Applications of an Extended Kalman Filter in nonlinear mechanics

by

Azeem Iqbal
ID# 15026050012

Supervisor: Dr. Muhammad Umar Suleman
External Supervisor: Dr. Muhammad Sabieh Anwar

in the

Department of Computer Science
School of Systems and Technology
University of Management and Technology

Spring 2019

Research Completion Certificate

It is certified that the research work contained in the thesis APPLICATIONS OF AN EXTENDED KALMAN FILTER IN NONLINEAR MECHANICS has been conducted under my supervision to my satisfaction by Mr. AZEEM IQBAL, ID, 15026050012, of MASTER OF SCIENCE IN COMPUTER SCIENCE program.

Signature: _____

Date: _____

Supervisor: Dr. Muhammad Umar Suleman

The Thesis titled

Applications of an Extended Kalman Filter in nonlinear mechanics

by

Azeem Iqbal

ID. 15026050012

has been examined by the undersigned panel of examiners and has received full approval for acceptance in partial fulfillment for the degree of

MS IN COMPUTER SCIENCE

Dated this _____

Dr. Muhammad Umar Suleman	_____
Supervisor	Signature
Dr. Muhammad Sabieh Anwar	_____
External Supervisor	Signature
Dr. Muzammil Hussain	_____
Director Graduate Studies	Signature

School of Systems and Technology

University of Management and Technology

Declaration of Authorship

I, AZEEM IQBAL, ID No. 15026050012 Session FALL 2015, hereby certify that this thesis titled, 'APPLICATIONS OF AN EXTENDED KALMAN FILTER IN NONLINEAR MECHANICS' is being submitted in partial fulfillment of the requirements for the degree in MASTER OF SCIENCE IN COMPUTER SCIENCE.

This thesis is my original work, and the data/material presented herein has not been used for the acquisition of any other degree from any institution.

The Similarity Index is below permissible limit.

Researcher Signature: _____

Date: _____

Researcher: Azeem Iqbal

Confirmed by

Supervisor's Signature: _____

"In the Name of Allah, the Most Gracious, the Most Merciful."

Abstract

Data processing is a challenging task in the realms of experimental physics. An experimental physicist is poised with statistical and probabilistic techniques that enable to cater uncertainties and find the unknown parameters. But in a teaching laboratory today there are transducers, highly susceptible to noise due to sensitivity. This is desirable in the context of exposing students to the realities of a real world experiment and also creates a potential for them to learn about data cleaning and finding the embedded unknowns.

In this regard, we propose the use of Kalman Filters to not only filter the system but also estimate its unknown parameters. In the $1960s$, the Kalman filter was applied to navigation system of the Apollo Project. It was a pivotal invention that enabled humans to reach moon. This application propelled Kalman filtering into the dynamical linear and nonlinear research of that time. It was explored in signal processing, navigational radar systems, robotics, GPS systems and in recent years in image processing and machine learning.

This research work concentrates on deploying Kalman Filter and its two variants in an experimental physics teaching laboratory. We explore the notions of filtering linear and non-linear mechanics problems, simulate the applications and then heuristically develop a sound understanding of the Kalman Filter and its facets in experimental physics.

Acknowledgements

In the Name of Allah, the Most Merciful, the Most Compassionate all praise be to Allah, the Lord of the worlds; and prayers and peace be upon Muhammad His servant and messenger.

First and foremost, I must acknowledge my limitless thanks to Allah, the Ever-Magnificent; the Ever-Thankful, for His help and blessings. I am sure that this work would have never become the truth, without His guidance.

I owe a deep debt of gratitude to my external supervisor Dr. Muhammad Sabieh Anwar, the Associate Professor of Physics at the Lahore University of Management Sciences (LUMS) and the founding member of the *Experimental Physics Laboratory* named *Physlab* at the said university. It was my utmost desire to work under his supervision and I thank the Almighty for providing me an opportunity to be his student. This research work has been carried out at Physlab under his supervision. He has provided me with valuable critique and insights to exploring deeper into every phase of this research work and has always been extremely generous in enlightening my thoughts with his wisdom and sea of knowledge.

I also highly appreciate the efforts expended by my supervisor Dr. Muhammad Umer Suleman, the Assistance Professor of Computer Science at the University of Management and Technology (UMT) for providing me continuous support and guidance throughout the course of completing this research work.

I also would like to express my wholehearted thanks to my mother and father for providing me the untiring support throughout my entire life! They are the soul that enlivens me every day to pursue challenging tasks and accomplish great goals. I would also like to thank my in-laws and especially my lovely wife for the unwavering patience and strong support in the process of pursuing this research work. Undoubtedly, without the support of my beautiful and strong family, I would have not been able to complete this thesis.

I would also like to thank my colleagues at Physlab for all their support. Especially, Dr. Ammar Khan (Assistant Professor of Physics), Mehran, Waseem and Adee (Lab Instructors) at the Physlab for taking up my lectures and lab work so that I could focus on my research work.

Contents

List of Figures

List of Tables

Abbreviations

LKF	**L**inear **K**alman **F**ilter
SKF	**S**calar **K**alman **F**ilter
EKF	**E**xtended **K**alman **F**ilter
UKF	**U**nscented **K**alman **F**ilter
MKF	**M**ultivariate **K**alman **F**ilter
KBF	**K**alman **B**ucy **F**ilter
KBWF	**K**alman **B**ucy **W**ater **F**ilter
QKF	**Q**uantum **K**alman **F**ilter
ASME	**A**merican **S**ociety of **M**echanical **E**ngineers
ZOH	**Z**ero **O**rder **H**old
DAC	**D**igital to **A**nalog **C**onverter
LQG	**L**inear to **Q**uadratic **G**aussian

Symbols

x	distance	m
P	power	W (Js^{-1})
m	mass	Kg
ω	angular frequency	$rads^{-1}$
ω_z	frequency along z direction	$rads^{-1}$
ω_θ	rotational angular frequency	$rads^{-1}$
k	spring constant	Nm^{-1}
δ	torsional constant	Nm^{-1}
w_k	zero mean white noise process	
v_k	zero mean white noise measurement	
A	System dynamics matrix	
B	Input matrix	
C	Output matrix	
D	Input-output matrix	
F	Discrete state transition matrix	
K	Kalman filter gain	
P	Error covariance matrix	
$\hat{\mathrm{P}}$	Error covariance matrix	
Q	Process noise covariance matrix	
R	Measurement noise covariance matrix	
U	Input vector	

x	State vector
$\hat{\mathrm{x}}$	Estimated state vector
y	Vector of outputs
Γ	Linearized plant noise matrix
θ	Vector of parameters
Λ	Linearized measurement noise matrix
x	Augmented state vector
σ^2	Covariance
Σ	Linearized output matrix
Φ	State transition matrix
ϕ	Nonlinear mapping matrix
$\mathcal{O}$	Observability matrix
$\mathcal{C}$	Controllability matrix

This thesis is dedicated to my mother, the greatest influence on my life, to my father the strongest influence on my life, to my wife for her patience, love and endless support to my son, Haider for his encouragement by keeping me awake at nights to work on the thesis, to my grand parents and especially my grandfather who was my source of scientific enlightenment while sitting for hours with him asking tons of questions and getting all my curiosity of nature answered.

Chapter 1

Introduction

In 1960, Rudolf E. Kalman, proposed a linear quadratic estimation (LQE) technique [1]. This technique involved a recursive algorithm that takes into account a series of measurements on the state that are affected by varying statistical noise over time and then outputs an estimation of the state that is more precise and accurate. The technique was named as "Kalman Filter (KF)". It transformed the world of signal processing and empowered researchers to develop highly sophisticated navigation systems [2], robotic systems, enhanced image processing and object tracking [3] as well as predictive economics [4] and stock forecasting systems [5]. It is considered among the three most influential and revolutionary algorithms, in the field of Bayesian State Estimation [6]. It is also one of the very few algorithms that researchers can call "complete" [7] as it solves the state estimation for linear Gaussian systems in an optimized manner.

The filter was originally developed for linear problems in the world of control systems but over time, subsequent versions attempted to relax the requirements for a linear Gaussian system and made it applicable to nonlinear systems making it more versatile and robust. One such extension of the original filter is known as the *Extended Kalman Filter* [8]. This filter linearizes a non-linear system using approximation techniques in order to find the transit model of the state dynamics to predict the future state.

Anyone who desires to build the mathematical and conceptual foundations of the filter, is invited to read the several primers on the subject [9, 10]. Similarly, this thesis not only exemplifies the use of Kalman Filters for nonlinear problems, but

is also a compact primer on the subject and will hopefully become an entry level spring board for experimenters especially in the domain of physics.

The detailed insight to filter's theoretical background, derivation of equations, the concept and design formulation, problem applications and performance gauging can also be found in [10]. Similarly, when one would like to explore hands-on applications, [11] is a great book to consult for programming the Kalman Filter and its variants using MATLAB. Recently, Python has emerged as an easy and versatile programming language. An outstanding book [12] provides vivid and dynamic hands-on python codes with breakdown of the filter structure. It gradually builds up from the concepts of Gaussian filters through hard coded examples and then quickly moves on to Kalman Filter and its variants. Interestingly, the Kalman Filter, can also be studied in the realms of finance and stock market predictions [13, 14].

The most frequent use of Kalman Filters, however, has been for the application of computer vision in object tracking [15]. It ranges from tracking projectile balls to airplanes and missiles through radar systems. Last, we highly recommend [16] which explores the conjunction of the Kalman filter and neural networks. Neural networks and machine learning are techniques that also derive their existence from Bayesian statistics coupled with prediction and estimation models.

This means that the Kalman Filter is a versatile algorithm that has been explored in vast areas of engineering, computer science and even management but a literature review reveals that it is still largely unexplored in the community dealing with physics laboratory instruction. Even though, the filter has shown its ability to filter and estimate dynamical systems such as projectile motions [17] but in the realms of the physics laboratory, it remains largely unexplored and not taught to students in physics who could utilize it to explore various linear and nonlinear problems. Therefore, in this research work we explore the applications of Kalman Filter particularly inside an experimental physics laboratory.

This work is entirely carried out at the *Centre for Experimental Physics Education (CEPE)* at the *Physics Department* of *Lahore University of Management Sciences (LUMS)* [1]. The centre arguably houses one of the most innovative experimental physics facility in the country. With an in-house mechanical workshop and a group of professional and highly skilled engineers and technicians, the laboratory builds

[1] http://www.physlab.org

Application	Dynamical System	Type of sensor
Process control	Chemical plant	Pressure
		Temperature
		Flow Rate
		Gas analyzer
Flood Prediction	River system	Water level
		Rain gauge
		Weather radar
Tracking	Spacecraft	Radar
		Imaging system
Navigation	Ships	Sextant
	Aircrafts, missiles	Log
	Smart bombs	Gyroscope
	Automobiles	Accelerometer
	Golf carts	GPS Receiver

TABLE 1.1: Examples of estimation problems that are currently widely solved using Kalman Filter.

its own experimental equipment using low-cost locally manufactured hardware and software. Due to this capacity building a number of student driven experimental projects have been carried out in the last decade. [2]. The research and development group at this lab has developed a number of home-grown experimental equipment diversely ranging from applications in mechanics, waves and oscillations, eletromagnetism and other branches of physics. The lab also ingeniously utilizes smart-phones and Inertial Meaurement Units (IMU)'s for physics experiments. Various sensors and transducers are also employed to measure temperature, pressure and strain and various other kinds of physical signals. In summary, expents the *Physlab* served as the playground to explore the applications of Kalman Filter.

1.1 Problem statement

In any experiment, "Noise", infests into signals and propagates through the experimental process to the end result, giving rise to imprecision and accuracy. In a physics laboratory, precision and accuracy are important measures that determines the stochastic acceptance of the outcome of an experiment. Thus, anything

[2] https://www.physlab.org/

that may contribute to disturbing the precision of an experiment needs to be address and quantified. Currently, techniques such as least squares curve fitting are the most widely used data processing algorithm that allows experimenters to fit proposed mathematical models to their noisy experimental data. The method involves minimizing the sum of the residuals between model and measured values. By large, it not only helps physicists verify theoretical models of various linear and non-linear physical phenomena but may also help predict future behavior of the system through extrapolation. However, here we would like to propose the use of Kalman Filtering as a modern technique for improving data integrity and reducing noise. This idea is currently underemployed in the realms of instructional physics laboratory. Students are generally taught only about the least squares curve fitting as the most optimum method for gauging the experiment's conformance to the true model. However, the process is largely part of the post data acquisition and analysis process where there could be an inclusion of an optimal estimator to not only filter the data but to also estimate the parameters and dynamics of the system over time.

1.2 Motivation

As a laboratory instructor at the Centre for Experimental Physics Education (CEPE), Physlab, LUMS based in Lahore, Pakistan I have also heavily relied on conventional data processing methods for data analysis. However, I took this project of deploying Kalman Filter to our in-house manufactured experiments as a challenge to contribute to the physics community of educators and experimenters to help them see how efficiently and elegantly the Kalman Filter can help them improve data integrity, reduce noise, predict system dynamics and estimate parameters. I hope this thesis not only helps a physics teacher gain a sound understanding of the basic application of Kalman Filter but also provides insight to other researchers to explore the algorithm's applications to more complex problems in non-linear physics.

Furthermore, students in Pakistan's physics programs, are not well versed with data analysis, noise and uncertainties. Noise and uncertainties are generally considered an anathema to lab practice. The quest for a utopian certainty, and accuracy, hides away the peculiar statistical fluctuations embedded in an experimental

endeavor. Kalman Filtering provides a unique, insightful and far-reaching tool to bridge the gap between theoretical certitude and empirical openness.

1.3 Objectives of the study

Through this work we aim to achieve following objectives:

- introducing the basics of Kalman Filtering and its variants, the Extended Kalman Filter and Unscented Kalman Filter,
- understanding the fundamentals of state space and application specific facets of the Kalman Filter,
- exploring the diversity and robustness of Kalman Filter in the realms of experimental physics laboratory,
- using Kalman Filter to filter noise from linear and nonlinear systems and estimating the state in the presence of measurement and process noise,
- utilizing filter dynamics to estimate the sub-parameters of the system upon which the dynamics of the system depend over time,
- applying Kalman Filter to home grown physics experimental setups, and
- providing an impetus to an experimental physicist to use Kalman Filter in the physics laboratories at the undergraduate and post graduate levels.

1.4 Limitations of the study

Kalman Filters find application in various domains of electrical engineering and computer science, such as, orientation and location signal processing. They are also explored in financial forecasting and prediction systems finding applications in stock market. However, the scope of this thesis is to explore the realms of *Kalman Filter* applications in an experimental physics laboratory. We want to investigate how the filter could prove to be useful for physics teachers who would be interested in teaching their students the art of noise filtration through state estimation and how it could also be used for parameter estimation.

We have also not tried to cover the complex probabilistic mathematical derivations of the filter's underlying structure. Rather, we have focused on keeping the derivations simple and intuitive for the reader to easily understand and use the provided MATLAB codes to build their own Kalman Filters through simple reading of the sections according to their specific applications in linear or nonlinear regime.

Further, this thesis can be used to explore applications in an experimental physics laboratory. The notions of coupled harmonics, linear and nonlinear pendulums, projectile motions, damped and simple harmonic oscillators, thermodynamics, fluid mechanics, nonlinear electronic circuits and experiments involving mechanical explorations all offer huge opportunities for investigation and exploration.

1.5 History of the Kalman filter

The Kalman Filter is named after a Hungarian-born American electrical engineer, mathematician, and inventor, *Rudolf E. Kalman*. He is believed to have co-invented the filter with *Richard S. Bucy* from the University of Southern California. This is the reason why sometimes Kalman Filter is also referred to as "Kalman-Bucy Filter". Professor Bucy contributed to the further development of the theoretical framework of the filter and since then, has written a number of books on filtering and stochastic processes. Interestingly, Kalman being an electrical engineer was unable to submit his phenomenal work in the community of electrical engineers and so had to submit his first paper [1] to the Journal of Basic Engineering by the American Society of Mechanical Engineers.

The first practical application of the Kalman's ingenious algorithm was in the first Apollo mission to moon. *Stanley F. Schmdit*, who was an Aerospace Engineer at the NASA Ames Research Center, discovered Kalman's ground breaking work and found it to be useful for the applying to the navigation system. The only problem he faced was that Kalman had developed the filter for linear problems through linear quadratic estimation technique whereas for the Apollo mission nonlinear version of the filter was required. Stanley worked a nonlinear adaptation which reduced the complexity of the filter and developed the first practical application of the nonlinear verion of the Kalman Filter, for this reason the filter is also referred to as *Schmidt-Kalman filter* [18]. However, in control engineering,

Kalman-Bucy Filter has served as the most spectacular failure in one of its another applications, the Kalman-Bucy Water Filter (KBWF) [19]. Despite gaining an immediate patent ($US314,159$), this product never found its market niche. The KBWF was used for continuous water flows, while related inventions, such as the Kalman water filter (KWF) and the extended Kalman water filter (EKWF), were used for bottled water and for filtering nonclear liquids, respectively. The patent abstract describes the operation of the filter: Tap water flows into the input hose y, where it is compared to the filtered water 'y'. The residual water goes through the separation pump 'L', which feeds the water state estimate differential 'x', into the water transition tank. The output 'x' of the water transition tank is extracted with a combination pump 'H', to provide filtered water 'y'. The separation pump is adjusted through the solution of a Riccati equation. The resulting water is optimally clean while the filter retains the residual sediment, which tends to be white.

The Kalman Filter was then also implemented in the navigation systems of the U.S. Navy and nuclear ballistic missile submarines [20]. The filter helped improve the guidance and navigation of the cruise missiles such as the U.S. Navy's Tomahawk missile and the U.S. Air Force's Air Launched Cruise Missile [20]. They are also used in the guidance and navigation systems of reusable launch vehicles and the attitude control and navigation systems of spacecraft which dock at the International Space Station.

1.6 Diverse applications

In our extensive literature review, we were unable to find work related to application of Kalman Filter in a teaching experimental physics laboratory. However, there numerous examples in various domains of physical and engineering sciences where it is already playing a pivotal role in advancing the data integrity, noise elimination, dynamic state predictions and parameter estimation.

1.6.1 Electrical and electronics engineering

An electronic device for three dimensional localization and inertial navigation through vision has been patented in the US [21]. It comprises a processor configured to apply an extended Kalman filter (EKF) as the electronic device traverses the trajectory. The filter is configured to maintain an array of estimates for a position of the electronic device within an environment along with estimates for one or more features. Another study has explored the application of the Kalman filter in dynamic X-ray tomography. [22]. With this method, the X-ray image is reconstructed through a low-dimensional pool of parameters. This has shown to optimize the computational speed of the overall process. These two examples are distinct but surely exhibit the robustness of Kalman Filter. Another study discusses the implementation of a Kalman variant called "Ensemble Kalman Filter" [23] presents a new algorithms using *Ensemble Kalman Filter* for sequential state and parameter estimation that combine the information about the parameters from data at different time points in a consistent probabilistic framework. This is helpful when we have a large number of parameters under observation.

1.6.2 Physics

In physics, the modeling of the thermosphere-ionosphere system is largely associated with uncertainty due to influences of external forces, e.g., high altitude convection and particle precipitation. In order to gauge the system, real-time measurements are acquired making the computation complex and extensive [24]. Therefore, through the implementation of the Kalman Filter reseachers have proposed improvements in comparison to the experiments being performed.

The Kalman Filter is primarily a stochastic filter that uses a series of measurements over time to produce estimates of unknown variables based on a dynamic model. In a quantum system, such an algorithm is provided by a quantum filter. For any linear quantum system that is dependent upon measurements and noise that is a white Gaussian distribution, the quantum filter reduces to a Quantum Kalman Filter (QKF). Using a commutative approximation and a time-varying linearization to non-commutative quantum stochastic differential equations (QSDEs) show that there are conditions under which a filter similar to the classical

Extended Kalman Filter can also be implemented for a quantum system. Interestingly, they have demonstrated the effectiveness of the Quantum Extended Kalman Filter by applying it to diverse quantum systems. Most of these systems typically involve multiple modes, nonlinear Hamiltonians and simultaneous jump-diffusive measurements [25].

The most recent and probably the most momentory implementation of Kalman Filter in Physics is in the discovery of gravitational waves [26]. The highly sensitive and peculiar *Laser Interferometer Gravitational-Wave Observatory (LIGO)* is a large scale physics instrument housed at the *Hanford Site, Livingston, USA*. The device is extremely sensitive and can detect a change in the 4 km mirror spacing of less than a ten-thousandth of the charge diameter of a proton, equivalent to measuring the distance from Earth to Proxima Centauri (4.244 ± 0.001) lyr [27] with an accuracy smaller than the width of a human hair [28]. Correspondingly, the system is sensitive to slightest change in the physical distance between the mirrors. In ground based tests, suspended pendula serve as the test masses approximated as "free" above the pendulum's fundamental frequency. The Gravitational waves caused a change in the distance between those pendulums. However, external influences such as thermal excitation may impart a narrow-band noise into the signal in turn hindering the detection of gravitational waves. Researchers design a sophisticated Kalman Filter to filter and estimate the state of position of masses from the detector's output.

A more comprehensive and detailed article in this realm talks about testing quantum mechanics itself using statistical approach [29]. The crux of the paper is based upon the fact that exploration of quantum versus classical system is limiting due to the increasing complexity in the dynamical systems. Also, the noisy and uncertain nature of the experimental quantum systems make it difficult to be sure about what quantum system is being observed. So, the paper highlights use of advanced statistical and Bayesian estimation techniques coupled with Kalman Filters employing an example from optomechanics.

A Markov diffusion process is characterized by a stochastic differential equation in a continuous-time process. However, a new approach of its quantization using Kalman Bucy linear filters is discussed in [30]. As KF is a self correcting algorithm, it follows the automorphism of Heisenberg algebra giving rise to new formulas and interpretations of the quantum mechanical systems [31].

1.6.3 Computer science and robotics

The most evident use of Kalman Filter has been in the realms of robotics and computer science. This is because the first application of Kalman Filter was in fact for a large space ship that we had to steer to reach the Moon. The dynamical control and noise free sensor feedback became the need of its immediate and visible application. Since then, Kalman Filters have extensively contributed to the optimization of robotic movements, tracking, and localization.

The power of Kalman Filter in robotics application comes from its ability to fuse multiple sensors, referred to as *Sensor Fusion*. The probabilistic combination model of the filter empowers the robot designer to include the feedback and response from multiple sensors, actuators and controls from the robotic system and derive the system based upon the filtered output. The major application in this realm is integration of GPS and IMU sensors to track the location of a robot [32].

1.6.4 Industrial engineering and management sciences

Industrial engineering deals with the optimization of systems and processes that involve people, money, machine and sub-processes that govern the nature of the overall system. In this context, Kalman Filters have been widely used for optimizing the mathematical models, manufacturing systems, supply chain management and predictive maintenance.

State estimation is a major problem in industrial systems. The accurate estimation of states leads to effective monitoring, fault diagnosis and good performance. For example, a supply chain is a large inter-connected system of suppliers, manufacturers, distributors and the end customers. This complex system is highly prone to communication disparities due to the generation of big data that resides in between the entities of the system. Kalman Filters work as an efficient state estimators to optimize this supply chain network. The article [33] proposes a rigorous approach using an extended Kalman filter with a network approach that models the supply chain as an abstraction. This approach is termed Augmented Trans-Nets. It allows multiple participants in a supply chain to be modeled without making the filter overly complex. Then states of the supply chain can be observed and estimated for different considerations.

In a manufacturing process Kalman Filter helps drastically improve the machine control that improves the capability of the overall process. For example in [34], a welding technique referred to as the Friction Stir Welding, improves the capability of joining hard materials such as 2000 and 7000 series aluminum alloys. They apply Kalman Filter to control the spindle speed and traversing speed optimizing the plunged depth, improving the welding process.

A quality control chart for monitoring a short run process during the start-up phase is presented in [35]. The chart uses a recursive Kalman Filter to stable a process where the process variance is unknown prior to the start of the production run. It is shown that the run length properties are independent of unknown process variance and these properties are appropriate for monitoring a stable process during start-up phase. An economic model for the optimal design of the control scheme is presented and illustrated with a wet etching process used in semiconductor manufacturing. Another example is the use of Unscented Kalman Filter to estimate the parameters of a train that is coasting on a flat track. One such important parameter is the resistance coefficient of the train, this is explored in [36]. The resistance parameter empowers engineers to optimize and improve the fuel efficiency of the train.

Chapter 2

Theoretical framework

This chapter covers the underlying concepts that are essential to understanding the Kalman Filter's structure and its applicability to solving practical problems. We begin by developing an understanding of linear dynamical systems in the state space commonly discussed in control systems theory [37]. We then explore the notion of linear and nonlinear observability that defines the applicability of the Kalman Filter to a specific linear or nonlinear problem respectively. We then define the Kalman Filter equations and describe the algorithm. We also discuss the tuning of the filter and higher ordered derivatives of the filter. Last, we share the Kalman Filters for nonlinear systems, i.e. the *Extended Kalman Filter* and the *Unscented Kalman Filter*. All versions of the filter are simulated or applied to real problems in subsequent chapters.

2.1 State space representation of dynamical systems

A system in control theory is a mathematical model that forms a relationship between the inputs and outputs based upon the differential equations that govern the evolution of the system in time. If the output of a system depends just on the current input it is referred to as a static system. For example,

$$y_t = 9x_t. \tag{2.1}$$

However, if the output of the system is dependent on current as well as values from the previous iteration, it is called a dynamical system. For example,

$$y_t = 9x_{t-1}. \tag{2.2}$$

Here, the output is dependent upon the input from the previous time step.

Dynamical systems are physical phenomena that can be modeled through differential equations. The mathematical description of such systems help in projecting the system to any past or future time step. These differential equations incorporate the inputs and system dynamics and yielding the outputs of the system. As a result , the temporal of the system can be described. The system is "represented" by a differential equation.

However, as the system's complexity grows, this representation becomes cumbersome. This is truer for the systems that have multiple inputs and outputs. So, the concept of *state space* is introduced which considerably simplifies this problem. The state space representation of a system replaces an $n'th$ order differential equation with a single first order differential equation. The gist of the state space is captured by the following two equations:

$$\dot{\mathrm{x}}(t) = \mathrm{A}\mathrm{x}(t) + \mathrm{B}\mathrm{u}(t) + \mathrm{w}(t) \tag{2.3}$$

$$\mathrm{y}(t) = \mathrm{C}\mathrm{x}(t) + \mathrm{D}\mathrm{u}(t) + \mathrm{v}(t) \tag{2.4}$$

The first Equation, (2.3) is called the state equation and the second Equation, (2.4) is called the output equation. The dot atop a variable represents time derivative. The variables used in these equations are now discussed.

2.1.1 State vector x

The variable x represents a column matrix that is a function of time and is called the state vector. It contains the variables of our dynamical system. For example if we are considering the position and velocity of a system under consideration then x state vector will be,

$$\mathrm{x} = \begin{bmatrix} \mathrm{x}_1 & \mathrm{x}_2 \end{bmatrix}^T \tag{2.5}$$

where, x_1 and x_2 represents position and velocity respectively. For an n th dimension system the size of the state vector x is $n \times 1$.

2.1.2 System dynamic matrix A

In Equation (2.3), the A is an $(n \times n)$ system matrix that consists of constants or parameters that govern the dyanmics of the state vector. This system matrix is in fact the process model of our system in continuous form. It represents the variation of the state vectors continuously with time.

$$\frac{d}{dt}\mathrm{x}(t) = \mathrm{A}\mathrm{x}(t). \tag{2.6}$$

For a physical system described by some mathematical equation, the system dynamics matrix A consists of the constant coefficients of the system equations.

In the next section, we derive a *state transition matrix* F which is the discretized version of the state dynamic matrix and it propagates the state vector matrix x through the time step Δt.

2.1.3 Control input u and control matrix B

For a system that is driven by some external force, Equation (2.3) also includes a B matrix. For an n dimensional system with r inputs, B is an $(n \times r)$ matrix and contains generally constant parameters. It determines how the inputs or controls variables in the u vector affect the state vector x.

2.1.4 Output equation

The output Equation (2.4) includes a column matrix y which is the output vector and is a function of time. The output matrix C contains the constant parameters which relates the state variables included in the state vector x to the output y.

The matrix D is the direct transition (or feedthrough) matrix, generally, a constant which relates the control input U to the output Y.

2.1.5 System and measurement noise

In both state space equations noise is an integral part. In Equation (2.3), W represents system noise, usually considered to be zero mean Gaussian white noise. In Equation (2.4), V represents measurement noise, also considered to be zero mean Gaussian white noise.

2.1.6 State transition matrix

The matrix A in Equation (2.3) is in continuous form and is referred to as the *system dynamics matrix*. It models a set of differential equations that govern the dynamics of the state variable X. However, we are not interested in the derivative of the variables but the transition of the variables from a previous time step $(t-\Delta t)$ to the current time step (t). For notational convenience we represent the previous time step with $k-1$ and the current time with k.

$$\mathrm{X}_k \;=\; \mathrm{F}_k\mathrm{X}_{k-1} + \mathrm{B}_k\mathrm{U}_k. \tag{2.7}$$

Here, F_k is referred to as the *state transition matrix* that transits the system from time $(t-\Delta t)$ in time step Δt. The matrix B, as defined before, is an $(n \times r)$ matrix that contains constant parameters and determines how the controls affect the state and U is the input control which is a function of time. Thus, the *state transition matrix* is a discretized matrix. It is found through the exponential of A.

2.1.6.1 Matrix exponential

The matrix exponential is a function on a square matrix analogous to the ordinary exponential function. We can derive the state transition matrix F by taking the exponential of the *state dynamic matrix* A which is a square matrix.

To understand this better let's take an example of a differential equation with a constant parameter k,

$$\frac{dx}{dt} = kx. \tag{2.8}$$

The Equation is solved as follows,

$$\begin{aligned} \frac{dx}{x} &= kdt \\ \int \frac{1}{x} dx &= \int k\ dt \\ logx &= kt + c \\ x &= e^{kt+c} \\ x &= e^c e^{kt} \\ x &= c_0 e^{kt} \end{aligned} \tag{2.9}$$

Now we use this methodology to solve Equation (2.3). We have,

$$\dot{\mathrm{x}}(t) = \mathrm{A}\mathrm{x}(t) \tag{2.10}$$

here, matrix A is constant and the overall equation is in differential form. We are interested in finding the matrix F derived from A which is a transition matrix containing the time step that transits our system between discrete time steps of $(t - \Delta t)$ and t. By substituting $\mathrm{F} = e^{\mathrm{A}t}$ we can derive F matrix from Equation (2.10). Following is an example showing how we can calculate F.

$$\begin{aligned} \frac{(\mathrm{x}_t + \Delta t) - \mathrm{x}_t}{\Delta t} &\approx \mathrm{A}\mathrm{x}_t \\ \mathrm{x}_t + \Delta t &\approx \mathrm{x}_t + \mathrm{A}\Delta t\mathrm{x}_t \\ &\approx (1 + \mathrm{A}\Delta t)\mathrm{x}_t \\ \mathrm{x}_t + \Delta t &\approx (1 + \mathrm{A}\Delta t)\mathrm{x}_t, \end{aligned} \tag{2.11}$$

where, $1 + \mathrm{A}\Delta t$ is just the lowest order term in

$$\mathrm{F} = e^{\mathrm{A}\Delta t} \tag{2.12}$$

So,

$$\begin{aligned} \mathrm{x}_{t+\Delta t} &= e^{\mathrm{A}\Delta t} \\ &= \mathrm{F}\mathrm{x}_t \end{aligned} \tag{2.13}$$

Using k as a notation to represent the time steps,

$$\mathrm{x}_{t+1} = \mathrm{F}\mathrm{x}_k \tag{2.14}$$

From the first order Equation (2.8) the linear matrix form can be written as,

Let $\mathrm{x}_1 = x$ and $\mathrm{x}_2 = v$,

$$\mathrm{x} = \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix} \tag{2.15}$$

$$\begin{bmatrix} \dot{\mathrm{x}}_1 \\ \dot{\mathrm{x}}_2 \end{bmatrix} = \begin{bmatrix} 0 & k \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix}, \tag{2.16}$$

where, $\mathrm{A} = \begin{bmatrix} 0 & k \\ 0 & 0 \end{bmatrix}$. The F matrix is the following expansion,

$$\mathrm{F} = e^{\mathrm{A}\Delta t} = \mathrm{I} + \mathrm{A}\Delta t + \frac{(\mathrm{A}\Delta t)^2}{2!} + \frac{(\mathrm{A}\Delta t)^3}{3!} + \cdots \tag{2.17}$$

Keeping only the lowest order terms, $\mathrm{F} \approx 1 + \mathrm{A}\Delta t$ Equation (2.12) becomes,

$$\begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix}_k = \begin{bmatrix} 1 & k\Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix}_{k-1} \tag{2.18}$$

Showing that the state transition matrix is $\begin{bmatrix} 1 & k\Delta t \\ 0 & 1 \end{bmatrix}$. Hence, we find the *state transition matrix* matrix F which is the discretized upgrade of the *state dynamic matrix* A and serves as the transition function through a time step Δt to project the state variables x from previous time step $k-1$ to current time step k.

We will be using the *Matrix Exponential* technique throughout this thesis to derive the *state transition matrix* for all of our linear and nonlinear dynamical problems. Our system equations will usually be of higher ordered, largely second ordered, and we will reduce them to first ordered equations represented by the matrix A. Then, their transition matrix F will be derived using the *Matrix Exponential* technique. This matrix will serve as the *process model* in Equations of the Kalman Filter and will project the state variables of our dynamical problems as well as the covariance matrix P from previous time step to the current time step.

2.1.7 Observability

The concept of observability is distinctively related to state space methods. It was introduced in the mid $1950's$ by Kalman himself as a way of exploring whether the internal states of a given dynamical system can be determined if the output parameters of the system are measured. Before actually applying Kalman Filter to our linear and nonlinear problems we would like to highlight the concept of observability in both cases as it would help us pre-determine the possibility of observing the states and parameters of our system based upon the knowledge of only the available outputs.

2.1.7.1 Observability of linear systems

A linear system is represented by a mathematical model that uses a *linear operator* to map the inputs of the system as a function of time t to the outputs of the system in such a way that the outputs are directly proportional to the inputs. Observability captures the ability to estimate the states of the system based upon the knowledge of only the outputs.

For example, consider a mass attached to a spring. The system is two-dimensional with position and velocity as the state variables. For this system, would it be

possible to determine position and velocity of the system if we only have a single sensor to measure position, or would it be possible to determine both position and velocity if we only have a sensor to measure velocity, or would it be possible to determine both if we only have an accelerometer etc. In all of these cases, it is the rank of the observability matrix that helps finding whether the states of the dynamical system can be observed or not given certain measurements.

Mathematically observability is determined by examining the observability matrix $\mathcal{O}$ which is defined as:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \\ \mathrm{CA}^2 \\ \vdots \\ \mathrm{CA}^{n-1} \end{bmatrix}, \tag{2.19}$$

where C is the output matrix, A is the system dynamic matrix from Equation (2.3) and n is the number of dimensions of the problem at hand.

The interpretation of this matrix is that only if the rank of the observability matrix $\mathcal{O}$ is equal to the order of the system dynamic matrix A, it is observable. i.e. $rank(\mathcal{O}) = n$. If the order is less than the dimensions of the system then further analysis to extract subsystems is carried out through the *Kalman Decomposition* [38] to identify the reduced subsystem of the state that is observable.

For example, consider a dynamical system represented by the following state space equations:

$$\begin{bmatrix} \dot{\mathrm{x}}_1 \\ \dot{\mathrm{x}}_2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix} \tag{2.20}$$

$$\mathrm{y}_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix}. \tag{2.21}$$

Where the system dynamic matrix A is $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ and the output matrix C is $\begin{bmatrix} 1 & 0 \end{bmatrix}$.

The matrix C tells us that we have only one measurement i.e. x_1.

The observability matrix for this second-order differential equation is given by:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix} \tag{2.22}$$

Since the rows of the matrix $\mathcal{O}$ are linearly independent, $rank(\mathcal{O}) = 2 = n$.

This means that the system is observable i.e. we can estimate both x_1 and x_2 using one sensor. We can further exploit this technique by considering a second sensor as well and retest for the observability.

However, if rank of the observability test matrix is not equal to the dimensions of the system, then the system is said to be unobservable. For example, consider another dynamical system that is represented by the following state space Equation. The system also has an input represented by u.

$$\begin{bmatrix} \dot{\mathrm{x}}_1 \\ \dot{\mathrm{x}}_2 \end{bmatrix} = \begin{bmatrix} -2 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \mathrm{u} \tag{2.23}$$

$$\mathrm{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix}. \tag{2.24}$$

Where the system dynamic matrix A is $\begin{bmatrix} -2 & 0 \\ 0 & -1 \end{bmatrix}$ and the output matrix C is $\begin{bmatrix} 1 & 0 \end{bmatrix}$.

The matrix C tells us that we have only one measurement i.e. x_1.

The observability matrix for this system is given by:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -2 & 0 \end{bmatrix} \tag{2.25}$$

Since the rows of the matrix $\mathcal{O}$ are not linearly independent and the $rank(\mathcal{O}) = 1 \neq n$, the system is unobservable.

2.1.7.2 Observability of nonlinear systems

In linear regime, the observability test effectively evaluates the possibility of estimating the states using the ouputs. However, in the case of nonlinearity, for example augmentation of the state variables with unknown sub-parameters, the linear observability test is prone to fail. So, research was carried out in early $1970's$ to find the observability of nonlinear systems [39]. It was shown that the observability matrix for a nonlinear system could be expressed using the high ordered Lie derivatives of the measurement function. This is now discussed.

Suppose we have a nonlinear system defined by the potentially nonlinear function,

$$\dot{\mathrm{X}} = f(t, \mathrm{X}) \tag{2.26}$$

and the measurement equation

$$\mathrm{Y} = h(t, \mathrm{X}). \tag{2.27}$$

The multiple high ordered Lie derivatives of the measurement equation are given by,

$$\begin{aligned}
\pounds_f^0(y(\mathrm{X})) &= y(\mathrm{X}) \\
\pounds_f^1(y(\mathrm{X})) &= \frac{\partial y(\mathrm{X})}{\partial \mathrm{X}} \cdot f(\mathrm{X}) \\
\pounds_f^2(y(\mathrm{X})) &= \frac{\partial}{\partial \mathrm{X}}\big[\pounds_f^1(y(\mathrm{X}))\big] \cdot f(\mathrm{X}) \\
&\vdots \\
\pounds_f^n(y(\mathrm{X})) &= \frac{\partial}{\partial \mathrm{X}}\big[\pounds_f^{n-1}(y(\mathrm{X}))\big] \cdot f(\mathrm{X}),
\end{aligned} \tag{2.28}$$

where n is the dimensionality of the system and $\pounds_f(y(\mathrm{X}))$ is the Lie Derivative of $y(\mathrm{X})$ along the vector field $f(\mathrm{X})$. These terms are then composed to form the mapping matrix ϕ,

$$\phi = \begin{bmatrix} \pounds_f^0(y(\mathrm{x})) \\ \pounds_f^1(y(\mathrm{x})) \\ \pounds_f^2(y(\mathrm{x})) \\ \vdots \\ \pounds_f^{n-1}(y(\mathrm{x})) \end{bmatrix}. \tag{2.29}$$

Taking the Jacobian of the matrix ϕ with respect to the state variables finally results in the observability test matrix for our nonlinear system.

$$\mathcal{O} = \frac{\partial \phi}{\partial \mathrm{x}} = \begin{bmatrix} \frac{\partial \pounds_f^0(y(\mathrm{x}))}{\partial \mathrm{x}_n} & \cdots & \frac{\partial \pounds_f^0(y(\mathrm{x}))}{\partial \mathrm{x}_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial \pounds_f^{n-1}(y(\mathrm{x}))}{\partial \mathrm{x}_1} & \cdots & \frac{\partial \pounds_f^{n-1}(y(\mathrm{x}))}{\partial \mathrm{x}_n} \end{bmatrix}. \tag{2.30}$$

If the rank of the observability test matrix (2.30) is equal to the dimensions of the state, then this nonlinear system is observable.

For example, consider a nonlinear system with the following system of reduced nonlinear equations,

$$\dot{\mathrm{x}}_1 = \frac{1}{2}\mathrm{x}_1^2 + e^{\mathrm{x}_2} + \mathrm{x}_2 \tag{2.31}$$

$$\dot{\mathrm{x}}_2 = \mathrm{x}_1^2, \tag{2.32}$$

with measurement equation,

$$\mathrm{y} = \mathrm{x}_1. \tag{2.33}$$

The Lie derivative of the measurement function y is given by,

$$\begin{aligned} \pounds_f^0(y(\mathrm{x})) &= y(\mathrm{x}) = \mathrm{x}_1 && (2.34) \\ \pounds_f^1(y(\mathrm{x})) &= \frac{\partial y}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\ &= \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2}\mathrm{x}_1^2 + e^{\mathrm{x}_2} + \mathrm{x}_2 \\ \mathrm{x}_2 \end{pmatrix} \\ &= \frac{1}{2}\mathrm{x}_1^2 + e^{\mathrm{x}_2} + \mathrm{x}_2 && (2.35) \end{aligned}$$

the observability mapping matrix ϕ here is given by,

$$\phi = \begin{bmatrix} \mathrm{x}_1 \\ \frac{1}{2}\mathrm{x}_1^2 + e^{\mathrm{x}_2} + \mathrm{x}_2 \end{bmatrix}, \tag{2.36}$$

the partial derivative of the mapping matrix $\mathrm{J}(\phi)$ results in the observability test matrix,

$$\mathcal{O} = \begin{bmatrix} 1 & 0 \\ \mathrm{x}_1 & e^{\mathrm{x}_2+1} \end{bmatrix} \tag{2.37}$$

The rank of this matrix is 2 which means that it is equal to the dimensions of the system so, this system is observable from the x_1 at the output.

2.1.8 Controllability

Controllability is a similar property that was also posited by Kalman as a means to identify the states of the system that are controllable using the inputs. This allows us to determine, for example, whether we have the opportunity to control position and velocity using acceleration as an input to the system. The controllability is an extremely important metric for control engineers as it helps them design systems with definite forms of controllable and uncontrollable states and parameters.

Mathematically, the controllability is verified by examining the controllability matrix $\mathcal{C}$.

$$\mathcal{C} = \begin{bmatrix} \mathrm{B} & \mathrm{AB} & \mathrm{A}^2\mathrm{B} & \cdots & \mathrm{A}^{n-1}\mathrm{B} \end{bmatrix} \tag{2.38}$$

If the rank of the controllability matrix is equal to the order of the system dynamic matrix A, it is controllable $rank(\mathcal{C}) = n$, if not then the system is uncontrollable.

Let us take an example of the following dynamical system represented by the state space equations:

$$\begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 5 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 2 \end{bmatrix} \mathrm{u} \tag{2.39}$$

$$\mathrm{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix}, \tag{2.40}$$

where the system dynamic matrix A is $\begin{bmatrix} 1 & 2 \\ 5 & 0 \end{bmatrix}$ and the control input matrix B is $\begin{bmatrix} 0 \\ 2 \end{bmatrix}$. The output matrix C is $\begin{bmatrix} 1 & 0 \end{bmatrix}$.

The controllability matrix for this state system is given by:

$$\mathcal{C} = \begin{bmatrix} \mathrm{B} & \mathrm{AB} \end{bmatrix} = \begin{bmatrix} 0 & 4 \\ 2 & 0 \end{bmatrix} \tag{2.41}$$

Since the rows of the matrix $\mathcal{C}$ are linearly independent, the $rank(\mathcal{C}) = 2 = n$. This means that the system is controllable.

2.2 The Kalman Filter

The Kalman Filter is a *recursive* algorithm that moves between a *prediction* step and an *estimation*. These steps generate states and associated errors of the system that are philosophically also referred to as *a priori* and *a posteriori*, which are Latin phrases, meaning, "from the earlier" and "from the later" respectively. These terms were popularized by Immanuel Kant's *Critique of Pure Reason*, one of the most influential works in the history of philosophy [40]. In much simpler terms, the Kalman Filter has the ability to predict the state of a system using a mathematical model of the dynamical system and then correct its prediction using real time information from a transducer that may contain varying statistical noise.

The simplest form of a Kalman Filter is the *Scalar Kalman Filter (SKF)* which consists of only one state variable. As the dimensions of the system grow the

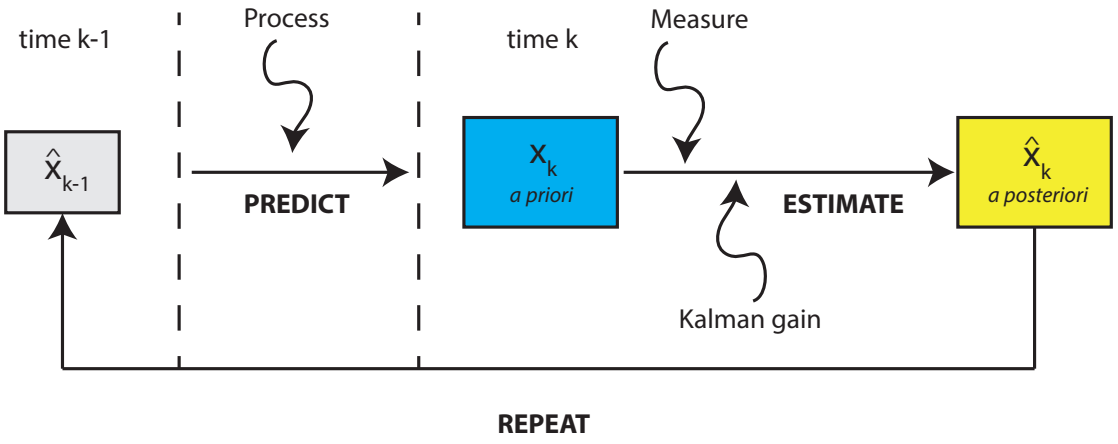

FIGURE 2.1: The block diagram of Kalman Filter

size of the vector space grows. This growth is augmented using matrices and the resulting filter is a *Multivariate Kalman Filter (MKF)*. These are used for linear and nonlinear problems that have multiple inputs and outputs. In this thesis, we have largely worked with multivariate linear and nonlinear problems and so, we only discuss the mathematical models of the multivariate Kalman filter's.

The crust of the Kalman Filter's function is packed in the following equation.

$$\hat{x}_k = x_k + K(z_k - Hx_k), \tag{2.42}$$

here, x_k is the current *a priori* prediction of the state, z_k is the current noisy measurement value,and $\hat{x}_k$ is the current *a posteriori* estimation of the state by the filter using the weight which is called the Kalman Gain K. This equation briefly describes the internal working of the Kalman Filter. The most important part of this equation is the parameter *Kalman Gain* at the current time step. We will discuss it in detail in the subsequent sections. The goal here is to find the $\hat{x}_k$ for each consequent time step k such that the *residual* error between the last and the current estimation $\mathbb{E}[(\hat{x}_k - \hat{x}_{k-1})^2]$ is minimized or the sum of the diagonal elements of the error covariance matrix P is minimized. This recursive process is the hallmark of Kalman Filter's success as it eliminates the filter's need to rely on all historical data.

2.2.1 The Kalman Filter Algorithm

The Kalman filter estimates a state value through a process using a feedback control in the form of noisy measurements. The filter can also be referred to as *predictor-corrector* algorithm. We highlight the flow of the algorithm in Figure 2.1 and describe it below.

2.2.1.1 Predicting the state

Consider the equation,

$$\mathrm{x}_k \;=\; \mathrm{F}\hat{\mathrm{x}}_{k-1}. \tag{2.43}$$

This is the *Time Update* equation or the *Prediction* equation. This equation predicts the current states from the previous estimates. Here, F is the state transition matrix, x_k is the prediction at current time step and $\hat{\mathrm{x}}_{k-1}$ is the state estimate from the previous time step.

Similarly, the uncertainty associated with each state variable embedded in the error covariance matrix P is also updated as given below,

$$\mathrm{P}_k \;=\; \mathrm{F}\hat{\mathrm{P}}_{k-1}\mathrm{F}^T + \mathrm{Q}, \tag{2.44}$$

where, P_k is the *predicted* error covariance matrix at current time step, $\hat{\mathrm{P}}_{k-1}$ is the estimated state covariance matrix from the previous and Q is the process noise covariance matrix.

This prediction is entirely based upon our belief modeled as a Gaussian distribution. The process noise Q is added to the noise covariance P such that it accounts for all the unrelated noise in the process model over time. This is the most important part of the filter as it also serves as a tuning parameter for the filter's performance. We will have more to say about these covariance matrices in the subsequent sections.

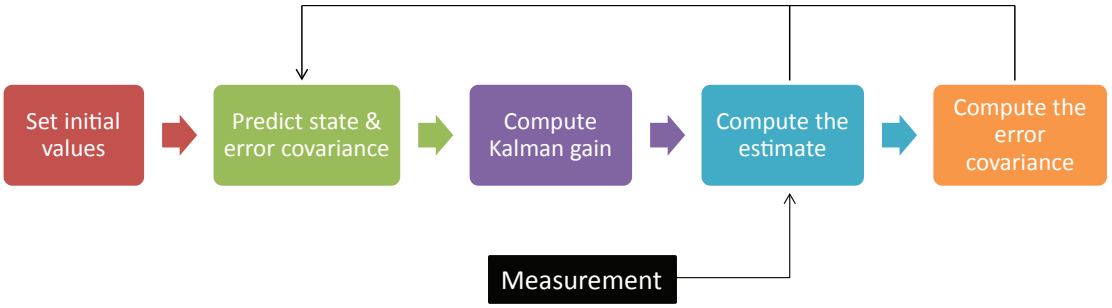

FIGURE 2.2: *An alternative flow chart of the Kalman Filter algorithm.*

2.2.1.2 Correcting the prediction

The *measurement update* equations, also referred to as the *correction* equations, act as a feedback channel to the filter's original prediction returned from the *time update* step. Following set of equations perform the correction step for the filter in every application.

$$\mathrm{Y}_k = \mathrm{Z}_k - \mathrm{H}\mathrm{X}_k \tag{2.45}$$

$$\mathrm{K}_k = \mathrm{P}_k\mathrm{H}^T(\mathrm{H}\mathrm{P}_k\mathrm{H}^T + \mathrm{R})^{-1} \tag{2.46}$$

$$\hat{\mathrm{X}}_k = \mathrm{X}_k + \mathrm{K}_k\mathrm{Y}_k \tag{2.47}$$

$$\hat{\mathrm{P}}_k = (\mathrm{I} - \mathrm{K}_k\mathrm{H})\mathrm{P}_k \tag{2.48}$$

We now describe the meaning of these equations. An alternative flow chart of this entire process is also exhibited in Figure 2.2.

The *measurement update* step begins by calculating the difference $(z_k - \mathrm{X}_k)$, also referred to as the *innovation* or *residual* between the measurement values and the predicted state of the system. Next, the gain factor referred to as the *Kalman Gain* using Equation (2.46) is computed. It acts as a weighing factor to *correct* the state. This implies that if the *a priori* error covariance P_k is smaller than the measurement noise R then the *Kalman Gain* would be smaller and the filter would lean towards the values of the measurements, see Equation (2.47). Conversely, if the measurement noise is smaller then the filter would lean towards the predicted values. Last, the filter also updates the error covariance associated with the state variables using Equation (2.48). As the residual gets minimized so is the belief on the filter's prediction improves.

This recursive process is the hallmark of the Kalman filter and it makes it a highly practical algorithm to apply in comparison to Weiner filters which apply to the holistic collection of measurement data to generate each prediction and estimate [41].

2.2.2 Process noise covariance matrix

The design of the process noise covariance matrix Q is considered to be the most difficult and interesting part of designing the Kalman Filter. This is because in realistic terms the process model is prone to be incomplete. Not everything from the real-world is entirely packed in the mathematical model. So, the process noise is designed to encompass those disturbances and compensate for the uncertainty in one knowledge of the state. But even if we have a perfect Q matrix i.e. a process noise with zero variance, the filter might completely ignore the valuable influence of the noisy measurement data.

Following are two common methodologies to compute the process noise covriance matrix for use in the filter's computational algorithm.

2.2.2.1 Continuous white noise model

If we suppose a two dimensional system of position and velocity, represented by the state vector $\mathrm{x} = \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix} = \begin{bmatrix} \mathrm{x} \\ \dot{\mathrm{x}} \end{bmatrix}$, with state transition matrix F,

$$\mathrm{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}. \tag{2.49}$$

We may like to assume the highest ordered term i.e. velocity is constant during each time step Δt. If this is the case, it means that the disturbance in the position of the system on average is zero. To find the cumulative effect in every time step we take an integral of the noise factor Q_c over the complete time update step.

$$\mathrm{Q} = \int_0^{\Delta t} \mathrm{F}_t \mathrm{Q}_c \mathrm{F}_t^T \, dt. \tag{2.50}$$

Where, Q_c is the continuous noise being added to our process model F over the discrete time interval $[0, \Delta t]$.

The continuous noise term Q_c for zero error in position is,

$$\mathrm{Q}_c = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \Phi_s \tag{2.51}$$

where, Φ_s is the spectral density of the white noise. This spectral density effectively increases or decreases the noise in the high ordered term in our process model and can be used as a tuning parameter to improve the overall filter's performance. The designer has to choose ϕ_s and Q_c to optimize the filter. Trial and error is a convenient starting point.

2.2.2.2 Piecewise white noise model

It is also possible that noise is assumed to be constant, not varying during the time step Δt and only differs across the various time steps. In this case, the prediction equation can be written as,

$$\mathrm{x}_k = \mathrm{F}\mathrm{x}_{k-1} + \Gamma \mathrm{w}_k \tag{2.52}$$

where Γ is the *noise gain* in the time period and w_k is the constant piecewise noise within the time step. Thus, the change in velocity of the system in one time period will be $\mathrm{w}_k \Delta t$, and the change in position will be $\mathrm{w}_k \Delta t^2/2$. This results in the following matrix form for the noise gain,

$$\Gamma = \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} \tag{2.53}$$

The resulting piecewise error covariance matrix Q is subsequently derived as,

$$\begin{aligned} \mathrm{Q} &= \mathbb{E}[\Gamma \mathrm{w}(t)\mathrm{w}(t)\Gamma^T] \\ &= \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 \end{bmatrix} \end{aligned} \tag{2.54}$$

This can also be referred to as $\Gamma\sigma_v^2\Gamma^T$, where, σ_v^2 is a value that increases or decreases the variance of the error between the two variables of the system. It acts as a tuning parameter of which we will observe the effects on our performance of the Kalman Filter in the simulated applications in the subsequent chapter, whose selection is the designer's purview.

2.2.3 Measurement noise

A real life phenomena is measured using a sensor. A sensor is an electronic transducer that converts a mechanical signal into an electrical signal that can be read by a digital device. The sensors embody sensitivity and uncertainty in their measured quantities. How much a particular sensor is infested with noise is usually supplied by the sensor's manufacturer in the provided specification sheet. This noise is mathematically represented in the form of an error or uncertainty and is referred to as the *Measurement Noise* R in the Kalman Filter algorithm. The primary role of the measurement noise is played in the computation of the *Kalman Gain* in the *Correction* step of the filter's algorithm as shown in Equation (2.46). If the noise from the transducer is larger than the uncertainty in the state variables then the Kalman filter gives more weightage to the system's process for the *a priori* prediction of the state x, whereas, if the sensor noise is less then the filter gives more weightage to the measurements for the *a posteriori* estimation of the state variable $\hat{\mathrm{x}}_k$.

2.2.4 Parameter estimation

The *system dynamical matrix* A in the state-space Equation (2.3) describes the process model of the system under observation. This process model may or may not be entirely known. We may also be ignorant or only partially cognizant of the system parameters themselves. For example, we may know the position and velocity of a mass oscillator, we may not even know the spring constant or damping coefficient on which the evolution of the states depend. Thus, an important application of the Kalman Filter lies in the estimation of system parameters themselves.

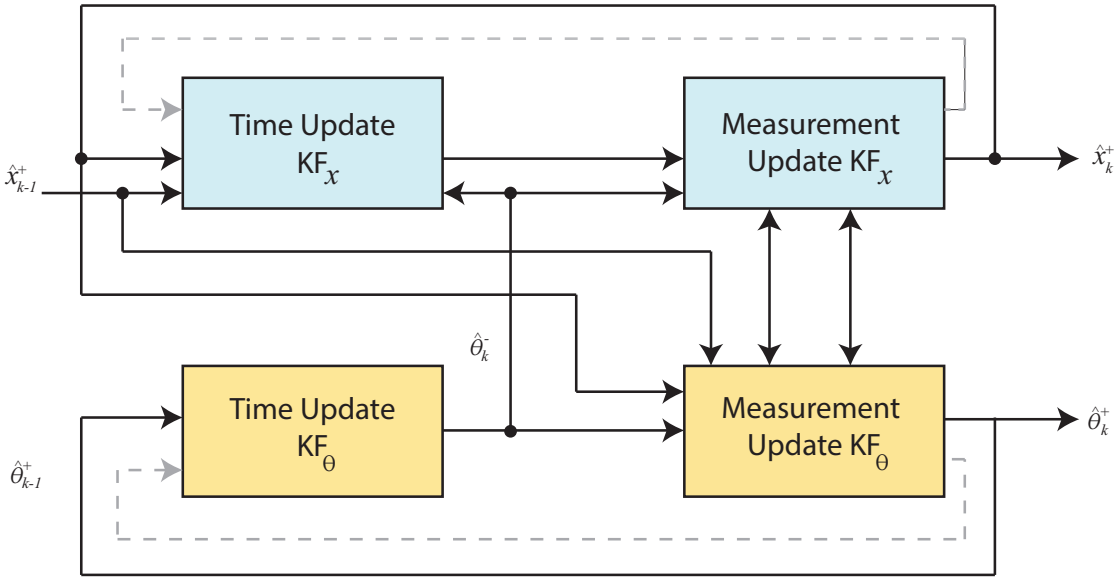

FIGURE 2.3: *The Dual State and Parameter Estimation block diagram with parallel application of Kalman Filters.*

In literature, parameter estimation using Kalman Filters is divided into two types. One is referred to as the *Dual State Parameter Estimation (DSPE)* and the second is referred to as the *Joint State Parameter Estimation (JSPE)*.

2.2.4.1 Dual State Parameter Estimation

In order to estimate the parameters of a system we can use the *Dual State Parameter Estimation (DSPE)* technique that uses two Kalman Filters in parallel to estimate the values of the parameteres [42]. The advantage of using dual filters for this problem is the reduced computational complexity as the number of matrix operations is conditioned to be happening in the two filter blocks separately. Figure 2.3 illustrates the block diagram of the dual state and parameter estimation working. Both the filters work in parallel and exchange information between themselves.

2.2.4.2 Joint State Parameter Estimation

Another technique to estimate the parameters of a system along with its states is the joint state and parameter estimation. In this technique the states and parameters are augmented into the single state space vector, expanding the dimensionality of the state space. This joint state space model of the system with

unknown parameters θ in discrete state space form can be stated as:

$$\begin{aligned} \mathrm{x}_k &= \mathrm{F}(\theta)\mathrm{x}_{k-1} + \mathrm{B}(\theta)\mathrm{u}_{k-1} + \mathrm{w}_k \\ \mathrm{y}_k &= \mathrm{H}(\theta)\mathrm{x}_k + \mathrm{v}_k, \end{aligned} \tag{2.55}$$

where θ is a vector of unknown parameters to be estimated and is augmented to include an estimate of the unknown parameters. The overall state space model for the joint state and parameter estimation Kalman Filter problem is as follows:

$$\begin{bmatrix} \mathrm{x}_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} \mathrm{F}(\theta_k)\mathrm{x}_{k-1} + \mathrm{B}(\theta_k)\mathrm{u}_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} \mathrm{w}_k \\ \mathrm{v}_k \end{bmatrix} \tag{2.56}$$

where $\mathbf{v}_k$ is a white Gaussian noise of appropriate strength allowing the exploration of the parameter space [43].

This augmented state space may in most cases become non-linear due to the existence of unknown terms in the state dynamic matrix. In order to solve this nonlinear joint state and parameter estimation problem the *Extended Kalman Filter (EKF)* or the *Unscented Kalman Filter (UKF)* can be used. A non-linear parameter augmented model can be seen in Equation (2.56). The next instance of states are dependent on a product of the states in the linear state space system and the estimated parameter, which is also a state in the augmented system. We will demonstrate examples of the joint state parameter estimation in the next chapters.

2.3 Extended Kalman Filter

The simplest example of handling non-linearities is by linearizing the system around a certain point. This technique is used in the *Extended Kalman Filter (EKF)* which is known to be sub-optimal, but has stood the test of time as a well proven method in practice.

A general non-linear system can be formulated as:

$$\begin{aligned} \mathrm{x}_k &= f(\mathrm{x}_{k-1}, \mathrm{u}_{k-1}, \mathrm{w}_{k-1}) \\ \mathrm{y}_k &= h(\mathrm{x}_k, \mathrm{v}_k), \end{aligned} \tag{2.57}$$

where x_k is a vector of state variables, u is a vector of inputs, w is the added noise, y is the measurements vector and v is the noise in the measurements space.

In Extended Kalman Filter the linearized system matrices are found by calculating the partial derivative (Jacobian) of the nonlinear funcion f with respect to the state variables x or the process noise w (both computed at the time step k).

$$\Phi_k = \left.\frac{\partial f(\mathrm{x},\mathrm{u},\mathrm{w})}{\partial \mathrm{x}}\right|_k \tag{2.58}$$

$$\Gamma_k = \left.\frac{\partial f(\mathrm{x},\mathrm{u},\mathrm{w})}{\partial \mathrm{w}}\right|_k . \tag{2.59}$$

The partial derivatives result in the formulation of Φ_k, which is the linearized state dynamic matrix, and Γ_k which is the linearized process noise matrix. The prediction step is performed using Equation (2.57), using the actual nonlinear model function f,

$$\begin{aligned} \mathrm{x}_k &= f(\hat{\mathrm{x}}_{k-1}, \mathrm{u}_k, \mathrm{w}) \\ \mathrm{P}_k &= \Phi_k \hat{\mathrm{P}}_{k-1} \Phi_k^T + \Gamma_k \mathrm{Q}_k \Gamma_k^T \end{aligned} \tag{2.60}$$

Notice that the state estimate is made on the non-linear propagation function f, while the covariance matrix estimate is made on the basis of linearized system matrix Φ_k. The rest of the partial derivatives are found based on the a priori x_k.

$$\Sigma_k = \left.\frac{\partial h(\mathrm{x},\mathrm{v})}{\partial \mathrm{x}}\right|_{\mathrm{x}_k} \tag{2.61}$$

$$\Lambda_k = \left.\frac{\partial h(\mathrm{x},\mathrm{v})}{\partial \mathrm{w}}\right|_{\mathrm{w}_k} \tag{2.62}$$

where Σ_k is the linearized output matrix, Λ_k is the linearized measurement noise matrix.

If non-linearity does not exist in a system matrix the partial derivative equals the system matrix, for example: if there is no non-linear formulation in $h(\mathrm{x},\mathrm{v})$ then we simplify,

$$\Sigma_k = \frac{\partial h(\mathrm{x}, \mathrm{v})}{\partial \mathrm{x}} = \mathrm{H}_k \tag{2.63}$$

$$\Lambda_k = \frac{\partial h(\mathrm{x}, \mathrm{v})}{\partial \mathrm{w}} = \mathrm{R}_k \tag{2.64}$$

For a system with a nonlinear function and a nonlinear transfer function, the update step of the Kalman filtering algorithm is applied,

$$\mathrm{y}_k = \mathrm{z}_k - \Sigma_k \mathrm{x}_k \tag{2.65}$$

$$\mathrm{K}_k = \mathrm{P}_k \Sigma_k^T (\Sigma_k \mathrm{P}_k \Sigma_k^T + \Lambda_k)^{-1} \tag{2.66}$$

$$\hat{\mathrm{x}}_k = \mathrm{x}_k + \mathrm{K}_k \mathrm{y}_k \tag{2.67}$$

$$\hat{\mathrm{P}}_k = (\mathrm{I} - \mathrm{K}_k \Sigma_k) \mathrm{P}_k \tag{2.68}$$

It is important to note that while Kalman filtering is optimal in the case of a linear system, the EKF is sub-optimal in the case of a non-linear system. This can still give adequate performance in the case when the nonlinearities are small, but when the system is highly nonlinear the performance is degraded. Furthermore, since the partial derivatives are to be calculated for every time step, computational load is increased compared to the linear counterpart.

Other methods for nonlinear system exist such as the Unscented Kalman Filter (UKF). While the UKF has a computational burden similar to the EKF [44], our investigation of UKF for the experimental physics problems discussed in thesis has proven to be more robust, easy and accurate in comparison to the performance of EKF.

In general, to avoid EKF divergence, one should be careful about following:

1. Properly specify system dynamics and measurement equations, e.g. not to describe a pendulum by a first order model.

2. Ensure that your system is observable, e.g. construct the observability test matrix and check its rank.

3. Properly specify process and measurement noise covariance matrices, e.g. not to be too optimistic regarding the accuracy of your model and sensors.

4. If your system is nonlinear, seed the EKF with the initial state estimate which is likely to be close to the true value.

The EKF works on the principle that a linearized transformation of means and covariances is approximately equal to the true nonlinear transformation. However, this approximation could be unsatisfactory in practice.

2.4 Unscented Kalman Filter

When the complexity of the dynamical system grows, the *Extended Kalman Filter* tends to diverge and becomes harder to implement. Furthermore, the linearization step involves calculation of partial derivatives for every iteration step of the filter. This makes the Extended Kalman Filter a computationally expensive algorithm. With its practically proven performance, *Unscented Kalman Filter* is considered to be an efficient alternative.

2.4.1 The Unscented Transform

The base of the Unscented Kalman Filter is in a process known as *The Unscented Transform*. In this process, a set of points referred to as the sigma points are passed through the nonlinear function of the system to estimate the mean and covariance of the new state distribution.

$$y_\sigma = f(\chi). \tag{2.69}$$

Here, y_σ is a function of the sigma points matrix χ. This creates another matrix containing the transformed sigma points that have been passed through the nonlinear function. This approximates the resulting Gaussian state distribution of the estimated state. This overall process is referred to as the *Unscented Transform*. A graphical representation can be visualized in Figure 2.4.

Following are the steps to perform the Unscented Transformation:

1. Calculate the set of sigma points,

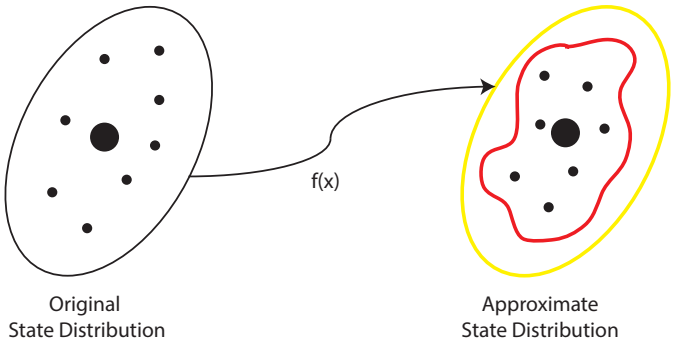

FIGURE 2.4: *The Gaussian approximation of the state distribution before and after Unscented Transform.*

2. Assign weights to each sigma point,
3. Transform these sigma points through the nonlinear function,
4. Compute the new state Gaussian distribution and associated weights,
5. Find mean and variance of the new Gaussian.

2.4.2 The sigma points

For the *Unscented Transform*, a set of scaled sample points referred to as the *Sigma Points* are calculated. These are a minimal set of carefully chosen points that approximately model the Gaussian distribution of the nonlinear system. When passed through the nonlinear function, they have the propensity to model the state and error propagation, in time, more accurately, than the approximation used in the Extended Kalman Filter.

The sigma points can be calculated using the *Van der Merwe Scaled Sigma Point Algorithm* [45]. Using just three parameters α, β and κ this algorithm computes a scaled weighted distributed set of sigma points that approximates the state distribution. Using this algorithm we generate a matrix of the sigma points χ with $2n+1$ columns, where n is the number of dimensions of the system. The first point χ_0 is the mean value of the input while the others are computed as below.

$$\chi_0 = \mu \tag{2.70}$$

$$\chi_i = \begin{cases} \mu + [\sqrt{(n+\lambda)\Sigma}]_i & for\ i = 1 \cdots n \\ \mu - [\sqrt{(n+\lambda)\Sigma}]_{i-n} & for\ i = (n+1) \cdots 2n \end{cases} \tag{2.71}$$

where, $\sqrt{(n+\lambda)\Sigma}$ is a matrix. The subscript i represents the column vector of this matrix. The $\lambda = \alpha^2(n+\kappa) - n$, α, and κ are the scaling parameters of the sigma points distribution and n is the dimensions of the state. Furthermore, we compute the square root of the covariance matrix Σ, scaled by factor γ, maintaining the symmetry by both adding and subtracting it from the mean.

This generates the following sigma points matrix.

$$\chi = \begin{bmatrix} \chi_{0,0} & \chi_{0,1} & \chi_{0,2} & \cdots & \chi_{0,2n+1} \\ \chi_{1,0} & \chi_{1,1} & \chi_{1,2} & \cdots & \chi_{1,2n+1} \\ \chi_{2,0} & \chi_{2,1} & \chi_{2,2} & \cdots & \chi_{2,2n+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \chi_{n-1,0} & \chi_{n-1,1} & \chi_{n-1,2} & \cdots & \chi_{n-1,2n+1} \end{bmatrix}, \tag{2.72}$$

where, n represents the dimensions of our system.

2.4.3 Computing the weights

The sigma points are also weighted using λ as a scaling factor. The associated weight for the first mean w_0^m value is calculated using the following Equation,

$$w_0^m = \frac{\lambda}{n+\lambda}, \tag{2.73}$$

where, n is the dimensions of the system and $\lambda = \alpha^2(n+\kappa) - n$.

The weights for the rest of the sigma points $\chi_1 \cdots \chi_{2n+1}$ are calculated using,

$$w_i^m = \frac{1}{2(n+\lambda)} \quad i = 1 \cdots 2n, \tag{2.74}$$

Next, we see how these sigma points and weights work together to predict and estimate the states of the nonlinear system in subsequent sections.

2.4.4 Prediction step

These transformed sigma points along with the weights are used to predict the *a priori* state and the Gaussian distribution of the error covariance using the following equations,

$$\mathrm{x}_k = \sum_{i=0}^{2n} w_i^m y_\sigma \tag{2.75}$$

$$\mathrm{P}_k = \sum_{i=0}^{2n} w_i^c (y_\sigma - \mathrm{x}_k)(y_\sigma - \mathrm{x}_k)^T + \mathrm{Q} \tag{2.76}$$

where, w^m are the weights associated with the mean values and w^c are the weights of the error covariance P.

2.4.5 Update step

We create a measurement function that converts the sigma points into the measurements space for the update step of the filter,

$$\mathbb{Z} = h(y_\sigma) \tag{2.77}$$

Again, using the *Unscented Transform* we pass the sigma points and their associated weights through the measurement function to compute the state and covariance in the measurement space as follows,

$$\mu_{\mathbb{Z}_k} = \sum_{i=0}^{2n} w_i^m \mathbb{Z} \tag{2.78}$$

$$\mathrm{P}_{\mathbb{Z}_k} = \sum_{i=0}^{2n} w_i^c (\mathbb{Z} - \mu_{z_k})(\mathbb{Z} - \mu_{z_k})^T + \mathrm{R} \tag{2.79}$$

then we compute the *residual* of the predicted state and true measurement value,

$$\mathrm{Y} = \mathrm{Z} - \mu_{z_k}, \tag{2.80}$$

here, Z represents the measurement values from the sensor and $\mu_{\mathbb{Z}_k}$ represents the weighted mean of the transformed sigma points.

2.4.5.1 Kalman Gain

The *Kalman Gain* is computed using the cross covariance matrix between the state and the measurements. The cross covariance matrix is calculated as follows,

$$\mathrm{P}_{\mathrm{X}\mathbb{Z}} = \sum w_i^c (y_\sigma - \mathrm{X})(\mathbb{Z} - \mu_z)^T \tag{2.81}$$

using in the calculation of *Kalman Gain*,

$$\mathrm{K} = \mathrm{P}_{\mathrm{X}\mathbb{Z}} \mathrm{P}_{\mathbb{Z}}^{-1} \tag{2.82}$$

Last, we calculate the weighted residual to compute the *a posteriori* estimate of the state $\hat{\mathrm{X}}_k$ and the corrected error covariance P as,

$$\hat{\mathrm{X}}_k = \mathrm{X}_k + \mathrm{K}y \tag{2.83}$$

$$\hat{\mathrm{P}}_k = \mathrm{P}_k - \mathrm{K}\mathrm{P}_{\mathbb{Z}}\mathrm{K}^T \tag{2.84}$$

Where, $\hat{\mathrm{X}}$ is the corrected estimate of the state X, and $\hat{\mathrm{P}}$ is the corrected error covariance.

Linear Kalman Filter	Extended Kalman Filter	Unscented Kalman Filter
$\mathrm{x}_k = \mathrm{F}\hat{\mathrm{x}}_{k-1} + \mathrm{B}\mathrm{u}_{k-1}$ $\mathrm{P}_k = \mathrm{F}\hat{\mathrm{P}}_{k-1}\mathrm{F}^T + \mathrm{Q}$	$\mathrm{F} = \frac{\partial f(\mathrm{x},\mathrm{u}))}{\partial \mathrm{x}}$ $\mathrm{x}_k = f(\hat{\mathrm{x}}_{k-1}, \mathrm{u})$ $\mathrm{P}_k = \mathrm{F}\hat{\mathrm{P}}_{k-1}\mathrm{F}^T + \mathrm{Q}$	$y_\sigma = f(\chi)$ $\mathrm{x}_k = \Sigma \mathrm{w}_m y_\sigma$ $\mathrm{P}_k = \sum w^c (y_\sigma - \mathrm{x}_k)(y_\sigma - \mathrm{x}_k)^T + \mathrm{Q}$
$\mathrm{y} = \mathrm{z} - \mathrm{H}\mathrm{x}$ $\mathrm{S} = \mathrm{H}\mathrm{P}_k\mathrm{H}^T + \mathrm{R}$ $\mathrm{K}_k = \mathrm{P}_k\mathrm{H}^T\mathrm{S}^{-1}$ $\hat{\mathrm{x}}_k = \mathrm{x}_k + \mathrm{K}_k\mathrm{y}$ $\hat{\mathrm{P}}_k = (\mathrm{I} - \mathrm{K}\mathrm{H})\mathrm{P}_k$	$\mathrm{H} = \frac{\partial h(\mathrm{x}_k)}{\partial \mathrm{x}}$ $\mathrm{y} = \mathrm{z} - h(\mathrm{x})$ $\mathrm{S} = \mathrm{H}\mathrm{P}_k\mathrm{H}^T + \mathrm{R}$ $\mathrm{K}_k = \mathrm{P}_k\mathrm{H}^T\mathrm{S}^{-1}$ $\hat{\mathrm{x}}_k = \mathrm{x}_k + \mathrm{K}_k\mathrm{y}$ $\hat{\mathrm{P}}_k = (\mathrm{I} - \mathrm{K}\mathrm{H})\mathrm{P}_k$	$\mathrm{y} = \mathrm{z} - \mu_z$ $\mathrm{P}_{\mathbb{Z}_k} = \sum w^c (\mathbb{Z} - \mu_{z_k})(\mathbb{Z} - \mu_{z_k})^T + \mathrm{R}$ $\mathrm{K} = [\sum w_i^c (y_\sigma - \mathrm{x})(\mathbb{Z} - \mu_z)^T]\mathrm{P}_{\mathbb{Z}}^{-1}$ $\hat{\mathrm{x}}_k = \mathrm{x}_k + \mathrm{K}_k\mathrm{y}$ $\hat{\mathrm{P}}_k = \mathrm{P}_k - \mathrm{K}\mathrm{P}_{\mathbb{Z}_k}\mathrm{K}^T$

Table 2.1: The mathematical comparison of the Linear, Extended and the Unscented Kalman Filter.

Chapter 3

Simulating Kalman Filters

This chapter covers the simulated application of Kalman Filtering to three vivid examples from mechanics (a) damped harmonic oscillator, (b) Duffing oscillator, and (c) Wilberforce pendulum. For each of these problems we derive the equations of motion using Euler-Lagrangian formulation and build their respective state space models amenable for Kalman Filtering. We explore the effects of increasing and decreasing the process and measurement noise on the performance of the filter and also evaluate how our problems are explained by the observability test.

3.1 Damped harmonic oscillator

We take example of a simple mass oscillator because harmonic oscillator is the underlying theory which form the bases for high ordered coupled complex problems such as vibrations. The fundamental role of pendulums and oscillations in describing myriad natural phenomena is the reason why we select it as our first example of the application of a Linear Kalman Filter in this chapter. We will then explore experimental realizations in Chapter 4.

3.1.1 Deriving the equation of motion using Lagrangian

Though the simple harmonic oscillator can be derived using the Newtonian mechanical system of equations, but, here we derive it from the system Lagrangian,

$$L = T - V \tag{3.1}$$

where L is called the *Lagrangian*, T is the *kinetic energy* and V is the *potential energy*. In a one dimensional horizontally placed mass-spring system Equation (3.1) takes the form:

$$L = \frac{1}{2}m\dot{x}^2 - \frac{1}{2}kx^2, \tag{3.2}$$

where x is the deviation from equilibrium.

For this system, we obtain $\partial L/\partial \dot{x} = m\dot{x}$ and $\partial L/\partial x = -kx$. This can be substituted into the Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = \frac{\partial L}{\partial x} \tag{3.3}$$

yielding the equation of motion,

$$\ddot{x} + \frac{k}{m}x = 0. \tag{3.4}$$

If there existed a damping effect to the simple harmonic oscillator then Equation (3.4) will also include a damping coefficient b:

$$\ddot{x} + \frac{b}{m}\dot{x} + \frac{k}{m}x = 0. \tag{3.5}$$

However, if the potential energy due to gravity is considered on a vertically oscillating system then:

$$V = \frac{1}{2}kx^2 + mg(h + x), \tag{3.6}$$

where, h is the height of the oscillator from the ground and x is the vertical displacement from equilibrium.

The Lagrangian with the inclusion of gravity takes the form,

$$\begin{aligned} L &= K - V \\ &= \frac{1}{2}m\dot{x}^2 - \frac{1}{2}kx^2 - mg(h + x) \end{aligned} \tag{3.7}$$

It is also possible to incorporate the damping term b right at the level of Euler-Lagrange equations. We define the energy lost in overcoming diction as,

$$P = \frac{1}{2} b\dot{x}^2. \tag{3.8}$$

With the generalized coordinate x and generalized applied force f, we can write the constrained Euler-Lagrange (E-L) equation:

$$\begin{aligned} f &= \frac{d}{dt}\Big(\frac{\partial L}{\partial \dot{x}}\Big) - \frac{\partial L}{\partial x} + \frac{\partial P}{\partial \dot{x}} \\ &= \frac{d}{dt}\Big(\frac{\partial}{\partial \dot{x}}\Big(\frac{1}{2}m\dot{x}^2 - \frac{1}{2}kx^2 - mg(h+x)\Big)\Big) \\ &- \frac{\partial}{\partial x}\Big(\frac{\partial}{\partial \dot{x}}\Big(\frac{1}{2}m\dot{x}^2 - \frac{1}{2}kx^2 - mg(h+x)\Big)\Big) + \frac{\partial}{\partial \dot{x}}\Big(\frac{1}{2}b\dot{x}^2\Big) \\ &= \frac{d}{dt}\Big(x\dot{x}^2\Big) - (-kx - mg) + (b\dot{x} \\ &= m\ddot{x} + b\dot{x} + kx + mg. \end{aligned} \tag{3.9}$$

where, k is an elastic constant that determines the restoring force when the oscillator is displaced from equilibrium, m is the mass, f is the driving force and b is the damping coefficient that results in the decaying behavior of the system over time.

The motion of the harmonic oscillator is periodic, repeating itself in a sinusoidal fashion with amplitude A. In addition to its amplitude, the motion of a simple harmonic oscillator is also characterized by its period T, the time for a single oscillation or its frequency i.e. $f = 1/T$ and $\omega = 2\pi f$ respectively. The position at a given time t also depends on the phase ϕ, which determines the starting point of the sine wave. The period and frequency are determined by the size of the mass m and the force constant k, while the amplitude and phase are determined by the starting position and velocity. When the motion of the oscillator reduces due to an external force it is referred to as damping. The amplitude in this case periodically decreasing, gradually bringing the system to rest. In this case, the energy of the oscillator dissipates continuously but for small damping the oscillations are approximately periodic.

The velocity and acceleration of a simple harmonic oscillator oscillates with the same frequency as the position, but with shifted phases. The velocity is maximal for zero displacement, while the acceleration is in the direction opposite to the displacement. These are basic results that are included by any textbook on mechanics [46].

3.1.2 State space model for the mass-spring system

In order to recast model of the vertically oscillating mass-spring damper system for Kalman Filtering we first convert the second order dynamic system (3.9) to a first ordered system in the state space. We define a column matrix for state variables $\mathrm{x}(t)$ where coordinates are:

$$\mathrm{x}_1 = x \tag{3.10}$$

$$\mathrm{x}_2 = \dot{x} = \dot{\mathrm{x}}_1 \tag{3.11}$$

which also gives us,

$$\dot{\mathrm{x}}_2 = \ddot{\mathrm{x}}_1 = \ddot{x} \tag{3.12}$$

Using these substitutions into Equation (3.9) we can write:

$$m\dot{\mathrm{x}}_2 + b\mathrm{x}_2 + k\mathrm{x}_1 = mg. \tag{3.13}$$

Here, we assume that, $f = 0$ (i.e. the system is oscillating freely).

From Equation (3.11), we know that,

$$\dot{\mathrm{x}}_1 = \mathrm{x}_2, \tag{3.14}$$

whereas from Equation (3.13) we deduce,

$$\dot{\mathrm{x}}_2 = -\frac{b}{m}\mathrm{x}_2 - \frac{k}{m}\mathrm{x}_1 + \frac{1}{m}u, \tag{3.15}$$

where, $u = g$ is treated as a control variable. The variables b, k, and m will later be treated as parameters. For the time being these are constants related to the physical states.

Equations (3.11) and (3.15) transform into matrix form, which is indeed the state space model,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u. \tag{3.16}$$

This is a first order equation. The matrices A and B are in accordance to Equation (2.3). The matrix A is referred to as the *system dynamic matrix* (3.17) and the matrix B is referred to as the *control matrix* (3.18).

$$\mathrm{A} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \tag{3.17}$$

$$\mathrm{B} = \begin{bmatrix} 0 \\ 1/m \end{bmatrix}. \tag{3.18}$$

Equation (3.16) is the continuous state space model for a simple one dimensional mass-spring damper system with one degree of freedom. Now, we proceed by converting this *continuous* state space model into a *discrete* time based state space model. This is because we are applying the Kalman Filter in a discrete state space. Through discretization as shown in Equation (2.17) we find the *state transition matrix* F. For simplification we also define $\omega^2 = k/m$ and $\gamma = b/m$. Upto first order in t we can write the following equation,

$$\mathrm{F} = \begin{bmatrix} 1 & \Delta t \\ -\omega^2 \Delta t & 1 - \gamma \Delta t \end{bmatrix}. \tag{3.19}$$

Equation (3.19) is called the process model for mass-spring damper sysem that we will use in the Kalman Filter.

3.1.3 Simulating the mass-oscillator

For this virtual experiment, we first run a simulation to exhibit the actual behavior of a simple harmonic oscillator and obtain the true values of position and velocity

given certain initial conditions and a fixed set of *known* parameters (k, m and b). This is done by solving the ordinary differential equation that governs this phenomenon (3.9) through numerical integration.

For the purpose of this thesis work we consider a two dimensional system of position and velocity. We assume that we have one position sensor with an absolute error of 0.1 m. We also assume that there is no driving force $f = 0$. Using the Matlab's $ODE45$ solver we solve the system to obtain the true values of the position and velocity of the oscillating system. The system simulation is shown in Figure 3.1.

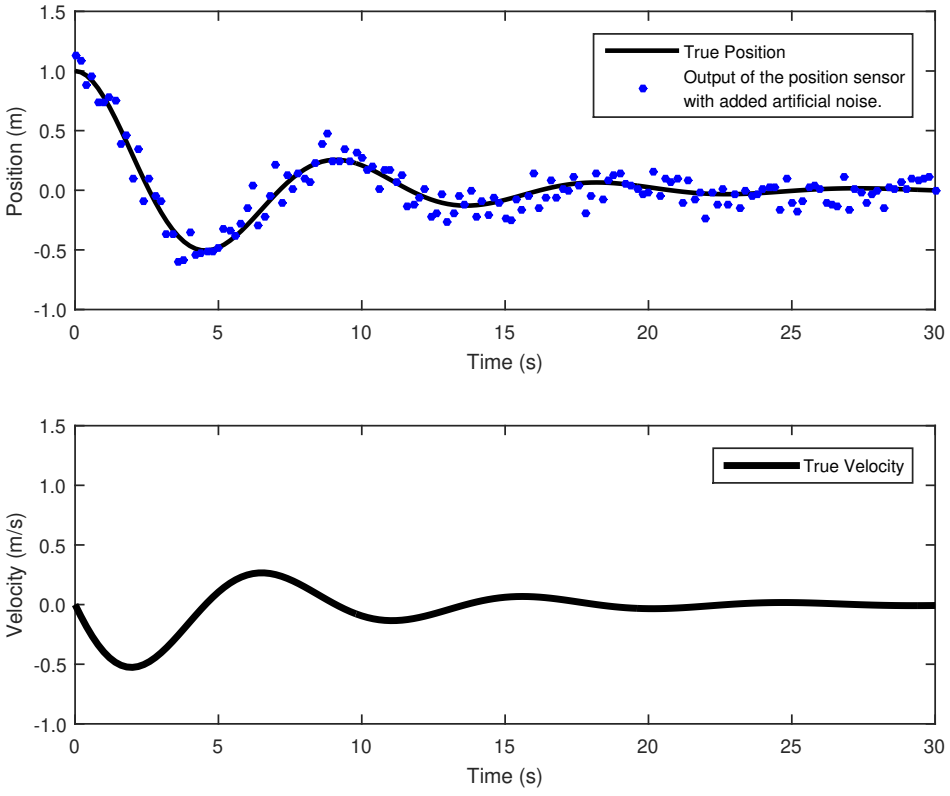

FIGURE 3.1: *The simulation of mass-spring damper system, (a) represents the position of oscillator in meters with spring constant $k = 5$ N/m, damping coefficient $b = 3$ Ns/m and mass $m = 10$ kg. (b) is the simulated velocity of the oscillator in meters per second.*

3.1.4 Observability test for the mass oscillator

Before applying Kalman Filter we test for the observability of the system, Using Equation (2.19), we solve for the observability test for our example of the mass-oscillator system for two cases, (a) one with only a position sensor and (b) with only a velocity sensor.

(a) Position sensor only: From Equation (3.17) the state dynamic matrix A in the damped mass-oscillator problem can be written as,

$$\mathrm{A} = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -\gamma \end{bmatrix}, \tag{3.20}$$

where we have defined $\omega^2 = k/m$ and $\gamma = b/m$.

We are only using a position sensor as the measurement input so the measurement matrix is:

$$\mathrm{C} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \tag{3.21}$$

In order to apply the observability test, we need to assign some values to these parameters. The value of the square of the frequency ω^2 is taken to be equal to 0.5 $rad^2 s^{-2}$ and the value of the damping factor γ is equal to 0.3 s^{-1}. Using only a position sensor the observability is as follows:

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.22}$$

The rank of this observability matrix is obviously 2, which is equal to the dimensions of the system. This means that our system is observable and we can infer both the position and velocity of the system using merely the position sensor.

(b) Velocity sensor only: Now let's test for the observability if we only have a velocity sensor. The only change is in the measurement matrix $C = \begin{pmatrix} 0 & 1 \end{pmatrix}$. The observability test yields.

$$\mathcal{O} = \begin{bmatrix} \mathrm{C} \\ \mathrm{CA} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -0.5 & -0.3 \end{bmatrix} \tag{3.23}$$

The rank of the observability matrix is 2, which is equal to the dimensions of the system, so this system is also observable using merely a velocity sensor.

Hence, for this two dimensional system we could use either a position or velocity sensor to filter out noise from the noisy oscillatory system and extract both the position and the velocity, thereby applying the Kalman Filter in both dimensions (position and velocity). We proceed to this step.

3.1.5 Applying the Kalman Filter

Using only a noisy position sensor, we now use the Kalman Filter to filter estimates of position and velocity as our first application of state estimation for a linear system.

We begin by initializing our state x_0 containing initial guess of position and velocity. We also specify the associated initial error covariance matrix P_0 as follows,

$$\mathrm{x}_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{3.24}$$

$$\mathrm{P}_0 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \tag{3.25}$$

where, P_0 contains the covariance of our initial position and velocity state variables. The selection of this variance is arbitrary but it has to be non-zero. This is because an initial error covariance of zero would mean that the initialization values are already true and thus the filter would not go through the test of its recursive belief prediction-correction algorithm.

In accordance with the filter equations described in Chapter 3 our time update equations are:

$$\mathrm{x}_k = \mathrm{F}\hat{\mathrm{x}}_{k-1}, \tag{3.26}$$

Equation (3.26) is the primary state prediction equation that projects the previous state vector estimation $\hat{\mathrm{x}}_{k-1}$ forward in time. Equation (3.19) is the state transition matrix F for our damped mass spring system. Placing this into Equation (3.26), we obtain:

$$\begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \end{bmatrix}_k = \begin{bmatrix} 1 & t \\ -\omega^2\Delta t & 1-\gamma\Delta t \end{bmatrix} \begin{bmatrix} \hat{\mathrm{x}}_1 \\ \hat{\mathrm{x}}_2 \end{bmatrix}_{k-1}. \tag{3.27}$$

Furthermore, the associated error covariance P_k of the states is also predicted to the current time step k from the corrected covariance $\hat{\mathrm{P}}_{k-1}$ in the last iteration. Here, in the first iteration we have initialized it with some non-zero covariance elements in P_0.

$$\mathrm{P}_k = \mathrm{F}\hat{\mathrm{P}}_{k-1}\mathrm{F}^T + \mathrm{Q}, \tag{3.28}$$

The process noise Q is added to the system to account for the consistent uncertainty in the state variables. In this case, we use the piecewise noise covariance matrix from Equation (2.54) as follows.

$$\begin{aligned} \mathrm{Q} &= \mathbb{E}[\Gamma\sigma^2\Gamma^T] \\ &= \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} \begin{bmatrix} \frac{1}{2}\Delta t^2 & \Delta t \end{bmatrix} \sigma^2 \\ &= \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 \end{bmatrix} \sigma^2 \end{aligned} \tag{3.29}$$

The current predicted error covariance P_k takes the following form:

$$\mathrm{P}_k = \begin{bmatrix} 1 & \Delta t \\ -\omega^2\Delta t & 1-\gamma\Delta t \end{bmatrix} \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} \begin{bmatrix} 1 & -\omega^2\Delta t \\ t & 1-\gamma\Delta t \end{bmatrix} + \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 \end{bmatrix} \tag{3.30}$$

After the *prediction* or the *time update*, the *measurement update* equations are as follows:

$$y_k = z_k - \mathrm{H}\mathrm{x}_k, \tag{3.31}$$

We simulated our damped harmonic oscillator using Equation (3.15). The MATLAB's *ODE*45 solver returned us the values of the true position of the oscillator. We infested the true position with an error of 0.1 m. This constituted our noisy measurement data in variable z_k that is accessed by the filter's measurement update Equation (3.31) and compared with the predicted state at current time step to calculate the *residual* Y_k of the system. The transfer function matrix H is used to bring the state being measured into the measurement space. In our case, as we measure the position directly $\mathrm{H} = \begin{bmatrix} 1 & 0 \end{bmatrix}$ and Equation (3.31) takes the form

$$y_k = z_k - \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{X}_1 \\ \mathrm{X}_2 \end{bmatrix}_k \tag{3.32}$$

$$= z_k - (\mathrm{X}_1)_k \tag{3.33}$$

Next, the *Kalman Gain* K is calculated using Equation (2.46). The gain decides relative priority to impart to the measurement values z_k or to use the predicted position values X_k. The gain is reproduced here for convenience.

$$\mathrm{K}_k = \mathrm{P}_k \mathrm{H}^T (\mathrm{H} \mathrm{P}_k \mathrm{H}^T + \mathrm{R})^{-1}, \tag{3.34}$$

Here, P_k is again the error covariance matrix, H is the transfer function matrix and R is the measurement noise that we infested into our simulated system, which represents that noisy sensor.

After incorporating the Kalman gain, the *a priori* prediction is corrected and we estimate the *a posteriori* value of the state.

$$\hat{\mathrm{X}}_k = \mathrm{X}_k + \mathrm{K}_k y_k, \tag{3.35}$$

We represent this estimated state vector $\hat{\mathrm{X}}_k$ with a *hat* and refer to it as the estimated state, or corrected state or the *a posterori* estimate.

Last, the filter rectifies the current predicted error covariance P_k using the error covariance estimate Equation (3.36). This updates the current belief which is used as input for the next iteration.

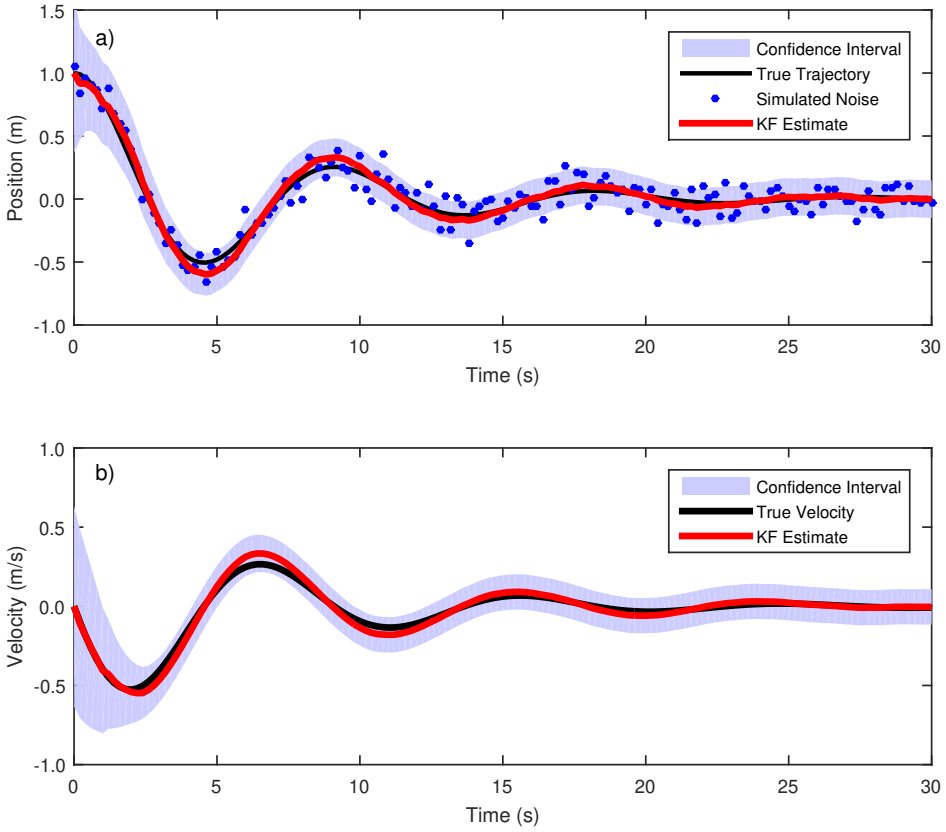

FIGURE 3.2: *Simulated application of the Kalman Filter on the noisy mass-spring damper system. Figure (a) shows the simulated noisy position values infested with* 0.1 *m white noise being filtered in the form of estimates. Figure (b) shows the estimated output of the filter along with the true simulated velocity of the system.*

$$\hat{\mathrm{P}}_k = (\mathrm{I} - \mathrm{K}_k\mathrm{H})\mathrm{P}_k. \tag{3.36}$$

The results of the Kalman Filter applied to the mass-spring damper system are shown in Figure 3.2. In Figure (a) we see the true and the noisy trajectory of the position of the damped harmonic oscillator simulated using Equation (3.15). The dots represent the artificial noise with a variance of 0.1 m we infested into our position sensor. The *KF Estimate* is the trajectory of the *a posteriori* state estimates from Equation (3.35). The shaded region is two times the standard deviation of the estimated states highlighting the 95% confidence interval of the filter's estimates. It is computed using the diagonal elements of P which represent the variance of state variable in x. As the filter converges, the spread of the

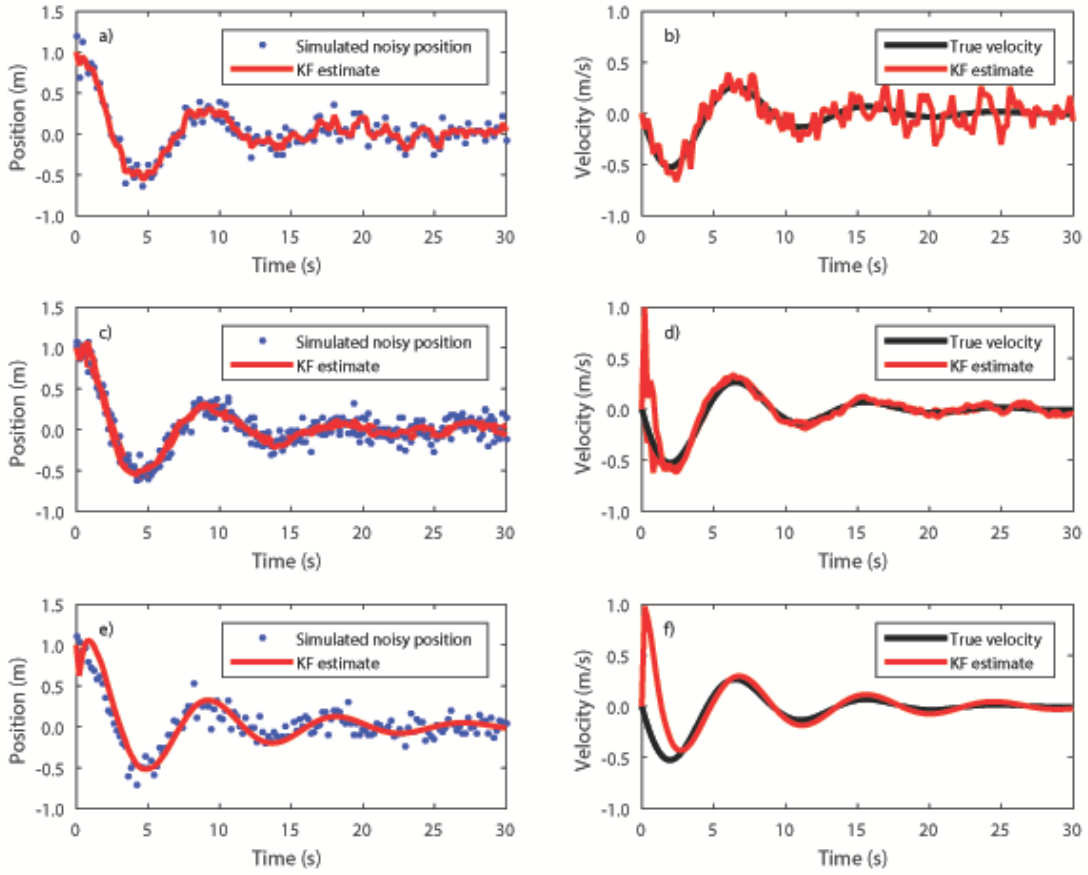

FIGURE 3.3: *A stack of plots exhibiting the effect of tuning the process noise covariance matrix* Q. *The Figure (a) and (b) depicts the effect of a large value of process noise variance* $\sigma^2 = 5.2^2$. *The Figure (c) and (d) show the effect of high initial covaraince matrix* $P_0 = 100$ *with high uncertainty in the process noise and Figure (e) and (f) show the response of the filter only due to a high initial error covariance. The measurement noise* R *remains constant at* 0.1 m.

estimates reduce, building the confidence on the estimates over time as shown by the shaded region in the plot.

We did not have a velocity sensor in this case, so, there is no noisy velocity trajectory seen in Figure (b). The simulated true trajectory of the velocity and the estimated velocity component of state is shown in the Figure (b).

3.1.6 Using process noise to tune the filter

The simulation discussed so far is an introductory example of applying Kalman Filter to a linear problem. We also explore the variation effects of varying the process noise covariance matrix on the performance of the filter. The filter operates on the weighted belief system between the sensor data and the modeled process.

In Figure 3.3, we can see the effect of varying the process noise covariance Q as a tuning parameter to adjust the filter's response to the noisy sensor data. If the value of Q is high as compared to the measurement error R, the filter considers the sensor data to be more trustworthy and starts to base its state estimates according to the sensor data, as shown in Figures (a) and (b).

If the provided initial error covariance P_0 has a large value, then the filter takes some time to converge to the optimum values of the state. This is shown in Figure (c) and (d) where we keep the diagonal values of the initial error covariance $P_0 = 100$. The process noise is kept high at $\sigma^2 = 5.2^2$, thus, the plots show how the filter still keeps following the noisy measurement values even after convergence. However, if the process noise Q is set to a value of zero, the filter assumes the process model to be completely accurate and completely ignores the true measurement values. In Figure (e) and (f) we keep the diagonal values of the initial covariance high i.e. $P_0 = 100$ and the variance of the process noise $\sigma^2 = 0$. This makes the initial convergence of the filter difficult and we see a delay in the state adapting to the true values of the system. However, since the process noise is completely zero, the filter completely ignores the measurement values and follows the ideal path of the underlying process model. These results show that the process noise covariance matrix Q is a primary tuning factor in determining the performance of the filter.

3.1.7 Kalman Gain

We have seen the effects of tuning the process noise covariance matrix Q and varying P in the previous section. This tuning also impacts how the Kalman Filter weighs its *a priori* prediction in comparison to the residual to make an *a posteriori* estimation. Figure 3.4 highlights the effects of incremental increase in the variance of the process noise covariance matrices Q on the *Kalman Gain* of the filter. From Equation (3.34), we can see that the Kalman Gain is a weighing factor that decides whether the Kalman Filter should prioritize the values of the noisy measurements obtained from the sensor or the uncertainty of the process model to make the *a posteriori* estimation of the state.

$$K_k = P_k H^T (H P_k H^T + R)^{-1}, \tag{3.37}$$

In Figure 3.4 we can see that for the highest value of the process noise variance at $\sigma = 50^2$ the filter completely ignores the process model and considers the value of the measurement input z_k coming from the transfer function to be the estimated output value of the filter. As we decrease the value of the variance σ^2 from 50^2 to 0.01^2 the Kalman gain decreases to the value around 0.01. This implies that the state estimates now reside more towards the process model rather than the noisy measurement values.

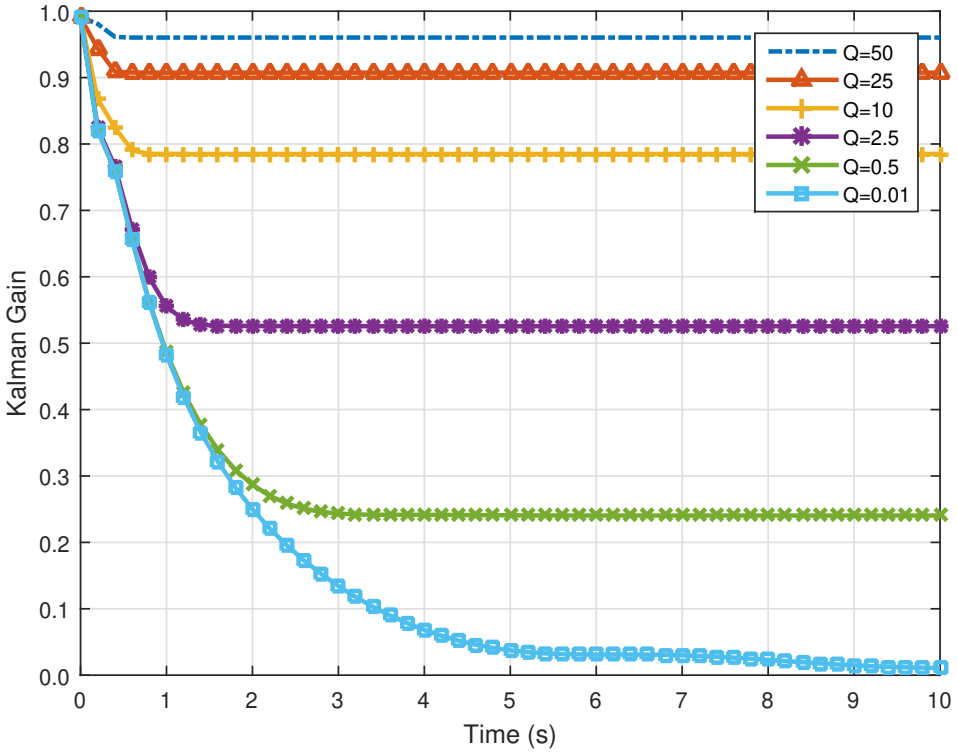

FIGURE 3.4: *The effects of tuning the process noise covariance matrix Q on the Kalman Gain. We keep a high initial error covariance* $P_0 = 10$

We keep the diagonal values of the initial covariance matrix to be $P_0 = 10^2$. This implies that the we want the filter to avoid trusting the process model in the beginning and later converge to the values according to the process noise covariance matrix. Otherwise, for a lower value, the filter would always consider the initial values of the state variable matrix x to be accurate and hence avoid measurement values as already discussed before.

3.1.8 Parameter Estimation

One of the physicist's motivations behind using the Kalman Filter is the fact that it empowers not only the estimation of complex dynamical states but also helps in estimating the many underlying sub-parameters which govern the properties of the system. For example, in our mass-spring damper system, the square of the natural frequency ω^2 and the damping factor γ are important parameters which determine the system kinematics. The Kalman Filter can also be used to estimate these parameters. The utility of this approach becomes evident when we compare it with the need to independently infer these quantities from independent, and sometimes elaborate experiments. For example, determining $\omega^2 = k/m$ requires knowledge of k which may require one to displace the oscillator by varying the mass and determining k through curve fit on a series of measurements. The Kalman Filter through state space approach with augmented parameters enables us to directly estimate these parameters by *measurements* of the time dynamics. The power of this technique is all the more conspicuous when dealing with complex, nonlinear systems. However, as a curtain raiser we implant this technique on a linear damped oscillator.

The parameter estimation by augmenting the parameters into the state space vector $\mathbf{x}$, makes the problem and the state update equations nonlinear. This means that now we need to resort to *nonlinear observability* tests and nonlinear incarnations of the Kalman Filter, such as the *Extended Kalman Filter* and the *Unscented Kalman Filter*. We explore the application of both to our simulated mass spring damper system and compare their performance.

3.1.8.1 Estimating the frequency and damping coefficient

For our damped mass oscillator, the frequency is dependent upon the spring constant and mass of the hanging object. During oscillation the oscillator is subjected to a damping force which is linearly dependent upon the velocity. In this process, the oscillations of the system experience an exponential decay which depends upon the damping coefficient γ.

Conventionally, the frequency and damping coefficient for a dynamical system are estimated using the time period of one oscillation and the gradient of the log decrements of the amplitudes. We use Extended Kalman Filter, described in

Section 2.3 to estimate these parameters. Due to unknowns in our state equations that system becomes nonlinear.

We first augment the state dynamic matrix with unknown parameters. The augmented state vector for the damped mass oscillator problem takes the following form:

$$\dot{\mathrm{X}} = \begin{bmatrix} \mathrm{x}_1 & \mathrm{x}_2 & \mathrm{x}_3 & \mathrm{x}_4, \end{bmatrix}^T \tag{3.38}$$

Here, x_1 is position, x_2 is velocity, x_3 is square of the frequency ω^2 and x_4 is the damping coefficient γ of the system. The system dynamical equation can be rewritten state space as follows:

$$\dot{\mathrm{X}} = f(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \mathrm{x}_4) \tag{3.39}$$

where the nonlinear function f is defined as,

$$f(\mathrm{x}) = \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_4\mathrm{x}_2 - \mathrm{x}_3\mathrm{x}_1 \\ 0 \\ 0 \end{pmatrix} \tag{3.40}$$

If only the position is being measured, the measurement matrix is

$$\mathrm{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}. \tag{3.41}$$

Now, due to state augmentation, the system becomes a nonlinear function of the parameters. Precisely, state propagation from time step $k-1$ to k is nonlinear while the measurement to state space conversion is linear. So, we linearize by taking the partial derivatives of the state equations with respect to the state variables. The resulting matrix is as follows:

$$\begin{aligned}\Phi &= \frac{\partial f(\mathrm{x})}{\partial \mathrm{x}} \\ &= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\mathrm{x}_3 & -\mathrm{x}_4 & -\mathrm{x}_1 & -\mathrm{x}_2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\end{aligned} \tag{3.42}$$

From Φ we derive the discretized state transfer matrix in accordance with Equation (2.17):

$$\begin{aligned}\mathrm{F} &= e^{\Phi \Delta t} \approx \mathrm{I} + \Phi \Delta t \\ &= \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ -\mathrm{x}_3 \Delta t & 1 - \mathrm{x}_4 \Delta t & -\mathrm{x}_1 \Delta t & -\mathrm{x}_2 \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\end{aligned} \tag{3.43}$$

Equation (3.43) is the discrete linear state transition matrix for parameter estimation in the mass spring damper system. This matrix is not used for the transition of state from previous time step to the current time step, rather, it is used in updating the belief or the error covariance from the *a posteriori* to the current *a priori* prediction. For state propagation we use the nonlinear function f, given in Equation (3.40) instead.

Before we apply the *Extended Kalman Filter* to estimate the parameters of this system we test for the observability of the system using only the position sensor.

3.1.8.2 Nonlinear observability test for parameter estimation in harmonic oscillator

The augmented state equations are nonlinear due to the presence of the sub-parameters. As discussed before the observability test is an important tool to know if it would be possible to estimate the state using a particular set of sensors at the output. In linear regime, an observability test is performed using Equation (2.19). However, in the case of a nonlinear system, the linear observability test

fails due to the nonlinearity of the state equations. For this, the observability test for a nonlinear system is performed using the derivation outlined around Equation (2.30).

From Equation (3.40), we know that:

$$f(\mathrm{x}) = \begin{bmatrix} \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ 0 \\ 0 \end{bmatrix} \tag{3.44}$$

and the (linear) observation equation is given by:

$$y(\mathrm{x}) = \mathrm{H}\mathrm{x} = \mathrm{x}_1 \tag{3.45}$$

The nonlinear observability is represented by the measurement function and its high ordered Lie derivatives with respect to the state. The basic idea is to compose the mapping matrix ϕ given in Equation (2.29).

Taking Lie derivatives of the measurement function yields

$$\begin{aligned} \pounds_f^0(y(\mathrm{x})) &= y(\mathrm{x}) = \mathrm{x}_1 \qquad (3.46) \\ \pounds_f^1(y(\mathrm{x})) &= \frac{\partial y}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\ &= \begin{pmatrix} \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_1} & \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_2} & \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_3} & \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_3} \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ 0 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ 0 \\ 0 \end{pmatrix} = \mathrm{x}_2 \qquad (3.47) \\ \pounds_f^2(y(\mathrm{x})) &= \frac{\partial \pounds_f^1}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \end{aligned}$$

$$
\begin{aligned}
&= \begin{pmatrix} \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_1} & \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_2} & \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_3} & \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_3} \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ 0 \\ 0 \end{pmatrix} = \mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\
\pounds_f^3(y(\mathrm{x})) &= \frac{\partial \pounds_f^2}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\
&= \begin{pmatrix} \frac{\partial -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2}{\partial \mathrm{x}_1} & \frac{\partial -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2}{\partial \mathrm{x}_2} & \frac{\partial -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2}{\partial \mathrm{x}_3} & \frac{\partial -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2}{\partial \mathrm{x}_3} \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} -\mathrm{x}_3 & -\mathrm{x}_4 & -\mathrm{x}_1 & -\mathrm{x}_2 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ 0 \\ 0 \end{pmatrix} \\
&= -\mathrm{x}_2\mathrm{x}_3 + \mathrm{x}_4\mathrm{x}_3\mathrm{x}_1 + \mathrm{x}_4^2\mathrm{x}_2 \qquad (3.48)
\end{aligned}
$$

Therefore, the resulting mapping matrix ϕ is:

$$
\phi = \begin{bmatrix} \pounds_f^0 \\ \pounds_f^1 \\ \pounds_f^2 \\ \pounds_f^3 \end{bmatrix} = \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \\ -\mathrm{x}_3\mathrm{x}_1 - \mathrm{x}_4\mathrm{x}_2 \\ -\mathrm{x}_2\mathrm{x}_3 + \mathrm{x}_4\mathrm{x}_3\mathrm{x}_1 + \mathrm{x}_4^2\mathrm{x}_2 \end{bmatrix} \qquad (3.49)
$$

Next, we take the Jacobian of the mapping matrix ϕ with respect to the state variables,

$$
\mathcal{O} = \frac{\partial \phi}{\partial \mathrm{x}} = \begin{bmatrix} \frac{\partial \pounds_f^0}{\partial \mathrm{x}_1} & \frac{\partial \pounds_f^0}{\partial \mathrm{x}_2} & \frac{\partial \pounds_f^0}{\partial \mathrm{x}_3} & \frac{\partial \pounds_f^0}{\partial \mathrm{x}_4} \\ \frac{\partial \pounds_f^1}{\partial \mathrm{x}_1} & \frac{\partial \pounds_f^1}{\partial \mathrm{x}_2} & \frac{\partial \pounds_f^1}{\partial \mathrm{x}_3} & \frac{\partial \pounds_f^1}{\partial \mathrm{x}_4} \\ \frac{\partial \pounds_f^2}{\partial \mathrm{x}_1} & \frac{\partial \pounds_f^2}{\partial \mathrm{x}_2} & \frac{\partial \pounds_f^2}{\partial \mathrm{x}_3} & \frac{\partial \pounds_f^2}{\partial \mathrm{x}_4} \\ \frac{\partial \pounds_f^3}{\partial \mathrm{x}_1} & \frac{\partial \pounds_f^3}{\partial \mathrm{x}_2} & \frac{\partial \pounds_f^3}{\partial \mathrm{x}_3} & \frac{\partial \pounds_f^3}{\partial \mathrm{x}_4} \end{bmatrix} \qquad (3.50)
$$

This results in the following nonlinear observability test matrix for the augmented parameter estimation of the damped harmonic oscillator model using only a position sensor.

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\mathrm{x}_3 & -\mathrm{x}_4 & -\mathrm{x}_1 & -\mathrm{x}_2 \\ (\mathrm{x}_4\mathrm{x}_3) & (-\mathrm{x}_3 + \mathrm{x}_4^2) & (-\mathrm{x}_2 + \mathrm{x}_4\mathrm{x}_1) & (\mathrm{x}_3\mathrm{x}_1 + 2\mathrm{x}_4\mathrm{x}_2) \end{bmatrix}, \tag{3.51}$$

The rank of this observability matrix is equal to the dimensions of the system i.e. $n = 4$. This means that this is observable.

3.1.8.3 Applying the Extended Kalman Filter to estimate parameters

We now apply an *Extended Kalman Filter* to estimate the unknown parameters. So far, we have derived the state transition matrix (3.43) and tested for the observability of this nonlinear system in (3.51) using one noisy position sensor (3.41).

Using simple Euler integration of our nonlinear state equation,

$$\dot{\mathrm{x}} = f(\mathrm{x}, u) \tag{3.52}$$

to propagate the state from time $k-1$ to k. The result of Euler integration of this function is:

$$\begin{aligned}
(\mathrm{x}_1)_k &= (\hat{\mathrm{x}}_1)_{k-1} + (\dot{\mathrm{x}}_2)_k \Delta t && (3.53)\\
(\dot{\mathrm{x}}_2)_k &= \frac{(\mathrm{x}_2)_k - (\hat{\mathrm{x}}_2)_{k-1}}{\Delta t} \\
(\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + (\dot{\mathrm{x}}_2)_k \Delta t \\
(\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + \left(-(\hat{\mathrm{x}}_4)_{k-1}(\hat{\mathrm{x}}_1)_{k-1} - (\hat{\mathrm{x}}_3)_{k-1}(\hat{\mathrm{x}}_2)_{k-1}\right) \Delta t && (3.54)\\
(\mathrm{x}_3)_k &= (\hat{\mathrm{x}}_3)_{k-1} && (3.55)\\
(\mathrm{x}_4)_k &= (\hat{\mathrm{x}}_4)_{k-1} && (3.56)
\end{aligned}$$

where x_1 is the position, x_2 is the velocity, x_3 is the square of the frequency, x_4 is the damping coefficient and Δt is the time step used for integration. Equation (3.52) is used to predict the position, velocity and parameters of the oscillator at the current time step. The system at every current time step will depend upon the estimated state variable $\hat{x}_{k-1}$ from the previous time step.

The error covariance matrix P_k is also propagated using Equation (3.28). We use the linearized discrete state transition matrix F given in Equation (3.43) in the propagation of the error covariance matrix. We initialize the error covariance matrix $\hat{P}_0$ as

$$\hat{P}_0 = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{3.57}$$

We use "hat" atop P to represent this initialization to be the initial *a posteriori* estimation of the error covariance of the state variables. We proceed by computing the *a priori* error covariance as follows,

$$P_k \;=\; F\hat{P}_{k-1}F^T + Q, \tag{3.58}$$

$$\begin{aligned} & P_k \\ = & \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ -x_3\Delta t & -x_4\Delta t & -x_1\Delta t & -x_2\Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix} \begin{bmatrix} 1 & -x_3\Delta t & 0 & 0 \\ \Delta t & -x_4\Delta t & 0 & 0 \\ 0 & -x_1\Delta t & 1 & 0 \\ 0 & -x_2\Delta t & 0 & 1 \end{bmatrix} \\ & + Q \end{aligned} \tag{3.59}$$

The process noise Q is computed using Equation (2.54) giving

$$Q = \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 & \frac{1}{2}\Delta t^2 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 & \Delta t & \Delta t \\ \frac{1}{2}\Delta t^2 & \Delta t & 1 & 1 \\ \frac{1}{2}\Delta t^2 & \Delta t & 1 & 1 \end{bmatrix} \sigma_v^2. \tag{3.60}$$

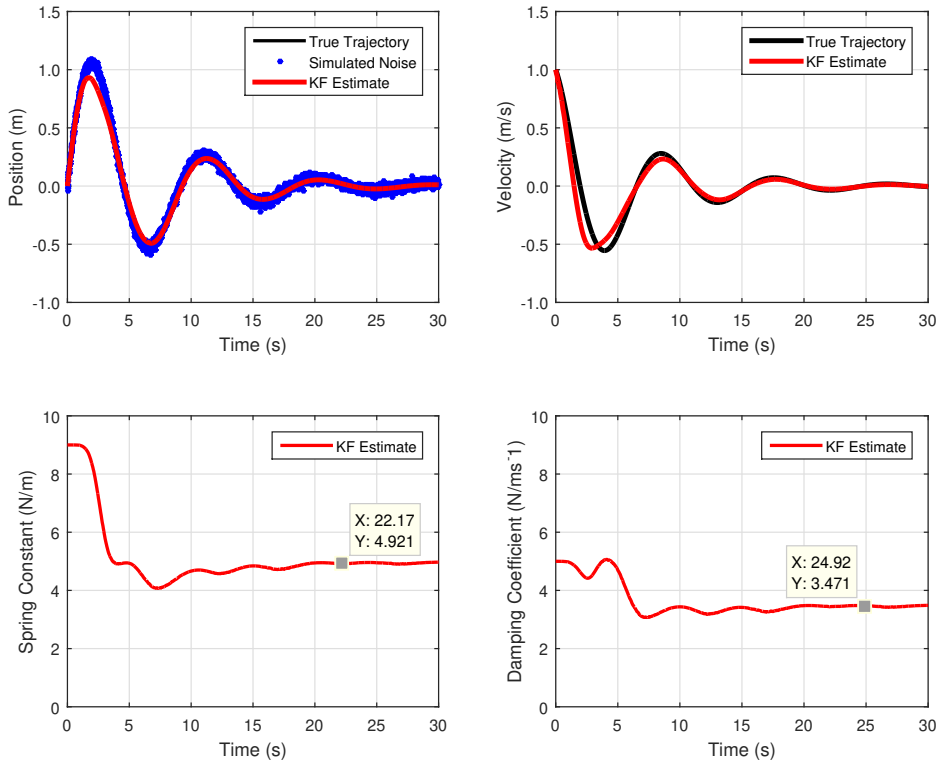

FIGURE 3.5: *The simulation of the joint state and parameter estimation of a damped mass spring system using Extended Kalman Filter. Figure (a) shows the simulated noisy position with the output of the filter's estimates KF Estimate. Figure (b) shows the true simulated and estimated velocity of the system through the process model of the filter. Figure (c) shows the estimated value of the spring constant, the true value was 5 N/m. Figure (d) shows the estimated value of the damping coefficient with true value 3 Ns/m.*

We inflate the process noise by altering the value of variance σ_v^2 to tune the performance of our Kalman Filter. In our simulated application we set this value to 0.01^2.

For our damped harmonic oscillator system we already have the simulated noisy position data that was previously used in the *residual* equation for the application of Kalman Filter. This step remains the same in parameter estimation as well. There is no change in rest of the algorithm and it is applied in the same fashion.

Kalman Gain is dependent upon the linearized error covariance matrix which is dependent upon the linearized state transition matrix. So, our state augmented nonlinear system for parameter estimation evolves with consideration of the uncertainties in the states and parameters to give us optimum estimated state outputs

from the filter. The Figure 3.5 shows the application of Extended Kalman Filter to the simulated damped harmonic oscillator with unknown parameters. The filter successfully estimates the spring constant and the damping coefficient of the system.

	Spring constant (N/m)	Damping coefficient (Ns/m)
True values	5	3
EKF estimate	4.921	3.471
RMSE	0.581	0.277

TABLE 3.1: The Root Mean Square Error (RMSE) values of true values and estimated values of spring constant and damping coefficient respectively.

3.1.9 Applying the Unscented Kalman Filter

Our difference equations of the system, already formulated in the previous application of the *Extended Kalman Filter*, are as follows,

$$
\begin{aligned}
(\mathrm{x}_1)_k &= (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1} T_s && (3.61)\\
(\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1} T_s \\
&= (\hat{\mathrm{x}}_2)_{k-1} + \left(-\frac{(\hat{\mathrm{x}}_4)_{k-1}}{m} (\hat{\mathrm{x}}_1)_{k-1} - \frac{(\hat{\mathrm{x}}_3)_{k-1}}{m} (\hat{\mathrm{x}}_2)_{k-1} \right) T_s && (3.62)\\
(\mathrm{x}_3)_k &= (\hat{\mathrm{x}}_3)_{k-1} T_s && (3.63)\\
(\mathrm{x}_4)_k &= (\hat{\mathrm{x}}_4)_{k-1} T_s && (3.64)
\end{aligned}
$$

We begin by calculating the scaled sigma points using the *Van der Merwe* algorithm. As we our system is now of $n = 4$ dimensions we have to create a matrix χ with size $2n + 1 = 9$.

$$
\chi = \begin{bmatrix}
\chi_{0,0} & \chi_{0,1} & \chi_{0,2} & \cdots & \chi_{0,9} \\
\chi_{1,0} & \chi_{1,1} & \chi_{1,2} & \cdots & \chi_{1,9} \\
\chi_{2,0} & \chi_{2,1} & \chi_{2,2} & \cdots & \chi_{2,9} \\
\chi_{3,0} & \chi_{3,1} & \chi_{3,2} & \cdots & \chi_{3,9}
\end{bmatrix}, \tag{3.65}
$$

The associated weights of these sigma points is,

$$\mathrm{w}_m = \begin{bmatrix} w_0 & w_1 & w_2 & \cdots & w_9 \end{bmatrix}^T \tag{3.66}$$

$$\mathrm{w}_c = \begin{bmatrix} w_0 & w_1 & w_2 & \cdots & w_9 \end{bmatrix}^T \tag{3.67}$$

where, w_m is the matrix of the associated weights in mean and w_c is the matrix of associated weights of the error covariance.

After we have generated sigma points and their associated weights, we pass them through the nonlinear function of the system.

$$y_\sigma = f(\chi) \tag{3.68}$$

This creates another matrix containing the transformed sigma points that have been passed through our nonlinear function. These transformed sigma points along with the weights are used to predict *a priori* state and the Gaussian distribution of the error covariance using the following equations,

$$\mathrm{x}_k = \sum_{i=0}^{8} \mathrm{w}_m y_\sigma \tag{3.69}$$

$$\mathrm{P}_k = \sum_{i=0}^{8} \mathrm{w}_c (y_\sigma - \mathrm{x}_k)(y_\sigma - \mathrm{x}_k)^T + \mathrm{Q} \tag{3.70}$$

where, w_m are the weights associated with the mean values and w_c are the weights of the error covariance P_k from the first and respective subsequent columns of the sigma points matrix.

Next, we create a measurement function that converts the sigma points into the measurements space for the update step of the filter,

$$\mathbb{Z} = h(y_\sigma) \tag{3.71}$$

Again, using the *Unscented Transform* we pass the sigma points and associated weights to the measurement function to compute the state and covariance in the measurement space as follows,

$$\mu_{z_k} = \sum_{i=0}^{8} \mathrm{w}_m \mathbb{Z} \tag{3.72}$$

$$\mathrm{P}_k = \sum_{i=0}^{8} \mathrm{w}_c (\mathbb{Z} - \mu_{z_k})(\mathbb{Z} - \mu_{z_k})^T + \mathrm{R} \tag{3.73}$$

then we compute the *residual* of the predicted state and true measurement value,

$$\mathrm{Y} = \mathrm{Z} - \mu_{z_k} \tag{3.74}$$

Next, the *Kalman Gain* is computed using the cross covariance matrix between the state and the measurements. The cross covariance matrix is calculated as follows,

$$\mathrm{P}_{\mathrm{X}\mathbb{Z}} = \sum_{i=0}^{8} \mathrm{w}_i^c (y_\sigma - \mathrm{X})(\mathbb{Z} - \mu_z))^T \tag{3.75}$$

using in the calculation of *Kalman Gain*,

$$\mathrm{K}_k = \mathrm{P}_{\mathrm{X}\mathbb{Z}} \mathrm{P}_{\mathbb{Z}}^{-1} \tag{3.76}$$

Last, we calculate the weighted residual to compute the *a posteriori* state estimate $\hat{\mathrm{X}}$ and the new error covariance as,

$$\hat{\mathrm{X}}_k = \mathrm{X}_k + \mathrm{K}y \tag{3.77}$$

$$\hat{\mathrm{P}}_k = \mathrm{P}_k - \mathrm{K}\mathrm{P}_{\mathbb{Z}}\mathrm{K}^T \tag{3.78}$$

The Figure 3.6 shows the simulated application of the *Unscented Kalman Filter* to our damped nonlinear harmonic oscillator problem for joint state and parameter estimation.

The performance of both, the Extended Kalman Filter and the Unscented Kalman Filter is compared by calculating the respective Root Mean Square Error (RMSE)

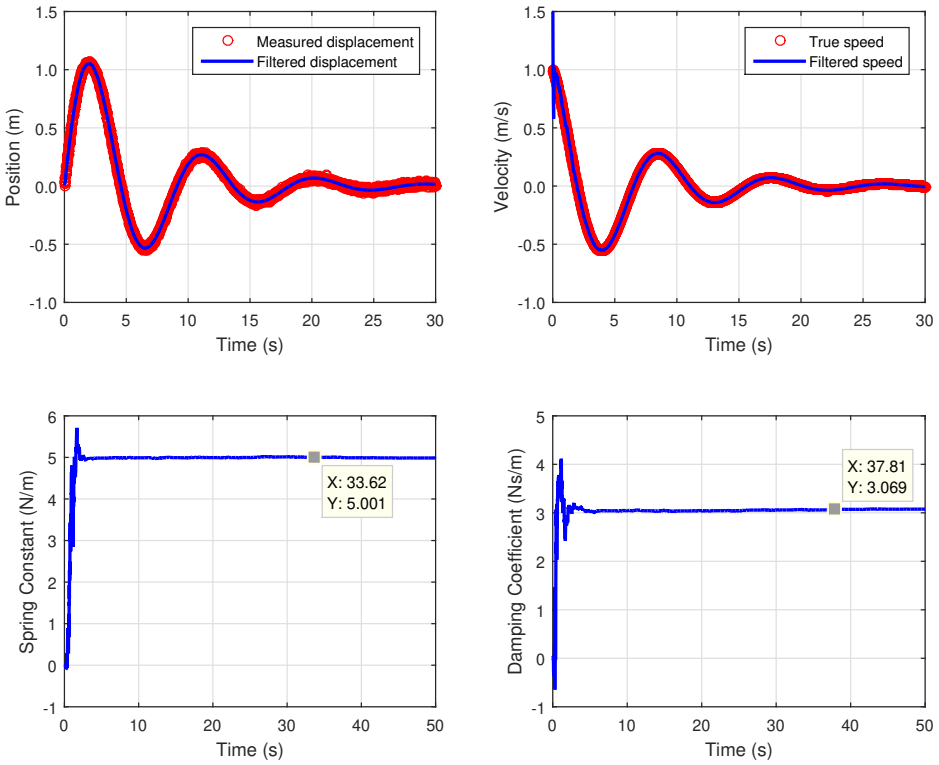

FIGURE 3.6: *The simulation of the joint state and parameter estimation of a damped mass spring system using Unscented Kalman Filter. (a) shows the simulated noisy position with the output of the filter's estimate. (b) shows the true simulated and estimated velocity of the system. (c) shows the estimated value of the spring constant, the true value is* 5 *N/m. (d) shows the estimated value of the damping coefficient with true value* 3 *Ns/m.*

of the estimated states and parameters of the system. It measures how much error is there between the estimated and actual values of the system.

The results are summarized in Table 3.1.9.

	Spring constant (N/m)		Damping coefficient (Ns/m)	
	EKF	UKF	EKF	UKF
True values	5		3	
Estimated	4.921	5.001	3.471	3.069
RMSE	0.619	0.686	0.629	0.379

TABLE 3.2: The performance comparison of Extended Kalman Filter and Unscented Kalman Filter using the Root Mean Square Error (RMSE) values for damped mass oscillator problem.

3.2 Duffing oscillator

In the previous section we covered the fundamental dynamical system of simple harmonic oscillator and observed how its linearity helps us in developing the foundation of Kalman Filter application for linear system problems. In this section we move a step further by observing an inherently nonlinear dynamical system and see how the Kalman Filter can help us in estimating the state in the presence of noise and also in estimating the system parameters.

3.2.1 Deriving the system equations

So far we have derived and observed the damped simple harmonic oscillator governed by the following equation:

$$m\ddot{x} = f(t) - b\dot{x} - kx \tag{3.79}$$

Equation (3.79) is a linear differential system, we were able to apply the simple *Linear Kalman Filter (LKF)* and understand its performance. However, this equation stems from an important approximation of the potential energy of the system $U = \frac{1}{2}kx^2$. From $F = \frac{-dU}{dx}$, the restoring force on the spring is $-k$. We wonder how high order terms in the potential energy would affect the behavior of system. For example, introducing the next order term in the potential energy expression gives,

$$U(x) \;=\; \frac{1}{2}kx^2 + \frac{1}{4}\beta x^3 \tag{3.80}$$

which results in the so-called *Duffing equation of motion.*

$$\ddot{x} + \omega^2 x + \beta x^3 \;=\; 0, \tag{3.81}$$

where we have assumed the absence of damping, $b = 0$, $\omega^2 = \frac{k}{m}$ and $f(t) = 0$. This is a non-linear differential equation and such an oscillator is sometimes referred to as an *anharmonic oscillator*. Reintroducing a periodic forcing $f(t) = a\cos\rho t$ and damping gives us,

$$\ddot{x} + \gamma\dot{x} + \omega^2 x + \beta x^3 = a\cos\rho t, \tag{3.82}$$

where γ, $\omega^2 = \frac{k}{m}$, β, a and ρ are the damping coefficient, square of the natural frequency, nonlinear cubic parameter and the excitation amplitude and frequency respectively.

3.2.2 Simulating the Actual Dynamics

We start by building the simulation model of the Duffing oscillator. We use the $ODE45$ solver to solve the Duffing Equation (3.82) for predefined parameters and then perform Kalman Filter on the measurement data. We will be able to estimate the state variables (position and velocity) as well as the system parameters γ, ω and β.

Working with pendulums, as we will explore in the next chapter dealing with experimental evaluations, we replace x, $\dot{\text{x}}$ with θ and $\dot{\theta}$, replacing all forces $f(t)$ with torques. In such a case Equation (3.82) takes the form,

$$\ddot{\theta} + \gamma\dot{\theta} + \omega^2\theta + \beta\theta^3 = a\cos\rho t, \tag{3.83}$$

where, $\ddot{x}$, $\dot{x}$, and k/m are replaced respectively by $\ddot{\theta}$, $\dot{\theta}$, and τ/I, τ and I being torsional constant (in place of spring constant) and moment of inertia (in place of mass).

We create a simulation model that exhibits the behavior of our non-linear oscillatory system. The system is excited by a known amplitude and frequency. We also create the recurrence Poincare map to track the periodicity of the oscillator and observe the dynamical pattern of the system in a phase space.

Using Equation (3.83) we can see how θ exhibits a strange behavior shown in Figure 3.7. In the phase space plot it becomes evident that the system is trapped in a strange attractor and keeps oscillating between them. The response of our system model is under purely deterministic parameters $\gamma = 0.3\ \ s^{-1}$, $\omega = -1$ rad/s, $\beta = 1$, $\rho = 1.25$ rad/s, and $a = 0.5$ rad.

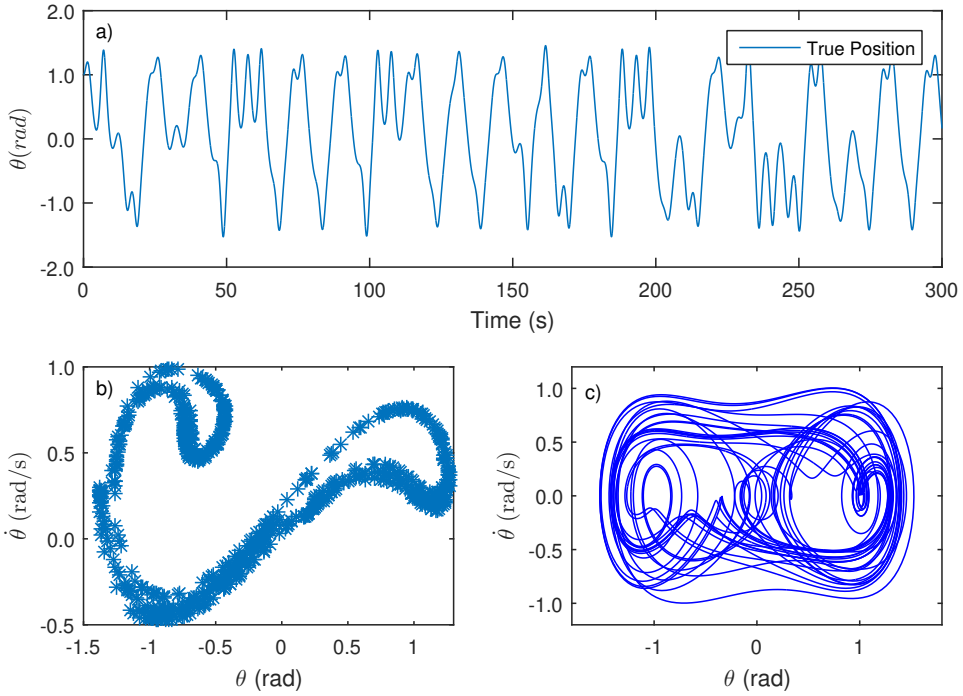

FIGURE 3.7: *The simulation of a Duffing oscillator. Figure (a) shows the chaotic position of the oscillator in time-domain. Figure (b) is a Poincare portrait of the phase space of the oscillator that captures a point in the phase after every* 2π *period. Figure (c) is the Phase Space plot that shows the double well strange attractor Duffing oscillator.*

3.2.3 Applying the Extended Kalman Filter

We use the *Extended Kalman Filter* to filter the nonlinear and noisy angular data of the duffing oscillator and then use it to estimate the nonlinear angular velocity $\dot{\theta}$ state of the system. Thus, our state-space again remains in only two dimensions.

3.2.3.1 State space model

In order to simulate the model for our nonlinear Duffing oscillator we first reduce the second order differential Equation (3.82) into state space as follows:

$$\mathrm{x}_1 = x$$
$$\mathrm{x}_2 = \dot{x}$$
$$\dot{\mathrm{x}}_1 = \mathrm{x}_2 \tag{3.84}$$
$$\dot{\mathrm{x}}_2 = -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \beta \mathrm{x}_1^3 + a \cos \rho t \tag{3.85}$$

Using these substitutions we derive the state space model for our Duffing oscillator system. As $\dot{\mathrm{x}} = f(x, u)$, the linearized form of the systems dynamic matrix A is modeled as follows,

$$\Phi = \begin{bmatrix} \frac{\partial \dot{\mathrm{x}}_1}{\partial \mathrm{x}} & \frac{\partial \dot{\mathrm{x}}_1}{\partial \mathrm{x}} \\ \frac{\partial \dot{\mathrm{x}}_2}{\partial \mathrm{x}} & \frac{\partial \dot{\mathrm{x}}_2}{\partial \mathrm{x}} \end{bmatrix} \tag{3.86}$$

the resulting linearized system dynamic matrix is,

$$\Phi = \begin{bmatrix} 0 & 1 \\ -\omega^2 - 3\beta \mathrm{x}_1^2 & -\gamma \end{bmatrix}, \tag{3.87}$$

This matrix is discretized using Equation (2.17),

$$\mathrm{F} \approx 1 + \Phi \Delta t \tag{3.88}$$
$$= \begin{bmatrix} 1 & \Delta t \\ (-\omega^2 - 3\beta \mathrm{x}_1^2)\Delta t & 1 - \gamma \Delta t \end{bmatrix}, \tag{3.89}$$

The resulting matrix F is the state transition matrix which is the discrete step process model for our Duffing system.

3.2.4 Applying Extended Kalman Filter for Duffing state estimation

3.2.4.1 Predicting the state

Using simple Euler integration of $\dot{x} = f(x, u)$, we propagate the state of the Duffing oscillator from the previous time step $k-1$ to the current time step k. For this we convert the first order differential equations of the system to first order difference equations as follows:

$$\begin{aligned} (\mathrm{x}_1)_k &= (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1}\Delta t \\ (\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + \left(-\gamma(\hat{\mathrm{x}}_2)_{k-1} - \omega^2(\hat{\mathrm{x}}_1)_{k-1} - \beta(\hat{\mathrm{x}}_1)^3_{k-1} + a\cos\rho t\right)\Delta t \end{aligned} \tag{3.90}$$

In this nonlinear propagation step, Δt is the time step which is chosen by the user and the instantaneous time $t = \Delta t$.

3.2.4.2 Predicting the error covariance

Even though the state is propagated using the compatible nonlinear function, propagation of error covariance matrix P requires use of the state transition matrix F which is already derived in Equation (3.89). The error covariance Equation (3.28) takes the following form,

$$\mathrm{P}_k = \begin{bmatrix} 1 & \Delta t \\ (-\omega^2 - 3\beta\mathrm{x}_1^2)\Delta t & 1-\gamma\Delta t \end{bmatrix}_k \hat{\mathrm{P}}_{k-1} \begin{bmatrix} 1 & (-\omega^2 - 3\beta\mathrm{x}_1^2)\Delta t \\ \Delta t & 1-\gamma\Delta t \end{bmatrix}_k + \mathrm{Q} \tag{3.91}$$

In this case, we also have a *Control Input* function that is exciting the *Duffing Oscillator* through a known frequency ρ and an amplitude a as seen on the right hand side of Equation (3.82).

$$\mathrm{U} = \begin{bmatrix} 0 \\ \cos\rho t \end{bmatrix} \tag{3.92}$$

In this case, the noise in the system is also infused in the control space so we must incorporate this control noise into the propagation of error covariance.

We first create a matrix Q_v that contains the noise variance in the control input.

$$\mathrm{Q}_v = \begin{bmatrix} 0 & 0 \\ 0 & \sigma_v^2 \end{bmatrix} \tag{3.93}$$

where, σ_v is the noise associated with the input variable $\cos \rho t$. Since, the system is nonlinear so we take partial derivative (Jacobian) of U to create a matrix Γ.

$$\Gamma = \frac{\partial f(x,u)}{\partial u} \begin{bmatrix} \frac{\partial f_1}{\partial \mathrm{U}_1} & \frac{\partial f_1}{\partial \mathrm{U}_2} \\ \frac{\partial f_2}{\partial \mathrm{U}_1} & \frac{\partial f_2}{\partial \mathrm{U}_2} \end{bmatrix} \tag{3.94}$$

The resulting linearized control matrix is,

$$\Gamma_k = \begin{bmatrix} 0 & 0 \\ 0 & -a \sin \rho t \end{bmatrix} \tag{3.95}$$

This is incorporated into Equation (3.91) to model the noise infestation to our nonlinear system as follows,

$$\mathrm{P}_k = \mathrm{F}_k \hat{\mathrm{P}}_{k-1} \mathrm{F}_k^T + \Gamma_k \mathrm{Q}_v \Gamma_k^T, \tag{3.96}$$

which shows the temporal propagation of the matrix P.

3.2.4.3 Correcting the state and error covariance

The measurement update equations will remain as before. As we are only simulating a noisy measurement of position so our measurement space is $h(x) = \mathrm{H}(x) = \mathrm{x}_1$ and

$$\begin{aligned} \mathrm{Y}_k &= z_k - h(x) \\ &= z_k - \mathrm{x}_1 \end{aligned} \tag{3.97}$$

The Kalman gain is calculated using (2.46).

$$\mathrm{K}_k = \mathrm{P}_k \mathrm{H}^T (\mathrm{H}\mathrm{P}_k \mathrm{H}^T + \mathrm{R})^{-1} \tag{3.98}$$

The residual Y is weighted with K gain and added to the current state prediction x_k to compute the corrected state estimation at the current step x. The same process is repeated to correct the error associated with the state in P_k at current time step k.

$$\hat{\mathrm{x}}_k = \mathrm{x}_k + \mathrm{K}_k \mathrm{Y}_k \tag{3.99}$$

$$\hat{\mathrm{P}}_k = (\mathrm{I} - \mathrm{K}_k \mathrm{H})\mathrm{P}_k \tag{3.100}$$

Figure 3.8 exhibits our application of the *Extended Kalman Filter* to filter out the noisy angular position of the Duffing oscillator along with giving an estimate of the angular velocity.

3.2.4.4 Root Mean Square Error (RMSE)

The performance of our filter is gauged by computing the *Root Mean Square Error (RMSE)*, which is also referred to as the *standard deviation of the residuals*. It measures how much error is there between two sets of data. Simply, in the case of Kalman filtering it compares the true values of the state with the estimated values output by the filter.

The RMSE is calculated by

$$RMSE \;=\; \sqrt{\frac{\Sigma_{i=1}^{n} (\hat{\mathrm{x}}_i - z_i)^2}{n}}, \tag{3.101}$$

where, $\hat{\mathrm{x}}$ is the state estimate from the filter and z is the current measurement value from the sensor.

The following table highlights the *RMSE* values of our estimated states compared with the true values. We will further compare these results with the application of the Unscented Kalman Filter as well as in Joint State and Parameter estimation regime.

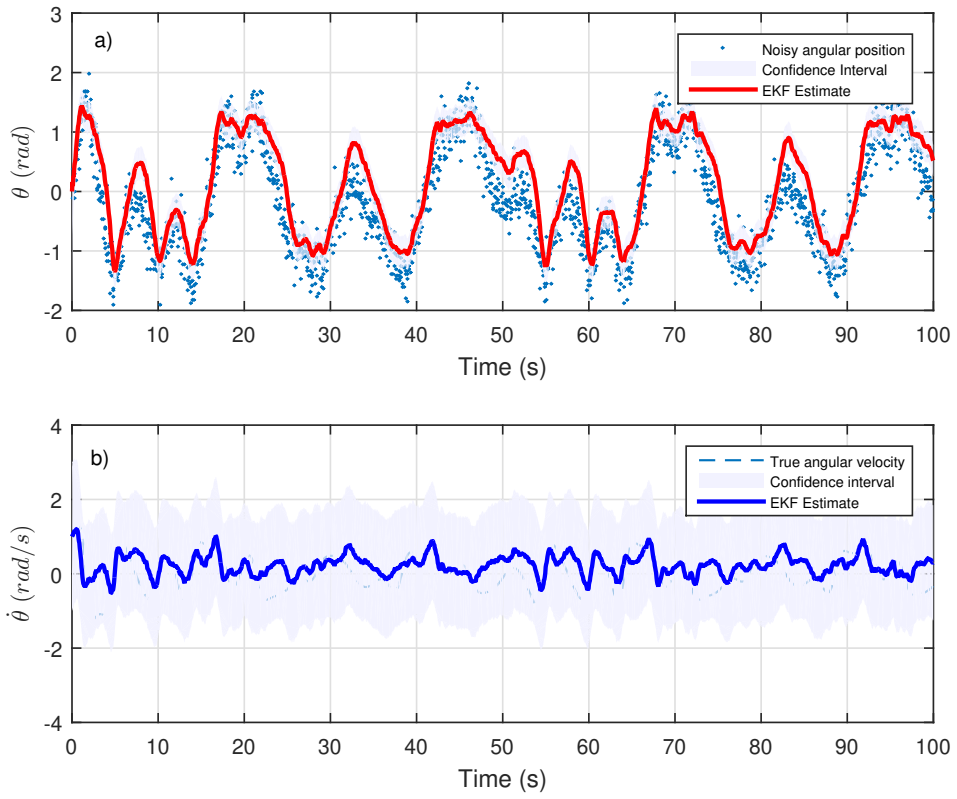

FIGURE 3.8: *The simulation of Extended Kalman Filter application to the Duffing Oscillator showing (a) shows the filtered angular position overlying the noisy trajectory of and θ rad (b) shows the estimated angular velocity over the true trajectory.*

	Angular position θ (rad)	Angular velocity $\dot{\theta}$ rad/s
RMSE	0.3889	0.4789

TABLE 3.3: The performance of the Extended Kalman Filter to filter a noisy Duffing oscillator

3.2.5 Estimating the cubic nonlinear parameter

The Duffing oscillator is a nonlinear system due to the cubic term in Equation (3.81). For parameter estimation the state space is augmented to include the unknown term which further increases the dimension and complexity of our system. So, before applying the filter to estimate the nonlinear parameter, we perform the observability test to see whether it would be possible for us to estimate β using merely an angular position sensor.

3.2.5.1 Nonlinear observability test for Duffing parameter estimation

We supplement the state vector x given in Equation (3.81) and (3.85) by $\mathrm{x}_3 = \beta$, $\dot{\mathrm{x}}_3 = 0$, allowing us to write the nonlinear propagation function as

$$f(\mathrm{x}) = \begin{bmatrix} \mathrm{x}_2 \\ -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \mathrm{x}_3 \mathrm{x}_1^3 \\ 0 \end{bmatrix} \tag{3.102}$$

and the measurement equation, which exhibits our use of an angle sensor for measuring the position in state x_1, is as follows,

$$y(\mathrm{x}) = \mathrm{H}(\mathrm{x}) = \mathrm{x}_1. \tag{3.103}$$

Taking Lie derivatives of this measurement function with the nonlinear function in Equation (3.102),

$$\begin{aligned} \pounds_f^0(y(\mathrm{x})) &= y(\mathrm{x}) = \mathrm{x}_1 && (3.104) \\ \pounds_f^1(y(\mathrm{x})) &= \frac{\partial y}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\ &= \begin{pmatrix} \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_1} & \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_2} & \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_3} \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \mathrm{x}_3 \mathrm{x}_1^3 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \mathrm{x}_3 \mathrm{x}_1^3 \\ 0 \end{pmatrix} = \mathrm{x}_2 && (3.105) \\ \pounds_f^2(y(\mathrm{x})) &= \frac{\partial \pounds_f^1}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\ &= \begin{pmatrix} \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_1} & \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_2} & \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_3} \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \mathrm{x}_3 \mathrm{x}_1^3 \\ 0 \end{pmatrix} \end{aligned}$$

$$
\begin{aligned}
&= \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \mathrm{x}_3 \mathrm{x}_1^3 \\ 0 \end{pmatrix} \\
&= -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \mathrm{x}_3 \mathrm{x}_1^3
\end{aligned} \tag{3.106}
$$

The resulting mapping matrix ϕ is,

$$
\phi = \begin{bmatrix} \pounds_f^0 \\ \pounds_f^1 \\ \pounds_f^2 \end{bmatrix} = \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \\ -\gamma \mathrm{x}_2 - \omega^2 \mathrm{x}_1 - \mathrm{x}_3 \mathrm{x}_1^3 \end{bmatrix} \tag{3.107}
$$

Taking partial derivatives of the mapping matrix ϕ with respect to the state variables x as shown in Equation (3.50), results in the following observability test matrix,

$$
\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\omega^2 - 3\mathrm{x}_1^2 \mathrm{x}_3 & -\gamma & -\mathrm{x}_1^3 \end{bmatrix}, \tag{3.108}
$$

The rank of this observability matrix is equal to the dimensions of the system i.e. $n = 3$. This means that this nonlinear system for joint state and parameter estimation of the Duffing oscillator using only an angular position sensor remains observable.

3.2.5.2 Estimating the cubic nonlinearity parameter using Extended Kalman Filter

We augment the state vector with the nonlinear coefficient as a new state variable $\mathrm{x}_3 = \beta$. The new joint state and parameter variables matrix is as follows:

$$
\mathrm{x} = \begin{bmatrix} \mathrm{x}_1 & \mathrm{x}_2 & \mathrm{x}_3 \end{bmatrix}^T \tag{3.109}
$$

Here, x_1 is the angular position, x_2 is the angular velocity and x_3 is the nonlinear cubic term β. Thus, the dynamics of the system can be rewritten as:

$$
\begin{aligned}
\mathrm{x}_1 &= x \\
\dot{\mathrm{x}}_1 &= \dot{x} \\
\dot{\mathrm{x}}_2 &= -\gamma\dot{\mathrm{x}}_1 + \omega^2\mathrm{x}_1 - \mathrm{x}_3\mathrm{x}_1^3 + f\cos\rho t \qquad (3.110)\\
\dot{\mathrm{x}}_3 &= 0 \qquad (3.111)\\
& \qquad (3.112)
\end{aligned}
$$

where only angular position is being measured as:

$$\mathrm{H} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \qquad (3.113)$$

In this case, our system dynamics model after taking linearization is,

$$\Phi_k = \begin{bmatrix} 0 & 1 & 0 \\ -\gamma & -\mathrm{x}_1^3 & \omega^2 - 3\mathrm{x}_3\mathrm{x}_1^2 \\ 0 & 0 & 0 \end{bmatrix} \qquad (3.114)$$

Using simple Euler integration of the state difference equations,

$$
\begin{aligned}
(x_1)_k &= (\hat{x}_1)_{k-1} + (\hat{x}_2)_{k-1}\Delta t \qquad (3.115)\\
(x_2)_k &= (\hat{x}_2)_{k-1} + \left(-\gamma(\hat{x}_2)_{k-1} - \omega^2(\hat{x}_1)_{k-1} - (\hat{x}_3)_{k-1}(\hat{x}_1)_{k-1}^3 + a\cos\rho t\right)\Delta t \qquad (3.116)\\
(x_3)_k &= (\hat{x}_3)_{k-1} \qquad (3.117)
\end{aligned}
$$

the states of the Duffing system along with the unknown cubic nonlinear parameter are propagated in time from $k-1$ to k. This is the same *a priori* or *Prediction step* as discussed previously.

The error covariance matrix P is propagated using Equation (3.96), where F is the linearized state matrix derived from Equation (2.17),

$$\mathrm{F}_k \approx 1 + \Phi_k \Delta t \tag{3.118}$$

$$= \begin{bmatrix} 1 & \Delta t & 0 \\ (-\gamma)\Delta t & (1 - \mathrm{x}_1^3)\Delta t & (\omega^2 - 3\mathrm{x}_3\mathrm{x}_1^2)\Delta t \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.119}$$

The remaining steps to estimate the current state and error covariance matrix remains the same as in previous sections. The Figure 3.9 exhibits the simulated application of the Extended Kalman Filter estimating the nonlinear cubic term as well as the states of the Duffing oscillator.

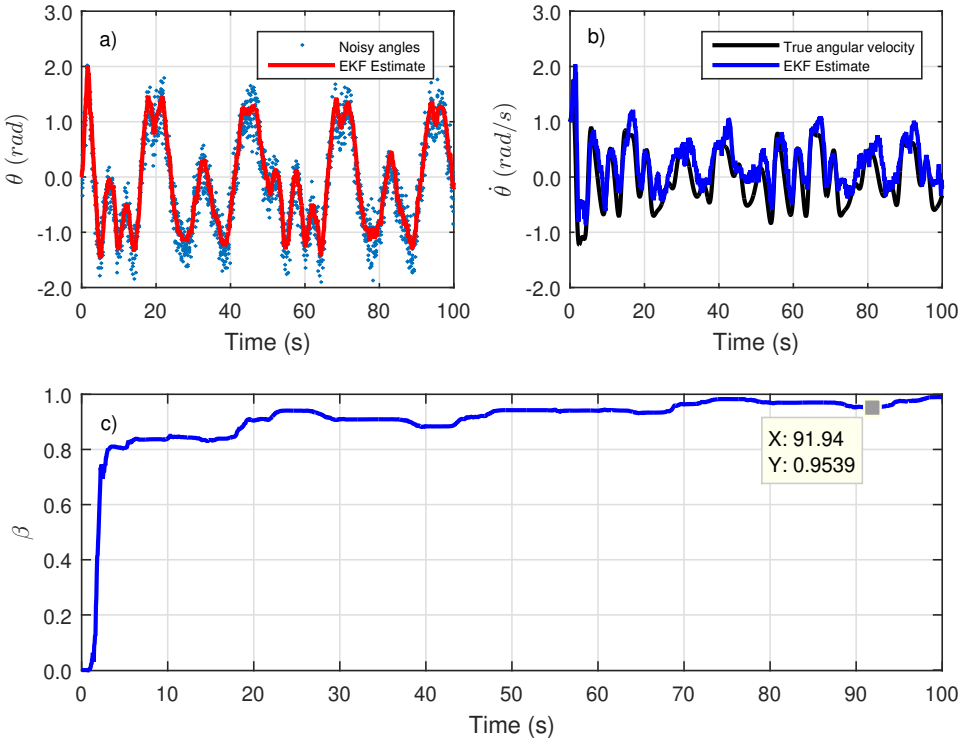

FIGURE 3.9: *The simulation of the joint state and parameter estimation of Duffing oscillator using Extended Kalman Filter. (a) shows the simulated noisy angular position with the output of the filter's estimates EKF Estimate. (b) shows the true simulated and estimated angular velocity of the system through the process model of the filter. (c) shows the estimated value of the nonlinear cubic parameter, the true value was* 1 N/rad^3.

3.2.6 Applying Unscented Kalman Filter for state estimation

To apply the *Unscented Kalman Filter* to the Duffing oscillator for the state estimation we first calculate the scaled sigma points using the *Van der Merwe* algorithm. In this application we first generate $2n + 1 = 5$ sigma points for our states.

Our nonlinear difference equations are,

$$\begin{aligned}(\mathrm{x}_1)_k &= (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1}\Delta t \\ (\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + \left(-\gamma(\hat{\mathrm{x}}_2)_{k-1} - \omega^2(\hat{\mathrm{x}}_1)_{k-1} - \beta(\hat{\mathrm{x}}_1)^3_{k-1} + a\cos\rho t\right)\Delta t.\end{aligned} \tag{3.120}$$

We get the following matrix containing the 5 scaled sigma points that approximate our states of angular position and velocity.

$$\chi = \begin{bmatrix} \chi_{0,0} & \chi_{0,1} & \chi_{0,2} & \chi_{0,3} & \chi_{0,4} & \chi_{0,5} \\ \chi_{1,0} & \chi_{1,1} & \chi_{1,2} & \chi_{1,3} & \chi_{1,4} & \chi_{1,5} \end{bmatrix}, \tag{3.121}$$

The associated weights for the sigma points are calculated using Equations (2.73) and (2.74).

$$\mathrm{w}_m = \begin{bmatrix} w_0 & w_1 & w_2 & w_3 & w_4 & w_5 \end{bmatrix}^T. \tag{3.122}$$

$$\mathrm{w}_c = \begin{bmatrix} w_0 & w_1 & w_2 & w_3 & w_4 & w_5 \end{bmatrix}^T. \tag{3.123}$$

After we have generated sigma points and their associated weights, we pass them through the nonlinear function,

$$y_\sigma = f(\chi_1) \tag{3.124}$$

$$= (\chi)_{k-1} + (\chi_2)_{k-1}\Delta t \tag{3.125}$$

$$= (\chi_2)_{k-1} + \left(-\gamma(\chi_2)_{k-1} - \omega^2(\chi_1)_{k-1} - \beta(\chi_1)^3_{k-1} + a\cos pt\right)\Delta t. \tag{3.126}$$

These transformed sigma points along with the weights are used to predict *a priori* state and the Gaussian distribution of the error covariance using the following equations,

$$\mathrm{x}_k = \sum_{i=0}^{4} \mathrm{w}_i^m y_\sigma \tag{3.127}$$

$$\mathrm{P}_k = \sum_{i=0}^{4} \mathrm{w}_i^c (y_\sigma - \mathrm{x}_k)(y_\sigma - \mathrm{x}_k)^T + \mathrm{Q} \tag{3.128}$$

where, w^m are the weights associated with the mean values and w^c are the weights of the error covariance P_k.

Next, we create a measurement function that converts the sigma points into the measurements space for the update step of the filter,

$$\mathbb{Z} = h(y_\sigma) \tag{3.129}$$

Again, using the *Unscented Transform* we pass the sigma points and associated weights to the measurement function to compute the state and covariance in the measurement space as follows,

$$\mu_{z_k} = \sum_{i=0}^{5} w^m \mathbb{Z} \tag{3.130}$$

$$\mathrm{P}_k = \sum_{i=0}^{5} w^c (\mathbb{Z} - \mu_{z_k})(\mathbb{Z} - \mu_{z_k})^T + R \tag{3.131}$$

then we compute the *residual* of the predicted state and true measurement value,

$$\mathrm{Y} = \mathrm{z} - \mu_{z_k} \tag{3.132}$$

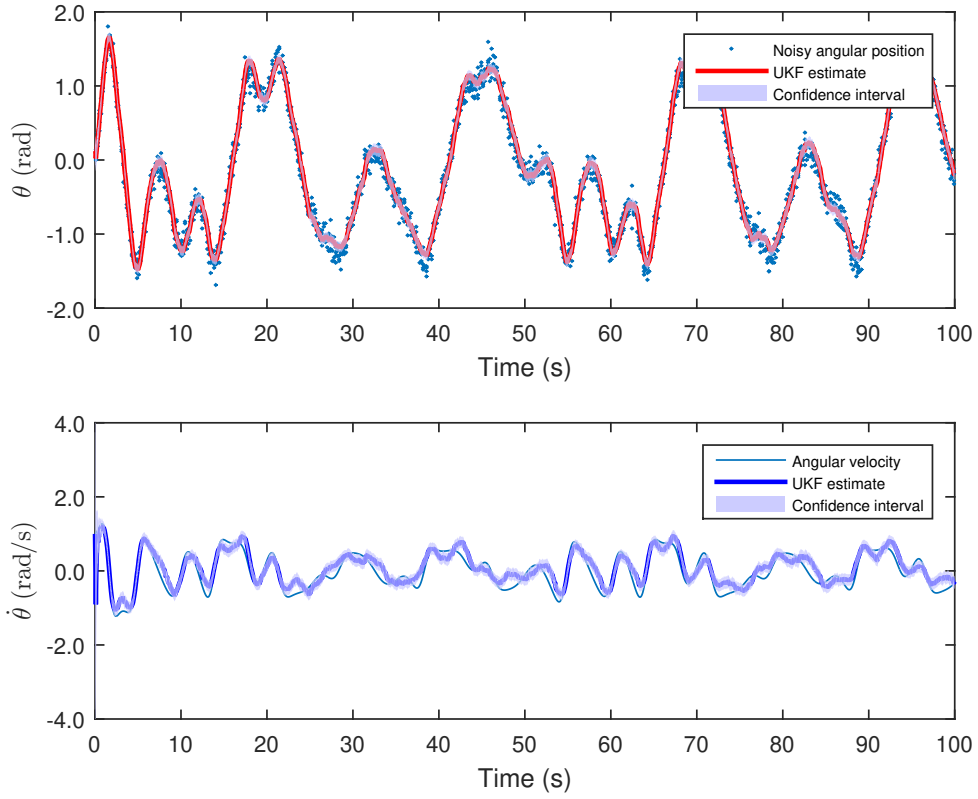

FIGURE 3.10: *The application of Unscented Kalman Filter to the Duffing Oscillator*

Next, the *Kalman Gain* is computed using the cross covariance matrix between the state and the measurements. The cross covariance matrix is calculated as follows,

$$\mathrm{P}_{\mathrm{x}\mathbb{Z}} = \sum w_i^c (y_\sigma - \mathrm{x})(\mathbb{Z} - \mu_z))^T \tag{3.133}$$

using in the calculation of *Kalman Gain*,

$$\mathrm{K} = \mathrm{P}_{\mathrm{x}\mathbb{Z}} \mathrm{P}_{\mathbb{Z}}^{-1} \tag{3.134}$$

Last, we calculate the weighted residual to compute the *a posteriori* state estimate $\hat{\mathrm{x}}$ and the new error covariance as,

$$\mathrm{x}_k = \hat{\mathrm{x}}_k + Ky \tag{3.135}$$

$$\mathrm{P}_k = \mathrm{P}_k - \mathrm{K}\mathrm{P}_{\mathbb{Z}}\mathrm{K}^T \tag{3.136}$$

The Figure 3.10 exhibits the simulated application of the Unscented Kalman Filter algorithm to estimate the states of a highly nonlinear classic Duffing oscillator problem. The robust sigma points calculation approximates the Duffing system in a much more robust form than the Extended Kalman Filter. We compare the state estimation results in Table 3.2.6.

	θ rad	$\dot{\theta}$ $rads^{-1}$
EKF	0.3889	0.4789
UKF	0.1128	0.2124

TABLE 3.4: The performance comparison of Extended Kalman Filter and Unscented Kalman Filter using the Root Mean Square Error (RMSE) values for the state estimation of the Duffing oscillator.

3.2.7 Estimating the nonlinear coefficient

We again calculate the sigma points for our joint state and parameter estimation. We will have $2n + 1 = 7$ sigma points to represent the distribution of our states and the unknown parameter. The total number of sigma points calculated using *Van Der Merwe sigma point algorithm* are packed in the sigma matrix as follows,

$$\chi = \begin{bmatrix} \chi_{0,0} & \chi_{0,1} & \chi_{0,2} & \chi_{0,3} & \chi_{0,4} & \chi_{0,5} & \chi_{0,6} & \chi_{0,7} \\ \chi_{1,0} & \chi_{1,1} & \chi_{1,2} & \chi_{1,3} & \chi_{1,4} & \chi_{1,5} & \chi_{1,6} & \chi_{1,7} \\ \chi_{2,0} & \chi_{2,1} & \chi_{2,2} & \chi_{2,3} & \chi_{2,4} & \chi_{2,5} & \chi_{2,6} & \chi_{2,7} \end{bmatrix}, \tag{3.137}$$

The associated weights for the sigma points matrix are,

$$\mathrm{w}_m = \begin{bmatrix} w_0 & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 \end{bmatrix}^T \tag{3.138}$$

$$\mathrm{w}_c = \begin{bmatrix} w_0 & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 \end{bmatrix}^T \tag{3.139}$$

Passing these through the nonlinear function of the system.

$$
\begin{aligned}
y_\sigma &= f(\chi) && (3.140)\\
&= (\chi_1)_{k-1} + (\chi_2)_{k-1}\Delta t && (3.141)\\
&= (\chi_2)_k + \left(-\gamma(\chi_2)_k - \omega^2(\chi_1)_k - \beta(\chi_1)_k^3 + a\cos pt\right)\Delta t && (3.142)\\
&= (\chi_3)_k && (3.143)
\end{aligned}
$$

Using the transformed sigma points the state containing unknown parameters along with the error covariance are propagated to the next step using the following equations,

$$
\begin{aligned}
\mathrm{x}_k &= \sum_{i=1}^{7} \mathrm{w}_i^m y_\sigma && (3.144)\\
\mathrm{P}_k &= \sum_{i=1}^{7} \mathrm{w}_i^c (y_\sigma - \mathrm{x}_k)(y_\sigma - \mathrm{x}_k)^T + \mathrm{Q} && (3.145)
\end{aligned}
$$

Next, the measurement function converts the sigma points into the measurements space for the update step as follows,

$$\mathbb{Z} = h(y_\sigma) \tag{3.146}$$

Again using the *Unscented Transform* we compute the Gaussian distribution for the estimated state and error covariance matrix of our system as,

$$
\begin{aligned}
\mu_{z_k} &= \sum_{i=1}^{7} w^m \mathbb{Z} && (3.147)\\
\mathrm{P}_k &= \sum_{i=1}^{7} w^c (\mathbb{Z} - \mu_{z_k})(\mathbb{Z} - \mu_{z_k})^T + R && (3.148)
\end{aligned}
$$

then we compute the *residual* of the predicted state and true measurement value,

$$\mathrm{Y} = \mathrm{Z} - \mu_{z_k} \tag{3.149}$$

Next, the *Kalman Gain* is computed using the cross covariance matrix between the state and the measurements. The cross covariance matrix is calculated as follows,

$$\mathrm{P}_{\mathrm{X}\mathbb{Z}} = \sum_{i=1}^{7} \mathrm{W}_i^c (y_\sigma - \mathrm{X})(\mathbb{Z} - \mu_z))^T \tag{3.150}$$

using in the calculation of *Kalman Gain*,

$$\mathrm{K} = \mathrm{P}_{\mathrm{X}\mathbb{Z}} \mathrm{P}_{\mathbb{Z}}^{-1} \tag{3.151}$$

Last, we calculate the weighted residual to compute the *a posteriori* state estimate $\hat{\mathrm{X}}$ and the new error covariance as,

$$\mathrm{X}_k = \hat{\mathrm{X}}_k + \mathrm{K}y \tag{3.152}$$

$$\mathrm{P}_k = \mathrm{P}_k - \mathrm{K}\mathrm{P}_{\mathbb{Z}}\mathrm{K}^T \tag{3.153}$$

The Figure 3.11 shows the simulated application of the *Unscented Kalman Filter* to our damped nonlinear harmonic oscillator problem for joint state and parameter estimation.

The performance of both, the Extended Kalman Filter and the Unscented Kalman Filter is compared by calculating the respective Root Mean Square Error (RMSE) of the estimated states and parameters of the system. It measures how much error is there between the estimated and actual values of the system.

The results are summarized in Table 3.2.7.

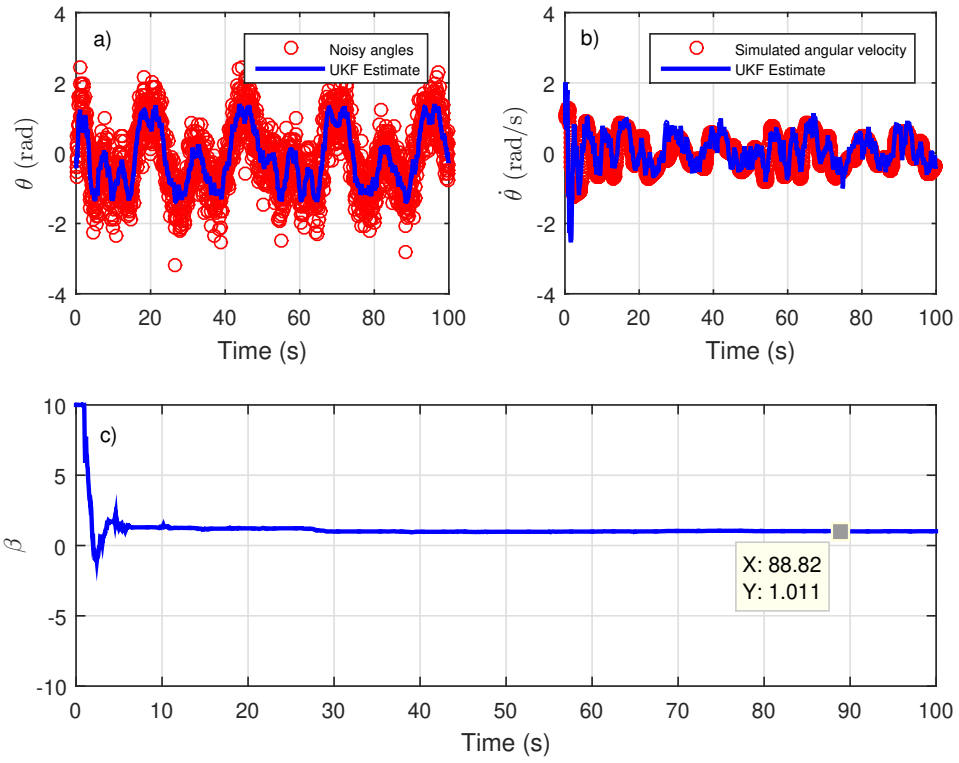

FIGURE 3.11: *The joint state and parameter estimation of a Duffing oscillator system using the Unscented Kalman Filter.*

	Nonlinear coefficient (β)	
	EKF	UKF
True value	1	
Estimated	0.985	1.011
RMSE	0.149	0.105

TABLE 3.5: The performance comparison of Extended Kalman Filter and Unscented Kalman Filter using the Root Mean Square Error (RMSE) values for the joint state and parameter estimation of the Duffing oscillator.

3.3 Wilberforce pendulum

The Wilberforce pendulum is commonly used as an important qualitative demonstration in introductory mechanics [47]. It is shown in Figure 3.12. It is a vivid demonstration of the interaction between two coupled harmonic oscillators: the longitudinal stretching and the torsional twisting of a spiral spring attached to a mass. It also illustrates the phenomenon of beats which arises because of the intermixing of two normal modes. You can find an interesting experiment on exploring the dynamics of the Wilberforce pendulum using a high speed digital camera here [1]. Furthermore, it is a mechanical demonstraiton of the phenomena of avoided crossings. Lionel Robert Wilberforce in 1896, proposed the use of a loaded spiral spring to determine the Young's modulus of the spring material and also identified the potential of this device to determine the transfer of energy between two coupled modes of a harmonic oscillator.

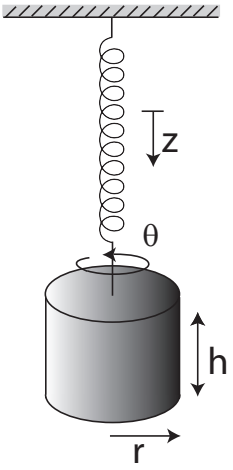

FIGURE 3.12: A Wilberforce pendulum showing oscillations along z and rotations through an angle θ. The radius and height of the cylinder are r and h respectively.

3.3.1 Simulating the system

A spiral spring with linear spring constant, k and torsional spring constant, δ is suspended from a fixed support. A metal cylinder with mass m, radius r and height h, attached to the free end of the spring. This is shown in Figure 3.12. The coordinate system is defined so that the z direction is along the axis of the spring

[1] https://www.physlab.org/experiment/wilberforce-pendulum/

with positive z going downwards, and the θ direction corresponds to rotation around the axis of symmetry of the system.

The equations of motion of this mass-spring system are given by,

$$m\left(\frac{d^2z}{dt^2}\right) + kz + \varepsilon\theta/2 \;=\; 0 \tag{3.154}$$

$$I\left(\frac{d^2\theta}{dt^2}\right) + \delta\theta + \varepsilon z/2 \;=\; 0 \tag{3.155}$$

where $\varepsilon z\theta/2$ is the coupling between the translational and torsional motion. The ε is a nonlinear parameter measuring the strength of the coupling, δ is the torsional spring constant and I is the moment of inertia $I = \frac{mr^2}{2}$. If $\varepsilon = 0$, the physical device would correspond simply two independent, uncoupled harmonic oscillators.

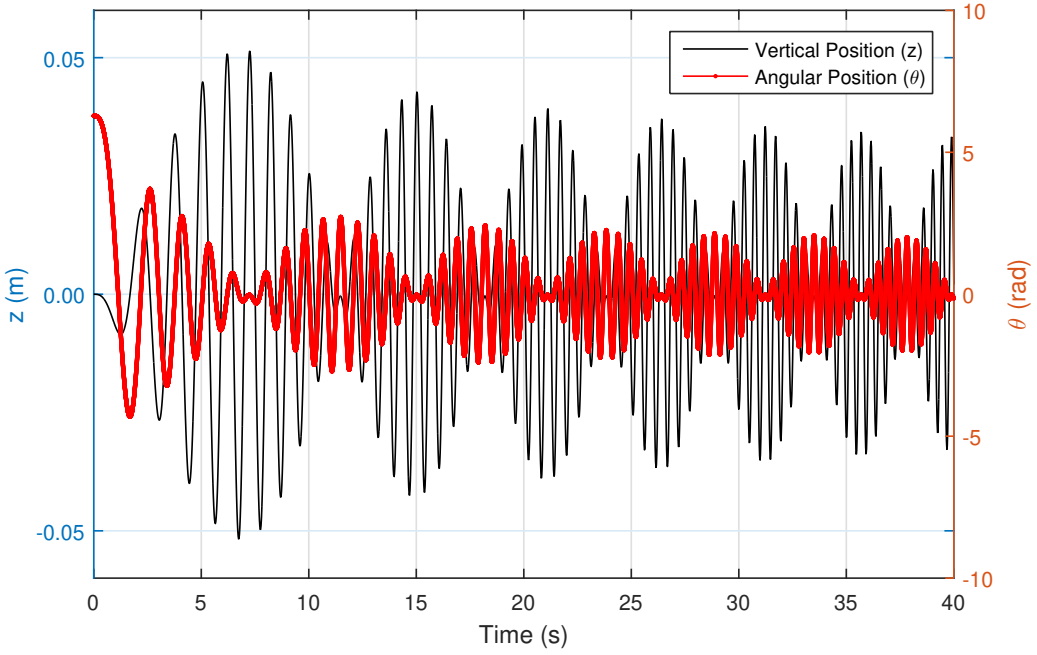

FIGURE 3.13: *The simulaiton of the Wilberforce pendulum's kinematics.*

Figure 3.13 shows the simulation of Wilberforce pendulum. The plot is a super imposition of the vertical and angular movement of the pendulum that clearly transit between each other. The vertical oscillations of the pendulum are represented by the axis on the left and the rotational movement of the pendulum around its own axis is represented by θ on the right axis. The simulation parameters were defined

as, $\omega^2 = 2.314\ rad^2/s^2$, $\epsilon = 9.27 \times 10^{-3}$ N, $m = 0.4905$ kg, $I = 1.39 \times 10^{-4}$ kgm^2, $\theta_0 = 2\pi$, $z_0 = 0$.

3.3.2 Deriving the state equations

For the state estimation of the Wilberforce pendulum our state equations are as follows:

$$\mathrm{x} = \begin{bmatrix} \mathrm{x}_1 & \mathrm{x}_2 & \mathrm{x}_3 & \mathrm{x}_4 \end{bmatrix}^T \tag{3.156}$$

where, $\mathrm{x}_1 = z$, $\mathrm{x}_2 = \dot{z}$, $\mathrm{x}_3 = \theta$, $\mathrm{x}_4 = \dot{\theta}$

$$\begin{aligned} \dot{\mathrm{x}}_1 &= \mathrm{x}_2 \\ \dot{\mathrm{x}}_2 &= \ddot{z} = -\frac{k}{m}\mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3 \end{aligned} \tag{3.157}$$

$$\begin{aligned} \dot{\mathrm{x}}_3 &= \mathrm{x}_4 \\ \dot{\mathrm{x}}_4 &= \ddot{\theta} = -\frac{\delta}{I}\mathrm{x}_3 - \frac{1}{2m}\epsilon \mathrm{x}_1 \end{aligned} \tag{3.158}$$

These state equations are modeled as discussed previously in Chapter 3. We also take $\omega_z^2 = \frac{k}{m}$ and $\omega_\theta^2 = \frac{\delta}{I}$

3.3.2.1 Nonlinear observability test for the Wilberforce pendulum

Before we begin applying our filters to the problem we will again perform the test for observability of this nonlinear system.

For the Wilberforce pendulum we know from Equation (3.157) and (3.158) that,

$$f(\mathrm{x}) = \begin{bmatrix} \mathrm{x}_2 \\ -\omega_z^2 \mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3 \\ \mathrm{x}_4 \\ -\omega_\theta^2 \mathrm{x}_3 - \frac{1}{2m}\epsilon \mathrm{x}_1 \end{bmatrix} \tag{3.159}$$

We use a linear position sensor to estimate the other states. So, the measurement equation is,

$$y(\mathrm{x}) = \mathrm{H}(\mathrm{x}) = \mathrm{x}_1 \tag{3.160}$$

Taking Lie derivatives of this measurement function,

$$\begin{aligned}
\pounds_f^0(y(\mathrm{x})) &= y(\mathrm{x}) = \mathrm{x}_1 && (3.161)\\
\pounds_f^1(y(\mathrm{x})) &= \frac{\partial y}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\
&= \begin{pmatrix} \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_1} & \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_2} & \frac{\partial \mathrm{x}_1}{\partial \mathrm{x}_3} \end{pmatrix} \cdot f(\mathrm{x}) \\
&= \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\omega_z^2 \mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3 \\ \mathrm{x}_4 \\ -\omega_\theta^2 \mathrm{x}_3 - \frac{1}{2m}\epsilon \mathrm{x}_1 \end{pmatrix} = \mathrm{x}_2 && (3.162)\\
\pounds_f^2(y(\mathrm{x})) &= \frac{\partial \pounds_f^1}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\
&= \begin{pmatrix} \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_1} & \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_2} & \frac{\partial \mathrm{x}_2}{\partial \mathrm{x}_3} \end{pmatrix} \cdot f(\mathrm{x}) \\
&= \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\omega_z^2 \mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3 \\ \mathrm{x}_4 \\ -\omega_\theta^2 \mathrm{x}_3 - \frac{1}{2m}\epsilon \mathrm{x}_1 \end{pmatrix} \\
&= -\omega_z^2 \mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3 && (3.163)\\
\pounds_f^3(y(\mathrm{x})) &= \frac{\partial \pounds_f^2}{\partial \mathrm{x}} \cdot f(\mathrm{x}) \\
&= \begin{pmatrix} \frac{\partial(-\omega_z^2 \mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3)}{\partial \mathrm{x}_1} & \cdots & \frac{\partial(-\omega_z^2 \mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3)}{\partial \mathrm{x}_4} \end{pmatrix} \cdot f(\mathrm{x}) \\
&= \begin{pmatrix} -\omega_z^2 & 0 & \frac{1}{2m}\epsilon & 0 \end{pmatrix} \begin{pmatrix} \mathrm{x}_2 \\ -\omega_z^2 \mathrm{x}_1 - \frac{1}{2m}\epsilon \mathrm{x}_3 \\ \mathrm{x}_4 \\ -\omega_\theta^2 \mathrm{x}_3 - \frac{1}{2m}\epsilon \mathrm{x}_1 \end{pmatrix} \\
&= -\mathrm{x}_2 \omega_z^2 + \frac{1}{2m}\epsilon \mathrm{x}_4 && (3.164)
\end{aligned}$$

The resulting mapping matrix ϕ is,

$$\phi = \begin{bmatrix} \pounds_f^0 \\ \pounds_f^1 \\ \pounds_f^2 \\ \pounds_f^3 \end{bmatrix} = \begin{bmatrix} \mathrm{X}_1 \\ \mathrm{X}_2 \\ -\omega_z^2 \mathrm{X}_1 - \frac{1}{2m}\epsilon \mathrm{X}_3 \\ -\omega_z^2 \mathrm{X}_2 + \frac{1}{2m}\epsilon \mathrm{X}_4 \end{bmatrix}. \tag{3.165}$$

Taking partial derivative of this mapping matrix with respect to the state variables yields the following observability test matrix,

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\omega^2 & 0 & \frac{1}{2m}\epsilon & 0 \\ 0 & -\omega_z^2 & 0 & \frac{1}{2m}\epsilon \end{bmatrix}. \tag{3.166}$$

The rank of this observability matrix is equal to the dimensions of the system i.e. $n = 4$ meaning we can use a single position sensor to estimate the states of linear velocity, angular position and angular velocity of a coupled Wilberforce pendulum.

Now, let's apply the Kalman Filters to estimate the states.

3.3.3 State estimation of Wilberforce pendulum using Extended Kalman Filter

The nonlinear equations for the state propagation of the Wilberforce pendulum are:

$$\begin{aligned} (\mathrm{X}_1)_k &= (\hat{\mathrm{X}}_1)_{k-1} + (\hat{\mathrm{X}}_2)_{k-1}\Delta t \\ (\mathrm{X}_2)_k &= (\hat{\mathrm{X}}_2)_{k-1} + (-\omega_z^2(\hat{\mathrm{X}}_1)_{k-1} - \frac{1}{2m}\epsilon(\hat{\mathrm{X}}_3)_{k-1})\Delta t \\ (\mathrm{X}_3)_k &= (\hat{\mathrm{X}}_3)_{k-1} + (\hat{\mathrm{X}}_4)_{k-1}\Delta t \\ (\mathrm{X}_4)_k &= (\hat{\mathrm{X}}_4)_{k-1} + (-\omega_\theta^2(\hat{\mathrm{X}}_3)_{k-1} - \frac{1}{2m}\epsilon(\hat{\mathrm{X}}_1)_{k-1})\Delta t \end{aligned} \tag{3.167}$$

For this case we assume our initial estimates of the state to be at around 0.1 m and 0.1 m/s in the vertical oscillation while 0.1 rad and 0.1 rad/s in the rotational movement of the pendulum respectively.

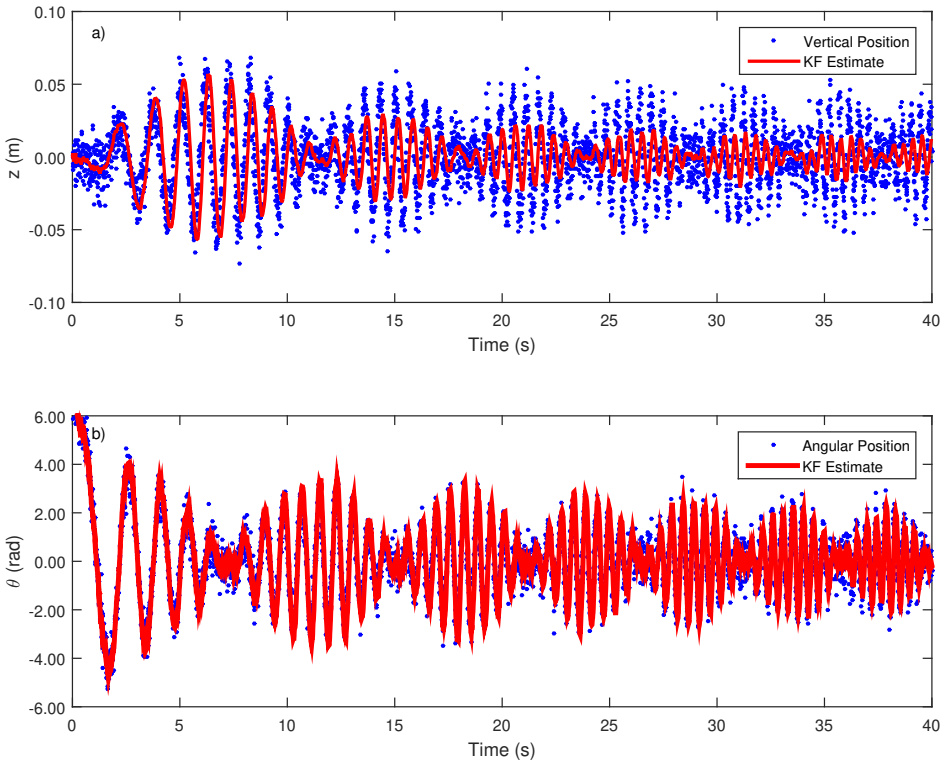

FIGURE 3.14: *Applying the Extended Kalman Filter to filter the Wilberforce pendulum's vertical and angular movement.*

$$\mathrm{P}_o = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{3.168}$$

The next most crucial step before moving on to the filter algorithm is the design of Q matrix. We consider the case of constant acceleration to model the process covariance matrix for both cases. Using the piecewise-noise model shown in Equation (2.54) we derive the following Q matrix:

$$\mathrm{Q} = \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} & 0 & 0 \\ \frac{\Delta t^3}{2} & \Delta t^2 & 0 & 0 \\ 0 & 0 & \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ 0 & 0 & \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \tag{3.169}$$

Using the state equations we design the system dynamic matrix A,

$$\mathrm{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\omega_z^2 & 1 & \frac{-1}{2m}\epsilon & 0 \\ 0 & 0 & 0 & 1 \\ \frac{-1}{2m}\epsilon & 0 & -\omega_\theta^2 & 1 \end{bmatrix} \tag{3.170}$$

Using (2.17) we derive the state transition matrix F of upto first order in Δt,

$$\mathrm{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ -\omega_z^2 \Delta t & 1+\Delta t & \frac{-1}{2m}\epsilon & 0 \\ 0 & 0 & 1 & \Delta t \\ \frac{-1}{2m}\epsilon & 0 & -\omega_\theta^2 \Delta t & 1+\Delta t \end{bmatrix} \tag{3.171}$$

Figure 3.14 exhibits the simulated application of the Kalman filter to the Wilberforce pendulum. (a) exhibits the filtered transnational motion in z direction (b) shows the estimated angular position simulated Wilberforce Pendulum that is infested with $\sigma^2 = 0.001$ m and $\sigma^2 = 0.01$ rad noise respectively.

3.3.4 Estimating the modes and coupling constant of the Wilberforce pendulum

Using the Extended Kalman Filter we try to estimate the parameters of the Wilberforce pendulum, ω_z^2, ω_θ^2 and ε. We first build the augmented state space model.

$$\mathrm{x} = \begin{bmatrix} \mathrm{x}_1 & \mathrm{x}_2 & \mathrm{x}_3 & \mathrm{x}_4 & \mathrm{x}_5 & \mathrm{x}_6 & \mathrm{x}_7 \end{bmatrix}^T \tag{3.172}$$

where, $\mathrm{x}_1 = z$, $\mathrm{x}_2 = \dot{z}$, $\mathrm{x}_3 = \theta$, $\mathrm{x}_4 = \dot{\theta}$, $\mathrm{x}_5 = \omega_z^2$, $\mathrm{x}_6 = \omega_\theta^2$, $\mathrm{x}_7 = \varepsilon$.

Our state equations in this case takes the following form,

$$f(\mathrm{x}) = \begin{bmatrix} \mathrm{x}_2 \\ -\mathrm{x}_5\mathrm{x}_1 - \frac{1}{2m}\mathrm{x}_7\mathrm{x}_3 \\ \mathrm{x}_4 \\ -\mathrm{x}_6\mathrm{x}_3 - \frac{1}{2m}\mathrm{x}_7\mathrm{x}_1 \\ \mathrm{x}_5 \\ \mathrm{x}_6 \\ \mathrm{x}_7 \end{bmatrix} \tag{3.173}$$

We linearize this system by taking Jacobian of the function $f(\mathrm{x})$ with respect to the state variables. The resulting matrix is as follows,

$$\Phi = \frac{\partial \mathrm{A}}{\partial \mathrm{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -\mathrm{x}_5 & 0 & \frac{-1}{2m}\mathrm{x}_7 & 0 & -\mathrm{x}_1 & 0 & \frac{1}{2m}\mathrm{x}_3 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \frac{-\mathrm{x}_7}{2I} & 0 & -\mathrm{x}_6 & 0 & 0 & -\mathrm{x}_3 & \frac{-1}{2I}\mathrm{x}_1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.174}$$

where, the resulting Φ matrix is the linearized system dynamic matrix. We discretize this system by taking the exponential which gives,

$$\mathrm{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 & 0 \\ (-\mathrm{x}_5)\Delta t & 1 & (\frac{-1}{2m}\mathrm{x}_7)\Delta t & 0 & (-\mathrm{x}_1)\Delta t & 0 & (\frac{1}{2m}\mathrm{x}_3)\Delta t \\ 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ (\frac{-\mathrm{x}_7}{2I})\Delta t & 0 & -(\mathrm{x}_6)\Delta t & 1 & 0 & (-\mathrm{x}_3)\Delta t & (\frac{-1}{2I}\mathrm{x}_1)\Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.175}$$

The state transition matrix F will be used to propagate the state error covariance matrix in the filter's algorithm.

We specify the initial error covariance $\hat{\mathrm{P}}_0$. For this case we assume that we are pretty sure about our initial estimate of the state and so we define the errors in our states to be at around 0.1 m and 0.1 m/s in the vertical oscillation while 0.1 rad and 0.1 rad/s in the torsional movement of the pendulum.

$$\hat{\mathrm{P}}_o = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{3.176}$$

Following set of equations are augmented to include the unknown parameters $\mathrm{x}_5 = \omega_z^2$, $\mathrm{x}_6 = \omega_\theta^2$ and $\mathrm{x}_7 = \varepsilon$. We propagate these through Euler integration in the *a priori* prediction step,

$$(\mathrm{x}_1)_k = (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1}\Delta t$$

$$(\mathrm{x}_2)_k = (\hat{\mathrm{x}}_2)_{k-1} + (-\hat{\mathrm{x}}_5(\hat{\mathrm{x}}_1)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_3)_{k-1})\Delta t \tag{3.177}$$

$$(\mathrm{x}_3)_k = (\hat{\mathrm{x}}_3)_{k-1} + (\hat{\mathrm{x}}_4)_{k-1}\Delta t$$

$$(\mathrm{x}_4)_k = (\hat{\mathrm{x}}_4)_{k-1} + (-\omega_\theta^2(\hat{\mathrm{x}}_3)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_1)_{k-1}) \tag{3.178}$$

$$(\mathrm{x}_5)_k = (\hat{\mathrm{x}}_5)_{k-1} \tag{3.179}$$

$$(\mathrm{x}_6)_k = (\hat{\mathrm{x}}_6)_{k-1} \tag{3.180}$$

$$(\mathrm{x}_7)_k = (\hat{\mathrm{x}}_7)_{k-1} \tag{3.181}$$

Using the piecewise-noise model shown in Equation (2.54) we derive the following Q matrix:

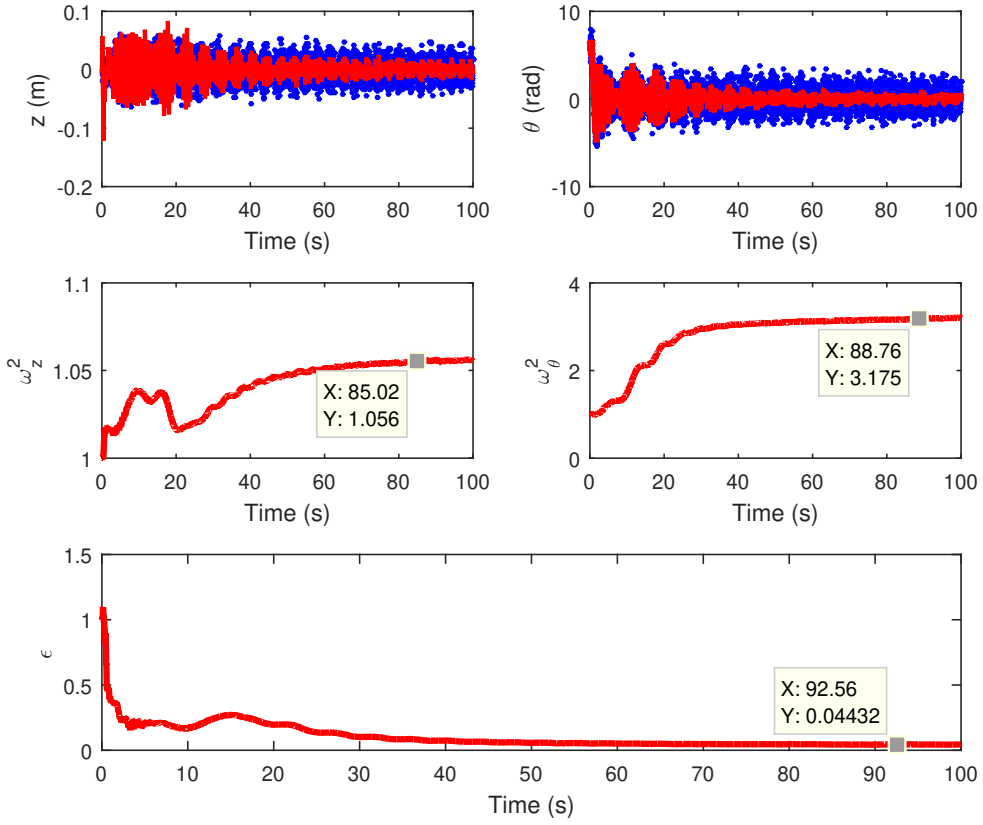

FIGURE 3.15: *Estimating the normal modes and the coupling constant of a Wilberforce pendulum using Extended Kalman Filter. The true values of the parameters are* $\omega_z^2 = 5.35$, $\omega_\theta^2 = 5.35$, *and* $\varepsilon = 9.29 \times 10^{-3}$.

$$\mathrm{Q} = \begin{bmatrix} \frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\ \frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\ \frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\ \frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \sigma_v^2 \tag{3.182}$$

The variance σ_v^2 is inflated with $\sigma_v^2 = 5.5^2$. As discussed before, this matrix acts as a tuning parameter and greatly impacts the estimation abilities of the Kalman filter.

The Figure 3.15 exhibits our application of the Extended Kalman Filter to estimate the normal modes and the coupling constant of the Wilberforce pendulum system.

The true values of the system are $\omega_z = 2.314$ rad/s, $\omega_\theta = 2.314$ rad/s and $\epsilon = 9.27 \times 10^{-3}$ N.

In this problem, we have also used an angle sensor in the simulation. Due to this inclusion the measurement matrix H takes the following form,

$$\mathrm{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.183}$$

This extracts the state of vertically oscillating position and the angular rotation into the measurement space. In the measurement space we have a noisy vertical position sensor infested with $\sigma_v^2 = 0.01$ m and a noisy rotatory sensor with $\sigma_v^2 = 0.8$ rad. Figure 3.15 exhibits this application. The normal mode along z direction is estimated to be $\omega_z^2 = 1.056\ m^2/s^2$, along θ to be $\omega_\theta^2 = 3.175\ rad^2/s^2$ and the coupling constant to be $\varepsilon = 0.044$ N by the filter.

3.3.5 Applying Unscented Kalman Filter

We can also estimate the Wilberforce parameters using the Unscented Kalman Filter.

Using the same state dynamic Equations (3.157) we first define our nonlinear function,

$$(\mathrm{x}_1)_k = (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1}\Delta t$$

$$(\mathrm{x}_2)_k = (\hat{\mathrm{x}}_2)_{k-1} + (-\hat{\mathrm{x}}_5(\hat{\mathrm{x}}_1)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_3)_{k-1})\Delta t \tag{3.184}$$

$$(\mathrm{x}_3)_k = (\hat{\mathrm{x}}_3)_{k-1} + (\hat{\mathrm{x}}_4)_{k-1}\Delta t$$

$$(\mathrm{x}_4)_k = (\hat{\mathrm{x}}_4)_{k-1} + (-\omega_\theta^2(\hat{\mathrm{x}}_3)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_1)_{k-1}) \tag{3.185}$$

$$(\mathrm{x}_5)_k = (\hat{\mathrm{x}}_5)_{k-1} \tag{3.186}$$

$$(\mathrm{x}_6)_k = (\hat{\mathrm{x}}_6)_{k-1} \tag{3.187}$$

$$(\mathrm{x}_7)_k = (\hat{\mathrm{x}}_7)_{k-1} \tag{3.188}$$

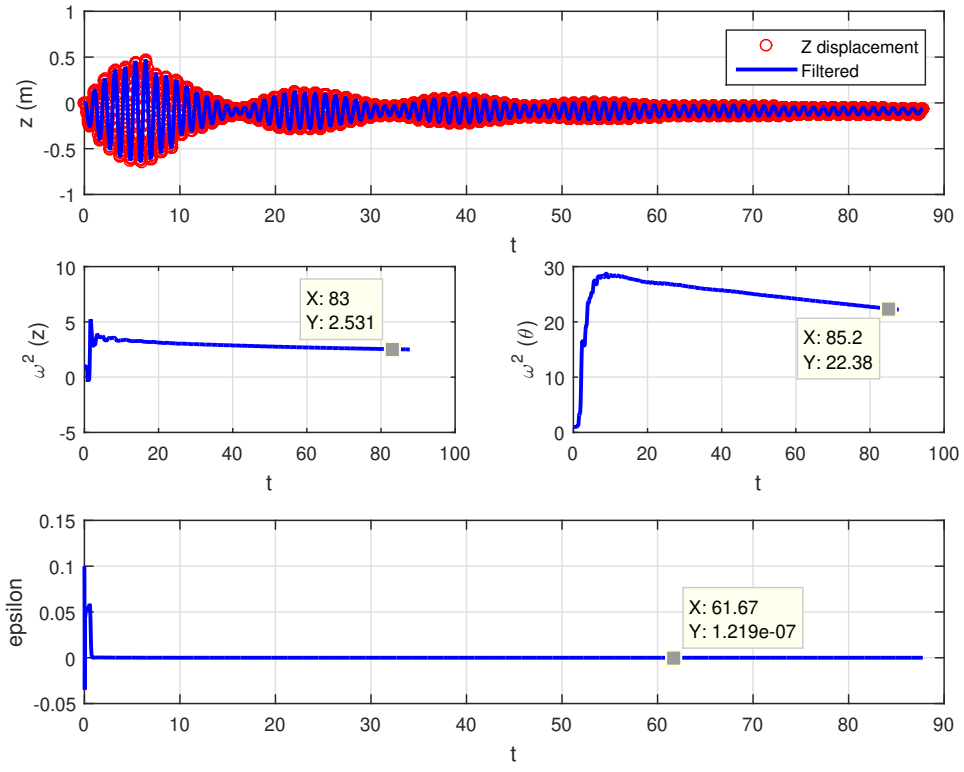

FIGURE 3.16: *Parameter estimation using Unscented Kalman Filter on Wilberforce Pendulum. The true values of the parameters are* $\omega_z^2 = 5.35$, $\omega_\theta^2 = 5.35$, *and* $\varepsilon = 9.29 \times 10^{-3}$.

We use the same Q matrix (3.189) as the piecewise noise model for our process error covariance matrix.

$$\mathrm{Q} = \begin{bmatrix} \frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\ \frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\ \frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\ \frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \sigma_v^2 \tag{3.189}$$

Furthermore, in both cases we have assumed to be using two noisy sensors. One sensor is to track the position of the oscillator in the vertical z direction in meters while the other sensor measures the angular position θ in radians.

Next, we have to initialize the belief in our system. We build our initial state error covariance matrix P_0 as follows:

$$P_0 = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{3.190}$$

For parameter estimation our process noise covariance matrix is also augmented. As we are trying to estimate a constant parameter we assume that there is no processing uncertainty in the parameter estimation.

The Figure 3.16 shows the results of using the *Unscented Kalman Filter* algorithm to filter a simulated Wilberforce Pendulum that is infested with $\sigma^2 = 0.001$ m and $\sigma^2 = 0.01$ rad noise in vertical position z and angular measurement respectively.

Chapter 4

Experimental work

The numerical simulations in previous chapter highlight the applicability of Kalman filter to some basic problems in physics. Now, we would like to explore it in the realms of hands-on experiments with uncertainties and noisy sensors. In our first experiment, we heat and cool a Peltier healter using a PID controller circuit and Kalman filter the temperature measurements acquired from a temperature sensor attached to it. The goal is to observe the variations of the filter against different values of process noise.

Similarly, in our second experiment, we simply remove noise from a series of temperature measurements of a liquid being heated over a hot plate and see how the performance of Kalman filter varies with the change in process noise. Next, using a force sensor we measure the acceleration of a damped harmonic oscillator. This acceleration is integrated to obtain the velocity and position. However, both integrated quantities, velocity and position, contain the typical integration drift. We use Kalman filter to filter out this drift and obtain true values of the velocity and position. Last, using parameter estimation, we determine the normal resonant modes and the coupling constant of a Wilberforce pendulum.

4.1 Changing temperature in steps

In our first experiment, we measure temperature of a Peltier heater that is heated and cooled using a PID controller circuit. A set point temperature is given to the circuit which regulates the voltage and flow of current to a Peltier heater. A Peltier

heater works on the "Peltier effect" which is created by temperature difference that is caused by the heat transfer between junctions of two electrical points. A voltage is supplied across the junctions to create an electric current. When the current flows through the junctions, heat is removed at one junction and cooling takes place, while heating takes effect at the other junction. Using a temperature sensor (*LM35*) we acquire the temperature readings into the computer using National Instrument's data acquisition card (*NI-PCI*-6221). Then we Kalman filter the data and verify the effects of process noise Q on the performance of the filter.

4.1.1 Apparatus and schematic diagram

The apparatus consists of two DC Power supplies (V & A Instruments) that supply negative and positive voltage to the Peltier heater. The supplies are connected to the control box which houses the PID controller circuit. The circuit is built using an $LM324$ operational amplifier. Depending upon the set point and gain, the PID circuit controls the amount of current supplied to the Peltier heater. The set-point and gain values are set using the resistors $RV3$ and $RV1$ respectively, as shown in Figure 4.2.

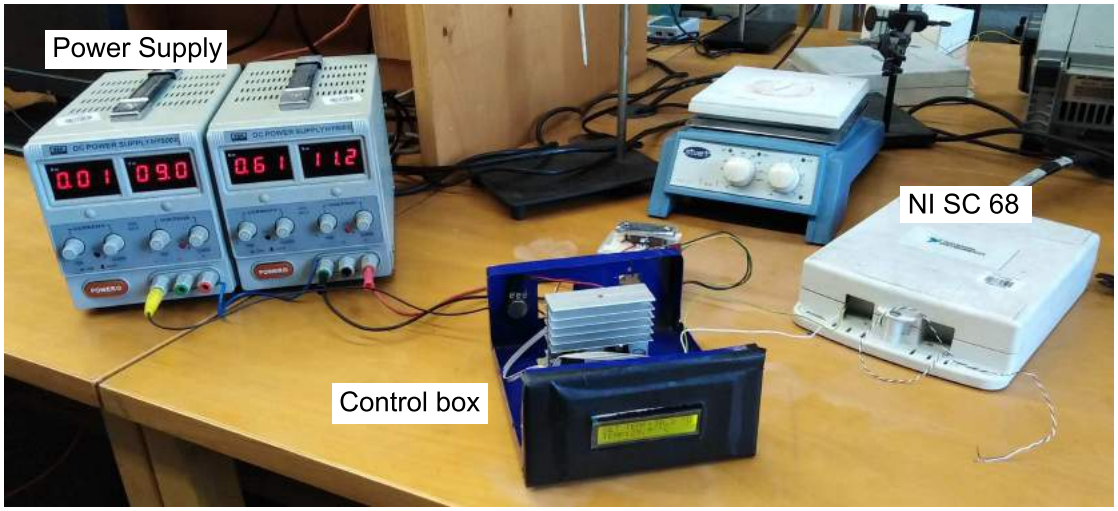

FIGURE 4.1: *Picture of the complete setup. The power supplies to provide voltage and current to the Peltier heater. The control box houses the PID circuit for temperature control. The NI SC 68 is National Instrument's external module connected to the PCI 6221 data acquisition card inside the computer.*

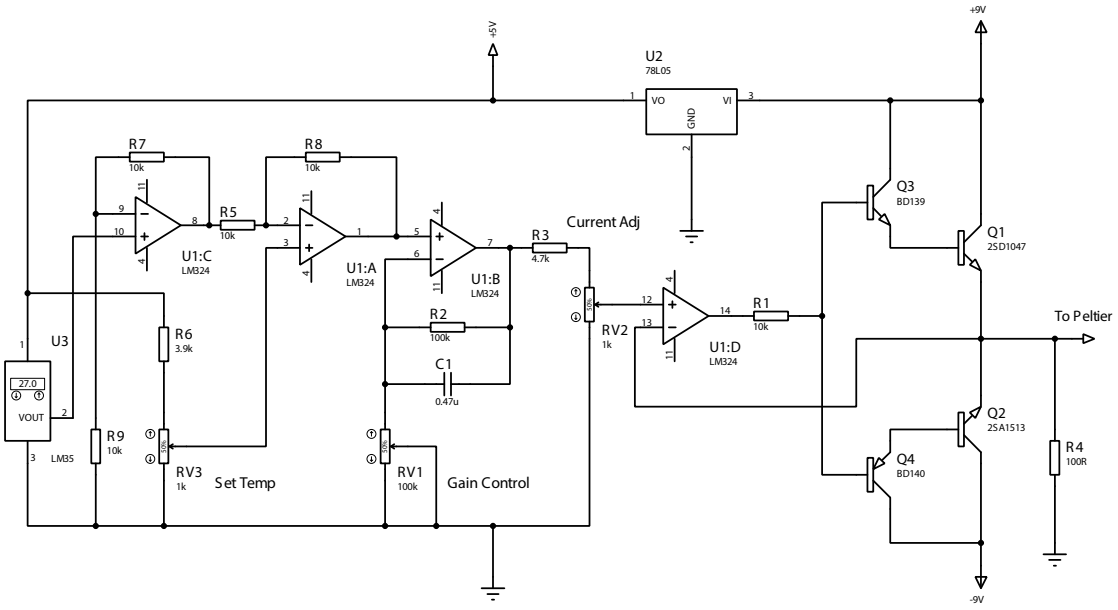

FIGURE 4.2: *The schematic diagram of the PID based control circuit.*

4.1.2 Applying the Kalman filter

This is an example of a one dimensional *Scalar Kalman Filter (SKF)*. The process model in this case is a constant state.

Hence, our time update equations are,

$$\mathrm{x}_k = \mathrm{F}\hat{\mathrm{x}}_{k-1} = \mathrm{x}_{k-1} \tag{4.1}$$

$$\mathrm{P}_k = \hat{\mathrm{P}}_{k-1} + \mathrm{Q} \tag{4.2}$$

where, $\mathrm{F} = 1$, $\mathrm{P} = 10$ is the initial error variance in the state variable while Q is the process variance of predicting the state at current time step.

Now, we want to see how the filter behaves when we change the values of Q as any change would effect the Kalman gain in,

$$\mathrm{K}_k = \mathrm{P}_k(\mathrm{P}_k + \mathrm{R})^{-1}. \tag{4.3}$$

Here, P is the uncertainty in the state prediction while R is the measurement noise. As discussed in previous chapters, increasing the value of Q also increases the value of P, resulting in the increase of Kalman gain. As the error in the process

is increased, the filter prioritizes the values of measurements for next iterations. However, as we reduce the value of Q, the Kalman gain decreases and the filter tends to diverge ignoring measured temperature values altogether.

The resulting state estimate $\hat{x}_k$ and its variation with regards to the altering process noise variance Q is shown in Figure 4.3.

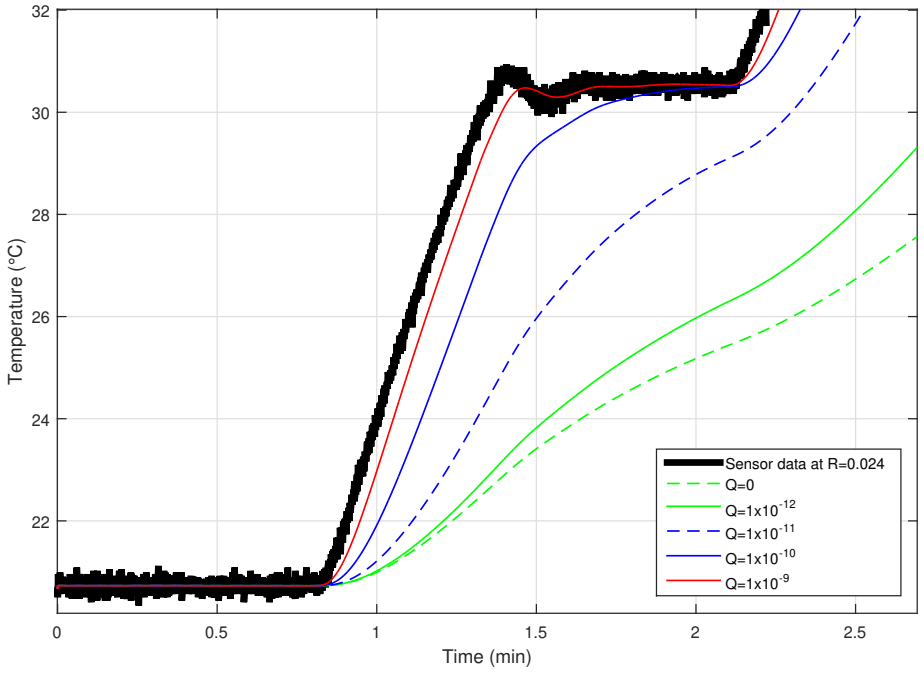

FIGURE 4.3: *Response of piecewise process noise model to the temperature profile.*

In Figure 4.3, we initially set the temperature value to 10°C for 1 min and then increase the set point to 30°C for 1.5 min. The PID circuit regulates the current to increase the temperature of Peltier to the set point temperatures. For a value $Q = 1 \times 10^{-12}$ which is significantly lower than the measurement noise of 0.024°C, the filter prominently avoids the sensor data. At $Q = 0$ we already have the value of $P = 10$ set as the initial error variance. So, the filter initially iterates with this value shows the largest divergence. If the value of P was also set to zero then we would see no change in temperature estimate and we would see a constant value from the initialization of the state variable.

Now, as we increase the uncertainty in the process model the filter adjusts to the measurement values and eventually starts following the exact sensor data iterating the fact that Q acts as a tuning parameter. Finally, the value $Q = 1 \times 10^{-9}$ seems

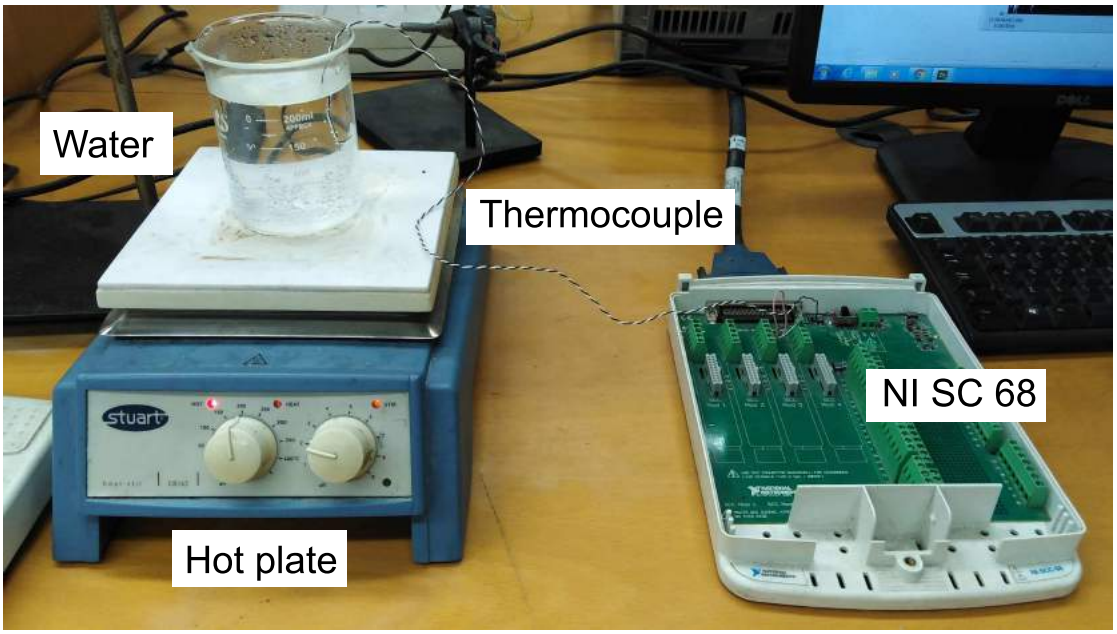

FIGURE 4.4: *Apparatus for water heating experiment consisted of a hot plate (Stuart CB − 168), a 200 ml beeker full of water, a k-type thermocouple and the NI SC − 68 is National Instrument's external module connected to the PCI 6221 data acquisition card inside the computer.*

optimum but exhibits a lag in the filter estimates throughout the regions of change in the temperature. This is because there is no process model provided to the filter to fit this phenomena. The only tuning parameter is the uncertainty comparison of the measurement noise R and a process noise Q.

4.2 Continuous increase of fluid temperature

As seen in previous example, when there is no process model available for a phenomena we see a lag in the performance of the Kalman filter. To observe this more closely we take a simple example of heating a beaker full of water.

Although we could model the heating of water using the mathematical model describing the heat capacity of water but for this example we consider there is no process model. The process model for the filter thus remains the same as,

$$\mathrm{x}_k = \mathrm{F}\hat{\mathrm{x}}_{k-1} \tag{4.4}$$

$$\mathrm{P}_k = \hat{\mathrm{P}}_{k-1} + \mathrm{Q} \tag{4.5}$$

where, F = 1 and P = 10°C while Q is the process variance that is to be varied.

4.2.1 Methodology

We take a 200 ml beaker full of water and heat it on a hot plate. The temperature of water starts to increase. Using the same data acquisition setup as shown in previous experiment we log the change in temperature over time. The complete setup is shown in Figure 4.4. The apparatus consisted of a hot plate (Stuart Equipment) with a magnetic stirrer, a 200 ml beaker with some water, a k-type thermocouple and National Instruments data acquisition card (NI SC 68) integrated with (NI PCI 6221) inside the computer. Using the hot plate we heat the water inside the beaker to around 90°C. Using the magnetic stirrer we keep the temperature in the liquid equally distributed.

In this case, the measurement uncertainty from the thermocouple readings is around 0.01°C. For measure this uncertainty by taking a series of measurements at a constant temperature and calculating its variance. Because the state will not change over time so there is no control input that will be altering it. However, there will be noise in our system because of the measurement sensor.

4.2.2 Effect of varying process noise

Presuming a small process variance, we let Q = 0.005. The blue shaded region is the output of the filter that is largely following the noisy data from the sensor. This means that we have to reduce the uncertainty value of Q to optimize the performance of the filter. We change the value to around $Q = 1 \times 10^{-6}$. The filter beautifully filters out the noise in the sensor. There is nothing stopping us from going further. We reduce the value of Q to zero. However, the filter clearly shows a huge lag from the actual values of the sensor. This lag is due to the fact that we did not have a process model for this system and so we only used the uncertainty in the model to tune the filter's performance.

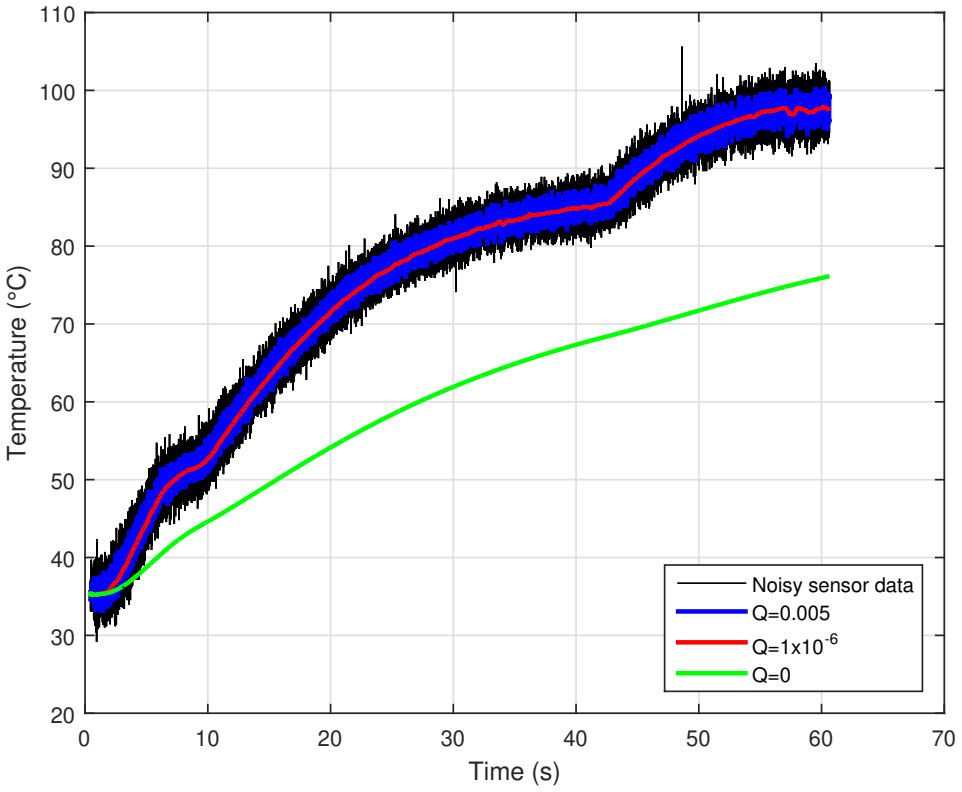

FIGURE 4.5: *The application of Kalman Filter to temperature measurements obtained on a beaker containing water, using a thermocouple.*

4.3 Damped harmonic oscillator

We have already created a simulation model of the mass spring damper system in Chapter 3. Now, we use Kalman Filter in two folds to filter and estimate the states of a mass spring damper experimental setup as well as estimate the parameters of the system. In particular, we try to estimate the spring constant k and the damping coefficient b. The actual values of both of these parameters have been independently measured to verify the estimation.

The setup consists of a mass of 100 g that is hanging with a *Force Sensor* which measures the downward force exerted by it. The mass is hung with a spring with a spring constant $k = 5.5$ N/m. Water in a beaker of 2000 ml volume is used as a damping medium. The force sensor is attached to the computer using *Vernier's Lab Quest* and a software *Logger Lite* is used to acquire data in the computer as shown in Figure 4.6.

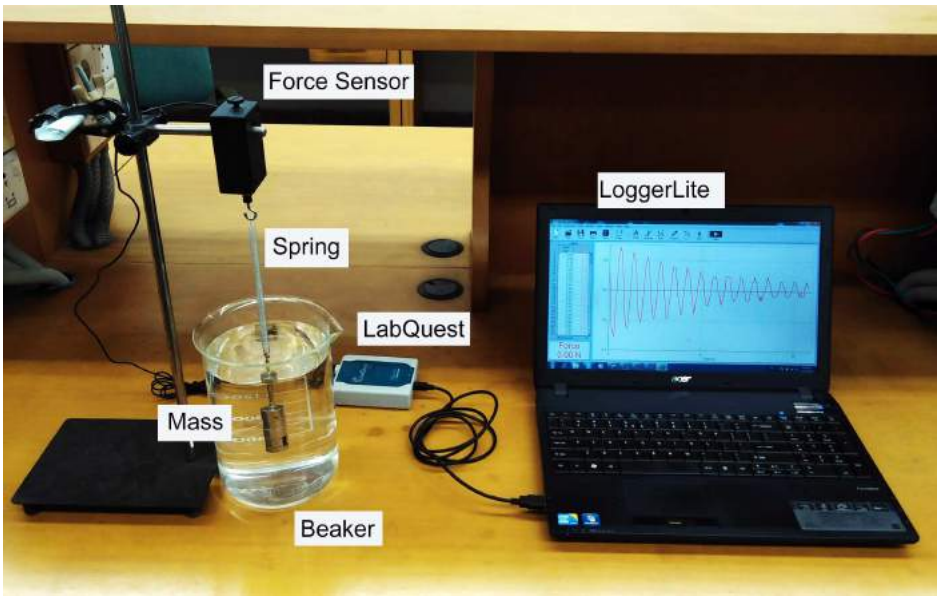

FIGURE 4.6: *Experimental setup for mass spring damper joint state and parameter estimation.*

From Newton's second law of motion,

$$F = ma \tag{4.6}$$

The mass is made to oscillate vertically inside the water. The force sensor records the amount of force being exerted the oscillating mass and reports it to the *Logger Lite* software. From this data we extract values of the acceleration using Equation (4.6). The sampling rate of was set to be 0.02 seconds/sample. We use the MATLAB's *cumtrapz* command to find the cumulative trapezoidal numerical integration of this data to obtain velocity and position of the oscillator respectively.

Due to the integration, the position is infested by the integration drift and noise ingrown due to the sensor's sensitivity.

In the first attempt we try to filter only the drift in position and velocity using the known parameters of this system through a Linear Kalman Filter (LKF).

4.3.1 Correcting the drift

The state dynamic equations for the filter are modeled as discussed previously in Chapter 3. For our first case, we have a known spring constant $k = 5.5$ N/m and

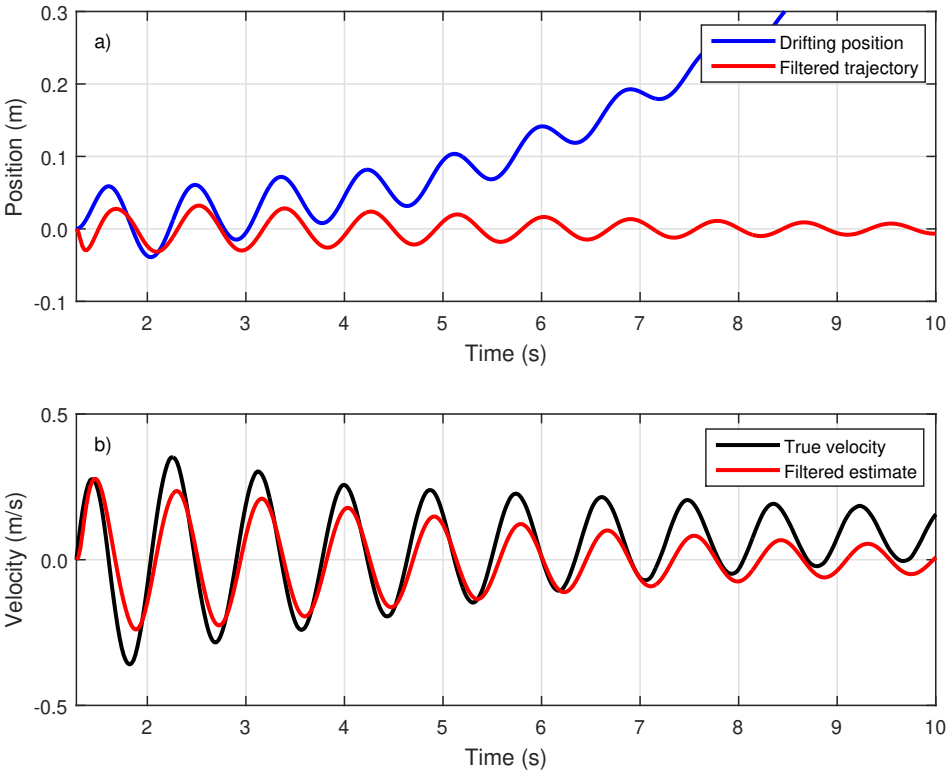

FIGURE 4.7: *Drift removal in position and velocity of the Damped Harmonic Oscillator system using a linear Kalman Filter.*

damping coefficient $b = 0.15$ Ns/m.

The state transition matrix is thus,

$$\mathrm{F} = \begin{bmatrix} 1 & \Delta t \\ (-k/m)\Delta t & (1 - b/m)\Delta t \end{bmatrix} \tag{4.7}$$

To improve the performance of filter we consider using the position and velocity obtained from the integration of acceleration.

For this, the measurement matrix is,

$$\mathrm{H} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.8}$$

The Kalman filter time and measurement update equations remain the same. The measurement noise in this case is set to $\mathrm{R} = 0.1^2$ for both position and velocity. Using the piecewise process noise model as derived in Equation (2.54) is,

$$\begin{aligned} \mathrm{Q} &= \mathbb{E}[\Gamma \mathrm{w}(t)\mathrm{w}(t)\Gamma^T] \\ &= \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 \end{bmatrix} \end{aligned} \tag{4.9}$$

Figure 4.7 shows the application of Kalman filter to the drifting damped harmonic oscillator. Figure (a) exhibits the drift correction in the position while Figure (b) shows the drift correction in the velocity that was obtained in the same integral before obtaining position.

4.3.2 Estimating ω^2 and γ

In the previous case, we successfully filtered the integration drift in position and velocity of a damped harmonic oscillator with known parameters. But, what if the underlying parameters of the phenomena were unknown and we had to correct the drift as well as estimate the parameters of the system. For this we explore the applicability of both the Extended Kalman Filter and the Unscented Kalman Filter.

4.3.2.1 Parameter estimation using Extended Kalman Filter

We have already performed the simulated application of this parameter estimation in Chapter 3. The state variables matrix x is augmented with the unknown variables as shown in Equation (3.38). Where x_1 is position, x_2 is velocity, x_3 is square of the frequency and x_4 is the damping coefficient.

The nonlinear function f of the system is thus again defined as,

$$f(\mathrm{x}) = \begin{pmatrix} \mathrm{x}_2 \\ -\mathrm{x}_4\mathrm{x}_2 - \mathrm{x}_3\mathrm{x}_1 \\ 0 \\ 0 \end{pmatrix} \tag{4.10}$$

Through Euler integration of Equation (4.10) the states are predicted in the time update step of the filter. We take both position and velocity as the measurement inputs to improve the performance of the filter. So, the measurement matrix is,

$$\mathrm{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.11}$$

The linearized state transition matrix for this system is already derived in Equation (3.43). We rewrite it here for convenience,

$$\begin{aligned} \mathrm{F} &= e^{\Phi\Delta t} \approx \mathrm{I} + \Phi\Delta t \\ &= \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ -\mathrm{x}_3\Delta t & 1 - \mathrm{x}_4\Delta t & -\mathrm{x}_1\Delta t & -\mathrm{x}_2\Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{4.12}$$

The initialization values of the state variable matrix are,

$$\mathrm{x}_0 = \begin{bmatrix} 0 & 0.01 & 7 & 2 \end{bmatrix}^T \tag{4.13}$$

Associated uncertainties of these variables in the initial error covariance matrix are set to be:

$$\mathrm{P}_0 = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix} \tag{4.14}$$

For this experimental investigation we derive the continuous process noise covariance matrix using Equation (2.50). The derivation is as follows,

$$
\begin{aligned}
\mathrm{Q} &= \int_0^{\Delta t} \mathrm{F}_t \mathrm{Q}_c \mathrm{F}_t^T dt \qquad (4.15)\\
&= \Phi_c \int_0^{\Delta t} dt \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ -\mathrm{x}_3\Delta t & 1-\mathrm{x}_4\Delta t & -\mathrm{x}_1\Delta t & -\mathrm{x}_2\Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & -\mathrm{x}_3\Delta t & 0 & 0 \\ \Delta t & (1-\mathrm{x}_4)\Delta t & 0 & 0 \\ 0 & -\mathrm{x}_1\Delta t & 1 & 0 \\ 0 & -\mathrm{x}_2\Delta t & 0 & 1 \end{bmatrix} \\
&= \Phi_c \int_0^{\Delta t} dt \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ -\mathrm{x}_3\Delta t & 1-\mathrm{x}_4\Delta t & -\mathrm{x}_1\Delta t & -\mathrm{x}_2\Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -\mathrm{x}_3\Delta t & 0 & 0 \\ 0 & (1-\mathrm{x}_4)\Delta t & 0 & 0 \\ 0 & -\mathrm{x}_1\Delta t & 0 & 0 \\ 0 & -\mathrm{x}_2\Delta t & 0 & 0 \end{bmatrix} \\
&= \Phi_c \begin{bmatrix} 0 & \frac{-\mathrm{x}_3\Delta t^2}{2} + \frac{\Delta t^2}{2} - \mathrm{x}_4\frac{\Delta t^3}{3} & 0 & 0 \\ 0 & -\mathrm{x}^3\frac{\Delta t^3}{3} + \Delta t - 2\mathrm{x}_4\frac{\Delta t^2}{2} + \frac{\mathrm{x}_4^3}{3}\frac{\Delta t^3}{3} + \mathrm{x}_4^2\frac{\Delta t^4}{4} + \mathrm{x}_2^2\frac{\Delta t^3}{3} & 0 & 0 \\ 0 & 0 & -\mathrm{x}_1\frac{\Delta t^2}{2} & 0 \\ 0 & 0 & 0 & -\mathrm{x}_2\frac{\Delta t^2}{2} \end{bmatrix} \qquad (4.16)
\end{aligned}
$$

We empirically tune the value of the spectral density Φ_c and find it to be 5.5^2. The uncertainty in position and velocity measurement quantities is a diagonal matrix with values,

$$
\mathrm{R} = \begin{bmatrix} 0.05 & 0 \\ 0 & 0.05 \end{bmatrix} \qquad (4.17)
$$

Figure 4.8 shows the application of the Extended Kalman Filter to our experimental regime. It is a joint state and parameter estimation along with drift removal in the integrated position and velocity of the mass oscillator. In this application, the primary role is played by the continuous piecewise noise covariance matrix Q_c. The matrix is derived as shown in Equation (4.16). This is because the drift in the integration implies a rapidly changing acceleration with everytime step. So, the piecewise noise model is not applicable in this case. The spectral noise density Φ_c acts as the tuning parameter in the continuous noise model. We empirically find the optimum value which filters the drift and provides optimum estimates of the unknown parameters of the system.

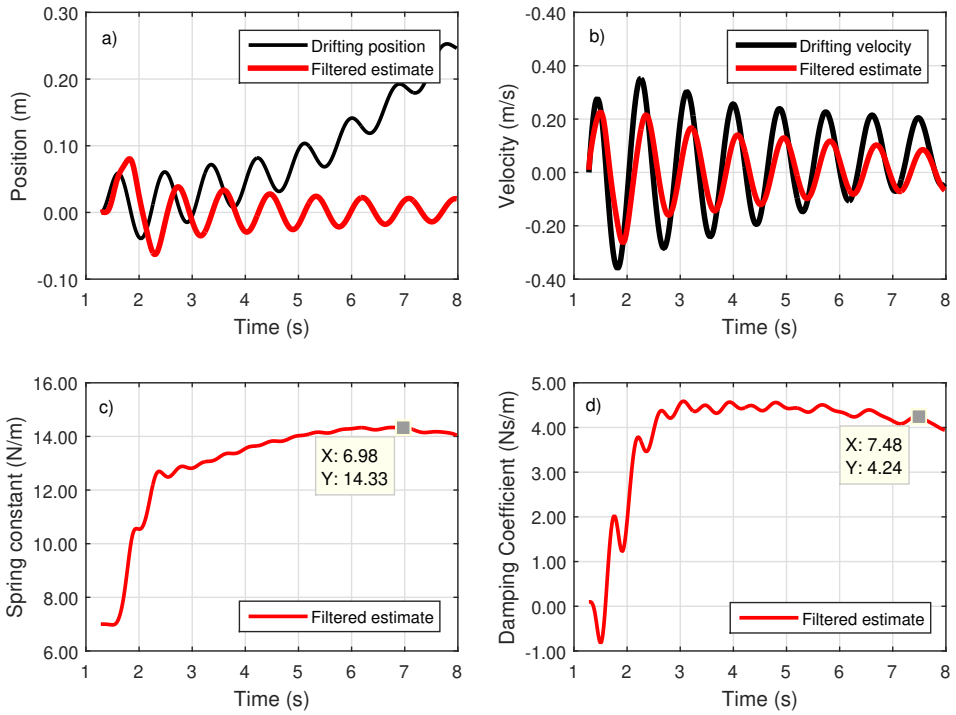

FIGURE 4.8: *Drift removal and joint state and parameter estimation of the Damped Harmonic Oscillator system using Extended Kalman Filter.*

4.3.2.2 Estimating parameters using Unscented Kalman Filter

In Chapter 3 we saw that the simulated application of Unscented Kalman Filter to the Damped Mass oscillator problem exhibited better performance as compared to the Extended Kalman Filter in the parameter estimation. We now verify this simulation by application Unscented Kalman Filter to this experimental example. Likewise, we will perform the drift removal and joint state and parameter estimation.

Before finding the sigma points we first define the difference equations.

$$(\mathrm{x}_1)_k = (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1} T_s \tag{4.18}$$

$$\begin{aligned}(\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1} T_s \\ &= (\hat{\mathrm{x}}_2)_{k-1} + \left(-(\hat{\mathrm{x}}_4)_{k-1}(\hat{\mathrm{x}}_1)_{k-1} - (\hat{\mathrm{x}}_3)_{k-1}(\hat{\mathrm{x}}_2)_{k-1}\right) T_s \end{aligned} \tag{4.19}$$

$$(\mathrm{x}_3)_k = (\hat{\mathrm{x}}_3)_{k-1} T_s \tag{4.20}$$

$$(\mathrm{x}_4)_k = (\hat{\mathrm{x}}_4)_{k-1} T_s \tag{4.21}$$

where x_1 is position, x_2 is velocity, x_3 is square of the frequency ω^2 and x_4 is the damping coefficient γ.

Using scaled sigma points and associated weights the states of the system are predicted using Equation (3.72) and Equation (3.73).

$$\mathrm{x}_k = \sum_{i=0}^{8} \mathrm{w}_m y_\sigma \tag{4.22}$$

$$\mathrm{P}_k = \sum_{i=0}^{8} \mathrm{w}_c (y_\sigma - \mathrm{x}_k)(y_\sigma - \mathrm{x}_k)^T + \mathrm{Q} \tag{4.23}$$

Now, to create the measurement function the sigma points from position and velocity states of the system are transformed into the measurement space for the update step of the filter in Equation (3.71).

$$\mathbb{Z} = \begin{bmatrix} h(y_\sigma)_p \\ h(y_\sigma)_v \end{bmatrix} \tag{4.24}$$

The state and error covariance in measurement space is updated then using Equation (3.72) and (3.73)

$$\mu_{z_k} = \sum_{i=0}^{8} \mathrm{w}_m \mathbb{Z} \tag{4.25}$$

$$\mathrm{P}_k = \sum_{i=0}^{8} \mathrm{w}_c (\mathbb{Z} - \mu_{z_k})(\mathbb{Z} - \mu_{z_k})^T + \mathrm{R} \tag{4.26}$$

As we have two states being measured the residual is calculated using Equation (3.149). The only difference is that the mean of the measurement space μ_{z_k} is two dimensional.

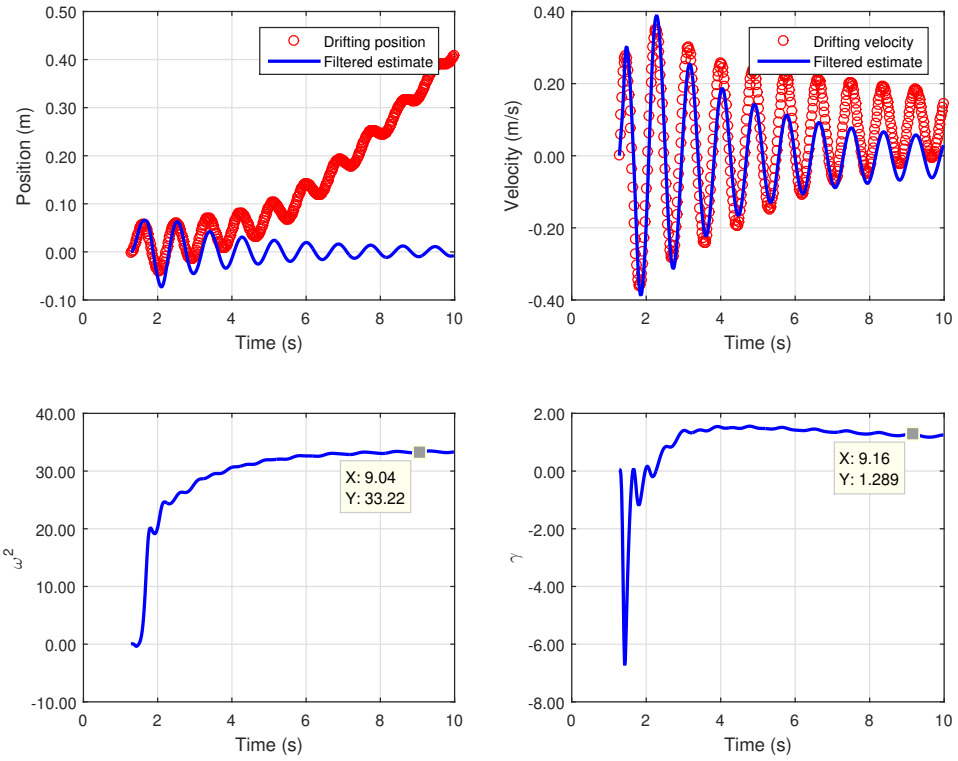

FIGURE 4.9: *Drift removal and joint state and parameter estimation of the Damped Harmonic Oscillator system using Unscented Kalman Filter.*

$$\mathrm{Y} = \mathrm{Z} - \mu_{z_k} \tag{4.27}$$

$$= \mathrm{Z} - \begin{bmatrix} \mu_{z_p} \\ \mu_{z_v} \end{bmatrix}_k \tag{4.28}$$

Figure 4.9 shows the application of the Unscented Kalman Filter to the joint state and parameter estimation with drift removal in the integrated position and velocity of the mass oscillator. The Unscented Kalman Filter also successfully filters out the drift in the position and gives a good estimate of the unknown parameters ω^2 and γ.

The following table highlights the performance of both filters in this application.

	Square of frequency ω^2 (rad^2/s^2)		Damping coefficient γ (s^{-1})	
True values	61.7		15.9	
	EKF	UKF	EKF	UKF
Estimated	14.33	33.22	4.24	1.289
RMSE	48.85	33.03	12.02	14.89

TABLE 4.1: The performance comparison of Extended Kalman Filter and Unscented Kalman Filter using the Root Mean Square Error (RMSE) values for estimating the parameters of the damped mass oscillator problem.

This methodology of using Kalman filter to remove drifts and estimate parameters can play a pivotal role for a physicist in an experimental physics laboratory. Various linear and nonlinear phenomena are current gauged using conventional methodologies and are usually cumbersome but with Kalman filter we only need a few sensors to study the phenomena and then use the filter to remove noise and estimate its unknown parameters.

4.4 Dynamics of the Wilberforce pendulum

We have already simulated the application of Kalman filtering to Wilberforce pendulum dynamics in Chapter 3. Now, we extend our work and perform Kalman filtering on the Wilberforce experimental setup from the Physlab and try to estimate the parameters that govern the nature of a Wilberforce pendulum.

4.4.1 Apparatus

The experimental apparatus consists of a standard laboratory stand with eight different machined oscillators. These oscillators are hung using a spring on a hooked arm clamped on to the lab stand. They have different geometry and mass distribution. This alters their moment of inertia. Last, we have a smartphone attached over a stand that is used to capture the video of the oscillating mass. Figure 4.10 shows the complete apparatus.

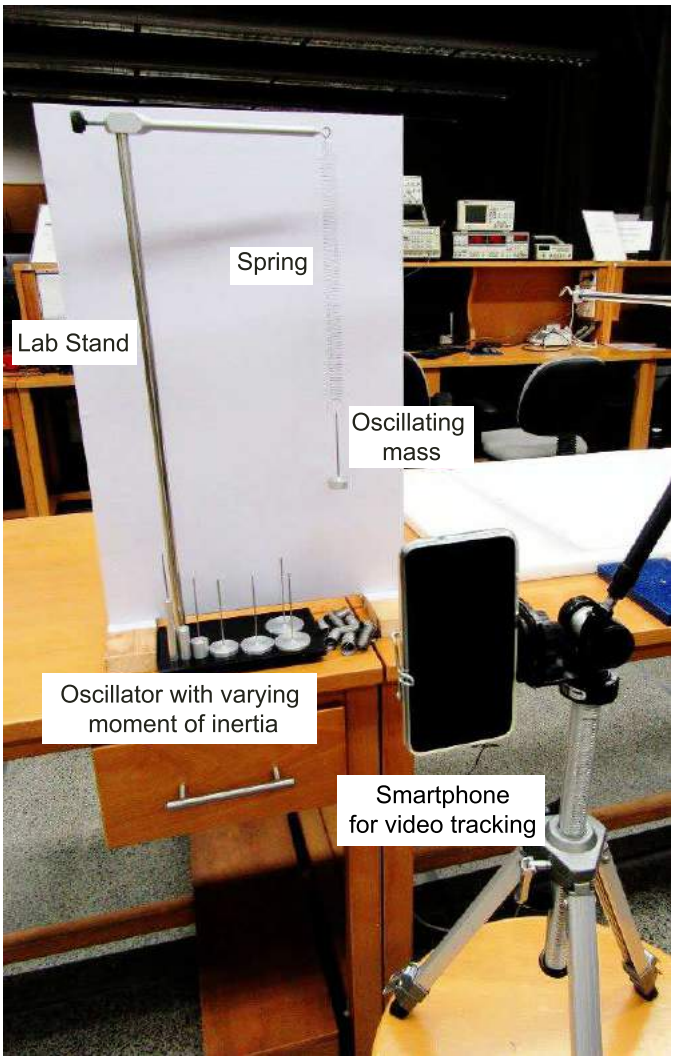

FIGURE 4.10: *Experimental setup to track the dynamics of the Wilberforce pendulum.*

4.4.2 Methodology and experiment

In the conventional laboratory method, the parameters of the Wilberforce pendulum are found through a tedious process of varying moment of inertia of the oscillators and then mathematically fitting the data to an quadratic mathematical model derived from Equations (3.154) and (3.155). These are the two equations of motion that describe the translational and rotational oscillations of the Wilberforce pendulum. These equations of motion are:

$$m\left(\frac{d^2z}{dt^2}\right) + kz + \varepsilon\theta/2 \;=\; 0 \tag{4.29}$$

$$I\left(\frac{d^2\theta}{dt^2}\right) + \delta\theta + \varepsilon z/2 \;=\; 0 \tag{4.30}$$

where $\varepsilon z\theta/2$ is the coupling between the translational and torsional motion. The parameter ε is a measure of the strength of the coupling. For an uncoupled system, the natural frequencies of the longitudinal oscillation and the torsional motion are given by,

$$\omega_z = \sqrt{\frac{k}{m}} \tag{4.31}$$

$$\omega_\theta = \sqrt{\frac{\delta}{I}}. \tag{4.32}$$

The two mutually coupled equations of motion can be simultaneously solved to determine the frequencies of the two normal modes of the system. This is what we attempt to achieve in the conventional method.

Using the frequencies of the normal modes and the definition $\omega = 2\pi f$ we derive the following algebraic equation,

$$(f_1^2 - f_2^2)^2 = \left(\frac{\delta^2}{16\pi^4}\right)\left(\frac{1}{I}\right)^2 + \left(\frac{\varepsilon^2 - 2\delta k}{16\pi^4 m}\right)\left(\frac{1}{I}\right) + \left(\frac{k^2}{16\pi^4 m^2}\right). \tag{4.33}$$

Defining two new variables,

$$\phi = (f_1^2 - f_2^2)^2, \tag{4.34}$$

$$J = \frac{1}{I}. \tag{4.35}$$

Equation (4.33) can be rearranged in its quadratic guise,

$$\phi = C_2 J^2 + C_1 J + C_o. \tag{4.36}$$

Now if we were to vary I and have some means of measuring the normal modes, f_1 and f_2, we can fit the variation of ϕ with $J = 1/I$. The fitting constants will

help us determine the parameters of the Wilberforce pendulum:

$$\delta = 4\pi^2\sqrt{C_2}, \tag{4.37}$$

$$k = 4\pi^2 m\sqrt{C_o}, \tag{4.38}$$

$$\varepsilon = \sqrt{16\pi^4 m C_1 + 2\delta\, k}. \tag{4.39}$$

The determination of the two normal mode frequencies, f_1 and f_2 as a function of moment of inertia I, therefore, in principle, allows to measure the coupling, spring and torsional constants. This is precisely the way we find the parameters conventionally. The students too observe the motion of the Wilberforce pendulum and extract the normal modes by Fourier transforming the z motion of the pendulum and quadratic fit against the changing inertia to find the parameters. Table 4.2 summarizes the dimensions of the masses, their respective moments of inertia and the associated normal modes extracted using Fast Fourier Transform (FFT).

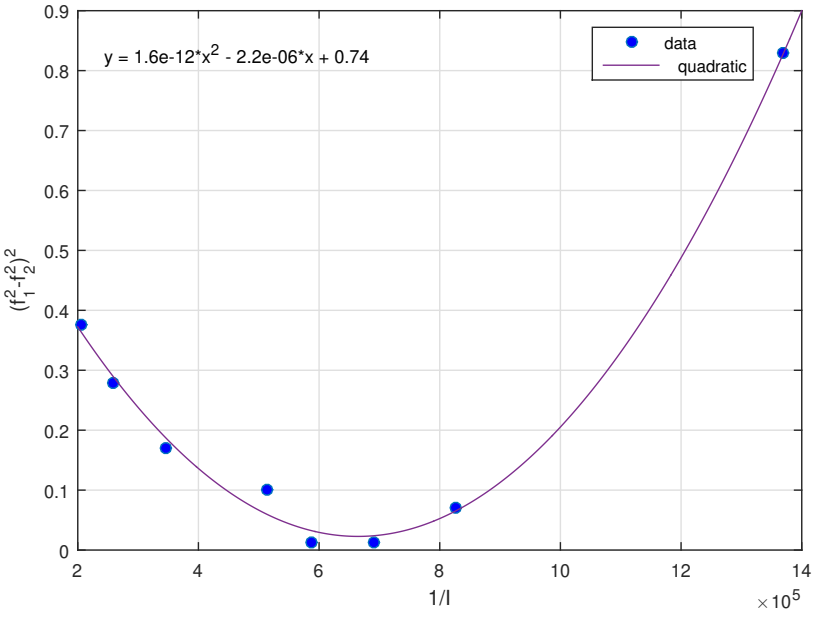

FIGURE 4.11: *The quadratic fit plot of the square of the difference between the squared frequencies of the system and the different moment of inertia.*

From the data obtained in Table 4.2 we create a plot shown in Figure 4.11. The quadratic fit of the plot reveals the values of the parameters $C_2 = 1.6 \times 10^{-12}$, $C_1 = 2.2 \times 10^{-}6$ and $C_o = 0.74$. Using these values we find the torsional constant, spring constant and the coupling constant of the pendulum. Solved as follows,

Oscillator	f_1	f_2	$(f_1^2 - f_2^2)^2$	Mass (Kg)	Diameter (m)	Height (m)	Moment of Inertia (Kgm^2)$\times 10^{-7}$
B	0.542	0.952	0.3752	0.0184	0.0459	0.0041	48.5
C	0.615	0.952	0.2789	0.0183	0.0411	0.0050	38.5
D	0.703	0.952	0.1699	0.0184	0.0356	0.0065	29.0
E	0.767	0.952	0.1011	0.0184	0.0291	0.0097	19.5
F	0.901	0.959	0.0119	0.0184	0.0272	0.0011	17.0
G	0.938	0.996	0.0128	0.0183	0.0252	0.0013	14.5
H	0.945	1.077	0.0714	0.0184	0.0230	0.0154	12.1
J	0.952	1.348	0.8292	0.0183	0.0178	0.0257	7.3

TABLE 4.2: The actual dimensions of the different mass oscillators used in the Wilberforce experiment along with their moment of inertia and the two resonant modes computed by Fast Fourier Transform.

Computing the torsional constant

$$\begin{aligned}\delta &= 4\pi^2\sqrt{C_2} \\ &= 4(3.14)^2\sqrt{1.6\times 10^{-12}} \\ &= 4.9886\times 10^{-5}\ kgm^2/s^2 \end{aligned} \tag{4.40}$$

Computing the spring constant

$$\begin{aligned}k &= 4\pi^2 m\sqrt{C_o}, \\ &= 4(3.14)^2(0.0184)(\sqrt{0.74}) \\ &= 0.6242\ N/m \end{aligned} \tag{4.41}$$

Computing the coupling constant

$$\begin{aligned}\varepsilon &= \sqrt{16\pi^4 m C_1 + 2\delta\, k}. \\ &= \sqrt{(16)(3.14)^4(0.0184)(-2.2\times 10^{-6}) + (2)(4.9886e-05)(0.6242)} \\ &= 6.84\times 10^{-7}\ N \end{aligned} \tag{4.42}$$

Once we have obtained the values of these parameters we now try to estimate these parameters using Extended and Unscented Kalman Filters.

4.4.3 Applying the Extended Kalman Filter

In this joint state and parameter estimation of the Wilberforce pendulum, the state space variable matrix is quite large. This is because the first four state variables represent the kinematic states i.e. x_1 and x_2 representing the position and velocity respectively in the vertical z- direction while x_3 and x_4 represent rotational position and angular velocity respectively. Whereas, the rest of the state variables represent the unknown modes $\mathrm{x}_5 = \omega_z^2$, $\mathrm{x}_6 = \omega_\theta^2$ and the coupling constant $\mathrm{x}_7 = \varepsilon$.

$$\mathrm{x} = \begin{bmatrix} \mathrm{x}_1 & \mathrm{x}_2 & \mathrm{x}_3 & \mathrm{x}_4 \end{bmatrix}^T \tag{4.43}$$

The discrete state transition matrix F in this case, as already derived in Equation (3.175) is following,

$$\mathrm{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 & 0 \\ (-\mathrm{x}_5)\Delta t & 1 & (\frac{-1}{2m}\mathrm{x}_7)\Delta t & 0 & (-\mathrm{x}_1)\Delta t & 0 & (\frac{1}{2m}\mathrm{x}_3)\Delta t \\ 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ (\frac{-\mathrm{x}_7}{2I})\Delta t & 0 & -(\mathrm{x}_6)\Delta t & 1 & 0 & (-\mathrm{x}_3)\Delta t & (\frac{-1}{2I}\mathrm{x}_1)\Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.44}$$

Likewise, we set the initial error covariance matrix P_0 as,

$$\hat{\mathrm{P}}_o = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix} \tag{4.45}$$

4.4.3.1 Predicting the state and errors for parameter estimation

Following set of equations are augmented to include the unknown parameters $\mathrm{x}_5 = \omega_z^2$, $\mathrm{x}_6 = \omega_\theta^2$ and $\mathrm{x}_7 = \varepsilon$. We propagate these through Euler integration in the *a priori* prediction step,

$$
\begin{aligned}
(\mathrm{x}_1)_k &= (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1}\Delta t \\
(\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + (-\hat{\mathrm{x}}_5(\hat{\mathrm{x}}_1)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_3)_{k-1})\Delta t \qquad (4.46) \\
(\mathrm{x}_3)_k &= (\hat{\mathrm{x}}_3)_{k-1} + (\hat{\mathrm{x}}_4)_{k-1}\Delta t \\
(\mathrm{x}_4)_k &= (\hat{\mathrm{x}}_4)_{k-1} + (-\omega_\theta^2(\hat{\mathrm{x}}_3)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_1)_{k-1}) \qquad (4.47) \\
(\mathrm{x}_5)_k &= (\hat{\mathrm{x}}_5)_{k-1} \qquad (4.48) \\
(\mathrm{x}_6)_k &= (\hat{\mathrm{x}}_6)_{k-1} \qquad (4.49) \\
(\mathrm{x}_7)_k &= (\hat{\mathrm{x}}_7)_{k-1} \qquad (4.50)
\end{aligned}
$$

Using the piecewise-noise model shown in Equation (2.54) we derive the following Q matrix:

$$
\mathrm{Q} = \begin{bmatrix}
\frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\
\frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\
\frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\
\frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \sigma_v^2 \qquad (4.51)
$$

The variance σ_v^2 is tuned to be $\sigma_v^2 = 5.5^2$.

The Figure 4.12 exhibits our application of the Extended Kalman Filter to estimate the normal modes and the coupling constant of the Wilberforce pendulum system. The true values of the system are already computed using the conventional method $\omega_z = 5.81$ rad/s, $\omega_\theta = 5.42$ rad/s and $\epsilon = 6.84 \times 10^{-7}$ N.

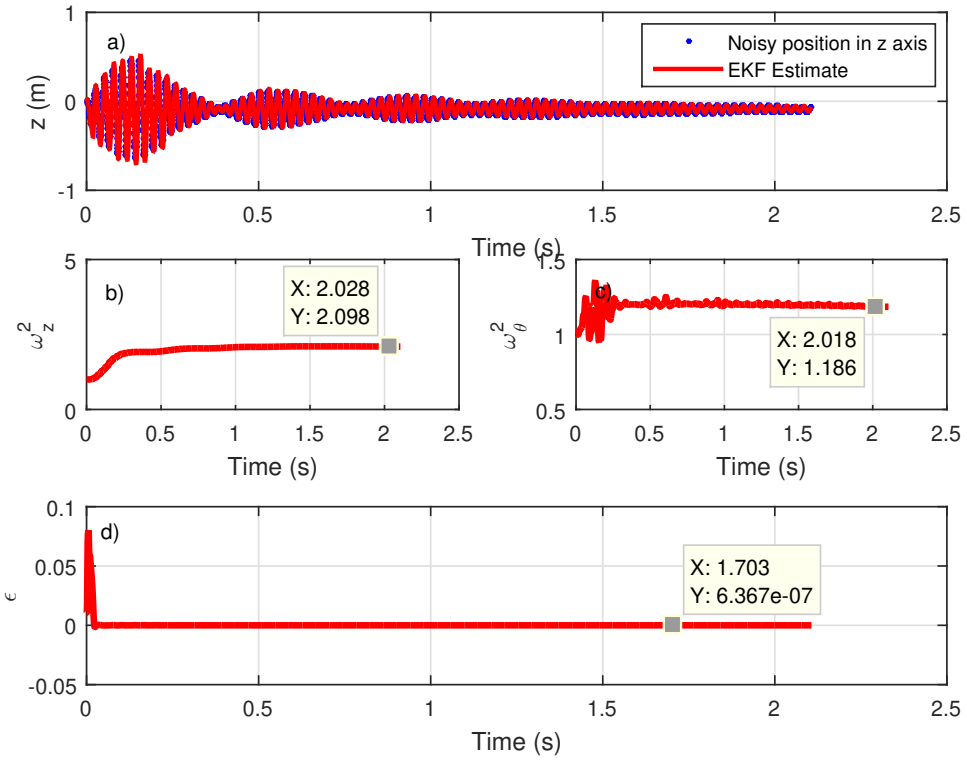

FIGURE 4.12: *The application of Extended Kalman filter to the Wilberforce pendulum, Figure (a) exhibits the video tracked position and the Kalman estimate, Figure (b) and Figure (c) shows the estimated translational and angular modes, the true values of which are* $\omega_z^2 = 33.70\ rad^2/s^2$ *and* $\omega_z^2 = 29.35\ rad^2/s^2$ *Figure (d) shows the estimated coupling constant* $6.84 \times 10^{-7}\ N$ *between the two motions of the Wilberforce pendulum.*

4.4.4 Applying Unscented Kalman Filter

We also estimate the Wilberforce parameters using the Unscented Kalman Filter. Using the same state dynamic Equations (3.157) we first state our nonlinear function,

For this application we will generate $2n+1 = 15$ scaled sigma points, where $n = 7$, that approximate the translational and rotational states along with the unknown parameters of the Wilberforce pendulum system.

Likewise, as seen in various previous examples, the associated weights for the sigma points are calculated using Equations (2.73) and (2.74).

After we have generated sigma points and their associated weights, we pass them through the following nonlinear function,

$$
\begin{aligned}
(\mathrm{x}_1)_k &= (\hat{\mathrm{x}}_1)_{k-1} + (\hat{\mathrm{x}}_2)_{k-1}\Delta t \\
(\mathrm{x}_2)_k &= (\hat{\mathrm{x}}_2)_{k-1} + (-\hat{\mathrm{x}}_5(\hat{\mathrm{x}}_1)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_3)_{k-1})\Delta t \qquad (4.52) \\
(\mathrm{x}_3)_k &= (\hat{\mathrm{x}}_3)_{k-1} + (\hat{\mathrm{x}}_4)_{k-1}\Delta t \\
(\mathrm{x}_4)_k &= (\hat{\mathrm{x}}_4)_{k-1} + (-\omega_\theta^2(\hat{\mathrm{x}}_3)_{k-1} - \frac{1}{2m}(\hat{\mathrm{x}}_7)_{k-1}(\hat{\mathrm{x}}_1)_{k-1}) \qquad (4.53) \\
(\mathrm{x}_5)_k &= (\hat{\mathrm{x}}_5)_{k-1} \qquad (4.54) \\
(\mathrm{x}_6)_k &= (\hat{\mathrm{x}}_6)_{k-1} \qquad (4.55) \\
(\mathrm{x}_7)_k &= (\hat{\mathrm{x}}_7)_{k-1} \qquad (4.56)
\end{aligned}
$$

The prediction and correction steps proceed in the usual way.

Next, we have to initialize the belief in our system. We build our initial state error covariance matrix P_0 as follows:

$$
P_0 = \begin{bmatrix}
0.1 & 0 & 0 & 0 & 0 \\
0 & 0.1 & 0 & 0 & 0 \\
0 & 0 & 0.1 & 0 & 0 \\
0 & 0 & 0 & 0.1 & 0 \\
0 & 0 & 0 & 0 & 0.1
\end{bmatrix} \qquad (4.57)
$$

We use the same process noise Q matrix (3.189) as the piecewise noise model for this system.

$$
\mathrm{Q} = \begin{bmatrix}
\frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\
\frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\
\frac{T_s^4}{4} & \frac{T_s^3}{2} & \frac{T_s^4}{4} & \frac{T_s^3}{2} & 0 & 0 & 0 \\
\frac{T_s^3}{2} & T_s^2 & \frac{T_s^3}{2} & T_s^2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \sigma_v^2 \qquad (4.58)
$$

We inflate the process variance σ_v^2 to tune the filter for getting the best estimate of the parameters and the states. in this application it was found to be $\sigma_v^2 = 5.5^2$ that was most optimum.

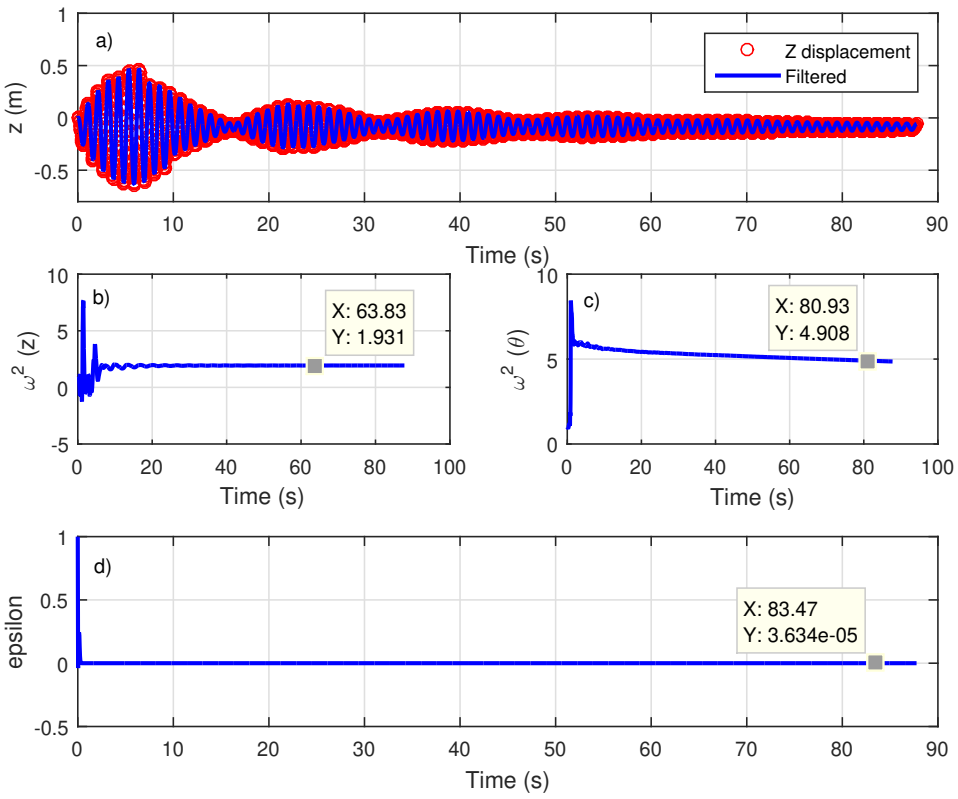

FIGURE 4.13: *The application of Unscented Kalman filter to the Wilberforce pendulum, Figure (a) exhibits the video tracked position, Figure (b) and Figure (c) shows the estimated translational and angular modes, the true values of which are* $\omega_z^2 = 33.70\ rad^2/s^2$ *and* $\omega_z^2 = 29.35\ rad^2/s^2$ *Figure (d) shows the estimated coupling constant* $6.84 \times 10^{-7}\ N$ *between the two motions of the Wilberforce pendulum.*

Furthermore, in both cases we are using only one noisy sensor to track the position of the oscillator in the vertical z direction in meters. The rest of the Unscented Kalman Filter algorithm runs as already described in various previous examples and simulations.

Figure 4.13 shows the results of using the *Unscented Kalman Filter* algorithm to filter a simulated Wilberforce Pendulum that is infested with $\sigma^2 = 0.001$ m and $\sigma^2 = 0.01$ rad noise in vertical position z and angular measurement respectively.

Chapter 5

Summary and Conclusion

5.1 Summary

In the end we would like to present the summary of our expedition of Kalman Filter's application in the realms of an experimental physics laboratory. In Chapter 1, we first share our motivation towards exploring these applications. We briefly introduced the filter's invention, historical background and its application areas in process control, flood prediction, tracking and navigation and that how it is integrated with sensors to output sophisticated data that can be used for better analysis. We also defined our problem statement that in an experimental laboratory students have to deal with infestation of noise into the experimental data which requires post acquisition processing and filtering for analysis and largely for estimating the unknown parameters of the system. We present Kalman Filter as a modern technique for data filtration and parameter estimation. We also shared the limitations of this dissertation and that how anyone could explore further applications in this unexplored potential area of experimental physics. We also shared the great historical background of the invention of Kalman Filter and that how it played its pivotal role in taking the mankind to the moon. We also discussed how the filter has evolved over time to find its place in various domains of sciences and engineering. We also present a literature review of the diverse ranged applications of Kalman Filter in fields of computer science, physics and industrial engineering.

In Chapter 2, we build the theoretical foundation of the reader. In order to understand Kalman Filter's mathematical structure one has to understand the state space formulation from the equation of motions of the system. From this

we further describe an essential property of the system referred to as observability test for both linear and nonlinear systems. The test evaluates whether using a given set of sensors could estimate the states of a system or not. In some cases we might not be fully estimate the states of the system using a particular set of sensors but only may be able to estimate sub parts of the system. For this, the sub-observable systems are extracted using the technique of *Kalman Decomposition*. In this dissertation, we did not need to use Kalman Decomposition as our systems were fully observable. However, the theoretical background was introduced as an essential knowledge for the reader. We then derive the complete Kalman Filter algorithm for a single dimensional state space problem and individually discuss the *Time Update* and *Measurement Update* equations which govern the recursive execution of the filter. We also describe the high dimensional *Multivariate Kalman Filter* where we have more than one dimensions of the system. The filter equations transform into high dimension matrices and the algorithm incorporates matrix manipulation rules in order to process the application. We then describe the second essence of this research work on using Kalman Filter for parameter estimation. The parameters govern the trajectory of the system over time. We share the two known forms of Kalman Filter for parameter estimation, i.e. *Dual-State* and *Joint-State Parameter estimation*. For our thesis we choose joint state and parameter estimation as our experimental systems were not complex. We then also discuss the nonlinear form of Kalman Filter referred to as the *Extended Kalman Filter*. The *EKF* uses first approximations to linearize the given systems state as well as the error covariances. This approximation is what makes Extended Kalman Filter sub-optimal in applications with high dimensions. So, in the last we discuss the *Unscented Kalman Filter* as an alternative to *Extended Kalman Filter* for performance enhancement and robustness. We describe the complete algorithm and underlying structure of the *Unscented Kalman Filter*.

In Chapter 3, we perform the numerical simulations of our application of *Extended* and *Unscented Kalman Filter* on a a number of linear and nonlinear noisy systems with known and unknown parameters. Our first simulations was of a noisy damped harmonic oscillator. We derive the equation of motion for the system using the Euler-Lagrange classical mechanical approach and transform the equations into state space as described in Chapter 3. We then derive the state space model of the mass-spring system and discretize it using the Taylor series approximation. We then derive the process noise covariance matrix and measurement noise matrix. We also perform the linear observability test and conclude that the

system is observable using position or velocity sensor. We then explore parameter estimation of the mass-oscillator problem using Extended and Unscented Kalman Filter. We perform the nonlinear observability test in this case as the augmented state becomes nonlinear due to the presence of the unknown parameter in the system. We successfully estimate the frequency ω^2 and the damping coefficient γ of the system. We also worked on the application of the filter to a highly nonlinear system. The *Duffing Oscillator* is a simple harmonic oscillator infested with a cubic elasticity. We also consider this system to be driven by a cosine function. We first simulate the true dynamics of the system and infest a simulated noise in the position measurements. We use a position sensor in the *Extended* and *Unscented* Kalman Filter to filter the noise in position and estimate the velocity of the system. We also check the application for the case of using only the velocity sensor. Both the filters work perfectly fine. However, the application of *UKF* is found to be more precise using the *Root Mean Squared Error* calculation. As this nonlinear system was driven we also explore the input part of the Kalman Filter algorithm which was not discussed in the previous simulation.

In Chapter 4, we proceed to the experimental realizations of the filter. We first begin with a simple application to a one dimensional problem of temperature measurement. Using a thermocouple we measure the temperature of a Peltier heater that is heated using a PID controller circut. The heater is given a set point temperature that is altered after regular intervals to see how the dynamics of the filter adjusts to the variation in temperature and the associated process noise. Using this system we explored the tuning factor of *Process Noise*. We then proceed to our experimental mass-spring damper system. Using a force sensor we measure the acceleration of an oscillating mass and convert it into position readings through integration which adds integration drift to the position measurement. We also aim to estimate the frequency ω^2 and damping coefficient γ of the system. We apply both the *Extended* and *Unscented Kalman Filter*. For our nonlinear system we use apparatus of the *Magnetic Pendulum* in our laboratory. This apparatus uses magnets to add nonlinear elasticity to a driven simple pendulum. We estimate the value of this nonlinear elasticity using the *Extended* and *Unscented Kalman Filter*.

5.2 Conclusion

Through this research work we have tried to highlight the applications of Kalman Filter and its variants in the largely unexplored realms of an experimental physics laboratory. Generally, the most common platform for Kalman filters is found in signal processing, robotics, image processing and navigation. However, use of noisy sensors and mathematical processes are also a part of physics exploration but students in this domain are rarely introduced to this modern technique.

Specifically, in an undergraduate physics laboratory, where through sensors and data acquisition equipment students explore basic physical phenomena of kinematics, electromagnetism etc. They are already engaged in applying filters to remove noise and perform tedious curve fitting to extract unknown parameters of the system. Their introduction to a more robust and modern technique of Kalman filters could help them not only in improving their exploration but it also helps in exploring complex and high dimensional problems as well. We have explored a few such examples through simulations and heuristic experiments. For example, in Figure 3.2 we have shown an application of Kalman Filter to a *Damped Harmonic Oscillator*. It is the most simplest and common example for a physics student. We saw how noise is effecting the system and how uncertainty in the process model effects the performance of the filter, shown in Figure 3.3 and then estimate its unknown parameters, the spring constant and damping coefficient shown in Figure 3.5 and 3.6.

The nonlinear physics is a huge area for further research in Kalman filter's application and so we have also applied the nonlinear variants of Kalman filter to nonlinear dynamical systems. Shown in Figure 3.8 is an application of the Extended Kalman Filter to a Duffing oscillator. Figure 3.9 shows the estimation of the nonlinear parameter of the system using Extended Kalman Filter but as EKF is considered to be sub-optimal we also applied the more robust Unscented Kalman Filter as shown in Figure 3.10 and 3.11. The performance of both filters is highlighted in Table 3.2.7. It verifies that the Unscented Kalman Filter performed better than the Extended Kalman Filter in estimating the cubic nonlinear parameter of the Duffing oscillator.

Last, we applied Kalman filter to a Wilberforce pendulum in an attempt to estimate its governing parameters. It is a coupled harmonic oscillator that experiences

coupling effect in between its translational and rotational displacement. The frequency modes in both along with a coupling constant is generally computed by varying the moments of inertia of the oscillator and curve fitting using a quadratic Equation (4.36). We have tried to show that by using merely a position sensor one can attempt to estimate the two modes as well as the coupling constant. However, as shown in Figure 4.12 and 4.13. The coupling constant is at a good estimation to around 6.367×10^{-7} N where the true value was 6.840×10^{-7} N making it a percentage difference of 6.9% but the normal modes ω_z^2 and ω_θ^2 are estimated by Extended Kalman Filter at a difference of 86.98% and 95.86% respectively from the true values. This can be improved by including a secondary sensor to measure the angular displacement of the pendulum as simulation in Chapter 3. Furthermore, by using Kalman decomposition [38] the Wilberforce pendulum system could be decomposed into observable and unobservable states using only a position sensor.

5.3 Future works

The smartphones today carry drifting inertial sensors. We can use these sensors to measure kinematics of various phenomena. We would like to apply Kalman filtering to remove the drifts in such sensors and estimate underlying parameters of the systems. We can also use a digital camera to record objects and track their trajectories, e.g. projectiles, using a tracking software [48].

We intend to apply Kalman filters to more and more problems in physics. In particular, to the experiments in linear and nonlinear mechanics, electromagnetism and nuclear physics.

Appendix A

MATLAB codes

A.1 Piecewise white noise model

```
function output=piecewise_white_noise(ndim,variance,dt)

% We have defined F for the standard kinematic equations.
% USage: piecewise_white_noise(ndim=2 or 3,variance of noise,dt=time step)

if ndim==5
G=@(t)[t^4/4;t^3/3;t^2/2;t;1];
Gc=@(t)[t^4/4 t^3/3 t^2/2 t 1];
output=variance*G(dt)*Gc(dt);

elseif ndim==4
G=@(t)[t^3/3;t^2/2;t;1];
Gc=@(t)[t^3/3 t^2/2 t 1];
output=variance*G(dt)*Gc(dt);

elseif ndim==3
G=@(t)[t^2/2;t;1];
Gc=@(t)[t^2/2 t 1];
output=variance*G(dt)*Gc(dt);

elseif ndim==2
G=@(t)[t^2/2;t];
Gc=@(t)[t^2/2 t];
output=variance*G(dt)*Gc(dt);
end
end
```

A.2 Continuous white noise model

```
function output=continuous_white_noise(spectral_density,time_step)

% We have defined F for the wilberforce linear and angular movement
% [position,velocity,angle,angular velocity]^T as the state function


F=@(t)[1 t 0 0;0 1 0 0;0 0 1 t;0 0 0 1];
Fc=@(t)[1 0 0 0;t 1 0 0;0 0 1 0;0 0 t 1];
Qc=[0 0 0 0;0 1 0 0;0 0 0 0;0 0 0 1];
output=integral(@(t) spectral_density*F(t)*Qc*Fc(t),0,time_step,'ArrayValued',
    true);
end
```

A.3 Van Der Merwe algorithm

```
function [chi,scalefactor,wm,wc]=vandermeer_sigma2(muu,sigma,alpha,beta,kappa)

%implementing van der Meer scaled sigma point calculation
%usage: [sigma,scalefector,wm,wc]=vandermeer_sigma1(mu,sigma,0.4,0.9,0.3)
%mu is column vector
%sigma is a square matrix

%generating the sigma points
ndim2=length(muu); %dimension of state space
lambda=alpha^2*(ndim2+kappa)-ndim2;
chi=zeros(ndim2,2*ndim2+1); %pre-allocating space for the sigma points
chi(:,1)=muu;

scalefactor=real(sqrtm((ndim2+lambda)*sigma));
%scalefactor=chol((ndim2+lambda)*sigma);

for k=2:ndim2+1
    chi(:,k)=muu+scalefactor(:,k-1);
end

for k=ndim2+2:2*ndim2+1
    chi(:,k)=muu-scalefactor(:,k-ndim2-1);
end

%generating the weights in means wm, and weights in covariances wc

wm=zeros(1,2*ndim2+1);
wc=zeros(1,2*ndim2+1);
wm(1)=lambda/(ndim2+lambda);
wc(1)=wm(1)+1-alpha^2+beta;

for k=2:2*ndim2+1
wm(k)=1/(2*(ndim2+lambda));
wc(k)=wm(k);
end

```

```
%figure; error_ellipse(sigma,mu); hold on; scatter(chi(1,:),chi(2,:));

end
```

A.4 Observability test for SHM

```
close all; clear all; clc;

% Assign model parameters
m = 10; % mass
k = 5; % spring constant
b = 3; % damping coefficient
A = [0 1; -k/m -b/m]; % coefficient matrix for continuous system
H=[1 0];

% Observability
Obs = obsv(A,H);

% Number of observable states in A
rank(Obs)
```

A.5 Damped harmonic oscillator function file

```
function xdot=damped_ho(t,x)

global kspring bspring mass

xdot=zeros(2,1);

xdot(1)=x(2);
xdot(2)=-(kspring/mass)*x(1)-(bspring/mass)*x(2);

end
```

A.6 Simulating application of Kalman Filter to a Simple Harmonic Oscillator

```
clear all
clc

% Assign model parameters
m = 10; % mass
```

```
k = 5; % spring constant
b = 3; % damping coefficient
A = [0 1; -k/m -b/m]; % coefficient matrix for continuous system
rhs = @(t,x) A*x; % right hand side of the function
% Simulate system
xinit = [1;0]; % initial value
h = 0.2; % time step
T = 30; %Final time value
time = 0:h:T; %Full time scale
[t,trueTrajectory] = ode45(rhs,time,xinit);
obsNoise = 0.1; %Define observation noise level
obs = trueTrajectory(:,1); %First state is observable
obs = obs+obsNoise*randn(size(obs)); %Add noise
xbar=[1;0];
xbarEstimate = [1;0]; %Initial state
P = .1*eye(length(xbar)); %Initial covariance
varEstimate = diag(P); %Initial state variance
F = eye(length(xbar))+h*A; %State transition matrix
H = [1 0]; %Observation function
Q_variance=0.1^2; %process noise variance, assume a piecewise constant white
    noise
Q=piecewise_white_noise(2,Q_variance,h);
Q=kron(Q,eye(1));
D = obsNoise;
Ob = obsv(A,H);
% If rank = ndim of state vector the system is observable
rank(Ob)

for i = 2:length(obs)
 % Prediction step
 xbar = F*xbar;
 P = F*P*F' + Q;
 % Observation update
 K = (P*H')/(H*P*H'+D); % aka K = ...
 %P*H'*inv(H*P*H'+D);
 xbar = xbar + K*(obs(i) - H*xbar);
 P = P - K*H*P;
 xbarEstimate(:,i) = xbar;
 varEstimate(:,i) = diag(P);

end

figure(1);subplot(2,1,1);
plot(time,trueTrajectory(:,1),'k','linewidth',2);
hold on;
lower = xbarEstimate(1,:)+2*sqrt(varEstimate(1,:));
upper = xbarEstimate(1,:)-2*sqrt(varEstimate(1,:));
ciplot(lower,upper,time,'b')
plot(time,obs,'bo','markerfacecolor','b','markersize',3)
plot(time,xbarEstimate(1,:),'r','linewidth',3);
axis([0 30 -1 1.5])
set(gca,'fontsize',8)
tix=get(gca,'ytick')';
set(gca,'yticklabel',num2str(tix,'%.1f'))
legend 'True Trajectory' 'Simulated Noise' 'KF Estimate' 'Confidence Interval'
```

```
xlabel('Time (s)')
ylabel('Position (m)')

subplot(2,1,2)
plot(time,trueTrajectory(:,2),'k','linewidth',3);
hold on;
lower = xbarEstimate(1,:)+2*sqrt(varEstimate(1,:));
upper = xbarEstimate(1,:)-2*sqrt(varEstimate(1,:));
ciplot(xbarEstimate(2,:)+2*sqrt(varEstimate(2,:)),xbarEstimate(2,:)-2*sqrt(
    varEstimate(2,:)),time,'b')
plot(time,xbarEstimate(2,:),'r','linewidth',3);
tix=get(gca,'ytick')';
set(gca,'fontsize',8)
set(gca,'yticklabel',num2str(tix,'%.1f'))
legend 'True Velocity' 'KF Estimate' 'Confidence Interval'
xlabel('Time (s)')
ylabel('Velocity (m/s)')
```

A.7 Parameter estimation on Damped Harmonic Oscillator using Extended Kalman Filter

```
close all
clear all
clc

% Assign model parameters
m = 10; % mass
k = 5; % spring constant
b = 3; % damping coefficient
omega=sqrt(k/m);
bspring = b/m;
A = [0 1; -k/m -b/m]; % coefficient matrix for continuous system
rhs = @(t,x) A*x; % right hand side of the function
% Simulate system
xinit = [0;1]; % initial value
h = 0.01; % time step
T = 100; %Final time value
time = 0:h:T; %Full time scale
[t,trueTrajectory] = ode45(rhs,time,xinit);
obsNoise = 0.1^2; %Define observation noise level
obs = trueTrajectory(:,1); %First state is observable
obs = obs+obsNoise*randn(size(obs)); %Add noise

omega=k/m;
gamma=b/m;


%% Extended Kalman Filter
xbar=[0;1;1;1]; %Initial state estimate
xbarEstimate=xbar;
```

```
P = diag([0.1 0.1 0.1 0.1]); %Initial covariance
varEstimate = diag(P); %Initial state variance
Aa = [0 1 0 0;-xbar(3)/m -xbar(4)/m -xbar(1)/m -xbar(2)/m; 0 0 0 0;0 0 0 0];
F = eye(length(xbar))+Aa*h+(Aa^2*h^2)/factorial(2)+(Aa^3*h^3)/factorial(3)+(Aa^4*
    h^4)/factorial(4)+(Aa^5*h^5)/factorial(5)+(Aa^6*h^6)/factorial(6); %State
    transition matrix
H = [1 0 0 0]; %Observation function
Q_variance=0.01^2; %process noise variance, assume a piecewise constant white
    noise
Q=piecewise_white_noise(4,Q_variance,h);
%Q=kron(Q1,eye(2));
R = obsNoise; %Measurement Noise

for i = 2:length(obs)
 %Prediction step
 f={@(x1,x2,x3,x4,t) (x1+x2*t);
    @(x1,x2,x3,x4,t) (x2+(-x4/m*x2-x3/m*x1)*t)
    @(x1,x2,x3,x4,t) (x3)
    @(x1,x2,x3,x4,t) (x4)};
    xbar(1,1)=f{1}(xbar(1),xbar(2),xbar(3),xbar(4),h);
    xbar(2,1)=f{2}(xbar(1),xbar(2),xbar(3),xbar(4),h);
    xbar(3,1)=f{3}(xbar(1),xbar(2),xbar(3),xbar(4),h);
    xbar(4,1)=f{4}(xbar(1),xbar(2),xbar(3),xbar(4),h);

 Aa = [0 1 0 0;-xbar(3)/m -xbar(4)/m -xbar(1)/m -xbar(2)/m; 0 0 0 0;0 0 0 0];
 F = eye(length(xbar))+Aa*h+(Aa^2*h^2)/factorial(2)+(Aa^3*h^3)/factorial(3)+(Aa
    ^4*h^4)/factorial(4)+(Aa^5*h^5)/factorial(5)+(Aa^6*h^6)/factorial(6); %State
    transition matrix

 P = F*P*F' + Q;

 % Observation update
 K = P*H'*inv(H*P*H'+R);
 y=obs(i) - H*xbar; %residual
 xbar = xbar + K*(y);
 P = P - K*H*P;
 xbarEstimate(:,i) = xbar(:,1);
 varEstimate(:,i) = diag(P);
 KalmanGain(:,i) = K;
 Residual(:,i)=y;
end

%% Position and velocity plot
figure('Renderer', 'painters', 'Position', [15 15 900 700])
subplot(2,2,1);
plot(time,trueTrajectory(:,1),'k','linewidth',2);
hold on;
plot(time,obs,'bo','markerfacecolor','b','markersize',3)
plot(time,xbarEstimate(1,:),'r','linewidth',3);
lower_x1 = xbarEstimate(1,:)+2*sqrt(varEstimate(1,:));
upper_x1 = xbarEstimate(1,:)-2*sqrt(varEstimate(1,:));
ciplot(lower_x1,upper_x1,time,'b')
tix=get(gca,'ytick')';
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
```

```
axis([0 30 ,-1 1.5])
grid on;
legend 'True Trajectory' 'Simulated Noise' 'KF Estimate' 'Confidence Interval'
xlabel('Time (s)')
ylabel('Position (m)')

subplot(2,2,2)
plot(time,trueTrajectory(:,2),'k','linewidth',3);
hold on;
plot(time,xbarEstimate(2,:),'r','linewidth',3);
axis([0 30 ,-1 1.5])
tix=get(gca,'ytick');
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
lower_x2 = xbarEstimate(2,:)+2*sqrt(varEstimate(2,:));
upper_x2 = xbarEstimate(2,:)-2*sqrt(varEstimate(2,:));
ciplot(lower_x2,upper_x2,time,'b')
grid on;
legend 'True Trajectory' 'KF Estimate' 'Confidence Interval'
xlabel('Time (s)')
ylabel('Velocity (m/s)')

%% Spring Constant and Damping coefficient plots
subplot(2,2,3);
plot(time,xbarEstimate(3,:),'-r','linewidth',2);
% tix=get(gca,'ytick');
% set(gca,'fontsize',10)
% set(gca,'yticklabel',num2str(tix,'%.1f'))
axis([0 30 ,0 20])
grid on;
legend 'KF Estimate';
xlabel('Time (s)'); ylabel('Spring Constant (N/m)')

subplot(2,2,4)
plot(time,xbarEstimate(4,:),'-r','linewidth',2);
tix=get(gca,'ytick');
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
axis([0 30 ,0 10])
grid on;
legend 'KF Estimate';
xlabel('Time (s)');
ylabel('Damping Coefficient (N/ms^-1)')

rmse_1 = sqrt(sum((xbarEstimate(3,:)-k.*ones(1,length(time))).^2)./numel(
    xbarEstimate(1,:)))
rmse_2 = sqrt(sum((xbarEstimate(4,:)-b.*ones(1,length(time))).^2)./numel(
    xbarEstimate(2,:)))
rmse_3 = sqrt(mean(xbarEstimate(3,:)-omega.*ones(1,length(time))).^2)
rmse_4 = sqrt(mean(xbarEstimate(4,:)-gamma.*ones(1,length(time))).^2)
```

A.8 Parameter estimation on Damped Harmonic Oscillator using Unscented Kalman Filter

```
clc; clear all; close all;
%some physical properties
global kspring bspring mass
kspring=5;
bspring=3;
mass=10;

%some parameters
Nk=5000; %Number of iterations
ndim=4;
n_meas=1; %we are only measuring the position with a displacement sensor
omega=sqrt(kspring/mass);
gamma=bspring/mass;

sensor_x_variance=0.01^2;
dt=0.01; %time step

Q_variance=0.01; %process noise variance, assume a piecewise constant white noise
Q1=piecewise_white_noise(2,Q_variance,dt);
Q=kron(Q1,eye(2));
R=diag([sensor_x_variance]); % Covariance for the measurement matrix

%initial state estimate
%a close to the initial measurement determines the initial state
x_initial_actual=[0; 1; 10; 10];
x=[-2;1.5; 0; 0]; %this is how the Kalman filter is initialized
P_max=10;
P=P_max*eye(4);

%choosing the sigma points, the covariance matrix is P
alpha=0.1; beta=2; kappa=3-length(x);
data=zeros(Nk,7); %placeholder for all the variables
%define the nonlinear process function called f
f={@(x1,x2,x3,x4,t) (x1+x2*t);
    @(x1,x2,x3,x4,t) (x2+(-(x3/mass)*x1-(x4/mass)*x2)*t)
    @(x1,x2,x3,x4,t) (x3)
    @(x1,x2,x3,x4,t) (x4)};
y_sigma=zeros(ndim,2*ndim+1);

time=dt:dt:Nk*dt;

%we first simulate the behavior of the harmonic oscillator yielding true
%values
[tt,model]=ode45('damped_ho',time,x_initial_actual(1:2));

for k=1:Nk

%sigma points
    [chi,scalefactor,wm,wc]=vandermeer_sigma2(x,P,alpha,beta,kappa);
%predict step
```

```
51
52  for kk=1:2*ndim+1
53  y_sigma(1,kk)=f{1}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),dt);
54  y_sigma(2,kk)=f{2}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),dt);
55  y_sigma(3,kk)=f{3}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),dt);
56  y_sigma(4,kk)=f{4}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),dt);
57  end
58
59  %here we perform the unscented transform
60  x=sum(repmat(wm,ndim,1).*y_sigma,2);
61  P_temp=zeros(ndim,ndim);
62  for kk=1:2*ndim+1
63  P_temp=P_temp+wc(kk)*(y_sigma(:,kk)-x)*(y_sigma(:,kk)-x)';
64  end
65  P=P_temp+Q;
66
67  %define the nonlinear measurement function denoted h
68  h={@(y1,y2,y3,y4) (y1);
69       };
70
71  %Map the predicted prior to the measurement space
72  %the mapped values are stored in the variable ZZ
73  ZZ=zeros(n_meas,2*ndim+1);
74  for kk=1:2*ndim+1
75  ZZ(1,kk)=h{1}(y_sigma(1,kk),y_sigma(2,kk));
76  end
77
78  %Applying the unscented transform in the measurement space
79  ZZmean=sum(repmat(wm,n_meas,1).*ZZ,2); %this is our mu_z
80  Pz_temp=zeros(n_meas,n_meas);
81  for kk=1:2*ndim+1
82  Pz_temp=Pz_temp+wc(kk)*(ZZ(:,kk)-ZZmean)*(ZZ(:,kk)-ZZmean)';
83  end
84  Pz=Pz_temp+R;
85
86  %measurement vector z must come here
87  z=model(k,1)+sqrt(sensor_x_variance)*randn; %this is the position measurement
88  y=z-ZZmean; %residual
89
90  %Finding the Kalman gain
91  Kg_temp=zeros(ndim,n_meas);
92  for kk=1:2*ndim+1
93  Kg_temp=Kg_temp+wc(kk)*(y_sigma(:,kk)-x)*(ZZ(:,kk)-ZZmean)';
94  end
95  Kg=Kg_temp*inv(Pz);
96  x=x+Kg*y;
97  P=P-Kg*Pz*Kg';
98
99  data(k,:)=[z(1) x(1) x(2) x(3) x(4) P(1,1) P(2,2)];
100 end
101 %
102 figure('Renderer', 'painters', 'Position', [15 15 900 700])
103 subplot(2,2,1);
104 plot(time,data(:,1),'ro'); hold on;
105 plot(time,data(:,2),'b-','linewidth',2);
```

```
tix=get(gca,'ytick')';
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
axis([0 30,-1 1.5])
legend('Measured displacement','Filtered displacement','location','northeast');
xlabel('Time (s)'); ylabel('Position (m)');
grid on; hold off;
%
subplot(2,2,2)
plot(time,model(:,2),'ro'); grid on; hold on;
plot(time,data(:,3),'b-','linewidth',2);
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
axis([0 30,-1 1.5])
legend('True speed','Filtered speed','location','northeast');
xlabel('Time (s)'); ylabel('Velocity (m/s)');

subplot(2,2,3)
plot(time,data(:,4),'b-','linewidth',2);
%axis([0 30],[0 10])
set(gca,'fontsize',10)
% set(gca,'yticklabel',num2str(tix,'%.1f'))
xlabel('Time (s)'); ylabel('Spring Constant (N/m)');
grid on;

subplot(2,2,4)
%plot(gamma.*ones(1,Nk),'r','linewidth',4); hold on;
plot(time,data(:,5),'b-','linewidth',2);
%axis([0 30],[0 10])
set(gca,'fontsize',10)
% set(gca,'yticklabel',num2str(tix,'%.1f'))
xlabel('Time (s)'); ylabel('Damping Coefficient (Ns/m)');
grid on;
%%%%%%%%%%
%%
%%

rmse_1 = sqrt(sum((data(:,4)-kspring.*ones(numel(data(:,3)),1)).^2)./numel(data
    (:,1)))
rmse_2 = sqrt(sum((data(:,5)-bspring.*ones(numel(data(:,3)),1)).^2)./numel(data
    (:,1)))
% rmse_2 = sqrt((mean(data(:,3)-model(:,2)).^2)./numel(data(:,1)))
% rmse_3 = sqrt((mean(data(:,4)'-kspring.*ones(1,Nk)).^2)./numel(data(:,1)))
% rmse_4 = sqrt((mean(data(:,5)'-bspring.*ones(1,Nk)).^2)./numel(data(:,1)))
```

A.9 Duffing oscillator function file

```
function xdot=duffing(t,x)

global gamma omega beta AMP rho

```

```
xdot(1)=x(2);
xdot(2)=-gamma*x(2)+omega^2*x(1)-beta*x(1)^3+AMP*cos(rho*t);

xdot=xdot';

end
```

A.10 Simulating the Duffing oscillator

```
clc
close all
clear all

%% Simulating Duffing Oscillator
global gamma omega beta F rho

gamma=0.3;  %Damping Coefficient
omega=-1;    %Stiffness Coefficient
beta=1;  %Stiffness Coefficient
F=0.5;  %Amplitude of driving force
rho=1.25;  %Frequency of driving force

[t x]=ode45(@duffing,0:2*pi/rho/100:4000,[1 0]);

dst=t(1:6000);  %time
yn=x(1:6000,1); %position
yv=x(1:6000,2); %velocity

%For Poincare Section
for i=200:100:78000
      n=(i-100)/100;
      x1(n)=x(i,1);
      x2(n)=x(i,2);
end

%% Plotting
subplot(2,2,[1 2])
plot(dst,yn,'-')
axis([0 300,-2 2]);
tix=get(gca,'ytick')';
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
xlabel('Time (s)')
ylabel('$\theta (rad)$','interpreter','latex')
legend('True Position','Position with noise','Linear Kalman Filter')

%Poincare Section
subplot(2,2,3)
plot(x1(:),x2(:),'*')
axis([-1.5 1.3,-0.5 1.0]);
tix=get(gca,'ytick')';
```

```
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
xlabel('\theta (rad)')
ylabel('$\dot{\theta}$ (rad/s)','interpreter','latex')

%Phase space
subplot(2,2,4)
plot(yn,yv,'b')
axis([-1.8 1.8,-1.2 1.2]);
tix=get(gca,'ytick')';
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
xlabel('\theta (rad)')
ylabel('$\dot{\theta}$ (rad/s)','interpreter','latex')
```

A.11 Simulating the application of EKF to Duffing oscillator

```
clc; clear all;

%% Simulating Duffing Oscillator
global gamma omega beta AMP rho

gamma=0.3;  %Damping Coefficient
omega=-1;    %Stiffness Coefficient k1
beta=1;  %Stiffness Coefficient k2
rho=1.25;  %Frequency of driving force
AMP=0.5;  %Amplitude of driving force
T = 100; %Final time value
dt = 2*pi/rho/T; % time step
time = 0:dt:T; %Full time scale
[t,trueTrajectory]=ode45(@duffing,time,[0 1]);
obsNoise = 0.5^2; %Define observation noise level
obs_pos = trueTrajectory(:,1); %First state is observable
n_obs_pos = obs_pos+obsNoise*randn(size(obs_pos)); %Add noise
yx=trueTrajectory(:,1); %True Position vector
yv=trueTrajectory(:,2); %True Velocity vector

%% Applying Extended Kalman Filter
xbar=[0;1]; %Initializing the EKF
xbarEstimate=xbar;
P = diag([1,1]); %Initial covariance
varEstimate = diag(P); %Initial state variance
H = [1 0]; %Observation function
Q_variance=2.5^2; %process noise variance, assume a piecewise constant white
    noise
Q=piecewise_white_noise(2,Q_variance,dt);
D = obsNoise; %Measurement Noise
var_c = 5^2; %control input noise

```

```
% Defining the nonlinear function
 f={@(x1,x2,t) (x1+x2*t);
    @(x1,x2,t) (x2+(-gamma*x2+omega^2*x1-beta*x1^3+AMP*cos(rho*t))*t)
    };

for i = 2:length(n_obs_pos)
%Prediction step
for kk=1:numel(xbar)
xbar(1)=f{1}(xbar(1),xbar(2),dt);
xbar(2)=f{2}(xbar(1),xbar(2),dt);
end
Aa = [0 1;-omega^2-3*beta*xbar(1)^2 -gamma];
F = eye(length(xbar))+Aa*dt+(Aa^2*dt^2)/factorial(2)+(Aa^3*dt^3)/factorial(3); %
    State transition matrix
P = diag([P(1,1),P(2,2)]);
M = [0 0;0 var_c];
V = [0 0;0 -AMP*sin(rho*dt)];
P = F*P*F'+V*M*V';
% Observation update
K=(P*H')/(H*P*H'+D);
y=n_obs_pos(i) - H*xbar; %Residual
xbar = xbar + K*(y);
P = P-K*H*P;
xbarEstimate(:,i) = xbar;
varEstimate(:,i) = diag(P);
residual(:,i)=y;
end

%% Position and velocity plot
subplot(2,1,1);
%plot(time,yx,'k','linewidth',2); % Plotting the true trajectory
plot(time,n_obs_pos,'o','markerfacecolor','bl','markersize',1); hold on;
lower_x1 = xbarEstimate(1,:)+2*sqrt(varEstimate(1,:));
upper_x1 = xbarEstimate(1,:)-2*sqrt(varEstimate(1,:));
ciplot(lower_x1,upper_x1,time,'b',0.05)
plot(time,xbarEstimate(1,:),'r','linewidth',2);
%set(gca,'fontsize',10)
legend({'Noisy angular position', 'Confidence Interval', 'EKF Estimate'},'
    FontSize',7)
xlabel('Time (s)')
hl = ylabel('$\theta~(rad)$');
set(hl, 'Interpreter', 'latex');
dim = [.14 .82 .1 .1];
str = 'a)';
annotation('textbox',dim,'String',str,'FitBoxToText','on','LineStyle','none')

grid on;
%
subplot(2,1,2)
plot(time,yv,'--','linewidth',1); % Plotting the true trajectory
%plot(time,n_obs_vel,'o','markerfacecolor','bl','markersize',1)
hold on;
lower_x2 = xbarEstimate(2,:)+2*sqrt(varEstimate(2,:));
upper_x2 = xbarEstimate(2,:)-2*sqrt(varEstimate(2,:));
ciplot(lower_x2,upper_x2,time,'b',0.05)
```

```
plot(time,xbarEstimate(2,:),'bl','linewidth',2);
legend({'True angular velocity','Confidence interval','EKF Estimate'},'FontSize'
    ,7)
xlabel('Time (s)')
hl = ylabel('$\dot{\theta}~(rad/s)$');
set(hl, 'Interpreter', 'latex');
grid on;

%%% RMSE plots
for res = 1:numel(xbarEstimate(1,:))
rmse_x1(res) = sqrt(sum(residual.^2)/res);
end
figure;
subplot(1,2,1)
plot(rmse_x1,'bl','linewidth',2)
xlabel('Number of iterations'); ylabel('RMSE')

for res2 = 1:numel(xbarEstimate(2,:))
rmse_x2(res2) = sqrt(sum((xbarEstimate(2,:)-yv')/res2));
end
subplot(1,2,2)
plot(rmse_x2,'bl','linewidth',2)
xlabel('Number of iterations'); ylabel('RMSE')
%
rmse_x1 = sqrt(sum((xbarEstimate(1,:)-yx').^2)/numel(time))
rmse_x2 = sqrt(sum((xbarEstimate(2,:)-yv').^2)/numel(time))
%%
```

A.12 Parameter estimation in Duffing oscillator using Extended Kalman Filter

```
clc; clear all;

%% Simulating Duffing Oscillator
global gamma omega beta AMP rho

gamma=0.3;  %Damping Coefficient
omega=-1;    %Stiffness Coefficient k1
beta=1;  %Stiffness Coefficient k2
rho=1.25;  %Frequency of driving force
AMP=0.5;  %Amplitude of driving force
T = 100; %Final time value
dt = 2*pi/rho/T; % time step
time = 0:dt:T; %Full time scale
[t,trueTrajectory]=ode45(@duffing,time,[0 1]);
obsNoise = 0.5^2; %Define observation noise level
obs_pos = trueTrajectory(:,1); %First state is observable
n_obs_pos = obs_pos+obsNoise*randn(size(obs_pos)); %Add noise
obs_vel = trueTrajectory(:,2); %Second state is observable
n_obs_vel = obs_vel+obsNoise*randn(size(obs_vel)); %Add noise
```

```
obs_epsilon=beta.*ones(numel(obs_vel),1);
yx=trueTrajectory(:,1); %Position vector
yv=trueTrajectory(:,2); %Velocity vector
%plot(yx,yv)
%plot(n_obs,yv)

tic
%% Applying Extended Kalman Filter
xbar=[0;1;0]; %Initializing the EKF
xbarEstimate=xbar;
P = diag([0.1,0.1,0.1]); %Initial covariance
varEstimate = diag(P); %Initial state variance
H = [1 0 0]; %Observation function
Q_variance=2.5^2; %process noise variance, assume a piecewise constant white
    noise
Q=piecewise_white_noise(3,Q_variance,dt);
D = obsNoise; %Measurement Noise

for i = 2:length(n_obs_pos)
% Defining the nonlinear process function
 f={@(x1,x2,x3,t) (x1+x2*t);
    @(x1,x2,x3,t) (x2+(-gamma*x2+omega^2*x1-x3*x1^3+AMP*cos(rho*t))*t)
    @(x1,x2,x3,t) (x3);
   };
    %Prediction step
    xbar(1)=f{1}(xbar(1),xbar(2),xbar(3),dt);
    xbar(2)=f{2}(xbar(1),xbar(2),xbar(3),dt);
    xbar(3)=f{3}(xbar(1),xbar(2),xbar(3),dt);

 Aa = [0 1 0;omega^2-3*xbar(3)*xbar(1)^2 -gamma -xbar(1)^3;0 0 0];
 F = eye(length(xbar))+Aa*dt+(Aa^2*dt^2)/factorial(2)+(Aa^3*dt^3)/factorial(3); %
    State transition matrix
 P = F*P*F'+Q;
 % Observation update
 K=(P*H')/(H*P*H'+D);
 xbar = xbar + K*(n_obs_pos(i) - H*xbar);
 P = P-K*H*P;
 xbarEstimate(:,i) = xbar;
 varEstimate(:,i) = diag(P);
 KalmanGain(:,i)=K;
end

toc

%% Position and velocity plot
figure('Renderer', 'painters', 'Position', [15 15 900 700])
subplot(2,2,1);
plot(time,yx,'k','linewidth',2); % Plotting the true trajectory
plot(time,n_obs_pos,'o','markerfacecolor','bl','markersize',1)
hold on;
plot(time,xbarEstimate(1,:),'r','linewidth',2);
tix=get(gca,'ytick')';
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
legend({'Noisy angles', 'EKF Estimate'},'FontSize',7)
```

```
xlabel('Time (s)')
hl = ylabel('$\theta~(rad)$');
set(hl, 'Interpreter', 'latex');
grid on;
%
subplot(2,2,2)
plot(time,yv,'k','linewidth',2); % Plotting the true trajectory
hold on;
plot(time,xbarEstimate(2,:),'bl','linewidth',2);
tix=get(gca,'ytick')';
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))
legend({'True angular velocity', 'EKF Estimate'},'FontSize',7)
xlabel('Time (s)')
hl = ylabel('$\dot{\theta}~(rad/s)$');
set(hl, 'Interpreter', 'latex');
grid on;
%
subplot(2,2,[3 4])
%plot(time,epsilon.*ones(1,length(time)),'r','linewidth',2); hold on;
plot(time,xbarEstimate(3,:),'b','linewidth',2);
%legend({'True value' 'EKF Estimate'},'FontSize',7)
xlabel('Time (s)')
hl = ylabel('$\beta$');
set(hl, 'Interpreter', 'latex');
grid on;
tix=get(gca,'ytick')';
set(gca,'fontsize',10)
set(gca,'yticklabel',num2str(tix,'%.1f'))

figure; subplot(3,1,1)
plot(time,KalmanGain(1,:),'b','linewidth',2);
xlabel('Time (s)')
hl = ylabel('Gain in $\theta$');
set(hl, 'Interpreter', 'latex');
grid on;

subplot(3,1,2)
plot(time,KalmanGain(2,:),'b','linewidth',2);
xlabel('Time (s)')
hl = ylabel('Gain in $\dot{\theta}$');
set(hl, 'Interpreter', 'latex');
grid on;

subplot(3,1,3)
plot(time,KalmanGain(3,:),'b','linewidth',2);
xlabel('Time (s)')
hl = ylabel('Gain in $\beta$');
set(hl, 'Interpreter', 'latex');
grid on;

%%%%%%%
% rmse_1 = sqrt((sum(xbarEstimate(1,:))-sum(yx)').^2/numel(yx))
% rmse_2 = sqrt((sum(xbarEstimate(2,:))-sum(yv)').^2/numel(yx))
```

```
rmse_3 = sqrt(sum((xbarEstimate(3,:)-beta.*ones(1,length(time))).^2)./numel(
    xbarEstimate(2,:)))
%
```

A.13 Parameter estimation in Duffing oscillator using Unscented Kalman Filter

```
clc; clear all;
global gamma omega epsilon AMP OMEG

gamma=0.3;  %Damping Coefficient
omega=-1;    %Stiffness Coefficient k1
epsilon=1;  %Stiffness Coefficient k2
OMEG=1.25;  %Frequency of driving force
AMP=0.5;  %Amplitude of driving force

%some parameters
Nk=100; %Number of iterations
ndim=3;
n_meas=1; %we are only measuring the position with a displacement sensor
% % omega=sqrt(kspring/mass);
% % gamma=bspring/mass;

sensor_x_variance=0.5^2;
dt=2*pi/OMEG/Nk; %time step

Q_variance=5.2; %process noise variance, assume a piecewise constant white noise
Q=piecewise_white_noise(3,Q_variance,dt);
Q=kron(Q,eye(1));

R=diag([sensor_x_variance]); % Covariance for the measurement matrix

%initial state estimate
%a close to the initial measurement determines the initial state
x_initial_actual=[0; 1; 10; 10; 20];
x=[0;1; 10]; %this is how the Kalman filter is initialized
P_max=10;
P=P_max*eye(3);

%choosing the sigma points, the covariance matrix is P
alpha=0.1; beta=2; kappa=3-length(x);
data=zeros(Nk,10); %placeholder for all the variables
%define the nonlinear process function called f
f={@(x1,x2,x3,t) (x1+x2*t);
    @(x1,x2,x3,t) (x2+(-gamma*x2+omega^2*x1-x3*x1^3+AMP*cos(OMEG))*t)
    @(x1,x2,x3,t) (x3)  %epsilon
    };
y_sigma=zeros(ndim,2*ndim+1);

time=0:dt:Nk;
```

```
44
45 %we first simulate the behavior of the harmonic oscillator yielding true
46 %values
47 [tt,model]=ode45(@duffing,time,x_initial_actual(1:2));
48
49 tic
50 for k=1:numel(model(:,1))
51
52 %sigma points
53     [chi,scalefactor,wm,wc]=vandermeer_sigma2(x,P,alpha,beta,kappa);
54 %predict step
55
56 for kk=1:2*ndim+1
57 y_sigma(1,kk)=f{1}(chi(1,kk),chi(2,kk),chi(3,kk),dt);
58 y_sigma(2,kk)=f{2}(chi(1,kk),chi(2,kk),chi(3,kk),dt);
59 y_sigma(3,kk)=f{3}(chi(1,kk),chi(2,kk),chi(3,kk),dt);
60 end
61
62 %here we perform the unscented transform
63 x=sum(repmat(wm,ndim,1).*y_sigma,2);
64 P_temp=zeros(ndim,ndim);
65 for kk=1:2*ndim+1
66 P_temp=P_temp+wc(kk)*(y_sigma(:,kk)-x)*(y_sigma(:,kk)-x)';
67 end
68 P=P_temp+Q;
69
70 %define the nonlinear measurement function denoted h
71 h={@(y1,y2,y3) (y1);
72     };
73
74 %Map the predicted prior to the measurement space
75 %the mapped values are stored in the variable ZZ
76 ZZ=zeros(n_meas,2*ndim+1);
77 for kk=1:2*ndim+1
78 ZZ(1,kk)=h{1}(y_sigma(1,kk),y_sigma(2,kk),y_sigma(3,kk));
79 end
80
81 %Applying the unscented transform in the measurement space
82 ZZmean=sum(repmat(wm,n_meas,1).*ZZ,2); %this is our mu_z
83 Pz_temp=zeros(n_meas,n_meas);
84 for kk=1:2*ndim+1
85 Pz_temp=Pz_temp+wc(kk)*(ZZ(:,kk)-ZZmean)*(ZZ(:,kk)-ZZmean)';
86 end
87 Pz=Pz_temp+R;
88
89 %measurement vector z must come here
90 z=model(k,1)+sqrt(sensor_x_variance)*randn; %this is the position measurement
91 y=z-ZZmean; %residual
92
93 %Finding the Kalman gain
94 Kg_temp=zeros(ndim,n_meas);
95 for kk=1:2*ndim+1
96 Kg_temp=Kg_temp+wc(kk)*(y_sigma(:,kk)-x)*(ZZ(:,kk)-ZZmean)';
97 end
98 Kg=Kg_temp*inv(Pz);
```

```
x=x+Kg*y;
P=P-Kg*Pz*Kg';

data(k,:)=[z(1) x(1) x(2) x(3) P(1,1) P(2,2) Kg(1) Kg(2) Kg(3) P(3,3)];
end

toc

%%
subplot(2,2,1)
plot(time,data(:,1),'ro'); hold on;
plot(time,data(:,2),'b-','linewidth',2);
%set(gca,'fontsize',10)
legend({'Raw data', 'UKF Estimate'},'FontSize',7);
%axis([0 100 -5 5])
xlabel('Time (s)')
ylabel('$\theta$ (rad)','interpreter','latex')
grid on; hold off;
%
subplot(2,2,2)
plot(time,model(:,2),'ro'); hold on;
plot(time,data(:,3),'b-','linewidth',2);
%set(gca,'fontsize',10)
legend({'Raw data', 'UKF Estimate'},'FontSize',7)
%axis([0 100 -5 5])
xlabel('Time (s)');
ylabel('$\dot{\theta}$ (rad/s)','interpreter','latex')
grid on;

subplot(2,2,[3 4])
plot(time,data(:,4),'b-','linewidth',2); hold on;
xlabel('Time (s)');
ylabel('$\beta$','interpreter','latex')
axis([0 100 -10 10])
grid on;

%% Kalman Gain plots
figure; subplot(3,1,1)
plot(time(1:200),data(1:200,7),'b-','linewidth',2); hold on;
ylabel('Gain $\theta$','interpreter','latex')
xlabel('Time (s)');
grid on;

subplot(3,1,2)
plot(time(1:200),data(1:200,8),'b-','linewidth',2); hold on;
xlabel('Time (s)');
ylabel('Gain $\dot{\theta}$','interpreter','latex')
grid on;

subplot(3,1,3)
plot(time(1:200),data(1:200,7),'b-','linewidth',2); hold on;
xlabel('Time (s)');
ylabel('Gain $\beta$','interpreter','latex')
grid on;

```

```
%% Error covariance
figure; subplot(3,1,1)
plot(time(1:200),data(1:200,5),'b-','linewidth',2); hold on;
xlabel('Time (s)');
ylabel('$P_{(1,1)}$','interpreter','latex')
grid on;

subplot(3,1,2)
plot(time(1:200),data(1:200,6),'b-','linewidth',2); hold on;
xlabel('Time (s)');
ylabel('$P_{(2,2)}$','interpreter','latex')
grid on;

subplot(3,1,3)
plot(time(1:200),data(1:200,10),'b-','linewidth',2); hold on;
xlabel('Time (s)');
ylabel('$P_{(3,3)}$','interpreter','latex')
grid on;

rmse_1=sqrt(sum((data(:,4)-epsilon).^2)./numel(data(:,1)))


%%%%%%%%%%%
%%%
%%%
```

A.14 Wilberforce pendulum function file

```
function xdot=wilberforce_func(t,x)

global omega mass inertia epsilon

xdot(1)=x(2);
xdot(2)=(-omega^2*x(1)-0.5/mass*epsilon*x(3))*t;
xdot(3)=x(4);
xdot(4)=(-omega^2*x(3)-0.5/inertia*epsilon*x(1))*t;

xdot=xdot';

end
```

A.15 Simulating the application of Kalman Filter to the Wilberforce pendulum

```
clc; clear all; close all;

```

```
% Assign model parameters
global omega epsilon mass inertia
omega = 2.314;          % rad.s-1
epsilon = 9.27e-3;      % N
mass = 0.4905;          % kg
inertia = 1.39e-4;      % kg.m2
theta=2*pi;             % Initial angle
z=0;                    % Starting position

% Simulate system
xinit = [0;0;2*pi;0]; % initial value
h = 0.01; % time step
T = 40; %Final time value
time = 0:h:T; %Full time scale
[t,trueTrajectory]=ode45(@wilberforce_func,time,xinit);
obsNoise_z = 0.01; %Noise in position measurement
obsNoise_ang = 0.5; %Noise in angle measurement
obs_z = trueTrajectory(:,1); %First state is observable
obs_z = obs_z+obsNoise_z*randn(size(obs_z)); %Add noise
obs_ang = trueTrajectory(:,3);
obs_ang = obs_ang+obsNoise_ang*randn(size(obs_ang)); %Add noise
%plotyy(t,obs_z,t,obs_ang)

%% Linear Kalman Filter Parameters
xbar=[0;0;2*pi;0];
xbarEstimate = xbar; %Initial state
P = 1*eye(length(xbar)); %Initial covariance
varEstimate = diag(P); %Initial state variance
A = [0 1 0 0;-omega^2 1 -0.5/mass*epsilon 0;0 0 0 1; -0.5/mass*epsilon 0 -omega^2
     1];
F = eye(length(xbar))+h*A; %State transition matrix
H = [1 0 0 0;0 0 1 0]; %Observation function
variance_process_x=1.5^2; %process variance in position z
variance_process_y=80^2; %process variance in angle theta
Q1=piecewise_white_noise(2,variance_process_x,h);
Q2=piecewise_white_noise(2,variance_process_y,h);
Q=zeros(4,4);
Q(1:2,1:2)=Q1;
Q(3:4,3:4)=Q2;

D = [obsNoise_z 0;0 obsNoise_ang];

% Ob = obsv(A,H);
%% If rank = ndim of state vector the system is observable
% rank(Ob)
%
for i = 2:length(time)
 % Prediction step
 xbar = F*xbar;
 P = F*P*F' + Q;
 % Observation update
 K = (P*H')/(H*P*H'+D); % aka K = ...
 P*H'*inv(H*P*H'+D);
 xbar = xbar + K*([obs_z(i);obs_ang(i)] - H*xbar);
 P = P - K*H*P;
```

```
 xbarEstimate(:,i) = xbar;
 varEstimate(:,i) = diag(P);

end
figure('Renderer', 'painters', 'Position', [15 15 900 700])
subplot(2,1,1);
%plot(time,trueTrajectory(:,1),'k','linewidth',3); hold on;
plot(time,obs_z,'bo','markerfacecolor','b','markersize',2); hold on;
plot(time,xbarEstimate(1,:),'r','linewidth',2);
upper_z = xbarEstimate(1,:)+2*sqrt(varEstimate(1,:));
lower_z = xbarEstimate(1,:)-2*sqrt(varEstimate(1,:));
ciplot(lower_z,upper_z,time,'b',0.08)
set(gca,'fontsize',10)
axis([0 T -0.1 0.1])
tix=get(gca,'ytick')';
set(gca,'yticklabel',num2str(tix,'%.2f'))
dim = [.14 .82 .1 .1];
str = 'a)';
annotation('textbox',dim,'String',str,'FitBoxToText','on','LineStyle','none')
legend 'Vertical Position' 'KF Estimate'
xlabel('Time (s)')
ylabel('z (m)')
%
subplot(2,1,2)
%plot(time,trueTrajectory(:,3),'k','linewidth',2); hold on;
plot(time,obs_ang,'bo','markerfacecolor','b','markersize',2); hold on;
plot(time,xbarEstimate(3,:),'r','linewidth',3);
% upper_theta = xbarEstimate(3,:)+2*sqrt(varEstimate(3,:));
% lower_theta = xbarEstimate(3,:)-2*sqrt(varEstimate(3,:));
% ciplot(lower_theta,upper_theta,time,'b',0.1)
tix=get(gca,'ytick')';
axis([0 T -6 6])
set(gca,'yticklabel',num2str(tix,'%.2f'))
legend 'Angular Position' 'KF Estimate'
dim = [.14 .35 .1 .1];
str = 'b)';
annotation('textbox',dim,'String',str,'FitBoxToText','on','LineStyle','none');
xlabel('Time (s)')
ylabel('\theta (rad)')
```

A.16 Parameter estimation in Wilberforce pendulum using Extended Kalman Filter

```
clc; clear all; close all;

% Assign model parameters
global omega epsilon mass inertia
omega = 2.314;          % rad.s-1
epsilon = 9.27e-3;      % N
mass = 0.4905;          % kg
```

```
inertia = 1.39e-4;      % kg.m2
theta=2*pi;             % Initial angle
z=0;                    % Starting position

% Simulate system
xinit = [0;0;2*pi;0]; % initial value
h = 0.02; % time step
T = 100; %Final time value
time = 0:h:T; %Full time scale
[t,trueTrajectory]=ode45(@wilberforce_func,time,xinit);
obsNoise_z = 0.01; %Noise in position measurement
obsNoise_ang = 0.1; %Noise in angle measurement
obs_z = trueTrajectory(:,1); %First state is observable
obs_z = obs_z+obsNoise_z*randn(size(obs_z)); %Add noise
obs_ang = trueTrajectory(:,3);
obs_ang = obs_ang+obsNoise_ang*randn(size(obs_ang)); %Add noise
%plotyy(t,obs_z,t,obs_ang)

%% Extended Kalman Filter Parameters
xbar=[0;1;2*pi;1;1;1;1];
xbarEstimate = xbar; %Initial state
P = 0.1*eye(length(xbar)); %Initial covariance
varEstimate = diag(P); %Initial state variance
H = [1 0 0 0 0 0 0;0 0 1 0 0 0 0]; %Observation function
Q=zeros(7,7);
variance_process=2.5^2; %process variance in position z
% variance_process_theta=0.2^2; %process variance in position z
%Q=piecewise_white_noise(7,variance_process,h);
%Q2=piecewise_white_noise(2,variance_process_theta,h);
% Q(1:2,1:2)=Q1;
% Q(3:4,3:4)=Q2;
s=0.01;
G=@(t)[t^2/2;t;t^2/2;t;0.01;0.01;0.01];
Gc=@(t)[t^2/2 t t^2/2 t 0.01 0.01 0.01];
Q=variance_process*G(h)*Gc(h);
D = [obsNoise_z 0;0 obsNoise_ang];
%
for i = 2:length(time)
 % Defining nonlinear function
 f={@(x1,x2,x3,x4,x5,x6,x7,t) (x1+x2*t);
    @(x1,x2,x3,x4,x5,x6,x7,t) (x2+(-x5*x1-x7/2/mass*x3)*t);
    @(x1,x2,x3,x4,x5,x6,x7,t) (x3+x4*t);
    @(x1,x2,x3,x4,x5,x6,x7,t) (x4+(-x6*x3-x7/2/inertia*x1)*t);
    @(x1,x2,x3,x4,x5,x6,x7,t) (x5);
    @(x1,x2,x3,x4,x5,x6,x7,t) (x6);
    @(x1,x2,x3,x4,x5,x6,x7,t) (x7)};
    % Prediction step
    xbar(1)=f{1}(xbar(1),xbar(2),xbar(3),xbar(4),xbar(5),xbar(6),xbar(7),h);
    xbar(2)=f{2}(xbar(1),xbar(2),xbar(3),xbar(4),xbar(5),xbar(6),xbar(7),h);
    xbar(3)=f{3}(xbar(1),xbar(2),xbar(3),xbar(4),xbar(5),xbar(6),xbar(7),h);
    xbar(4)=f{4}(xbar(1),xbar(2),xbar(3),xbar(4),xbar(5),xbar(6),xbar(7),h);
    xbar(5)=f{5}(xbar(1),xbar(2),xbar(3),xbar(4),xbar(5),xbar(6),xbar(7),h);
    xbar(6)=f{6}(xbar(1),xbar(2),xbar(3),xbar(4),xbar(5),xbar(6),xbar(7),h);
    xbar(7)=f{7}(xbar(1),xbar(2),xbar(3),xbar(4),xbar(5),xbar(6),xbar(7),h);
  A = [0 1 0 0 0 0 0;
```

```
      -xbar(5) 0 -0.5/mass*xbar(7) 0 -xbar(1) 0 -0.5/mass*xbar(3);
       0 0 0 1 0 0 0;
       -0.5/inertia*xbar(7) 0 -xbar(6) 0 0 -xbar(3) -0.5/inertia*xbar(1);
       0 0 0 0 0 0 0;
       0 0 0 0 0 0 0;
       0 0 0 0 0 0 0];
  F = eye(length(xbar))+h*A; %State transition matrix
  P = F*P*F' + Q;
 % Observation update
 K = (P*H')/(H*P*H'+D); % aka K = ...
 %P*H'*inv(H*P*H'+D);
 xbar = xbar + K*([obs_z(i);obs_ang(i)] - H*xbar);
 P = P - K*H*P;
 xbarEstimate(:,i) = xbar;
 varEstimate(:,i) = diag(P);

end

figure;subplot(3,2,1);
plot(time,obs_z,'bo','markerfacecolor','b','markersize',2); hold on;
plot(time,xbarEstimate(1,:),'r','linewidth',2);
set(gca,'fontsize',10)
legend 'Noisy position in z axis' 'EKF Estimate'
xlabel('Time (s)')
ylabel('z (m)')
%
subplot(3,2,2)
plot(time,obs_ang,'bo','markerfacecolor','b','markersize',2); hold on;
plot(time,xbarEstimate(3,:),'r','linewidth',3);
set(gca,'fontsize',10)
legend 'Noisy angle' 'EKF Estimate'
xlabel('Time (s)')
ylabel('\theta (rad)')

subplot(3,2,3)
plot(time,xbarEstimate(5,:),'r','linewidth',3);
set(gca,'fontsize',10)
xlabel('Time (s)'); ylabel('\omega^2_z')

subplot(3,2,4)
plot(time,xbarEstimate(6,:),'r','linewidth',3);
set(gca,'fontsize',10)
xlabel('Time (s)'); ylabel('\omega^2_\theta')

subplot(3,2,[5 6])
plot(time,xbarEstimate(7,:),'r','linewidth',3);
set(gca,'fontsize',10)
xlabel('Time (s)'); ylabel('\epsilon')
```

A.17 Parameter estimation in Wilberforce pendulum using Unscented Kalman Filter

```
clc; clear all; close all;

%some physical properties
global omega gamma epsilon mass inertia theta z

omega = 2.314;          % m.s-1
gamma = 2.314;          % rad.s-1
epsilon = 9.27e-3;      % N
mass = 0.4905;          % kg
inertia = 1.39e-4;      % kg.m2
theta=2*pi;             % Initial angle
z=0;                    % Starting position

%some parameters
Nk=5000; %Number of iterations
ndim=7;
n_meas=2; %we are only measuring z with a motion sensor

sensor_x_variance=0.001^2;   %Position measurement error
sensor_y_variance=0.01^2;   %Angular measurement error

dt=0.01; %time step

variance_process_x=0.05^2; %process variance in position z
variance_process_y=0.5^2; %process variance in angle theta
Q1=piecewise_white_noise(2,variance_process_x,dt);
Q2=piecewise_white_noise(2,variance_process_y,dt);
Q=zeros(7,7);
Q(1:2,1:2)=Q1;
Q(3:4,3:4)=Q2;
Q=kron(Q,eye(1));

R=[sensor_x_variance 0;0 sensor_y_variance]; % Covariance for the measurement
    matrix

%initial state estimate
%a close to the initial measurement determines the initial state
x_initial_actual=[z;0;theta;0];
x=[z;0;theta;0;10;10;5]; %this is how the Kalman filter is initialized
P_max=10;
P=P_max*eye(7);

%choosing the sigma points, the covariance matrix is P
alpha=0.1; beta=2; kappa=3-length(x);
data=zeros(Nk,10); %placeholder for all the variables

%define the nonlinear process function called f
f={@(x1,x2,x3,x4,x5,x6,x7,t) (x1+x2*t);
   @(x1,x2,x3,x4,x5,x6,x7,t) (x2+(-x5^2*x1-(0.5/mass*x7*x3*x1))*t);
   @(x1,x2,x3,x4,x5,x6,x7,t) (x3+x4*t);
```

```
50      @(x1,x2,x3,x4,x5,x6,x7,t) (x4+(-x6^2*x3-(0.5/inertia*x7*x1*x3))*t);
51      @(x1,x2,x3,x4,x5,x6,x7,t) (x5);
52      @(x1,x2,x3,x4,x5,x6,x7,t) (x6);
53      @(x1,x2,x3,x4,x5,x6,x7,t) (x7);
54      };
55  y_sigma=zeros(ndim,2*ndim+1);
56
57  %we first simulate the behavior of the harmonic oscillator yielding true
58  %values
59  [tt,model]=ode45('wilberforce_func2',dt:dt:Nk*dt,x_initial_actual);
60
61  for k=1:Nk
62
63  %sigma points
64      [chi,scalefactor,wm,wc]=vandermeer_sigma2(x,P,alpha,beta,kappa);
65  %predict step
66
67  for kk=1:2*ndim+1
68  y_sigma(1,kk)=f{1}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),chi(5,kk),chi(6,kk),
        chi(7,kk),dt);
69  y_sigma(2,kk)=f{2}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),chi(5,kk),chi(6,kk),
        chi(7,kk),dt);
70  y_sigma(3,kk)=f{3}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),chi(5,kk),chi(6,kk),
        chi(7,kk),dt);
71  y_sigma(4,kk)=f{4}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),chi(5,kk),chi(6,kk),
        chi(7,kk),dt);
72  y_sigma(5,kk)=f{5}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),chi(5,kk),chi(6,kk),
        chi(7,kk),dt);
73  y_sigma(6,kk)=f{6}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),chi(5,kk),chi(6,kk),
        chi(7,kk),dt);
74  y_sigma(7,kk)=f{7}(chi(1,kk),chi(2,kk),chi(3,kk),chi(4,kk),chi(5,kk),chi(6,kk),
        chi(7,kk),dt);
75  end
76
77  %here we perform the unscented transform
78  x=sum(repmat(wm,ndim,1).*y_sigma,2);
79  P_temp=zeros(ndim,ndim);
80  for kk=1:2*ndim+1
81  P_temp=P_temp+wc(kk)*(y_sigma(:,kk)-x)*(y_sigma(:,kk)-x)';
82  end
83  P=P_temp+Q;
84
85  %define the nonlinear measurement function denoted h
86  h={@(y1,y3) (y1);
87      @(y1,y3) (y3);
88       };
89
90  %Map the predicted prior to the measurement space
91  %the mapped values are stored in the variable ZZ
92  ZZ=zeros(n_meas,2*ndim+1);
93  for kk=1:2*ndim+1
94  ZZ(1,kk)=h{1}(y_sigma(1,kk),y_sigma(2,kk));
95  ZZ(2,kk)=h{2}(y_sigma(4,kk),y_sigma(3,kk));
96  end
97
```

```
%Applying the unscented transform in the measurement space
ZZmean=sum(repmat(wm,n_meas,1).*ZZ,2); %this is our mu_z
Pz_temp=zeros(n_meas,n_meas);
for kk=1:2*ndim+1
Pz_temp=Pz_temp+wc(kk)*(ZZ(:,kk)-ZZmean)*(ZZ(:,kk)-ZZmean)';
end
Pz=Pz_temp+R;

%measurement vector z must come here
zz(1)=model(k,1)+sqrt(sensor_x_variance)*randn; %this is the position measurement
zz(2)=model(k,3)+sqrt(sensor_y_variance)*randn; %this is the position measurement
y=zz'-ZZmean; %residual

%Finding the Kalman gain
Kg_temp=zeros(ndim,n_meas);
for kk=1:2*ndim+1
Kg_temp=Kg_temp+wc(kk)*(y_sigma(:,kk)-x)*(ZZ(:,kk)-ZZmean)';
end
Kg=Kg_temp*inv(Pz);
x=x+Kg*y;
P=P-Kg*Pz*Kg';

data(k,:)=[zz(1) zz(2) x(1) x(2) x(3) x(4) x(5) x(6) x(7) P(1,1)];
end
%
figure; subplot(3,2,1);
plot(data(:,1),'ro'); hold on;
plot(data(:,3),'b-','linewidth',2);
legend('Measured','Filtered','location','northeast');
xlabel('t'); ylabel('z');
grid on; hold off;
%
subplot(3,2,2)
plot(data(:,2),'ro');
xlabel('t'); ylabel('\theta');
grid on; hold on;
plot(data(:,5),'b-','linewidth',2);
legend('\theta','Filtered','location','northeast');
xlabel('t'); ylabel('\theta');

subplot(3,2,3)
plot(data(:,7),'b-','linewidth',2);
xlabel('t'); ylabel('\omega^2_z');
grid on;

subplot(3,2,4)
plot(data(:,8),'b-','linewidth',2);
xlabel('t'); ylabel('\omega^2_\theta');
grid on;

subplot(3,2,[5 6])
plot(data(:,9),'b-','linewidth',2);
xlabel('t'); ylabel('\epsilon');
grid on;
```

Bibliography

[1] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[2] Enrico Canuto, Ph D Carlos Norberto Pérez-Montenegro, Ph D Mauricio Alejandro Lotufo, and Jiawei Wu. Navigation based on extended kalman filter for borea project. 2015.

[3] Hitesh A Patel and Darshak G Thakore. Moving object tracking using kalman filter. *International Journal of Computer Science and Mobile Computing*, 2(4):326–332, 2013.

[4] Yan Xu and Guosheng Zhang. Application of kalman filter in the prediction of stock price. In *International Symposium on Knowledge Acquisition and Modeling (KAM). Atlantis press*, pages 197–198, 2015.

[5] H Haleh, B Akbari Moghaddam, and S Ebrahimijam. A new approach to forecasting stock price with ekf data fusion. *International Journal of Trade, Economics and Finance*, 2(2):109, 2011.

[6] Donald E Catlin. *Estimation, control, and the discrete Kalman filter*, volume 71. Springer Science & Business Media, 2012.

[7] Corey Montella. The kalman filter and related algorithms.

[8] Lennart Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *IEEE Transactions on Automatic Control*, 24(1):36–50, 1979.

[9] Randall L Eubank. *A Kalman filter primer*. Chapman and Hall/CRC, 2005.

[10] Harold Wayne Sorenson. *Kalman filtering: theory and application*. IEEE, 1985.

[11] SG Mohinder and PA Augus. Kalman filtering: Theory and practice with matlab, 2001.

[12] Roger Labbe. Kalman and bayesian filters in python, 2014.

[13] Andrew C Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[14] Curt Wells. *The Kalman filter in finance*, volume 32. Springer Science & Business Media, 2013.

[15] Eli Brookner. *Tracking and Kalman filtering made easy*. Wiley New York, 1998.

[16] Simon Haykin. *Kalman filtering and neural networks*, volume 47. John Wiley & Sons, 2004.

[17] Paul Zarchan and Howard Musoff. *Fundamentals of Kalman filtering: a practical approach*. American Institute of Aeronautics and Astronautics, Inc., 2013.

[18] Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–194. International Society for Optics and Photonics, 1997.

[19] Danny Abramovitch. Great failures in control engineering: The kalman-bucy water filter. *IEEE CONTROL SYSTEMS MAGAZINE*, 1066(033X/06), 2006.

[20] Mohinder S Grewal and Angus P Andrews. Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]. *IEEE Control Systems*, 30(3):69–78, 2010.

[21] Stergios I Roumeliotis and Anastasios I Mourikis. Extended kalman filter for 3d localization and vision-aided inertial navigation, January 25 2018. US Patent App. 15/706,149.

[22] Janne Hakkarainen, Zenith Purisha, Antti Solonen, and Samuli Siltanen. Undersampled dynamic x-ray tomography with dimension reduction kalman filter. *arXiv preprint arXiv:1805.00871*, 2018.

[23] Jonathan R Stroud, Matthias Katzfuss, and Christopher K Wikle. A bayesian adaptive ensemble kalman filter for sequential state and parameter estimation. *Monthly Weather Review*, 146(1):373–386, 2018.

[24] SM Codrescu, MV Codrescu, and M Fedrizzi. An ensemble kalman filter for the thermosphere-ionosphere. *Space Weather*, 16(1):57–68, 2018.

[25] Muhammad F Emzir, Matthew J Woolley, and Ian R Petersen. A quantum extended kalman filter. *Journal of Physics A: Mathematical and Theoretical*, 50(22):225301, 2017.

[26] Lee Samuel Finn and Soma Mukherjee. Data conditioning for gravitational wave detectors: A kalman filter for regressing suspension violin modes. *Physical Review D*, 63(6):062004, 2001.

[27] M Fukugita, Craig J Hogan, and Phillip James Edwin Peebles. The cosmic distance scale and the hubble constant. *Nature*, 366(6453):309, 1993.

[28] Gregory M Harry, LIGO Scientific Collaboration, et al. Advanced ligo: the next generation of gravitational wave detectors. *Classical and Quantum Gravity*, 27(8):084006, 2010.

[29] Mankei Tsang. Testing quantum mechanics: a statistical approach. *Quantum Measurements and Quantum Metrology*, 1:84–109, 2013.

[30] AI Ovseevich. Kalman filter and quantization. In *Doklady Mathematics*, volume 75, pages 436–439. Springer, 2007.

[31] Jose A Hernando. The kalman filter technique applied to track fitting in glast. *World Wide Web: http://scipp. ucsc. edu/groups/glast/pub/offline/kalman. ps*, 1998.

[32] Francois Caron, Emmanuel Duflos, Denis Pomorski, and Philippe Vanheeghe. Gps/imu data fusion using multisensor kalman filtering: introduction of contextual aspects. *Information fusion*, 7(2):221–230, 2006.

[33] T Wu and P O'Grady. An extended kalman filter for collaborative supply chains. *International journal of production research*, 42(12):2457–2475, 2004.

[34] Thomas Oakes, Lie Tang, Robert G Landers, and SN Balakrishnan. Kalman filtering for manufacturing processes. In *Kalman Filter Recent Advances and Applications*. InTech, 2009.

[35] Enrique Del Castillo and Douglas C Montgomery. A kalman filtering process control scheme with an application in semiconductor short run manufacturing. *Quality and Reliability Engineering International*, 11(2):101–105, 1995.

[36] Phil Howlett, Peter Pudney, Xuan Vu, et al. *Estimating train parameters with an unscented kalman filter*. PhD thesis, Queensland University of Technology, 2004.

[37] Joao P Hespanha. *Linear systems theory*. Princeton university press, 2018.

[38] A Banaszuk, M Kociecki, and FL Lewis. Kalman decomposition for implicit linear systems. *IEEE transactions on automatic control*, 37(10):1509–1514, 1992.

[39] Robert Hermann and Arthur Krener. Nonlinear controllability and observability. *IEEE Transactions on automatic control*, 22(5):728–740, 1977.

[40] Frederick Copleston. *History of philosophy*, volume 8. Bloomsbury Publishing, 1999.

[41] AJ Berkhout and PR Zaanen. A comparison between wiener filtering, kalman filtering, and deterministic least squares estimation. *Geophysical prospecting*, 24(1):141–197, 1976.

[42] Thomas A Wenzel, KJ Burnham, MV Blundell, and RA Williams. Dual extended kalman filter for vehicle state and parameter estimation. *Vehicle System Dynamics*, 44(2):153–171, 2006.

[43] Fredrik Lindsten, Thomas B Schön, and Lennart Svensson. A non-degenerate rao-blackwellised particle filter for estimating static parameters in dynamical models. In *IFAC Proceedings. 16th IFAC Symposium on System Identification*, volume 16, pages 1149–1154, 2012.

[44] Giancarlo Marafioti. Enhanced model predictive control: Dual control approach and state estimation issues. 2010.

[45] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter. *Kalman filtering and neural networks*, pages 221–280, 2001.

[46] Herbert Goldstein, Charles Poole, and John Safko. Classical mechanics, 2002.

[47] Richard E Berg and Todd S Marshall. Wilberforce pendulum oscillations and normal modes. *American Journal of Physics*, 59(1):32–38, 1991.

[48] Muhammad Umar Hassan and Muhammad Sabieh Anwar. 'phystrack': a matlab based environment for video tracking of kinematics in the physics laboratory. *European Journal of Physics*, 38(4):045007, 2017.