EXTENDED KALMAN FILTER APPROACH TO
ELECTRIC ARC FURNACE
FORECASTING

by

Ali Hosseingholizadeh

Bachelor of Science in Mechanical Engineering, Azad University,
Science & Research Campus, Tehran - Iran, 2002

A thesis
presented to Ryerson University
In partial fulfillment of the
requirements for the degree of
Master of Science
In the program of
Computer Science

Toronto, Ontario, Canada, 2015

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

EXTENDED KALMAN FILTER APPROACH TO ELECTRIC ARC FURNACE FORECASTING

Ali Hosseingholizadeh

M. Sc. in Computer Science, 2015

Ryerson University, Toronto, Canada

Abstract

In this thesis, we propose and implement a new approach for building an online self-adjusting model for prediction of v-i characteristic of a multivariate time series obtained from an operational electrical arc furnace. The proposed methodology is based on the Kalman filtering method, and is used for prediction of the arc furnace voltage using the past history of the current and voltage. The main advantage of the proposed approach over similar earlier related work is the ability to adapt during the operation of the furnace. In this study, three different hybrid models have been developed based on the extended Kalman filtering technique and one of the following methodologies: (i) a linear auto regressive model; (ii) fuzzy logic, (iii) wavelet analysis. The results compare well with those of earlier work and clearly indicate that the augmentation of the above mentioned approaches with the extended Kalman filter improves the prediction accuracy.

Acknowledgements

I would like to express my special gratitude to my supervisor, Prof. Alireza Sadeghian, for his support, motivation, enthusiasm, and immense knowledge. His guidance was a great help for me throughout the research and writing of this thesis.

Besides my advisor, I would also like to thank my thesis committee members, Prof. Cherie Ding, Prof. Isaac Woungang and Prof. Ali Miri, for their encouragement and insightful comments.

Lastly, I would also like to sincerely thank my parents, Kobra and Rahman, and my sister, Fariba, for all their unconditional support.

Dedication

To my parents, Kobra and Rahman, for their unconditional love and support …

TABLE OF CONTENTS

List of tables

List of figures

List of Appendices

List of abbreviations

ANFIS	Adaptive Neuro Fuzzy Interface System
AR	Autoregression
BP	Back-Propagation
BPA	Back-Propagation Algorithm
CDF	Cumulative Density Function
DDM	Data Driven Methods
DWA	Discrete Wavelet Analysis
EAF	Electrical Arc Furnace
EKF	Extended Kalman Filter
FIS	Fuzzy Logic Interface
HMM	Hidden Markov Model
KCL	Kirchoff's Curent Law
KF	Kalman Filter
LQE	Linear Quadratic Estimator
LUP	Lookup Table
MRA	Multi Resolution Analysis
PDF	Probability Density Function
PNN	Probabilistic Neural Network
PQ	Power Quality
PSD	Power Spectral Density
RBFNN	Radial Basis Function Neural Networks

1. INTRODUCTION

1.1. Electrical Arc Furnace

An Electrical Arc Furnace (EAF) is a highly efficient melting apparatus which uses electrical arc to melt down the charged material. The EAF's are generally used for production of steel from steel scraps [1]. The first stage of process, known as charging stage, begins with loading of the furnace with steel scraps. Then the second stage (melting stage) starts with lowering of the graphite electrodes into the scraps. During this phase an intermediate voltage is supplied to the electrodes, and consequently the electrical arcs strike between electrodes and the scrap which helps them to bore into the charge. To assist the process the scraps are layered and the heavy ones are placed at the bottom of the charge [2]. When the electrodes are sufficiently penetrated the high voltage is used to produce long arcs. The heat and radiation of the arcs melt the scrap and a pool of molten steel forms at the bottom of the furnace [2]. The process continues by three other stages, refining, de-slagging and tapping, which are briefly explained in the following.

The next stage is known as the refining stage. In this stage, oxygen is blown to the flat bath of molten charge at the hearth of the furnace. The influx of oxygen helps the removal of the unwanted impurities in the molten steel. The oxygen oxidizes the impurities (e.g., phosphorus, sulfur, aluminum, silicon, manganese and carbon), and consequently they leave the molten steel either in form of escaping gases or floating slag which forms on the top of the molten steel.

On the next stage, the floating slag is removed from furnace. This process should be done when the bath temperature is still comparatively low as some impurities will revert back to the molten steel in higher temperatures. The process is performed by opening of the slag door and tilting the furnace to allow the slag flow out.

Finally, at the last stage by adding bulk addition alloys and some de-oxidizers, steel with desired grade and temperature is achieved. At this point the steel is ready to be removed from furnace for further processing. At the end the tap-hole gets open and the furnace content is poured into the ladle and transferred to downstream operations.

1.2. Problem with EAF

The main problem with EAFs is their chaotic $v-i$ characteristics. In the beginning of the melting stage, the arcs are unstable and erratic hence the current and voltage characteristics are extremely nonlinear and unsteady (see Figure 1-1) [1][3]. As the temperature goes up and the atmosphere of the furnaces heats up the current and voltage characteristics become to some extent stable; however because of the nature of electrical arc and smelting process these characteristics remain chaotic and nonlinear [2][3]. Moreover high power consumption of the EAF causes significant power quality (PQ) disturbance in form of unwanted harmonics injection and voltage fluctuations on the power network [3]. The voltage flickering problem specially happens when the furnace rating (its power consumption) becomes comparable to the Short Circuit Capacity of the network [4]. These disturbances must be controlled otherwise they can have very undesirable impacts on the power network in a form of failure or damage to other electrical power consumers on the network [4]. To avoid this problem regulatory bodies placed some regulations in effect, and these regulations must be met [3]. For solving the aforementioned problem and satisfying the regulations some remedial approaches must be taken.

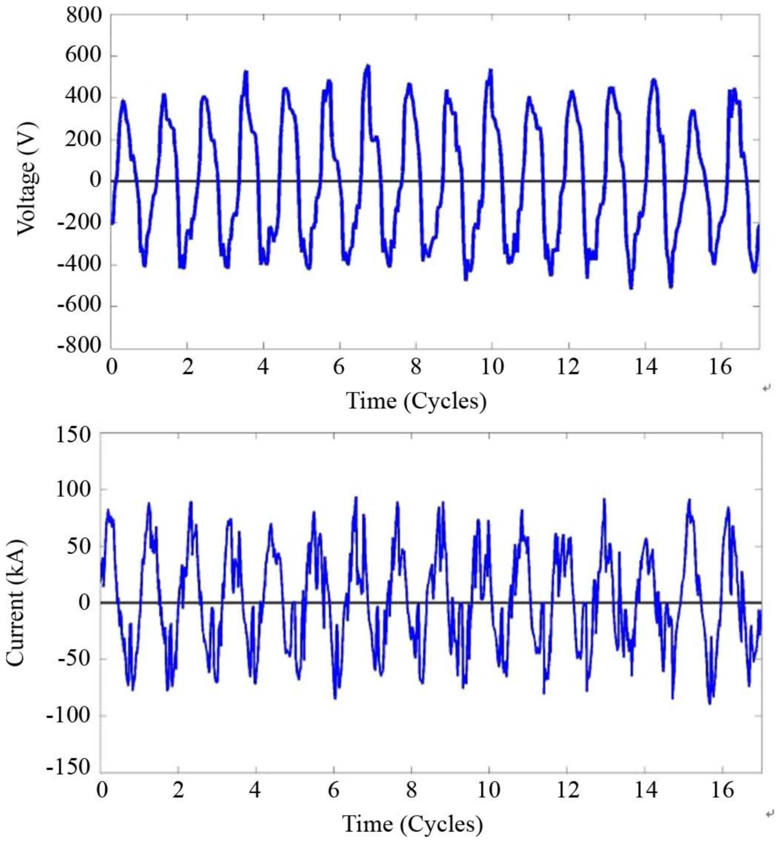


Figure 1-1: Measured Voltage and Current during penetration [3]

A typical mitigating remedy is the installation of a reactive power compensator [3]. The compensator tries to dynamically compensate the reactive power of the EAF, thus increasing the PQ [3]. Due to the nonlinear and chaotic characteristics of the EAF, it is extremely difficult to develop mathematical model to successfully represent them; however for PQ studies and subsequent design of compensation device it is necessary to have such a model [1][3].

1.3. Objective

The objective of this thesis is to develop three different models for predication of EAF characteristics. The reason for development of such a model is that it has practical use in PQ studies. These types of models will be used by electrical engineers for development of reactive power compensators to increase PQ of the power network. A Number of different techniques have been proposed to characterize EAFs. They can broadly be categorized to data driven methods (DDM) and explicit mathematical methods. Some example of these methods are Artificial Neural Networks (ANN), rule based fuzzy logic [1], Adaptive Neuro Fuzzy Interface System (ANFIS) [1], Radial Basis Function Neural Networks (RBFNN) [4], hybrid neuro-wavelet approach [3] and explicit mathematical models [5][6]. In this study the Extended Kalman Filter (EKF) will be used for development of a model for prediction of EAF’s arc voltage. By using EKF method the developed model will have the capability to adjust itself online, hence it can refine itself continuously.

1.4. Context

As mentioned in previous section, the main objective of thesis is the development of three different online and self-adjusting forecasting models. These models will be used for prediction of the voltage value of an Electrical Arc France (EAF). These models will utilize present value of the current and previous values of the voltage to do their predictions. The motivation behind this work can be expressed from two different angles.

Firstly, development of a forecasting model is a necessity for design and development of reactive power compensators. As it has been described in section 1.2, EAFs are extremely chaotic and nonlinear. These properties of EAFs will cause voltage fluctuations on power grid which can be very harmful for other power consumers. The job of the reactive power compensator is to regulate these fluctuations and eliminate them as much as possible. Therefore from industrial point of view development of these models are very useful.

Secondly, we know that the chaotic and nonlinear behaviour of EAFs are not fixed. That means the nonlinear and chaotic characteristics of EAFs will change over different stages of their operation [2][3]. Moreover, these characteristics are also under influence of other operating conditions of furnaces (e.g. wear of graphite electrodes or consistency of the charged materials, etc.). All of these situations present us with one important problem; that is any dataset collected from an EAF is just representative of a specific working condition. That means any model developed from this dataset could only be useful for the corresponding working condition, and probably would not do well for other working conditions. Development of an online self-adjusting model can eliminates this problem. An online self-adjusting model will perpetually correct and adapt itself to the new conditions; therefore the aforementioned problem will be solved effectively.

For development of such a model, Extended Kalman Filter (EKF) technique has been selected. Kalman filtering has very successful history in industrial applications (See section 1.5). EKF offers several benefits, however the most important one for us in this application is the online self-adjusting capability of the EKF.

The data set which has been used in this work is collected for a typical industrial electrical arc furnace utilized in steel production. Since the models, we will develop in this study are self-adjusting, the selection of a specific part of dataset is not very important. During their operation, these models can adapt themselves to new working conditions. For our experiments, one part of dataset with 5000 samples has been selected randomly for training and test phases. The details of our experiments can be found in Chapter 5.

1.5. Preliminary introduction to Kalman Filter and its applications

In 1960, Dr. Rudolf Emil Kalman, a Hungarian-born American electrical engineer, published his influential paper [9] on the subject of linear filtering. By his paper, Dr. Kalman made a significant contribution to the system estimation and the control theory. In his paper he proposed a sequential solution to the time varying filtering problem by eliminating the stationary requirement of the Weiner filter. In fact his pivotal idea was to apply the notion of state variables to the Wiener Filter [7]. By adding the assumption of finite dimensionally, he was able to derive Wiener Filter with much simpler mathematical complexity as it become accessible even to most undergraduate students.

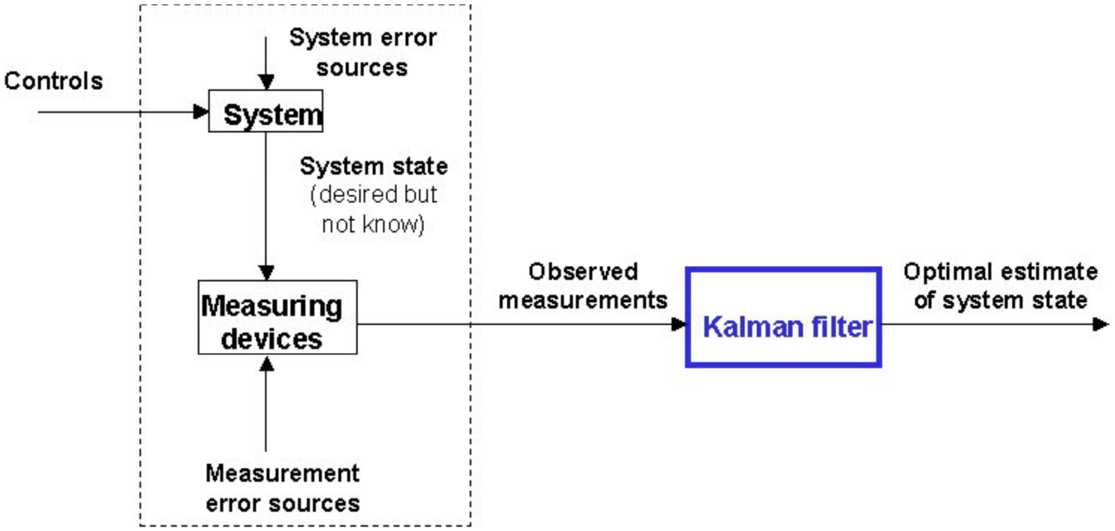

Controls
System error sources
System
System state (desired but not know)
Measuring devices
Measurement error sources
Observed measurements
Kalman filter
Optimal estimate of system state

Figure 1-2: Typical application of Kalman Filter [8]

The Kalman Filter is one of the most influential discoveries in signal processing and control theory in twentieth century [7]. Theoretically Kalman filter, also known as LQE (Linear Quadratic Estimator), is a state estimator for linear dynamic systems which are perturbed by white noise[1] [8]. Kalman Filter provides optimal system estimation through a recursive process by extracting the signal from its noise. Kalman Filter has vast application in engineering especially in the control of complex dynamic systems such as space crafts,

[1] In signal processing a random signal with constant Power Spectral Density (PSD) is known as white noise.

aircrafts, ships or continuous manufacturing processes [7]. In many of these cases it is not possible to measure all variables of the systems which we might want to control; in these situations the Kalman Filter helps us to infer missing information from indirect and noisy measurements. Kalman Filter can also be used, as in this study, to predict likely future state of a dynamic system such as the trajectory of celestial bodies, economical prices and indexes, or even course of water during flood season [7].

Figure 1-2 illustrates typical application of Kalman Filter. A dynamic system is driven by a set of inputs and/or control inputs. The external output of the system is observed by measuring devices which have some uncertainty by their nature. The job of the Kalman Filter is to estimate the state of the system given the inputs/controls and the observed uncertain state. To explain it in simple terms, the Kalman Filter tries to eliminate uncertainty of the output of the system as much as possible. Generally any system which can be fitted into this scheme can be modeled by Kalman Filter if some certain mathematical properties are met.

From historical point of view, an early application of Kalman filter was in trajectory estimation and navigation control systems [7][10]. During early 60s NASA was involved with the Apollo project, and Ames Research Center of NASA was wrestling with trajectory and navigation problem. Since Ames engineers and researchers were already heavily involved with the field and contemplated about a filtering approach to their problem, Dr. Kalman found them to be very receptive to his work. From there by collaboration of Schmidt and Kalman, the concept of linear perturbation of a non-linear system applied to the Kalman Filter and Extended Kalman Filter (EKF) was developed. From there by influence of Schmidt Kalman Filter became an important part of the Apollo guidance system [7]. The Kalman filter effectively solved the data fusion problem between on-board sensor data and radar data. Soon other researchers and engineers followed the lead and the Kalman Filter become an integral part of almost any onboard trajectory estimation and navigation control systems [7][10].

The Kalman Filter has variety of applications in different fields. One of the application of the Kalman Filter in the computer vision is the "visual tracking" [11]. Since this particular application is quite interesting and also very intuitive, we will discuss it here briefly to convey a preliminary understanding of the Kalman filter capabilities and utilizations.

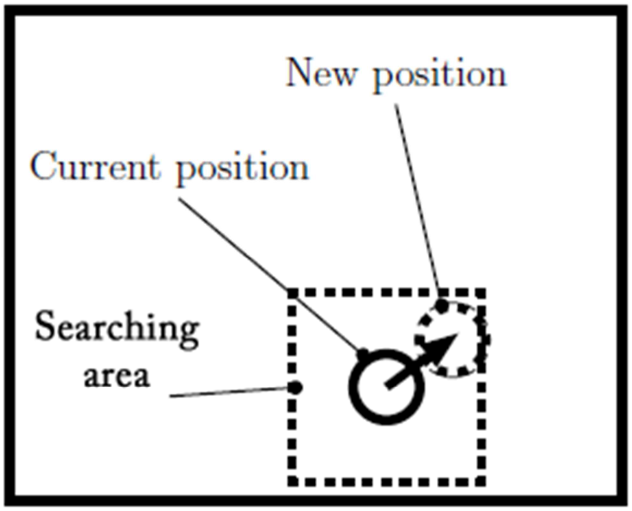


Figure 1-3: New position is searched within the searching area [11]

Visual tracking is the process of determining the location of a specific object over time in the sequence of images [11]. The object can be anything, a car in a video captured by a traffic camera or a head in a video streaming by web-cam. The process begins with the selection of some specific features within the target object (e.g. corners, specific patterns, etc.). The relative position of these features gives us the orientation and the positon of the target. Now in the next image the orientation and the position of the target must be recalculated. Ideally the tracking algorithm must be able to find the position of the target anywhere within the new image. Nevertheless this approach is not practical since it is computationally expensive especially in the cases of real-time applications. For solving this problem the searching area should be limited (see Figure 1-3).

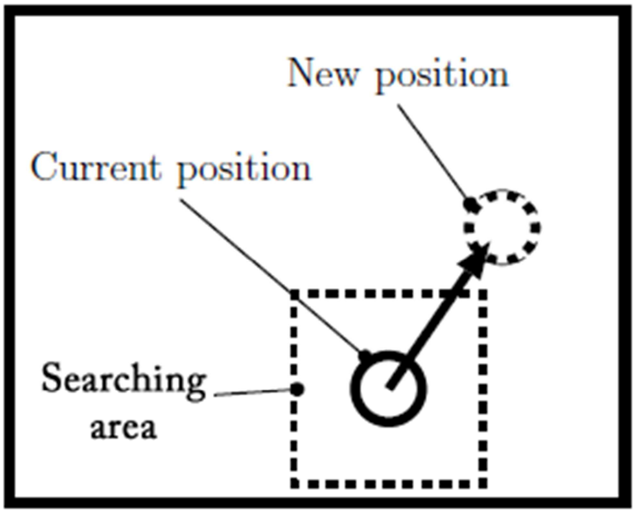


Figure 1-4: Target outside of searching area [11]

Clearly adaptation of the “searching area” approach can help us reduce the complexity of the tracking problem, but on the other hand it will cause another problem. If the speed of the target is too high or the frame rate is too low or the searching area is too small, the target can be well outside of the area (see Figure 1-4). For resolving the issue we have to predict the probable location of the target, and set the center of the searching area in that location. Normally the previously obtained location is used for guessing the new “would-be” location, but since the tracking algorithms are not perfect[2] the previous location itself is not accurate. In other words what we are trying to do here is the prediction of the new location based on an inaccurate previous one, and that is the task which the Kalman Filter has been designed to accomplish. The Kalman Filter is developed to handle these types of uncertainties and give us better predictions/estimations. Some other applications of Kalman Filter in the computer vision are also worth mentioning. Among them we can name de-nosing [14], depth measurement stabilization [15], cluster tracking [16] and sensor and optical data fusion [17].

Kalman Filter has also been used as a training algorithm [12][13]. In a higher view training is nothing more than an estimation of the free parameters of a model by an estimator. Kalman Filter can be utilized for this purpose. For instance the Kalman Filter has been used for training of the Neural Networks [18]. The most well-known training algorithm of the Neural Networks is the back-propagation algorithm (BPA). The BPA uses first-order stochastic gradient decent method to adjust the free parameters of the neural network; however in some situations it can be very slow. Several attempts have been made to address this problem such as classical nonlinear programming [20], or adaptation of Hessian matrix [19]. The Extended Kalman Filter (EKF) as full second order gradient decent method has also been used for speeding up the BPA [18]. By using EKF, more information is extracted from the surface of

[2] Some phenomena such as occlusions, changes in lighting and shadows can affect the appearance of the target hence causing difficulty in tracking.

the error function hence the training becomes more efficient and faster [12][19]. Another example of utilization of the Kalman Filter in training algorithms can be found in Fuzzy Logic Interface (FIS) [12]. In his work, Dan Simon used the Extended Kalman Filter algorithm to train a fuzzy system. His objective was to model fuzzy estimator for motor current windings.

Today the Kalman filter has numerous applications in different scientific and engineering fields. Some applications which have been discussed here were trajectory estimation, navigation, vehicle control (especially aircraft and spacecraft) and visual tracking. Furthermore, we briefly discussed the application of the Kalman Filter as a derivative based method in the training algorithms. Here in this study we will try to use it as a training/adjusting tool for development of a model for prediction of future value of a typical EAF time-series.

1.6. Contribution

The main contribution of this study is the development of an online self-adjusting predictive model. Models which have been previously developed by other researchers do not have this capacity. The working conditions of an EAF can vary depending on the operation stage, wear of graphite electrodes, consistency of charged scrap metals, charge level, etc. All of these varying properties will cause difficulty in obtaining a universal training set. Any training dataset would eventually be representative of some of these conditions and cannot cover all working conditions. Consequently the output of models, trained by specific training set, is unknown if the conditions differ for training set. In this thesis we propose a self-adjusting predictive model based on Kalman Filtering to mitigate this problem.

1.7. Thesis overview

This thesis is organized in six chapters. In chapter one, we talked about Electrical Arc Furnaces, and we give a brief introduction about the operation of the furnace. We also provide a preliminary introduction to Kalman Filtering and its applications in this chapter.

In the second chapter we will discuss about Kalman filtering and derivation of Kalman Filter. If readers are familiar with the Kalman Filtering they can skip this chapter.

In the third chapter we will discuss about the previous works on EAF modeling. The previous models have been categorized into two broad groups of white and black box models each with corresponding subcategories.

In the fourth chapter, we will discuss about our methodology and how the Kalman Filtering can be utilized for the problem in hand. In this chapter three models are proposed, and the interworking of these models are depicted and explained.

In the fifth chapter, the experimental results of the implemented models will be discussed with graphs and tables.

Finally in the last chapter, chapter six, the conclusion will be drawn. At the end some possible avenues for future works of this kind will be discussed.

2. KALMAN FILTER

2.1. Estimation theory and Kalman Filter

The Kalman Filter belongs to the class of mathematical techniques best known as estimation methods. The development of the mathematical concepts of the estimation theory goes well back to ancient time. Figure 2-1 shows the last 500 years of its development. Carl Friedrich Gauss (1777-1855) is the one who is generally recognized for the discovery of the first method for extracting an optimal estimation from noisy data [7]. The inevitability of noise in measured data was well-known since the time of Galileo Galilei (1564-1642), but it was Gauss who first formally proposed a mathematical method to deal with them. Today this method is commonly known as least square method (See Appendix A). In fact this method is a special case of the Kalman Filter [22].

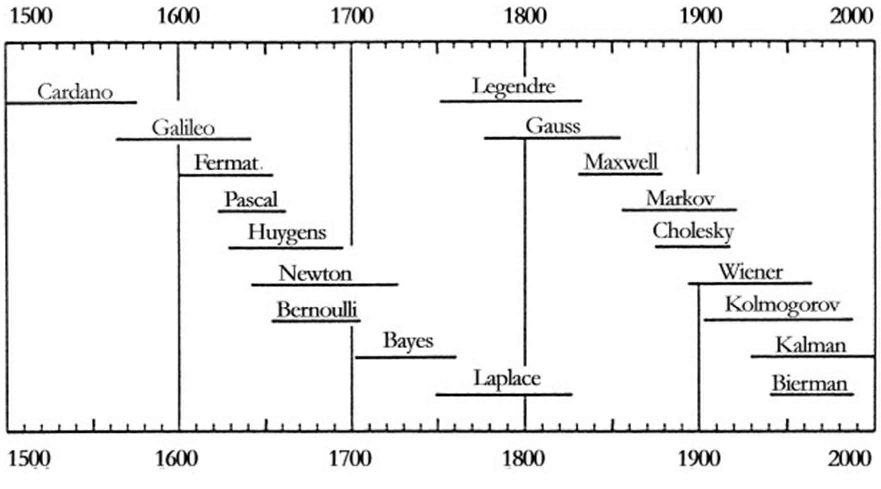


Figure 2-1: Historical progress of estimation theory [7]

The development of the Kalman Filter is the logical progression from least square method and other underlying foundations (See Figure 2-2). Kalman Filter works as an estimator for stochastic dynamic systems. A Dynamic system is a time variant systems which its properties can change over time (See Appendix B). Stochastic systems are another group of systems in which we have some random properties. These random properties have impact on behaviour of the system (See Appendix C). Kalman Filter essentially provides a mathematical technique to estimate stochastic dynamic systems through a probabilistic approach (Basic background of probability theory is provided in Appendix D).

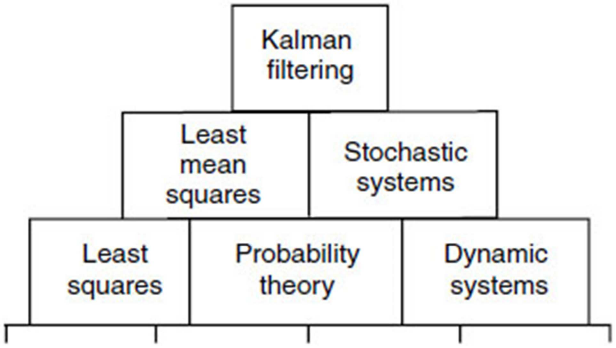


Figure 2-2: Mathematical foundation of the Kalman Filter [7]

The term “filter” in Kalman Filter name, refers to the intended capacity of Kalman Filter for elimination of the noise from real signal which we try to estimate. Indeed in the case of linear Kalman Filtering, this is done optimally in mathematical sense. Nevertheless the “filtering” term does not do justice to the capability of Kalman Filter, and in fact what the Kalman Filter does is well beyond that scope. Kalman Filter also includes the solution of an “inversion problem” and we briefly explain what we mean by that.

Inverse or inversion problems are a class of mathematical problems in which the observed data are used to extract some information about a physical system [21]. In almost all cases the observed data does not give us a determined system hence an objective function needs to be defined to solve the problem. The objective function is used for estimation of the system parameters. The inverse problem has various applications in many branches of science and engineering including, but not limited to machine learning, computer vision, natural language processing statistics, statistical inference, geophysics, medical imaging, remote sensing, ocean acoustic tomography, non-destructive testing, astronomy and physics [21]. In the case of the Kalman Filter, it essentially inverts and estimates the functional relationship of independent variables of the system (system’s state variables) with observed/measured variables.

2.2. Kalman filtering

The general form of stochastic dynamic systems can be presented as following (See Appendix C)

$$x_{k+1} = f_k(x_k, u_k, w_k) \qquad k \geq 0 \tag{2.1}$$

$$z_k = h_k(x_k, u_k, v_k) \tag{2.2}$$

The equation (2.1) is known as the system equation. The output of this equation provides us with the state of the system in next time step. The other equation, (2.2), is known as measurement equation. This equation gives us a secondary value (measured value of z_k) which depends on the same input vectors as the equation (2.1). The only difference here is that the random vector w_k is replace with v_k, however both of these vector represent error in the system, while the first on represent error in the system equation, and the next one error in the measurement equation.

In a high overview, what we are trying to do with Kalman Filtering is that we use the system equation to predict the value of the state vector x_{k+1}. Then in the next step, we measure/observe the value of z_{k+1} from the system. This value will be used in Kalman filtering algorithm to correct our prediction (x_{k+1}), and achieve better value closer to real value of the state vector. Now that the concept of the Kalman Filter is clear, we continue to formulate the Kalman filtering problem.

Assuming that $f_k(\dots)$ and $h_k(\dots)$ are linear, we can rewrite equation (2.1) and (2.2) in the linear form respectively

$$x_{k+1} = A_k x_k + B_k u_k + G w_k \qquad k \geq 0 \tag{2.3}$$

$$y_k = C_k x_k + v_k \tag{2.4}$$

where

$x(k) \in R^n$ is the system state vector,

$A_k x_k + B_k u_k + G w_k$ is the linear form of system equation ($f_k(\dots)$),

$u(k) \in R^m$ is the deterministic input or control vector,

$w(k) \in R^n$ is the vector conveying the system error,

$y(k) \in R^r$ is known as observation vector,

$C_k x_k + v_k$ is the linear form of measurement function,

and finally, $v(k) \in R^r$ is the vector which constitutes the error associated with measurement.

$\{w_k\}$ and $\{v_k\}$ are assumed to be sequence of white, zero mean, Gaussian noise hence the expected value of w_k and v_k are

$$E[w_k] = E[v_k] = 0 \tag{2.5}$$

the joint covariance matrix of w_k and v_k is considered to be known

$$E\left[\begin{pmatrix} w_k \\ v_k \end{pmatrix} \quad \begin{pmatrix} w_k^T & v_k^T \end{pmatrix}\right] = \begin{bmatrix} Q_k & 0 \\ 0 & R_k \end{bmatrix} \tag{2.6}$$

the initial state x_0 is Gaussian random vector with expected/mean value of,

$$E[x_0] = \bar{x}_0 \tag{2.7}$$

and the covariance matrix,

$$E[(x_0 - \bar{x}_0)(x_0 - \bar{x}_0)^T] = \Sigma_0 \quad (2.8)$$

Σ_0 is the initial covariance matrix of the state vector x_0 in the beginning of the algorithm (time step 0). This matrix must be set by try and error approach through judgment of an expert, therefore we assume it to be known. Now having the input/control set $\{u_0, u_1, \dots, u_{k-1}\}$ and the measurement set $\{y_1, \dots, y_k\}$ the goal is to obtain the best estimate of $x(k)$. By taking a Bayesian approach the filter tries to calculate an optimal value of the system's state recursively. The term optimal conveys that minimization of state estimation error. It is done by propagating the conditional probability density function of the desired quantities (state estimates), given the known information coming from measurement and input. Hence the filter evaluates and propagates the following conditional PDF for increasing values of k (The concept of conditional probability has been explained in Appendix D)

$$p(x_k | y_1, \dots, y_k, u_0, \dots, u_{k-1}) \quad (2.9)$$

For given time instance of k consider that the sequence of previous inputs and the sequence of previous measurements are denoted by

$$U_0^{k-1} = \{u_0, u_1, \dots, u_{k-1}\} \quad (2.10)$$

$$Y_1^K = \{y_1, y_2, \dots, y_k\} \quad (2.11)$$

then the entire filtering process can be described [8][27] as the following evolutionary process,

- **Given:** x_0

    - System apply w_0 (caused by the uncertainty of the system),
    - We apply u_0 (as input or control signals),
    - The system moves to state x_1,
    - We make a measurement y_1,
- **Question:** Which is the best estimate of x_1? **Answer:** Is Obtained from $p(x_1|Y_1^1, U_0^0)$,
    - System apply w_1,
    - We apply u_1,
    - The system moves to state x_2,
    - We make a measurement y_2,
- **Question:** Which is the best estimate of x_2? **Answer:** Is Obtained from $p(x_2|Y_1^2, U_0^1)$,
    - System apply w_2,
    - We apply u_2,
    - The system moves to state x_3,
    - We make a measurement y_3,
- **Question:** Which is the best estimate of x_k? **Answer:** Is Obtained from $p(x_k|Y_1^k, U_0^{k-1})$,
    - System apply w_k,
    - We apply u_k,
    - The system moves to state x_{k+1},
    - We make a measurement y_{k+1},

For the system represented by equations (2.3) and (2.4), the Kalman Filter will estimate the state vector with minimum mean-square error. Indeed when the system and measurement noises are white and Gaussian and the x_0 is Gaussian vector then the conditional probability density functions $p(x_k|Y_1^k, U_0^{k-1})$ are Gaussian for any k and consequently we can express it as the following [8]

$$p\left(x_k \middle| Y_1^k, U_0^{k-1}\right) \sim \mathcal{N}(\hat{x}(k|k), P(k|k)) \qquad (2.12)$$

where $\hat{x}(k|k)$ and $P(k|k)$[3] represent first and second moment of the probability distribution respectively[4]

$$\hat{x}(k|k) = E[x(k)\middle| Y_1^k, U_0^{k-1}] \qquad (2.13)$$

$$P(k|k) = E[(x(k) - \hat{x}(k|k))(x(k) - \hat{x}(k|k))^T|Y_1^k, U_0^{k-1}] \qquad (2.14)$$

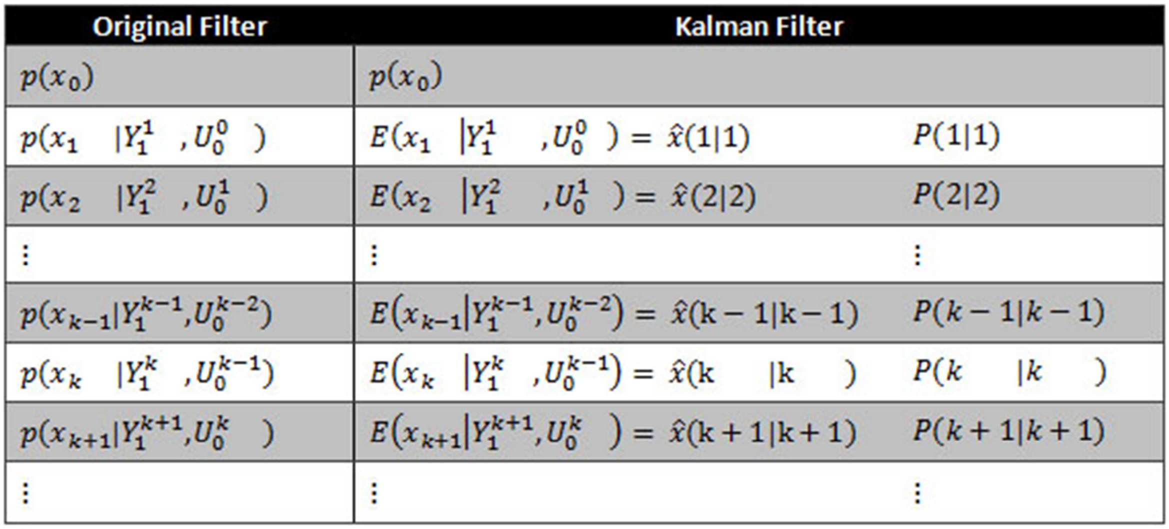
In fact with the Kalman Filter instead of propagating the entire conditional PDF we only propagate the mean and the covariance matrix of the distribution (Figure 2-3). Therefore the dynamic of the Kalman Filter is the general transformation from $p\left(x_k \middle| Y_1^k, U_0^{k-1}\right)$ to $p\left(x_{k+1} \middle| Y_1^{k+1}, U_0^k\right)$

$$p\left(x_k \middle| Y_1^k, U_0^{k-1}\right) - - - - \rightarrow p\left(x_{k+1} \middle| Y_1^{k+1}, U_0^k\right) \qquad (2.15)$$

[3] The meaning of the conditional term "$k|k$" is that given time k we have estimate value of $\hat{x}$ and P at the time k according to the formula. As we go ahead in this section, we will see at some point we need to estimate the value of $\hat{x}$ and P one time step ahead. We express it as following $\hat{x}(k+1|k)$ and $P(k+1|k)$. That means that at given time k we have the estimate value of $\hat{x}$ and P for time $k+1$.

[4] The first and second moments are mean and covariance matrix of the conditional PDF respectively. The $P(k|k)$ is also known as error covariance matrix.

where both probability distributions are Gaussian and the input and the measurement vectors are available at time k and $k+1$.



Original Filter	Kalman Filter	
$p(x_0)$	$p(x_0)$	
$p(x_1 \mid Y_1^1, U_0^0)$	$E(x_1 \mid Y_1^1, U_0^0) = \hat{x}(1\mid1)$	$P(1\mid1)$
$p(x_2 \mid Y_1^2, U_0^1)$	$E(x_2 \mid Y_1^2, U_0^1) = \hat{x}(2\mid2)$	$P(2\mid2)$
$\vdots$	$\vdots$	$\vdots$
$p(x_{k-1} \mid Y_1^{k-1}, U_0^{k-2})$	$E(x_{k-1} \mid Y_1^{k-1}, U_0^{k-2}) = \hat{x}(\mathrm{k}-1\mid\mathrm{k}-1)$	$P(k-1\mid k-1)$
$p(x_k \mid Y_1^k, U_0^{k-1})$	$E(x_k \mid Y_1^k, U_0^{k-1}) = \hat{x}(\mathrm{k}\mid\mathrm{k})$	$P(k\mid k)$
$p(x_{k+1} \mid Y_1^{k+1}, U_0^k)$	$E(x_{k+1} \mid Y_1^{k+1}, U_0^k) = \hat{x}(\mathrm{k}+1\mid\mathrm{k}+1)$	$P(k+1\mid k+1)$
$\vdots$	$\vdots$	$\vdots$

Figure 2-3: Propagation of conditional PDF in KF [8]

The transition depicted in equation (2.15) is a two-step transition rather than one step. Therefore instead of jumping from the evaluation of $p\left(x_k \middle| Y_1^k, U_0^{k-1}\right)$ to the evaluation of $p\left(x_{k+1} \middle| Y_1^{k+1}, U_0^k\right)$, we first evaluate $p\left(x_{k+1} \middle| Y_1^k, U_0^k\right)$ and from there we will evaluate $p\left(x_{k+1} \middle| Y_1^{k+1}, U_0^k\right)$

$$p\left(x_k \middle| Y_1^k, U_0^{k-1}\right) - - - - \rightarrow p\left(x_{k+1} \middle| Y_1^k, U_0^k\right) \tag{2.16}$$

$$p\left(x_{k+1} \middle| Y_1^k, U_0^k\right) - - - - \rightarrow p\left(x_{k+1} \middle| Y_1^{k+1}, U_0^k\right) \tag{2.17}$$

These two transitions, (2.16) and (2.17), are known as the prediction and the filtering/update cycles respectively (Figure 2-4).

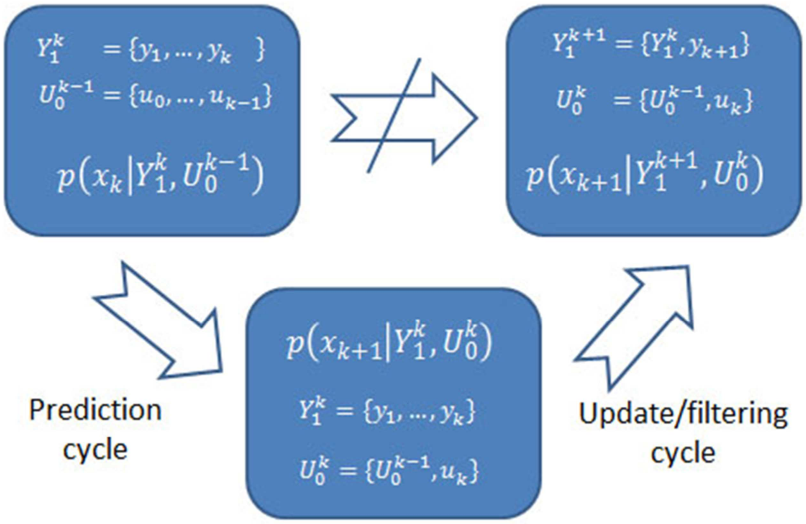

$Y_1^k \quad = \{y_1, \ldots, y_k \quad \}$
$U_0^{k-1} = \{u_0, \ldots, u_{k-1}\}$
$p(x_k | Y_1^k, U_0^{k-1})$
$Y_1^{k+1} = \{Y_1^k, y_{k+1}\}$
$U_0^k \quad = \{U_0^{k-1}, u_k\}$
$p(x_{k+1} | Y_1^{k+1}, U_0^k)$
$p(x_{k+1} | Y_1^k, U_0^k)$
$Y_1^k = \{y_1, \ldots, y_k\}$
$U_0^k = \{U_0^{k-1}, u_k\}$
Prediction cycle
Update/filtering cycle

Figure 2-4: Two cycles of the Kalman Filter dynamics [8]

It should be noted that in the first cycle the evaluation of $p\left(x_{k+1} \middle| Y_1^k, U_0^k\right)$ is done at time instant $k+1$, however the estimation of $x(k+1)$ is "*before*" the observation of $y(k+1)$. That is why we call this cycle the "prediction cycle". In the next cycle we are still at time instant $k+1$, however the predicted value of $x(k+1)$ is updated "*after*" the observation of $y(k+1)$ hence this cycle is called the "update/filtering cycle". By looking at Figure 2-5 one can understand how the recursive application of the prediction and the update/filtering cycles works.

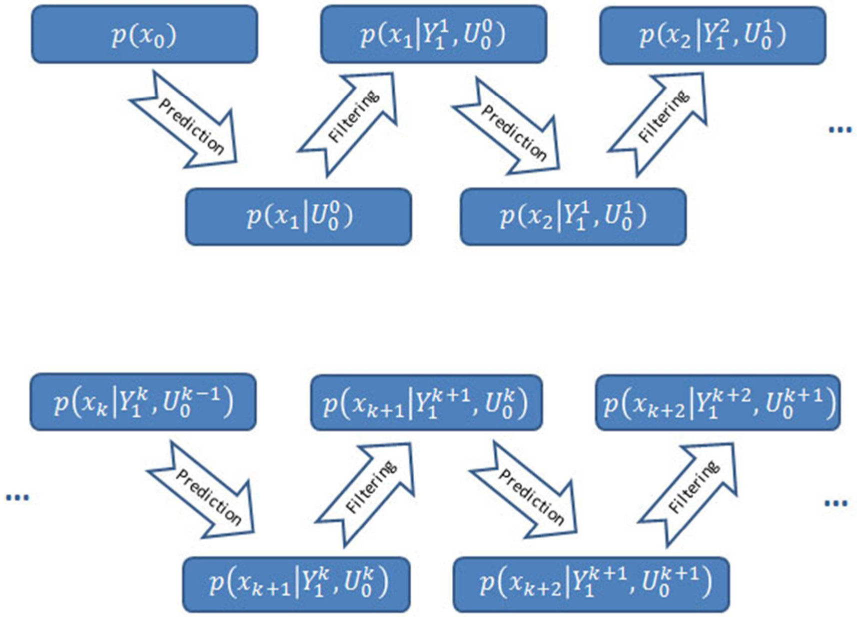


Figure 2-5: Consecutive cycles of KF dynamics [8]

2.2.1. *Kalman Filter derivation*

Derivation of the Kalman Filter formulation is straight forward. Our goal is to calculate $\hat{x}(k+1|k+1)$ estimation. As explained $p\left(x_k\middle|Y_1^k,U_0^{k-1}\right)$ and $p\left(x_{k+1}\middle|Y_1^k,U_0^k\right)$ can be given as the following

$$p\left(x_k\middle|Y_1^k,U_0^{k-1}\right) \sim \mathcal{N}(\hat{x}(k|k),P(k|k)) \tag{2.18}$$

$$p\left(x_{k+1}\middle|Y_1^k,U_0^k\right) \sim \mathcal{N}(\hat{x}(k+1|k),P(k+1|k)) \tag{2.19}$$

where

$$\hat{x}(k|k) = E[x(k)|Y_1^k, U_0^{k-1}] \tag{2.20}$$

$$\hat{x}(k+1|k) = E[x(k+1)|Y_1^k, U_0^k] \tag{2.21}$$

and

$$P(k|k) = E[\big(x_k - \hat{x}(k|k)\big)\big(x_k - \ \hat{x}(k|k)\big)^T|Y_1^k, U_0^{k-1}] \tag{2.22}$$

$$P(k+1|k) = E[\big(x_{k+1} - \hat{x}(k+1|k)\big)\big(x_{k+1} - \ \hat{x}(k+1|k)\big)^T|Y_1^k, U_0^k] \tag{2.23}$$

Assuming that $\hat{x}(k|k)$ and $P(k|k)$ are known at this stage, first we evaluate $p\big(x_{k+1}\big|Y_1^k, U_0^k\big)$. For doing so, $\hat{x}(k+1|k)$ and $P(k+1|k)$ should be calculated. Now by substitution of equation (2.3) in (2.21) and knowing that $E[w_k\big|Y_1^k, U_0^k] = 0$ we have

$$E[x_{k+1}\big|Y_1^k, U_0^k] = A_k E[x_k\big|Y_1^k, U_0^k] + B_k E[u_k\big|Y_1^k, U_0^k] + GE[w_k\big|Y_1^k, U_0^k] \tag{2.24}$$

$$\hat{x}(k+1|k) = A_k \hat{x}(k|k) + B_k u_k \tag{2.25}$$

By defining the prediction error as

$$\tilde{x}(k+1|k) \triangleq x(k+1) - \hat{x}(k+1|k) \tag{2.26}$$

and also the filtering error as

$$\tilde{x}(k|k) \triangleq x(k) - \hat{x}(k|k) \tag{2.27}$$

and working with these equations we eventually have[5]

$$P(k+1|k) = A_k P(k|k) A_k^T + G_k Q_k G_k^T \tag{2.28}$$

With the same method we can show that the evaluation of $p(y_{k+1}|Y_1^k, U_0^k)$ leads to

$$E[x_{k+1}\tilde{y}^T(k+1|k)] = P(k+1|k) C_{k+1}^T \tag{2.29}$$

and finally using equation (2.29) the evaluation of $p(x_{x+1}|Y_1^{k+1}, U_0^k)$ yields

$$\hat{x}_{(k+1|k+1)} = \hat{x}_{(k+1|k)} + K_{(k+1)}[y_{(k+1)} - C_{(k+1)}\hat{x}_{(k+1|k)}] \tag{2.30}$$

$$P_{(k+1|k+1)} = P_{(k+1|k)} - K_{(k+1)} C_{k+1} P_{(k+1|k)} \tag{2.31}$$

where $K(k+1)$ is called “Kalman Gain” and defined as

$$K_{(k+1)} = P_{(k+1|k)} C_{k+1}^T \left[C_{k+1} P_{(k+1|k)} C_{k+1}^T + R\right]^{-1} \tag{2.32}$$

2.2.2. *Kalman Filter algorithm*

The Kalman Filter algorithm is quite simple and can be followed from previous section nevertheless we reiterate it here again to make it clearer. Assuming that a linear time-varying system is governed by the following two equations,

[5] The detailed steps can be found on various references (e.g. [8])

$$x_{k+1} = A_k x_k + B_k u_k + G w_k \qquad k \geq 0 \tag{2.33}$$

$$y_k = C_k x_k + v_k \tag{2.34}$$

the algorithm goes as following,

Step 0 (Initial condition)

At step 0 we collect the necessary information from previous step which are the posterior estimate and error covariance matrix; $\hat{x}(k|k)$ and $P(k|k)$ respectively.

Step 1 (Prediction)

$$\hat{x}_{k+1|k} = A_k \hat{x}_{k|k} + B_k u_k \tag{2.35}$$

$$P_{k+1|k} = A_k P_{k|k} A_k^T + G_k Q_k G_k^T \tag{2.36}$$

Note that at the very first step we consider that the prior estimate and error covariance matrix are

$$\hat{x}(0|-1) = \bar{x}_0 \tag{2.37}$$

$$P(0|-1) = \Sigma_0 \tag{2.38}$$

hence we don’t need any information from none-exiting last step ($\hat{x}_{(-1|-1)}$ and $P_{(-1|-1)}$). $\bar{x}_0$ and Σ_0 are average and covariance matrices of state vector x at time step 0 respectively, and they are set by try and error approach through expert knowledge.

Step 2 (Filtering)

$$K_{k+1} = P_{k+1|k}C_{k+1}^T\left[C_{k+1}P_{k+1|k}C_{k+1}^T + R\right]^{-1} \tag{2.39}$$

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1}[y_{k+1} - C_{k+1}\hat{x}_{k+1|k}] \tag{2.40}$$

$$P_{k+1|k+1} = [I - K_{k+1}C_{k+1}]P_{k+1|k} \tag{2.41}$$

where the I is identity matrix. For the next iteration of the algorithm we take $\hat{x}_{k+1|k+1}$ and $P_{k+1|k+1}$ as $\hat{x}_{k|k}$ and $P_{k|k}$ respectively and will repeat the process again.

2.3. Extended Kalman filtering

In section 2.2 we discussed about the Kalman Filter in the context of linear dynamic systems governed by linear stochastic difference equations. Now the question is what we should do if the system is not linear. Similar to a Tylor series, what can be done here is the linearization of the estimation around the current estimate using partial derivative of nonlinear process and measurement functions. We consider a nonlinear dynamic system without external input as below

$$x_{k+1} = f_k(x_k) + w_k \tag{2.42}$$

$$y_k = h_k(x_k) + v_k \tag{2.43}$$

where

$$
\begin{gathered}
x_k \in \mathcal{R}^n \\
y_k \in \mathcal{R}^r \\
v_k \in \mathcal{R}^r \\
w_k \in \mathcal{R}^n \\
f_k(x_k): \mathcal{R}^n \rightarrow \mathcal{R}^n \\
h_k(x_k): \mathcal{R}^n \rightarrow \mathcal{R}^r
\end{gathered}
\tag{2.44}
$$

$\{w_k\}$ and $\{v_k\}$ are white Gaussian, independent and random process and measurement noises with zero means and covariance matrices

$$
E[v_k v_k^T] = R_k, \qquad E[w_k w_k^T] = Q_k \tag{2.45}
$$

and finally x_0 is the initial state of the system as a Gaussian random vector

$$
x_0 \sim \mathcal{N}(\bar{x}_0, \Sigma_0) \tag{2.46}
$$

Similar to linear Kalman Filter, $\bar{x}_0$ and Σ_0 are average and covariance matrices of state vector x at time step 0 respectively, and they are set by try and error approach through expert knowledge. As the derivation of the EKF is lengthy we do not discuss it here for further study one can refer to [8], or for more generalized form to [28].

2.3.1. *Extended Kalman Filter algorithm*

The algorithm of the EKF is very similar to the KF algorithm,

Step 0 (Initial condition)

At step 0 we collect the necessary information from previous step which are the posterior estimate $\hat{x}(k|k)$ and matrix $P(k|k)$. It should be noted that in the EKF $P(k|k)$ is no longer representative of the error covariance.

Step 1 (Prediction)

$$\hat{x}_{k+1|k} = f_k(\hat{x}_{k|k}) \tag{2.47}$$

$$P_{k+1|k} = F_k P_{k|k} F_k^T + Q_k \tag{2.48}$$

where F_k is Jacobean matrix of $f_k(.)$ at $\hat{x}_{k|k}$ and denoted by

$$F_k = \nabla f_k|_{\hat{x}_{k|k}} \tag{2.49}$$

The Jacobian matrix is the matrix of first partial derivatives of a vector-valued function. What the Jacobian matrix does here is the linearization of the process function around $\hat{x}_{k|k}$. The Jacobean matrix is defined as the following,

For a vector-valued function $\mathcal{F}(x)$,

$$\mathcal{F}(x): \mathcal{R}^n \rightarrow \mathcal{R}^m \tag{2.50}$$

The Jacobian matrix is

$$J = \frac{d\mathcal{F}}{dx} = \begin{bmatrix} \frac{\partial \mathcal{F}}{\partial x_1} & \cdots & \frac{\partial \mathcal{F}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \tag{2.51}$$

Step 2 (Filtering)

For filtering step first we calculate the Kalman Gain as defined below

$$K_{k+1} = P_{k+1|k} H_{k+1}^T \left[H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1} \right]^{-1} \tag{2.52}$$

where H_{k+1} is Jacobean matrix of $h_k(.)$ at $\hat{x}_{k+1|k}$ and denoted by

$$H_{k+1} = \nabla h_{k+1}|_{\hat{x}_{k+1|k}} \tag{2.53}$$

the next step is the calculation of the posterior estimate

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1}[y_{k+1} - h_{k+1}(\hat{x}_{k+1|k})] \tag{2.54}$$

and $P_{k+1|k+1}$ matrix

$$P_{k+1|k+1} = [I - K_{k+1} H_{k+1}] P_{k+1|k} \tag{2.55}$$

where I is the identity matrix.

For the next iteration of the algorithm we take $\hat{x}_{k+1|k+1}$ and $P_{k+1|k+1}$ as $\hat{x}_{k|k}$ and $P_{k|k}$ respectively and repeat the process again.

3. ELECTRICAL ARC FURNACE MODELING

3.1. On Mathematical modeling of physical phenomena

Mathematical modeling of a system is description of the system by mathematical language and concepts. The origin of word “Modeling” comes from Latin world “modellus” describing a typical human behaviour of copying with reality [29]. The history of the modeling goes back to the ancient civilizations of the Middle East and Greek. Perhaps the first recognizable models were “numbers” and development of ability to count and record on primitive media (e.g. bones or cave’s walls). From there humanity has come a long way in development of its ability to make better understanding of the world by mathematical modeling.

Models can be classified in many different categories depending on how they are analyzed. Some examples of these categorizations are “linear vs. nonlinear”, “continuous vs. discrete”, “static vs. dynamic”, “deterministic vs. stochastic”, “implicit vs. explicit”, “deductive vs. inductive”, “white box vs. black box”, etc. Most of these categorization schemes are self-explanatory, however we discuss the last three here briefly.

Explicit or implicit modeling is often referred to numerical solution schemes that are implemented in modeling. In the explicit methods the state of a system at a later time ($x_{k+\Delta k}$) is calculated from the state of the system at the current time (x_k) through an explicit equation,

$$x(k + \Delta k) = F(x(k)) \tag{3.1}$$

while in implicit method the solution is obtained from an equation involving both current and later state of the system,

$$G\big(x(k + \Delta k), x(k)\big) = 0 \tag{3.2}$$

The main difference of the explicit and implicit methods is the stability of the solution with regard to the size of the time step (Δk). The implicit method tends to be unstable on large time steps while the implicit method provides stable solution using an iterative numerical method. Selection between these two approaches is based on the nature of the problem on hand [30].

A deductive model is developed based on the theory while an inductive model arises from empirical information and generalization from them. In practice no model is purely deductive or inductive but resides somewhere in between. These terms are mostly used for the models developed for the human science applications (e.g. sociology, management and, etc.) [31].

Finally the last pair reflects a generalized categorization of the mathematical modeling which arises from the way the model is seen. A black box is a model that the user or the developer does not have any knowledge of its internal workings therefore it is merely seen as an input to output model. Here the most immediate question is if the internal working of the model is completely unknown how the model is developed at first place. The answer to this question is the architecture of the model is partly or completely known; however the parameters of the model are unknown initially and will be learned through a learning process. Considering a model with numerous parameters it would apparently be very difficult to infer any meaningful relation between parameters and the outputs therefore the internal working of the model is assumed to be unknown. On the other hand in the white box modeling a priori information on the system is available, and this knowledge is used for establishment of relationship between inputs and outputs. The developed models are directly based on these relationships therefore models are transparent, and their internal working are fully understood. It is quite clear that in the real world applications no model is absolutely white or black, thus the practical models are somewhere between the two ends of the spectrum.

3.2. Electrical Arc Furnace Modeling

In this section we use the general "white box" and "black box" categories to classify the previous works on the EAF modeling. Nevertheless, it should be mentioned again that no model is fully white or black, but at least by this categorization one can see how the researchers try to approach the problem.

3.2.1. *White box models of EAFs*

A Large number of models can be considered as white box models. In fact there are numerous papers in which the authors try to adapt an explicit framework for modeling of the dynamic behaviour of the EAFs. There are a variety of techniques which can be implemented as an explicit framework for EAF modeling. Most of these models take some oversimplifying assumptions. Nevertheless some of these methods are reviewed in this section.

Differential equations based on Kirchoff’s Curent Law

In their work [32], Benoit Boulet et al. pursue an analytical approach to model the dynamic of EAFs. For the purpose of an efficient power controlling a good dynamic model must be developed. As first step, Benoit Boulet et al. utilized the Kirchoff’s Curent Law (KCL) to relate currents and voltages. With some assumptions on the resistance of circuit components, they developed the current equations of the electrodes. Using these equations, they developed an open-loop system by Simulink/MATLAB software to investigate different cases of several electrode positions. Finally by adding a feedback loop and models of other dynamic components (electrode’s hydraulic actuators, valves, etc.) a closed-loop dynamic system for controlling the power is developed by them.

Chaos theory and differential equations

Chaos theory is the study of the behavior of dynamical systems that exhibit high sensitivity to initial conditions [33]. Chaos theory has numerous applications in different scientific fields such as fluids mechanics, economics, biology, and computer science. Chaos theory in conjugation with different methods can provide reasonable framework for EAF modeling.

In [34], Hariyanto et al. investigate an analytical approach to characterize voltage and current of EAFs. At first step the dynamic component of EAF is obtained by solving the energy equation of EAF. On the next step using “Chua’s Chaotic Circuit” the chaotic behaviour of the EAF is modeled. Using Simulink/MATLAB software the researchers combine the dynamic and the chaotic components, and obtain a pre-phased model. Finally at the last step the pre-phased model is connected to each phase of the EAF load model, and then the developed model is used to simulate EAF for the refining stage. The results of study show that the generated voltage and current successfully resemble actual EAF especially in harmonic content.

Ozgun and Abur‘s work [35] is predecessor of the previous work [34]. They also use chaos theory for EAF modeling. From historical point of view, we have two general approaches to the problem of arc furnace modeling: stochastic and chaotic. The main advantage of chaotic approach is that models developed by this method will capture changes in the $i-v$ characteristics as the operating conditions change. In contrast, stochastic approaches or some other empirical relations are not normally able to do that. [35] Ozgun et al. start with modeling of EAF’s dynamic behaviour based on Collantes-Bellido and T. Gomez work [36] using energy equations. The second step is similar to Hariyanto et al. work [34] as the low-frequency chaotic signal generated by the simulation of Chua’s circuit is combined with dynamic model obtained from previous step. The main differences of these two studies are that Hariyanto et al. work [34] is extended to three phased EAF while Ozgun and Abur ‘s work is limited to one phase. The simulation results show that the voltage and current resemble actual EAF.

Probability theory

Another well-known approach for dynamic modeling of chaotic and non-linear systems is probabilistic approach. Within chaos and nonlinearity there is hidden order which can be investigated with probabilistic methods. EAF modeling can also be tackled with this view point.

Using probability techniques, Petersen et al. [37] provide two models to study the flickering phenomena of EAFs. In the first model, "Arc-Voltage Model", the fluctuation of the arc voltage is assumed to be Gaussian approximately. The primary source of this fluctuation is the variation of the arc length caused by arc movement and sudden collapse of metal scarps. The inherent response-time limitation of the electrode displacement control system is another contributor. By using mean voltage and variance of square arc voltage the authors propose a model to simulate the arc voltage. In the second model, "Arc-resistance model", the authors try to model the arc resistance by statistical properties as well. Although true distribution of the furnace resistance is more concentrated around mean value than what is expected for normal distribution, an assumption of Gaussian distribution provides an acceptable model. In their conclusions, authors acknowledge that though models provide accurate results, they are only simulated for one case study. For that reason, the models should be investigated on several other furnace installations to establish a confidence level of developed models.

Hidden Markov theory

Hidden Markov Models (HMMs) are another statistical scheme to model stochastic dynamic system. They originally have been developed for speech recognition, and became extremely popular [38].

In their work [39] Esfahani and Vahidi use hidden Markov model (HMM) to approach EAF problem. At first authors analyze the properties of the arc. Based on observed characteristics they argue that current properties of the arc depend on its current and previous states. Hence, by sampling from the voltage and the current of an actual arc, the hidden Markov theory can be adopted. For this purpose the arc $v - i$ characteristic is divided into four regions. Then, different operating points are created in the aforesaid regions. Using the actual measured current and voltage of an electric arc in several working cycles, the statistical probability of the operating point is obtained under hidden Markov theory. According the authors the main advantages of this model are its non-approximation and accuracy in modeling which stems from the fact that the model is developed by experimental samples and applied HMM rather than specific mathematical equations. For that reason the developed model cannot completely be considered as white model.

At the end it should be mentioned that that white box models do not generally offer accurate dynamic models for EAFs. Even for some models the random states of the arc is non-existence. That is primarily due to the fact that most of these models use a specific set of mathematical equations to characterize the behavior of electric arc. Hence, the exhibited behaviour of the models is limited to that specific framework, and consequently the true randomness of the electric arc is not correctly modeled.

3.2.2. *Black box models of EAFs*

Black box modeling is useful when “a priori” knowledge is not available or the underlying process is so complex that it cannot be described into any conventional mathematical model accurately. In these cases the primary goal is fitting available data to a model regardless of its detailed mathematical structure. In general the model would have some free parameters that must be adjusted by several training cycles. This scheme is very much sensible for EAF modelling as EAFs are chaotic and highly nonlinear by nature. Many frameworks can be adopted to model an EAF in this context. Here we will discuss some with one or two examples.

Fuzzy Interface System approach

Fuzzy Interface System (FIS) is a reasoning system which is used for decision making and mathematical modeling. Compared to the classical logic this system accommodates partial true/false values, and for that reason these systems are quite versatile. The main advantage of FIS over other modeling systems is its capability in utilization of qualitative “if-then” rules to express nonlinear input/output relationships.

Sadeghian and Lavers use FIS approach to model an EAF in their work [40]. Within this study two models have been developed. The first model tries to solve the problem with classical fuzzy logic method, and the second model uses adaptive fuzzy logic method. Here we will discuss the first approach and the other will be discussed later. In general for classical fuzzy logic, rule development is a try and error process. The existence of expert knowledge is also very helpful as it narrows the search space; Nevertheless with large number of inputs and chaotic/nonlinear behaviour of EAF the try and error approach, even with expert knowledge, would not be feasible. To solve this problem the authors use a straightforward fuzzy partitioning method.

At first, the input/output space is identified. Then the input and output spaces are divided to arbitrary number of clusters with Gaussian membership functions. At the next step, each training data point is used to associate the input clusters to the output clusters in fuzzy rule format. In general with this method, a large number of data points tend to produce a large number of rules, however a trimming algorithm can be utilized to trim and resolve the redundant/conflicting rules. With the larger training set, this technique does not necessarily provide better result yet the training would be faster since it is done in one pass.

Adaptive Neuro Fuzzy approach

Adaptive neuro-fuzzy networks are hybrid of classical fuzzy rule based systems and neural networks. They encode the reasoning style of fuzzy systems into a neural network structure; hence the models developed by this method can be trained by proper training algorithm (e.g.

BP algorithm) based on training/verification sets. In their papers [1][40] Sadeghian and Lavers use adaptive neuro-fuzzy interface system (ANFIS) to model an EAF. The ANFIS adopts Gaussian membership functions for the input sets, linear functions for the rule outputs, and Sugeno's inference for aggregation of the rule outputs.

At first step, authors use subtractive clustering algorithm to extract fuzzy rules from training data set. The main advantages of clustering algorithms are fuzzy rule reduction and association of good initial value to the model parameters. Then a hybrid algorithm consisting of the gradient descent and the least-squares estimate is deployed to tune up these parameters. Each iteration of this algorithm has a forward pass and a backward pass. In the forward pass, while the antecedent parameters are fixed, using the linear least-square estimate the consequent parameters are optimized. In the backward pass, the consequent parameters are considered to be fixed, and the output error is back propagated through the network. Then by using BP concept, the antecedent parameters will be updated.

The results of Sadeghian and Lavers' study show that ANFIS provides accurate result with fast training. Their ANFIS model [1] offers a better model than their classical fuzzy logic model [40].

Neural network approach

The idea of artificial neural networks (ANNs) is based on simplified biological neural networks. The success of biological neural network (e.g. animal brains) in nature has been an inspiring force for researchers to advance this field throughout decades of research. ANNs are utilized when the problem on hand cannot be easily materialized with conventional mathematical models. As all other statistical/adaptive models ANNs have to go through an training process which can be supervised or unsupervised depending on the type of network and the methodology used for training. With this brief introduction it is quite natural to see ANNs as a good candidate for EAF modeling since EAFs are chaotic and untamed mathematically. Several studies are conducted for EAF modeling with different variations of ANNs.

In their work [41], Chang and Chen propose an ANN model for EAFs. The classical multilayered perceptrons, also known as backpropagation networks (BPNs), are the most popular artificial networks. They are very effective in handling nonlinear problems, and providing results with high prediction. Normally these networks are trained with the backpropagation algorithm; however this algorithm has two undesirable drawbacks namely slow convergence and convergence to local minima of the error surface instead of the global minimum [42]. The radial basis function neural networks (RBFNN) are an alternative to BPNs with simpler structure. As RBFNNs are very suitable for function learning and modeling the authors use them to model EAFs. Generally BPN and RBFNN both assume that there is a stationary relationship between input and output. Consequently they cannot provide good model for the systems which their state are changing chaotically [40] (such as EAFs). To solve this problem authors combine RBFNN with lookup table (LUT) method. In fact LUT provides a framework for the network to memorize the dynamic characteristic of EAF waveform. Simulation shows that their method can accurately model the $i - v$ characteristic for assessment of waveform distortions, flickers, and reactive power consumptions.

Some other techniques can also be combined with ANNs. For instance Mishra et al. [43] use the combination of S-transforms and probabilistic neural network (PNN) to do power quality disturbances characterization and detection. In [44] Santoso et al. propose the combination of Wavelet analysis and ANN for disturbance detection. In general these hybridizations are done by feature extraction. In these models instead of feeding raw input vectors to the neural networks the input vectors are pre-processed by one of these methods, and some feature vectors will be extracted. On the next step, feature vectors alone or combined with original inputs are feed to the networks.

4. PROPOSED METHODOLOGY

4.1. Reasons for Using Kalman Filter

Kalman Filter is a well-known tool for estimation and prediction of dynamic systems since 1960s. There are substantial studies regarding to Kalman Filter and time series analysis [45][47][46][47][48]. Kalman filter provides an online algorithm that constantly updates itself to produce new estimate based on space state models [46] . It can effectively handle the missing data which is quite common in real applications as well [46]. Extended Kalman Filter, as a full second order gradient decent method, has faster convergence rate than the first order gradient decent algorithm since it normally converges with fewer iterations. All of these properties make Kalman Filter to be a good candidate for EAF modeling.

4.2. Data acquisition

In this study, the data that we are using for modeling is collected from a typical EAF. The dataset has two variables, current and voltage of the electrical arc; therefore it is considered as a multivariate time-series. The sampling rate of data acquisition (the Nyquist frequency) is 1920 Hz. No information is available about the transducers or the techniques involved in data collection hence the uncertainty level is unknown. Nevertheless it should be mentioned that as the data is not synthetic, with no doubt we do have some level of uncertainty in our data. The source of uncertainty for these types of measurements are normally bonded to transducers properties (accuracy, response time, etc.) as well as measurement techniques (e.g. direct measurement, indirect measurement, etc.). In this study we treat the dataset as it is.

4.3. Preprocessing of data

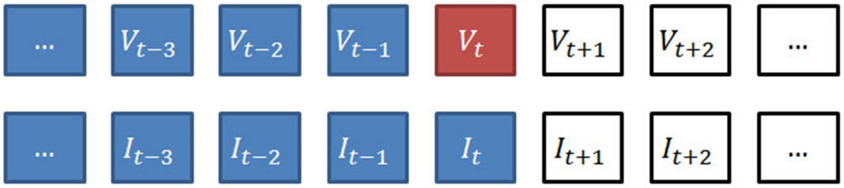
The data set might or might not be preprocessed. Preprocessing depends on the structure of the model and modeling technique. After selection of the model and its structure, the free parameter of the model will be adjusted and updated by Extended Kalman Filter (EKF). Here we will discuss three models. Pre-preprocessing of the data (if needed) will be discussed for each model individually in section 4.5.

4.4. Training/adjustment of the model by Kalman filter

The objective of the model is to predict the present value of the arc voltage based on previous values of the current and the voltage. We also consider that the present value of the current, corresponding to the unknown arc voltage, is known. That means at time t the V_t is unknown, while the known variables are $V_{t-1}, V_{t-2}, \ldots$ and $I_t, I_{t-1}, I_{t-2}, \ldots$. This can be seen in Figure 4-1. In this figure known data are color-coded in blue. The model's job is mapping from known data to the unknown value (V_t). This mapping can be described as a following function $h(.)$,

$$\begin{aligned} &h : \mathcal{R}^n \rightarrow \mathcal{R} \\ &V_t = h(Input\ vector\ ,\ \ Parameter\ vector) \end{aligned} \tag{4.1}$$

The input vector is the information obtained from previous values of the voltage and the current (which also includes the present value of the current as mentioned earlier). The input vector can simply be raw values of the V and I, or depending on the model's structure some extracted features can also be included. Nevertheless all extracted features are also derived from previous values of V and I. The parameter vector is the collection of adjusting parameters of the model. These parameters are updated by Kalman filter in each time step hence the model is trained online.



…	V_{t-3}	V_{t-2}	V_{t-1}	V_t	V_{t+1}	V_{t+2}	…
…	I_{t-3}	I_{t-2}	I_{t-1}	I_t	I_{t+1}	I_{t+2}	…

Figure 4-1: The objective of the model is to predict V_t

Now by representing the parameter vector (state vector) and input vector as X_t and U_t respectively the following is the general form of the model,

$$V_t = h(U_t, X_t) \tag{4.2}$$

In his work [12], Dan Simon used the Extended Kalman Filter algorithm to train a fuzzy system. His objective was to model fuzzy estimator for motor current windings. In this work we use the same concept with some adjustments to accommodate our problem. We will describe the mathematical formulation of our method in the following.

Equation (4.2) provides an estimate for present value of the voltage. By assuming that this equation provides the answer with some error, the desired value (d_t) can be written as the following,

$$d_t = h(U_t, X_t) + v_t \tag{4.3}$$

In above equation the desired value is actually the target value, and the model must converge to this value in each step. Now by assuming the state vector is mapped by $f(.)$ with Gaussian error w, we would have,

$$X_t = f(X_{t-1}) + w_{t-1} \tag{4.4}$$

In fact equation (4.4) is the system equation and (4.3) is the measurement equation in Kalman filter terminology. To summarize, different terms in these two equations are,

$X_t \in \mathcal{R}^n$ is the system state vector ($X_t = [x_n,\ x_{n-1},\ x_{n-2}, \dots ,\ x_1]$),

$f(X_t)$ is the system equation; mapping from present state vector to the next one.

$U_t \in \mathcal{R}^m$ is the input vector,

$w_t \in \mathcal{R}^n$ is the vector conveying the system error,

d_t is the target value of the voltage,

$h(U_t, X_t)$ is model mapping and provides the prediction of the voltage value (V_t),

and finally v_t is the error related to the model.

Since $f(X_t)$ is unknown we simply consider it as identity mapping therefore our equations are reduced to,

$$X_t = X_{t-1} + w_{t-1} \tag{4.5}$$

$$d_t = h(U_t, X_t) + v_t \tag{4.6}$$

Assuming that the initial state of model is $X_{\mathbf{0}}$ and sequences $\{v_n\}$ and $\{w_n\}$ are Gaussian and independent from each other with,

$$E[X_0] = \bar{X}_0 \tag{4.7}$$

$$E[(X_0 - \bar{X}_0)(X_0 - \bar{X}_0)^T] = P_0 \tag{4.8}$$

$$E[w_k] = E[v_k] = 0 \tag{4.9}$$

$$E\left[\begin{pmatrix} w_t \\ v_t \end{pmatrix} \quad (w_t^T \quad v_t^T)\right] = \begin{bmatrix} Q_t & 0 \\ 0 & R_t \end{bmatrix} \tag{4.10}$$

where, the Q_t and R_t are joint covariance matrix of w_t and v_t respectively.

With aforementioned assumptions and assuming that the nonlinearities of (4.6) are sufficiently smooth so that the EKF algorithm can be applied. By referring to the section 2.3.1 we have,

Step 1 (Prediction)

$$\hat{X}_{t|t-1} = \hat{X}_{t-1|t-1} \tag{4.11}$$

$$P_{t|t-1} = FP_{t-1|t-1}F^T + Q_t \tag{4.12}$$

Since we assumed that the $f(.)$ is identify function, its Jacobean matrix is equal to identity matrix,

$$F = \nabla f|_{\hat{X}_{t-1|t-1}} = I \tag{4.13}$$

hence the equation (4.12) is reduced to,

$$P_{t|t-1} = P_{t-1|t-1} + Q_{t-1} \tag{4.14}$$

Step 2 (Filtering)

First we calculate the Kalman Gain,

$$K_t = P_{t|t-1}H_t^T\left[H_tP_{t|t-1}H_t^T + R_t\right]^{-1} \tag{4.15}$$

where H_t is Jacobean matrix of $h(.)$ around $\hat{X}_{t|t-1}$ and denoted by,

$$H_t = \nabla h|_{\hat{X}_{t|t-1}} \tag{4.16}$$

The next step is the calculation of the posterior estimate,

$$X_{t|t} = \hat{X}_{t|t-1} + K_t[d_t - h_t(\hat{X}_{t|t-1})] \tag{4.17}$$

And finally the last step of the Kalman algorithm is the update of covariance matrix for next iteration of the algorithm.

$$P_{t|t} = [I - K_t H_t]P_{t|t-1} \tag{4.18}$$

The results of equations (4.17) and (4.18) give us the estimate and covariance matrix for the next iteration of EKF algorithm. The equation (4.17) provides the updated state vector for the calculation of the output (predicted voltage through equation (4.2)) on the next iteration of the algorithm. Figure 4-2 shows the flow chart of algorithm.

In the beginning, we start with initialization of Q_0, R_0, P_0 and $\hat{X}_0$. Q_0 and R_0 are the covariance matrices of w_0 and v_0 respectively. As mentioned earlier, w_n and v_n are artificial noises that have been added to the process. Their values can be defined as fixed or variable during the process, nevertheless they control convergence speed and accuracy of the training. Selection of proper value for w_n and v_n is subjected to try and error method. $\hat{X}_0$ is also initialized by try and error method as improper values may prevent convergence of the algorithm. Finally since P and Q are closely related, as suggested by equation (4.14), we initialized them to be equal to Q_0 for the first iteration.

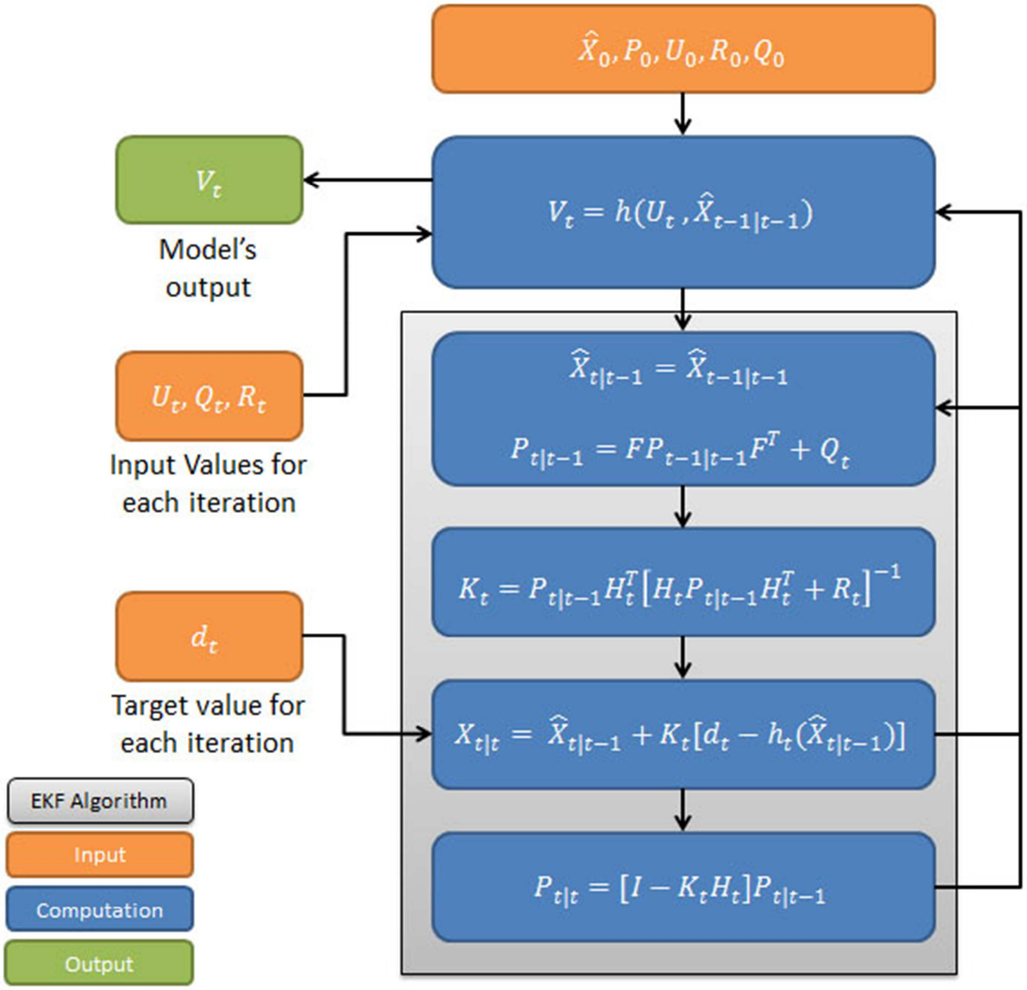

$\hat{X}_0, P_0, U_0, R_0, Q_0$
V_t
Model's output
$V_t = h(U_t, \hat{X}_{t-1|t-1})$
U_t, Q_t, R_t
Input Values for each iteration
$\hat{X}_{t|t-1} = \hat{X}_{t-1|t-1}$
$P_{t|t-1} = FP_{t-1|t-1}F^T + Q_t$
$K_t = P_{t|t-1}H_t^T\left[H_tP_{t|t-1}H_t^T + R_t\right]^{-1}$
d_t
Target value for each iteration
$X_{t|t} = \hat{X}_{t|t-1} + K_t[d_t - h_t(\hat{X}_{t|t-1})]$
$P_{t|t} = [I - K_tH_t]P_{t|t-1}$
EKF Algorithm
Input
Computation
Output

Figure 4-2: Flow chart of algorithm. The model is updated by EKF

Moving down in the flow chart, the next step is the calculation of the model's output by equation (4.2). The rest of the flow chart is simply EKF algorithm in which the new estimate of state vector ($X_{t|t}$) and the new error covariance[6] matrix ($P_{t|t}$) are computed for the next iteration.

4.5. Building the mapping function *h*

As it has been discussed earlier our model is represented by Function h described in equation (4.2). In this section we will discuss three different approaches to define this function.

4.5.1. *Linear Auto-regression model*

The linear model is the simplest model we can consider. In this model we assume that the present value of the arc voltage can be predicted based on linear summation of previous values of the voltage and also present and pervious values of the current,

$$V_t = \sum_{i=1}^{N} \alpha_i V_{t-i} + \sum_{i=1}^{M} \beta_i I_{t+1-i} + \gamma \tag{4.19}$$

where,

α_i, β_i and γ are the parameters of the model, and form the state vector,

$$X_t = [\alpha_1, \dots, \alpha_N, \beta_1, \dots, \beta_M, \gamma]_t \tag{4.20}$$

[6] In reality $P_{t|t}$ does not represent error covariance matrix as this can only be true when $f(.)$ and $h(.)$ are linear.

This vector is learned and adjusted in each time step as it has been discussed in section 4.4.

N is the number of previous values of voltage,

M is the number of present and previous values of current.

For implementation of the algorithm MATLAB platform has been utilized.

4.5.2. *Fuzzy – Kalman model*

In this section we combine our previous linear autoregressive model with fuzzy subtractive clustering algorithm. The subtractive clustering algorithm is a modified version of the Mountain Method [51]. In the beginning, the algorithm assumes each data point is a cluster center, and calculates its likelihood based on the density of the surrounding data points. Then the algorithm selects the data point with the highest potential and removes all data points in its vicinity according to a predetermined radius. This process is iterated until all data points are within the vicinity of at least one cluster.

In this work we use the subtractive clustering algorithm which has been provided by MATLAB. The "*subclust*" function of MATLAB receives a training data set and computes the number of the clusters based on radius of influence [52]. The eventual output of algorithm is a series of Gaussian membership functions for each cluster and each dimension of data point. For instance if the dimension of our data points was three, and the algorithm provided us with two clusters; then we would have two sets of three membership functions. By utilizing a fuzzy inference engine (most commonly Sugeno inference system) the degree of membership for each cluster can be calculated.

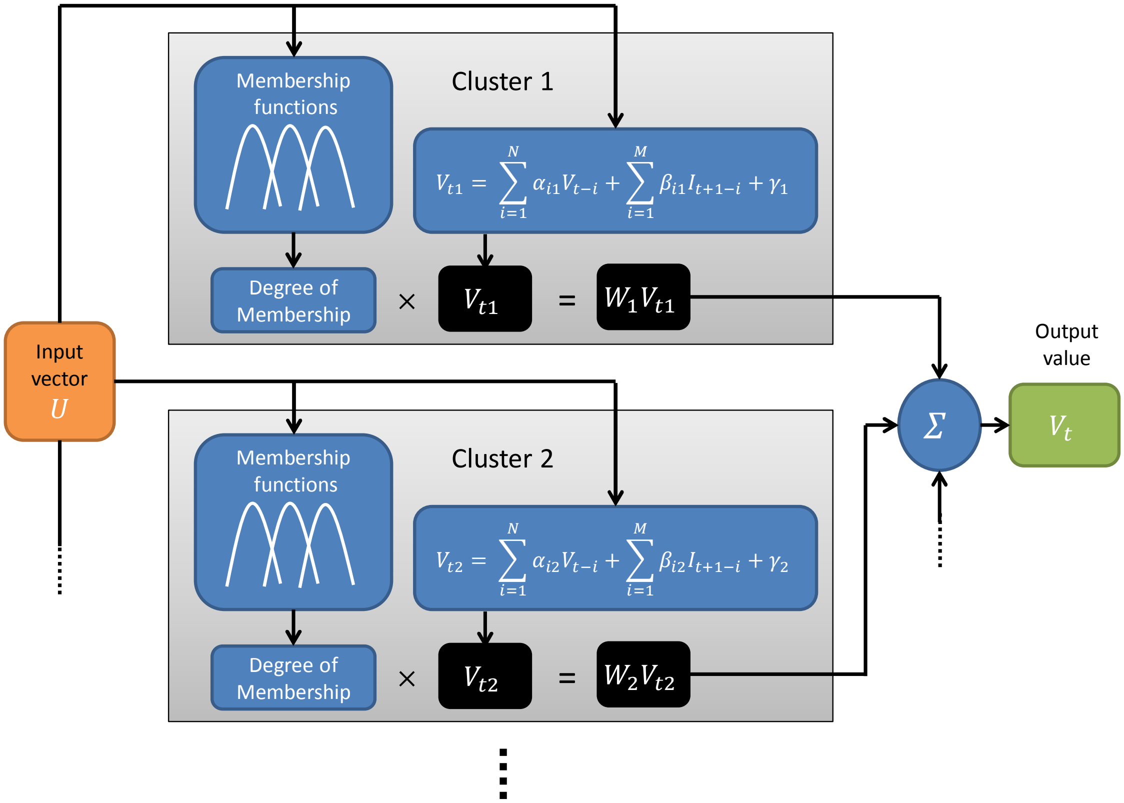

Membership functions
Cluster 1
$V_{t1} = \sum_{i=1}^{N} \alpha_{i1} V_{t-i} + \sum_{i=1}^{M} \beta_{i1} I_{t+1-i} + \gamma_1$
Degree of Membership
×
V_{t1}
=
$W_1 V_{t1}$
Input vector U
Output value
Σ
V_t
Membership functions
Cluster 2
$V_{t2} = \sum_{i=1}^{N} \alpha_{i2} V_{t-i} + \sum_{i=1}^{M} \beta_{i2} I_{t+1-i} + \gamma_2$
Degree of Membership
×
V_{t2}
=
$W_2 V_{t2}$

Figure 4-3: Fuzzy - Kalman model (Function $h(.)$)

For building the fuzzy model, we start with the subtractive clustering algorithm on our training dataset. After obtaining a number of clusters (C), we assume that each cluster has its own linear equation (4.19) to predict the present value of the voltage. The final output of the model is the linear summation of each equation’s output weighted by corresponding membership degree of the cluster. Figure 4-3 shows the process for the fuzzy model. In this model the free parameters are $[a_{11}, \dots , a_{N1}, \beta_{11}, \dots , \beta_{M1}, \gamma_1]_t$, $[a_{12}, \dots , a_{N2}, \beta_{12}, \dots , \beta_{M2}, \gamma_2]_t$, $\dots$, $[a_{1C}, \dots , a_{NC}, \beta_{1C}, \dots , \beta_{MC} , \gamma_C]_t$. Consequently the system state vector is the aggregation all these vectors,

$$X_t = [a_{11}, \dots , a_{N1} , \beta_{11}, \dots , \beta_{M1} , \gamma_1, \dots , a_{1C}, \dots , a_{NC} , \beta_{1C}, \dots , \beta_{MC} , \gamma_C]_t \qquad (4.21)$$

Here the Kalman Filter will adjust and update this vector at each time step.

4.5.3. *Wavelet - Kalman model*

Discrete Wavelet Analysis (DWA) is a branch of signal analysis which is commonly known as Multi Resolution Analysis (MRA). Methodologically wavelet analysis is similar to Fourier analysis as both methods break down a signal into its components for further analysis. Fourier analysis does this job by decomposing a signal into series of sine waves with different frequencies while the wavelet analysis does it by decomposing the signal into its wavelets through scaling and shifting of the *“mother wavelet”*. However there are some fundamental differences between Fourier and wavelet analysis. As it can be seen in Figure 4-4 sine waves, in Fourier analysis, are smooth and of infinite length while, on the other hand, wavelets are temporally localized and irregular in shape. These properties make wavelet to be an efficient tool for analysis of non-stationary signals [53].

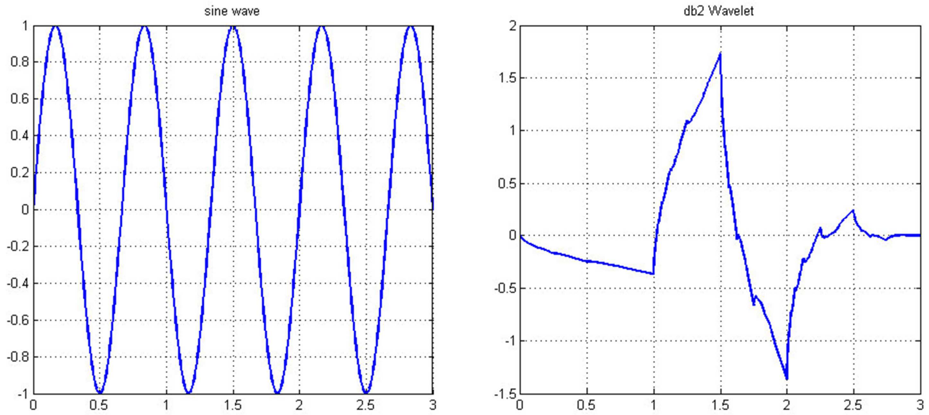


Figure 4-4: Sine wave (left), Daubechies db2 (right)

The DWT of a signal is calculated by passing it through a tree of low and high pass filters. The high pass filter provides us with the detail coefficients while the low pass filter gives us the approximation coefficients of the signal [54]. Since the signal loses half of its frequency by the filtering process, according to the Nyquist theorem, half of the samples are discarded. The algorithm is perpetually repeated on approximation coefficients, and consequently new details of lower resolution are obtained. Figure 4-5 depicts the output of multiresolution analysis up to level three.

In our third model the wavelet analysis is used as preprocessing tool for our raw data. For this purpose the previous values of the voltage and current are individually subjected to DWT algorithm to calculate approximation and detail coefficients to a desired level. Then the result, accompanying with some or all of the raw data (voltage and current without any preprocessing) forms our input vector.

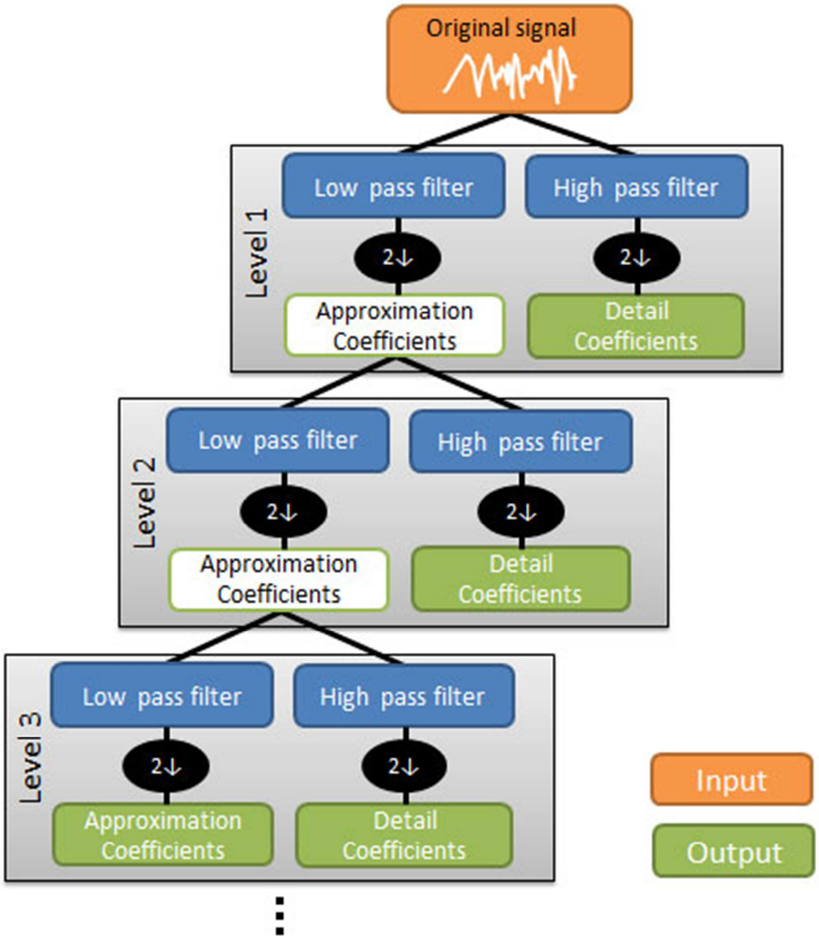


Figure 4-5: Filter bank representation of the DWT

The input vector is defined as following,

$$U_t = [\tilde{A}_L^I, \widetilde{D}_{1,\dots,L}^I, \tilde{A}_L^V, \widetilde{D}_{1,\dots,L}^V, V_{t-1,\dots,t-\,n}, I_{t,\dots,t-m}] \qquad (4.22)$$

where,

$\tilde{A}_L^I$: Approximation coefficients of vector $I_{t,\dots,M}$ at level L.

$\widetilde{D}_{1,\dots,L}^I$: Detail coefficients of vector $I_{t,\dots,M}$ from level 1 to level L.

$\tilde{A}_L^V$: Approximation coefficients of vector $V_{t-1,\dots,N}$ at level L.

$\widetilde{D}_{1,\dots,L}^V$: Detail coefficients of vector $V_{t-1,\dots,N}$ from level 1 to level L.

$V_{t-1,\dots,t-n}$: Desired number of samples from the original voltage input vector ($V_{t-1,\dots,t-N}$).

$I_{t,\dots,t-m}$: Desired number of samples from the original current input vector ($I_{t,\dots,t-M}$).

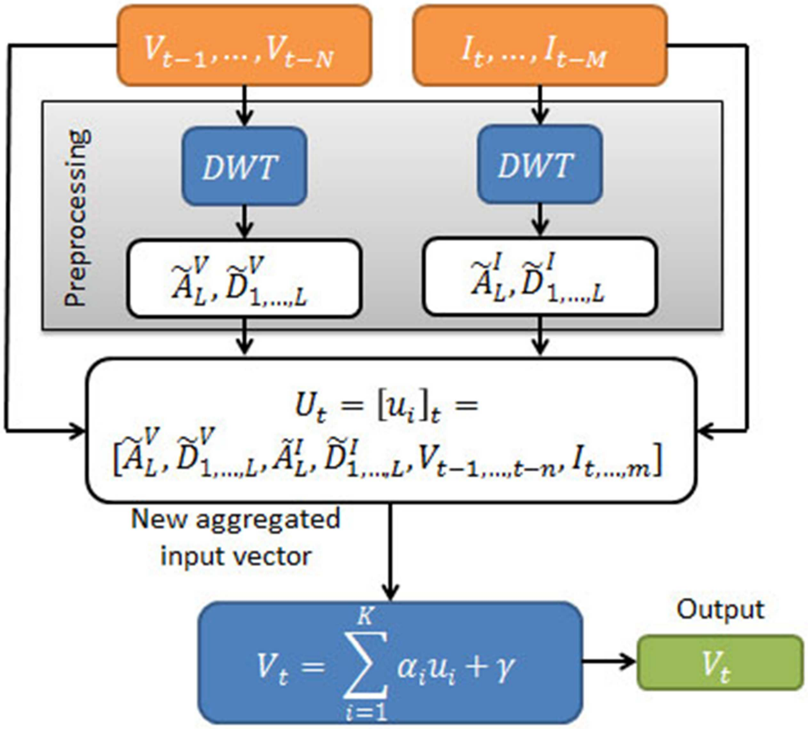

$V_{t-1},\dots,V_{t-N}$
$I_t,\dots,I_{t-M}$
Preprocessing
DWT
DWT
$\tilde{A}_L^V, \tilde{D}_{1,\dots,L}^V$
$\tilde{A}_L^I, \tilde{D}_{1,\dots,L}^I$
$U_t = [u_i]_t =$
$[\tilde{A}_L^V, \tilde{D}_{1,\dots,L}^V, \tilde{A}_L^I, \tilde{D}_{1,\dots,L}^I, V_{t-1,\dots,t-n}, I_{t,\dots,m}]$
New aggregated input vector
$V_t = \sum_{i=1}^{K} \alpha_i u_i + \gamma$
Output
V_t

Figure 4-6: Wavelet model (Function $h(.)$)

Now at the next step our input vector is used in a linear regression function,

$$V_t = \sum_{i=1}^{K} \alpha_i u_i + \gamma \tag{4.23}$$

where K is the number of elements in input vector U_t (See Figure 4-6). The α_i and γ are the free parameters of the model which form the system state vector,

$$X_t = [\alpha_i, \dots, \alpha_K, \gamma]_t \tag{4.24}$$

This vector is updated and adjusted by Kalman Filter through each time step.

4.6. Computational complexity of the proposed algorithm

In general Kalman Filter is computationally expensive. Since the core of the proposed algorithm is based on Extended Kalman Filter, our learning algorithm is also computationally expensive. A quick review of the algorithm (depicted in Figure 4-2) indicates that the main burden of computation lies on the step where the Kalman Gain is calculated,

$$K_t = P_{t|t-1} H_t^T \left[H_t P_{t|t-1} H_t^T + R_t \right]^{-1} \tag{4.25}$$

Inversing matrix $H_t P_{t|t-1} H_t^T + R_t$ by Gaussian Elimination method leads to $O(n^3)$ complexity. In general complexity of matrix inversion is between $O(n^3)$ to $O(n^{2.37})$ depending on property of the matrix and the implemented algorithm [57]. Generally algorithms with equal or larger than quadratic time, $O(n^2)$, do not scale well. Therefore we should be careful not to develop a mapping function with too many free parameters for adjustment, for the proposed algorithm would be too slow.

5. DISCUSSIONS OF RESULTS

5.1. Result of Linear Auto-regression model

Our first model is the Linear Auto-regression model. As it has been described in section 4.5.1 the free parameters of the following equation (α_i, β_i, γ) are adjusted by Kalman Filter,

$$V_t = \sum_{i=1}^{N} \alpha_i V_{t-i} + \sum_{i=1}^{M} \beta_i I_{t+1-i} + \gamma \tag{5.1}$$

N and M represent the number of previous values of the current and voltage used in the mapping function. We consider them to be equal in our model therefore when we state that N is two, it means that two previous value of the voltage and one previous value of the current (in addition to its present value) is fed to the algorithm in each time step. It should also be noted that the present value of the current is known to our model hence we only need to go one step back for the current.

On the next step Q_0, R_0, P_0 and $\hat{X}_0$ must be initialized for the algorithm. Q_0 and R_0 are the covariance matrices of w_0 and v_0 respectively, and calculated as following,

$$Q_0 = \sigma(\{w\})^2 I_{1\times1} \tag{5.2}$$

$$R_0 = \sigma(\{v\})^2 I_{2\times N+1} \tag{5.3}$$

The selection of proper value for standard deviation of $\{w\}$ and $\{v\}$ is subjected to try and error. After some tries, we decided to set them to be equal to

$$\sigma(\{w\}) = \sigma(\{v\}) = 0.1 \tag{5.4}$$

The $\hat{X}_0$ is also subjected to try and error and we set it to be

$$\hat{X}_0 = 0.001 \times I_{2\times N+1} \tag{5.5}$$

where I is the identity matrix, and $2 \times N + 1$ represents the number of free parameters in the model.

After initialization, the model is fed with 600 samples of the time series repeatedly for 150 times. This helps the free parameters to settle to their stable values therefore this phase can be seen as training/adjustment phase. The selection of number of samples (600) and number of epochs (150) is done by the observation of the error fluctuation and the level of generalization of the model. In this study we check two error indexes, None Dimensional Error Index (NDEI) and Root Mean Square Error (RMSE). The first is normalized dimensionless error index and the second is the standard deviation of the differences between predicted values and observed values. The NDEI and RMSE are calculated by following formulas respectively

$$NDEI = \frac{RSME}{\sqrt{\frac{1}{T} \times \sum_{t=1}^{T}\big(d_t - E(d_t)\big)^2}} \tag{5.6}$$

$$RMSE = \sqrt{\frac{\sum_{t=1}^{T}(V_t - d_t)^2}{T}} \tag{5.7}$$

where d_t is the desired (target) value of the voltage and $E(d_t)$ represents the average of the desired values. V_t represents model's output, and finally T represents the number of samples.

Table 5-1: Result of first experiment - Linear Auto-regression model

	Training/Adjustment		Test with EKF ON		Test with EKF Off	
N	**NDEI**	**RMSE (V)**	**NDEI**	**RMSE (V)**	**NDEI**	**RMSE (V)**
1	0.1616	39.7911	0.2060	50.6568	0.6988	94.4757
2	0.1310	32.2646	0.1356	33.4040	0.3350	82.5498
4	0.0597	14.7226	0.0661	16.2804	0.0597	14.7130
8	0.0494	12.2315	0.0530	13.0603	0.0692	17.0308
16	0.0412	10.1656	0.0457	11.2648	0.0825	20.3180
32	0.0321	7.9226	0.0351	8.6477	0.0871	21.4609
64	0.0320	7.8854	0.0364	8.9768	0.1376	33.8931

Note: $\boldsymbol{N}$ represents the number of previous values of the current and voltage used in the mapping function $\boldsymbol{h}(.)$, hence the size of input vector is $\mathbf{2} \times \boldsymbol{N}$. The number of adjusting free parameter is $\mathbf{2} \times \boldsymbol{N} + \mathbf{1}$.

On the next phase, the adjusted model is presented with a new set of data with size of 4400 samples to see how it performs. Consequently this phase can be called the test phase of our experiment. We do this in two ways. First we discount EKF part of the algorithm meaning that the free parameters of the model do not update on each time step. Contrarily in second try we make sure that the EKF part of the algorithm is still engaged and the free parameters are constantly adjusted in each time step. We also repeat the process for different size of input vector (N: $1, 2, 4, 8, 16\ , 32\ and\ 64$) to study the effect of number of previous values of the current and voltage on the prediction. The result of study can be seen in Table 5-1 and

Figure 5-1. Figure 5-2, Figure 5-3 and Figure 5-4 shows outputs of model against target values. The model's output is color coded in blue, while the target value is presented with red colored line. The None Dimensional Error Index (NDEI) is presented in these figures with black colored line. As it can be seen, for windowing size of $N = 32$ (Figure 5-2) the red line can hardly be seen which means the model has good prediction. The situation is not as good as Figure 5-2 for Figure 5-3 and Figure 5-4, as they have higher level of error, $NDEI = 0.053$ and $NDEI = 0.135$ respectively. The reason for this phenomenon is the windowing size for Figure 5-3 and Figure 5-4 are $N = 8$ and $N = 2$ respectively which is smaller than $N = 32$ of Figure 5-2. In general the output Figure 5-1 shows that increase in windowing size will lead to lower level of error, however this effect is not that much, if we go higher than $N = 32$.

By reviewing Table 5-1 and Figure 5-1, it is clear that the output of model is much better when the EKF is engaged. That means that the developed model cannot be used in "train and use" scheme. In fact the EKF part of algorithm must always be engaged to perpetually adjust the parameters.

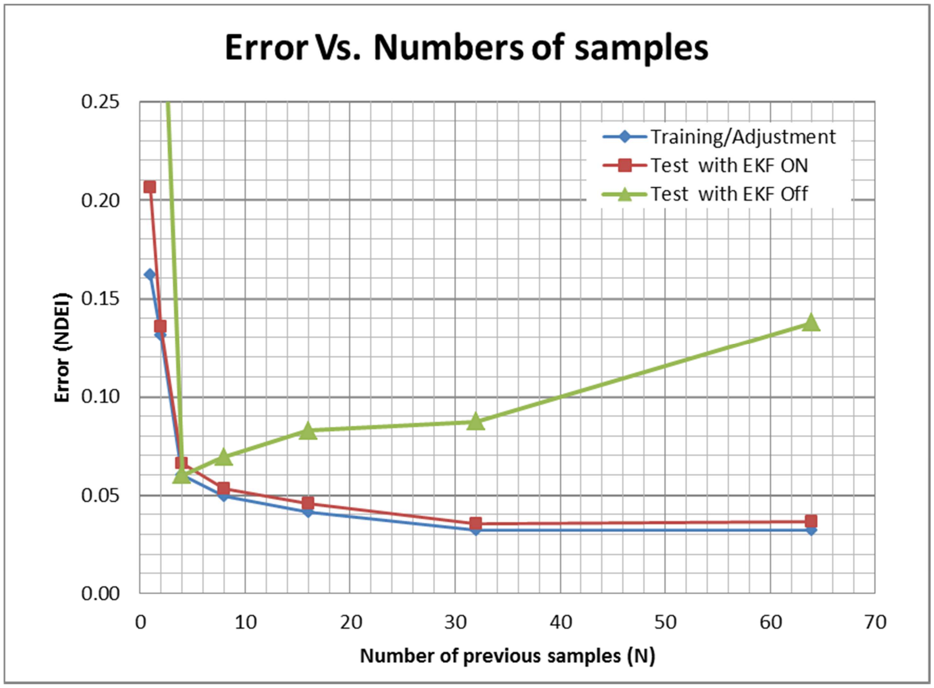


Figure 5-1: Error Vs. numbers of previous values used in modeling - Linear Auto-regression model

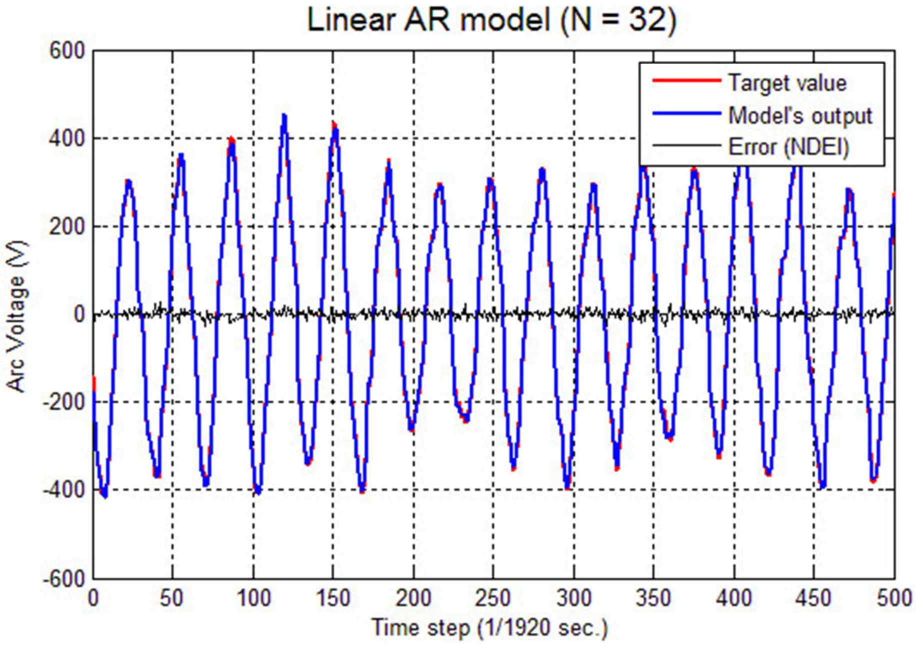


Figure 5-2: Linear auto regression model – N: 32 – NDEI: 0.035

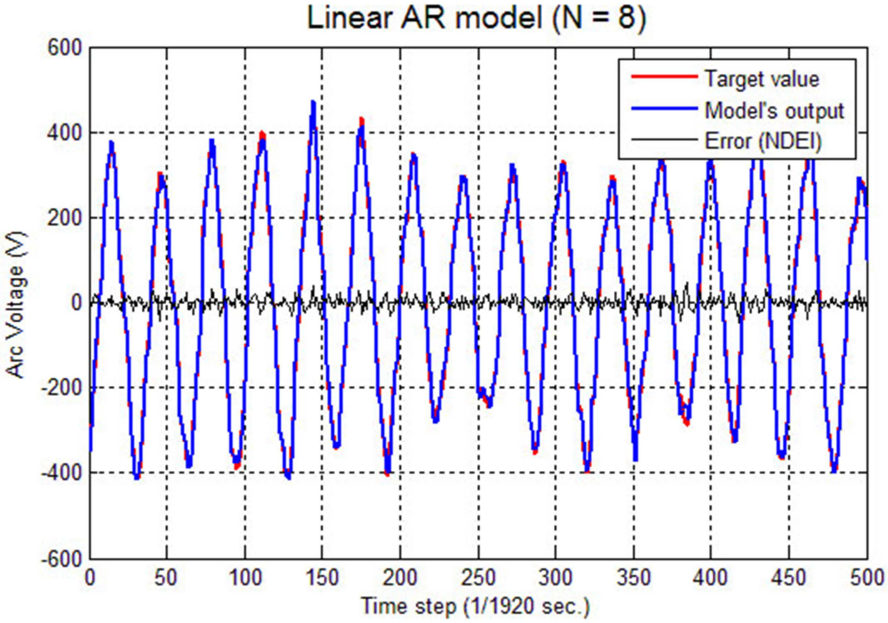


Figure 5-3: Linear auto regression model – N: 8 – NDEI: 0.053

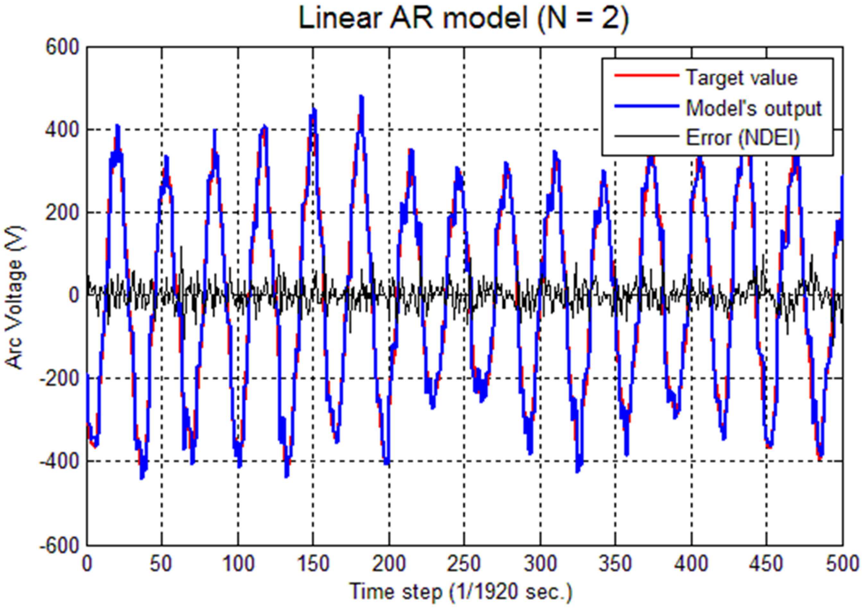


Figure 5-4: Linear auto regression model – N: 2 – NDEI: 0.135

5.2. Result of Fuzzy - Kalman model

As it has been discussed in section 4.5.2, our second model is developed by combination of the Fuzzy model and Kalman filtering. After series of try and error, it has been decided to perform fuzzy clustering on the latest value of the voltage and current available in each time step. In has been observed that using more training values of current and voltage does not provide any meaningful differences in the results.

We start with dividing our dataset into two parts. As the first model, 600 samples of the time series are selected for training/adjusted phase and the rest (4400 samples) will be used in the testing phase. At first the samples are fed to subtractive fuzzy clustering algorithm to obtain clusters and their membership functions. The result of fuzzy clustering algorithm is four clusters. Consequently for each attribute of the clusters (latest value of current and voltage) we have four Gaussian membership functions. Using Larsen Product Implication, membership degree of each cluster at each time step is calculated from following equation,

$$\phi_t^{ClusterN}\left(\mu_V^{ClusterN}, \mu_C^{ClusterN}\right) = \mu_V^{ClusterN} \times \mu_C^{ClusterN} \qquad (5.8)$$

where,

$\phi_t^{ClusterN}$ is the degree of membership for $ClusterN$ (N: 1, 2, 3 or 4).

$\mu_V^{ClusterN}$ is Gaussian membership function for latest available value of voltage. Meaning that for time step t the V_{t-1} is used for calculation of the membership function’s value while the spread and center of membership function are calculated by fuzzy clustering algorithm.

$\mu_C^{ClusterN}$ is Gaussian membership function for latest available value of current. Meaning that for time step t the I_t is used for calculation of the membership function’s value. As of $\mu_V^{ClusterN}$, the spread and center of the membership function is also provided by fuzzy clustering algorithm.

As depicted in Figure 4-3 each one of the four clusters has its own linear regression functions. Nevertheless the final output is calculated by the summation of weighted output of each function based on the membership degree of corresponding cluster. During the training/adjustment phase the free parameters of these four equations are adjusted by EKF algorithm. Now if we consider that we use the N latest values of voltage and current, then we have to adjust $4 \times (2 \times N + 1)$ or $8 \times N + 4$ free parameters for the whole model.

The initialization of Q_0, R_0, P_0 and $\hat{X}_0$ is done similar to the first model. Now we are all set to start training/adjustment phase. 600 samples of the dataset are fed to the model for 150 times. This helps the free parameters of the model settle to their stable values. Then the developed model will be exposed to the rest of dataset (4400 samples) to check its performance. This is done in two times. First we disengage the EKF part of algorithm, and in the second try we let the EKF part of algorithm to stay engaged during the test phase. This will help us to study the role of EKF on testing phase as well. The performance of the model is monitored with two error indexes which has been introduce in section 5.1. The results of experiment are given in Table 5-2 and Figure 5-5 to Figure 5-9.

Studying Figure 5-6 to Figure 5-9 reveals the same finding similar to our previous model (linear auto-regression model). Increase in windowing size will give us better error rate, however the improvement quickly becomes insignificant once N, the windowing size, increases. This trend can be seen in Figure 5-5 clearly.

A quick review on Table 5-2 and Figure 5-5 indicates that similar to our first experiment (linear auto-regression model), the accuracy of the model heavily depends on the engagement of the EKF algorithm during the test phase. A cross examination of Table 5-2 and Table 5-1 reveals that the error rate of Fuzzy-Kalman models are slightly more than linear auto-regression models.

Table 5-2: Result of second experiment, Fuzzy - Kalman model

	Training/Adjustment		Test with EKF ON		Test with EKF Off	
N	**NDEI**	**RMSE (V)**	**NDEI**	**RMSE (V)**	**NDEI**	**RMSE (V)**
1	0.1412	34.7833	0.1927	47.4730	0.2936	72.3382
2	0.1282	31.5884	0.1345	33.1531	0.3181	78.3888
4	0.0578	14.2530	0.0635	15.6544	0.0769	18.9357
8	0.0474	11.7272	0.0511	12.5961	0.0836	20.5798
16	0.0406	10.0086	0.0457	11.2645	0.0986	24.2955
32	0.0358	8.8446	0.0393	9.6831	0.1181	29.0893

Note: $\boldsymbol{N}$ represents the number of previous values of the current and voltage used in the mapping function $\boldsymbol{h}(.)$ hence the size of input vector is $\mathbf{2} \times \boldsymbol{N}$. As we have four clusters in our Fuzzy – Kalman model the size of the free parameter vector is $\mathbf{8} \times \boldsymbol{N} + \mathbf{4}$.

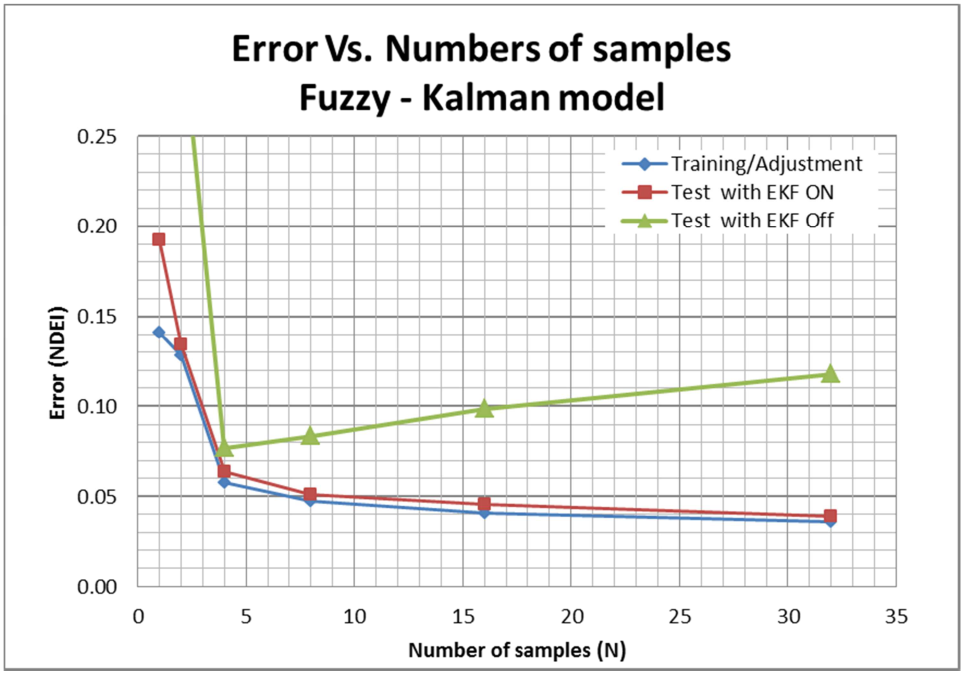


Figure 5-5: Error Vs. numbers of previous values used in modeling, Fuzzy - Kalman model

The main rationale behind incorporating fuzzy logic with Kalman filtering was to provide a decision making capability for the model. However to our surprise, it did not improve model's error rate. In fact, it caused slight increase in the error. In addition, Fuzzy-Kalman model has more free parameters comparing to our first model ($8N + 4$ to $2N + 1$, since fuzzy clustering divides dataset to four clusters). That must theoretically help Fuzzy-Kalman model to have better prediction which did not happen here. In fact, having more free parameters means more computation, and consequently this leads to lower efficiency of the model from computational point of view.

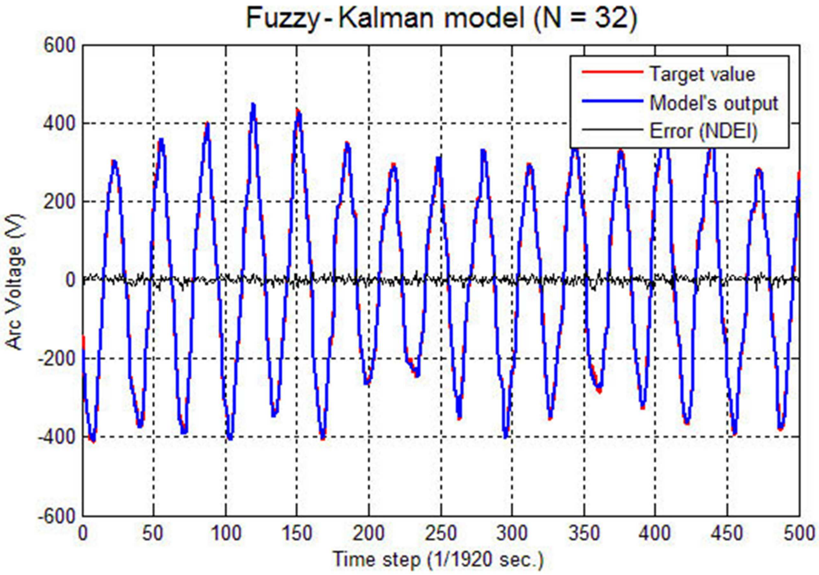


Figure 5-6: Fuzzy – Kalman model, N: 32, NDEI: 0.0393

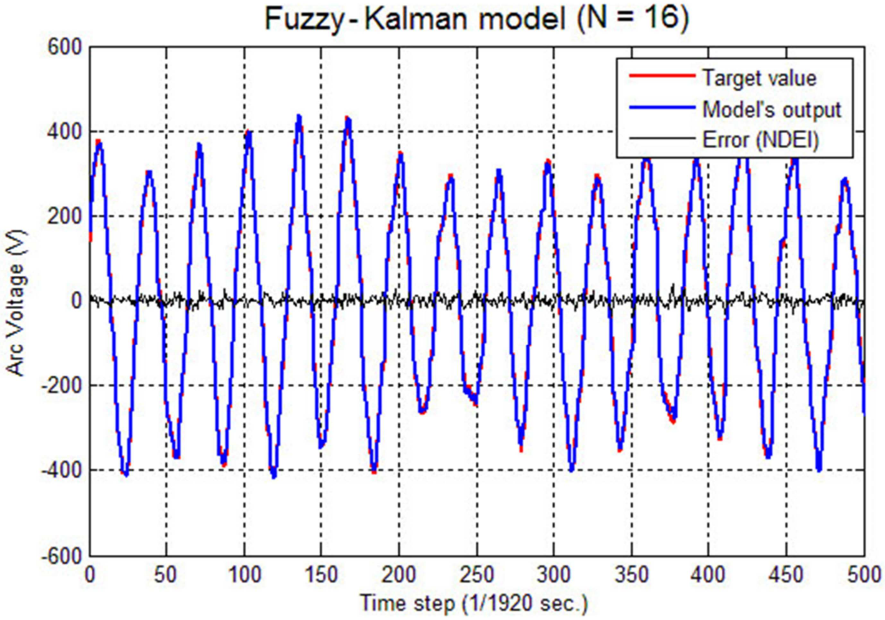


Figure 5-7: Fuzzy – Kalman model, N: 16, NDEI: 0.0457

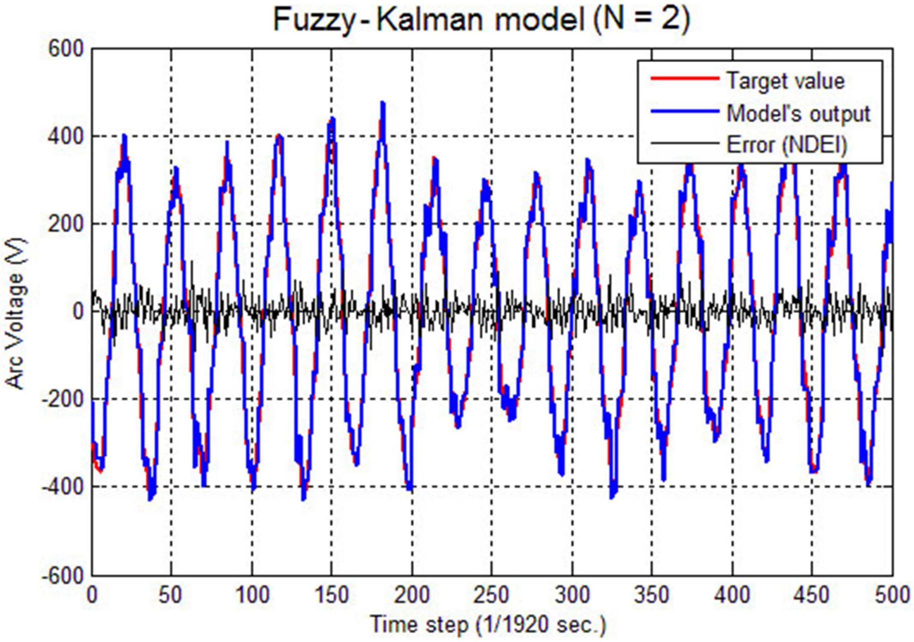


Figure 5-8: Fuzzy – Kalman model, N: 2, NDEI: 0.1345

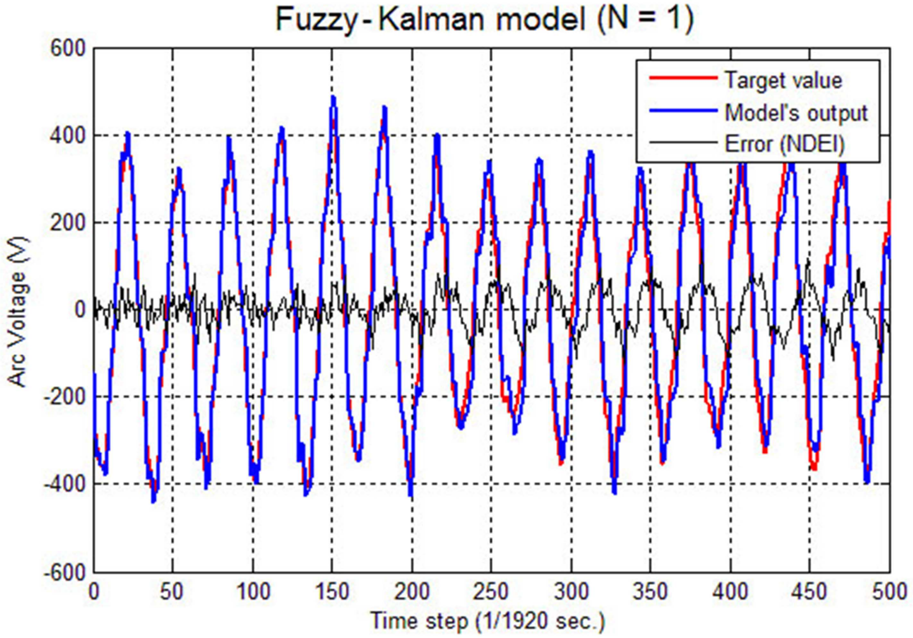


Figure 5-9: Fuzzy – Kalman model, N: 1, NDEI: 0.1927

5.3. Result of Wavelet–Kalman model

In the third experiment we use DWT for preprocessing of the data. The initialization of the Q_0, R_0, P_0 and $\hat{X}_0$ is done in a similar manner to our previous models. Again we spilt the dataset into two parts. The first part has 600 samples, and will be used for training/adjustment phase. The rest of dataset (4400 samples) will be used for the test phase. Preprocessing of the data is quite straight forward (see Figure 4-6). Voltage and current vectors are individually decomposed by DWT algorithm to desired level. The latest values of the voltage (V_t) and current (I_{t-1}) alongside of the two decomposed vectors are concatenated and fed to the EKF algorithm to adjust model's free parameters. After 150 times of iteration on the training/adjustment dataset, the value of free parameters settles to their stable values and the model is ready for performance test.

The performance test is also done in a similar fashion to our previous models. We first disengage the EKF algorithm to see how model performs without online adjustment of Kalman algorithm. Then in the second run the EKF is engaged and continuously adjusts model's free parameters online.

The process of training/adjustment and performance test has been done several times with decomposition levels of 1, 2 and 3. Since the minimum number of samples required for level 3 decomposition is equal to 8, the smallest windowing size of 8 is selected for these models. In general, minimum number of samples for decomposition level of n is equal to 2^n.

The results of our third experiment can be seen in Table 5-3, Table 5-4 and Table 5-5. Table 5-3 provides us with error rates of decomposition level of 1 for different windowing size (Starting from $N = 8$ up to $N = 64$), while Table 5-4 and Table 5-5 provide the same information about decomposition level of 2 and 3 respectively. By reviewing these three tables, one can infer that there is no significant improvement in error rate when we increase the decomposition level. For instance, by comparing the None Dimensional Error Index (NDEI) for $N = 64$, we virtually observe the same error rate for EKF engaged test phase. The error rate of decomposition level 1 is 0.0335 while level 2 and 3 provide 0.0334. Moreover, a quick glance on Figure 5-13, Figure 5-13 and Figure 5-15 show us that the output of models, with windowing size of 32, for all decomposition level is identical.

Table 5-3: Result of third experiment – Wavelet-Kalman - decomposition level : 1

	Wavelet-Kalman - decomposition level : 1 - Haar wavelet					
	Training/Adjustment		**Test with EKF ON**		**Test with EKF Off**	
N	**NDEI**	**RMSE (V)**	**NDEI**	**RMSE (V)**	**NDEI**	**RMSE (V)**
8	0.0492	12.1645	0.0528	13.0070	0.0661	16.2792
16	0.0414	10.2061	0.0460	11.3439	0.0843	20.7725
32	0.0316	7.7917	0.0349	8.5871	0.0915	22.5429
64	0.0283	6.9779	0.0335	8.2424	0.0651	16.0371
Note: $\boldsymbol{N}$ represents the number of previous values of the current and voltage used in the mapping function $\boldsymbol{h}(.)$. Since the latest values of voltage and current is also concatenated to the decomposed vectors the number of model's free parameter is $\mathbf{2} \times \boldsymbol{N} + \mathbf{3}$.						

Table 5-4: Result of third experiment – Wavelet-Kalman - decomposition level : 2

	Wavelet-Kalman - decomposition level : 2 - Haar wavelet					
	Training/Adjustment		Test with EKF ON		Test with EKF Off	
N	NDEI	RMSE (V)	NDEI	RMSE (V)	NDEI	RMSE (V)
8	0.0492	12.1647	0.0528	13.0073	0.0661	16.2803
16	0.0414	10.2068	0.0460	11.3450	0.0843	20.7749
32	0.0316	7.7908	0.0349	8.5869	0.0915	22.5433
64	0.0283	6.9763	0.0334	8.2402	0.0651	16.0282
Note: N represents the number of previous values of the current and voltage used in the mapping function $h(.)$. Since the latest values of voltage and current is also concatenated to the decomposed vectors the number of model's free parameter is $2 \times N + 3$.						

Table 5-5: Result of third experiment – Wavelet-Kalman - decomposition level : 3

	Wavelet-Kalman - decomposition level : 3 - Haar wavelet					
	Training/Adjustment		Test with EKF ON		Test with EKF Off	
N	NDEI	RMSE (V)	NDEI	RMSE (V)	NDEI	RMSE (V)
8	0.0492	12.1645	0.0528	13.0071	0.0661	16.2797
16	0.0414	10.2077	0.0461	11.3461	0.0843	20.7763
32	0.0316	7.7914	0.0349	8.5871	0.0915	22.5478
64	0.0283	6.9742	0.0334	8.2382	0.0651	16.0258
Note: N represents the number of previous values of the current and voltage used in the mapping function $h(.)$. Since the latest values of voltage and current is also concatenated to the decomposed vectors the number of model's free parameter is $2 \times N + 3$.						

Studying Figure 5-10, Figure 5-11 and Figure 5-12 show that similar to our two previous experiments (linear auto-regression and Fuzzy-Kalman models), the Wavelet-Kalman models need to have EKF algorithm engaged during the test phase. The comparison between these three experimental models (linear auto-regression, Fuzzy-Kalman and Wavelet-Kalman) reveals that the Wavelet models have slightly better outcome than linear regressive models. That is not very surprizing since the decomposed vectors of current and voltage contains coefficients reflecting both low and high frequencies. Low frequency coefficients (approximations coef.) present information about the global trend of the signal, while high frequency coefficients of different decomposition levels (detail coef.) provide more transient oriented information. We believe that is the main reason for better performance of Wavelet-Kalman models.

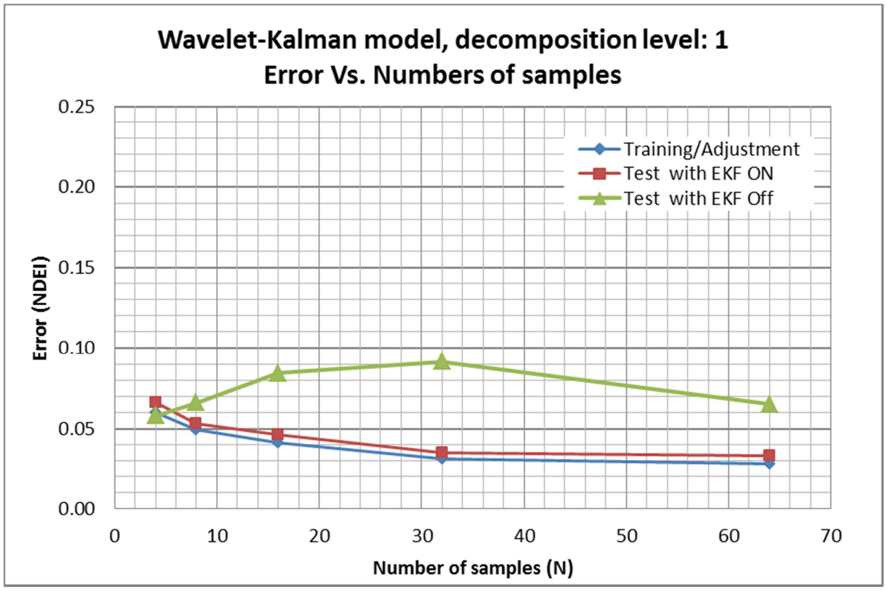


Figure 5-10: Error vs. number of previous samples used in modeling – Wavelet-Kalman model – decomposition level: 1

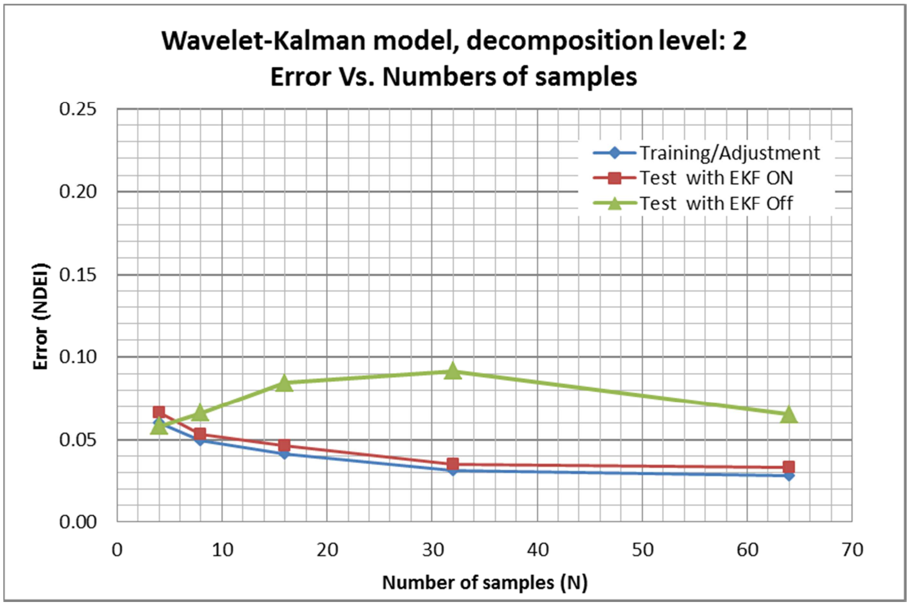


Figure 5-11: Error vs. number of previous samples used in modeling – Wavelet-Kalman model – decomposition level: 2

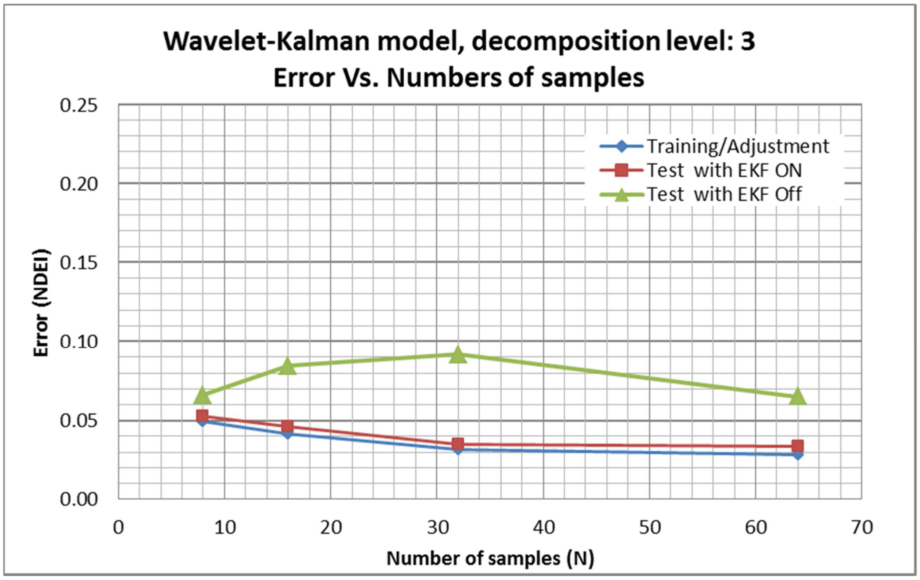


Figure 5-12: Error vs. number of previous samples used in modeling – Wavelet-Kalman model – decomposition level: 3

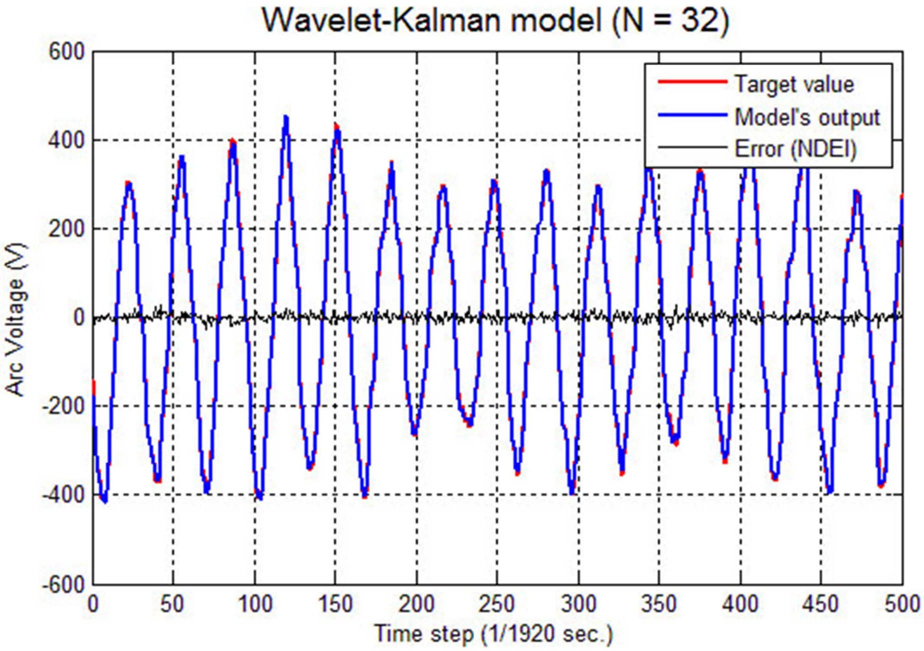


Figure 5-13: Wavelet–Kalman model, N: 32, Dec. level:1, NDEI: 0.0349

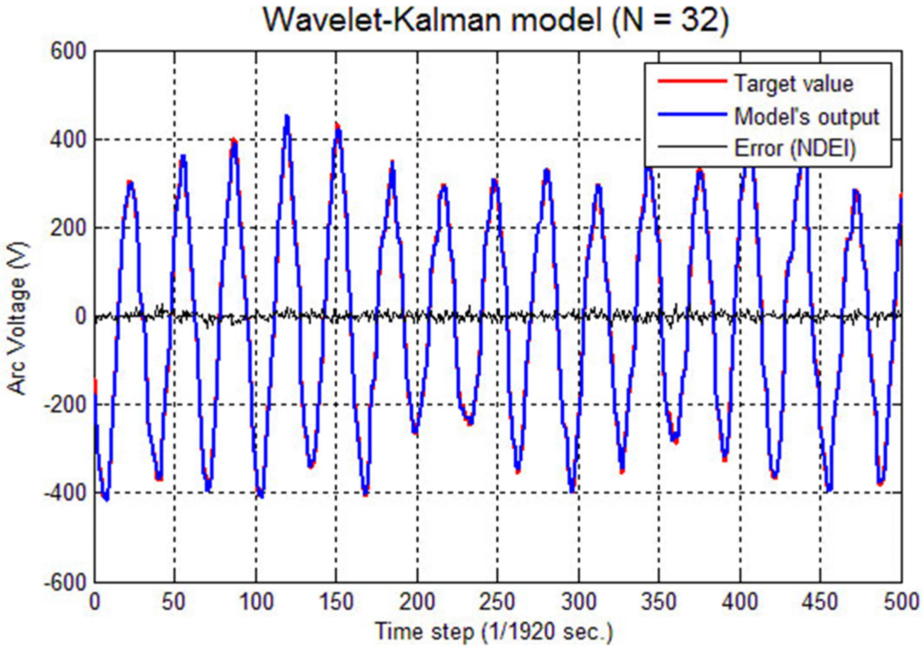


Figure 5-14: Wavelet–Kalman model, N: 32, Dec. level:2, NDEI: 0.0349

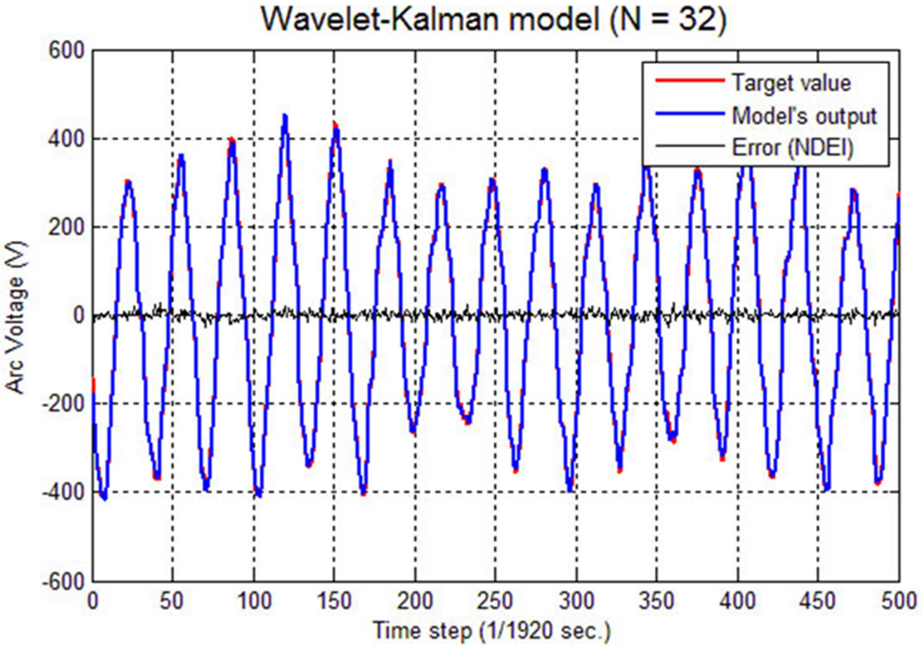


Figure 5-15: Wavelet–Kalman model, N: 32, Dec. level:3, NDEI: 0.0349

6. CONCLUSIONS AND FUTURE WORKS

6.1. Summary and Conclusions

In this work, we tried to approach the arc furnace prediction problem using Kalman Filtering. Other attempts have also been made to generate a good model for this purpose, and some provided very good results (e.g. [40]). However, all of these attempts might be compromised by one fundamental problem. When we are using a training set to train a model, we limit the model to the conditions which exist within that set. Normally real world processes are very complicated. Hence, obtaining a training set that could effectively encompass all possible working conditions is a hard or even an impossible task. In the case of electrical furnace, there are numerous conditions that could change from time to time. For instance, the graphite electrodes are constantly eroded during operation of the furnace or the consistency of the metal scraps differs from one charge to another. All of these varying conditions have impacts on the transfer function that we want to model. Consequently, when a model is trained by a training set, it would only respond best to conditions resemble to that set, and its response to other working conditions would be unknown. Using an online self-adjusting model can eliminate this problem, and that is what we tried in this work.

In this study we used Kalman Filter algorithm to develop a number of online self-adjusting forecasting models. In the conducted experiments, we developed three models based on Kalman Filtering. The first model was a linear auto-regressive model based of EKF algorithm. The best results by this model was achieved, when the windowing size of $N = 32$ was selected. The error rate of this model was 0.0351 with aforementioned windowing size. In second experiment we tried to incorporate Fuzzy logic and Kalman Filter to improve the error rate. However, to our surprise there was no improvement in the output of Fuzzy-Kalman model. The best error rate was achieved with windowing size of $N = 32$, and it was 0.0393. This shows that the error performance of this model is less than linear auto-regression model. Finally, in the third attempt, we tried to combine the Wavelet analysis with Kalman Filtering. This approach proved to be relatively successful, and the developed model gave us relatively better results. A Wavelet-Kalman model with decomposition level 1 and the windowing size of $N = 64$, provided the best error performance. In fact, it gave us 0.0334 as the error rate. The cross examination these three models are given in Table 6-1.

Table 6-1: Comparing three developed models

		Error rate (NDEI)	Error rate (RMS)	Improvement to Model 1
Model 1 **N = 32**	**Linear AR-Kalman**	0.0351	8.6477 V	N/A
Model 2 **N = 32**	**Fuzzy-Kalman**	0.0393	9.6831 V	-11.97 %
Model 3 **N = 64**	**Wavelet-Kalman**	0.0334	8.2382 V	4.74 %

Developed models can also be analyzed from speed performance point of view as well. Linear AR-Kalman model (linear auto regression model) does not need any pre-processing, hence it provides the fastest performance among our models. On the other hand, Fuzzy-Kalman model has more free parameters for adjustment. The number of free parameters for Fuzzy-Kalman model is proportional to the number of clusters used in modeling. For instance, if we use four clusters, then the number of free parameter is four times of the Linear AR-Kalman with same windowing size. Finally, Wavelet-Kalman model is not computationally expensive as Fuzzy-Kalman models. That is true because Harr wavelet decomposition can be done in linear time, $O(n)$ [58]. From Section 4.6, we know that the complexity of our algorithm is something between cubic and quadratic depending on implementation of the EKF algorithm. It is quite clear that the linear complexity of Harr wavelet decomposition does not add any burden to the complexity of the whole algorithm. Consequently, the speed performance of the Linear AR-Kalman and the Wavelet-Kalman models are very similar when they have same windowing size.

6.2. Possible future works

Generally the proposed model can be improved by two methods. First we can use more intelligent functions as a measurement function. In this thesis we tried three different functions. Perhaps some types of neural networks or non-linear regression function could also be incorporated with Kalman Filtering. Another aspect of the model which can be improved is the prediction step of EKF algorithm. In this work we assumed that the predicted values of the state variables are equal to its previous values,

$$\hat{X}_{t|t-1} = \hat{X}_{t-1|t-1} \tag{6.1}$$

Now instead of using this passive approach we might have a more educated guess. One example of such approach can be borrowed from neural network learning algorithms. The momentum technique [55] provides higher convergence rate for neural network learning, and probably the concept can be adapted here. It should also be noted that this technique is originally designed for first order gradient descent algorithm, and it proved to be very useful in deep learning [56]. Nevertheless the usefulness of some variation of this method for the EKF algorithm, which is a second order algorithm, should be thoroughly examined. Finally it is worth mentioning that Kalman Filtering is generally computationally expensive. That is true especially when we are dealing with state space of higher dimensions. The main reason behind this is the calculation of an inverse matrix during the KF algorithm. The size of this square matrix is equal to the size of state vector; therefore any improvement which requires more state variables is not quite the right direction to follow.

APPENDICES

Appendix A: Least square method

Least square method is used for estimation of unknown values in an overdetermined system. Gauss had discovered if a system of equation is written in the matrix form then the problem can be solved by finding an estimate for unknown variables,

$$\begin{bmatrix} h_{11} & h_{12} & \dots & h_{1m} \\ h_{21} & h_{11} & \dots & h_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n1} & h_{n2} & \dots & h_{nm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} \quad (A.1)$$

or in the compact form,

$$Hx = z \quad (A.2)$$

The estimate value of x is denoted by $\hat{x}$, and can be calculated by minimizing the estimate error. The estimate error is defined as Euclidian vector norm,

$$\varepsilon^2(\hat{x}) = |H\hat{x} - z|^2 \quad (A.3)$$

$$\varepsilon^2(\hat{x}) = \sum_{i=1}^{m} \left[\sum_{j=1}^{n} h_{ij}\hat{x}_j - z_j \right]^2 \quad (A.4)$$

To calculate the estimate value the derivative of equation (A.4) must be equal to zero,

$$\frac{\partial \varepsilon^2}{\partial \hat{x}_k} = 2\sum_{i=1}^{m} h_{ik}\left[\sum_{j=1}^{n} h_{ij}\hat{x}_j - z_i\right] = 0 \tag{A.5}$$

The last term of equation can be expressed as following,

$$\sum_{j=1}^{n} h_{ij}\hat{x}_j - z_i = \{H\hat{x} - z\}_i \tag{A.6}$$

hence the equation (A.5) can be written as,

$$2H^T[H\hat{x} - z] = 2H^T H\hat{x} - 2H^T z = 0 \tag{A.7}$$

or,

$$H^T H\hat{x} = 2H^T z \tag{A.8}$$

the equation (A.8) is known as “Normal Equation”. Finally when the normal equation is solved for $\hat{x}$ we have,

$$\hat{x} = (H^T H)^{-1} H^T z \tag{A.9}$$

The key term in the equation (A.9) is $H^T H$. This term is known as “Gramian Matrix”,

$$\mathcal{G} = H^T H \tag{A.10}$$

The $\mathcal{G}$ resolves how the equation (A.9) behaves. If the Gramian Matrix is non-singular (i.e., invertible) then $\hat{x}$ can be determined. On the other hand if the Gramian Matrix is singular (i.e., non-invertible) then the column vectors of H are linearly dependent therefore $\hat{x}$ cannot uniquely be determined. In this case we say the system is underdetermined. The equation (A.9) gives an approximate solution when no exact solution exists (overdetermined system), and when the exact solution does exist (determined system) it will provide us with the “exact” solution. The least square method is an optimal estimator when the system is overdetermined.

Appendix B: Dynamic Systems

A system is an assembly of interrelated or interdependent entities which can be seen as an integrated whole [7]. If some properties of a system change over time, then the system is time variant therefore called a "Dynamic System". Essentially anything which evolves through time can be considered as a dynamic system. Dynamic systems can broadly be categorized into linear and nonlinear.

Dynamic systems are usually described [7] by differential equations. The following differential equation describes a dynamic system with time-varying dynamic characteristics in general form,

$$\frac{d(x(t))}{d(t)} = f(t, x(t), u(t)) \tag{B.1}$$

$x(t)$ and $u(t)$ are vectors, and they are considered as state variables and input variables of the system respectively. The notion of "state variable" is quite important since these variables dictate how the system will react at a given time along with the corresponding input. The concept becomes clearer if we write the equation (B.1) in its extended format,

$$\begin{aligned} \frac{d\big(x_1(t)\big)}{d(t)} &= f_1\big(t, x_1(t), x_2(t), \dots, x_n(t),\ u_1(t), u_2(t), \dots, u_r(t)\big) \\ \frac{d\big(x_2(t)\big)}{d(t)} &= f_2\big(t, x_1(t), x_2(t), \dots, x_n(t),\ u_1(t), u_2(t), \dots, u_r(t)\big) \\ &\vdots \\ \frac{d\big(x_n(t)\big)}{d(t)} &= f_n\big(t, x_1(t), x_2(t), \dots, x_n(t),\ u_1(t), u_2(t), \dots, u_r(t)\big) \end{aligned} \tag{B.2}$$

As in can be seen variables x_1 ... x_n are all dependent, meaning that for finding their values, the given differential equation system must be solved. Here the only independent variable is the time (t).

In the equation (B.1) the state variables (x_1 ... x_n) and the input variables (u_1 ... u_r) are collected in two vectors. The first one (B.3) is the state vector and the second one (B.4) is the input vector.

$$x(t) = [x_1, x_2, x_3, \dots, x_n]^T \tag{B.3}$$

$$u(t) = [u_1, u_2, u_3, \dots, u_r]^T \tag{B.4}$$

The n value of state vector can change independently at any time; hence the degree of freedom of the system is equal to n.

The equation (B.1) is an instance of a continuous time system, nevertheless in the most engineering applications we are dealing with the discrete time. In this case the equivalent equation is written in the recursive format,

$$x(t_{k+1}) = f(t_k, x(t_k), u(t_k)) \tag{B.5}$$

given the time step is fixed ($t_k = k\Delta t$). We simply can rewrite it in the compact subscript format as well,

$$x_{k+1} = f(t_k, x_k, u_k) \tag{B.6}$$

where x_k and u_k are the state and the input vectors at the time t_k respectively.

In some cases what we want from a dynamic system is the state itself, however in greater majority of cases the state variables are used to get a variable or variables which are the actual properties of interest. These variables are considered as the "outputs" of the dynamic system. To make it clear we provide a simple example.

Consider a simple harmonic system. This dynamic system can fully be described by the speed and the position of the mass as the system variables (Figure B-1). Now if someone is interested on the energy of the system, then the system variables (speed and position) can be used to calculate it. The energy of a simple harmonic system is equal to the summation of the kinetic energy of the mass and the potential energy of the spring,

$$E = U + K = \frac{1}{2}kx^2 + \frac{1}{2}mV^2 \tag{B.7}$$

therefore for a simple harmonic system energy is a function of state variables,

$$E(t) = h(x_t, V_t) \tag{B.8}$$

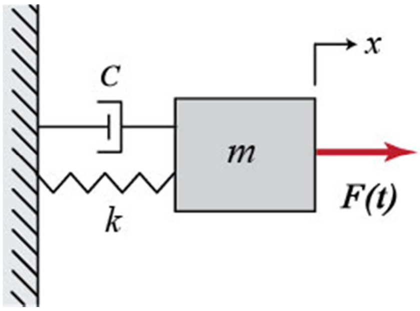


Figure B-1: Simple harmonic system

In general the outputs of a dynamic system depend on the state variables and the inputs of the system as well. Consequently the general form of the dynamic system outputs can be written as the following,

$$z = h(t, x(t), u(t)) \tag{B.9}$$

where z is the output vector of the dynamic system.

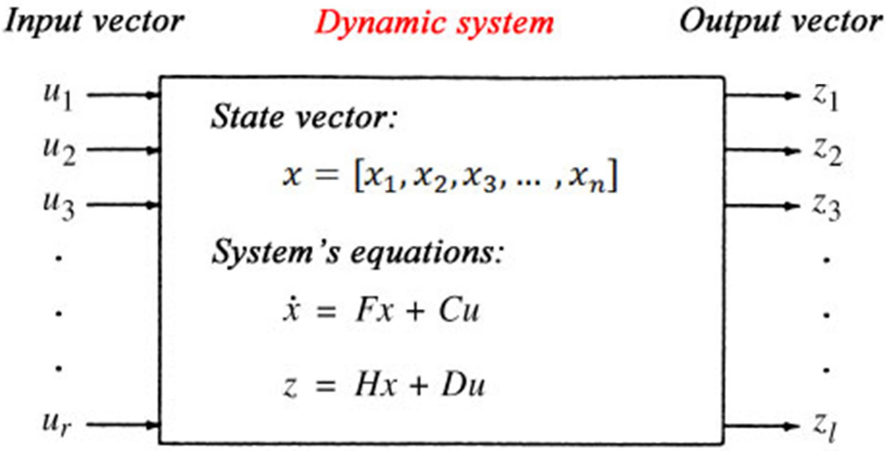


Figure B-2: Dynamic system block diagram [7]

Equation (B.1) and (B.9) are in general form. The linear form of these equations can be written respectively as following,

$$\frac{d(x(t))}{d(t)} = F(t)x(t) + C(t)u(t) \tag{B.10}$$

$$z(t) = H(t)x(t) + D(t)u(t) \tag{B.11}$$

where the $F(t)$, $C(t)$, $H(t)$ and $D(t)$ are time varying matrices. The matrix $F(t)$ is called "Dynamic Matrix", and its elements are known as "Dynamic Coefficients". Similarly the matrix $C(t)$ is called "Coupling Matrix" and its elements are best known as "Coupling Coefficients". For formulation of Kalman Filter problem the discrete form of the equations (B.10) and (B.11) are used.

Appendix C: Stochastic systems and Random Processes

If the dynamic system is relatively non-complex and with small number of components the system equations can be expressed explicitly. In this case, if the dynamic system is deterministic and it can be approximated to a linear form, then the prediction of the future state of the system is not mathematically a difficult task. Unfortunately in practice, what we are dealing with is a complex system possibly with a large number of components. Normally these systems are highly nonlinear in most cases. In this situation, it is quite clear that the deterministic approach will not work, and we have to think about a new approach. This new approach is “statistical”. In a nutshell by using statistical approach, the underlying dynamics of the system is treated as a random process and by combining the statistical and deterministic mathematical models, we will end up with “stochastic systems” [7].

The following equation provides us a deterministic model for the discrete process under investigation,

$$x(t_{k+1}) = f(t_k, x(t_k), u(t_k)) \tag{C.1}$$

The obvious implication of this equation is that the outcome of the equation at arbitrary time step $(k+1)$ only depends on the initial value of the state variables at first time step and the input vectors between first step and the step of interest $(k+1)$. This can be shown by recursive calculation of x_{k+1},

$$\begin{aligned} x_{k+1} &= f(t_k, x_k, u_k) \\ x_k \quad &= f(t_{k-1}, x_{k-1}, u_{k-1}) \\ x_{k-1} &= f(t_{k-2}, x_{k-2}, u_{k-2}) \\ &\vdots \\ x_1 \quad &= f(t_0, x_0, u_0) \end{aligned} \tag{C.2}$$

or,

$$x_{k+1} = f(f(f(f(\dots f(x_0, t_0, u_0), \dots), t_{k-2}, u_{k-2}), t_{k-1}, u_{k-1}), t_k, u_k) \tag{C.3}$$

Noticeably all intermediate values of x are eliminated, hence the output of this recursive function only depends on the initial state vector x_0 and known input vectors ($\{u_0,\ u_1,\ u_2, \dots,$ $u_k\}$). Clearly this model is not realistic as the real world systems with known inputs are not solely dependent on initial state [25]. It is quite well known that some level of uncertainty is always present in any natural or engineering process. This uncertainty can stem from the imperfection in design of an engineering process or complexity of the natural process under investigation; as all involved factors cannot be considered due to the enormous complexity. In this situation the corrective approach is to add another input vector whose values are not known to the dynamic system equation,

$$x_{k+1} = f(t_k, x_k, u_k, w_k) \tag{C.4}$$

By adding w_k the future state of the system is no longer bonded just to the initial state. The equation (C.4) can also be expressed slightly differently, if we consider that each time step has its own function, and consequently the time variable (t_k) can be taken out,

$$x_{k+1} = f_k(x_k, u_k, w_k) \tag{C.5}$$

Although w_k is assumed to be unknown but its statistical properties are considered to be known. In fact the sequence $\{w_0,\ w_1,\ w_2, \dots ,\ w_k\}$ is a stochastic process with known probability law, which means the joint probability distribution of the random vector w_k is known for each k.

The same notion can also be applied to the secondary system equation (B.9), hence it can be written as following,

$$z_k = h_k(x_k, u_k, v_k) \tag{C.6}$$

where the sequence of $\{v_0,\ v_1,\ v_2, \dots ,\ v_k\}$ is also a stochastic process which the joint probability distribution of its elements are known.

Appendix D: Random variables and probability

A random variable is basically a function which maps from its domain to real valued numbers. For instance the random variable $X(t)$ can maps time to position meaning that at any given time t the $X(t)$ gives us the “expected” position. The term “expected” emphasized on the randomness of the value. This indicates we don’t know what the position is before its occurrence. Instead of the exact value of the function, we have some probabilistic information about its value.

D.1 Probability

The probability of an event A is defined as the likeliness of the occurrence of that event in the sample space,

$$P(A) = \frac{Possible\ outcomes\ favoring\ event\ A}{Total\ number\ of\ possible\ outcomes} \tag{D.1}$$

If two events are mutually exclusive then the probability of an outcome favoring either $A\ or\ B$ is,

$$P(A\ or\ B) = P(A \cup B) = P(A) + P(B) \tag{D.2}$$

The joint probability of two independent events A and B is the likeliness of an outcome favoring both $A\ and\ B$,

$$P(A \; and \; B) = P(A \cap B) = P(A)P(B) \tag{D.3}$$

Finally, the conditional probability of A given B is defined as the outcome of event A given the occurrence of event B with $P(B) \; > \; 0$,

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{D.4}$$

D.2 Cumulative Density Function

The cumulative distribution function (CDF) is defined as the probability of a real valued random variable (e.g. $X(t)$) to be found to acquire a value less or equal to a specific number (e.g. x),

$$F_X(x) = p(-\infty, x] \tag{D.5}$$

D.3 Probability Density Function

The derivative of the equation (D.5) is more common and known as the Probability Density Function (PDF). The PDF is none negative function,

$$f_X(x) = \frac{d}{dx} F_X(x) \tag{D.6}$$

The PDF is used for calculation of the probability over any interval $[a, b]$,

$$P_X[a,b] = \int_a^b f_X(x)dx \tag{D.7}$$

REFERENCES

[1] A. Sadeghian and J. D. Lavers, “Dynamic reconstruction of nonlinear v-i characteristic in electric arc furnaces using adaptive neuro-fuzzy rule-based networks,” Appl. Soft Computing., vol. 11, no. 1, pp. 1448–1456, Jan. 2011.

[2] Y. N. Toulouevski, I. Y. Zinurov, “Electric Arc Furnaces Scientific Basis for Selection,” Springer Heidelberg Dordrecht London New York. 2010, ISBN: 978-3-642-03800-6 (Print) 978-3-642-03802-0 (Online).

[3] G. W. Chang, M. F. Shih, Y. Y. Chen, and Y. J. Liang, “A Hybrid Wavelet Transform and Neural-Network-Based Approach for Modelling Dynamic Voltage-Current Characteristics of Electric Arc Furnace,” IEEE Transactions on Power Delivery, Vol. 29, No. 2, April 2014, Pages 815 – 824.

[4] A. Sadeghian and J. D. Lavers, “Application of radial basis function networks to model electric arc furnaces in Neural Networks,” 1999. IJCNN '99. International Joint Conference On, 1999, pp. 3996-4001 vol.6.

[5] A. A. Mahmoud, R.D. Stahlhut, “Modeling of a resistance regulated arc furnace,” IEEE Trans. Power Apparatus Syst. PAS-104 (January (1)) (1985) 58–66.

[6] E. Acha, A. Semlyen and N. Rajakovic, “A harmonic domain computational package for nonlinear problems and its application to electric arc,” IEEE Trans. Power Delivery 5 (July (3)) (1990) 1390–1397.

[7] M. S. Grewal, A. P. Andrews, “Kalman Filtering: Theory and Practice Using MATLAB,” Wiley-interscience Publication, second edition, 2001, ISBN 0-471-39254-5.

[8] M. I. Ribeiro, “Kalman and Extended Kalman Filters: Concept, Derivation and Properties,” Institute for Systems and Robotics. Instituto Superior Técnico, 2004.

[9] R. E. Kalman, “A New Approach toad Linear Filtering and Prediction Problems,” Research Institute for Advanced Study Baltimore, Md. 1960.

[10] L. A. McGee and S. F. Schmidt, “Discovery of the Kalman Filter as a practical tool for Aerospace and Industry,” Ames Research Center, NASA, Technical Memorandum 86847, 1985.

[11] N. Funk, “A Study of the Kalman Filter applied to Visual Tracking,” University of Alberta, December 7, 2003.

[12] D. Simon, “Training fuzzy systems with the extended Kalman filter,” Fuzzy Sets and Systems, Volume 132, Issue 2, 1 December 2002, Pages 189–199.

[13] P. Trebatický, J. Pospíchal, "Neural Network Training with Extended Kalman Filter Using Graphics Processing Unit," Lecture Notes in Computer Science Volume 5164, 2008, pp 198-207, Artificial Neural Networks - ICANN 2008.

[14] F. Conte, A. Germani, G. Iannello,"A Kalman filter approach for de-noising and de-blurring 3-D microscopy images," IEEE Trans Image Process. 2013 Dec; 22(12):5306-21.

[15] L. H. Matthies, Szeliski, T. Kanade, "Kalman filter-based algorithms for estimating depth from image sequences," Carnegie Mellon University, Research Showcase at CMU, Computer Science Department, 1987

[16] H. Medeiros, J. Park, A. C. Kak, "Distributed Object Tracking Using a Cluster-Based Kalman Filter in Wireless Camera Networks, IEEE Journal Of Selected Topics In Signal Processing," Vol. 2, No. 4, August 2008.

[17] W. Hwang, H. Kwon, J. Kim ; C. Lee, M. L. Anjum, K. Kim, D. Cho, "High performance vision tracking system for mobile robot using sensor data fusion with Kalman filter," The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, October 18-22, 2010.

[18] J. Sum, C. Leung, G. H. Young, W. Kan, "On the Kalman Filtering Method in Neural-Network Training and Pruning," IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 10, NO. 1, JANUARY 1999.

[19] O. S. P. Bojarczak, M. Stodolski, "Fast second-order learning algorithm for feedforward multilayer neural networks and its application," Neural Networks, vol. 9, no. 9, pp. 1583–1596, 1996.

[20] S. Haykin, "Neural Networks: A Comprehensive Foundation," New York: Macmillan, 1994.

[21] A. Tarantola, "Inverse Problem Theory and Methods for Model Parameter Estimation," Institut de Physique du Globe de Paris Université de Paris 6, Paris, France, Society for Industrial and Applied Mathematics 2005, ISBN 0-89871-572-5 (pbk.)

[22] H. W. Sorenson, "Least square estimation: from Gauss to Kalman," IEEE Spectrum, vol. 7, pp. 63-68, July 1970.

[23] E. R. Scheinerman, "Invitation to Dynamical Systems," Department of Mathematical Sciences, The Johns Hopkins University, 2000, ISBN 0-13-185000-8

[24] T. Kailath, A. H. Sayed, B. Hassibi, "Linear Estimation," Prentice Hall, 2000, ISBN 0-13-022464-2

[25] P.R. Kumar, University of Illinois, P. Varaiya, University of California ,“Stochastic Systems - Estimation, Identification and Adaptive Control,” Prentice-Hall Inc., 1986, ISBN 0-13-846684-X 025

[26] M. Brugnach (Uniyersity of Osnabrlick), C. Pahl-Wostl (Uniyersity of Osnabrlick), K. E. Lindenschmidt (GeoForschungsZentrum Potsdam Sektion), J. A.E.B. Janssen (University of Twente),T. Filatova (University of Twente), “Complexity and Uncertainty: Rethinking The Modelling Activity,” University of Nebraska – Lincoln, U.S., Environmental Protection Agency Papers, United States Environmental Protection Agency, 2008

[27] M. Athans, “Dynamic Stochastic Estimation, Prediction and Smoothing,” Series of Lectures, Spring 1999.

[28] G. Welch, G. Bishop, “An Introduction to the Kalman Filter,” University of North Carolina at Chapel Hill Department of Computer Science Chapel Hill, NC 27599-3175, 2001

[29] H. Schichl, Institut f¨ur Mathematik der Universit¨at Wien Strudlhofgasse 4, A-1090 Wien, Austria, “Models and History of Modeling,” 2004

[30] Y. Y. Lu, “Numerical Methods for Differential Equations,” Department of Mathematics, City University of Hong Kong, Kowloon, Hong Kong

[31] A. Gröβle, P. M. Milling, “Inductive and Deductive System Dynamics Modeling,” Radboud University, Nijmegen, the Netherlands

[32] B. Boulet, G. Lalli, M. Ajersch, Centre for Intelligent Machines McGill University, “Modeling and control of an electric arc furnace,” American Control Conference, 2003. Proceedings of the 2003 (Volume:4)

[33] M. Baranger, “Chaos, Complexity, and Entropy: A physics talk for non-physicists”, Center for Theoretical Physics, Laboratory for Nuclear Science and Department of Physics Massachusetts Institute of Technology, Cambridge, MA 02139, USA, New England Complex Systems Institute, Cambridge, MA 02138, USA, MIT-CTP-3112, URL: http://necsi.edu/projects/baranger/cce.pdf

[34] N. Hariyanto, M. Nurdin, P .G. A. Tanthio, “Characteristic Study of Three-phase AC Electric Arc Furnace Model,” IEEE, Power Engineering and Renewable Energy (ICPERE) 2014

[35] O. Ozgun, A. Abur, “Flicker Study Using a Novel Arc Furnace Model,” IEEE Transactions on Power Delivery, vol. 17, no. 4, October 2002

[36] R. Collantes-Bellido and T. Gomez, “Identification and modeling of a three phase arc furnace for voltage disturbance systems,” IEEE Trans. Power Delivery, vol. 12, pp. 1812–1817, Oct. 1997.

[37] H. M. Petersen, R. G. Koch, P. H. Swart and R. V. Heerden, “Modelling arc furnace flicker and investigating compensation techniques,” in Industry Applications Conference, 1995.

[38] L. R. Rabiner , B. H. Juang, “An Introduction to Hidden Markov Models,” IEEE ASSP MAGAZINE JANUARY 1986

[39] M. T. Esfahani and B. Vahidi, “A New Stochastic Model of Electric Arc Furnace Based on Hidden Markov Model: A Study of Its Effects on the Power System,” IEEE Transactions on Power Delivery, Vol. 27, No. 4, October 2012.

[40] A. R. Sadeghian, J. D. Lavers, “Nonlinear Black-Box Modeling of Electric Arc Furnace: An application of Fuzzy logic systems,” IEEE International Fuzzy Systems Conference Proceedings, Seoul Korea, August 1999

[41] G. W. Chang, Ch. I. Chen, “Neural-Network-Based Method of Modeling Electric Arc Furnace Load for Power Engineering Study,” IEEE Transactions on Power Systems, Vol. 25, No. 1, February 2010

[42] S. Chen, C. F. N. Cowan, and P. M. Grant, “Orthogonal least squares learning algorithm for radial basis function networks,” IEEE Trans. Neural Network, vol. 2, no. 2, pp. 302–309, Mar. 1991.

[43] S. Mishra, C. N. Bhende, and B. K. Panigrahi, “Detection and Classification of Power Quality Disturbances Using S-Transform and Probabilistic Neural Network,” IEEE Transactions on Power Delivery, Vol. 23, No. 1, January 2008

[44] S. Santoso, E. J. Powers, and W. Grady, “Power quality disturbance identification using wavelet transformers and artificial neutral network,” in Proc. Int. Conf. Harmonic and Quality of Power, Las Vegas, NV, 1996, pp. 615–618.

[45] J. G. DeGooijer, R. J. Hyndman, “25 years of time series forecasting,” Elsevier, International Journal of Forecasting 22 (2006) 443 – 473

[46] A.C. Harvey, “Forecasting, Structural Time Series Models and the Kalman Filter,” Cambridge University press (2001), ISBN: 978-0-521-32196-9 hardback

[47] M. Han, J. Xi, and S. Xu, "Chaotic system identification based on Kalman filter," Proceedings of the 2002 International Joint Conference on Neural Networks, Honolulu, USA, 2002, vol.1, pp. 675-680.

[48] J. Ym, V. L. Syrmos, D. Y. Y. Yun, “System identification using the extended Kalman filter with application to medical imaging,” Proceedings of the 2000 American Control Conference, Chicago, USA, 2000, vol.5, pp. 2957-2961.

[49] S. Haykin, “Kalman Filtering and Neural Networks,” John Wiley & Sons, Inc., ISBNs: 0-471-36998-5 (Hardback); 0-471-22154-6 (Electronic)

[50] F. Wang, Z. Jin, Z. Zhu, X. Wang, “Application of Extended Kalman Filter to the Modeling of Electric Arc Furnace for Power Quality Issues,” Neural Networks and Brain, 2005. ICNN&B '05. International Conference on (Volume: 2)

[51] S. L. Chiu, “Fuzzy model identification based on cluster estimation,” J. Intell. Fuzzy Systems 2 (1994) 267–278

[52] http://www.mathworks.com/help/fuzzy/subclust.html

[53] A. N. Akansu, R. A. Haddad, “Multiresolution Signal Decomposition Transforms, Subbands, and Wavelets,” Second Edition, ACADEMIC PRESS, ISBN: 0-12-047141-8

[54] S. G. Mallat, “A Theory for Multiresolution Signal Decomposition: The Wavelet Representation,” IEEE Transactions on Pattern Analysis And Machine Intelligence. Vol 2. No. 7. July 1989.

[55] N. Qian, Center for Neurobiology and Behavior, Columbia University, “On the momentum term in gradient descent learning algorithms,” Neural Networks 12 (1999) 145–151, accepted 6 August 1998

[56] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton. “On the importance of initialization and momentum in deep learning.”, In Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013, volume 28 of JMLR Proceedings, pages 1139–1147. JMLR.org, 2013.

[57] V. V. Williams, “Multiplying matrices in $O(n^{2.373})$ time,” Stanford university, July 1, 2014

[58] D. Sundararajan, “Fundamentals of the Discrete Haar Wavelet Transform,” 2011. www.dsprelated.com/Documents/d_sundararajan_lpaper.pdf