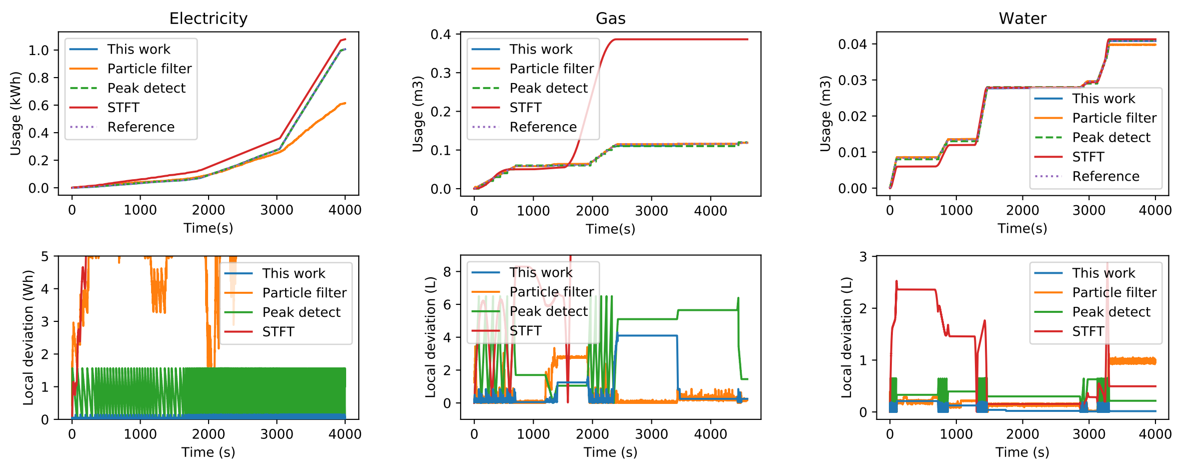


Figure 4.6: In the top graph, the usage pattern for the three different meter types is shown. This load curves that were used are the same as in Figure 4.1, but in this graph the accumulated consumption is shown instead of the consumption rate. The top graphs show the estimates for each of the algorithms, and the bottom graphs show a more close-up view of the error with the reference. In the bottom graph, it can be seen that although the peak detect has a good tracking performance, it has larger deviations in error than the proposed method.

Table 4.1: In adition to Figure 4.6 that show the estimates of the total consumed quantity, the table below shows RMS error of the consumption *rate* estimate of the four different algorithms. In addition to the three datasets of Figure 4.6, results are shown for the mirrorles gas meter of Figure 4.7 and the solar load curve of Figure 4.5. It can be seen from the table that the difference in error between the proposed method and other algorithms is the largest for the latter two datasets.

	This work	**Peak detection**	**Particle filter**	**STFT**
Electricity (W)	35.24	49.03	788.18	179.65
Gas (L/h)	49.01	67.80	124.45	581.79
Water(L/h)	19.32	25.16	64.37	55.37
Gas no mirror (L/h)	61.23	613.38	401.04	4687.28
Electricity solar (W)	43.07	258.81	287.16	252.73

[63] commonly found in speech processing-based applications was used to track the frequency over time. This frequency estimate was integrated to obtain a phase estimate. As a third reference, a particle filter [35] was implemented that generates a number of particles and propagate the particles through the nonlinear model that was described in this thesis. A more extensive discussion of those methods can be found in Chapter 2.

Some adjustments were made to the algorithm parameters depending on the dataset. For the particle filter, the update step was slightly adjusted for the different datasets to match the expected speed of the different meter types. For the STFT method, the window size was also varied per dataset optimise the time-frequency resolution. For the peak-detect algorithm, the threshold levels were selected manually to ensure the best algorithm performance. The phase estimation algorithm presented in this thesis was run with a 75-sample template that was compressed by the APCA method. Figure 4.6 shows the test load pattern for the electricity, water and gas dataset, and the corresponding absolute error of the estimated usage over time. From the top graphs in Figure 4.6, it can be seen that the particle filter and STFT method are not always able to accurately track the consumption and start to deviate. The method of this thesis and the peak detection method perform better. In the bottom graphs, the local error of the estimate is shown. Here, it can be seen that the method of this thesis has very low deviation in the consumption estimate, while the peak detection algorithm deviates more, because the estimate cannot be updated when there is no peak in the signal. The RMS error of the consumption rate is shown in Table 4.1.

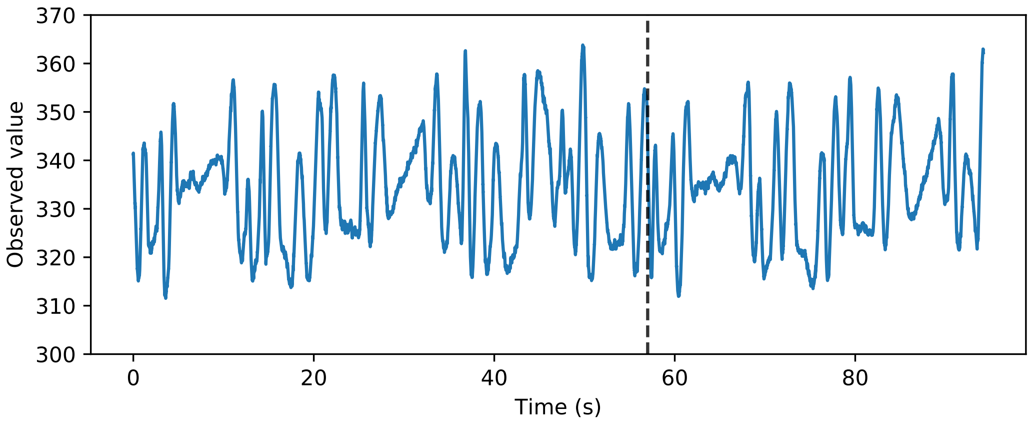


Figure 4.7: An observation from a gas meter with no mirror. In the figure, approximately one and a half revolutions of the meter are visible. Upon closer inspection, ten peaks can be seen in one period of the signal, caused by the numbers 0-9 on the display. The phase estimation algorithm converged to near zero accumulated error.

4.3. Algorithm complexity

Finally, the last objective for the algorithm is to have a computational complexity that allows the algorithm to run in real-time on an embedded device. In this section, we will demonstrate the feasibility of this with an approximate calculation. From the previous results, we can conclude at a sampling frequency of 200 Hz, the phase of the meter can accurately be estimated using a template of approximately 75 samples. Under those conditions, the algorithm has to consider 75×200 states per second. If finding the lowest cost for one state takes approximately 100 clock cycles, this means that the algorithm requires at least 1.5 million clock cycles per second. On a modern microcontroller running on a clock speed of 8 or 32 MHz, this should be feasible, leaving time for the backtracking procedure and other tasks. The memory usage of the algorithm can be quite high for a microprocessor, but the window size of the real-time algorithm as described in Section 3.4.5 can be adjusted to match the available memory of the microcontroller. On a modern computer, a C++-implementation of the algorithm could be compute the estimate for one hour in approximately two seconds, with a template size of 75 and sampling frequency of 200 Hz.

5 Discussion

5.1. Evaluation of the objectives

In Section 1.3, the problem statement for the phase estimation algorithm was introduced. In short, the objectives for the algorithm are as follows:

1. **Adaptability**: Performance of the algorithm on signals from various meter types.
2. **Resolution**: Decreasing the smallest detectable phase difference of the consumption meter.
3. **Able to detect direction**: Estimating the phase both when the consumption meter rotates forward and when the meter rotates backwards.
4. **Computationally efficient**: Sufficiently computationally efficient to run in real-time on an embedded device.

We will now shortly reflect on each of those goals.

Adaptability and ability to detect direction

In Chapter 4, it was shown that the phase estimation algorithm performed well on all the datasets. Two datasets were presented where all the comparative algorithms failed. The first test was the example of the gas meter with no clear peak, and the second example is the solar usage pattern, where the meter rotated backwards. In both cases, the algorithm performed well and demonstrated its adaptability.

Resolution

The algorithm that was designed allows for a trade-off between computational complexity and the resolution of the estimate by changing the number of samples in the template signal. Since the comparative algorithms require at least one full rotation or multiple periods to estimate the phase of the meter, this algorithm improves the resolution of the estimate.

Computational efficiency

The algorithm runs in $\mathcal{O}(MN)$ time, where M is the number of signals in the template signal, and N is the number of samples in the observation. That means that with regards to the data size, the algorithm scales with $\mathcal{O}(N)$. It was shown in Section 4.3 that the algorithm has a complexity that is low enough to run on an embedded device.

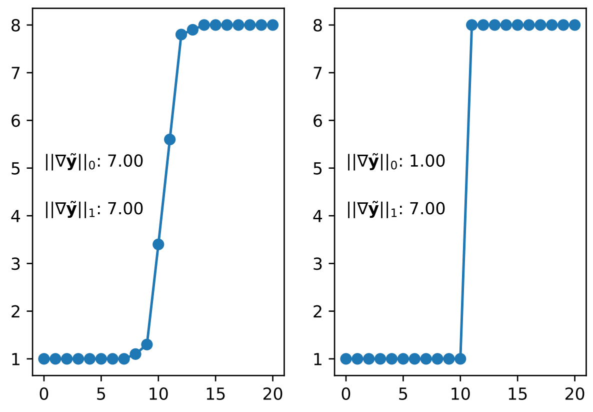


Figure 5.1: In total variation minimisation, the goal is to find a signal approximation $\tilde{\boldsymbol{y}}$ with a sparse derivative. On the left, a signal with a steep transition is shown. On the right, an approximation with a sparse derivative is shown with only one nonzero element. It is clear from the costs of the norm that the ℓ_1-norm does not change, while the ℓ_0-norm shows a big difference. Hence, the ℓ_1-norm is less suited for finding a sparse derivative

5.2. Analysis of compression methods

The results of Section 4.1 show that the performance of the algorithm depends on what type of compression method was used. The APCA method and the downsampling technique generally have the best performance, while the Fisher information sample selection and the TV minimisation show lesser results. To provide a motivation behind this difference in performance, let us recall the error of the compressed templates with their respective uncompressed templates from Table 3.2. In Table 3.2 we can see that the compression methods with lower reconstruction error also tend to have better performance with the phase estimation algorithm.

Adaptive piecewise constant approximation and uniform downsampling

Of the four compression methods that were evaluated in Section 3.6, the APCA method is the method that has the lowest reconstruction error with the original template. In the results of Section 4.1, it can be seen that in many cases the APCA method needs the fewest samples for the accumulative error to converge to zero. When the number of samples increases, the RMS error of the consumption rate was sometimes higher for APCA than for the downsampled signal. This could be because of the non-uniform spacing of the APCA method, which causes an uneven spreading of the phase estimate, leading to a higher RMS error.

Total variation minimisation

In TV minimisation, the goal is to find a signal with a sparse derivative that has a low reconstruction error with the original signal. To minimise the number of elements in the derivative, the ℓ_0 norm can be used to minimise the number of elements in the derivative of the signal. The optimisation problem could be formulated as:

$$\min_{\tilde{\boldsymbol{y}}} \left\|\boldsymbol{y}-\tilde{\boldsymbol{y}}\right\|_2+\lambda\left\|\nabla\tilde{\boldsymbol{y}}\right\|_0, \tag{5.1}$$

where $\tilde{\boldsymbol{y}}$ represents the approximation of the signal, $\nabla\tilde{\boldsymbol{y}}$ the first order difference of the signal, and $\boldsymbol{y}$ the original signal. $\|\cdot\|_2$ and $\|\cdot\|_0$ represent the ℓ_0 and ℓ_2-norm respectively. The first term of the cost function minimises the squared error with the original signal, and the second term minimises the number of nonzero elements in the derivative. The main problem is that the ℓ_0-norm makes the problem a non-convex problem, and thus difficult to solve efficiently [59, p.310]. For this reason, the ℓ_0-norm is relaxed to a ℓ_1-norm, which in many cases was shown to approximate the original problem well [64],[59, p.314].
In this case, this relaxation impacts the performance of the algorithm when trying to find an approximation of the template with a fixed number of segments (or equivalently, nonzero elements of the derivative).

Consider a template signal with a steep transition in the middle of the template such as shown in Figure 5.1, with a number of successive samples on the edge of the transition. If the edge is steep, it could be approximated with a single step between the state before the edge and the state after the edge, without any intermediate samples. With the ℓ_0-norm, this would be a likely outcome of the TV minimisation, since the signal is well reconstructed with just one nonzero element of the derivative. With the ℓ_1-norm however, including all transitions yields approximately the same cost, since the ℓ_1-norm of the sum of all the transitions that form the edge has approximately the same cost as one single transition for the edge under the ℓ_1-norm. Because of this approximation, the compression based on TV-minimisation tends to select all samples on the steep edges of the signal, leaving fewer samples for other details in the template. This can also be seen in the plots of the compressed templates in Figure 3.8.

Fisher information-based sample selection

Recall the Fisher information derived for our statistical model as derived in Section 3.6:

$$I[y(\theta[i])] = \frac{1}{\sigma^2}\frac{\partial^2 y(\theta[i])}{\partial\theta[i]^2}. \qquad \text{(Equation (3.39) revisited)}$$

The Fisher information is a measure of how accurately θ can be estimated with respect to the measurement noise of the model. That is, the locations in the template where a deviation in the measurement $x[i]$ causes the smallest deviation in the phase estimate θ. From the equation we can see that the Fisher information increases with the amplitude of the derivative of the template. By definition, the derivative of $y[\theta]$ relates how a small change in input in θ relates to a change in $y[\theta]$. Therefore, for high derivatives a disturbance in $x[i]$ will result in a small change in the estimated θ, and when the derivative is low in amplitude, a disturbance in $x[i]$ will have a bigger impact on the estimate θ.

Although this is true locally, it does not take into account that even in the noise-free case, it is still not always possible to recover the phase directly from an observation, since the template signal can be such that multiple θ satisfy $y[\theta] = x[i]$. If the model was a one-to-one function, this method would likely perform better. In this case, the Fisher-information sampling technique optimizes for local performance with respect to the noise, but does not optimize for a good reconstruction of the template, which results in a worse performance than the other methods.

5.3. Comparison with other algorithms

In Section 4.2, the performance of the phase estimation algorithm was compared to conventional methods for tracking time series signals. In this section, a brief discussion of the performance of each of the algorithms is provided.

Peak detection method

The conventional peak detection method worked reliable on the three main datasets for the water meter, gas meter and electricity meter. Since it was able to detect all the pulses, the accumulative phase error was approximately zero for those three datasets. The RMS error of the consumption rate was significantly higher for the peak detection algorithm, especially in the case of the electricity meter and gas meter. When the gas or water usage drops from a certain consumption rate to zero, it might take a long time for the next pulse comes by, and the frequency cannot accurately be estimated in the time between the two pulses. The phase estimation algorithm presented in this thesis was able to estimate the steep declines in consumption rate and low consumption rates better, since it can detect when the meter stops rotating.
Secondly, two types of datasets were presented that could not be tracked by the peak detection algorithm. For the usage pattern with negative consumption, the peak detection is still able to detect the peaks, but not the direction of the signal. This means that all negative value consumption would be counted as positive consumption. The second example showed an example signal that did not have a clear peak in the signal. In this case, the peak detection would not be able to count the number of revolutions.

Particle filtering method

The particle filter is a technique that generates a number of particles randomly. Each particle represents a system state, and the particles propagate according to the (stochastic) system model. At each iteration, new particles are generated randomly based on the likelihood of the predicted output of the particle state. The particle filter is a relatively simple model that can handle the nonlinearity of the template-based model well. The particle filter was able to track the signal in most cases, but proved to be strongly dependent on the update step of the model. When the samples were allowed to spread to different states too much, the system did not converge. When there was too little spread in the update step, there were not enough samples that moved enough to track the signal. Possibly, the quality of the estimation could be improved when the particle state is extended to contain an estimate of the frequency and phase, instead of only a phase estimate. Furthermore, since the particle filtering method is a stochastic method, it did not produce the same results when it was applied twice to the same observation. The particle filter roughly has the same time complexity as the phase estimation algorithm presented in this thesis, but performed less in the tests that were conducted.

Short time Fourier transform

The short time Fourier transform (STFT) is a commonly used technique to analyse the frequency of a signal over time. Generally, the assumption is made that within a short period of time, the signal is locally stationary. Over this time period, the Fourier transform is applied, and in the Fourier domain the main frequency of the signal is determined. For the electricity signal, the STFT performed relatively well. For the water and gas dataset, the performance was less. This is mainly because the electricity meter generally rotates faster, and almost never stops due to a typical standby usage of several tens of Watts in a common household. Because of this it is more likely in the case of the electricity that there are multiple periods of the signal within one time frame of the STFT, so that the frequency is accurately estimated. When the signal is moving to slow to capture a revolution in a STFT time frame or when the frequency changed within a time frame, the STFT method was not able to detect the main signal frequency correctly. For the gas and water meter, this happened more often due to the pulse-like usage patterns and due to the slow moving meters.

Kalman filters

Kalman filtering is a filtering technique for signals embedded in noise. The Kalman filter is a minimum mean-squared error (MMSE) estimator that uses the information of a dynamic system model to estimate the model parameters. The estimator is optimal under the assumption of a linear model and Gaussian noise [37, p. 446]. A Kalman filter predicts the next value of the estimated parameter using the statistical model, and combines the estimate of the model with the observed measurement. When a new measurement is observed, the error between the predicted value of the measurement and the observed value can be computed.

The Kalman filter was not included as a reference method, since applying Kalman filtering techniques to the model in this thesis brings forth a number of challenges that are nontrivial. For the ordinary Kalman filter, a model is required with linear update equations. For nonlinear models such as the model in this thesis, the Extended Kalman filter (EKF) [38, p. 400] and Unscented Kalman filter (UKF) [39] were developed. The model that was presented in Section 3.1 consists of noisy observations $x[i]$ from the internal state $\theta[i]$ and a template function y. One of the problems with this model is that no update rule for $\theta[i]$ is assumed. Secondly, the Kalman filter will make an estimate for the state only based on the previous estimate and the current observation, whereas the model presented in this work finds the maximum likelihood estimate for the model given the entire sequence of observations. For these reasons, implementing the Kalman-filtering techniques was not considered as a viable reference method for this problem.

Analogy with Hidden Markov models

The model presented in this thesis bears resemblance with the Hidden Markov Modelling (HMM) technique and the corresponding Viterbi algorithm [21, p. 523] as described in [5, 65]. The Hidden Markov Model is based on a Markov model with a number of states that are not directly observable. Instead, only the stochastic output of the state is observable. When applied to the problem of this thesis, the states could represent different phases of the consumption meter, and the output the observed reflection that corresponds with that phase. HMMs have been used for similar problems such as the analysis of ECG signals [66] or EEG rhythms [2].

The main difference between an HMM and the model presented in this thesis is in the transition probabilities. In an HMM, it is assumed that the transition probabilities between different states are fixed and are known, or can be estimated. The Viterbi algorithm uses a similar dynamic programming technique to find the sequence of states that has the highest probability. In this work, it was assumed that the maximum difference between two succeeding phase estimated is limited, thereby constraining the slope of the phase estimate.

When the transition probabilities of the HMM are set equal for all states that satisfy this slope constraint, and zero for all states that violate the constraint, the Viterbi algorithm could be used to yield the same result as the phase estimation algorithm of this thesis. In the model of this thesis, the transition between different states of the template is dependent on the underlying frequency of the consumption meter, and can vary greatly over time.

5.4. Vulnerabilities of the algorithm

Although the algorithm performs well and achieves all the goals that were set in the objective, there are a few situations where the performance of the algorithm quickly degrades. There are two notable cases that will be discussed: sensitivity against offsets or gain errors and having an incorrect number of samples in the template.

Offset and gain errors

One of the main assumptions of the algorithm is that both the template and observation have the same offset and scaling. Since the algorithm tries to find the best match of the *values* of the observed signal with the values of the template signal, the algorithm is sensitive to deviations the signal values. Figure 5.2 shows an illustration of a phase estimation procedure where the offset is slowly increased. At first, the algorithm will track the phase of the signal without any problems, but as the offset error increases, the observed signals has a progression of values that cannot be matched with the values in the template. To minimise the alignment cost, the algorithm will start to favour the template samples with higher values, until finally all the samples of the observation lie outside of the range of the template, and the best estimate that the algorithm can make is the template state with the highest value.
For the gain offset, a similar situation occurs where the values of the observation can no longer efficiently matched with the template signal. An example of this is shown at the bottom of Figure 5.2. It is difficult to avoid this kind of vulnerability in the algorithm, since the phase estimation algorithm *has* to rely on the values of the time series, since the frequency of the meter can change at any time.

Therefore, the algorithm has to rely on the normalisation step to correct for the gain and offset of the observed signal. However, this estimation step will of course not always be perfect. Since the gain and vertical offset are only updated during stationary periods of the signal, the normalisation step could fail if there is a big change in the gain or vertical offset during a nonstationary period of the signal. One approach to mitigate this issue could be to monitor the alignment cost of the algorithm. If there is a sudden increase in the warping or alignment cost, this could be an indicator that a sudden change in gain or offset has occurred, and that the aligned template does not match the observed signal well. This increase in cost could trigger a recalculation of the gain and offset, or could serve as an indicator that the estimate is unreliable. In the case of the meter phase estimation problem this problem is unlikely to occur, since many meters are installed in a dark environment where the conditions in ambient light do not change.

Incorrect number of template samples

When the number of samples in the template is either too high or too low, the estimation procedure can lead to erroneous results. First, consider the case where there are a low number of samples in the (compressed) template signal. Since the algorithm is applied directly to the template values, the number of values in the

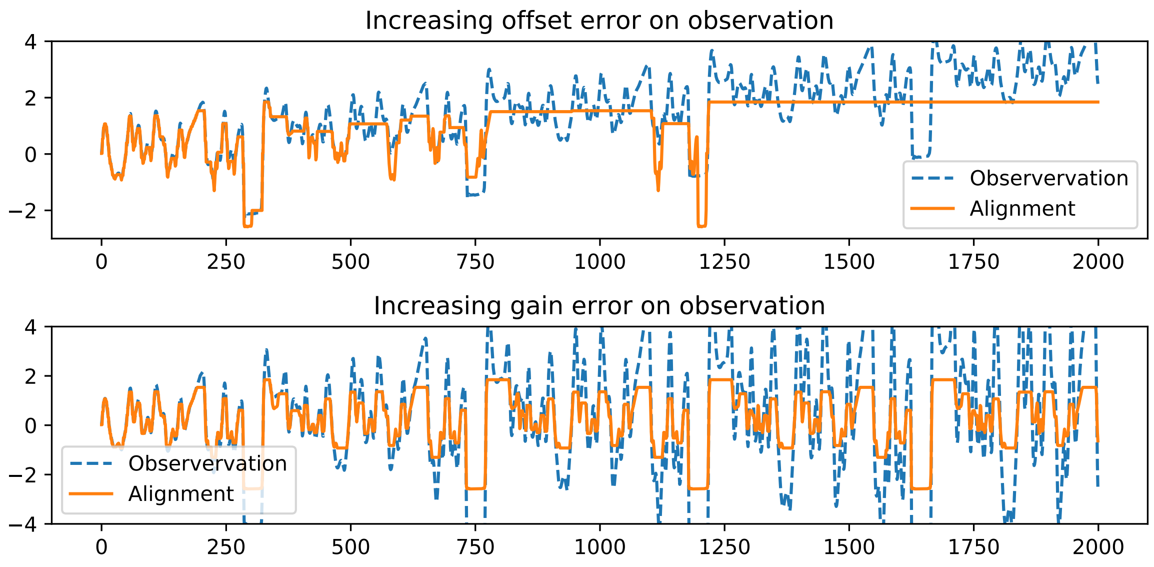


Figure 5.2: An observed signal with an increasing gain and offset error are shown. When the offset error increases too much, only the states with the highest values of the template will be used. For the gain error, the template can only be matched at the zero crossings, and for the other parts the extreme values of the template are used

template also influences the slope constraint of the algorithm. If there are only three samples in the template and the slope is limited to one state per observed sample as described in Table 3.1, the maximum slope would be one period in only three samples. Note that when the observed signal is noisy, the algorithm can quickly skip to a state that matches the current value best. Therefore, the sampling rate of the observed signal and the slope constraint of the algorithm should be matched so that the slope constraint approximately matches with the maximum expected frequency based on the maximum consumption rate for a household.

A similar problem can occur when there are too many samples in the template and the algorithm tracks both forward and backwards motion. Consider the case where there is a steep edge in the template signal, but the edge is still represented by many samples. In that case, the algorithm could in some cases find an optimal alignment by just moving along that edge, and 'selecting' the best matching value by moving forwards and backwards along that edge. This problem only occurs when there is much noise on the signal. It can be prevented by filtering out a part of the noise, or by adding constraints to the algorithm that limit the number of times the algorithm can switch between backwards and forwards rotation.

One other example of a mismatch in the estimated phase occurs when the template signal is sufficiently symmetric, and the observation is noisy. In this case it can occur that the phase estimation algorithm aligns the signal by moving the meter backwards and forwards, but never more than a full rotation. Because of the template symmetry, an alignment is generated that matches the observed signal well.

In all of the cases described above, the phase estimation algorithm still finds the optimal cost within the constraints that were provided. The problem here is that although the algorithm finds the optimal solution for the model and the constraints that were specified, it does not always find a solution that is close to the true phase. This typically occurs when there are too little samples in the template or when the noise on the signal is high.

6 Conclusions and future work

A phase estimation method was developed that is able to track the phase of recurring patterns in nonstationary time series, with an application to the monitoring of analogue consumption meters. The time series signal was modelled with a template pattern that is repeated over time at varying frequencies, where the frequency can change rapidly over time, and no assumption about local stationary are made. Using the dynamic programming methods common to the Dynamic Time Warping (DTW) and Viterbi algorithm, a constrained maximum-likelihood estimate for the phase over time was derived. A general framework was proposed that can be applied to any time series with repeating patterns that correspond with the template-based signal model.

The phase estimation algorithm was shown to improve in accuracy and adaptability over the conventional peak detection algorithm, at the cost of increased computational complexity. The algorithm was tested on an electricity meter, a gas meter and a water meter with realistic phase signals. Two examples were shown of datasets with signals with negative rotation and signals without a distinctive peak, that could not be tracked by the conventional peak detection method. The proposed algorithm could accurately estimate the phase in both cases.

To reduce the complexity of the algorithm, a number of different compression techniques for the signal template were implemented, and the performance of the different compression methods was compared. Using the Adaptive piecewise constant approximation (APCA) technique, an accurate estimate could be obtained for the three common meter types with a time complexity that is low enough to implement the algorithm on a simple embedded device.

Some conditions were described that lead to erroneous estimations of the algorithm, due to an incorrect number of samples in the template, incorrect sample frequency or changing lighting conditions. With the appropriate preconditions, those circumstances can be avoided.

Suggestions for future work

In this thesis, a framework for estimating the phase of consumption meters was tested. The work has been verified on simulated observations and on some home recordings, but further testing is needed to test the robustness of the algorithm in real-life situations. The algorithm relies on an accurate estimate of the signal template. In practice, the estimated signal template might be noisy or might have a slightly incorrect length. The algorithm will still be able to track the phase when the template is distorted, but it is unknown what the precise effect of distortion of the template signal will be. Tests with the algorithm in a real-life situation or with simulated distortions in the template could provide more insight into how the template influences estimation performance. One other parameter of interest is the window size of the algorithm. The effect of the windowing procedure might affects the performance of the algorithm, and real-life test could provide more insight in the minimum window length that is required.

Furthermore, the algorithm is sensitive against disturbances in the vertical offset and gain of the signal. In many cases the meter can be placed in an environment with fixed lighting conditions and this problem can be avoided, but there will also be cases where ambient light influences the meter. Further research is needed to investigate how strong this influence is, and if it is possible to accurately track the gain and vertical offset parameters in rapidly changing lighting conditions, such as partly cloudy weather for meters that are placed outside.

Next to this, the model could be extended with statistics about the expected frequency of the meter. In the signal model of this thesis, no assumptions about the progression of the meter were made, other than that its maximum frequency is bounded. The model could be extended further with statistics about the frequency of the meter. Public datasets and probabilistic models for energy usage have already been published [67, 68], and those could be used as prior information about the phase over time.

Finally, the applications of the template-based tracking technique could be explored. The algorithm as presented could by applied to other nonstationary time series that are generated from a recurring pattern. Examples of such time series are ECG recordings, EEG recordings or other set-ups where a sensor is connected to a rotating or oscillating object.

Bibliography

[1] Commission Benchmarking smart metering deployment in the EU-27 with a focus on electricity. Cost-benefit analyses & state of play of smart metering deployment in the EU-27. Technical report, Europian Union, 2014. URL `https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52014SC0189&from=EN`.

[2] Silvia Chiappa and Samy Bengio. HMM and IOHMM modeling of EEG rhythms for asynchronous BCI systems. Technical report, IDIAP, 2003.

[3] Hau-Tieng Wu, Gregory F Lewis, Maria I Davila, Ingrid Daubechies, Stephen W Porges, et al. Optimizing estimates of instantaneous heart rate from pulse wave signals with the synchrosqueezing transform. *Methods of Information in Medicine*, 55(5):463–472, 2016.

[4] Huanmei Wu, Betty Salzberg, and Donghui Zhang. Online event-driven subsequence matching over financial data streams. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 23–34. ACM, 2004.

[5] Lawrence R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Readings in speech recognition*, pages 267–296. Elsevier, 1990.

[6] Woojay Jeon and Changxue Ma. Efficient search of music pitch contours using wavelet transforms and segmented dynamic time warping. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 2304–2307. IEEE, 2011.

[7] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. Time-series clustering–A decade review. *Information Systems*, 53:16–38, 2015.

[8] Tak-chung Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24 (1):164–181, 2011.

[9] Philippe Esling and Carlos Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1):12, 2012.

[10] Xiaoyue Wang, Abdullah Mueen, Hui Ding, Goce Trajcevski, Peter Scheuermann, and Eamonn Keogh. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery*, pages 1–35, 2013.

[11] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[12] Monson H Hayes. *Statistical digital signal processing and modeling*. John Wiley & Sons, 2009.

[13] John Paparrizos and Luis Gravano. k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1855–1870. ACM, 2015.

[14] Yasushi Makihara, Muhammad Rasyid Aqmar, Ngo Thanh Trung, Hajime Nagahara, Ryusuke Sagawa, Yasuhiro Mukaigawa, and Yasushi Yagi. Phase estimation of a single quasi-periodic signal. *IEEE Transactions on Signal Processing*, 62(8):2066–2079, 2014.

[15] Lawrence Rabiner. On the use of autocorrelation analysis for pitch detection. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(1):24–33, 1977.

[16] Michail Vlachos, Philip Yu, and Vittorio Castelli. On periodicity detection and structural periodic similarity. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 449–460. SIAM, 2005.

[17] John R Deller, John HL Hansen, and John G Proakis. Discrete-time processing of speech signals. 2000.

[18] RM Fano. Short-Time Autocorrelation Functions and Power Spectra. *The Journal of the Acoustical Society of America*, 22(5):546–550, 1950.

[19] Ingrid Daubechies, Jianfeng Lu, and Hau-Tieng Wu. Synchrosqueezed wavelet transforms: A tool for empirical mode decomposition. *arXiv preprint arXiv:0912.2437*, 2009.

[20] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[21] S Theodoridis and K Koutroumbas. Pattern recognition, 2009.

[22] Fumitada Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):67–72, 1975.

[23] Eamonn J Keogh and Michael J Pazzani. Derivative dynamic time warping. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pages 1–11. SIAM, 2001.

[24] Simon Dixon. An On-Line Time Warping Algorithm for Tracking Musical Performances. In *IJCAI*, pages 1727–1728, 2005.

[25] Ali Zifan, Sohrab Saberi, Mohammad Hassan Moradi, and Farzad Towhidkhah. Automated ECG segmentation using piecewise derivative dynamic time warping. *International Journal of Biological and Medical Sciences*, 1(3), 2006.

[26] Guiling Li, Yuanzhen Wang, Min Li, and Zongda Wu. Similarity match in time series streams under dynamic time warping distance. In *Computer Science and Software Engineering, 2008 International Conference on*, volume 4, pages 399–402. IEEE, 2008.

[27] Louis Comtet. *Advanced Combinatorics: The art of finite and infinite expansions.* Springer Science & Business Media, 2012.

[28] Richard Bellman. *Dynamic programming.* Courier Corporation, 1957.

[29] Meinard Müller. *Information retrieval for music and motion*, volume 2. Springer, 2007.

[30] CS Myers and L Rabiner. Connected digit recognition using a level-building DTW algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(3):351–363, 1981.

[31] Hailin Li, Xiaoji Wan, Ye Liang, and Shile Gao. Dynamic time warping based on cubic spline interpolation for time series data mining. In *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, pages 19–26. IEEE, 2014.

[32] Martin Schmidt, Mathias Baumert, Alberto Porta, Hagen Malberg, and Sebastian Zaunseder. Two-dimensional warping for one-dimensional signals—conceptual framework and application to ECG processing. *IEEE Transactions on Signal Processing*, 62(21):5577–5588, 2014.

[33] Robert Macrae and Simon Dixon. Accurate Real-time Windowed Time Warping. In *ISMIR*, pages 423–428, 2010.

[34] Hoang Anh Dau, Diego Furtado Silva, François Petitjean, Germain Forestier, Anthony Bagnall, and Eamonn Keogh. Judicious Setting of Dynamic Time Warping's Window Width Allows More Accurate Classification of Time Series. IEEE Conference Publications, 2017.

[35] Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

[36] Xiao-Li Hu, Thomas Schön, and Lennart Ljung. A basic convergence result for particle filtering. *IEEE Transactions on Signal Processing*, 56(4):1337–1348, 2008.

[37] Steven M Kay. *Fundamentals of statistical signal processing, volume I: estimation theory*. Prentice Hall, 1993.

[38] Dan Simon. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[39] Eric A Wan and Rudolph Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. Ieee, 2000.

[40] Habib Hajimolahoseini, Rassoul Amirfattahi, Hamid Soltanian-Zadeh, and Saeed Gazor. Instantaneous fundamental frequency estimation of non-stationary periodic signals using non-linear recursive filters. *IET Signal Processing*, 9(2):143–153, 2015.

[41] H Hajimolahoseini, MR Taban, and HR Abutalebi. Improvement of extended kalman filter frequency tracker for nonstationary harmonic signals. In *Telecommunications, 2008. IST 2008. International Symposium on*, pages 592–597. IEEE, 2008.

[42] Lawrence Rabiner and Biing-Hwang Juang. Fundamentals of speech processing. *Prantice Hall*, 1993.

[43] Eamonn J Keogh and Michael J Pazzani. A simple dimensionality reduction technique for fast similarity search in large time series databases. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 122–133. Springer, 2000.

[44] Eamonn J Keogh and Michael J Pazzani. An Enhanced Representation of Time Series Which Allows Fast and Accurate Classification, Clustering and Relevance Feedback. In *Kdd*, volume 98, pages 239–243, 1998.

[45] Kaushik Chakrabarti, Eamonn Keogh, Sharad Mehrotra, and Michael Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Transactions on Database Systems (TODS)*, 27(2):188–228, 2002.

[46] Juozas Gordevičius, Johann Gamper, and Michael Böhlen. Parsimonious temporal aggregation. *The VLDB Journal*, 21(3):309–332, 2012.

[47] JINFEI Xie and WY Yan. Pattern-based characterization of time series. *International Journal of Information and Systems Science*, 3(3):479–491, 2007.

[48] Yueguo Chen, Mario A Nascimento, Beng Chin Ooi, and Anthony KH Tung. Spade: On shape-based pattern detection in streaming time series. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 786–795. IEEE, 2007.

[49] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15(2):107–144, 2007.

[50] Fu-Lai Chung, Tak-Chung Fu, Robert Luk, and V Ng. Flexible time series pattern matching based on perceptually important points. 2001.

[51] Fabian Mörchen. Time series feature extraction for data mining using DWT and DFT, 2003.

[52] Daniel TL Lee and Akio Yamamoto. Wavelet analysis: theory and applications. *Hewlett Packard journal*, 45:44–44, 1994.

[53] Yuhan Cai and Raymond Ng. Indexing spatio-temporal trajectories with Chebyshev polynomials. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 599–610. ACM, 2004.

[54] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.

[55] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[56] Michael Lustig, David Donoho, and John M Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic resonance in medicine*, 58(6):1182–1195, 2007.

[57] Felix Krahmer, Christian Kruschel, and Michael Sandbichler. Total Variation Minimization in Compressed Sensing. In *Compressed Sensing and its Applications*, pages 333–358. Springer, 2017.

[58] David Strong and Tony Chan. Edge-preserving and scale-dependent properties of total variation regularization. *Inverse problems*, 19(6):S165, 2003.

[59] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[60] Johan Swärd, Filip Elvander, and Andreas Jakobsson. Designing Sampling Schemes for Multi-Dimensional Data. *arXiv preprint arXiv:1707.03247*, 2017.

[61] Julius O. Smith. *Mathematics of the Discrete Fourier Transform (DFT)*. W3K Publishing, http://www.w3k.org/books/, 2007. ISBN 978-0-9745607-4-8.

[62] M. van der Reek. Energy-Efficient Electricity-Meter Monitoring. Master's thesis, TU Delft, 2017.

[63] Konstantinos Paraschakis and Rainer Dahlhaus. Frequency and phase estimation in time series with quasi periodic components. *Journal of Time Series Analysis*, 33(1):13–31, 2012.

[64] Carlos Ramirez, Vladik Kreinovich, and Miguel Argaez. Why l1 is a good approximation to l0: A geometric explanation. *Journal of Uncertain Systems*, 7(3):203–207, 2013.

[65] Zoubin Ghahramani. An introduction to hidden Markov models and Bayesian networks. *International journal of pattern recognition and artificial intelligence*, 15(01):9–42, 2001.

[66] Rodrigo Varejão Andreão, Bernadette Dorizzi, and Jérôme Boudy. ECG signal analysis through hidden Markov models. *IEEE Transactions on Biomedical engineering*, 53(8):1541–1549, 2006.

[67] Joakim Munkhammar, Joakim Widén, and Jesper Rydén. On a probability distribution model combining household power consumption, electric vehicle home-charging and photovoltaic power production. *Applied Energy*, 142:135–143, 2015.

[68] Oliver Parson, Grant Fisher, April Hersey, Nipun Batra, Jack Kelly, Amarjeet Singh, William Knottenbelt, and Alex Rogers. Dataport and nilmtk: A building data set designed for non-intrusive load monitoring. 2015.

A

MVU estimators for vertical offset and gain

In this section, the MVU estimators for the vertical offset b and gain a of the model presented in Section 3.1 are derived.

$$x[i] = Ay[\theta[i]] + b + \epsilon[i]\,, \tag{A.1}$$

where $\epsilon[i]$ is zero mean Gaussian noise with variance σ, $\boldsymbol{y}$ is the template vector, indexed by the phase $\theta[i]$. From this model, we have the probability density function

$$p_X(\boldsymbol{x} \mid \boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{2/N}} \exp\left(\frac{-\sum_{i=0}^{N-1}(x[i] - (Ay[\theta[i]] + b))^2}{2\sigma^2}\right). \tag{A.2}$$

with corresponding log-likelihood

$$\ln p_X(x \mid \theta[i]) = \ln\left(\frac{1}{(2\pi\sigma^2)^{2/N}}\right) - \frac{(x - (Ay[\theta[i]] + b))^2}{2\sigma^2}. \tag{A.3}$$

We define the model parameters $\boldsymbol{\theta}$ as $[A\,b]^T$. For those parameters, the Fisher information matrix is

$$\boldsymbol{I}(\boldsymbol{\theta}) = \frac{1}{\sigma^2}\begin{bmatrix} \sum y[\theta[i]]^2 & \sum y[\theta[i]] \\ \sum y[\theta[i]] & N \end{bmatrix}. \tag{A.4}$$

The MVU estimator must satisfy the following relation [37]:

$$\frac{\partial p_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \boldsymbol{I}(\boldsymbol{\theta})(\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{\theta})\,, \tag{A.5}$$

where $\boldsymbol{g}(\boldsymbol{x})$ is the estimator function. Substituting the probability density function of Equation (A.3) and the Fisher information of Equation (A.4) we obtain

$$\frac{1}{\sigma^2}\begin{bmatrix} \sum(x[i] - Ay[\theta[i]] - b)y[\theta[i]] \\ \sum(x[i] - Ay[\theta[i]] - b) \end{bmatrix} = \frac{1}{\sigma^2}\begin{bmatrix} \sum y[\theta[i]]^2 & \sum y[\theta[i]] \\ \sum y[\theta[i]] & N \end{bmatrix}\begin{bmatrix} \hat{A} - A \\ \hat{b} - b \end{bmatrix}. \tag{A.6}$$

Computing the product on the right hand side and rearranging for $\hat{A}$ and $\hat{b}$ we find

$$\begin{bmatrix} \hat{A} \\ \hat{b} \end{bmatrix} = \begin{bmatrix} \dfrac{\sum x[i]y[\theta[i]] - \hat{b}\sum y[\theta[i]]}{\sum y[\theta[i]]} \\ \dfrac{1}{N}\sum x[i] - \dfrac{1}{N}\hat{A}\sum y[\theta[i]] \end{bmatrix}. \tag{A.7}$$

Solving for $\hat{A}$ we find the MVU estimator for A as

$$\hat{A} = \frac{\sum x[i]\,y[\theta[i]] - \frac{1}{N}\sum x[i]\sum[y\theta[i]]}{\sum y[\theta[i]]^2 - \frac{1}{N}\left(\sum[y\theta[i]]\right)^2} = \frac{\mathrm{cov}(\boldsymbol{x},\boldsymbol{y})}{\mathrm{cov}(\boldsymbol{y},\boldsymbol{y})}\,. \tag{A.8}$$

From this it follows that an efficient estimator for $\hat{A}$ and $\hat{b}$ exists, but that the estimator requires the template $\boldsymbol{y}$ to be known.