UNIVERSITY OF CALIFORNIA, SAN DIEGO

Bayesian Estimation for Nonlinear Dynamic Systems: New Developments and Applications

A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy in Engineering Sciences (Mechanical Engineering)

by

Huazhen Fang

Committee in charge:

Professor Raymond A. de Callafon, Chair
Professor Robert Bitmead
Professor Jorge Cortés
Professor Peter J.S. Franks
Professor Jules S. Jaffe

2014
The dissertation of Huazhen Fang is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2014
DEDICATION

To my family.
# TABLE OF CONTENTS

Signature Page .......................................................... iii
Dedication ................................................................. iv
Table of Contents .......................................................... v
List of Figures ............................................................. viii
List of Tables .............................................................. x
List of Abbreviations ...................................................... xi
Acknowledgements ........................................................ xii
Vita ............................................................................ xv
Abstract of the Dissertation ........................................... xvii

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.1 Motivating Problems</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.2 Bayesian Estimation</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1.3 Summary of Main Contributions</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>1.4 Organization</td>
<td>9</td>
</tr>
</tbody>
</table>

## I Nonlinear State Estimation

<table>
<thead>
<tr>
<th>Chapter 2</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bayesian State Filtering</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>2.1 Introduction</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>2.2 Bayesian Paradigm for State Filtering</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>2.3 Gaussian State Filtering</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>2.4 Radial Basis Gaussian State Filtering</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>2.4.1 RBF-based Filtering Integral Evaluation</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>2.4.2 RBF Approximation via Neural Network Learning</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>2.5 Ensemble Kalman Filtering</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>2.6 Iterated Ensemble Kalman Filtering</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>2.6.1 Iterated Ensemble Kalman Filter - Type 1</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>2.6.2 Iterated Ensemble Kalman Filtering - Type 2</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>2.7 Conclusion</td>
<td>34</td>
</tr>
<tr>
<td>Chapter</td>
<td>Title</td>
<td>Pages</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>3</td>
<td>State-of-charge Estimation for Batteries</td>
<td>37-66</td>
</tr>
<tr>
<td>3.1</td>
<td>A Reduced-Complexity Model</td>
<td>40</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Working Mechanism of Li$^+$ Batteries</td>
<td>40</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Single Particle Model</td>
<td>41</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Reduced-Complexity State-Space Model</td>
<td>44</td>
</tr>
<tr>
<td>3.2</td>
<td>Multiple Model Construction and Analysis</td>
<td>48</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Multiple Model Construction</td>
<td>48</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Observability Analysis</td>
<td>49</td>
</tr>
<tr>
<td>3.3</td>
<td>Basics of Multi-model Estimation</td>
<td>52</td>
</tr>
<tr>
<td>3.4</td>
<td>Multi-model Adaptive SoC Estimation</td>
<td>56</td>
</tr>
<tr>
<td>3.5</td>
<td>Application Examples</td>
<td>60</td>
</tr>
<tr>
<td>3.6</td>
<td>Conclusion</td>
<td>66</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>II</th>
<th>Simultaneous Input and State Estimation</th>
<th>68</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Iterated Nonlinear Simultaneous Input and State Filtering</td>
<td>69-100</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>69</td>
</tr>
<tr>
<td>4.2</td>
<td>SISF for Nonlinear Systems with Direct Feed-through</td>
<td>71-93</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Bayesian Paradigm</td>
<td>72</td>
</tr>
<tr>
<td>4.2.2</td>
<td>SISF Algorithm Development</td>
<td>74</td>
</tr>
<tr>
<td>4.3</td>
<td>SISF for Nonlinear Systems without Direct Feedthrough</td>
<td>80-100</td>
</tr>
<tr>
<td>4.4</td>
<td>SISF for Linear Systems</td>
<td>85-100</td>
</tr>
<tr>
<td>4.4.1</td>
<td>SISF Algorithms for Linear Systems</td>
<td>85</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Connections with Other Algorithms</td>
<td>87</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Asymptotic Stability of Linear SISF Algorithms</td>
<td>93</td>
</tr>
<tr>
<td>4.5</td>
<td>Conclusion</td>
<td>100</td>
</tr>
</tbody>
</table>

| 5  | Iterated Nonlinear Simultaneous Input and State Smoothing | 102-117 |
| 5.1| Introduction | 102 |
| 5.2| SISS for Nonlinear Systems with Direct Feed-through | 103-117 |
| 5.2.1| Bayesian Paradigm | 103 |
| 5.2.2| SISS Algorithm Development | 105 |
| 5.3| SISS for Nonlinear Systems without Direct Feedthrough | 108-117 |
| 5.4| SISS for Linear Systems | 114-117 |
| 5.5| Conclusions | 117 |

| 6  | Ensemble Simultaneous Input and State Estimation | 119-120 |
| 6.1| Introduction | 119 |
| 6.2| Ensemble SISF | 120 |
| 6.2.1| Ensemble SISF for Nonlinear Systems with Direct Feedthrough | 120 |
6.2.2 Ensemble SISF for Nonlinear Systems without Direct Feedthrough .................................. 126
6.3 Ensemble SISS ....................................................................................................................... 127
6.3.1 Ensemble SISS for Nonlinear Systems with Direct Feedthrough .................................. 128
6.3.2 Ensemble SISS for Nonlinear Systems without Direct Feedthrough ............................. 130
6.4 Conclusion .......................................................................................................................... 133

Chapter 7 Oceanic Flow Field Reconstruction ........................................................................ 134
7.1 Introduction ......................................................................................................................... 134
7.2 Drifter Dynamics .................................................................................................................. 136
7.3 Simulation Examples .......................................................................................................... 139
7.4 Sea Test Example ............................................................................................................... 146
7.5 Conclusion .......................................................................................................................... 146

Chapter 8 Conclusion and Future Work ................................................................................. 148

Appendix A Appendix ............................................................................................................. 151

Bibliography ............................................................................................................................. 156
LIST OF FIGURES

Figure 1.1: Major Bayesian state filters. ........................................ 6

Figure 2.1: The architecture of a RBFNN ................................. 22

Figure 3.1: A schematic description of adaptive SoC estimation. ...... 38
Figure 3.2: (a) Schematic characterization of a Li$^+$ battery; (b) the single-particle model. ......................................................... 41
Figure 3.3: The relationship between key variables in the SPM. ........ 44
Figure 3.4: The structure of a multi-model estimator. .................... 52
Figure 3.5: (a) The input current-output voltage profile; (b) SoC estimates versus truth over time; (c) fusion weights for $M_1$ and $M_2$ versus time; (d) comparison with the EKF. ........................................ 62
Figure 3.6: The input-output (current-voltage) profile. ................. 63
Figure 3.7: (a) SoC estimates versus time; (b) SoC estimates during the initial 450s; (c) fusion weights for $M_1$ and $M_2$ versus time; (d) fusion weights during the initial 450s; (e) comparison between the true and the one-step-forward predicted voltage. ........ 64

Figure 7.1: (a) The scenario for flow-field reconstruction based on submersible drifters; (b) the traveling profile (submerging/surfacing) of a drifter. ................................................................. 135
Figure 7.2: (a) depth-width cross section of the density profile; (b) depth-width cross section of the along-front velocity field; (c) drifters traveling in cubic unidirectional flow domain using the drogues (solid circles: drifters; dashed lines: trajectory of a drifter). ... 137
Figure 7.3: Depth profile of the drifter. ....................................... 138
Figure 7.4: True values (blue solid lines) and estimated values (red dashed lines): (a) $u$ vs. $\hat{u}$; (b) $x_1$ vs. $\hat{x}_1$; (c) $x_2$ vs. $\hat{x}_2$. ...................................................... 139
Figure 7.5: Comparison between the true flow field and the estimated one: (a) true; (b) estimated. .................................................. 140
Figure 7.6: (a) The three-dimensional flow field; (b) top view of the flow field. 141
Figure 7.7: Smoothing results for the drifters released at (-400,-600)m: (a) $x_1$ — displacement along $x$-direction; (b) $x_2$ — velocity along $x$-direction; (c) $u$ — flow velocity; (d) trajectory of the drifter, where the circle denotes the location where the drifter is released. 142
Figure 7.8: (a) True surficial flow velocity along $x$-direction; (b) smoothed surficial flow velocity along $x$-direction; (c) True surficial flow velocity along $y$-direction; (d) smoothed surficial flow velocity along $y$-direction. ........................................... 143
Figure 7.9: (a) $xy$-trajectories of the drifters in the sea test; (b) $xyz$-trajectories of the drifters in the sea test. .......................... 144
Figure 7.10: (a) Estimated flow velocity along $x$-direction; (b) estimated flow velocity along $y$-direction; (c) estimated flow velocity along $z$-direction.
**LIST OF TABLES**

Table 2.1: The *GSF* algorithm: Gaussian state filter. ........................................ 18
Table 2.2: The *RB–GSF* algorithm: Radial Basis Gaussian State Filter. .......... 24
Table 2.3: The *IEnKF-1* algorithm: Type-1 Iterated ensemble Kalman filter for nonlinear systems. .................................................. 31
Table 2.4: The *IEnKF-2* algorithm: Type-2 iterated ensemble Kalman filter for nonlinear systems. .................................................. 35

Table 3.1: Nomenclature for the battery model. ................................................. 42
Table 3.2: The *MM-AdaSoC* algorithm: Adaptive SoC estimation using multiple models. ................................................................. 58

Table 4.1: The *ISISF-wDF* algorithm: Iterated SISF for nonlinear systems with direct feedthrough. .................................................. 80
Table 4.2: The *ISISF-w/oDF* algorithm: Iterated SISF for nonlinear systems without direct feedthrough. .................................................. 84
Table 4.3: The *LSISF-wDF* algorithm: SISF for linear systems with direct feedthrough. ................................................................. 86
Table 4.4: The *LSISF-w/oDF* algorithm: SISF for linear systems without direct feedthrough. ................................................................. 87

Table 5.1: The *ISISS-wDF* algorithm: Iterated SISS for nonlinear systems with direct feedthrough. .................................................. 109
Table 5.2: The *ISISS-w/oDF* algorithm: Iterated SISS for nonlinear systems with direct feedthrough. .................................................. 114
Table 5.3: The *LSISS-wDF* algorithm: SISS for linear systems with direct feedthrough. ................................................................. 116
Table 5.4: The *LSISS-w/oDF* algorithm: SISS for linear systems with direct feedthrough. ................................................................. 117

Table 6.1: The *GSISF-wDF* algorithm: Gaussian SISF for nonlinear systems with direct feedthrough. .................................................. 123
Table 6.2: The *EnSISF-wDF* algorithm: Ensemble SISF for nonlinear systems with direct feedthrough. .................................................. 125
Table 6.3: The *EnSISF-w/oDF* algorithm: Ensemble SISF for nonlinear systems without direct feedthrough. .................................................. 128
Table 6.4: The *EnSISS-wDF* algorithm: ensemble SISS for nonlinear systems with direct feedthrough. .................................................. 131
Table 6.5: The *EnSISS-w/oDF* algorithm: ensemble SISS for nonlinear systems with direct feedthrough. .................................................. 132
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADF</td>
<td>Assumed Density Filtering</td>
</tr>
<tr>
<td>BMS</td>
<td>Battery Management System</td>
</tr>
<tr>
<td>CDF</td>
<td>Central Difference Filter</td>
</tr>
<tr>
<td>CKF</td>
<td>Cubature Kalman Filter</td>
</tr>
<tr>
<td>EKF</td>
<td>Extended Kalman Filter</td>
</tr>
<tr>
<td>EnKF</td>
<td>Ensemble Kalman Filter</td>
</tr>
<tr>
<td>GHF</td>
<td>Gauss-Hermite Filter</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>IEKF</td>
<td>Iterated Extended Kalman Filter</td>
</tr>
<tr>
<td>IEnKF</td>
<td>Iterated Ensemble Kalman Filter</td>
</tr>
<tr>
<td>KF</td>
<td>Kalman filter</td>
</tr>
<tr>
<td>MAP</td>
<td>Maximum a Posteriori</td>
</tr>
<tr>
<td>MMSE</td>
<td>Minimum Mean Square Error</td>
</tr>
<tr>
<td>MHE</td>
<td>Moving Horizon Estimation</td>
</tr>
<tr>
<td>MVUE</td>
<td>Minimum-variance Unbiased Estimation</td>
</tr>
<tr>
<td>OCV</td>
<td>Open-circuit Voltage</td>
</tr>
<tr>
<td>PF</td>
<td>Particle Filter</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial Basis Function</td>
</tr>
<tr>
<td>RBFNN</td>
<td>RBF Neural Network</td>
</tr>
<tr>
<td>SISE</td>
<td>Simultaneous Input and State Estimation</td>
</tr>
<tr>
<td>SISF</td>
<td>Simultaneous Input and State Filtering</td>
</tr>
<tr>
<td>SISS</td>
<td>Simultaneous Input and State Smoothing</td>
</tr>
<tr>
<td>SoC</td>
<td>State-of-charge</td>
</tr>
<tr>
<td>SPM</td>
<td>Single Particle Model</td>
</tr>
<tr>
<td>UKF</td>
<td>Unscented Kalman Filter</td>
</tr>
</tbody>
</table>
ACKNOWLEDGEMENTS

I have received support and encouragement from many people over the past five years. Raymond A. de Callafon has been a mentor and colleague. He has given me guidance and opportunities, which were invaluable and made my Ph.D. journey rewarding. I would like to express my sincerest appreciation to my dissertation committee of Robert Bitmead, Jorge Cortés, Peter J.S. Franks and Jules S. Jaffe for their constructive support.

I took some terrific classes offered by Thomas Bewley, Robert Bitmead, Jorge Cortés, Raymond A. de Callafon, William M. McEneaney, Miroslav Krstic, Maurício de Oliveira. My grateful thanks go to them for sharing their time and knowledge and walking me through different interesting control subjects. They are outstanding researchers and teachers.

I owe my thanks to Charlotte Lauve, Linda McKamey and Marina Robenko for their help and patience in handling my study and funding affairs.

I was grateful to Yebin Wang and Zafer Sahinoglu of Mitsubishi Electric Research Laboratories and Rakesh Patil of NEC Laboratories America for offering me internship opportunities. It was a wonderful pleasure to work with them, and the experiences were absolutely memorable.

I would like to thank my labmates, Youghee Han, Chad Holcomb, Daniel Miller and Xin Zhao, whose dedication, talent and friendliness impressed and inspired me. I fortunately met many great friends at UCSD, Daniele Cavaglieri, Ruinan Chang, Ying Chen, Minyi Ji, Yao Liu, Danyong Lu, Beibei Ren, Yu Ru, Shuxia Tang, Qiyun Zhao, Yuchun Zhou and Minghui Zhu. I would like to thank them, wishing them the best of luck in the future.

I have deepest thanks for my parents and elder sister for their love and support through all the years. I dedicate the dissertation to them.

In this dissertation, the results in Chapter 2 are based the papers that were published or will be submitted:


Chapter 3 is based on the following published papers:


Chapter 4 includes the content from the following published papers:


Chapter 5 is based on the following papers that were published or submitted:


Chapter 6 is based on the following papers that will be submitted:


The results presented in Chapter 7 are part of the following papers that were published or submitted:


VITA

2006 Bachelor of Engineering in Computer Science & Technology, Northwestern Polytechnic University, China

2009 Master of Science in Mechanical Engineering, University of Saskatchewan, Canada

2010 Gordon Engineering Leadership Scholar

2012 Research Intern, Mitsubishi Electric Research Laboratories

2013 Research Intern, NEC Laboratories America

2014 Doctor of Philosophy in Engineering Sciences (Mechanical Engineering), University of California, San Diego

PUBLICATIONS

Journal Publications


Conference Proceedings


ABSTRACT OF THE DISSERTATION

Bayesian Estimation for Nonlinear Dynamic Systems: New Developments and Applications

by

Huazhen Fang

Doctor of Philosophy in Engineering Sciences (Mechanical Engineering)

University of California, San Diego, 2014

Professor Raymond A. de Callafon, Chair

Estimation of unknown quantities in a nonlinear dynamic system has been a challenge of great interest and importance, which is encountered in different research fields. Founded on a perspective of updating probabilistic belief on unknown quantities with observations, Bayesian analysis has provided a useful methodology and framework for construction of various estimation techniques. This dissertation presents a study of some new developments and applications of Bayesian estimation theory. Both filtering and smoothing will be considered — the former concerns estimation of the present situation using measurements up until the present time, and the latter is about estimation of the past using all the measurements.

In the dissertation, we investigate both state estimation and simultaneous
input and state estimation. For the former, Bayesian filtering in a Gaussian context is discussed. We propose to use the radial basis function approximation as a desirable option to realize the Gaussian state filtering. We then improve the standard ensemble Kalman filter by introducing iterative optimization. Simultaneous input and state estimation has emerged as a new challenge. We extend the Bayesian methodology to deal with this problem and consider both filtering and smoothing cases. The Bayesian paradigms are built as a statistical foundation to fulfill this task, are built. On such a basis, we then develop a series of estimation methods using iterative optimization and Monte Carlo-based ensemble approaches. We further examine the link between our methods and the existing ones and analyze their properties especially in the linear case.

The dissertation also studies application of estimation techniques to some real-world issues. We investigate real-time state-of-charge estimation for batteries, proposing an adaptive method based on multi-model state estimation. It allows for accurate estimation in the presence of uncertain or unknown variables and can promote the battery monitoring performance and operational safety potentially. The other application presented is oceanic flow field reconstruction. Flows exist everywhere in the ocean, playing a crucial role in many aspects of marine environment and biology. As part of a collaborative effort with Scripps Institution of Oceanography to build an original ocean observing system based on a group of drifters, we apply the simultaneous input and state estimation methods proposed to analyze the data collected by drifters to estimate the flow velocity and monitor the drifter’s motion status. This work can be conducive to future endeavors in oceanographic research.
Chapter 1

Introduction

1.1 Motivating Problems

The model of a nonlinear dynamic system relates quantities such as inputs, states, outputs and parameters, some of which may be unknown. Estimation of them is often necessary in order to understand, monitor and regulate the system. It is essentially concerned with how to estimate the unknown quantities or processes characterizing a dynamic system using the measurements collected from the system. As a core subject of control systems theory, it has remained a highly active research focus during the past few decades. Considerable attention has also been gained from a wider community of researchers, thanks to its significant implications for signal processing, navigation and guidance and econometrics just to name a few. In this dissertation, we will study two theoretical problems regarding estimation for nonlinear dynamic systems and investigate two applications.

The first problem is state estimation for nonlinear systems, which is classical and commonly encountered. The state of a system comprises a set of variables that fully describes the status or condition of the system. It evolves with time as a result of the system dynamics, which can often be driven by an external input. The process of states over time hence represents the system’s behavior. Since it is unrealistic to measure the complete state in most circumstances, state estimation is needed to determine the state of a system from the input and output measurements. Its importance comes from the crucial role it plays in tracking
and control of systems. First, one can monitor how a system behaves with state information and take corresponding actions when any adjustment is necessary. This is particularly important to ensure the detection and handling of internal faults and anomalies at the earliest phase. Second, high-quality state estimation is the basis for the design and implementation of many control strategies, since the ultimate goal of control is to regulate the system’s state-based behavior. The Kalman filter (KF), named after Rudolf E. Kálmán for his work [1], is arguably the most celebrated state estimation approach for linear systems. However, despite numerous research efforts and some promising solutions, nonlinear state estimation is still far from being fully settled, leaving us much space for further research.

The other one is *simultaneous input and state estimation for nonlinear systems*. For this problem, not only is the state unknown, but the input to the system is also unavailable or inaccessible. It will become more complicated when the input is completely unknown with neither *a priori* knowledge nor model-based description available. What will be of interest is to estimate both the input and state jointly using the output measurements. A comparison with state estimation shows the difference of “estimating more from less”, which can also be the main source of difficulty for the design of estimation approaches. Recent years have seen an increasing interest in this problem because of its broad range of applications. While most research efforts have been concentrated on linear systems, the nonlinear case has been studied insufficiently, with very few methods reported in the literature.

Appreciation of estimation theory is largely due to its practical significance. In this dissertation, we will investigate estimation-based solutions to two real-world problems. First, nonlinear state estimation is applied to *battery management*. Batteries are everywhere nowadays, with widespread use in consumer electronics and power tools. Looking into the future, they will form a crucial part of the energy networks and a cornerstone for electrified transportation and renewable energy penetration. An effective management system is often needed to monitor the battery status and regulate the charging and discharging processes for operational safety and performance enhancement. Estimation of the state-of-charge (SoC) is a basic function of this system. The SoC is a measure of how much energy is
present in a battery. In practice, batteries are expected to operate within a certain SoC range for safety and lifetime considerations and especially avoid over-charging and over-discharging. This requires reliable SoC information, which, however, can only be indirectly determined from measurements of applied current and terminal voltage. It has been observed that construction of high-performance SoC estimation methods can greatly benefit from exploitation of nonlinear state estimation techniques.

The second application to consider is reconstruction of three-dimensional oceanic flow fields as part of an effort to monitor a small-scale ocean domain with high accuracy. Its value is grounded on the pivotal role of flows in the marine physical, chemical and biological environment and the climate change. An ocean observing system is intended to be built and deployed, which consists of a group of submersible drifters. The drifters are capable of vertical migration. After being released, each one follows the submerging-surfacing cycle when traveling along the flow. While underwater, it measures its acceleration via on-board accelerometers; when at surface, it determines its position by the satellite-based Global Positioning System (GPS) and transmits all the collected data to the remote central server. A primary goal of the system is to reconstruct the flow field. Less important, but still nontrivial, is to track the underwater status of each drifter. This motivates a joint estimation of the flow velocities, which act as external input to the drifter, and the drifter’s trajectory and velocities, which can be viewed as its state, from its position (intermittently available) and acceleration (incessant) measurements.

As is seen, this boils down to the aforementioned problem of simultaneous input and state estimation.

1.2 Bayesian Estimation

We will take Bayesian estimation approaches to address the problems listed above. As a useful estimation tool enjoying wide popularity nowadays, Bayesian analysis traces back to a 1763 essay [2], published two years after the death of its author Thomas Bayes. The seminal work was meant to tackle the following
question: “Given the number of times in which an unknown event has happened and failed: Required the chance that the probability of its happening in a single trial lies somewhere between any two degrees of probability that can be named”. Bayes developed a solution, an early form of the Bayes’ theorem, which examined the case of only continuous probability, single parameter and a uniform prior. Despite its preciousness, this work remained obscure for many scientists and even mathematicians of that era. The change came when the French mathematician Pierre-Simon de Laplace rediscovered the result and presented the theorem in the complete and modern form. A historical account and comparison of Bayes’ and Laplace’s work can be found in [3]. From today’s perspective, the Bayes’ theorem is a probability-based answer to a philosophical question: How should one update an existing belief when given new evidence? [4]. Quantifying the degree of belief by probability, the theorem modifies the original belief via producing the probability conditioned on evidence from initial probability. It has been applied in the past century from one field to another whenever the belief update question arose. A remarkable exploration in this respect is Bayesian estimation, which is about how to determine unknown variables of a system using the Bayesian theory.

The thinking of Bayesian estimation is built upon three essential elements. First and foremost, all the unknown quantities or uncertainties in a system, e.g., state, are expressed in terms of probability. In other words, any unknown variable will be regarded as a random variable. Second, the output measurement of the system are samples drawn from a certain probability distribution dependent on the concerned variables. Finally, a model-based description is available for the system, which characterizes the propagation and observation of the system’s unknown variables over time adequately in the situation of interest. Originating from the philosophical abstraction that anything unknown, in one’s mind, is subject to variations due to chance, the randomness-based representation enjoys universal applicability even when the unknown or uncertain quantities are not necessarily random in physical sense. In addition, it can easily translate into a convenient ‘engineering’ way for estimation of the unknown.

At the core of Bayesian estimation for dynamic systems is to determine the
probabilities of unknown quantities and their evolution over time given the measurements. This requires the development of a Bayesian paradigm, which dictates the probabilistic update of the unknown quantities in light of new measurements at each time instant. To be more specific, it usually considers the \textit{a priori} probability and the \textit{a posteriori} one. The former captures our belief over the unknown quantities in the presence of only the prior evidence, and the latter updates the belief using the Bayesian theory when new evidence becomes available. The paradigm is time-dependent in general because the dynamic system evolve with time. Hence, it is often performed sequentially or one time instant after another.

The capacity of the Bayesian analysis to offer a conceptual framework for estimation has been well recognized. A typical estimation method within the framework is the celebrated Kalman filter (KF), which “revolutionized the field of estimation ... (and) opened up many new theoretical and practical possibilities” [5]. Despite its least squares-based original development in the 1960 paper [1], it was revealed in [6] to be interpretable from a Bayesian perspective, only four years after its invention. Further envisioned in [6] was that “the Bayesian approach offers a unified and intuitive viewpoint particularly adaptable to handling modern-day control problems”. This investigation and vision ushered a new statistical treatment of nonlinear estimation problems, laying foundation for prosperity of research on this subject. The existing Bayesian state estimation methods can be roughly divided into three categories.

- **Gaussian-based**: Gaussian filters are derived from the Bayesian paradigm under Gaussian distribution assumptions. It simply reduces to the KF in the linear case. For the nonlinear case, the key is to evaluate integrals involving Gaussian functions arising from calculating the state mean and covariance. A few quadrature and cubature rules can be used here, producing the Gauss-Hermite filter (GHF) [7], cubature Kalman filter (CKF) [8], central difference filter (CDF). The unscented Kalman filter (UKF) [9, 10, 11], which was initially developed via unscented transformation of a random variable, is also a type of GHF. We will obtain the ensemble Kalman filter (EnKF) [12] when Monte Carlo-based integral evaluation is adopted. Combining Gaussian
filtering with first-order Taylor approximation of the state-space functions will give rise to the extended Kalman filter (EKF).

- **Optimization-based:** The Bayesian paradigm can be translated into optimization problems under some probabilistic assumptions like Gaussianity. There are two directions towards optimal Bayesian estimation. The first one is local optimization, which maximizes the \(a \text{ posteriori}\) probability at each time instant. Due to the nonlinearity, it is fulfilled typically by numerical searching such as the Newton’s or Gauss-Newton methods, thus giving rise to estimators with local iterations. The iterated extended KF (IEKF) [13, 14] is a primary work in this respect. The EKF is indeed the IEKF with a single iteration applied at each time. The other direction goes towards a global estimation, which aims to maximize the \(a \text{ posteriori}\) probability of a trajectory of state values. This, in a probabilistic manner, leads to moving horizon estimation (MHE) [15, 16], which essentially is a constrained nonlinear optimization problem.

- **Monte Carlo-based:** Monte Carlo approximation implements numerical computation on random samples to obtain a sample-based description of the unknown probability distribution. The EnKF can be viewed as a Monte Carlo extension of the standard KF in the Gaussian context. Particle filtering
(PF) is another important class of Monte Carlo filtering methods [17, 18]. It manages a set of samples or particles, each with an assigned weight, to represent the a posteriori state probability, and unlike the EnKF, directly runs on the Bayesian paradigm to sequentially update and propagate the samples. The PF does not require specific probabilistic assumptions about the state-space model and the state and noise variables.

Fig. 1.1 shows the major Bayesian state filters mentioned above and their categories. Their importance has been highly appreciated thanks to numerous applications in a broad range of areas. Accompanying such success, a stronger need is observed for extensions and new developments of the Bayesian estimation theory to address more emerging problems.

1.3 Summary of Main Contributions

In the dissertation, we aim to push forward the Bayesian estimation theory on the basis of existing work. The following is a summary of our main contributions.

• Development of new Bayesian state filters: We focus on the Gaussian-based Bayesian filters. We propose a new state filtering algorithm that uses radial basis functions (RBFs) to approximate the nonlinear process and measurement functions of a system, considering their superior approximation performance. Optimal determination of the approximators is achieved by the radial basis function neural network learning. Using the RBF based function approximation, the challenging problem of integral evaluation in Gaussian filtering can be well solved, guaranteeing the filtering performance. Furthermore, we propose to improve the performance of EnKF in terms of accuracy by refining the state estimates through iterative optimization. This will produce two new EnKF-based filters with boosted accuracy and a globally recursive, locally iterative structure.
• **Battery SoC estimation:** Adaptive SoC estimation for batteries is increasingly appealing, thanks to its ability to accommodate uncertain or time-varying model parameters. We propose to improve the adaptive SoC estimation using multiple models in this study, developing a unique algorithm. Specifically, two submodels in state-space form are generated from a modified Nernst battery model. Both are shown to be locally observable with admissible inputs. The a state filter is then applied to each submodel in parallel, estimating simultaneously the SoC variable and unknown parameters. The SoC estimates obtained from the two separately implemented filters are fused to yield the final overall SoC estimates, which tend to have higher accuracy than those obtained from a single model. Its effectiveness is demonstrated using simulation and experiments. The notion of multi-model estimation can be extended promisingly to the development of many other advanced battery management and control strategies. Besides, we study the battery estimation and convert a well-known electrochemical model into one with reduced complexity and equivalent to an equivalent circuit model, shedding light on the connections among models derived from electrochemical principles and equivalent circuits.

• **Simultaneous input and state estimation:** We investigate both simultaneous input and state filtering (SISF) and simultaneous input and state smoothing (SISS) for nonlinear systems. Filtering is forward estimation of the unknown present-time quantities using all the measurements up until the present, and smoothing estimates the past quantities backward using all the obtained measurements. The former is the option for real-time estimation, and the latter, implemented in non-real time, usually possesses better accuracy. Even though SISF has been studied in the context of linear systems, research on the nonlinear case is still rare. SISS, in spite of its potential, has been given limited attention thus far and even not been investigated by other researchers to the best or our knowledge. We develop approaches for both SISF and SISS via taking a Bayesian perspective. The Bayesian paradigms are established first as the foundations on which the filtering and smoothing
proceed. We then design two schemes. The first one gives rise to nonlinear maximum a posteriori (MAP) optimization problems for SISF and SISS, respectively, which we solve using the classical Gauss-Newton method. Design of the other scheme is based on the notion of ensemble estimation. We use ensembles to represent the unknown input and state probability, propagate them over time, and extract the information such as their means and covariances at each time instant. This scheme allows for SISF and SISS for large-scale nonlinear systems. A discussion as to what the proposed algorithms will be like for linear systems is presented. We find that the linear algorithms generalizes a number of linear SISF methods available in the literature and study their asymptotic stability properties.

- **Oceanic flow field reconstruction:** We apply the proposed SISF and SISS algorithms to reconstruction of oceanic flow fields. The study of flow fields is a compelling problem in oceanography, and our study will contribute to better solving this open problem. Flows play a key role in phenomena such as the transportation of nutrients, the motion of biological species in their early life, and the diffusion of contaminants and algal blooms. To reconstruct a flow field, we consider deploying a group of buoyancy-controlled drogues capable of arbitrary vertical migration behaviors in the ocean domain. For a drifter, the lateral motion is driven by the flow, which can be regarded as the unknown input, and its motion status such as position and velocity is the state. The input and state here will be estimated from the collected measurement data via our algorithms, with satisfactory results obtained.

### 1.4 Organization

The remainder of the dissertation is structured into two parts. Part I concerns theory and application of nonlinear state estimation, including Chapters 2-3. Part II investigates the theory and application of simultaneous input and state estimation, including Chapters 4-7.

In Chapter 2, we consider the Bayesian state filtering for nonlinear systems.
We offer a review of the sequential Bayesian filtering paradigm and present its different realizations that lead to different filters.

In Chapter 3, we develop adaptive SoC estimation algorithms for battery management through applying nonlinear state estimation techniques. A complete study is presented. It starts from battery modeling using electrochemical principles, followed by model simplification and observability analysis. We then construct SoC estimation algorithms using the IEKF.

In Chapter 4, we consider nonlinear SISF. From a statistical perspective, we build Bayesian filtering paradigms for this problem. Then we derive the SISF algorithms via performing nonlinear iterative optimization. We also study their linear versions, reporting their link with some existing methods and proving their asymptotic stability conditions.

In Chapter 5, we extend the results in the previous chapter to the smoothing. Following similar lines, we build the Bayesian paradigms first as the basis of SISS and develop optimization-based algorithms.

In Chapter 6, we develop ensemble approaches for SISF and SISS based on the paradigms developed in Chapters 4-5. With a higher efficiency, the proposed approaches are particularly suitable for handling large-scale systems.

In Chapter 7, we offer an application study of our SISF and SISS algorithms to reconstruction of oceanic flow fields.

In Chapter 8, we summarize and draw concluding remarks for the dissertation. Suggestions for some future work are presented as well in this chapter.
Part I

Nonlinear State Estimation
Chapter 2

Bayesian State Filtering

2.1 Introduction

This chapter studies the problem of state filtering for nonlinear systems from a Bayesian perspective. Bayesian analysis has provided a useful and proven way to study nonlinear state estimation. It concentrates on sequentially updating the conditional probability density functions (pdf’s) of unknown state variables given the output measurements [19]. Research on this subject has drawn a large amount of effort during the past decades, with many approaches either evolving from or interpretable by Bayesian estimation.

A topic of intensive investigation in this respect is Gaussian state filtering, which builds on Bayesian estimation and assumed density filtering (ADF). The ADF assumes a particular form of density, which are often mathematically tractable to deal with, for the pdf’s involved in the Bayesian estimator, and then computes the state estimates [20]. If the pdf’s are assumed Gaussian, the ADF will lead to the Gaussian filters. A crucial issue in Gaussian filtering is to evaluate a number of integrals. The integrand of each integral is a product of a nonlinear function (stemming from the nonlinear system equations) and a Gaussian density function. To address the issue, two main approaches have been proposed.

One of them is based on Monte Carlo implementation. A representative method is the EnKF [12, 21]. It uses an ensemble of point samples to represent the state probability and applies the KF computation to update the ensemble,
one sample after another, to track the systems’s state. Then samples of the state ensemble will be aggregated to yield the final state estimate. The aforementioned integrals are implicitly approximated during the process. Thanks to its computational competitiveness, the EnKF is very desirable when it comes to filtering for large-scale systems.

The other approach is direct numerical integration via quadrature or cubature techniques. Making use of the Gauss-Hermite quadrature rule, the Gauss-Hermite filter (GHF) is capable of giving almost accurate evaluation of the integrals arising in Gaussian filtering through a weighted summation of the nonlinear function evaluated at some fixed points [22, 7]. The well-known unscented Kalman filter (UKF), which was developed in [10, 11] from the viewpoint of nonlinear transformation of random variables, is a special case of GHF. In [8, 23], the cubature KF (CKF) is proposed, which adopts a spherical-radial cubature rule for numerical computation of these integrals.

This chapter will develop two new Gaussian state filters. The first one is realized by a radial basis function (RBF) based approach. Widely used in pattern classification problems [24], RBFs also find important applications in the development of intelligent control systems, e.g, [25, 26, 27, 28]. We propose that integral evaluation in Gaussian filtering can be reduced to function approximation, the construction of which can be achieved using a set of RBFs. Here we will use the Gaussian RBF (GRBF), because Gaussian-type functions, which frequently appears in Gaussian filtering, are easily manipulatable to derive integration in closed form. With an equivalent structure to a RBF Neural Network (RBFNN), the RBF based function approximator can be established through training the RBFNN. The obtained filter, which we refer to as the radial basis Gaussian state filter or RB-GSF, has high estimation performance and computational efficiency.

We then craft an iterated EnKF (IEnKF) method as an improved version of the EnKF. Despite its success in solving many problems, the EnKF may still be subject to failures [29], implying that there is both need and space for improvement. It is noted that the standard EnKF updates the state ensemble in just one step. This can result in a coarse ensemble that fails to characterize the state probability.
In addition, such an update will be too crude to make the best use of the newly arrived measurement data for correction of estimation. Thus we are motivated to modify the EnKF update by carrying out iterative optimization to obtain the best state ensemble.

The remainder of this chapter is organized as follows. Section 2.2 presents the Bayesian paradigm, which is the foundation of Bayesian state filtering. Section 2.3 offers an overview of the Gaussian states filtering based on Bayesian estimation theory. We then develop the RB-GSF algorithm in Section 2.4, showing the novel application of the RBFs to Gaussian filtering. The EnKF, another form of Gaussian filtering, is reviewed in Section 2.5, and modified into the IEnKF using local optimization in Section 2.6. Finally, some concluding remarks are given in Section 2.7.

### 2.2 Bayesian Paradigm for State Filtering

Let us consider the following nonlinear discrete-time system:

\[
\begin{align*}
    x_{k+1} &= f(x_k) + w_k, \\
    y_k &= h(x_k) + v_k,
\end{align*}
\]

where \(x_k \in \mathbb{R}^{n_x}\) is the unknown system state and \(y_k \in \mathbb{R}^{n_y}\) is the output. The process noise \(w_k\) and the measurement noise \(v_k\) are mutually independent, zero-mean white Gaussian sequences with covariances \(Q_k\) and \(R_k\), respectively. The nonlinear mappings \(f: \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_x}\) and \(h: \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_y}\) represent the process dynamics and the measurement model, respectively. The system in (2.1) is assumed input-free for simplicity, but the obtained results about SE can be easily extended to systems with external input.

Define the measurement set \(Y_k := \{y_1, y_2, \cdots, y_k\}\). The problem of interest is to estimate the system state \(x_k\) from \(Y_k\) at each time instant \(k\). The system is stochastic, and \(x, y, w\) and \(v\) can be viewed as random variables. At time \(k - 1\), a statistical description of \(x_k\) from \(Y_{k-1}\) is given as the \textit{a priori} pdf \(p(x_k|Y_{k-1})\). When the new measurement \(y_k\) containing information about \(x_k\) ar-
rives, it will update \( p(x_k|Y_{k-1}) \) to the \textit{a posteriori} pdf \( p(x_k|Y_k) \). It is understood that \( p(x_k|Y_{k-1}) \) and \( p(x_k|Y_k) \) characterize our belief on \( x_k \) before and after \( y_k \) is available, respectively, and such a belief underlies the estimation. The Bayesian paradigm, as stated in the following theorem, shows how they are propagated at each time instant.

**Theorem 2.1.** [13, 30] For the system in (2.1), the Bayesian paradigm for state filtering is given as follows:

\[
p(x_k|Y_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|Y_{k-1})dx_{k-1},
\]

(2.2)

\[
p(x_k|Y_k) = \frac{p(y_k|x_k)p(x_k|Y_{k-1})}{p(y_k|Y_{k-1})}.
\]

(2.3)

**Proof.** By the Chapman-Kolmogorov equation [31], it can be seen that

\[
p(x_k|Y_{k-1}) = \int p(x_k, x_{k-1}|Y_{k-1})dx_{k-1},
\]

which, according to the Bayes’s theorem, yields

\[
p(x_k|Y_{k-1}) = \int p(x_k|x_{k-1}, Y_{k-1})p(x_{k-1}|Y_{k-1})dx_{k-1}.
\]

It will reduce to (2.2) because \( p(x_k|x_{k-1}, Y_{k-1}) = p(x_k|x_{k-1}) \) as a result of the Markovian propagation of the state. Let us now examine \( p(x_k|Y_k) \). By the Bayes’ theorem, we have

\[
p(x_k|Y_k) = \frac{p(x_k, Y_k)}{p(Y_k)} = \frac{p(x_k, y_k, Y_{k-1})}{p(y_k, Y_{k-1})}
= \frac{p(y_k|x_k, Y_{k-1})p(x_k, Y_{k-1})}{p(y_k, Y_{k-1})}
= \frac{p(y_k|x_k, Y_{k-1})p(x_k|Y_{k-1})}{p(y_k|Y_{k-1})}.
\]

From the fact that \( y_k \) only depends on \( x_k \), \( p(y_k|x_k, Y_{k-1}) = p(y_k|x_k) \) follows. Thus (2.3) is obtained from the above equation.

Theorem 2.1 presents the fundamental statistical principle of Bayesian state
filtering for dynamic systems, describing the sequential update of the a priori and a posteriori pdf’s. It consists of two steps, (2.2) and (2.3), which are referred to as prediction and update, respectively, and implemented alternately in real time. Within the framework offered by Theorem 2.1, a variety of filtering methods can be developed. In the following, we will show several filters which can be derived using Theorem 2.1.

2.3 Gaussian State Filtering

It is understood that, for the Bayesian filtering paradigm in (2.2)-(2.3), it usually is intractable to compute the pdf’s in an analytical or closed-form way due to the nonlinearities. Certain simplification is hence needed to develop implementable filtering methods. In this section, we will show the Gaussian state filtering method. As the name suggests, it is constructed in a Gaussian setting, where the pdf’s are approximated by Gaussian distributions. The following two assumptions are made:

**Assumption 2.1.** \( p(x_k|Y_{k-1}) \) is a Gaussian distribution with mean \( \hat{x}_{k|k} \) and \( P_{x|k}^{x} \).

**Assumption 2.2.** \( p(x_k|x_{k-1}) \) is a Gaussian distribution with mean \( f(x_{k-1}) \) and \( Q_{k-1} \).

Let the mean of \( x_k \) given \( Y_{k-1} \) be the prediction of \( x_k \), denoted as \( \hat{x}_{k|k-1} \). Then,

\[
\hat{x}_{k|k-1} = \int x_k p(x_k|Y_{k-1})dx_k \\
= \int \left[ \int x_k p(x_k|x_{k-1})dx_k \right] p(x_{k-1}|Y_{k-1})dx_{k-1} \\
= \int f(x_{k-1}) \cdot G(x_{k-1}; \hat{x}_{k-1|k-1}, P_{x|k-1}^{x})dx_{k-1}.
\] (2.4)

The associated prediction error covariance is

\[
P_{x|k-1}^{x} = \int (x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})^\top p(x_k|Y_{k-1})dx_k
\]
\[
\begin{align*}
&= \int \mathbf{x}_k \mathbf{x}_k^\top p(\mathbf{x}_k | \mathbf{y}_{k-1}) d\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1} \hat{\mathbf{x}}_{k|k-1}^\top \\
&= \int \mathbf{f}(\mathbf{x}_{k-1}) \mathbf{f}(\mathbf{x}_{k-1})^\top \cdot \mathcal{G}(\mathbf{x}_{k-1}; \mathbf{x}_{k-1|k-1}, \mathbf{P}^x_{k-1|k-1}) d\mathbf{x}_{k-1} \\
&\quad - \hat{\mathbf{x}}_{k|k-1} \hat{\mathbf{x}}_{k|k-1}^\top + \mathbf{Q}_{k-1}.
\end{align*}
\] (2.5)

When \(y_k\) becomes available, we consider the joint conditional pdf of \(x_k\) and \(y_k\) given \(Y_{k-1}\), which is assumed Gaussian:

**Assumption 2.3.** 
\[p \left( \begin{bmatrix} x_k^\top & y_k^\top \end{bmatrix} \right| Y_{k-1} \right) \] is a Gaussian distribution:

\[
p \left( \begin{bmatrix} x_k \\ y_k \end{bmatrix} \right| Y_{k-1} \right) = \mathcal{G} \left( \begin{bmatrix} x_k \\ y_k \end{bmatrix}; \hat{\mathbf{x}}_{k|k-1}, \begin{bmatrix} \mathbf{P}^x_{k|k-1} \\ \mathbf{P}^y_{k|k-1} \end{bmatrix} \right). \] (2.6)

In above, \(\hat{y}_{k|k-1}\) is the prediction of \(y_k\) given \(Y_{k-1}\), given by

\[
\hat{y}_{k|k-1} = \int y_k p(y_k | Y_{k-1}) dy_k. \] (2.7)

It is noted that

\[
p(y_k | Y_{k-1}) = \int p(x_k, y_k | Y_{k-1}) d\mathbf{x}_k = \int p(y_k | x_k) p(x_k | Y_{k-1}) d\mathbf{x}_k.
\]

Inserting the above equation into (2.7) yields

\[
\hat{y}_{k|k-1} = \int \left[ \int y_k p(y_k | \mathbf{x}_k) dy_k \right] p(\mathbf{x}_k | Y_{k-1}) d\mathbf{x}_k \\
= \int \mathbf{h}(\mathbf{x}_k) p(\mathbf{x}_k | Y_{k-1}) d\mathbf{x}_k \\
= \int \mathbf{h}(\mathbf{x}_k) \cdot \mathcal{G}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^x_{k|k-1}) d\mathbf{x}_k.
\] (2.8)

The associated covariance is

\[
\mathbf{P}^y_{k|k-1} = \int (y_k - \hat{y}_{k|k-1}) (y_k - \hat{y}_{k|k-1})^\top p(y_k | Y_{k-1}) dy_k \\
= \int \mathbf{h}(\mathbf{x}_k) \mathbf{h}^\top(\mathbf{x}_k) \cdot \mathcal{G}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^x_{k|k-1}) d\mathbf{x}_k - \hat{y}_{k|k-1} \hat{y}_{k|k-1}^\top + \mathbf{R}_k.
\] (2.9)
and the cross-covariance is

\[
P_{xy}^{k|k-1} = \int \int (x_k - \hat{x}_{k|k-1})(y_k - \hat{y}_{k|k-1})^\top p(x_k, y_k|Y_{k-1}) \, dx_k \, dy_k
\]

\[
= \int x_k h(x_k)^\top \cdot G(x_k; \hat{x}_{k|k-1}, P_{x}^{k|k-1}) \, dx_k - \hat{x}_{k|k-1} y_{k|k-1}^\top.
\] (2.10)

It follows from (2.6) and Lemma A.3 that

\[
p(x_k|Y_k) = G(x_k; \hat{x}_{k|k}, P_{x}^{k|k}),
\]

where

\[
\hat{x}_{k|k} = \hat{x}_{k|k-1} + P_{xy}^{k|k-1} \left( P_{y}^{k|k-1} \right)^{-1} (y_k - \hat{y}_{k|k-1}),
\] (2.11)

\[
P_{x}^{k|k} = P_{x}^{k|k-1} - P_{xy}^{k|k-1} \left( P_{y}^{k|k-1} \right)^{-1} \left( P_{xy}^{k|k-1} \right)^\top.
\] (2.12)

The Gaussian filter in its basic form is summarized in Table 2.1.

Table 2.1: The GSF algorithm: Gaussian state filter.

| **initialize:** | \( k = 0 \), \( \hat{x}_{0|0} = E(x_0) \), \( P_{x0}^{x} = p_0 I \), where \( p_0 > 0 \), typically a large positive value |
| **repeat** | \( k \leftarrow k + 1 \) |
| **Prediction:** | project the state ahead via (2.4) compute the prediction error covariance via (2.5) |
| **Update:** | project the output via (2.8) with the associated covariance via (2.9) compute the cross-covariance via (2.10) update the state estimate via (2.11) compute the estimation error covariance via (2.12) |
| **until** | no more measurements arrive |

It is noteworthy, however, that the Gaussian filter only delineates a conceptual framework for this type of filtering methods. To make it truly applicable in practice, it is necessary to develop methods for evaluation of the integrals in (2.4)-(2.5) and (2.8)-(2.10).
2.4 Radial Basis Gaussian State Filtering

2.4.1 RBF-based Filtering Integral Evaluation

As is observed, the integrals in (2.4)-(2.5) and (2.8)-(2.10) take one of the following forms:

\[ \Omega_1 = \int g(x) \cdot \mathcal{G}(x; \mu, \Sigma) dx, \quad (2.13) \]
\[ \Omega_2 = \int xg^\top(x) \cdot \mathcal{G}(x; \mu, \Sigma) dx, \quad (2.14) \]
\[ \Omega_3 = \int g(x)g^\top(x) \cdot \mathcal{G}(x; \mu, \Sigma) dx. \quad (2.15) \]

where \( x \in \mathbb{R}^n \), and \( g \) is assumed without loss of generality to be a mapping over \( \mathbb{R}^n \rightarrow \mathbb{R}^m \), where \( n \) and \( m \) are an arbitrary positive integers. In this section, we introduce the notion and realization of the RBF approximation of \( g(x) \), and then continue to show how to construct the RB-GF, based on the proposed function approximation.

For \( g_i(x) \) that is the \( i \)-th element of \( g(x) \), we consider a nonlinearly parameterized approximator

\[ \hat{g}_i(x) = \sum_{j=1}^{N} w_{ij} s_j(x), \quad (2.16) \]

where \( s_j(x) \) for \( j = 1, 2, \cdots, N \) is a set of \( N \) RBFs and \( w_{ij} \)'s are the weighting factors. A wide variety of RBFs such as multi-quadratics, inverse multi-quadratics and Gaussian functions, have been studied in the literature. We propose to use the Gaussian RBFs (GRBFs), which will facilitate addressing the problem of Gaussian filtering. Then \( s_j(x) \) is given by

\[ s_j(x) = \exp \left[ -\frac{(x - c_j)^\top(x - c_j)}{2\sigma_j^2} \right] \]
\[ = \alpha_j \cdot \mathcal{G}(x; c_j, \sigma_j^2 I), \]

where \( \alpha_j = (2\pi\sigma_j^2)^{\frac{n}{2}} \), \( c_j \) and \( \sigma_j \) are the center and width of the RBF, respectively. For simplicity, we assume that \( c_j \)'s and \( \sigma_j \)'s are fixed and known. The assumption
does not limit the extension of the ensuing derivation to the case when both of them are unknown and need to be determined.

It follows from Lemma A.4 that multiplying two Gaussian functions gives another unnormalized Gaussian function. Hence, we consider

$$
\beta_j \cdot \mathcal{G}(x; \bar{\mu}_j, \bar{\Sigma}_j) = \mathcal{G}(x; c_j, \sigma^2_j I) \cdot \mathcal{G}(x; \mu, \Sigma),
$$

$$
\gamma_{jl} \cdot \mathcal{G}(x; \mu_{jl}, \Sigma_{jl}) = \mathcal{G}(x; c_j, \sigma^2_j I) \cdot \mathcal{G}(x; c_l, \sigma^2_l I) \cdot \mathcal{G}(x; \mu, \Sigma),
$$

where

$$
\bar{\Sigma}_j = (\sigma^{-2}_j I + \Sigma^{-1})^{-1},
$$

$$
\bar{\mu}_j = \Sigma_j (\sigma^{-2}_j c_j + \Sigma^{-1}_j \mu),
$$

$$
\beta_j = (2\pi)^{-\frac{n}{2}} \sigma^{-n}_j |\Sigma_j|^{-\frac{1}{2}} \bar{\Sigma}_j^{\frac{1}{2}} \cdot \exp \left[ -\frac{1}{2} \left( \sigma^{-2}_j c_j + \mu^{\top} \Sigma^{-1}_j \mu - \bar{\mu}_j \Sigma_j^{-1} \bar{\mu}_j \right) \right],
$$

$$
\Sigma_{jl} = \left( \sigma^{-2}_l I + \Sigma^{-1}_j \right)^{-1},
$$

$$
\mu_{jl} = \Sigma_{jl}(\sigma^{-2}_l c_l + \Sigma^{-1}_j \mu),
$$

$$
\gamma_{jl} = \beta_j (2\pi)^{-\frac{n}{2}} \sigma^{-n}_l |\Sigma_j|^{-\frac{1}{2}} |\Sigma_{jl}|^{\frac{1}{2}} \cdot \exp \left[ -\frac{1}{2} \left( \sigma^{-2}_l c_l + \bar{\mu}_j \Sigma_j^{-1} \bar{\mu}_j - \mu_{jl}^{\top} \Sigma_{jl}^{-1} \mu_{jl} \right) \right].
$$

Note that if $\mu$ and $\Sigma$ are variables, $\bar{\Sigma}_j, \bar{\mu}_j, \beta_j, \Sigma_{jl}, \mu_{jl}$ and $\gamma_{jl}$ are all functions of $\mu$ and $\Sigma$. We have the following integration formulae before proceeding further:

$$
\int s_j(x) \cdot \mathcal{G}(x; \mu, \Sigma) dx = \alpha_j \cdot \beta_j(\mu, \Sigma),
$$

$$
\int x s_j(x) \cdot \mathcal{G}(x; \mu, \Sigma) dx = \alpha_j \cdot \beta_j(\mu, \Sigma) \cdot \bar{\mu}_j(\mu, \Sigma),
$$

$$
\int s_j(x)s_l(x) \cdot \mathcal{G}(x; \mu, \Sigma) dx = \alpha_j \alpha_l \cdot \gamma_{jl}(\mu, \Sigma).
$$

Define the following matrices and vectors:

$$
W = \begin{bmatrix} : & : & : \\
\vdots & \cdots & w_{ij} & \cdots \\
: & : & : \end{bmatrix},
$$
\[
\mathbf{s}(x) = \begin{bmatrix} \cdots & s_j(x) & \cdots \end{bmatrix}^\top,
\]

\[
\beta(\mu, \Sigma) = \begin{bmatrix} \alpha_j \beta_j(\mu, \Sigma) \\
\vdots \\
\alpha_j \beta_j(\mu, \Sigma) \cdot \bar{\mu}_j(\mu, \Sigma) \\
\vdots \\
\alpha_j \beta_j(\mu, \Sigma) \cdot \bar{\mu}_j(\mu, \Sigma) \end{bmatrix},
\]

\[
\Psi(\mu, \Sigma) = \begin{bmatrix} \cdots & \alpha_j \beta_j(\mu, \Sigma) \cdot \bar{\mu}_j(\mu, \Sigma) & \cdots \end{bmatrix},
\]

\[
\Gamma(\mu, \Sigma) = \begin{bmatrix} \cdots & \alpha_j \alpha_l \cdot \gamma_{jl}(\mu, \Sigma) & \cdots \\
\vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots 
\end{bmatrix}.
\]

Here, \( W \in \mathbb{R}^{m \times N}, s \in \mathbb{R}^N, \beta \in \mathbb{R}^N, \Psi \in \mathbb{R}^{N \times m} \) and \( \Gamma \in \mathbb{R}^{N \times N} \). We then have \( \hat{g}(x) = W \cdot s(x) \) and

\[
\Omega_1 \approx \int \hat{g}(x) \cdot \mathcal{N}(x|\mu, \Sigma) dx
= W \cdot \beta(\mu, \Sigma),
\tag{2.17}
\]

\[
\Omega_2 \approx \int x\hat{g}^\top(x) \cdot \mathcal{N}(x|\mu, \Sigma) dx
= \Psi(\mu, \Sigma) \cdot W^\top,
\tag{2.18}
\]

\[
\Omega_3 \approx \int \hat{g}(x)\hat{g}^\top(x) \cdot \mathcal{N}(x|\mu, \Sigma) dx
= W \cdot \Gamma(\mu, \Sigma) \cdot W^\top.
\tag{2.19}
\]

We see that (2.17)-(2.19) construct a computational foundation, on which the Gaussian filtering integrals in (2.4)-(2.5) and (2.8)-(2.10) can be evaluated easily.

### 2.4.2 RBF Approximation via Neural Network Learning

Prior to integral evaluation in (2.17)-(2.19), the weight matrix \( W \) must be determined optimally in the sense that approximation error between \( g(x) \) and \( \hat{g}(x) \) is minimized. A formulation of this problem is: Given a data set containing \( M \) elements, \( \{(d_j, z_j)|z_j = g(d_j), d_j \in \mathbb{R}^n, z_j \in \mathbb{R}^m, j = 1, 2, \ldots, M\} \), find the
optimal $W$ to minimize the convex cost function $J(W)$ defined as

$$J(W) = \frac{1}{2} \sum_{j=1}^{M} \|z_j - \hat{g}(d_j)\|^2$$

$$= \frac{1}{2} \sum_{j=1}^{M} \|z_j - W \cdot s(d_j)\|^2$$

$$= \frac{1}{2} \sum_{j=1}^{M} \sum_{i=1}^{m} \|z_{ji} - w_i \cdot s(d_j)\|^2,$$

(2.20)

where $z_{ji}$ is the $i$-th element of $z_j$ and $w_i$ is the $i$-th row vector of $W$. It is noteworthy that each $w_i$ can be determined separately by minimizing

$$J(w_i) = \sum_{j=1}^{M} \|z_{ji} - w_i \cdot s(d_j)\|^2.$$

The above weight determination problem is equivalent to training a RBF Neural Network (RBFNN). A RBFNN usually performs curve fitting in a high dimensional space, or more specifically, to find a hypersurface that provides a best fit for the high-dimensional training data [24].

For $g_i(x)$, the schematic diagram of a RBFNN is shown in Fig. 2.1. It has three layers. The first one is the input layer, which has $n$ nodes corresponding to each element of the input vector $d$. The second layer is a hidden layer with $N$
units, to each of which all nodes in the first layer are connected to. The activation functions of the $j$-th unit is the GRBF $s_j(x)$, indicating that this is indeed a GRBFNN. Each $s_j(x)$ in the hidden layer is connected through the weight $w_{ij}$ to the output layer that has only a single unit. This unit computes a weighted sum of the outputs of the hidden units as the output of the network.

It is noted that the RBFNN translates the function approximation under consideration into neural network learning, which applies learning strategies to the training data set $D$ to determine the weights of the output layer. A few different types of learning strategies have been proposed in the literature. A straightforward approach is to use the pseudoinverse method to derive the least squares solution to (3.4). However, it is computationally inefficient, especially whenever new data become available, and also poorly scalable to large data sets. To remedy this situation, most other approaches for RBFNN learning carry out recursive updating. Among them, we highlight the one based on gradient descent [24]. Consider $\hat{g}_i$ in (2.16), which can be rewritten as $\hat{g}_i(x) = w_i \cdot s(x)$. The recursive learning procedure of $w_i$ is expressed as

$$\phi(\ell) = \sum_{j=1}^{M} s^\top(d_j) [z_{ji} - w_i(\ell) \cdot s(d_j)],$$

(2.21)

$$w_i(\ell + 1) = w_i(\ell) - \eta \cdot \phi(\ell),$$

(2.22)

where $\phi = \nabla_{w_i} J(w_i)$ is the gradient, $\eta$ is the learning coefficient and $\ell$ denotes the recursion step.

Approximation properties of the RBFNN is of much significance in practical implementation. The Universal Approximation Theorem states that, if $g(x)$ is continuous, then there is a RBFNN such that the function $\hat{g}(x)$ realized by the RBFNN is close to $g(x)$ in the $L_p$ norm for $p \in [1, \infty]$ [24]. Furthermore, it is pointed out in [26] that $\hat{g}(x)$ can approximate the continuous $g(x)$ to an arbitrary accuracy over a compact set. Thus for the considered Gaussian filtering problem, if the function approximators are well designed via RBFNNs, high-accuracy approximation can be achieved, thus ensuring the filtering performance.

Putting together the formulae in the Gaussian filter, function approximation
and integral evaluation yields the RB-GSF algorithm, as is described in Table 2.2.

**Remark 2.1.** Essentially a Gaussian filter with RBFNNs employed to assist in dealing with integral evaluation, the proposed RB-GSF algorithm differs significantly from existing RBFNN based nonlinear filtering schemes, e.g., [32], in which RBFNNs are used to model unknown or uncertain system dynamics. In addition, instead of doing fixed-point quadrature or cubature approximation like in [7, 8], it is a customized filter incorporating construction of approximators for different nonlinear functions in different systems.

Table 2.2: The RB-GSF algorithm: Radial Basis Gaussian State Filter.

<table>
<thead>
<tr>
<th>Function approximation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Construction of the RBF based approximators for $f$ and $h$ using RBFNN via (2.21)-(2.22)</td>
</tr>
</tbody>
</table>

**initialize:** $k = 0$, $\hat{x}_{0|0} = E(x_0)$, $P_{0|0} = p_0 I$, where $p_0 > 0$, typically a large positive value

**repeat**

$k \leftarrow k + 1$

**Prediction:**
State prediction via (2.4) and (2.17)
Computation of prediction error covariance via (2.5) and (2.19)

**Update:**
Measurement prediction via (2.8) and (2.17) with the associated covariance via (2.9) and (2.19)
Computation of the cross-covariance via (2.10) and (2.18)
State estimation via (2.11)
Computation of the estimation error covariance via (2.12)

**until** no more measurements arrive

2.5 **Ensemble Kalman Filtering**

In this section, we introduce the EnKF below, which is a Monte Carlo realization of the GSF algorithm. The Monte Carlo method concerns simulation-based approximation of probability distributions using direct draws from certain
distributions. It has gained widespread use in numerical integration [33]. A brief overview is as follows. Suppose the integral of $g(x)$ over the domain $D$ is to be evaluated:

$$T = \int_D g(x)dx.$$  

Let $g(x)$ be factored into a product of $r(x)$ and $p(x)$, i.e.,

$$g(x) = r(x)p(x),$$

where $p(x)$ is a pdf defined over $D$ with $\int_D p(x)dx = 1$. Then,

$$T = E[r(x)] = \int_D r(x)p(x)dx.$$  

Draw $N_s$ random sample points from the distribution $p(x)$, $\{x_i, i = 1, 2, \cdots, N_s\}$, which can be used to empirically describe $p(x)$, i.e.,

$$p(x) \approx \frac{1}{N_s} \sum_{i=1}^{N_s} \delta(x - x_i).$$

The value of $T$ is then approximated by

$$T = \frac{1}{N_s} \sum_{i=1}^{N_s} r(x_i). \quad (2.23)$$

Now let us apply the Monte Carlo method to deriving the EnKF. When there is a set of $N_s$ samples, $\{\hat{x}_{i-1 \mid k-1}, i = 1, 2, \cdots, N_s\}$ drawn from $p(x_{k-1} \mid Y_{k-1})$, it can be propagated ahead to generate the ensemble that represents $p(x_k \mid Y_{k-1})$, $\{\hat{x}_{k \mid k-1}\}$, with

$$\hat{x}_{k \mid k-1} = f(\hat{x}_{k-1 \mid k-1}) + w_{k-1}^i. \quad (2.24)$$

The ensemble $\{w_{k-1}^i\}$ in above is built i.i.d from the Gaussian distribution with zero mean and covariance $Q_{k-1}$ to account for the process noise. Computing the
sample mean and sample covariance of \( \{ \hat{x}^i_{k|k-1} \} \) by (2.4)-(2.5), we have

\[
\hat{x}_{k|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}^i_{k|k-1},
\]

(2.25)

\[
P_{x|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}^i_{k|k-1} \hat{x}^i_{k|k-1}^\top - \hat{x}_{k|k-1} \hat{x}_{k|k-1}^\top,
\]

(2.26)

which form the prediction formulae. By (2.8), the ensemble for \( p(y_k|Y_{k-1}) \), \( \{ \hat{y}^i_{k|k-1} \} \), can be constructed via

\[
\hat{y}^i_{k|k-1} = h(\hat{x}^i_{k|k-1}) + v^i_k,
\]

(2.27)

where \( \{ v^i_k \} \) is generated as per the Gaussian distribution with zero mean and covariance \( R_k \). The sample mean is

\[
\hat{y}_{k|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{y}^i_{k|k-1}.
\]

(2.28)

According to (2.9), the covariance of \( \{ \hat{y}^i_{k|k-1} \} \) is

\[
P_{y|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{y}^i_{k|k-1} \hat{y}^i_{k|k-1}^\top - \hat{y}_{k|k-1} \hat{y}_{k|k-1}^\top,
\]

(2.29)

and by (2.10), its cross-covariance with \( \{ \hat{x}^i_{k|k-1} \} \) is

\[
P_{xy|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}^i_{k|k-1} \hat{y}^i_{k|k-1}^\top - \hat{x}_{k|k-1} \hat{y}_{k|k-1}^\top.
\]

(2.30)

The new measurement \( y_k \) can then be used to update \( \{ \hat{x}^i_{k|k-1} \} \) like (2.11) to compute the ensemble for \( p(x_k|Y_k) \), \( \{ \hat{x}^i_{k|k} \} \), where

\[
\hat{x}^i_{k|k} = \hat{x}^i_{k|k-1} + P_{xy|k-1} \left( P_{y|k-1}^{-1} (y_k - \hat{y}^i_{k|k-1}) \right).
\]

(2.31)
The updated state estimate and its error covariance is

$$\hat{x}_{k|k} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}_{k|k}^i, \quad (2.32)$$

$$P_{k|k} = \frac{1}{N_s} \sum_{i=1}^{N_s} (\hat{x}_{k|k}^i)^\top - \hat{x}_{k|k} \hat{x}_{k|k}^\top. \quad (2.33)$$

The above process will be repeated through time, with the ensemble \( \{ \hat{x}_{k|k}^i \} \) tracking the system state and the state estimate \( \hat{x}_{k|k} \) computed at each time instant.

**Remark 2.2.** As a Monte Carlo-based extension of the KF, the EnKF operates directly on the ensembles. It represents the pdf’s using ensembles of samples, propagates the ensembles and makes estimation by computing the sample mean and sample covariance of the state ensemble at each time instant. Making the computation derivative-free and the propagation of the estimation error covariance matrix necessary, it can be implemented efficiently and thus is often the method of choice for high-dimensional nonlinear systems. It has been reported that convergence of the EnKF can be quite fast even with a reasonable small ensemble size [21, 34]. Its converge to the KF in the limit for large ensemble size and Gaussian state probability distributions is proven in [34].

### 2.6 Iterated Ensemble Kalman Filtering

In this section, we will develop two IEnKFs to improve the capacity of the EnKF. The first one is developed to attain better estimation performance when significant measurement nonlinearities exist. The key is to introduce local iterations to continually refine the state estimate. This work is partially inspired by the IEKF [13] and found equivalent to [29]. The second IEnKF is proposed to deal with both process and measurement nonlinearities. Its development is based on the iterated linear filter-smoother (ILFS) in [13], which conducts filtering and one-step-back smoothing iteratively.
2.6.1 Iterated Ensemble Kalman Filter - Type 1

Consider state filtering for the system in (2.1). The IEKF is constructed on two steps, prediction and update. The prediction at time instant $k$ is given by

\[
\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}),
\]

(2.34)

\[
P_{x_{k|k-1}} = F_{k-1} P_{x_{k-1|k-1}} F_{k-1}^T + Q_{k-1},
\]

(2.35)

where

\[
F_{k-1} = \frac{\partial f}{\partial x}(\hat{x}_{k-1|k-1}).
\]

The update step uses the new measurement $y_k$ to correct $\hat{x}_{k|k-1}$. Involving local iterations, it is realized by

\[
K_{k}^{(\ell)} = P_{x_{k|k-1}}^{x} H_{k}^{(\ell)^T} \left[ H_{k}^{(\ell)} P_{x_{k|k-1}}^{x} H_{k}^{(\ell)^T} + R \right]^{-1},
\]

(2.36)

\[
\hat{y}_{k}^{(\ell)} = h\left(\hat{x}_{k|k}^{(\ell)}\right) + H_{k}^{(\ell)} \left(\hat{x}_{k|k-1} - \hat{x}_{k|k}^{(\ell)}\right),
\]

(2.37)

\[
\hat{x}_{k|k}^{(\ell+1)} = \hat{x}_{k|k-1} + K_{k}^{(\ell)} \left(y_k - \hat{y}_{k}^{(\ell)}\right),
\]

(2.38)

where $\ell$ denotes the iteration index and

\[
H_{k}^{(\ell)} = \frac{\partial h}{\partial x}(\hat{x}_{k|k}^{(\ell)}).
\]

It is natural to assign $\hat{x}_{k|k}^{\ell_{\text{max}}}$ to $\hat{x}_{k|k}$, with the error covariance

\[
P_{x_{k|k}}^{x} = \left[I - K_{k}^{(\ell_{\text{max}})} H_{k}^{(\ell_{\text{max}})}\right] P_{x_{k|k-1}}^{x},
\]

(2.39)

where $\ell_{\text{max}}$ is the maximum iteration number. Note that the IEKF will reduce to the EKF if $\ell_{\text{max}} = 1$. Even though the EKF is one of the most popular techniques [35], its performance is often unsatisfactory in terms of convergence speed and robustness to nonlinearities. The IEKF offers a good means to improve EKF. The iteration shown in (2.36)-(2.38) linearizes the measurement function repeatedly around the new state operating point and then updates the state estimate to
improve accuracy. Such a procedure can effectively inhibit the negative effect of measurement function nonlinearity on the filtering performance [13, 36].

The IEKF can be well interpreted from a Bayesian perspective using (2.2)-(2.3). We consider the maximum a posteriori (MAP) estimation for state filtering. That being said, the best estimate of $x_k$ in the prediction step is the one that maximizes $p(x_k | Y_{k-1})$, and in the update step, the one that maximizes $p(x_k | Y_k)$.

Assume that $p(x_{k-1} | Y_{k-1})$ is a Gaussian distribution with mean $\hat{x}_{k-1}$ and covariance $P_{k-1}^{x}$, and that $p(x_k | x_{k-1})$ is Gaussian with mean $f(x_{k-1})$ and covariance $Q_{k-1}$. Then by (2.2), the predicted estimate of $x_k$ and the first-order approximation of the prediction error covariance will be given by (2.34)-(2.35), respectively.

For the update, we similarly let $p(x_k | Y_k)$ be Gaussian with mean $\hat{x}_k$ and covariance $P_{k|k-1}^{x}$ and $p(y_k | x_k)$ be Gaussian with mean $h(x_k)$ and covariance $R_k$. It then follows from (2.3) that maximization of $p(x_k | Y_k)$ with respect to $x_k$ is equivalent to minimizing the MAP cost function

$$L(x_k) = \left[ y_k - h(x_k) \right]^{\top} R_k^{-1} \left[ y_k - h(x_k) \right] + (x_k - \hat{x}_k) \left( P_{k|k-1}^{x} \right)^{-1} (x_k - \hat{x}_k).$$

Applying either the Newton’s method or the Gauss-Newton method to the above problem will lead to an iterative process of searching the minimum point shown in the formulae (2.36)-(2.38) [13, 14].

Next we intend to develop the IEnKF method, with the aim of further improving the estimation performance and accelerating the convergence of the EnKF via introducing local iterations with inspirations from the IEKF.

The IEnKF also operates on ensembles. The prediction step is the same as (2.24)-(2.25), with the state ensemble $\{ \hat{x}_{k-1|k-1} \}$ projected ahead to generate $\{ \hat{x}_{k|k-1} \}$ simply using the system model. However, rather than computing a single-step solution, the update step refines the state ensemble $\{ \hat{x}_{k|k}^{i,(l)} \}$ iteratively, where
\( \ell \) is the iteration index. Its sample mean is
\[
\hat{x}_{k|k}^{(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}_{k|k}^{i,(\ell)}.
\] (2.40)

Initially, let \( \{ \hat{x}_{k|k}^{i,(0)} \} = \{ \hat{x}_{k|k-1}^{i} \} \) and \( \hat{x}_{k|k}^{(0)} = \hat{x}_{k|k-1} \).

The initial output ensemble \( \{ \hat{y}_{k}^{i,(0)} \} \) is built by
\[
\hat{y}_{k}^{i,(0)} = h \left( \hat{x}_{k|k}^{i,(0)} \right) + v_{k}^{i,(0)}.
\] (2.41)

For \( \ell \geq 1 \), it is given by
\[
\hat{y}_{k}^{i,(\ell)} = h \left( \hat{x}_{k|k}^{i,(\ell)} \right) + P_{x,y}^{x,(\ell-1)} \left( P_{k}^{x,(\ell-1)} \right)^{-1} \left( \hat{x}_{k|k}^{i,(\ell)} - \hat{x}_{k|k-1} \right) + v_{k}^{i,(\ell)},
\] (2.42)

where \( P_{x,y}^{x,(\ell-1)} \left( P_{k}^{x,(\ell-1)} \right)^{-1} \) approximates the measurement matrix obtained by linearizing the measurement function \( h(x_k) \) around \( \hat{x}_{k|k}^{(\ell-1)} \). For \( \{ y_{k}^{i,(\ell)} \} \), its sample mean and sample covariance are
\[
y_{k}^{(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} y_{k}^{i,(\ell)},
\] (2.43)
\[
P_{k}^{y,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} y_{k}^{i,(\ell)} y_{k}^{i,(\ell)\top} - y_{k|k-1} y_{k|k-1}\top.
\] (2.44)

The cross-covariance between \( \{ \hat{x}_{k|k}^{i,(\ell)} \} \) and \( \{ \hat{y}_{k}^{i,(\ell)} \} \) is
\[
P_{x,y}^{x,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}_{k|k}^{i,(\ell)} y_{k}^{i,(\ell)\top} - \hat{x}_{k|k} y_{k}\top.
\] (2.45)

Then the state ensemble can be updated using the prediction error between \( y_{k} \) and \( \{ \hat{y}_{k}^{i,(\ell)} \} \), i.e.,
\[
\hat{x}_{k|k}^{i,(\ell+1)} = \hat{x}_{k|k}^{i} + P_{x,y}^{x,(\ell)} \left( P_{k}^{y,(\ell)} \right)^{-1} \left( y_{k} - \hat{y}_{k}^{i,(\ell)} \right).
\] (2.46)
The above iteration continually refines the state ensemble. It will stop when the prespecified maximum iteration number $\ell_{\text{max}}$ is reached or the error between two consecutive iterations is less than a threshold value. Then the sample mean and covariance of $\{\hat{x}^{i,\ell_{\text{max}}}_{k|k}\}$ can be assigned to $\hat{x}_{k|k}$ and $P_{k|k}$, respectively. We summarize the equations above to obtain the IEnKF-1 algorithm in Table 2.3.

Table 2.3: The IEnKF-1 algorithm: Type-1 Iterated ensemble Kalman filter for nonlinear systems.

| initialize: $k = 0$, generate the ensemble $\{\hat{x}^i_{0|0}, i = 1, 2, \ldots, N_s\}$ |
| --- |
| repeat |
| $k \leftarrow k + 1$ |
| Prediction: |
| generate the noise ensemble $\{w^i_{k-1}\}$ |
| project the state ensemble $\{\hat{x}^i_{k-1|k-1}\}$ ahead into $\{\hat{x}^i_{k|k-1}\}$ via (2.24) |
| compute the sample mean and covariance via (2.25)-(2.26) |
| Update: |
| initialize: $\ell = 0$, $\{\hat{x}^{i,(0)}_{k|k}\} = \{\hat{x}^i_{k|k-1}\}$ |
| while $\ell < \ell_{\text{max}}$ do |
| compute the sample mean of $\{\hat{x}^{i,(\ell)}_{k|k}\}$ via (2.40) |
| generate the noise ensemble $\{v^i_{\ell}\}$ |
| generate the output ensemble $\{\hat{y}^{i,(\ell)}_{k}\}$ (2.41)-(2.42) |
| compute the mean and covariance of $\{\hat{y}^{i,(\ell)}_{k}\}$ via (2.43)-(2.44) |
| compute the cross-covariance between $\{\hat{x}^{i,(\ell)}_{k|k}\}$ and $\{\hat{y}^{i,(\ell)}_{k}\}$ via (2.45) |
| update the ensemble $\{\hat{x}^{i,(\ell)}_{k|k}\}$ to $\{\hat{x}^{i,(\ell+1)}_{k|k}\}$ via (2.46) |
| $\ell \leftarrow \ell + 1$ |
| end while |
| compute $\hat{x}^{(\ell_{\text{max}})}_{k|k}$ and assign it to $\hat{x}_{k|k}$ |
| compute the estimation error covariance $P_{k|k}$ |
| until no more measurements arrive |

Remark 2.3. The IEnKF-1 algorithm makes local iterations to update the state ensemble and thus have better estimation accuracy. It can remedy the drawback that the EnKF’s single-step update is unable to achieve the MAP solution when the measurement function is nonlinear. It is worth noting that the improvement
comes at the expense of higher computational cost. Thus we would recommend IEnKF if the system involves significant measurement nonlinearity and not too large state space.

\textbf{Remark 2.4.} The IEnKF-1 algorithm can be viewed as merging the IEKF and EnKF to build a new filter blending their merits. We note that it is equivalent to another IEnKF proposed in [29], where the development is based on Gauss-Newton optimization of a cost function.

\subsection*{2.6.2 Iterated Ensemble Kalman Filtering - Type 2}

Compared with the EnKF, the IEnKF-1 algorithm offers a good solution to handle nonlinearity in the measurement function. Then a question naturally arising is how to improve filtering performance in the presence of both process and measurement nonlinearities. To address this problem, we combine filtering, one-step-back smoothing and iterative optimization to further robustify the IEnKF-1 algorithm.

Consider the time instant $k$ and the state ensemble obtained previously at $k-1$, $\{\hat{x}_{k-1|k-1}^i\}$. Let $\bar{x}_{k-1|k-1}^{i,(\ell)}$ represent the one-step-back pdf $p(x_{k-1|Y_k})$. Initially,

$$\bar{x}_{k-1|k-1}^{i,(0)} = \hat{x}_{k-1|k-1}^i. \tag{2.47}$$

The ensemble for $x_k$ projected from $\{\bar{x}_{k-1|k-1}^{i,(\ell)}\}$ is $\{\bar{x}_k^{i,(\ell)}\}$. It is initialized by

$$\bar{x}_k^{i,(0)} = f(\bar{x}_{k-1|k-1}^{i,(0)}) + w_{k-1}^{i,(0)}. \tag{2.48}$$

For $\ell \geq 1$, it is generated by

$$\bar{x}_k^{i,(\ell)} = f(\bar{x}_{k-1|k-1}^{i,(\ell)}) + \bar{P}_{x_{k-1|k-1}}^{x_{k-1|k-1}^{i,(\ell)}} \left( \bar{P}_{x_{k-1|k-1}^{i,(\ell-1)}}^{x_{k-1|k-1}} \right)^{-1} \left( \bar{x}_{k-1|k-1}^{i,(\ell-1)} - \bar{x}_{k|k-1}^{i,(\ell)} \right) + w_{k-1}^{i,(\ell)}, \tag{2.49}$$

where $\bar{P}_{x_{k-1|k-1}^{i,(\ell-1)}}^{x_{k-1|k-1}} \left( \bar{P}_{x_{k-1|k-1}^{i,(\ell-1)}}^{x_{k-1|k-1}} \right)^{-1}$ is an approximation of the state transition matrix obtained by linearizing $f(x_{k-1})$ around $x_{k}^{i,(\ell-1)}$.

For the update, we use the ensemble $\{\bar{x}_{k|k}^{i,(\ell)}\}$ to represent $p(x_k|Y_k)$. For
\( \ell = 0, \)
\[
\mathbf{x}_{k|k}^{i,(0)} = \mathbf{x}_{k}^{i,(0)}. \tag{2.50}
\]

The output ensemble projected using \( \{ \mathbf{x}_{k|k}^{i,(\ell)} \} \) is \( \{ \mathbf{y}_{k}^{i,(\ell)} \} \). For \( \ell = 0 \),
\[
\mathbf{y}_{k}^{i,(0)} = \mathbf{h} \left( \mathbf{x}_{k|k}^{i,(0)} \right) + \mathbf{v}_{k}^{i,(0)}, \tag{2.51}
\]
and for \( \ell \geq 1 \),
\[
\mathbf{y}_{k}^{i,(\ell)} = \mathbf{h} \left( \mathbf{x}_{k|k}^{i,(\ell)} \right) + \mathbf{P}_{k}^{x,y,(\ell-1)\top} \left( \mathbf{P}_{k|k}^{x,(\ell-1)} \right)^{-1} \left( \mathbf{x}_{k}^{i,(\ell)} - \mathbf{x}_{k|k}^{i,(\ell)} \right) + \mathbf{v}_{k}^{i,(\ell)}. \tag{2.52}
\]

Then a series of means of and covariances between the ensembles can be computed, that is,

\[
\mathbf{x}_{k-1}^{(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{x}_{k-1}^{i,(\ell)}, \tag{2.53}
\]
\[
\mathbf{x}_{k}^{(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{x}_{k}^{i,(\ell)}, \tag{2.54}
\]
\[
\mathbf{y}_{k}^{(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{y}_{k}^{i,(\ell)}, \tag{2.55}
\]
\[
\mathbf{x}_{k|k}^{(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{x}_{k|k}^{i,(\ell)}, \tag{2.56}
\]
\[
\mathbf{P}_{k-1}^{x,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{x}_{k-1}^{i,(\ell)}\mathbf{x}_{k-1}^{i,(\ell)\top} - \mathbf{x}_{k-1}^{(\ell)}\mathbf{x}_{k-1}^{(\ell)\top}, \tag{2.57}
\]
\[
\mathbf{P}_{k-1|k}^{x,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{x}_{k-1}^{i,(\ell)}\mathbf{x}_{k}^{i,(\ell)\top} - \mathbf{x}_{k-1}^{(\ell)}\mathbf{x}_{k}^{(\ell)\top}, \tag{2.58}
\]
\[
\mathbf{P}_{k}^{x,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{x}_{k}^{i,(\ell)}\mathbf{x}_{k}^{i,(\ell)\top} - \mathbf{x}_{k}^{(\ell)}\mathbf{x}_{k}^{(\ell)\top}, \tag{2.59}
\]
\[
\mathbf{P}_{k|k}^{x,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbf{x}_{k|k}^{i,(\ell)}\mathbf{x}_{k|k}^{i,(\ell)\top} - \mathbf{x}_{k|k}^{(\ell)}\mathbf{x}_{k|k}^{(\ell)\top}. \tag{2.60}
\]
\[ P_{k}^{xy,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}_{k|i}^{i,(\ell)} \hat{y}_{k}^{i,(\ell)\top} - \hat{x}_{k}^{(\ell)} \hat{y}_{k}^{(\ell)\top}, \]  
(2.61)

\[ P_{k}^{y,(\ell)} = \frac{1}{N_s} \sum_{i=1}^{N_s} y_{k}^{i,(\ell)} \hat{y}_{k}^{i,(\ell)\top} - \hat{y}_{k|k-1} \hat{y}_{k|k-1}^{\top}. \]  
(2.62)

The state ensemble can be updated by

\[ \tilde{x}_{k|k}^{i,(\ell+1)} = \tilde{x}_{k}^{i,(\ell)} + P_{k}^{xy,(\ell)} \left( P_{k}^{y,(\ell)} \right)^{-1} \left( y_{k} - \hat{y}_{k}^{i,(\ell+1)} \right), \]  
(2.63)

and \( \{ \tilde{x}_{k-1}^{i,(\ell)} \} \) can be smoothed via

\[ \tilde{x}_{k-1}^{i,(\ell+1)} = \tilde{x}_{k-1|k-1}^{i} + P_{k-1,k}^{x,(\ell)} \left( P_{k}^{x,(\ell)} \right)^{-1} \left( \tilde{x}_{k|k}^{i,(\ell+1)} - \tilde{x}_{k}^{i,(\ell)} \right). \]  
(2.64)

The above prediction-update-smoothing process can be repeated continually over time. Collecting the equations, we obtain the IEnKF-2 algorithm.

**Remark 2.5.** To improve IEnKF-1, the IEnKF-2 algorithm performs filtering and one-step-back smoothing iteratively for each recursion. It can reduce the effects of nonlinearities on estimation that exist in both process and measurement functions. In addition, the notion of local iterative optimization can be applied to enhance other state filtering algorithms such as the UKF.

### 2.7 Conclusion

In this chapter, we have studied the nonlinear Bayesian estimation with the development of new Gaussian state filters. A distinct advantage of Gaussian filtering is the closed-form description. However, their practical implementation requires evaluation of certain forms of integrals. To deal with this challenge, we have proposed to build approximators composed of a weighed sum of RBFs for nonlinear system functions. Determination of the approximators, i.e., optimal selection of the weights, can be addressed by RBFNN learning. It has been shown that, with the RBF based function approximators, the integrals in Gaussian filtering can be delicately evaluated, giving rise to the RB-GSF algorithm. We then turned atten-
Table 2.4: The IEnKF-2 algorithm: Type-2 iterated ensemble Kalman filter for nonlinear systems.

| initialize: $k = 0$, generate the ensemble $\{\hat{x}^i_{0|0}, i = 1, 2, \ldots, N_s\}$ |
| repeat |
| $k \leftarrow k + 1$ |
| initialize via (2.47) for $\ell = 0$ |
| while $\ell < \ell_{\text{max}}$ do |
| Prediction: |
| generate the noise ensemble $\{w^i_{k-1}\}$ |
| project the state ensemble $\{\bar{x}^i_{k-1}\}$ ahead into $\{\bar{x}^i_{k}\}$ via (2.48)-(2.49) |
| Update: |
| generate the noise ensemble $\{v^i_k\}$ |
| generate the output ensemble $\{\hat{y}^i_k\}$ (2.51)-(2.52) |
| compute the means and covariances via (2.53)-(2.62) |
| update the ensemble $\{\bar{x}^i_{k|\ell}\}$ to $\{\bar{x}^i_{k|\ell+1}\}$ via (2.63) |
| Update: |
| smooth the ensemble $\{\bar{x}^i_{k-1}\}$ to obtain $\{\bar{x}^i_{k-1|\ell+1}\}$ via (2.64) |
| $\ell \leftarrow \ell + 1$ |
| end while |
| compute $\bar{x}^{(\ell_{\text{max}})}_{k|k}$ and assign it to $\hat{x}_{k|k}$ |
| compute the estimation error covariance $P^x_{k|k}$ |
| until no more measurements arrive |

Computationally attractive, it is a valuable tool to handle large state spaces. However, its capacity is limited by its single-step update. Thus we have investigated incorporating local iterations into the EnKF, which implicitly optimizes an MAP cost function as in the IEKF. The obtained methods, IEnKF-1 and IEnKF-2, are expected to have superior performance to the EnKF.

This chapter is based on the following papers that were published or will be submitted:

Chapter 3

State-of-charge Estimation for Batteries

Nowadays rechargeable batteries are everywhere from consumer electronics to power tools. An even stronger demand for them in the next decades is foreseeable due to recent advances in electrified transportation and renewable energy harvesting. Deployment of battery management systems (BMSs) in such applications is indispensable to monitor the battery status and regulate the charging and discharging processes [37]. A fundamental component of any BMS is real-time state-of-charge (SoC) estimation to determine how much energy is present in the battery. In this chapter, we will develop multi-model adaptive approaches based on nonlinear filtering techniques to perform SoC estimation in the presence of uncertain or time-varying model parameters.

SoC is the percentage ratio of the present battery capacity to its nominal capacity. Two straightforward yet typical non-model-based SoC estimation methods are voltage translation and Coulomb counting. The former infers the SoC from a predetermined lookup table that relates open-circuit (OCV) voltage with SoC using the OCV measurement. Despite reliability, it requires the battery to rest for a long period with cutting off from the external circuit to measure the OCV, thus restricting its practical implementation without interrupting system operation. Coulomb counting, which is based on numerical integration of the current over time, may suffer from a ‘drift’ of SoC estimates from the true values due to
cumulative integration errors and noise corruption. For a survey of both methods, please refer to [38, 39] and the references therein.

A notable recent trend is the increasing emphasis on model-based estimation methods. The dynamic models, derived from either equivalent circuits or electrochemical principles, facilitate the assimilation of the battery data and lead to real-time SoC estimation with bounded errors. While battery modeling has been well-accomplished [40], more attention is being geared towards the development of estimation algorithms. Application of the KF techniques has been remarkable in this respect. The classical linear KF and its extensions to nonlinear systems, including the EKF, UKF and IEKF, have been used to deal with SoC estimation based on electrochemical and equivalent circuit models, see [41, 42, 43, 44, 45, 46, 47, 48, 49, 50]. A variety of other state observers originating from control approaches have also played a role in constructing SoC estimators. Here, we highlight the sliding mode observer [51], adaptive model reference observer [52], Lyapunov-based observer [53] and PDE-based observer [54, 55].

Since a good model is a prerequisite, model-based SoC estimation typically follows after the procedures of dynamic modeling and parameter identification. However, accurate identification is challenging. First, the parameters in a battery model are often subject to changes with time and operational conditions. For instance, the internal resistance will rise and the capacity diminish as a result of battery aging. Another example is the charging and discharging efficiencies, which are dependent on the SoC, magnitude of current and temperature. Second, the parameters may differ from one battery to another, making identification for
each battery at least rather cumbersome. Therefore, adaptive approaches are more desirable, merging both identification and SoC estimation in one step. As shown in Fig. 3.1, an adaptive SoC estimator gives not only the SoC estimates but also the estimates of the model parameters in real time after assimilating the current-voltage data on the basis of a model. The parameter estimates will then be used to update the model to aid the next-step estimation.

Adaptive SoC estimation has been attracting considerable attention in the recent literature. An adaptive EKF-based SoC estimator is designed in [46], which interacts with a parameter estimator. In [48], state augmentation is conducted to incorporate the SoC variable and model parameters, and then the UKF is applied to estimate the augmented state. However, the convergence, and as a result, the accuracy, are noted to be difficult to guarantee. In [50, 56], an adaptive SoC estimator is developed using the IEKF, guided by an analysis of the observability/identifiability. Novel adaptive PDE observers for SoC estimation have also been reported in [57]. Here, we aim to make new contributions to study of this topic, with the aim of attaining adaptive, high-fidelity and easy-to-implement SoC estimation.

Our primary contribution lies in seamlessly linking the notion of ‘multiple models’ and adaptive SoC estimation and taking a lead with the development of multi-model adaptive SoC estimation approaches. A multitude of models, compared to a single one, can give a better description of complicated uncertain dynamics [35, 58, 59], thus particularly suitable to deal with the tasks relevant to batteries. In addition, we introduce the multi-model framework for battery management and control, pointing out that many more battery management strategies can be improved on a multi-model basis.

In this study, an electrochemical model with reduced complexity in structure is obtained in the first place. Here, we take the Li\(^+\) batteries for an example, which are the most prevalent batteries today. Multiple submodels are brought up from this model by fixing some parameters and assuming the others unknown. Each submodel is shown locally observable with admissible inputs by rigorous analysis. This attempt, despite its importance, has been rarely made in the literature
on adaptive SoC estimation. Then, an adaptive SoC estimation scheme will be implemented simultaneously but separately to each submodel, with the submodel in each implementation assumed true. The SoC estimates resulting from different submodels will be fused in the light of a certain strategy to obtain the final estimate. As such, we boost the accuracy of SoC estimation despite the presence of uncertainties plaguing battery models.

3.1 A Reduced-Complexity Model

In this section, the working mechanism of Li$^+$ batteries is briefly introduced first, followed by a review of the single particle model (SPM) is presented. Then a reduction of the SPM is developed for the purpose of SoC estimation.

3.1.1 Working Mechanism of Li$^+$ Batteries

A schematic description of a Li$^+$ battery is shown in Fig a. The positive electrode is typically made from Li compounds, e.g., $\text{Li}_x\text{Mn}_2\text{O}_4$ and $\text{Li}_x\text{CoO}_2$. Small solid particles of the compounds are compressed together, giving birth to a porous structure. The negative electrode is also porous, which usually contains graphite particles. The interstitial pores at both electrodes provide intercalation space, where the Li$^+$ ions can be moved in and out and stored. The electrolyte contains free ions and is electrically conductive, where the Li$^+$ ions can be transported easily. The separator physically separates the electrodes apart. It allows the migration of Li$^+$ ions from one side to the other, but prevents electrons from passing through. The electrons are thus forced to flow through the external circuit.

During the charging process, Li$^+$ ions are extracted from the particles at the positive electrode into the electrolyte, and the particles at the negative electrode absorbs Li$^+$ ions from the electrolyte. This process not only generates an influx of Li$^+$ ions within the battery, but also builds up a potential difference between the positive and negative electrodes. In the reverse process the battery becomes discharged. The following equations exemplify the chemical reactions in the positive
Figure 3.2: (a) Schematic characterization of a Li$^+$ battery; (b) the single-particle model.

and negative electrodes:

\[
\begin{align*}
\text{Li}_x\text{Mn}_2\text{O}_4 & \xrightleftharpoons{\text{charge}}^{\text{discharge}} \text{Li}_{x-y}\text{Mn}_2\text{O}_4 + y \text{Li}^+ + ye^- \\
x \text{Li}^+ + x e^- + C & \xrightleftharpoons{\text{charge}}^{\text{discharge}} Li_xC
\end{align*}
\]

3.1.2 Single Particle Model

Let us consider the single particle model (SPM) (see Fig. 3.2b), which, as the name suggests, simplifies each electrode as a spherical particle with area equivalent to the active area of this electrode [60, 61]. Although only able to capture key physical and chemical phenomena, it decreases complexities in identification,
Table 3.1: Nomenclature for the battery model.

<table>
<thead>
<tr>
<th>Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Phi_s$</td>
<td>electric potential in the solid electrode</td>
</tr>
<tr>
<td>$\Phi_e$</td>
<td>electric potential in the electrolyte</td>
</tr>
<tr>
<td>$c_s$</td>
<td>concentration of Li$^+$ in the solid electrode</td>
</tr>
<tr>
<td>$c_{ss}$</td>
<td>concentration of Li$^+$ at a particle’s spherical surface</td>
</tr>
<tr>
<td>$J$</td>
<td>molar flux of Li$^+$ at the particle’s surface</td>
</tr>
<tr>
<td>$J_0$</td>
<td>exchange current density</td>
</tr>
<tr>
<td>$\eta$</td>
<td>overpotential of reaction in the cell</td>
</tr>
<tr>
<td>$U$</td>
<td>open-circuit potential</td>
</tr>
<tr>
<td>$I$</td>
<td>external circuit current</td>
</tr>
<tr>
<td>$V$</td>
<td>terminal voltage</td>
</tr>
<tr>
<td>$r$</td>
<td>radial dimension of the particle</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Physical parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_s$</td>
<td>diffusion coefficient of Li$^+$ in the solid electrode</td>
</tr>
<tr>
<td>$\bar{r}$</td>
<td>radius of the spherical particle</td>
</tr>
<tr>
<td>$F$</td>
<td>Farady’s constant</td>
</tr>
<tr>
<td>$S$</td>
<td>specific interfacial area</td>
</tr>
<tr>
<td>$T$</td>
<td>temperature of the cell</td>
</tr>
<tr>
<td>$\alpha^a$</td>
<td>anodic charge transport coefficient</td>
</tr>
<tr>
<td>$\alpha^c$</td>
<td>cathodic charge transport coefficient</td>
</tr>
<tr>
<td>$R$</td>
<td>universal gas constant</td>
</tr>
<tr>
<td>$R_c$</td>
<td>phase resistance</td>
</tr>
<tr>
<td>$R_f$</td>
<td>film resistance of the solid electrolyte interphase</td>
</tr>
<tr>
<td>$\rho$</td>
<td>Coulombic efficiency of the cell</td>
</tr>
<tr>
<td>$C_n$</td>
<td>nominal capacity of the cell</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subscripts</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>solid electrode phase</td>
</tr>
<tr>
<td>$e$</td>
<td>electrolyte phase</td>
</tr>
<tr>
<td>$n$</td>
<td>negative electrode</td>
</tr>
<tr>
<td>$p$</td>
<td>positive electrode</td>
</tr>
<tr>
<td>$j$</td>
<td>$n$ or $p$</td>
</tr>
</tbody>
</table>


estimation and control design for battery operations to a large extent [43, 55]. To proceed further, an introduction of the SPM is given below, with the nomenclature shown in Table 3.1.

**Input and output of the battery:** The external input to the battery is the current $I(t)$ with $I(t) < 0$ for charge and $I(t) > 0$ for discharge. The output terminal voltage is the potential difference between the two electrodes, that is,

$$V(t) = \Phi_{s,p}(t) - \Phi_{s,n}(t).$$ (3.1)

**Conservation of $Li^+$ in the electrode phase:** The migration of $Li^+$ ions inside a solid particle is caused by the gradient-induced diffusion. It follows from the Fick’s laws of diffusion that

$$\frac{\partial c_{s,j}(r,t)}{\partial t} = \frac{1}{r^2} \frac{\partial}{\partial r} \left( D_{s,j} r^2 \frac{\partial c_{s,j}(r,t)}{\partial r} \right),$$ (3.2)

with the initial and boundary conditions given by

$$c_{s,j}(r,0) = c^0_s, \quad \frac{\partial c_{s,j}}{\partial r} \bigg|_{r=0} = 0, \quad \frac{\partial c_{s,j}}{\partial r} \bigg|_{r=r_j} = -\frac{1}{D_{s,j}} J_j.$$

Here, $J_j$ is the molar flux at the electrode/electrolyte interface of a single particle. When $j = n$ and $p$, respectively,

$$J_n(t) = \frac{I(t)}{F S_n}, \quad J_p(t) = -\frac{I(t)}{F S_p}.$$ (3.3)

**Electrochemical kinetics:** The molar flux $J_j$ is governed by the Butler-Volmer equation:

$$J_j(t) = \frac{J_{0,j}}{F} \left[ \exp \left( \frac{\alpha_e F \eta_j(t)}{R T} \right) - \exp \left( -\frac{\alpha_e F \eta_j(t)}{R T} \right) \right],$$ (3.4)

where

$$\eta_j(t) = \Phi_{s,j}(t) - \Phi_{e,j}(t) - U(c_{ss,j}(t)) - F R_j J_j(t).$$
The electrolyte phase can be represented by a resistor $R_{c,j}$ in the SPM, implying $\Phi_{c,j}$ can be expressed as

$$\Phi_{e,j}(t) = R_{c,j} I(t).$$

Hence, $\eta_j$ becomes

$$\eta_j(t) = \Phi_{s,j}(t) - U(c_{ss,j}(t)) - F\bar{R}_j J_j(t), \quad (3.5)$$

where $\bar{R}_j = R_{c,j} + R_{f,j}$.

To conclude the model review, the SPM is composed of (3.1)-(3.4). A visualization of the relationship between key variables is given in Fig. 3.3, in which $I$ is the external input, $c_{s,j}$ and $\Phi_{s,j}$ are the variables showing the battery status, and $V$ is the output.

### 3.1.3 Reduced-Complexity State-Space Model

*Average Li$^+$ concentration in the electrode phase:* The average concentration of Li$^+$ ions in the particle is considered as the measure of the present battery capacity, or equivalently, the SoC. It is defined as

$$c_{s,j}^{\text{avg}}(t) = \frac{1}{\Omega} \int_{\Omega} c_{s,j}(r,t)d\Omega, \quad (3.6)$$

where $\Omega$ denotes the volume of the particle sphere. From (3.2), it is obtained that

$$\dot{c}_{s,j}^{\text{avg}}(t) = \frac{1}{\Omega} \int_{\Omega} \frac{\partial c_{s,j}(r,t)}{\partial t} d\Omega.$$
\[
\frac{1}{\Omega} \int_{\Omega} \frac{1}{r^2} \frac{\partial}{\partial r} \left( D_{s,j} r^2 \frac{\partial c_{s,j}(r,t)}{\partial r} \right) d\Omega
= \epsilon_j D_{s,j} \left. \frac{\partial c_{s,j}(r,t)}{\partial r} \right|_{r=r_j},
\]
(3.7)

where \( \epsilon_j \) is a constant coefficient. Depending on the electrode polarity, (3.7) splits into

\[
c_{s,n}^{\text{avg}}(t) = -\frac{\epsilon_n}{FS_n} I(t),
\]
(3.8)
\[
c_{s,p}^{\text{avg}}(t) = \frac{\epsilon_p}{FS_p} I(t).
\]
(3.9)

By (3.8)-(3.9), the rate of change of \( c_{s,j}^{\text{avg}} \) is linearly proportional to the input current \( I \). In other words, \( c_{s,j}^{\text{avg}} \) is equal to the initial value \( c_{s,j}^{\text{avg}}(0) \) plus integration of \( I \) over time. This illustrates that the change of SoC depends linearly on \( I \) as a result of \( c_{s,j}^{\text{avg}} \) indicating SoC. Such a relationship has not only been presented for electrochemical models, e.g., [62], but also justified in ECMs, e.g., [42, 63] and the references therein.

**Terminal voltage:** Suppose there exists a function \( \varphi \) such that \( c_{s,j}^{\text{avg}}(t) = \varphi(c_{s,j}^{\text{avg}}(t)) \) and define \( \bar{U} = U \circ \varphi \), where ‘\( \circ \)’ denotes composition of two functions. Using (3.5), (3.1) becomes

\[
V(t) = \bar{U}(c_{s,p}^{\text{avg}}(t)) - \bar{U}(c_{s,n}^{\text{avg}}(t)) + \eta_p(t) - \eta_n(t) + (\bar{R}_p - \bar{R}_n)I(t).
\]

With \( \alpha_a = \alpha_c = 0.5 \), it follows from (3.4) that

\[
\eta_n(t) = \frac{2RT}{F} \sinh^{-1} \left( \frac{J_n(t)F}{2J_{0,n}} \right) = \frac{2RT}{F} \sinh^{-1} \left( \frac{\epsilon_n I(t)}{2J_{0,n}} \right),
\]
\[
\eta_p(t) = \frac{2RT}{F} \sinh^{-1} \left( \frac{J_p(t)F}{2J_{0,p}} \right) = \frac{2RT}{F} \sinh^{-1} \left( -\frac{\epsilon_p I(t)}{2J_{0,p}} \right).
\]

Thus \( V(t) \) becomes

\[
V(t) = \bar{U}(c_{s,p}^{\text{avg}}) - \bar{U}(c_{s,n}^{\text{avg}})
\]
\[
\frac{2RT}{F} \left[ \sinh^{-1} \left( -\frac{\epsilon_p I(t)}{2J_{0,p}} \right) - \sinh^{-1} \left( \frac{\epsilon_n I(t)}{2J_{0,n}} \right) \right] + (R_p - R_n) I(t).
\] 

(3.10)

As such, \( V(t) \) consists of two parts. The first is the open-circuit voltage (OCV) that relies on \( \bar{U}(c_{s,j}^{\text{avg}}) \), and the second part is the direct feedthrough from \( I \) to \( V \).

Construction of the state-space model: It is seen from above that (3.8)-(3.10) provide a concise characterization of the battery dynamics. To convert them into a state-space model for SoC estimation, denote the SoC by a state vector \( x \in [0, 1] \). The input \( u \) and the output \( y \) of the model can be defined as \( u = I \) and \( y = V \), respectively. Since \( c_{s,j}^{\text{avg}} \) is arguably equivalent to the SoC, the following state-space model can then be constructed on the basis of (3.8)-(3.10):

\[
\begin{cases}
\dot{\text{SoC}}(t) = -au(t), \\
y(t) = h(\text{SoC}(t)) + g(u(t)).
\end{cases}
\]

In above, \( a = \rho/C_n \), where the Coulombic efficiency \( \rho \) is a measure of the transfer efficiency of the current charge and \( C_n \) is the battery’s nominal capacity, \( g_1(\cdot) \) is the counterpart of the part containing \( \bar{U} \) in (3.10), and \( g_2(\cdot) \) corresponds to the part involving \( I \) in (3.10). Discretization of the above system yields

\[
\begin{cases}
\text{SoC}_{k+1} = \text{SoC}_k - K_0 u_k, \\
y_k = g_1(\text{SoC}_k) + g_2(u_k).
\end{cases}
\]

(3.11)

where \( K_0 = aT = \rho T/C_n \) and \( T \) is the sampling period.

Note that, \( h(\cdot) \) represents the SoC-OCV relationship and thus varies with different batteries. For the battery under consideration, it takes the parametric form as follows:

\[
g_1(x) = K_1 + K_2 \ln(\tau_1 + \text{SoC}_k) + K_3 \ln(\tau_2 + 1 - \text{SoC}_k),
\]

where \( K_i \) for \( i = 1, 2, 3 \) are parameters and \( \tau_1 \) and \( \tau_2 \) are predetermined constants.
In addition, \( g(\cdot) \) can be determined from (3.10):

\[
g_2(u) = \gamma_0 \left[ \sinh^{-1}(\gamma_1 u) - \sinh^{-1}(\gamma_2 u) \right] + \gamma_3 u,
\]

where \( \gamma_i \) for \( i = 0, 1, 2, 3 \) are parameters from (3.10). The order of magnitude of \( \gamma_i \) for each \( i \) is: \( \gamma_0 \approx 10^{-2} \), \( \gamma_1 \approx -(10^{-7} \sim 10^{-6}) \), \( \gamma_2 \approx 10^{-7} \sim 10^{-6} \) and \( \gamma_3 \approx 10^{-3} \sim 10^{-2} \). A typical \( u_k \) lies within the range of \(-20 \sim 20\). This indicates that the term \( \gamma_0 \left[ \sinh^{-1}(\gamma_1 u) - \sinh^{-1}(\gamma_2 u) \right] \) plays a negligible role in \( g_2(u) \). Therefore, further simplification can be made to reduce \( g_2(u) \) as \( g_2(u) = Ru \), where \( R \) is the internal resistance.

Hence, the model we will consider in the follow is expressed as

\[
\begin{aligned}
\text{SoC}_{k+1} &= \text{SoC}_k - K_0 u_k, \\
y_k &= K_1 + K_2 \ln(\tau_1 + \text{SoC}_k) + K_3 \ln(\tau_2 + 1 - \text{SoC}_k) + Ru_k,
\end{aligned}
\tag{3.12}
\]

which will be called ‘modified Nernst model’. Developed for SoC estimation, the model in (3.12) contains parameters \( K_i \) for \( i = 0, 1, 2, 3 \) and \( R \). Their values are often hard to determine jointly and may even be subject to change over time in practice. However, we may gain some \textit{a priori} knowledge of \( K_i \) for \( i = 1, 2, 3 \) from the experiment that calibrates SoC and OCV. Specifically, let us apply a very small current to charge the battery from zero to full capacity, and then discharge from full to zero. Because the Coulombic efficiency \( \rho \approx 1 \) and \( Ru_k \) can be ignored when the magnitude of the charging/discharging current is low enough, the SoC can be directly calculated through integration of the current over time, and the measured output voltage can be regarded as the OCV. Using the SoC-OCV data set collected, the parameters \( K_i \) for \( i = 1, 2, 3 \) can be estimated. This can be formulated as a nonlinear least squares data fitting problem, which can be easily addressed by numerical methods such as the Gauss-Newton [64]. It deserves attention that such estimates \( K_i \) for \( i = 1, 2, 3 \) are rough in general since variations will happen in the running of batteries. Next, we will develop multi-model adaptive SoC estimation approaches based on the model in (3.12).
3.2 Multiple Model Construction and Analysis

3.2.1 Multiple Model Construction

For adaptive SoC estimation, we will perform simultaneous estimation of the SoC and the parameters. To obtain a locally observable model, one or several parameters usually need to be fixed in order to estimate the others and the SoC. A few options may exist regarding which parameters are fixed. Based on our experience with the considered model, we separate the parameters into two sets, fix one set and augment the state vector to incorporate the SoC and the other set. Accordingly, two submodels will be constructed.

Letting $K_0$ and $K_1$ be fixed, the first one can be obtained:

$$\mathcal{M}_1 : \begin{cases} x_{k+1}^1 = f^1(x_k^1, u_k), \\ y_k = h^1(x_k^1, u_k), \end{cases} \quad (3.13)$$

where

$$x_k^1 = \begin{bmatrix} \text{SoC} & K_2 & K_3 & R \end{bmatrix}^T,$$

$$f^1(x_k^1, u_k) = x_k^1 - \left[ K_0 \ 0 \ 0 \right]^T u_k,$$

$$h^1(x_k^1, u_k) = K_1 + x_{k,2}^1 \ln(\tau_1 + x_{k,1}^1) + x_{k,3}^1 \ln(\tau_2 + 1 - x_{k,1}^1) - x_{k,4}^1 u_k.$$

Analogously, by fixing $K_i$ for $i = 1, 2, 3$, we have

$$\mathcal{M}_2 : \begin{cases} x_{k+1}^2 = f^2(x_k^2, u_k), \\ y_k = h^2(x_k^2, u_k), \end{cases} \quad (3.14)$$

where

$$x_k^2 = \begin{bmatrix} \text{SoC} & K_0 & R \end{bmatrix}^T,$$

$$f^2(x_k^2, u_k) = \begin{bmatrix} x_{k,1}^2 - x_{k,2}^2 u_k & x_{k,2}^2 & x_{k,3}^2 \end{bmatrix}^T,$$

$$h^2(x_k^2, u_k) = K_1 + K_2 \ln(\tau_1 + x_{k,1}^2) + K_3 \ln(\tau_2 + 1 - x_{k,1}^2) - x_{k,3}^2 u_k.$$
Remark 3.1. In an implicit manner, \( \mathcal{M}_1 \) places more confidence on the state equation of (3.12), assuming that \( K_0 \) is accurate, while the belief in the measurement equation of (3.12) is emphasized in \( \mathcal{M}_2 \) similarly. Nevertheless, it is noteworthy that the confidence level on each submodel during the estimation process is dynamically determined by the fusion strategy outlined earlier in Section 3.3.

Remark 3.2. An extended series can be constructed on the basis of each submodel if we let the parameters take different values that are believed to be close or equal to the truth. For instance, the Coulombic efficiency may be 100%, 90% or even 80% depending on the operating conditions. Then \( \mathcal{M}_1 \) will give birth to three more submodels if \( K_0 \) assumes \( T/C_n, 0.9T/C_n \) and \( 0.8T/C_n \), respectively. This allows considerable flexibility for us to describe the battery dynamics and brings improvements to the single-model case.

3.2.2 Observability Analysis

It is well-known that state estimation requires a ‘certain’ kind of observability of the system. Hence, we will analyze the observability properties of \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \) before proceeding to SoC estimation.

Consider a general single-input-single-output (SISO) system

\[
\mathcal{S} : \begin{cases} 
  x_{k+1} = f(x_k, u_k), \\
  y_k = h(x_k, u_k), 
\end{cases}
\tag{3.15}
\]

where \( x \in X \) of dimension \( n \), \( y \in Y \) and \( u \in U \). We assume that 1) \( X \) and \( Y \) connected, second countable, Hausdorff, differentiable manifolds of class \( C^q \) with \( q \in \mathbb{N} \), 2) \( U \) is an open interval of \( \mathbb{R} \), and 3) \( f : X \times U \to X \) and \( h : X \to Y \) are of class \( C^q \). For convenience, \( f(x, u) \) is denoted as \( f^u(x) \), and \( h(f(x, u_0), u_1) = h^u_1 \circ f^u_0(x) \). Following [65, 66], the local observability for \( \mathcal{S} \) is defined as follows:

Definition 3.1. (Distinguishability) Two states \( x \) and \( x^* \) are said to be indistinguishable, written as \( x \rightleftharpoons x^* \), if for each \( l \neq 0 \) and for each input sequence,
\{u_0, \ldots, u_l\} \in \mathcal{U}^l, \text{ we have}

\[ h^{u_i} \circ f^{u_{i-1}} \circ \cdots \circ f^{u_0}(x) = h^{u_i} \circ f^{u_{i-1}} \cdots \circ f^{u_0}(x^*) \].

Otherwise, they are distinguishable.

**Definition 3.2.** *(Local observability)* The system \( \mathcal{S} \) is locally observable if for any state \( x^o \in X \), there exists a neighborhood \( \mathcal{D} \) of \( x^o \) such that, \( x \leftrightarrow x^* \) implies \( x = x^* \) for each \( x, x^* \in \mathcal{D} \).

By Definitions 3.1-3.2, local observability means that \( x^o \) can be distinguished from its neighbors given the input sequence \( \{u_0, \ldots, u_l\} \) and the output sequence \( y_0, \ldots, y_l \). It should be noted that this definition of observability depends not only on the system itself but also on the applied inputs, unlike the uniform observability for any inputs defined in [67]. While one sees various definitions of nonlinear observability in the literature, this does not obstruct our discussion since they are usually about ‘different measurements results from different initial states (for admissible inputs)’. The interested reader can refer to the literature on the subject, e.g., [68].

To address the observability condition, the following sets of functions are defined:

\[ \Omega_0 = \{h(\cdot)\} \]
\[ \Omega_l = \{h^{u_j} \circ f^{u_{j-1}} \circ \cdots \circ f^{u_0}(\cdot): u_i \in \mathcal{U} \forall i = 1, \ldots, j \text{ and } 1 \leq j \leq l\} \]
\[ \Omega = \bigcup_{j \geq 0} \Omega_l \].

An observability criterion is presented in the following theorem, please see [65] for the proof.

**Theorem 3.3.** [65] If \( \dim d\Omega(x) = n \ \forall x \in X \), then the system \( \mathcal{S} \) is locally observable.

Theorem 3.3 gives a sufficient condition to determine the local observability by relating it to the full dimensionality of the codistribution \( d\Omega \). Now the local
observability of $\mathcal{M}_1$ and $\mathcal{M}_2$ can be analyzed using Theorem 3.3. Let us take $\mathcal{M}_1$ for an example since the analysis for both follows similar lines.

Note that $f^1$ and $h^1$ are of class $C^\infty$. Suppose that the initial state is $x^1_0$ for $\mathcal{M}_1$ and that there are $L$ measurements $\{y_1, \ldots, y_L\}$. By (3.13), $x^1_k$ is given by

$$x^1_k = x^1_0 - \left[ K_0 \begin{array}{ccc} 0 & 0 & 0 \end{array} \right]^\top \sum_{i=0}^{k-1} u_i.$$ 

Hence, we have

$$\bar{h}^1_k(x^1_0) = h^1 u_k \circ f^{1u_{k-1}} \circ \cdots \circ f^{1u_0}(x^1_0)$$

$$= K_1 + x^1_{0,2} \ln \left( \tau_1 + x^1_{0,1} - K_0 \sum_{i=0}^{k-1} u_i \right)$$

$$+ x^1_{0,3} \ln \left( \tau_2 + 1 - x^1_{0,1} + K_0 \sum_{i=0}^{k-1} u_i \right) - x^1_{0,4} u_k,$$

where $\bar{h}^1_k \in \Omega$. Define a matrix $J$ with dimensions $L \times 4$:

$$J = \left[ \frac{d\bar{h}^1_1}{dx^1_0} \quad \cdots \quad \frac{d\bar{h}^1_1}{dx^1_0} \quad \cdots \quad \frac{d\bar{h}^1_L}{dx^1_0} \right]^\top.$$

The elements in the $k$-th row of $J$ are

$$J_{k,1} = \frac{\partial \bar{h}^1_k}{\partial x^1_{0,1}} = \frac{x^1_{0,2}}{\tau_1 + x^1_{0,1} - K_0 \sum_{i=0}^{k-1} u_i} - \frac{x^1_{0,3}}{\tau_2 + 1 - x^1_{0,1} + K_0 \sum_{i=0}^{k-1} u_i},$$

$$J_{k,2} = \frac{\partial \bar{h}^1_k}{\partial x^1_{0,2}} = \ln \left( \tau_1 + x^1_{0,1} - K_0 \sum_{i=0}^{k-1} u_i \right),$$

$$J_{k,3} = \frac{\partial \bar{h}^1_k}{\partial x^1_{0,3}} = \ln \left( \tau_2 + 1 - x^1_{0,1} + K_0 \sum_{i=0}^{k-1} u_i \right),$$

$$J_{k,4} = \frac{\partial \bar{h}^1_k}{\partial x^1_{0,4}} = -u_k.$$

By observation, we have the following conclusions:

- The submodel $\mathcal{M}_1$ is locally observable if a suitable input sequence $\{u_k\}$
is applied. By ‘suitable’, we mean that $u_k$ varies sufficiently in magnitude over time, or in other words, \{u_k\} contains a rich mix of frequency contents. In this case, $J$ will have full column rank, and as a result, $\dim d\Omega$ has a dimension of 4, satisfying the condition in Theorem 3.3. It should be emphasized such a condition imposed on the input is a mild constraint that can be easily satisfied when a battery is in use.

- We can analogously determine that $M_2$ is also locally observable if a suitable \{u_k\} is used to excite the system.

- Additional submodels other than $M_1$ and $M_2$ can be constructed by fixing different parameters. An example is to fix only $K_1$, which will lead to another locally observable model. However, no matter how many submodels are used, the essence of multi-model adaptive SoC estimation remains the same, as will be seen in Section 3.4. It is also noteworthy that the resultant submodel will be unobservable if all the parameters are assumed unknown.

### 3.3 Basics of Multi-model Estimation

The structure of a typical multi-model estimator is shown in Fig. 3.4. In this section, we give a review of the multi-model estimation, with an emphasis on the estimate fusion strategy.

Its first part is composed of a bank of parallel filters based on different models. Each filter assimilates the data to produce its own estimate. All the
estimates will then be fused to give the best estimate. Many options exist for the elemental filter, such as the KF for a linear model or the EKF for a nonlinear one. What is of particular interest here is the design of the fusion strategy.

Let us consider a general system. Its unknown state at time instant \( k \) is denoted by \( x_k \in \mathbb{R}^n \) and its measurement by \( z_k \in \mathbb{R}^n \). Different models are available to describe the system, leading to a model set \( M = \{ M_1, M_2, \cdots, M_N \} \). Suppose that \( M_i \) is given by

\[
M_i : \begin{align*}
x_{k+1} &= f_i(x_k) + w^i_k, \\
z_k &= h_i(x_k) + v^i_k,
\end{align*}
\]

where \( f^i \) and \( h^i \) are \( C^1 \) functions to represent the state transition and measurement, respectively, and \( \{w^i_k\} \) and \( \{v^i_k\} \) are uncorrelated, zero-mean, white Gaussian noise sequences with covariances \( Q^i_k \geq 0 \) and \( R^i_k > 0 \), respectively. While assuming that the true system coincides with one model at each time instant, we do not know which model matches the system at any time. Thus a probabilistic description is used. Let \( s_k \) denote the system running status at \( k \). It may take any \( M_i \) for \( i = 1, 2, \cdots, N \) to address the uncertainty of model matching. The probability of the event \( s_k = M_i \) is denoted as \( p(s_k = M_i) \), or simply, \( p(s^i_k) \). In other words, \( p(s^i_k) \) indicates the a priori probability that the true model is \( M_i \) at time \( k \). Obviously, \( \sum_{i=1}^{N} p(s^i_k) = 1 \).

From a statistical perspective, \( x_k \) and \( z_k \) are continuous random variables and \( s_k \) a discrete one. Without causing confusion, we use the symbol \( p \) to denote the probability density function (pdf), probability mass function (pmf) or mixed pdf-pmf in the sequel for convenience. We define the information set as \( Z_k = \{z_1, z_2, \cdots, z_k\} \) and intend to estimate \( x_k \) from \( Z_k \), hence considering \( p(x_k|Z_k) \).

By the Bayes’ theorem, we have

\[
p(x_k|Z_k) = \sum_{i=1}^{N} p(x_k, s^i_k|Z_k) = \sum_{i=1}^{N} p(x_k|s^i_k, Z_k)p(s^i_k|Z_k).
\]

(3.17)
When \( p(x_k|Z_k) \) becomes available, we can carry out minimum-mean-square-error (MMSE) estimation or Maximum a Posteriori (MAP) estimation of \( x_k \):

**MMSE:** \( \hat{x}_{k|k} = \mathbb{E}(x_k|Z_k) = \int x_k p(x_k|Z_k) \, dx_k \),

**MAP:** \( \hat{x}_{k|k} = \arg \max_{x_k} p(x_k|Z_k) \).

Independent of the method (MMSE or MAP) used, it follows from (3.17) that

\[
\hat{x}_{k|k} = \sum_{i=1}^{N} \hat{x}_{k|k}^i p(s_k^i|Z_k),
\]

(3.18)

where \( \hat{x}_{k|k}^i \) is the estimate of \( x_k \) based on the model \( M_i \). An observation from this analysis is that \( p(s_k^i|Z_k) \) turns out to be a probabilistic weight coefficient. The associated estimation error covariance is

\[
P_{k|k} = \mathbb{E} \left[ (\hat{x}_k - x_k)(\hat{x}_k - x_k)^\top | Z_k \right]
= \int (\hat{x}_k - x_k)(\hat{x}_k - x_k)^\top p(x_k|Z_k) \, dx_k
= \sum_{i=1}^{N} \int (\hat{x}_k - x_k)(\hat{x}_k - x_k)^\top p(x_k, s_k^i|Z_k) \, dx_k
= \sum_{i=1}^{N} \int (\hat{x}_k - x_k)(\hat{x}_k - x_k)^\top p(x_k|s_k^i, Z_k) \, dx_k p(s_k^i|Z_k)
= \sum_{i=1}^{N} \left[ P_{k|k}^i + (\hat{x}_k - \hat{x}_k^i)(\hat{x}_k - \hat{x}_k^i)^\top \right] p(s_k^i|Z_k).
\]

(3.19)

Let us take a closer look at \( p(s_k^i|Z_k) \). Using the Bayes’ theorem again, we see that

\[
p(s_k^i|Z_k) = \frac{p(s_k^i, Z_k)}{p(Z_k)}
= \frac{p(z_k|s_k^i, Z_{k-1}) p(s_k^i|Z_{k-1})}{p(z_k|Z_{k-1})}
= \frac{p(z_k|s_k^i, Z_{k-1}) p(s_k^i|Z_{k-1})}{\sum_{j=1}^{N} p(z_k, s_k^j|Z_{k-1})}
\]
\[
p(z_k|s_k^i, Z_{k-1}) = \frac{p(s_k^i|Z_{k-1}) p(s_k^i|Z_{k-1})}{\sum_{j=1}^{N_j} p(z_k|s_k^j, Z_{k-1}) p(s_k^j|Z_{k-1})}, \quad (3.20)
\]

Furthermore, we have

\[
p(z_k|s_k^i, Z_{k-1}) = \int p(z_k, x_k|s_k^i, Z_{k-1}) dx_k
\]

\[
= \int p(z_k|x_k, s_k^i, Z_{k-1}) p(x_k|s_k^i, Z_{k-1}) dx_k
\]

\[
= \int p(z_k|x_k, s_k^i) p(x_k|s_k^i, Z_{k-1}) dx_k.
\]

Under the mildly simplified assumption that \(p(z_k|x_k, s_k^i, Z_{k-1}) = \mathcal{N}(h^i(x_k), R^i_k)\) and \(p(x_k|s_k^i, Z_{k-1}) = \mathcal{N}(\dot{x}_{k|k-1}^i, P_{k|k-1}^i)\), \(p(z_k|s_k^i, Z_{k-1})\) can be approximated as

\[
p(z_k|s_k^i, Z_{k-1}) \approx (2\pi)^{-n/2} \left| S_k^i \right|^{-1/2} \exp \left[ -\frac{1}{2} (\tilde{z}_k^i)^\top S_k^i \tilde{z}_k^i \right], \quad (3.21)
\]

where

\[
\tilde{z}_k^i = z_k - h^i(\dot{x}_{k|k-1}^i),
\]

\[
S_k^i = H_k^i P_{k|k-1}^i (H_k^i)^\top + R_k^i,
\]

\[
H_k^i = \frac{\partial h^i}{\partial x} (\dot{x}_{k|k-1}^i).
\]

Furthermore,

\[
p(s_k^i|Z_{k-1}) = \frac{p(Z_{k-1}|s_k^i) p(s_k^i)}{p(Z_{k-1})} = p(s_k^i),
\]

since \(p(Z_{k-1}|s_k^i) = 1\) and \(p(Z_{k-1}) = 1\) because \(Z_{k-1}\) is an event with probability 1 at time \(k\). If we define \(\mu_k^i = p(s_k^i|Z_k)\) and \(w_k^i = p(z_k|s_k^i, Z_{k-1})\) and suppose \(\pi_k^i = p(s_k^i)\), (3.20) becomes

\[
\mu_k^i = \frac{w_k^i \pi_k^i}{\sum_{j=1}^{N_j} w_k^j \pi_k^j}. \quad (3.22)
\]

Hence, by (3.18)-(3.19), the fusion strategy, or the fuser as is called, is given
by

$$\hat{x}_{k|k} = \sum_{i=1}^{N} \hat{x}_{k|i} \mu_i^k, \quad (3.23)$$

$$P_{k|k} = \sum_{i=1}^{N} \left[ P_{k|i} + (\hat{x}_{k|k} - \hat{x}_{k|i})(\hat{x}_{k|k} - \hat{x}_{k|i})^\top \right] \mu_i^k. \quad (3.24)$$

The final conclusion drawn from this analysis is as follows: the fused estimate (covariance) is a linear weighted combination of the estimates from the elemental filters. It can be noted that

- The estimation is based on a series of elemental filters and the fusion. The process is similar to a ‘weight-based reconciliation’, which balances the role that different models potentially play in the estimation task.

- The residuals of the elemental filter based on the ‘correct’ model that best match the true system is expected to be remarkably smaller than those of the others [58]. As a result, the probabilistic weight associated to this filter, say, $\mu_k^*$, will tend to increase and downplay the others. The fused estimate will approach the estimate based on the correct model.

### 3.4 Multi-model Adaptive SoC Estimation

In this section, we study multi-model adaptive SoC estimation on the basis of Sections 3.3 and 3.2. An IEKF-based elemental filter will be applied to $M_1$ and $M_2$, respectively, for adaptive SoC estimation. The overall estimate will be obtained by fusing all the estimates for the elemental filters, leading to the MM-AdaSoC algorithm.

Consider applying the IEKF to the system in (3.15). At $k - 1$, prediction can be made about the next time instant. The formulas are as follows:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}), \quad (3.25)$$

$$P_{k|k-1} = F_{k-1}^T P_{k-1|k-1} F_{k-1}^T + Q, \quad (3.26)$$
where $\hat{x}$ is the estimate of $x$, $P$ is the error covariance, $Q \geq 0$ is an adjustable matrix to account for the process noise, and $F$ is given by

$$F_{k-1} = \frac{\partial f}{\partial x}(\hat{x}_{k-1|k-1}, u_{k-1}).$$

When the measurement $y_k$ arrives, $\hat{x}_{k|k-1}$ can be updated by the new information $y_k$ brings. The procedure is based on iteration. Let $\ell$ denote the iteration number and $\hat{x}_{k|k}^{(0)} = \hat{x}_{k|k-1}$ for $\ell = 0$. The update formulas are

$$K_k^{(\ell)} = P_{k|k-1}^{x} H_k^{(\ell)} \left[ H_k^{(\ell)} P_{k|k-1}^{x} H_k^{(\ell)\top} + R \right]^{-1}, \quad (3.27)$$

$$\hat{y}_k^{(\ell)} = h\left(\hat{x}_{k|k}^{(\ell)}, u_k\right) + H_k^{(\ell)} \left(\hat{x}_{k|k} - \hat{x}_{k|k}^{(\ell)}\right), \quad (3.28)$$

$$\hat{x}_{k|k}^{(\ell+1)} = \hat{x}_{k|k-1} + K_k^{(\ell)} \left(y_k - \hat{y}_k^{(\ell)}\right), \quad (3.29)$$

where $R > 0$ accounts for the measurement noise and

$$H_k^{(\ell)} = \frac{\partial h}{\partial x}(\hat{x}_{k|k}^{(\ell)}, u_k).$$

The iteration process stops when $\ell$ achieves the pre-specified maximum iteration number $\ell_{\text{max}}$ or when the error between two consecutive iterations is less than the pre-selected tolerance level. Then $\hat{x}_{k|k} = \hat{x}_{k|k}^{(\ell_{\text{max}})}$, and the associated error covariance is given by

$$P_{k|k}^{x} = \left[I - K_k^{(\ell_{\text{max}})} H_k^{(\ell_{\text{max}})}\right] P_{k|k-1}^{x}. \quad (3.30)$$

Following the above description, the IEKF can be applied as an elemental filter to $\mathcal{M}_1$ and $\mathcal{M}_2$. The resultant state estimates are $\hat{x}_{k|k}^{1}$ and $\hat{x}_{k|k}^{2}$, respectively. Accordingly, the SoC estimates are denoted as $\hat{\text{SoC}}_k^{1} = \hat{x}_{k|k}^{1}$ and $\hat{\text{SoC}}_k^{2} = \hat{x}_{k|k}^{2}$, respectively.

The SoC estimates produced from $\mathcal{M}_1$ and $\mathcal{M}_2$, $\hat{\text{SoC}}_k^{1}$ and $\hat{\text{SoC}}_k^{2}$, respectively, can be combined weightedly to generate the overall estimate $\hat{\text{SoC}}_k$. In the
Table 3.2: The MM-AdaSoC algorithm: Adaptive SoC estimation using multiple models.

<table>
<thead>
<tr>
<th>Algorithm Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialize</strong>: $k = 0$, $\hat{x}_{0</td>
</tr>
<tr>
<td><strong>Repeat</strong> $k \leftarrow k + 1$</td>
</tr>
<tr>
<td><strong>IEKF based adaptive SoC estimation</strong></td>
</tr>
<tr>
<td>for $i = 1$ to $2$ do</td>
</tr>
<tr>
<td>import the submodel $\mathcal{M}_i$</td>
</tr>
<tr>
<td>$\mathcal{M}_i$-based prediction (time-update):</td>
</tr>
<tr>
<td>project the state ahead to obtain $\hat{x}_{k</td>
</tr>
<tr>
<td>project the error covariance ahead to obtain $P_{k</td>
</tr>
<tr>
<td>$\mathcal{M}_i$-based update:</td>
</tr>
<tr>
<td>initialize the iteration procedure: $\ell = 0$, $\hat{x}_{k</td>
</tr>
<tr>
<td>while $\ell &lt; \ell_{\text{max}}$ do</td>
</tr>
<tr>
<td>$\ell \leftarrow \ell + 1$</td>
</tr>
<tr>
<td>compute the Kalman gain matrix via (3.27)</td>
</tr>
<tr>
<td>compute the predicted voltage via (3.28)</td>
</tr>
<tr>
<td>update the state estimate via (3.29)</td>
</tr>
<tr>
<td>end while</td>
</tr>
<tr>
<td>assign $\hat{x}_{k</td>
</tr>
<tr>
<td>update the error covariance via (3.30)</td>
</tr>
<tr>
<td>export $\mathcal{M}_i$-based SoC estimate $\hat{\text{SoC}}<em>k^i = \hat{x}</em>{k</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td><strong>Estimation fusion</strong></td>
</tr>
<tr>
<td>determine the probability $\pi_k^i$ that the battery runs on $\mathcal{M}<em>i$ for $i = 1, 2$ with $\sum</em>{i=1}^2 \pi_k^i = 1$</td>
</tr>
<tr>
<td>for $i = 1$ to $2$ do</td>
</tr>
<tr>
<td>compute the initial weights via (3.21)</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td>compute the normalized weights via (3.22)</td>
</tr>
<tr>
<td>fuse the SoC estimates from $\mathcal{M}_1$ and $\mathcal{M}_2$ via (3.31) based on (3.23)-(3.24)</td>
</tr>
<tr>
<td>until SoC estimation task ends</td>
</tr>
</tbody>
</table>

Putting together the results, we obtain the MM-AdaSoC algorithm, as is summarized in Table 3.2.

**Remark 3.3.** The underlying idea of the proposed MM-AdaSoC algorithm is that
the IEKF-based adaptive SoC estimation is carried out for multiple models and then the estimation results are fused to yield the overall SoC estimate. For the MM-AdaSoC, the recursive and real-time implementation cuts down the amount of stored data. Meanwhile, higher estimation accuracy is achieved, because the update procedure relies on iterative searching at each recursion. Another noteworthy advantage is that a good balance is maintained between the estimation performance and the computational complexity, conceding a generally linear moderate increase of the demanded computing power depending on the number of models used.

**Remark 3.4.** The applicability of the proposed MM-AdaSoC algorithm to different types of batteries is quite promising. Due to its parameterized characterization, the Nernst model has been found capable of describing the dynamics of many batteries, e.g., nickel metal hydride (NiMH), LiMn$_2$O$_4$ and LiCoO$_2$. As a result, the MM-AdaSoC algorithm can be well applied to such batteries for its construction based on the Nernst model.

**Remark 3.5.** Not limited to the MM-AdaSoC algorithm at all, the role that multi-model estimation can play is more profound. It can be developed as a framework, within which variety of advanced estimation methods can be built for battery applications. Here, we identify five potential sources of multiple models:

- a set of submodels established from a battery model by fixing certain parameters for adaptive SoC estimation, as we have done here,
- a set of submodels established from a model by assuming different sets of values for model parameters,
- a set of different models constructed in different ways, such as an equivalent-circuit model and an electrochemical-principles-based model,
- a set of models capturing different characteristics of batteries, e.g., the charging and discharging processes, cycling and aging effects, and
- a multitude of (sub)models combining the above four cases.
The multi-model approach promises three-fold benefits.

- It better apprehends the battery dynamics known to be complex and multi-faceted, thus promoting the accuracy and robustness of SoC estimation.

- It reduces the complexity of estimator design, especially when highly nonlinear battery dynamics are involved, in a ‘divide-and-conquer’ manner. Simple and elegant solutions will be achieved and theoretical analysis is made easier.

- It can even provide useful model interpretation and comparison in some circumstances. We note that research on relevant topics would be of much interest and requires further exploration.

To fully realize its potential and benefits, multi-model estimation/control for batteries needs to be further studied in the future.

3.5 Application Examples

In this section, we present two examples using simulation and experiment data, respectively, to evaluate the MM-AdaSoC algorithm.

**Example 1.** This example is based on simulation with a model used in [52] for a NiMH battery system. For simulation purpose, we employ certain minor modification, but the obtained model is still considered to be a sufficiently accurate representation of the NiMH battery dynamics in most circumstances. The change of SoC is governed by

\[
\text{SoC}_{k+1} = \text{SoC}_k - \frac{\eta \cdot \Delta T}{C_0} u_k - \frac{S_D(T_{\text{ref}}) \cdot \Delta T}{100},
\]

where the third term on the right-hand side represents self-discharge with

\[
S_D(T_{\text{ref}}) = k_0 \exp \left( -\frac{E_{A,S}}{R_g T_{\text{ref}}} \right) \text{SoC}.
\]

Here, \(k_0 = 1.0683 \times 10^7\) per hour, \(E_{A,S}/R_g = 6.789\)K, the current efficiency \(\eta = 1\) for discharging and 0.99 for charging near 50% SoC, the nominal capacity
$C_0 = 1.25 \text{Ah}$, the reference temperature $T = 35^\circ C \ (308.15 \text{K})$, and the sampling period $\Delta T = 1 \text{s}$. The initial SoC is assumed to be 50%. The terminal voltage is equal to the OCV plus internal-resistance-induced drop, that is,

$$y_k = V_{oc,k} - \bar{R} u_k.$$  

The OCV $V_{oc}$ is given by the following equation with the inclusion of voltage hysteresis:

$$V_{oc} = U_0 + \frac{R_g T_{\text{ref}}}{n_e F} \ln \left( \frac{\text{SoC}_k - \Pi}{1 - \text{SoC}_k} \right) + V_{H,k},$$

where the varying voltage hysteresis $V_{H}$ is characterized by an empirical expression

$$V_{H,k+1} = V_{H,k} - \beta \cdot \eta \cdot [V_{H,\text{max}} + \text{sign}(u_k) \cdot V_{H,k}] \cdot \Delta T \cdot u_k.$$  

The resistance $\bar{R}$ is described by

$$\bar{R} = \sum_{j=0}^{n} a_j \text{SoC}^j.$$  

The Farady’s constant $F = 96,487 \text{C mol}^{-1}$, $n_e = 1$, $R_g = 8.314 \text{J mol}^{-1} \text{K}^{-1}$, $U_0 = 1.37 \text{V}$, $\Pi = 0.08$, $\beta = 3 \times 10^{-5} \text{C}^{-1}$, $V_{H,\text{max}} = 0.05 \text{V}$. The initial $V_H$ is $0.005 \text{V}$. In addition, $a_0 = 4.1252 \times 10^{-2}$, $a_1 = 8.9691 \times 10^{-4}$, $a_2 = 1.6760 \times 10^{-5}$, $a_3 = -1.4435 \times 10^{-7}$ and $a_4 = 4.7223 \times 10^{-10}$.

During the simulation, we do not assume that this model is fully available for SoC estimation. Instead, the modified Nernst model presented in Section 3.2 will be used for approximate description of the above true model. Let $K_0 = \eta \cdot \Delta T / C_0$, $K_1 = U_0$, $K_2 = K_3 = R_g T_{\text{ref}}/(n_e F)$, $\tau_1 = -\Pi$, $\tau_2 = 0$ and $R = a_0$. The current signal applied as the input to the battery was a pseudo-random binary sequence (PRBS) stretched by 100 times over the time axis. Its magnitude is 1A. A view of the input current and output voltage during the first 2000 time instants is given in Fig. 3.5a. Let the true initial SoC be 55% and the initial SoC estimate be 65%. The initial weights assigned to $\mathcal{M}_1$ and $\mathcal{M}_2$ are 0.7 and 0.3, respectively.
Figure 3.5: (a) The input current-output voltage profile; (b) SoC estimates versus truth over time; (c) fusion weights for $M_1$ and $M_2$ versus time; (d) comparison with the EKF.
In this setting, we face hysteresis, model mismatch and incorrect initial estimate, which together make SoC estimation a tougher challenge. As described in previous sections, we consider two submodels, with the first one assuming known $K_0$ and $K_1$ and the second assuming known $K_1$, $K_2$ and $K_3$. When the MM-AdaSoC algorithm is applied, Fig. 3.5b shows the estimation of the SoC over time. We see that both $M_1$- and $M_2$-based estimates differ from the actual values with bounded errors. However, it turns out that the overall estimates given by the MM-AdaSoC algorithm become more accurate, demonstrating that the estimation errors can be reduced effectively by fusion of the multi-model estimates. The weights of the two models are compared in Fig. 3.5c. Obviously, $M_2$ weighs much more than $M_1$ in this case. This is because $M_1$ is sensitive to the initial SoC estimate, relying on the state equation based on Coulomb counting. Furthermore, we also apply the well-known EKF to the modified Nernst model with known $K_i$ for $i = 0, 1, 2, 3$ for SoC estimation. As shown in Fig. 3.5d, the EKF yields unreliable results in this situation in comparison to the MM-AdaSoC algorithm.

**Example 2.** For the experimental evaluation of the MM-AdaSoC algorithm, data was collected from a LiMn$_2$O$_4$/hard-carbon battery in the Advanced Technology R&D Center, Mitsubishi Electric Corporation. The experiment was conducted using a rechargeable battery test equipment produced by Fujitsu Telecom Networks. The current input was a PRBS signal stretched by 10 times over the time axis with a magnitude of 5A. Despite many other options, we chose the PRBS because...
Figure 3.7: (a) SoC estimates versus time; (b) SoC estimates during the initial 450s; (c) fusion weights for $M_1$ and $M_2$ versus time; (d) fusion weights during the initial 450s; (e) comparison between the true and the one-step-forward predicted voltage.
it has white-noise-like properties and is admissible for observability. The profile of
the input current and the output voltage is shown in Fig. 3.6. The battery has a
nominal capacity of 4.93Ah. The sampling period was 1s. During the experiment,
the ambient temperature in the chamber was maintained at 25.8°C.

We consider the model in (3.12). The Coulombic efficiency constant \( K_0 = 5.6342 \times 10^{-5} \) when \( u_k > 0 \) (100% for discharging) and \( K_0 = 4.7891 \times 10^{-5} \) when
\( u_k < 0 \) (85% for charging). From the SoC-OCV data collected from this type
of batteries, it can be determined that \( K_1 = 1.294, K_2 = 0.0984, K_3 = 3.972, \)
\( \tau_1 = \tau_2 = 0.3. \)

As aforementioned, the actual values of the parameters \( K_i \) for \( i = 0, \cdots, 3 \)
can change as a result of the operating conditions. Hence, rather than depending
fully on their nominal values, we perform multi-model adaptive SoC estimation by
applying the \textit{MM-AdaSoC} algorithm. The construction of two submodels from (3.12)
is described in Section 3.2.1.

The SoC estimation results are shown in Fig. 3.7. The full view over the
available experimental data is given in Fig. 3.7a. The initial SoC of the battery
is known to be approximately 50%. It is seen that there is a difference of approx-
imately 5% between the \( \mathcal{M}_1 \)-based and \( \mathcal{M}_2 \)-based estimates. Based on our
experience, \( \mathcal{M}_1 \) tends to yield conservative estimates in this case and \( \mathcal{M}_2 \) does the
opposite. The \textit{MM-AdaSoC} algorithm, through the fusion strategy, makes adjust-
ment to give neutralized overall estimates. Although the true SoC data are not
available, we still judge that the estimates are close to the truth, based on our \textit{a priori}
knowledge about the battery behavior. Fig. 3.7b illustrates what happens
during the initial 450s. It is seen from Figs. 3.7a-3.7b that the overall estimates
are closer to those based on \( \mathcal{M}_2 \). This is verified in Figs. 3.7c-3.7d, where the
weight \( \mu_1 \) for \( \mathcal{M}_1 \) fluctuates slightly around 0.63 and \( \mu_2 \) around 0.37. Thus, with a
larger weight, \( \mathcal{M}_1 \) is given more confidence than \( \mathcal{M}_2 \) by the \textit{MM-AdaSoC} algorithm
during the implementation. It is understood that the fusion depends on the perfor-
mance of one-step-forward prediction of the terminal voltage. Fig. 3.7e compares
the measured data with the prediction based on \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \), respectively. The
prediction is satisfactory for both submodels, but \( \mathcal{M}_2 \) is observed to lead to the
better predicted voltage.

From the above results, we believe that the MM-AdaSoC algorithm is quite effective, supported by the findings that the obtained SoC estimates exhibit considerable accuracy and that the voltage prediction approximates the truth well. Through experiments with charging/discharging rates of 0.5A, 1A, 10A and 15A, we consistently observe similar estimation results, which shows that the applicability of the model and the power of the MM-AdaSoC algorithm.

3.6 Conclusion

Development of adaptive approaches for SoC estimation is of practical significance, because battery dynamics are often hard to fully determine and are time-varying. In this chapter, we focused on improving the adaptive SoC estimation via launching a multi-model strategy in, motivated by the proven success of multi-model estimation in addressing problems involving structural and parameter changes.

Rather than following the trilogy of ‘modeling—identification—SoC estimation’, we integrated SoC estimation with parameter identification here. The main contribution was the development and validation of the MM-AdaSoC algorithm. It was built to estimate a battery’s SoC in real time through carrying out simultaneous state and parameter estimation on a set of (sub)models. A reduced-complexity model in general state-space form was derived from the single particle model. We first constructed two submodels by fixing different parameters, with both shown to be locally observable with admissible inputs. The well-known IEKF was then applied to each submodel to produce the SoC and parameter estimates. The final overall estimates were generated by fusing the submodel-based estimates, and it was shown that the fusion is a linear weighted combination of the estimates. Simulation and experimental results were presented to demonstrate and validate the effectiveness of the algorithm.

Apart from the MM-AdaSoC algorithm, we also emphasized the potential of the multi-model framework for battery applications. The initial success reported
here would provide strong incentives for further development of a wide range of multi-model methods to better monitor the status and health of a battery.

This chapter is based on the following published papers:


Part II

Simultaneous Input and State Estimation
Chapter 4

Iterated Nonlinear Simultaneous Input and State Filtering

4.1 Introduction

Simultaneous input and state estimation (SISE) has many applications in areas such as disturbance rejection [69], maneuvering target tracking[70], fault detection [71, 72], weather forecasting [73] and oceanography [74]. A continued research interest in this topic has been observed during the past decades. This chapter deals with the problem of simultaneous input and state filtering (SISF). We consider a nonlinear dynamic systems where both the state describing the system behavior and the input that excites the system are unknown. The only available measurement information is the output. The SISF problem then consists of obtaining estimates of both the input and state variables sequentially and in real time as output measurements become available.

Early work in [75] studies state estimation with unknown inputs which are modeled by stochastic processes with known wide-sense description (e.g., mean and covariance). Minimum variance unbiased estimation (MVUE) is employed [73] to deal with the scenario of completely unknown inputs. Based on [73], recent MVUE-based works [76, 77, 78] establish the conditions for the existence of unbiased estimators and the stability of the filters developed therein. In the above mentioned
works, only the state estimation is conducted, leaving the unknown input estimation untackled. As input information is often as important as state information, SISE puts the emphasis on estimating both of them concurrently. A majority of works consider linear discrete-time dynamical systems. An early contribution in this respect is about developing a KF-based approach to estimate the state and external white process noise of a linear discrete-time system [79], with the assumption of known noise covariance. Most current works treat the case of completely unknown inputs and build on existing state estimation techniques. Among them, we highlight KF [80, 81], moving horizon estimation (MHE) [82], $H_{\infty}$-filtering [83], sliding mode observers [84, 85], and MVUE [86, 87]. The MVUE-based filters in [87, 86] are optimal among all linear unbiased state and input estimators in the sense of minimum mean square error. However, unlike the KF, they have not been proven to be derivable using Bayesian analysis, though the state and output signals of the system are stochastic processes indeed.

SISF for nonlinear systems is a difficult challenge. As is well known, the EKF for nonlinear systems proceeds by linearizing about the current state estimate and using the KF on the resulting system. Our contribution here is partially motivated by the fact that the idea of simple linearization in developing the EKF from the KF is not truly useful in developing nonlinear SISF, because unknown inputs make it futile to linearize nonlinear functions where the state and input are coupled. Among the few results in the literature on nonlinear SISF, we highlight [88, 89] on a special class of nonlinear systems that consists of a nominally linear part and a nonlinear part: the former work regards the nonlinear term as unknown state-dependent input to estimate, while the latter considers a Lipschitz nonlinear function with respect to both state and input.

The main contribution of this chapter is the development of a Bayesian framework dealing with simultaneous input and state estimation of nonlinear systems. Our treatment extends the Bayesian approach employed in other works for state and parameter estimation to jointly estimate the input and state. We employ the Bayesian paradigm to provide a suitable framework to incorporate the information contained in the data into the probability densities of the input, state, and
measurement variables. We define input and state estimates as solutions to a MAP-based nonlinear optimization problem formulated with Gaussianity assumptions. As the nonlinearity hinders the analytical calculation of the estimates, we use a Gauss-Newton method for approximate numerical optimization [14, 90, 91]. The obtained solutions for systems with and without direct input-output feedthrough are named ISISF-wDF and ISISF-w/oDF, respectively. We show that the linear versions of ISISF-wDF and ISISF-w/oDF, named LSISF-wDF and LSISF-w/oDF, respectively, are equivalent to those presented in [87, 86] and also generalize other works on state estimation with unknown inputs [73, 76, 77, 80]. Furthermore, we analyze the asymptotic stability of LSISF-wDF and LSISF-w/oDF and find the sufficient conditions for the stability, with inspiration from the Kalman filter stability analysis.

4.2 SISF for Nonlinear Systems with Direct Feedthrough

We consider nonlinear systems with direct feedthrough of the form

\[
\begin{align*}
\mathbf{x}_{k+1} &= f(x_k, u_k) + w_k, \\
y_k &= h(x_k, u_k) + v_k,
\end{align*}
\]

where \( u \in \mathbb{R}^{n_u} \) is the input vector, \( x \in \mathbb{R}^{n_x} \) is the state vector, \( y \in \mathbb{R}^{n_y} \) is measurement vector, and \( w \in \mathbb{R}^{n_x} \) and \( v \in \mathbb{R}^{n_y} \) are mutually independent zero-mean white Gaussian noise sequences, with covariances \( Q_k \) and \( R_k \), respectively. The mappings \( f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x} \) and \( h : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_y} \) define the state transition and measurement functions, respectively, which is assumed to be \( C^1 \). We also assume \( \nabla u h \) has full rank. For the above system, our objective is to design a NL-SISE filter to estimate \( u_k \) and \( x_k \) from the measurements \( y_1, y_2, \cdots, y_k \).
4.2.1 Bayesian Paradigm

As aforementioned, Bayesian analysis have historically provided a framework for developing the estimation schemes such as the classical KF and particle filters [92]. A Bayesian estimator proceeds by estimating the probability density functions (pdf’s) of unknown variables conditioned on available measurements. Denote $\mathbb{Y}_k$ as the set of \{\(y_1, y_2, \ldots, y_k\}\}, containing all measurement information collected until time instant \(k\). For the considered problem, both the input \(u_k\) and state \(x_k\) are to be estimated from $\mathbb{Y}_k$. In fact, they can be estimated once the joint conditional pdf $p(u_k, x_k|\mathbb{Y}_k)$ is obtained for each time instant \(k\). The goal in this section is to sequentially compute $p(u_k, x_k|\mathbb{Y}_k)$ from $p(u_{k-1}, x_{k-1}|\mathbb{Y}_{k-1})$. Like in Bayesian state estimation, this can also be accomplished in a two-step procedure of prediction and update. Prediction is to determine $p(x_k|\mathbb{Y}_{k-1})$ – the \textit{a priori} conditional pdf of $x_k$ given measurements up until time instant $k-1$. At time instant $k$, the measurement $y_k$ can be used to update $p(x_k|\mathbb{Y}_{k-1})$ and, at the same time, to jointly estimate the conditional pdf of $u_k$ (since it is the first measurement containing information about $u_k$) via the \textit{a posteriori} conditional pdf $p(u_k, x_k|\mathbb{Y}_k)$. Let us denote $\xi_k = [u_k^\top x_k^\top]^\top$ for notational simplicity.

To proceed, we introduce the following assumption:

**Assumption 4.1.** \{\(u_k\)\} is a \textit{white} process, independent of \(x_0\), \{\(w_k\)\} and \{\(v_k\)\}.

Here, ‘white’ means that $u_k$ and $u_l$ are independent random variables for $k \neq l$. Completely unknown to us, \{\(u_k\)\} may assume all possible values, suggesting that it can be treated as a stochastic process [75]. However, what is different from [75] is that no additional assumption is imposed to require known wide-sense description of \{\(u_k\)\}. From (A1), it can be easily seen that $u_k$ is independent of $x_k$ and $\mathbb{Y}_{k-1}$ [93].

The following theorem illustrates the Bayesian paradigm that governs the probability-based SISF.

**Theorem 4.1.** Consider the system in (4.1) and suppose Assumption 4.1 holds.
The Bayesian paradigm for SISF is given by

\[ p(x_k|Y_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|Y_{k-1})d\xi_{k-1}, \]  

(4.2)

\[ p(\xi_k|Y_k) = \frac{p(y_k|\xi_k)p(x_k|Y_{k-1})p(u_k)}{p(y_k|Y_{k-1})}. \]  

(4.3)

Proof. It follows from the Chapman-Kolmogorov equation [31] that

\[ p(x_k|Y_{k-1}) = \int \int p(x_k|u_{k-1}, x_{k-1}, Y_{k-1})p(u_{k-1}, x_{k-1}|Y_{k-1})du_{k-1}dx_{k-1}. \]

It is noted that

\[ p(x_k|u_{k-1}, x_{k-1}, Y_{k-1}) = p(x_k|u_{k-1}, x_{k-1}), \]

since \( x_k \) depends on \( u_{k-1} \) and \( x_{k-1} \), as the state equation in (4.1) is Markovian with order one. Hence (4.2) follows and \( p(x_k|u_{k-1}, x_{k-1}) \) relies on (4.1). Then using the Bayes’ theorem repeatedly, we obtain

\[ p(u_k, x_k|Y_k) = \frac{p(u_k, x_k, Y_k)}{p(Y_k)} \]

\[ = \frac{p(u_k, x_k, y_k, Y_{k-1})}{p(y_k, Y_{k-1})} \]

\[ = \frac{p(y_k|u_k, x_k, Y_{k-1})p(u_k, x_k|Y_{k-1})p(Y_{k-1})}{p(y_k|Y_{k-1})p(Y_{k-1})} \]

\[ = \frac{p(y_k|u_k, x_k, Y_{k-1})p(u_k, x_k|Y_{k-1})}{p(y_k|Y_{k-1})}. \]

Note that

\[ p(y_k|u_k, x_k, Y_{k-1}) = p(y_k|u_k, x_k), \]  

(4.4)

\[ p(u_k, x_k|Y_{k-1}) = p(x_k|Y_{k-1})p(u_k). \]  

(4.5)

Here, (4.4) is due to the fact that \( y_k \) entirely depends on \( u_k \) and \( x_k \), and (4.5)
results from $u_k$’s independence from $x_k$ and $Y_{k-1}$. Consequently,

$$p(u_k, x_k | Y_k) = \frac{p(y_k | u_k, x_k) p(x_k | Y_{k-1}) p(u_k)}{p(y_k | Y_{k-1})},$$

which is equivalent to (4.3).

**Remark 4.1.** In Theorem 4.1, $p(u_k)$ is noteworthy because $u_k$ is considered completely unknown. Such a pdf is called noninformative prior in statistics because no \textit{a priori} information is available. It has long been a controversial topic how to deal with a noninformative prior [94, 95, 96, 97, 98]. One popular opinion, supported by Thomas Bayes and Pierre-Simon de Laplace even 200 years ago, is to let it be a uniform distribution, or in other words, let all possible outcomes have equal probability. This is known as the \textit{principle of indifference} or \textit{principle of insufficient reason}. However, it is opposed by many other statisticians such as Sir Ronald Aylmer Fisher and John Maynard Keynes, who pointed out that the principle can easily lead to contradictions. A few other methods have been suggested to construct noninformative priors in the literature, see [94] for a survey. However, no consensus has been reached among statisticians as to which distribution best describes noninformative priors. When no objective or unique noninformative prior exists for $u_k$, we can choose to treat $p(u_k)$ as an ignorable coefficient, which should be a tenable means to minimize its influence on the estimation.

Sequentially updating (4.2)-(4.3) not only provides the Bayesian solution to SISF in the presence of direct feedthrough, but also yields a statistical conceptual framework or methodology which can be translated into different methods. Our next step is to derive some explicit SISF algorithms using (4.2)-(4.3).

### 4.2.2 SISF Algorithm Development

This section develops an SISF algorithm for nonlinear systems with direct feedthrough. We begin with some multivariate Gaussian distribution assumptions and then proceed to the algorithm derivation on the basis of the Bayesian paradigm (4.2)-(4.3).

In order to continue the investigation, we make the following assumptions:
Assumption 4.2. $p(\xi_k|Y_k)$ is a Gaussian distribution with mean $\hat{\xi}_{k|k}$ and covariance $P^\xi_{k|k}$.

Assumption 4.3. $p(y_k|\xi_k)$ is a Gaussian distribution with mean $h(\xi_k)$ and $R_k$.

Assumption 4.4. $p(x_k|Y_{k-1})$ is a Gaussian distribution with mean $\hat{x}_{k|k-1}$ and $P^x_{k|k-1}$.

In above, $\hat{\xi}_{k|k}$ is the estimate of $\xi_k$ given $Y_k$, $\hat{x}_{k|k-1}$ is the estimate of $x_k$ given $Y_{k-1}$, and $P^\xi_k$ and $P^x_{k|k-1}$ are error covariances.

Ideally, if knowledge of $p(\xi_k|Y_k)$ and $p(x_k|Y_{k-1})$ is available, the estimation can be simply obtained by MAP or conditional mean calculation. However, it is an almost impossible task to compute the pdf’s accurately for nonlinear systems. In order to overcome the problem, Assumptions 4.2-4.4 are made to approximately describe the pdf’s, thus paving the way for development of a sequentially updating nonlinear SISF algorithm from the Bayesian paradigm. Gaussian distribution assumptions analogous to them are usual in estimation algorithms for nonlinear systems, e.g., [14, 90, 30].

To develop the state prediction procedure, let us use (4.2). Define

$$\hat{x}_{k|k-1} = \arg \max_{x_k} p(x_k|Y_{k-1}). \tag{4.6}$$

According to (4.2), the above maximization requires knowledge of $p(x_k|\xi_{k-1})$ and $p(\xi_{k-1}|Y_{k-1})$. We see that Assumption 4.2 indicates

$$p(\xi_{k-1}|Y_{k-1}) = \mathcal{G}(\xi_{k-1}; \hat{\xi}_{k-1|k-1}, P^\xi_{k-1|k-1}) \tag{4.7}$$

Furthermore, we have

$$p(x_k|\xi_{k-1}) \approx \mathcal{G} \left( x_k; f(\hat{\xi}_{k-1|k-1}), W_{k-1} \right), \tag{4.8}$$

where

$$W_{k-1} = \nabla_\xi f(\hat{\xi}_{k-1|k-1}) P^\xi_{k-1|k-1} \nabla_\xi f^T(\hat{\xi}_{k-1|k-1}) + Q_{k-1}.$$
Note that $\nabla_\xi f = [\nabla_u f \ \nabla_x f]$. The result in (4.8) follows from the assumption Assumption 4.2 and the first-order Taylor series expansion of $f(\xi_{k-1})$ around $(\hat{\xi}_{k-1|k-1})$, that is,

$$f(\xi_{k-1}) \approx f(\hat{\xi}_{k-1|k-1}) + \nabla_\xi f(\hat{\xi}_{k-1|k-1}) (\xi_{k-1} - \hat{\xi}_{k-1|k-1}).$$

Substituting the Gaussian pdf’s of (4.7) and (4.8) into (4.2), we obtain

$$p(x_k|y_{k-1}) = \mathcal{G}(x_k; f(\hat{\xi}_{k-1|k-1}), W_{k-1}).$$

Therefore, the solution to (4.6) is

$$\hat{x}_{k|k-1} = f(\hat{\xi}_{k-1|k-1}), \quad (4.9)$$

with associated prediction error covariance $P_{x_{k|k-1}}$ given by

$$P_{x_{k|k-1}} = W_{k-1}. \quad (4.10)$$

Thus (4.9) and (4.10) constitute the prediction formulae together, computing the predicted values of the state and error covariance, respectively.

When the new measurement $y_k$ arrives, $\hat{x}_{k|k-1}$ can be updated to $\hat{x}_{k|k}$ that has better accuracy, and $\hat{u}_{k|k}$ can be estimated simultaneously. Define

$$\hat{\xi}_{k|k} = \arg \max_{\xi_{k|k}, \hat{x}_k} p(\xi_k|y_k). \quad (4.11)$$

The approach used in [14, 90] can be modified to address the above maximization problem to obtain $\hat{\xi}_{k|k}$, by applying the Gauss-Newton method to approximate the MAP estimates.

Referring to (4.3), $p(\xi_k|y_k)$ is proportional to the multiplication of $p(y_k|\xi_k)$ and $p(x_k|y_{k-1})$. We note that $p(y_k|\xi_k)$ and $p(x_k|y_{k-1})$ are shown in Assumptions 4.2-4.3, respectively. Define $p(\xi_k|y_k)$ as the MAP cost function $(\xi_k)$, i.e.,

$$\mathcal{L}(\xi_k) := p(\xi_k|y_k) = \lambda \cdot \exp \left[ -\alpha_k^T R_k^{-1} \alpha_k - \beta_k^T (P_{x_{k|k-1}})^{-1} \beta_k \right],$$

$$\xi_k := \arg \max_{\xi_k} \mathcal{L}(\xi_k).$$
where \( \lambda \) combines all the constants, and

\[
\alpha_k = y_k - h(\xi_k), \quad \beta_k = x_k - \hat{x}_{k|k-1}.
\]

It is easier to deal with the logarithmic cost function:

\[
L(\xi_k) = -\ln L(\xi_k) = \delta + \boldsymbol{r}^\top(\xi_k)\boldsymbol{r}(\xi_k),
\]

where \( \delta = -\ln \lambda \) and

\[
\boldsymbol{r}(\xi_k) = \begin{bmatrix} R_k^{-\frac{1}{2}} \alpha_k \\ (P_{x|k-1})^{-\frac{1}{2}} \beta_k \end{bmatrix}.
\]

Then by (4.11), the joint input and state estimate \( \hat{\xi}_{k|k} \) can be equivalently defined as

\[
\hat{\xi}_{k|k} = \arg \min_{u_k, x_k} L(\xi_k).
\]

The MAP estimation problem in (4.13) requires nonlinear optimization, which makes the derivation of an analytical solution difficult. However, indeed being a nonlinear least-squares problem, it can be numerically solved using the Gauss-Newton method [99].

To find \( \hat{\xi}_{k|k} \), the classical Gauss-Newton method iteratively computes the sequences of approximations \( \hat{\xi}_{k|k}^{(\ell)} \), where \( \ell \) denotes the iteration index. Specifically,

\[
\hat{\xi}_{k|k}^{(\ell+1)} = \hat{\xi}_{k|k}^{(\ell)} - \left[ \nabla_{\xi}^\top \boldsymbol{r} \left( \hat{\xi}_{k|k}^{(\ell)} \right) \nabla_{\xi} \boldsymbol{r} \left( \hat{\xi}_{k|k}^{(\ell)} \right) \right]^{-1} \cdot \nabla_{\xi}^\top \boldsymbol{r} \left( \hat{\xi}_{k|k}^{(\ell)} \right) \cdot \boldsymbol{r} \left( \hat{\xi}_{k|k}^{(\ell)} \right),
\]

where

\[
\nabla_{\xi} \boldsymbol{r} = \begin{bmatrix} \nabla_u \boldsymbol{r} & \nabla_x \boldsymbol{r} \end{bmatrix}.
\]
We may let the initial guess be $\hat{\xi}^{(0)}_{k|k} = [0^\top \hat{x}_{k|k-1}^\top]^\top$ for convenience, though it can be set to arbitrary values. The iteration process continues until the iteration step $(\ell)$ reaches the preselected maximum $\ell_{\text{max}}$ or the difference between two consecutive iterations is less than a preselected value, with $\hat{\xi}^{(\ell)}_{k|k}$ obtained in the final iteration assigned to $\hat{\xi}_{k|k}$, respectively.

The iteration process in (4.14) refines the input and state estimates continually by re-evaluating the joint estimator around the latest estimated input and state operating point. Despite demanding more computational power, the iteration-based refinement enhances not only the estimation performance but also the robustness to nonlinearities.

The error covariance associated with (4.14) is equal to the inverse of the Fisher information matrix, as is common in MAP estimators under Gaussian distributions [100]. Then we have

$$\mathcal{P}_{k|k} = \begin{bmatrix} P^{u}_{k|k} & P^{ux}_{k|k} \\ (P^{ux}_{k|k})^\top & P^{x}_{k|k} \end{bmatrix} = \mathcal{I}^{-1}(\hat{u}_{k|k}, \hat{x}_{k|k}),$$  

where the Fisher information matrix $\mathcal{I}$ is defined as

$$\mathcal{I} = \begin{bmatrix} \mathcal{I}^{u} & \mathcal{I}^{ux} \\ (\mathcal{I}^{ux})^\top & \mathcal{I}^{x} \end{bmatrix} = E \left[ \begin{bmatrix} \nabla_{u}^{\top}L \\ \nabla_{x}^{\top}L \end{bmatrix} \begin{bmatrix} \nabla_{u}L \\ \nabla_{x}L \end{bmatrix} \right].$$  

The explicit formulae for the gradients are as follows:

$$\nabla_{u}r = \begin{bmatrix} -R^{-\frac{1}{2}} \nabla_{u}h \\ 0 \end{bmatrix},$$

$$\nabla_{x}r = \begin{bmatrix} -R^{-\frac{1}{2}} \nabla_{x}h \\ (P^{x}_{k|k-1})^{-\frac{1}{2}} \end{bmatrix},$$

$$\nabla_{u}L = r^\top \nabla_{u}r = \alpha^\top R^{-1} \nabla_{u}h,$$

$$\nabla_{x}L = r^\top \nabla_{x}r = \alpha^\top R^{-1} \nabla_{x}h + \beta^\top (P^{x}_{k|k-1})^{-1}. $$
Hence, $\mathcal{I}$ is given by

$$
\mathcal{I} = \begin{bmatrix}
\nabla^\top_u h R_k^{-1} \nabla_u h & \nabla^\top_u h R_k^{-1} \nabla_x h \\
\nabla^\top_x h R_k^{-1} \nabla_u h & \nabla^\top_x h R_k^{-1} \nabla_x h + (P_{x|k-1}^x)^{-1}
\end{bmatrix}.
$$

(4.17)

The updated error covariance in (4.15) can then be computed by inserting $\hat{u}_{k|k}$ and $\hat{x}_{k|k}$ into $\mathcal{I}$.

**Remark 4.2.** While the basic Gauss-Newton iteration shown in (4.14) solves linear problems within only a single iteration and has fast local convergence on mildly nonlinear problems, it may suffer from divergence for some nonlinear problems. To improve the convergence performance, a damping coefficient $\alpha^{(\ell)} > 0$ can be added:

$$
\hat{\xi}^{(\ell+1)}_{k|k} = \hat{\xi}^{(\ell)}_{k|k} - \alpha^{(\ell)} \left[ \nabla^\top_\xi r \left( \hat{\xi}^{(\ell)}_{k|k} \right) \nabla_\xi r \left( \hat{\xi}^{(\ell)}_{k|k} \right) \right]^{-1} \cdot \nabla^\top_\xi r \left( \hat{\xi}^{(\ell)}_{k|k} \right) \cdot r \left( \hat{\xi}^{(\ell)}_{k|k} \right).
$$

(4.18)

It can be proven that the damped Gauss-Newton iteration keeps moving to the critical point in a descent direction for sufficiently small $\alpha^{(\ell)} > 0$, thus guaranteeing its local convergence. In fact, it is usually globally convergent. Yet $\alpha^{(\ell)}$ must be selected with caution to ensure the viability of the damped Gauss-Newton, and a few methods have been proposed, e.g., the Armijo-Goldstein step length principle. A further improvement is to introduce a stabilizing term:

$$
\hat{\xi}^{(\ell+1)}_{k|k} = \hat{\xi}^{(\ell)}_{k|k} - \alpha^{(\ell)} \left[ \nabla^\top_\xi r \left( \hat{\xi}^{(\ell)}_{k|k} \right) \nabla_\xi r \left( \hat{\xi}^{(\ell)}_{k|k} \right) + \delta^{(\ell)} D^{(\ell)} \right]^{-1} \cdot \nabla^\top_\xi r \left( \hat{\xi}^{(\ell)}_{k|k} \right) \cdot r \left( \hat{\xi}^{(\ell)}_{k|k} \right),
$$

whereby the rank deficiency problem of $\left( \nabla^\top_\xi r \nabla_\xi r \right)$ that may appear in (4.14) and (4.18) can be avoided, given that $\delta^{(\ell)} > 0$ and $D^{(\ell)}$ is a specified SPD matrix. This is known as the trust region method or Levenberg-Marquardt method. For more details about Gauss-Newton-type methods, the reader is referred to [99]. Similarly, to avoid rank deficiency in computation of the estimation error covariance, $\epsilon I$ may be added to $\mathcal{I}$, where $\epsilon > 0$ is a very small value.

Putting the above equations together yields the **ISISF-wDF** algorithm, formally described in Table 4.1, for nonlinear systems of the form (4.1). The algorithm
includes the prediction (4.9)-(4.10) and update (4.14)-(4.17) procedures. While the procedures are sequential, the update is implemented iteratively.

Table 4.1: The ISISF-wDF algorithm: Iterated SISF for nonlinear systems with direct feedthrough.

| Initialize: $k = 0$, $\hat{\xi}_{0|0} = \mathbb{E}(\xi_0)$, $P_{0}^\xi = p_0 I$, where $p_0 > 0$, typically a large positive value |
|---|
| Repeat |
| $k \leftarrow k + 1$ |
| Prediction: |
| project the state ahead via (4.9) |
| compute the prediction error covariance via (4.10) |
| Update: |
| Initialize: $\ell = 0$, $\hat{\xi}_{k|k}^{(0)} = [0^T \hat{x}_{k|k-1}^T]^T$ |
| While $\ell < \ell_{\text{max}}$ do |
| perform Gauss-Newton based joint input and state estimation via (4.14) |
| $\ell \leftarrow \ell + 1$ |
| End while |
| Export $\hat{u}_{k|k}$ and $\hat{x}_{k|k}$ from $\hat{\xi}_{k|k} = \hat{\xi}_{k|k}^{(\ell_{\text{max}})}$ |
| Compute joint estimation error covariance via (4.15)-(4.17) |
| Until no more measurements arrive |

The ISISF-wDF introduces a novel Bayesian perspective to the SISF problems, since most current works consider the problem from the viewpoint of filter design and optimal gain selection. Another advantage is that it can be applied to nonlinear systems in general form, instead of being restricted to some special ones.

### 4.3 SISF for Nonlinear Systems without Direct Feedthrough

Consider a nonlinear system described by equations of the form

\[
\begin{align*}
  x_{k+1} &= f(x_k, u_k) + w_k, \\
  y_k &= h(x_k) + v_k,
\end{align*}
\]  

(4.19)
where no direct feedthrough exists. In this situation, the input estimation is delayed by one time unit, considering that the first measurement containing information about $u_{k-1}$ is $y_k$. Therefore, it is needed to sequentially update the conditional pdf $p(u_{k-1}, x_k | Y_k)$. We impose the same assumption as Assumption 4.1 to $u_k$ for the system in (4.19), i.e., $\{u_k\}$ is a white process independent of $x_0$, $\{w_k\}$ and $\{v_k\}$. Denote $\sigma_k = [u_{k-1}^T, x_k^T]^T$. The Bayesian paradigm then is shown in the following theorem.

**Theorem 4.2.** Consider the system in (4.19) and suppose $\{u_k\}$ is a white process independent of $x_0$, $\{w_k\}$ and $\{v_k\}$. The Bayesian paradigm for SISE is

$$p(\sigma_k | Y_k) \propto p(y_k | x_k) \int p(x_k | u_{k-1}, x_{k-1}) p(x_{k-1} | Y_{k-1}) dx_{k-1},$$

(4.20)

**Proof.** Using the Bayes’ theorem, we obtain

$$p(u_{k-1}, x_k | Y_k) = \frac{p(u_{k-1})}{p(y_k | Y_{k-1})} \int p(y_k | x_k) p(x_k | u_{k-1}, x_{k-1})$$

$$\cdot p(u_{k-2}, x_{k-1} | Y_{k-1}) du_{k-2} dx_{k-1}$$

$$= \frac{p(u_{k-1})}{p(y_k | Y_{k-1})} p(y_k | x_k) \int p(x_k | u_{k-1}, x_{k-1}) p(x_{k-1} | Y_{k-1}) dx_{k-1}$$

$$\propto p(y_k | x_k) \int p(x_k | u_{k-1}, x_{k-1}) p(x_{k-1} | Y_{k-1}) dx_{k-1},$$

which constructs the Bayesian paradigm for (4.19).

Based on the state equation of (4.19), estimation of $x_k$ requires information of $x_{k-1}$ and $u_{k-1}$. Yet the earliest measurement set containing information about $u_{k-1}$ is $Y_k$ instead of $Y_{k-1}$, implying that the prediction step of determining $p(x_k | Y_{k-1})$ is impossible to achieve. This explains why the Bayesian paradigm in this case is single-step as shown in (4.20).

As an analogy to Assumptions 4.2-4.4, we suppose

$$p(\sigma_k | Y_k) = \mathcal{G} (\sigma_k; \bar{\sigma}_k | k, P^\sigma_{k | k}),$$

(4.21)

$$p(y_k | x_k) = \mathcal{G} (y_k; h(x_k), R_k),$$

(4.22)
where $\hat{\sigma}_{k|k}$ is the estimate of $\sigma_k$ given $Y_k$ and $P_{\sigma|k}$ is the associated estimation covariance. From (4.20), a MAP cost function can be defined as done previously in (4.12), that is,

$$L(\sigma_k) = \delta + r^\top(\sigma_k)r(\sigma_k).$$

(4.23)

Here, $\delta = \ln \lambda$ and

$$r(u_{k-1}, x_k) = \begin{bmatrix} R_k^{-\frac{1}{2}} \rho_k \\ \Pi^{-\frac{1}{2}}(u_{k-1}) \zeta_k \end{bmatrix},$$

$$\rho_k = y_k - h(x_k),$$

$$\zeta_k = x_k - f(u_{k-1}, \hat{x}_{k-1|k-1}),$$

$$\Pi(u_k) = \nabla_x f(u_k, \hat{x}_k) P_{x|k} \nabla_x f(u_k, \hat{x}_k) + Q_k.$$ 

Using the Gauss-Newton method to compute the estimates $\hat{\sigma}_{k|k}$,

$$\hat{\sigma}_{k|k}^{(\ell+1)} = \hat{\sigma}_{k|k}^{(\ell)} - \left[ \nabla_\sigma r \left( \hat{\sigma}_{k|k}^{(\ell)} \right) \right]^{-1} \cdot \nabla_\sigma r \left( \hat{\sigma}_{k|k}^{(\ell)} \right) \cdot r \left( \hat{\sigma}_{k|k}^{(\ell)} \right),$$

(4.24)

where

$$\nabla_\sigma r = \begin{bmatrix} \nabla_u r \\ \nabla_x r \end{bmatrix}.$$ 

An optional initial condition is given by $\hat{\sigma}_{k|k}^{(0)} = 0$, and the finally obtained $\hat{\sigma}_{k|k}^{(\ell_{\text{max}})}$ will then be assigned to $\hat{u}_{k-1|k}$ and $\hat{x}_{k|k}$. A small damping coefficient and a stabilizing term can also be added to overcome possible divergence and matrix singularity problems. The associated estimation error covariance matrix can be computed by evaluating the Fisher information matrix at $\hat{u}_{k-1|k}$ and $\hat{x}_{k|k}$, i.e.,

$$P_{\sigma|k} = \begin{bmatrix} P_{u|k-1|k}^{u} & P_{ux|k-1|k}^{u} \\ (P_{ux|k-1|k}^{u})^\top & P_{x|k|k}^{\sigma} \end{bmatrix} = I^{-1}(\hat{u}_{k-1|k}, \hat{x}_{k|k}),$$

(4.25)

where the definition of $I$ is given in (4.16).

The $l$-th column of the gradient matrix of $r$ with respect to (w.r.t.) $u$ is
given by,
\[ \frac{\partial r}{\partial u_l} = \begin{bmatrix} -\Pi^{-\frac{1}{2}} \frac{\partial f}{\partial u_l} - \frac{1}{2} \Pi^{\frac{1}{2}} \Pi^{-1} \frac{\partial \Pi}{\partial u_l} \Pi^{-1} \Pi^{-1} \zeta \end{bmatrix}. \]

The following relation is used here:
\[ \frac{\partial X^\frac{1}{2}}{\partial \tau} = -\frac{1}{2} X^\frac{1}{2} X^{-1} \frac{\partial X}{\partial \tau} X^{-1}, \]
where \( X \) is a SPD matrix dependent on \( \tau \) [90]. The \( l \)-th column of the gradient matrix of \( r \) w.r.t. \( x \) is
\[ \frac{\partial r}{\partial x_l} = \begin{bmatrix} -R^{-\frac{1}{2}} \frac{\partial h}{\partial x_l} \\ \Pi^{-\frac{1}{2}} e_l \end{bmatrix} \]
where \( e_l = \begin{bmatrix} 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}^T \). This is equivalent to
\[ \nabla_x r = \begin{bmatrix} -R^{-\frac{1}{2}} \nabla_x h \\ \Pi^{-\frac{1}{2}} \end{bmatrix}. \]

The \( lj \)-th entries of \( \nabla^T u r \nabla u r \), \( \nabla^T u r \nabla x r \) and \( \nabla^T x r \nabla x r \) are expressed as, respectively,
\[ \frac{\partial r^T \partial r}{\partial u_l \partial u_j} = \frac{\partial f^T}{\partial u_l} \Pi^{-1} \frac{\partial f}{\partial u_j} + \frac{1}{2} \zeta^T \Pi^{-1} \left( \frac{\partial \Pi}{\partial u_l} \Pi^{-1} \frac{\partial f}{\partial u_j} + \frac{\partial \Pi}{\partial u_j} \Pi^{-1} \frac{\partial f}{\partial u_l} \right) + \frac{1}{4} \zeta^T \Pi^{-1} \frac{\partial \Pi}{\partial u_l} \Pi^{-1} \frac{\partial \Pi}{\partial u_j} \Pi^{-1} \zeta, \]
\[ \frac{\partial r^T \partial r}{\partial u_l \partial x_j} = -\frac{\partial f^T}{\partial u_l} \Pi^{-1} e_j - \frac{1}{2} \zeta^T \Pi^{-1} \frac{\partial \Pi}{\partial u_l} \Pi^{-1} e_j, \]
\[ \frac{\partial r^T \partial r}{\partial x_l \partial x_j} = \frac{\partial h^T}{\partial x_l} R^{-1} \frac{\partial h}{\partial x_j} + e_l^T \Pi^{-1} e_j, \]
\[ \nabla^T x r \nabla x r = \nabla^T x h R^{-1} \nabla x h + \Pi^{-1}. \]
Then we have
\[
\frac{\partial L}{\partial u_l} = r^\top \frac{\partial r}{\partial u_l} = -\zeta^{\top} \Pi^{-1} \frac{\partial f}{\partial u_l} - \frac{1}{2} \zeta^{\top} \Pi^{-1} \frac{\partial \Pi}{\partial u_l} \Pi^{-1} \zeta,
\]
\[
\frac{\partial L}{\partial x_l} = r^\top \frac{\partial r}{\partial x_l} = -\rho^{\top} R^{-1} \frac{\partial h}{\partial x_l} + \zeta^{\top} \Pi^{-1} e_l,
\]
\[
\nabla_x L = -\rho^{\top} R^{-1} \nabla_x h + \zeta^{\top} \Pi^{-1}.
\]

To compute the Fisher information matrix, we need \( E(\nabla_u^\top L \nabla_u L) \), \( E(\nabla_u^\top L \nabla_x L) \) and \( E(\nabla_x^\top L \nabla_x L) \). Their \( lj \)-th entries are
\[
E\left( \frac{\partial L^\top}{\partial u_l} \frac{\partial L}{\partial u_j} \right) = \frac{\partial f^\top}{\partial u_l} \Pi^{-1} \frac{\partial f}{\partial u_j} + \frac{1}{4} \text{tr} \left( \frac{\partial \Pi}{\partial u_l} \Pi^{-1} \frac{\partial \Pi}{\partial u_j} \Pi^{-1} \right),
\]
\[
E\left( \frac{\partial L^\top}{\partial u_l} \frac{\partial L}{\partial x_j} \right) = -\frac{\partial f^\top}{\partial u_l} \Pi^{-1} e_j,
\]
\[
E\left( \frac{\partial L^\top}{\partial x_l} \frac{\partial L}{\partial x_j} \right) = \frac{\partial h^\top}{\partial x_l} \Pi^{-1} \frac{\partial h}{\partial x_j} + e_l^{\top} \Pi^{-1} e_j,
\]
\[
E(\nabla_x^\top L \nabla_x L) = \nabla_x h R^{-1} \nabla_x L + \Pi^{-1}.
\]

The above equations lead to the ISISF-w/oDF algorithm for nonlinear systems without direct feedthrough, which is summarized in Table 4.2.

**Table 4.2**: The ISISF-w/oDF algorithm: Iterated SISF for nonlinear systems without direct feedthrough.

- **initialize**: \( k = 0, \hat{x}_0 = E(x_0), P_0^x = p_0 I \), where \( p_0 \) is a large positive value
- **repeat**
  - \( k \leftarrow k + 1 \)
  - initialize: \( i = 0, \hat{\sigma}^{(0)}_{k|k} = 0 \), i.e., \( \hat{u}^{(0)}_{k-1|k} = 0, \hat{x}^{(0)}_{k|k} = 0 \)
- while \( \ell < \ell_{\text{max}} \) do
  - perform Gauss-Newton based joint input and state estimation via (4.24)
  - \( \ell \leftarrow \ell + 1 \)
- end while
- export \( \hat{u}_{k-1|k} \) and \( \hat{x}_{k|k} \) from \( \hat{\sigma}_{k}^{(\ell_{\text{max}})} \)
- compute joint estimation error covariance via (4.25)
- **until** no more measurements arrive
4.4 SISF for Linear Systems

4.4.1 SISF Algorithms for Linear Systems

Developed for nonlinear systems, the proposed SISF-wDF and SISF-w/oDF algorithms can be readily reduced for linear systems. Let us begin with the case when direct-feedthrough exists. Consider the following system:

\[
\begin{align*}
\begin{cases}
x_{k+1} &= F_k x_k + G_k u_k + w_k, \\
y_k &= H_k x_k + J_k u_k + v_k.
\end{cases}
\end{align*}
\] (4.26)

It can be rewritten as

\[
\begin{align*}
\begin{cases}
x_{k+1} &= \bar{F}_k \xi_k + w_k, \\
y_k &= \bar{H}_k \xi_k + v_k,
\end{cases}
\end{align*}
\] (4.27)

where

\[
\begin{align*}
\bar{F}_k &= \begin{bmatrix} G_k & F_k \end{bmatrix}, \\
\bar{H}_k &= \begin{bmatrix} J_k & H_k \end{bmatrix}.
\end{align*}
\]

According to (4.9), the prediction step is given by

\[
\begin{align*}
\hat{x}_{k|k-1} &= \bar{F}_{k-1} \hat{\xi}_{k-1|k-1}, \\
P_{x_{k|k-1}} &= \bar{F}_{k-1} P_{\xi_{k-1|k-1}} \bar{F}_{k-1}^\top + Q_{k-1}.
\end{align*}
\] (4.28)

(4.29)

For a linear system, the Gauss-Newton method can reach the minimum point in one single iteration. Thus the update step is

\[
\begin{align*}
\hat{\xi}_{k|k} &= \hat{\xi}_{k|k}^{(0)} - \\
&\quad \left[ \begin{bmatrix} J_k^\top R_k^{-1} J_k & J_k^\top R_k^{-1} H_k \\
H_k^\top R_k^{-1} J_k & H_k^\top R_k^{-1} H_k + (P_{x_{k|k-1}}) \end{bmatrix} \right]^{-1} \\
&\quad \cdot \left[ \begin{bmatrix} -J_k^\top R_k^{-1} \\
-H_k^\top R_k^{-1} \end{bmatrix} \right] (y_k - H_k \hat{x}_{k|k-1}),
\end{align*}
\] (4.30)

\[
\begin{align*}
P_{\xi_{k|k}} &= \left[ \begin{bmatrix} J_k^\top R_k^{-1} J_k & J_k^\top R_k^{-1} H_k \\
H_k^\top R_k^{-1} J_k & H_k^\top R_k^{-1} H_k + (P_{x_{k|k-1}}) \end{bmatrix} \right]^{-1}.
\end{align*}
\] (4.31)
where $\hat{\xi}^{(0)}_{k|k} = [0^\top \hat{x}_{k|k-1}^\top]^\top$. The obtained linear SISF algorithm, named LSISF-wDF, is shown in Table 4.3.

**Table 4.3:** The LSISF-wDF algorithm: SISF for linear systems with direct feedthrough.

| initialize: | $k = 0$, $\hat{\xi}_{0|0} = E(\xi_0)$, $P_{0|0}^\xi = p_0 I$, where $p_0 > 0$, typically a large positive value |
| repeat | $k \leftarrow k + 1$ |
|Prediction: | project the state ahead via (4.28) |
| | compute the prediction error covariance via (4.29) |
|Update: | initialize: $i = 0$, $\hat{\xi}^{(0)}_{k|k} = [0^\top \hat{x}_{k|k-1}^\top]^\top$ |
| | compute the joint input and state estimate via (4.30) |
| | compute the joint estimation error covariance via (4.31) |
| | export $\hat{u}_{k|k}$ and $\hat{x}_{k|k}$ from $\hat{\xi}_{k|k}$ |
| until | no more measurements arrive |

Now consider the linear system without direct feedthrough:

$$
\begin{align*}
\begin{cases}
\dot{x}_{k+1} = F_k x_k + G_k u_k + w_k, \\
y_k = H_k x_k + v_k.
\end{cases}
\end{align*}
$$

(4.32)

Following Section 4.3, the SISF for (4.32) can be carried out as follows:

$$
\hat{\sigma}_{k|k} = \left(\begin{bmatrix}
G_{k-1}^\top V_{k-1} G_{k-1} & G_{k-1}^\top V_{k-1} \\
V_{k-1}^{-1} G_{k-1} & H_k R_k^{-1} H_k + V_{k-1}^{-1}
\end{bmatrix}\right)^{-1}
\begin{bmatrix}
G_{k-1}^\top V_{k-1}^{-1} F_{k-1} \hat{x}_{k-1|k-1} \\
H_k R_k^{-1} y_k - V_{k-1}^{-1} F_{k-1} \hat{x}_{k-1|k-1}
\end{bmatrix}.
$$

(4.33)

where

$$
V_{k-1} = F_{k-1} P_{k-1|k-1}^x F_{k-1}^\top + Q_{k-1}.
$$
The joint estimation error covariance is given by

\[ P_{k|k}^\sigma = \left( \begin{bmatrix} G_{k-1}^T V_{k-1}^{-1} G_{k-1} & G_{k-1}^T V_{k-1}^{-1} H_k R_k^{-1} H_k + V_{k-1}^{-1} \\ V_{k-1}^{-1} G_{k-1} & H_k R_k^{-1} H_k + V_{k-1}^{-1} \end{bmatrix} \right)^{-1}. \]  

(4.34)

Table 4.4 summarizes the algorithm.

**Table 4.4**: The LSISF-w/oDF algorithm: SISF for linear systems without direct feedthrough.

| initialize: k = 0, \( \hat{x}_0 = E(x_0) \), \( P_{x0} = p_0 I \), where \( p_0 \) is a large positive value |
| repeat |
| \( k \leftarrow k + 1 \) |
| compute joint input and state estimate via (4.33) |
| export \( u_{k-1|k} \) and \( \hat{x}_{k|k} \) from \( \hat{\sigma}_{k|k} \) |
| compute joint estimation error covariance via (4.34) |
| until no more measurements arrive |

### 4.4.2 Connections with Other Algorithms

The LSISF-wDF and LSISF-w/oDF algorithms have connections with some methods in the literature, which are worth discussing here.

The work [86, 87] introduces algorithms for SISF of linear systems with and without direct feedthrough, respectively, based on recursive filter design that couples the input and state estimation. The design builds on minimum variance unbiased estimation (MVUE). Here, we present a brief review of the Gillijns-De Moor’s SISE algorithms for linear discrete-time systems.

Let us first consider a the system in (4.26), for which a sequential three-step filter of the following form is designed:

\[
\begin{align*}
\hat{x}_{k|k-1} &= F_{k-1} \hat{x}_{k-1|k-1} + G_{k-1} \hat{u}_{k-1|k-1}, \\
\hat{u}_{k|k} &= M_k (y_k - H_k \hat{x}_{k|k-1}), \\
\hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k (y_k - H_k \hat{x}_{k|k-1}),
\end{align*}
\]

(4.35) (4.36) (4.37)

where \( M_k \) and \( L_k \) are gain matrices to design. Here, (4.35) yields \( \hat{x}_{k|k-1} \) as
prediction of $x_k$, then the input $u_k$ is estimated in (4.36) when $y_k$ arrives, and finally, (4.37) updates $\hat{x}_{k|k-1}$ in (4.35) to $\hat{x}_{k|k}$ using $y_k$.

The Gillijns-De Moor’s algorithm [86] for the linear system in (4.26) is given as follows:

- Prediction:

$$\hat{x}_{k|k-1} = F_{k-1} \hat{x}_{k-1|k-1} + G_{k-1} \hat{u}_{k-1|k-1},$$

$$P_{x_{k|k-1}} = \begin{bmatrix} F_{k-1} & G_{k-1} \\ \end{bmatrix} \begin{bmatrix} P_{x_{k-1|k-1}} & P_{xu_{k-1}} \\ P_{u_{x_{k-1}}} & P_{u_{k-1|k-1}} \\ \end{bmatrix} \begin{bmatrix} F_{k-1}^\top \\ G_{k-1}^\top \\ \end{bmatrix} + Q_{k-1}. \quad (4.39)$$

- Unknown input estimation:

$$\tilde{R}_k = H_k P_{x_{k|k-1}} H_k^\top + R_k,$$

$$M_k = \left( J_k^\top \tilde{R}_k^{-1} J_k \right)^{-1} J_k^\top \tilde{R}_k^{-1},$$

$$\hat{u}_{k|k} = M_k (y_k - H_k \hat{x}_{k|k-1}),$$

$$P_{u_{k|k}} = \left( J_k^\top \tilde{R}_k^{-1} J_k \right)^{-1}. \quad (4.43)$$

- Update:

$$K_k = P_{x_{k|k-1}} H_k^\top \tilde{R}_k^{-1},$$

$$\hat{x}_{kk} = \hat{x}_{k|k-1} + K_k (y_k - H_k \hat{x}_{k|k-1} - J_k \hat{u}_{k|k}),$$

$$P_{x_{k|k}} = P_{x_{k|k-1}} - K_k (\tilde{R}_k - J_k P_{u_{k|k}} J_k^\top) K_k^\top,$$

$$P_{xu_{k|k}} = (P_{u_{x_{k|k}}})^\top = -K_k J_k P_{u_{k|k}}. \quad (4.47)$$

The algorithm development is based on the MVUE technique, through which the input and state estimation are guaranteed to have the lowest variances under unbiased constraints. Essentially being a linear optimal filter, it has a KF-like structure — if the input is known (i.e., $\hat{u}_{k|k} = u_k$ and $P_{u_{k|k}} = 0$), the state estimation becomes exactly the standard KF.

Now consider the system in (4.32). Since there is no direct feedthrough
from the input to the output in the system, the input estimation is delayed by one
time unit. In this case, an optimal SISE algorithm building on MVUE is developed
in an analogous way in [87]. The filter designed accordingly is

\[
\hat{x}_{k|k-1} = F_{k-1} \hat{x}_{k-1|k-1}, \quad (4.48)
\]

\[
\hat{u}_{k-1|k} = M_k(y_k - H_k \hat{x}_{k|k-1}), \quad (4.49)
\]

\[
\hat{x}_k^* = \hat{x}_{k|k-1} + G_{k-1} \hat{u}_{k-1|k}, \quad (4.50)
\]

\[
\hat{x}_{k|k} = \hat{x}_k^* + K_k(y_k - H_k \hat{x}_{k|k-1}). \quad (4.51)
\]

For the above filter, the MVUE-based Gillijns-De Moor’s algorithm is ob-
tained as follows:

- **Unknown input estimation:**

  \[
  \hat{R}_k = H_k (F_{k-1} P^x_{k-1} F_{k-1} + Q_k) H_k^\top + R_k,
  \]

  \[
  M_k = \left( G_{k-1} H_k^\top \hat{R}_k H_k G_{k-1} \right)^{-1} G_{k-1} H_k^\top \hat{R}_k^{-1},
  \]

  \[
  \hat{u}_{k-1|k} = M_k(y_k - H_k \hat{x}_{k|k-1}),
  \]

  \[
  P^u_{k-1|k} = \left( G_{k-1} H_k^\top \hat{R}_k H_k G_{k-1} \right)^{-1}.
  \]

- **State estimation:**

  \[
  \hat{x}_k^* = \hat{x}_{k|k-1} + G_{k-1} \hat{u}_{k-1|k},
  \]

  \[
  F^*_k = (I - G_{k-1} M_k H_k) F_{k-1},
  \]

  \[
  Q^*_k = (I - G_{k-1} M_k H_k) Q_{k-1} (I - G_{k-1} M_k H_k)^\top
  \]

  \[
  + G_{k-1} M_k R_k M_k^\top G_{k-1}^\top,
  \]

  \[
  P^*_k = F^*_k P_{k-1|k-1} F^*_k + Q^*_k,
  \]

  \[
  S^*_k = -G_{k-1} M_k R_k,
  \]

  \[
  K_k = (P^*_k H_k^\top + S^*_k) U_k^\top (U_k \hat{R}_k^* U_k^\top)^{-1} U_k,
  \]

  \[
  \hat{x}_{k|k} = \hat{x}_k^* + K_k(y_k - H_k \hat{x}_{k|k-1}),
  \]

  \[
  P^x_{k|k} = P^*_k - K_k (P^*_k H_k^\top + S^*_k)^\top.
  \]
Here, determination of the optimal gain matrix $K_k$ involves a singular matrix $\tilde{R}_k^* = H_k P_{k}^* H_k^\top + R_k + H_k S_{k}^* + S_k^* H_k^\top$. Due to the singularity problem, the optimal $K_k$ is not unique. Thus it is parameterized by $U_k$, where $U_k \in \mathbb{R}^{q \times p}$, with $q$ being the rank of $\tilde{R}_k$, is an arbitrary matrix chosen such that $U_k \tilde{R}_k^* U_k^\top$ has full rank.

The following result shows that the LSISF-wDF is a generalization of the Gillijns-De Moor’s algorithm.

**Theorem 4.3.** The LSISF-wDF applied to a linear system with direct feedthrough yields the same input and state estimation as in [86].

**Proof.** Direct observation shows that the prediction equations of LSISF-wDF match (4.38)-(4.39). Let us consider (4.30) for joint input and state estimation of LSISF-wDF, which can be rewritten as

$$
\begin{bmatrix}
\hat{u}_k \\
\hat{x}_{k|k}
\end{bmatrix} = \begin{bmatrix}
0 \\
\hat{x}_{k|k-1}
\end{bmatrix} - \begin{bmatrix}
\Phi_{11} & \Phi_{12} \\
\Phi_{21} & \Phi_{22}
\end{bmatrix}^{-1} \begin{bmatrix}
\Omega_1 \\
\Omega_2
\end{bmatrix} (y_k - H_k \hat{x}_{k|k-1}),
$$

where

$$
\Phi_{11} = J_k^\top R_k^{-1} J_k,
$$

$$
\Phi_{12} = \Phi_{21} = J_k^\top R_k^{-1} H_k,
$$

$$
\Phi_{22} = H_k^\top R_k^{-1} H_k + (P_{x|k|k-1})^{-1},
$$

$$
\Omega_1 = -J_k^\top R_k^{-1},
$$

$$
\Omega_2 = -H_k^\top R_k^{-1}.
$$

Let $\Delta$ be the inverse of $\Phi$. Then we have

$$
\Delta_{11} = (\Phi_{11} - \Phi_{12} \Phi_{22}^{-1} \Phi_{21})^{-1}
= (J_k^\top R_k^{-1} J_k - J_k^\top R_k^{-1} H_k \Phi_{22}^{-1} H_k^\top R_k^{-1} J_k)^{-1}
= (J_k^\top \tilde{R}_k^{-1} J_k)^{-1},
$$

$$
\Delta_{12} = -\Delta_{11} \Phi_{12} \Phi_{22}^{-1}
= -(J_k^\top \tilde{R}_k^{-1} J_k)^{-1} J_k^\top R_k^{-1} H_k \Phi_{22}^{-1},
$$

$$
\Delta_{21} = \Phi_{11}^{-1} \Phi_{12} \Phi_{22}^{-1}
= \Phi_{11}^{-1} (J_k^\top \tilde{R}_k^{-1} J_k)^{-1} J_k^\top R_k^{-1} H_k \Phi_{22}^{-1},
$$

$$
\Delta_{22} = \Phi_{22}^{-1} - \Phi_{22}^{-1} \Phi_{11} \Phi_{12} \Phi_{22}^{-1}
= \Phi_{22}^{-1} -(J_k^\top \tilde{R}_k^{-1} J_k)^{-1} J_k^\top R_k^{-1} H_k \Phi_{22}^{-1}.
$$
\[ \Delta_{21} = \Delta_{12}^{\top}, \]
\[ \Delta_{22} = \Phi^{-1}_{22} + \Phi_{22}^{-1} \Phi_{21} \Delta_{11} \Phi_{12} \Phi_{22}^{-1} \]
\[ = \Phi^{-1}_{22} + \Phi_{22}^{-1} H_{k}^{\top} R_{k}^{-1} J_{k} (J_{k}^{\top} R_{k}^{-1} J_{k})^{-1} J_{k}^{\top} R_{k}^{-1} H_{k} \Phi^{-1}_{22}, \]

where \( \tilde{R}_{k} := H P_{k|k-1}^{x} H_{k}^{\top} + R_{k} \), as is defined in [86]. The input estimation formula can be expressed as

\[
\hat{u}_{k|k} = -\frac{\left(\Delta_{11} \Omega_{1} + \Delta_{12} \Omega_{2}\right)}{L_{k}} (y_{k} - H_{k} \hat{x}_{k|k-1})
\]
\[ = T_{k} (y_{k} - H_{k} \hat{x}_{k|k-1}). \]

It can be easily verified that \( T_{k} = M_{k} \), where \( M_{k} \) is given in (4.41).

For state estimation, we have

\[
\hat{x}_{k|k} = \hat{x}_{k|k-1} - \frac{\left(\Delta_{21} \Omega_{1} + \Delta_{22} \Omega_{2}\right)}{L_{k}} (y_{k} - H_{k} \hat{x}_{k|k-1}).
\]

It follows that

\[
L_{k} = -\Phi^{-1}_{22} H_{k}^{\top} R_{k}^{-1} J_{k} (J_{k}^{\top} \tilde{R}_{k} J_{k})^{-1} J_{k}^{\top} R_{k}^{-1}
\]
\[ + \left(\Phi^{-1}_{22} + \Phi_{22}^{-1} H_{k}^{\top} R_{k}^{-1} J_{k} (J_{k}^{\top} \tilde{R}_{k} J_{k})^{-1} J_{k}^{\top} R_{k}^{-1} H_{k} \Phi^{-1}_{22}\right) H_{k}^{\top} R_{k}^{-1}
\]
\[ = \Phi^{-1}_{22} H_{k}^{\top} R_{k}^{-1} (I - J_{k} M_{k})
\]
\[ = S_{k}(I - J_{k} M_{k}), \]

where

\[ S_{k} := \Phi_{22}^{-1} H_{k}^{\top} R_{k}^{-1} \]
\[ = (H_{k}^{\top} R_{k}^{-1} H_{k} + (P_{k|k-1}^{x})^{-1})^{-1} H_{k}^{\top} R_{k}^{-1}
\]
\[ = \left(\frac{P_{k|k-1}^{x}}{P_{k|k-1}^{x} + H_{k} \tilde{R}_{k} H_{k}^{\top} P_{k|k-1}^{x} H_{k}^{\top} R_{k}^{-1}}\right) H_{k}^{\top} R_{k}^{-1}
\]
\[ = P_{k|k-1}^{x} H_{k}^{\top} R_{k}^{-1} - P_{k|k-1}^{x} H_{k}^{\top} R_{k}^{-1} H_{k} P_{k|k-1}^{x} H_{k}^{\top} R_{k}^{-1}
\]
\[ = P_{k|k-1}^{x} H_{k}^{\top} R_{k}^{-1} - P_{k|k-1}^{x} H_{k}^{\top} R_{k}^{-1} (\tilde{R}_{k} - R_{k}) R_{k}^{-1} \]
\[ = P_{k|k-1}^x H_k^\top \tilde{R}_k^{-1}. \]

Hence,

\[
\hat{x}_{k|k} = \hat{x}_{k|k-1} + L_k (y_k - H_k x_{k|k-1}) \\
= \hat{x}_{k|k-1} + S_k (I - J_k M_k) (y_k - H_k x_{k|k-1}) \\
= \hat{x}_{k|k-1} + S_k (y_k - H_k x_{k|k-1} - J_k \hat{u}_{k|k}).
\]

It is seen that \( S_k \) is exactly \( K_k \) in (4.44), and that the above state estimation formula is identical to (4.45).

Finally, we investigate the estimation error covariances. It is found in the linear situation, the Fisher information matrix \( \mathcal{I} \) is equivalent to \( \Phi \). Therefore,

\[
P_{k|k}^u = \Delta_{11} = (J_k^\top \tilde{R}_k^{-1} J_k)^{-1},
\]

\[
P_{k|k}^x = \Phi_{22}^{-1} + \Phi_{22}^{-1} H_k^\top \tilde{R}_k^{-1} J_k (J_k^\top \tilde{R}_k^{-1} J_k)^{-1} J_k^\top \tilde{R}_k^{-1} H_k \Phi_{22}^{-1} \\
= \Phi_{22}^{-1} + S_k J_k P_{k|k}^{u} J_k^\top S_k^\top \\
= P_{k|k-1}^x - S_k (\tilde{R}_k - J_k P_{k|k}^{u} J_k^\top) S_k^\top, \\
P_{k|k}^{xu} = (P_{k|k}^{ux})^\top = \Delta_{21} = -S_k J_k P_{k|k}^{u}.
\]

since \( \Phi_{22}^{-1} = P_{k|k}^x - S_k \tilde{R}_k S_k^\top \). The above covariance formulae match (4.43), (4.46), (4.47), respectively. The proof is complete.

**Remark 4.3.** The algorithm in [86] has the same state update as that in [77], which deals with MVUE-based state estimation of linear systems with unknown inputs. Thus Theorem 4.3 reflects the fact that the Bayesian paradigm in (4.2)-(4.3) offers a general framework to address SISE problems. This is further stressed by the fact that the Gillijns-De Moor’s algorithm and the Darouach et al.’s algorithm can be directly developed from the Bayesian paradigm in conjunction with MAP estimation, without using the numerical Gauss-Newton method, under the following assumptions

1. \( x_0 \sim \mathcal{N}(\hat{x}_{0|0}, P_{0|0}^x), \)
\( \{ w_k \} \) and \( \{ v_k \} \) are zero-mean white Gaussian noise sequences,\n
\( x_0, \{ w_k \} \) and \( \{ v_k \} \) are independent of each other, and \n
\( \{ u_k \} \) is a white Gaussian process, independent of \( x_0, \{ w_k \} \) and \( \{ v_k \} \).

Compared to the assumptions made for the classical Kalman filter, (4.3) is the only additional requirement, which ensures the state propagation and output measurement sequences are Gaussian distributed.

In the next theorem, we show that LSISF-w/oDF is equivalent the Gillijns-De Moor algorithm in [87]. The proof is similar to that of Theorem 4.3 and is therefore omitted here.

**Theorem 4.4.** If the ISISF-w/oDF is applied to a linear system without direct feedthrough, it yields the same input and state estimation as in [87].

It should be pointed out that the optimal state estimation gain matrix in [87] cannot be determined uniquely. One among all possible options is given by Eqn. (20) therein, which matches the state estimation gain that is obtained with the ISISF-w/oDF.

**Remark 4.4.** The same state update is given in [87] as [73, 76] and the same input update as [80]. Theorem 4.4 indicates that these methods can be regarded as special cases of the ISISF-w/oDF. By analogy, if the conditions (1)-(4) proposed in Remark 4.3 are also valid for the system in (4.32), the algorithm in [87] can be directly derived from the Bayesian paradigm by MAP estimation.

### 4.4.3 Asymptotic Stability of Linear SISF Algorithms

An unanswered question for SISF is: *Under what conditions the algorithms achieves time invariance and is asymptotically stable when the system is linear time-invariant?* It is well known that stability of the classical Kalman filter depends on detectability of \((F,H)\) and stabilizability of \((F,Q^{1/2})\) – see more details in [20] and [30]. Here, we will develop a parallel to establish the asymptotic stability properties of the linear SISF algorithms, LSISF-wDF and LSISF-w/oDF. The
analysis will be performed over their equivalences in [86, 87]. The results can be extremely useful not only in analysis of the SISE filter behavior, but also in its practical implementation.

We begin the investigation with the system in (4.32). Assume $X \in S = \{S \in \mathbb{R}^{n \times n} | S = S^\top, S \geq 0\}$, $M \in \mathbb{R}^{n_u \times n_y}$ and $K \in \mathbb{R}^{n_x \times n_y}$. We define the operator

$$
\phi(M, K, X) = \tilde{F}X\tilde{F}^\top + \tilde{A}Q\tilde{A}^\top + \tilde{G}R\tilde{G}^\top,
$$

where

$$
\tilde{F}(M, K) = (I - KH)(I - GMH)F, \quad (4.52)
$$

$$
\tilde{A}(M, K) = -(I - KH)(I - GMH), \quad (4.53)
$$

$$
\tilde{G}(M, K) = (I - KH)GM + K. \quad (4.54)
$$

Note that the filter in (4.48)-(4.51) can be rewritten as

$$
\hat{u}_{k-1|k} = M_k(y_k - HF\hat{x}_{k-1|k-1}), \quad (4.55)
$$

$$
\hat{x}_{k|k} = F\hat{x}_{k-1|k-1} + G\hat{u}_{k-1|k} + K_k(y_k - HF\hat{x}_{k-1|k-1} - HG\hat{u}_{k-1|k}). \quad (4.56)
$$

Define $\tilde{x}_k = x_k - \hat{x}_{k|k}$. It can be verified that

$$
\tilde{x}_k = \tilde{F}(M_k, K_k)\tilde{x}_{k-1} - \begin{bmatrix}
\tilde{A}(M_k, K_k) & \tilde{G}(M_k, K_k)
\end{bmatrix}
\begin{bmatrix}
w_{k-1} \\
v_k
\end{bmatrix}, \quad (4.57)
$$

The associated covariance matrix is given by

$$
P_{k|k} = \phi(M_k, K_k, P_{k-1|k-1}), \quad (4.58)
$$

where $\phi$ can be determined from (4.57).

The filter designed in [87] has global optimality. First, it yields state estimation equivalent to the filters in [73, 76] – both are globally optimal [101]. Second, the global optimality of input estimation depends on the filter form and the state
estimation. As the proposed recursive form is proven optimal and state estimation is optimal as aforementioned, the input estimation is also globally optimal. Thus we have the following lemma.

**Lemma 4.1.** [87] The gain matrices $M_k$ and $K_k$ designed in Theorems 2 and 7 in [87], denoted as $M_k^*$ and $K_k^*$, respectively, minimize the estimation covariance $P_{k|k}$ subject to the unbiasedness constraint

$$M_k HG = I. \quad (4.59)$$

If letting

$$\psi(X) = \phi(M^*, K^*, X) = \min_{M, K} \phi(M, K, X), \quad (4.60)$$

where $M^*$ and $K^*$ are the optimal $M$ and $K$, then Lemma 4.1 implies for the designed optimal filter that

$$P_{k|k} = \psi(P_{k|k-1}). \quad (4.61)$$

As pointed out in [30], we are usually concerned with two properties for the filter. The first one, *asymptotic time invariance*, is whether the recursive computation in (4.61) leads to a fixed point $\bar{P}$, where

$$\lim_{k \to \infty} P_{k|k} = \bar{P}. \quad (4.62)$$

If this happens, $\bar{P}$ is also the asymptotically constant (i.e., limiting) solution of (4.61). That is, $\bar{P}$ satisfies

$$\bar{P} = \psi(\bar{P}). \quad (4.63)$$

The other one is *asymptotic stability*. It is about whether the filter is asymptotically stable given the limiting gain matrices associated with $\bar{P}$, denoted as $M^*$ and $K^*$. We are interested in deriving conditions that guarantee both the properties simultaneously.

Let us introduce the notation $\lambda_i(X)$ to denote the $i$-th eigenvalue of a
square matrix $X$. We give the lemma below as the initial step for the property analysis.

**Lemma 4.2.** If there exist $M \in \mathbb{R}^{m \times p}$ and $K \in \mathbb{R}^{n \times p}$ such that $M$ satisfies (4.59) and

$$
|\lambda_i \left[ \bar{F}(M, K) \right] | < 1
$$

for $i = 1, 2, \cdots, n$, then the sequence $\{P_{k|k}^x\}$ is bounded, i.e., $P_{k|k}^x \leq \infty$ for any $k$ given any initial condition $0 \leq P_{0|0}^x \leq \infty$.

**Proof:** To assist in showing this, let us construct a suboptimal filter. Choose fixed $M^s \in \mathbb{R}^{m \times p}$ and $K^s \in \mathbb{R}^{n \times p}$ such that both (4.59) and (4.64) are satisfied. Following (4.55), consider the unbiased suboptimal filter

$$
\hat{u}_{k-1|k}^s = M^s(y_k - HF\hat{x}_{k-1|k-1}^s),
$$

$$
\hat{x}_{k|k}^s = F\hat{x}_{k-1}^s + G\hat{u}_{k-1|k}^s + K^s(y_k - HF\hat{x}_{k-1|k-1}^s - HG\hat{u}_{k-1|k}^s).
$$

Then following (4.57), the state estimation error $\tilde{x}_k^s = x_k - \hat{x}_{k|k}^s$ is given by

$$
\tilde{x}_k^s = \bar{F}(M^s, K^s)\tilde{x}_{k-1}^s - \begin{bmatrix} \tilde{A}(M^s, K^s) & \tilde{G}(M^s, K^s) \end{bmatrix} \begin{bmatrix} w_{k-1} \\ v_k \end{bmatrix},
$$

with the associated covariance matrix $P_{k+1}^s$ is given by

$$
P_k^s = \phi(M^s, K^s, P_{k-1}^s).
$$

Note that the suboptimal filter is asymptotically stable, since $|\lambda_i \left[ \bar{F}(M^s, K^s) \right] | < 1$ for all $i$. Thus $P_k^s$ is bounded for any nonnegative initial condition. Comparing the above suboptimal filter to the designed optimal filter, we note that, if both of them are initialized by $P_{0|0}^x$, then the optimality suggests $P_{k|k}^x \leq P_k^s$. This proves the boundedness of $P_{k|k}^x$. □

The following theorem establishes conditions for asymptotic time invariance and stability of the consequent time-invariant filter, extending the ideas used in
Kalman filter analysis in [30, 102].

**Theorem 4.5.** If there exist $M \in \mathbb{R}^{m \times p}$ and $K \in \mathbb{R}^{n \times p}$ satisfying (4.59) and (4.64), and if $(F, Q^\frac{1}{2})$ is stabilizable, then $P_{k|k}^x$ converges to a unique fixed point $\tilde{P}$ for any initial condition $P_{0|0}^x$, where $\tilde{P}$ is the unique positive semi-definite solution of $\tilde{P} = \psi(\tilde{P})$. Moreover, with the associated limiting gain matrices $\tilde{M}^*$ and $\tilde{K}^*$, the time-invariant filter is stable, i.e., $|\lambda_i[\tilde{A}(\tilde{M}^*, \tilde{K}^*)]| < 1$ for $i = 1, 2, \ldots, n$.

The proof is organized as follows. First, it is illustrated that $P_{k|k}^x$ is monotonically increasing and converges to a fixed point with zero initial condition. Second, the asymptotic stability is shown by proving that the time-invariant steady-state filter is stable. Finally, the convergence of $P_{k|k}^x$ and the stability of the filter are proven to hold for arbitrary nonnegative initial condition.

**Proof of Theorem 4.5.** If $P_{0|0}^x = 0$, $P_{k|k}^x$ is found monotonically increasing and thus convergent due to its boundedness proven in Lemma 4.2. To show this, we note that

$$P_{0|0}^x \leq P_{1|1}^x,$$

when $P_{0|0}^x = 0$. Repeatedly using the fact that, if $X \leq Y$, then

$$\psi(X) = \phi(M_X^*, K_X^*, X) \leq \phi(M_Y^*, K_Y^*, X) \leq \phi(M_Y^*, K_Y^*, Y) = \psi(Y),$$

we obtain

$$P_{k|k}^x \leq P_{k+1|k+1}^x. \quad (4.65)$$

Thus $P_{k|k}^x$ is monotonically increasing for zero initial condition. With the boundedness of $P_{k|k}^x$ proven in Lemma 4.2, we can conclude that $P_{k|k}^x$ converges to a fixed point $\tilde{P}$, which is the solution of (4.58).

For notational convenience, define $\tilde{F} = \tilde{F}(\tilde{M}^*, \tilde{K}^*)$, $\tilde{A} = \tilde{A}(\tilde{M}^*, \tilde{K}^*)$ and $\tilde{G} = \tilde{G}(\tilde{M}^*, \tilde{K}^*)$. Let us now show asymptotic stability, resorting to proof by contradiction. Assume that the time-invariant filter is unstable. Then there exist some eigenvalue $|\lambda| \geq 1$ and corresponding eigenvector $\omega$ such that $\tilde{F}\omega = \lambda\omega$. 
From (4.61) we have
\[ \omega^* \check{P} \omega = \omega^* \bar{F} \bar{F}^\top \omega + \omega^* \bar{A} Q \bar{A}^\top \omega + \omega^* \bar{G} R \bar{G}^\top \omega \]
\[ = |\lambda|^2 \omega^* \bar{P} \omega + \omega^* \bar{A} Q \bar{A}^\top \omega + \omega^* \bar{G} R \bar{G}^\top \omega, \]
where the superscript * denotes the complex conjugate as no confusion arises. Equivalently,
\[ (1 - |\lambda|^2) \omega^* \check{P} \omega = \omega^* \bar{A} Q \bar{A}^\top \omega + \omega^* \bar{G} R \bar{G}^\top \omega. \]
Since |\lambda| \geq 1, then both sides must be 0 such that the equality holds. Then,
\[ \left( A Q^{1/2} \right)^\top \omega = - \left[ (I - \bar{G} H) Q^{1/2} \right]^\top \omega = 0, \]
\[ \bar{G}^\top \omega = 0, \]
from which we have
\[ \bar{F}^\top \omega = \left( F - \bar{G} H \bar{F} \right)^\top \omega = F^\top \omega = \lambda \omega, \]
\[ Q^{1/2} \omega = 0, \]
since \( Q \geq 0 \) and \( R > 0 \). The above equations indeed imply \( (F, Q^{1/2}) \) is not stabilizable. The contradiction disproves the assumption that the time-invariant filter is unstable.

We now demonstrate that \( P_{x|k}^x \) approaches \( P \) for any \( P_{0|0}^x \geq 0 \). Comparing \( P_{x|k}^x \) and \( \check{P} \), we have
\[ P_{x|k}^x - \check{P} = \phi(\check{M}^*_k, \check{K}^*_k, P_{x|k-1}^x) - \phi(\check{M}^*, \check{K}^*, \check{P}) \leq \phi(\check{M}^*_k, \check{K}^*_k, P_{x|k-1}^x) - \phi(\check{M}^*, \check{K}^*, \check{P}) = \bar{F}(P_{x|k-1}^x - \check{P}) \bar{F}^\top. \]
As shown before, \( \bar{F} \) is stable. The right hand side obviously approaches 0 as
\( k \to \infty \), thus proving
\[
\lim_{k \to \infty} P_{x|k}^x \leq \bar{P}.
\] (4.66)

On the other hand,
\[
P_{x|k}^x \geq \psi^k(0),
\]
which implies
\[
\lim_{k \to \infty} P_{x|k}^x \geq \lim_{k \to \infty} \psi^k(0) = \bar{P}.
\] (4.67)

Consequently, we can conclude from (4.66)-(4.67) that (4.62) holds for an arbitrary positive semi-definite \( P_{x|0}^x \).

Finally, the uniqueness of \( \bar{P} \) can be established. Assume that there exists another fixed point \( \bar{P}' \), and that \( P_{x|0}^x = \bar{P}' \). It is observed from previous proof that \( P_{x|k}^x \to \bar{P} \). Thus \( \bar{P}' = \bar{P} \). \( \square \)

For linear time-invariant systems with direct feedthrough, unbiased minimum-variance input and state estimation is studied analogously in [86]. Consider a linear system in (4.26), the filter for which is designed as
\[
\dot{x}_{k|k-1} = F\dot{x}_{k-1|k-1} + G\dot{u}_{k-1|k-1},
\] (4.68)
\[
\dot{u}_{k|k} = M_k(y_k - H\dot{x}_{k|k-1}),
\] (4.69)
\[
\dot{x}_{k|k} = \dot{x}_{k|k-1} + L_k(y_k - H\dot{x}_{k|k-1}).
\] (4.70)

Here, \( M_k \) and \( L_k \) are gain matrices, which are determined via MVUE in [86]. The global optimality of the above filter is proven in [103].

Accordingly, define the following operator
\[
\varphi(X) = (F - FL^*H - GM^*H)X(F - FL^*H - GM^*H)^\top + Q
\]
\[ + (FL^* + GM^*)R(FL^* + GM^*)^\top,
\]
where

\[ M^* = \left[ J^\top \tilde{R}^{-1} J \right]^{-1} J^\top \tilde{R}^{-1}, \]
\[ L^* = X C^\top \tilde{R}^{-1}, \]
\[ \tilde{R} = H X H^\top + R. \]

The property analysis results for the filter in (4.68)-(4.70) are presented in the next theorem.

**Theorem 4.6.** For the linear time-invariant system in [86], if there exist \( M \in \mathbb{R}^{m \times p} \) and \( L \in \mathbb{R}^{n \times p} \) satisfying

\[
MJ = I, \\
LJ = 0, \\
\left| \lambda_i \left[ F - (FL + GM)H \right] \right| < 1
\]

for all \( i \), and if \( (F, Q^{\frac{1}{2}}) \) is stabilizable, then \( P_{k|k-1} \) converges to a unique fixed point \( \hat{P} \) for any initial condition \( P_{0|0}^{-1} \). \( \hat{P} \) is the unique positive semi-definite solution of the equation \( \hat{P} = \varphi(\hat{P}) \). Moreover, with the associated limiting gain matrices, the time-invariant filter is stable.

The proof can be done along the same line as that of Theorem 4.5, so we only provide a sketch. The case of zero initial condition is first studied. It can be proven that \( P_{k|k-1} \) will converge to a fixed point \( \hat{P} = \varphi(\hat{P}) \) if \( P_{0|0}^{-1} = 0 \). This is from the boundedness and increasing monotonicity of \( P_{k|k-1} \) with \( P_{0|0}^{-1} = 0 \). The obtained time-invariant filter can also be shown to be stable. Next, convergence of \( P_{k|k-1} \) and stability of the time-invariant filter can then be readily proven for an arbitrary initial condition, yielding the statement of Theorem 4.6.

### 4.5 Conclusion

We have investigated the SISF problem for nonlinear systems with and without direct feedthrough. Building on a general Bayesian statistical paradigm,
we have designed two algorithms that construct state and input estimators using MAP optimization for each type of systems. The algorithms are sequential, consisting of the prediction and update stages; furthermore, each update involves iterative searching as a result of using the Gauss-Newton method to minimize the nonlinear MAP cost function. We have shown that the proposed algorithms generalize some existing methods for input and/or state estimation for linear systems. In addition, we have analyzed the asymptotic stability properties of the linear SISF methods. Sufficient conditions are proposed to ensure that the state estimation error covariance approaches a unique fixed point, which is the solution of a matrix equation. It is further shown that the time-invariant filter is stable. The results are an extension of the KF stability theory to linear SISF.

This chapter includes the content from the following published papers:


Chapter 5

Iterated Nonlinear Simultaneous Input and State Smoothing

5.1 Introduction

This chapter studies *simultaneous input and state smoothing (SISS)*. SISF in Chapter 4 is about “estimating the present”, which uses all the measurements up until the present time to estimate the unknown input and state at the current time instant. By comparison, SISS “estimates the past input and state” at a time instant using a batch of measurements inclusive of those lagging behind that time instant. It is usually a desirable choice when real-time filtering is not required for being able to provide more accurate estimates. A very early exploration related with SISS is has been taken in [104] with the development of estimators and smoothers of a white noise process regarded as the input to a linear state-space system. Since then, almost all the works on joint input and state estimation are focused on the filtering problem or SISF, please see Chapter 4 for a detailed survey.

SISS in a *forward-backward* manner will be considered. The inputs are assumed completely unknown. Following similar lines in Chapter 4, we pose this problem in a Bayesian setting. A forward-backward Bayesian smoothing paradigm is constructed first for SISS. On the basis of the Bayesian paradigm, a practically implementable SISS algorithm is developed through MAP optimization. We begin
the investigation into SISS for nonlinear systems with direct feedthrough, and then extend the results to nonlinear systems without direct feedthrough, building the \text{ISISS-wDF} and \text{ISISS-w/oDF} algorithms, respectively. They are then specialized for linear systems, leading to the \text{LSISS-wDF} and \text{LSISS-w/oDF} algorithms.

### 5.2 SISS for Nonlinear Systems with Direct Feedthrough

#### 5.2.1 Bayesian Paradigm

Let us consider the following nonlinear discrete-time system with direct feedthrough in (4.1). From a statistical point of view, \( u_k, x_k \) and \( y_k \) are random variables, the sequences of which over time are different stochastic processes, propagating according to the state space equations. The measurements are sampled data containing information about the input and state variables. Define \( \mathbb{Y}_{1:k} := \{ y_1, y_2, \ldots, y_k \} \), which is a set of all the measurements up until time \( k \).

The real-time SISF, as discussed in Chapter 4, is aimed to compute the joint pdf of \( u_k, x_k \) conditioned on \( \mathbb{Y}_{1:k} \), \( p(u_k, x_k | \mathbb{Y}_{1:k}) \), or more specifically, the passing to \( p(u_k, x_k | \mathbb{Y}_{1:k}) \) from \( p(u_{k-1}, x_{k-1} | \mathbb{Y}_{1:k-1}) \). SISS, however, uses additional measurement data or the data after time \( k \) to obtain the joint pdf of \( u_k \) and \( x_k \) with higher accuracy. That is, \( p(u_k, x_k | \mathbb{Y}_{1:N}) \) for \( N \geq k \) is of particular interest to us in this situation.

For notational convenience, we define the augmented vector \( \xi_k = [u_k^\top \ x_k^\top]^\top \), which is to be estimated. For fully unknown inputs, we still adopt Assumption 4.1, which states that \( \{ u_k \} \) is a process white and independent of \( x_0, \{ w_k \} \) and \( \{ v_k \} \). Since no \textit{a priori} knowledge about \( u_k \) is available, it may take any possible values, and no information about \( u_l \) can be extracted from \( u_k \), thus justifying the whiteness assumption in A1. By this assumption, \( u_k \) is independent of \( \mathbb{X}_k := \{ x_0, x_1, \ldots, x_k \} \) and \( \mathbb{Y}_{k-1} \) [93].

The Bayesian paradigm for SISS can be established using the Bayes’ rule to build the backward recursion of \( p(\xi_k | \mathbb{Y}_{1:N}) \) from \( p(x_{k+1} | \mathbb{Y}_{1:N}) \), as shown in the
following theorem.

**Theorem 5.1.** Suppose that Assumption 4.1 holds. For the system in (4.1), the Bayesian smoothing paradigm for input and state estimation is given by

\[
p(\xi_k | Y_{1:N}) = p(\xi_k | Y_{1:k}) \int p(x_{k+1} | \xi_k) p(x_{k+1} | Y_{1:N}) \frac{p(x_{k+1} | Y_{1:k})}{p(x_{k+1} | Y_{1:k})} \, dx_{k+1}. \tag{5.1}
\]

**Proof.** It is seen that

\[
p(\xi_k | Y_{1:N}) = \int p(\xi_k, \xi_{k+1} | Y_{1:N}) \, d\xi_{k+1}
= \int p(\xi_k | \xi_{k+1}, Y_{1:N}) \, p(\xi_{k+1} | Y_{1:N}) \, d\xi_{k+1}.
\]

Note that \( \xi_k \) is independent of \( Y_{k+1:N} \) and \( u_{k+1} \) given \( x_{k+1} \), due to the Markovian propagation of the state and the assumption A1. Hence,

\[
p(\xi_k | x_{k+1}, Y_{1:k}) = p(\xi_k | x_{k+1}, Y_{1:k}).
\]

Then we have

\[
p(\xi_k | Y_{1:N}) = \int p(\xi_k | x_{k+1}, Y_{1:k}) \, p(x_{k+1} | Y_{1:N}) \, dx_{k+1}
= \int p(\xi_k | x_{k+1}, Y_{1:k}) \, p(x_{k+1} | Y_{1:N}) \, dx_{k+1}. \tag{5.2}
\]

Meanwhile, note that

\[
p(\xi_k | x_{k+1}, Y_{1:k}) = \frac{p(\xi_k, x_{k+1}, Y_{1:k})}{p(x_{k+1}, Y_{1:k})}
= \frac{p(x_{k+1} | \xi_k, Y_{1:k}) p(\xi_k | Y_{1:k})}{p(x_{k+1} | Y_{1:k})}
= \frac{p(x_{k+1} | \xi_k) p(\xi_k | Y_{1:k})}{p(x_{k+1} | Y_{1:k})}. \tag{5.3}
\]

Inserting (5.3) into (5.2), we obtain (5.1).

Illustrating how to calculate the conditional pdf of \( \xi_k \) given \( Y_{1:N} \), the Bayesian paradigm in (5.1) is an input and state smoother in a statistical sense.
However, direct, analytical evaluation of pdf’s is known to be intractable, if not impossible, for nonlinear systems, as nonlinear transformation of random variables is hard to deal with. Hence, we will seek a numerically feasible solution by introducing certain approximations.

### 5.2.2 SISS Algorithm Development

In this section, we present the development of the SISS algorithm for the nonlinear system in (4.1). Then we specialize the algorithm to the linear case.

The following Gaussian approximations are made:

**Assumption 5.1.** $p(x_{k+1}|\xi_k)$ is a Gaussian distribution with mean $f(\xi_k)$ and covariance $Q_k$.

**Assumption 5.2.** $p(\xi_k|\tilde{Y}_{1:k})$ is a Gaussian distribution with mean $\hat{\xi}_{k|k}$ and covariance $P_{k|k}^\xi$.

**Assumption 5.3.** $p(x_{k+1}|\tilde{Y}_{1:N})$ is a Gaussian distribution with mean $\hat{x}_{k+1|N}$ and covariance $P_{k+1|N}^x$.

**Assumption 5.4.** $p(x_{k+1}|\tilde{Y}_{1:k})$ is a Gaussian distribution with mean $\hat{x}_{k+1|k}$ and covariance $P_{k+1|k}^x$.

Here, $\hat{\xi}_{k|k}$ is the filtering estimate of $\xi_k$ given $\tilde{Y}_{1:k}$, $P_{k|k}^\xi$ is the filtering error covariance, $\hat{x}_{k+1|N}$ is the smoothing estimate of $x_{k+1}$ given $\tilde{Y}_{1:N}$, and $P_{k+1|N}^x$ is the smoothing error covariance. The assumption that $w_k$ is zero-mean and Gaussian implies A2. Random variables with unknown distributions are often approximated to be Gaussian, based on the justification by the central limit theorem. The Gaussian distribution functions are also convenient to deal with. The above assumptions, as will be seen, bridge the gap from the Bayesian paradigm in (5.1) to a numerical algorithm.

It is appealing to consider a joint smoother of input $u_k$ and state $x_k$ that maximizes $p(\xi_k|\tilde{Y}_{1:N})$. It can be expressed as follows:

$$\hat{\xi}_{k|N} = \arg \max_{\xi_k} p(\xi_k|\tilde{Y}_{1:N}). \quad (5.4)$$
The above maximization of $p(\xi_k | \mathbf{y}_{1:N} )$ can be transformed into the familiar form of minimization of a cost function with assistance of the assumptions A1-A5, as shown in the next theorem.

**Theorem 5.2.** Suppose that Assumptions 5.1-5.4 hold. For the system in (4.1), an equivalence of the smoother in (5.4) is given by

$$\hat{\xi}_{k|N} = \arg \min_{\xi_k} L(\xi_k),$$

(5.5)

where

$$L(\xi_k) = (\xi_k - \hat{\xi}_{k|k})^\top (P_{k|k}^\xi)^{-1} (\xi_k - \hat{\xi}_{k|k})$$

$$+ [f(\xi_k) - \delta_k]^\top \Delta_k^{-1} [f(\xi_k) - \delta_k],$$

(5.6)

$$\Delta_k = \left[ (P_{k+1|N}^x)^{-1} - (P_{k+1|k}^x)^{-1} \right]^{-1} + Q_k,$$

(5.7)

$$\delta_k = \left[ (P_{k+1|N}^x)^{-1} - (P_{k+1|k}^x)^{-1} \right]^{-1} \cdot \left[ (P_{k+1|N}^x)^{-1} \tilde{x}_{k+1|N} - (P_{k+1|k}^x)^{-1} \tilde{x}_{k+1|k} \right].$$

(5.8)

**Proof.** Consider the Bayesian paradigm in (5.1), where the probabilities can be substituted using the assumptions A2-A5. By Lemma A.4 in Appendix, we have

$$\frac{G(\mathbf{x}_{k+1} ; \mathbf{x}_{k+1|N} , P_{k+1|N}^x)}{G(\mathbf{x}_{k+1} ; \mathbf{x}_{k+1|k} , P_{k+1|k}^x)} \propto G(\mathbf{x}_{k+1} ; \mathbf{x}_{k+1} , \tilde{P}_{k+1}^x),$$

where

$$\tilde{P}_{k+1}^x = \left[ (P_{k+1|N}^x)^{-1} - (P_{k+1|k}^x)^{-1} \right]^{-1},$$

$$\tilde{x}_{k+1} = \tilde{P}_{k+1}^x \left[ (P_{k+1|N}^x)^{-1} \tilde{x}_{k+1|N} - (P_{k+1|k}^x)^{-1} \tilde{x}_{k+1|k} \right].$$

It then follows from Lemma A.5 that

$$\int G(\mathbf{x}_{k+1} ; f(\xi_k), Q_k) \cdot G(\mathbf{x}_{k+1} ; \mathbf{x}_{k+1}, \tilde{P}_{k+1}^x) d\mathbf{x}_{k+1} \propto G(\mathbf{f}(\xi_k) ; \delta_k, \Delta_k),$$
where

\[ \Delta_k = \bar{P}^x_{k+1} + Q_k, \]
\[ \delta_k = \bar{x}_{k+1}. \]

By (5.1), the likelihood function \( L(\xi_k) = p(\xi_k | \mathcal{Y}_{1:N}) \) is then

\[ L(\xi_k) \propto G (\xi_k; \hat{\xi}_k|k, P_{\xi | k}) G (f(\xi_k); \delta_k, \Delta_k). \]

Hence, the log-likelihood function \( L(\xi_k) \) to minimize is exactly the one in (5.6).

**Remark 5.1.** It is noted that (5.1) computes \( p(\xi_k | \mathcal{Y}_{1:N}) \) that is the *a posteriori* distribution of \( \xi_k \) given \( \mathcal{Y}_{1:N} \). Thus the estimator designed in (5.4) or (5.5) is exactly based on the MAP estimation. Therefore, \( L(\xi_k) \) in (5.6) is a log-MAP cost function. MAP estimation formulated within Bayesian framework has often been used to develop optimal filtering methods, e.g., [19]. Here, we extend this technique to address the smoothing problem and especially for the purpose of joint input and state estimation.

It is often extremely difficult to obtain an analytical solution to the optimization problem in (5.5) due to the nonlinearities, motivating the development of an approximate numerical solution. By observation, the sum of the weighted 2-norms in (5.6) can be rewritten as

\[ L(\xi_k) = s^T(\xi_k) s(\xi_k), \tag{5.9} \]

where

\[ s_k(\xi_k) = \begin{bmatrix} \left( P_{\xi | k} \right)^{-\frac{1}{2}} (\xi_k - \hat{\xi}_{k|k}) \\ \Delta_k^{-\frac{1}{2}} (f(\xi_k) - \delta_k) \end{bmatrix}. \]

The classical Gauss-Newton method, which has been developed for nonlinear least squares problems, can then be applied here. It is an iterative searching process that linearizes around the current arrival point, determines the best search direction and then moves forward to the next point. For the input and state smoother designed
in (5.5), the Gauss-Newton based solution $\hat{\xi}_{k|N} = \hat{\xi}^{(\ell_{\text{max}})}_{k|N}$ is iteratively computed by

$$
\hat{\xi}_{k|N}^{(\ell+1)} = \hat{\xi}_{k|N}^{(\ell)} - \nabla_{\xi} s \left( \hat{\xi}_{k|N}^{(\ell)} \right) \nabla_{\xi} s \left( \hat{\xi}_{k|N}^{(\ell)} \right)^{-1} \nabla_{\xi} s \left( \hat{\xi}_{k|N}^{(\ell)} \right) s \left( \hat{\xi}_{k|N}^{(\ell)} \right), \quad (5.10)
$$

where $(\ell)$ denotes the iteration number, $\ell_{\text{max}}$ is the pre-selected max number of iterations, and

$$
\nabla_{\xi} s(\xi_k) = \begin{bmatrix}
\left( P_{k|k}^{\xi} \right)^{-\frac{1}{2}} \\
\Delta_k^{-\frac{1}{2}} \nabla_{\xi} f(\xi_k)
\end{bmatrix}.
$$

According to the MAP estimation theory, the error covariance can be approximately estimated by the inverse of the Fisher information matrix $\mathcal{F}$ evaluated at the estimate:

$$
P_{k|N}^{\xi} = \mathcal{F}^{-1}(\hat{\xi}_{k|N}), \quad (5.11)
$$

where $\mathcal{F}$ is given by

$$
\mathcal{F}(\xi_k) = \left( P_{k|k}^{\xi} \right)^{-1} + \nabla_{\xi}^T f(\xi_k) \Delta_k^{-1} \nabla_{\xi} f(\xi_k). \quad (5.12)
$$

We say that (5.10)-(5.12) are the backward smoothing equations for input and state estimation. The corresponding forward filtering is based on the ISISS-wDF presented in Chapter 4. Summarizing (5.10)-(5.12), we obtain the ISISS-wDF algorithm shown in Table 5.1.

### 5.3 SISS for Nonlinear Systems without Direct Feedthrough

In this section, extension of the Bayesian smoothing of inputs and states to systems without direct feedthrough is performed. Consider the system in (4.19) It is differentiated from (4.1) in that the first measurement containing the information
Table 5.1: The ISISS-wDF algorithm: Iterated SISS for nonlinear systems with direct feedthrough.

<table>
<thead>
<tr>
<th>Forward filtering:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>for ( k = 1 ) to ( N ) do</td>
<td>compute filtered input and state estimate via the ISISF-wDF algorithm</td>
</tr>
<tr>
<td>end for</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Backward smoothing:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>for ( k = N - 1 ) to 1 do</td>
<td>initialize: ( \ell = 0 ), ( \hat{\xi}^{(0)}_{k</td>
</tr>
<tr>
<td>while ( \ell &lt; \ell_{\text{max}} ) do</td>
<td>perform Gauss-Newton-based joint input and state smoothing via (5.10)</td>
</tr>
<tr>
<td>( \ell \leftarrow \ell + 1 )</td>
<td></td>
</tr>
<tr>
<td>end while</td>
<td>export ( \hat{u}_{k</td>
</tr>
<tr>
<td>compute joint smoothing error covariance via (5.11)-(5.12)</td>
<td></td>
</tr>
<tr>
<td>end for</td>
<td></td>
</tr>
</tbody>
</table>

of \( u_k \) is \( y_{k+1} \) instead of \( y_k \). Thus another SISS algorithm needs to be developed for such a system, following roughly similar lines as in Sections 5.2.1.

We use the notation \( \boldsymbol{\sigma}_k = [u_{k-1}^T \ x_k^T]^T \) and Assumption 4.1. It is adopted because an unknown signal such as \( \{u_k\} \) can be regarded white as a result of its unpredictable variation in magnitude over time scale. The next theorem establishes the Bayesian paradigm for computation of \( p(\sigma_k | Y_{1:N}) \).

**Theorem 5.3.** Suppose that Assumption 4.1 holds. Given the system in (4.19), the Bayesian smoothing paradigm for input and state estimation is given by

\[
p(\sigma_k | Y_{1:N}) = p(\sigma_k | Y_{1:k}) \int \int \frac{p(x_{k+1} | x_k, u_k)p(\sigma_{k+1} | Y_{1:N})}{p(x_{k+1} | x_k, u_k)p(x_{k} | Y_{1:k})} d\mathbf{x}_k d\mathbf{x}_{k+1}. \tag{5.13}
\]

**Proof.** It is seen that

\[
p(\sigma_k | Y_{1:N}) = \int \int p(u_{k-1}, x_{k-1}, u_k, x_k, x_{k+1} | Y_{1:N}) d\mathbf{x}_{k-1} d\mathbf{u}_k d\mathbf{x}_{k+1}
\]

\[
= \int \int p(u_{k-1}, x_{k-1} | u_k, x_k, x_{k+1}, Y_{1:N})
\]

\[
\cdot p(x_k | u_k, x_{k+1}, Y_{1:N}) p(u_{k+1} | Y_{1:N}) d\mathbf{x}_{k-1} d\mathbf{u}_k d\mathbf{x}_{k+1}. \tag{5.14}
\]
The following equations hold:

\[
p(u_{k-1}, x_{k-1}|u_k, x_k, x_{k+1}, Y_{1:N}) = p(u_{k-1}, x_{k-1}|x_k, Y_{1:k})
= \frac{p(x_{k-1}|u_{k-1}, x_k, Y_{1:k}) p(u_{k-1}, x_k|Y_{1:k})}{p(x_k|Y_{1:k})},
\]

(5.15)

\[
p(x_k|u_k, x_{k+1}, Y_{1:N}) = p(x_k|u_k, x_{k+1}, Y_{1:k})
= \frac{p(x_{k+1}|x_k, u_k) p(x_k, u_k|Y_{1:k})}{\int p(x_{k+1}|x_k, u_k) p(x_k, u_k|Y_{1:k}) dx_k}
= \frac{p(x_{k+1}|x_k, u_k) p(x_k|Y_{1:k})}{\int p(x_{k+1}|x_k, u_k) p(x_k|Y_{1:k}) dx_k},
\]

(5.16)

where the Markovian state propagation and the assumption A6 are used. Substituting (5.15)-(5.16) into (5.14) and integrating with respect to \(x_{k-1}\), we obtain (5.13).

Theorem 5.3 establishes the Bayesian solution framework to address SISS for systems without direct feedthrough. To derive an implementable algorithm, we approximate the following conditional random variables to be Gaussian.

**Assumption 5.5.** \(p(x_{k+1}|x_k, u_k)\) is a Gaussian distribution with mean \(f(x_k, u_k)\) and covariance \(Q_k\).

**Assumption 5.6.** \(p(\sigma_k|Y_{1:k})\) is a Gaussian distribution with mean \(\hat{\sigma}_{k|k}\) and covariance \(P_{\sigma k|k}\).

**Assumption 5.7.** \(p(\sigma_{k+1}|Y_{1:N})\) is a Gaussian distribution with mean \(\hat{\sigma}_{k+1|N}\) and covariance \(P_{\sigma k+1|N}\).

**Assumption 5.8.** \(p(x_k|Y_{1:k})\) is a Gaussian distribution with mean \(\hat{x}_{k|k}\) and covariance \(P_{\sigma k|k}\).

Apart from the above assumptions, we further approximate \(f(x_k, u_k)\) using the first-order Taylor series expansion around \((\hat{x}_{k|k}, \hat{u}_{k|N})\). That is,

\[
f(x_k, u_k) \approx \hat{F}_k x_k + \hat{G}_k u_k + \eta_k,
\]

(5.17)
where \( \hat{F}_k = \nabla_x f(\hat{x}_{k|k}, \hat{u}_{k|N}) \), \( \hat{G}_k = \nabla_u f(\hat{x}_{k|k}, \hat{u}_{k|N}) \) and \( \eta_k = f(\hat{x}_{k|k}, \hat{u}_{k|N}) - \hat{F}_k x_{k|k} - \hat{G}_k u_{k|N} \). The next theorem presents the MAP estimation problem to attain SISS for the system in (4.19).

**Theorem 5.4.** Suppose that Assumptions 5.5-5.8 hold. For the system in (4.19), the smoother \( \sigma_{k|N} = \arg \max_{\sigma_k} p(\sigma_k | Y_{1:N}) \) can be approximated using (5.17) by

\[
\hat{\sigma}_{k|N} = \arg \min_{\sigma_k} L(\sigma_k), \tag{5.18}
\]

where

\[
L(\sigma_k) = (\sigma_k - \hat{\sigma}_{k|k})^T (P^\sigma_{k|k})^{-1} (\sigma_k - \hat{\sigma}_{k|k}) + \left( \hat{F}_k N \sigma_k - \delta_k \right) \Delta_k^{-1} \left( \hat{F}_k N \sigma_k - \delta_k \right), \tag{5.19}
\]

\[
M_k = \begin{bmatrix} -\hat{G}_k & I \end{bmatrix}, \quad N = \begin{bmatrix} 0 & I \end{bmatrix}, \tag{5.20}
\]

\[
\Delta_k = \left( (M_k P^\sigma_{k+1|N} M_k^T)^{-1} - \Theta_{k+1}^{-1} \right)^{-1} + Q_k \tag{5.21}
\]

\[
\delta_k = \hat{F}_k \hat{x}_{k|k} - \Theta_{k+1} \left( M_k P_{k+1|N} M_k^T - \Theta_{k+1} \right)^{-1} \left( \hat{x}_{k+1|N} - f(\hat{x}_{k|k}, \hat{u}_{k|N}) \right), \tag{5.22}
\]

\[
\Theta_{k+1} = \hat{F}_k P^\sigma_{k|k} \hat{F}_k^T + Q_k. \tag{5.23}
\]

**Proof.** The proof is grounded on the Bayesian paradigm in (5.13). With Assumptions 5.5-(5.8), all the involved probabilities in (5.13) can be expressed as Gaussian functions. First consider

\[
\int G(x_{k+1}; f(x_k, u_k), Q_k) \cdot G(x_k; \hat{x}_{k|k}, P^x_{k|k}) \, dx_k, \tag{5.24}
\]

where \( f(x_k, u_k) \) can be replaced by its first-order Taylor series in (5.17). Then by Lemma A.7, we have

\[
\int G(x_{k+1}; \hat{F}_k x_k + \hat{G}_k u_k + \eta_k, Q_k) \cdot G(x_k; \hat{x}_{k|k}, P^x_{k|k}) \, dx_k \propto G(M_k \sigma_{k+1; \theta_{k+1}}, \Theta_{k+1}),
\]
where

\[ \Theta_{k+1} = \hat{F}_k P_{k|k} \hat{F}_k^\top + Q_k, \]

\[ \theta_{k+1} = \hat{F}_k \hat{x}_{k|k} + \eta_k = f(\hat{x}_{k|k}, \hat{u}_{k|N}) - \hat{G}_k \hat{u}_{k|N}. \]

Then it is seen from Lemma A.6 that

\[ G(\sigma_{k+1}; \sigma_{k+1|N}, P_{k+1|N}^\sigma) \]

\[ \frac{G(M_k \sigma_{k+1}; \theta_{k+1}, \Theta_{k+1})}{G(M_k \sigma_{k+1}; \theta_{k+1}, \Theta_{k+1})} = G(\sigma_{k+1}; \sigma_{k+1}, P_{k+1}^\sigma), \]

where

\[ \bar{P}_{k+1}^\sigma = \left( (P_{k+1|N}^\sigma)^{-1} - M_k^\top \Theta_{k+1}^{-1} M_k \right)^{-1}, \]

\[ \sigma_{k+1} = \bar{P}_{k+1}^\sigma \left( (P_{k+1|N}^\sigma)^{-1} \sigma_{k+1|N} - M_k^\top \Theta_{k+1}^{-1} \theta_{k+1} \right). \]

Using (5.17) and Lemma A.6, we have

\[ \int \int G(x_{k+1}; f(x_k, u_k), Q_k) \cdot G(\sigma_{k+1}; \sigma_{k+1}, \bar{P}_{k+1}^\sigma) du_k dx_{k+1} \]

\[ = \int \int G(x_{k+1}; \hat{F}_k \hat{x}_k + \hat{G}_k u_k + \eta_k, Q_k) \cdot G(\sigma_{k+1}; \sigma_{k+1}, \bar{P}_{k+1}^\sigma) du_k dx_{k+1} \]

\[ \propto \int \int G(M_k \sigma_{k+1}; \hat{F}_k \hat{x}_k + \eta_k, Q_k) \cdot G(\sigma_{k+1}; \sigma_{k+1}, \bar{P}_{k+1}^\sigma) d\sigma_{k+1} \]

\[ \propto G(\hat{F}_k \hat{x}_k; \delta_k, \Delta_k), \]

where

\[ \Delta_k = M_k P_{k+1}^\sigma M_k^\top + Q_k \]

\[ = M_k \left( (P_{k+1|N}^\sigma)^{-1} - M_k^\top \Theta_{k+1}^{-1} M_k \right)^{-1} M_k^\top + Q_k \]

\[ = \left[ (M_k P_{k+1|N}^\sigma M_k^\top)^{-1} - \Theta_{k+1}^{-1} \right]^{-1} + Q_k, \]

\[ \delta_k = M_k \sigma_{k+1} - \eta_k \]

\[ = M_k \left( (P_{k+1|N}^\sigma)^{-1} - M_k^\top \Theta_{k+1}^{-1} M_k \right)^{-1} \]
Remark 5.2. Theorem 5.4 formulates an easy-to-solve optimization problem due to the first-order Taylor approximation of establishing an optimization problem without Taylor approximation, since accuracy for smoothing but is still needed. We note that it is not possible here to proof.

The likelihood function \( L(\sigma_k) \) is

\[
L(\sigma_k) = p(\sigma_k|Y_{1:N}) = \mathcal{G}(\sigma_0, \theta, \sigma_{k|k}, P_{k|k}^\sigma) \cdot \mathcal{G}(\hat{F}_k N \sigma_k, \delta_k, \Delta_k),
\]

the logarithmic version of which, \( L(\sigma_k) \), is given by (5.19). This concludes the proof.

Remark 5.2. Theorem 5.4 formulates an easy-to-solve optimization problem due to the first-order Taylor approximation of \( f(x_k, u_k) \). It may lead to some loss of accuracy for smoothing but is still needed. We note that it is not possible here to establish an optimization problem without Taylor approximation, since \( u_{k-1} \) and \( x_k \), with different time indexes, are intercoupled and estimated simultaneously.

By Lemma A.8, the solution to (5.18) is

\[
\hat{\sigma}_{k|N} = \hat{\sigma}_{k|k} + P_{k|k}^\sigma N^T \hat{F}_k \left( \Delta_k + \hat{F}_k N P_{k|k}^\sigma N^T \hat{F}_k \right)^{-1} \left( \delta_k - \hat{F}_k N \hat{\sigma}_{k|k} \right)
\]

\[
= \hat{\sigma}_{k|k} - P_{k|k}^\sigma N^T \hat{F}_k \left[ \left( M_k P_{k|k+1}^\sigma N M_k^T - \Theta_{k+1} \right)^{-1} + \Theta_{k+1} \right]^{-1} \cdot \Theta_{k+1}^{-1} (\hat{x}_{k+1|N} - f(\hat{x}_{k|k}, \hat{u}_{k|N}))
\]

\[
= \hat{\sigma}_{k|k} + P_{k|k}^\sigma N^T \hat{F}_k \Theta_{k+1}^{-1} (\hat{x}_{k+1|N} - f(\hat{x}_{k|k}, \hat{u}_{k|N})).
\]

The associated smoothing covariance is then given by

\[
P_{k|N}^\sigma = P_{k|k}^\sigma - P_{k|k}^\sigma N^T \hat{F}_k \left( \Delta_k + \hat{F}_k N P_{k|k}^\sigma N^T \hat{F}_k \right)^{-1} \hat{F}_k N P_{k|k}^\sigma
\]

\[
= P_{k|k}^\sigma - P_{k|k}^\sigma N^T \hat{F}_k \left[ \left( M_k P_{k|k+1}^\sigma N M_k^T - \Theta_{k+1} \right)^{-1} + \Theta_{k+1} \right]^{-1} \hat{F}_k N P_{k|k}^\sigma
\]

\[
= P_{k|k}^\sigma + P_{k|k}^\sigma N^T \hat{F}_k \Theta_{k+1}^{-1} (M_k P_{k|k+1}^\sigma N M_k^T - \Theta_{k+1})^{-1} \Theta_{k+1}^{-1} \hat{F}_k N P_{k|k}^\sigma.
\]
Summarizing the above equations, we obtain the SISS formulae as follows:

\[
\hat{\sigma}_{k|N} = \sigma_{k|k} + P_{k|k}^\sigma N^T F_k^T \Theta_{k+1}^{-1} (\hat{x}_{k+1|N} - f(\hat{x}_{k|k}, \hat{u}_{k|N})),
\]

\[
P_{k|N}^\sigma = P_{k|k}^\sigma + P_{k|k}^\sigma N^T \bar{F}_k^T \Theta_{k+1}^{-1} (M_k P_{k+1|N} M_k^T - \Theta_{k+1})^{-1} \Theta_{k+1}^{-1} \bar{F}_k N P_{k|k}^\sigma,
\]

which form the ISISS-w/oDF algorithm shown in Table 5.2.

**Table 5.2:** The ISISS-w/oDF algorithm: Iterated SISS for nonlinear systems with direct feedthrough.

<table>
<thead>
<tr>
<th>Forward filtering:</th>
</tr>
</thead>
<tbody>
<tr>
<td>for k = 1 to N do</td>
</tr>
<tr>
<td>compute filtered input and state estimate via the ISISF-w/oDF algorithm</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Backward smoothing:</th>
</tr>
</thead>
<tbody>
<tr>
<td>for k = N − 1 to 2 do</td>
</tr>
<tr>
<td>perform joint input and state smoothing via (5.25)</td>
</tr>
<tr>
<td>export ( \hat{u}_{k-1</td>
</tr>
<tr>
<td>compute joint smoothing error covariance via (5.26)</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

### 5.4 SISS for Linear Systems

Developed for nonlinear systems, the ISISS-wDF and ISISS-w/oDF algorithms are examined for the linear case in this section. Considering the linear system with direct feedthrough in (4.26), we have

\[
s(\xi_k) = \begin{bmatrix} \left( P_{k|k}^\xi \right)^{-\frac{1}{2}} (\xi_k - \hat{\xi}_{k|k}) \\ \Delta_k^{-\frac{1}{2}} (F_k \xi_k - \delta_k) \end{bmatrix}, \quad \nabla_\xi s(\xi_k) = \begin{bmatrix} \left( P_{k|k}^\xi \right)^{-\frac{1}{2}} \\ \Delta_k^{-\frac{1}{2}} F_k \end{bmatrix}.
\]

Let \( \hat{\xi}_{k|N}^{(0)} \) be the initial guess of \( \hat{\xi}_{k|N} \). As the minimum point can be reached in a single iteration, the smoother in (5.10) becomes

\[
\hat{\xi}_{k|N} = \hat{\xi}_{k|N}^{(0)} - \left( P_{k|k}^\xi \right)^{-1} + \bar{F}_k^T \Delta_k^{-1} \bar{F}_k \right]^{-1} \left( P_{k|k}^\xi \right)^{-1} \left( \hat{\xi}_{k|N}^{(0)} - \hat{\xi}_{k|k} \right)
\]
\[ + F_k^T \Delta_k^{-1} \left( F_k \hat{\xi}^{(0)} - \delta_k \right) \]
\[ = \left[ \left( P^x_{k|k} \right)^{-1} + \bar{F}_k \Delta_k^{-1} \bar{F}_k \right]^{-1} \left[ \left( P^x_{k|k} \right)^{-1} \hat{\xi}_{k|k} + \bar{F}_k \Delta_k^{-1} \delta_k \right] \]
\[ = \hat{\xi}_{k|k} + \left[ \left( P^x_{k|k} \right)^{-1} + \bar{F}_k \Delta_k^{-1} \bar{F}_k \right]^{-1} \bar{F}_k \Delta_k^{-1} (\delta_k - \bar{F}_k \hat{\xi}_{k|k}) \]
\[ = \hat{\xi}_{k|k} + P^x_{k|k} \bar{F}_k \left( \Delta_k + \bar{F}_k P^x_{k|k} \bar{F}_k^T \right)^{-1} (\delta_k - \bar{F}_k \hat{\xi}_{k|k}). \]

Let \( \mu_k = \left( \Delta_k + \bar{F}_k P^x_{k|k} \bar{F}_k^T \right)^{-1} (\delta_k - \bar{F}_k \hat{\xi}_{k|k}) \). By the LSIF-wDF algorithm, we have \( \hat{x}_{k+1|k} = \bar{F}_k \hat{\xi}_{k|N} \). Then

\[ \mu_k = \left( \left[ \left( P^x_{k+1|N} \right)^{-1} - \left( P^x_{k+1|k} \right)^{-1} + P^x_{k+1|k} \right]^{-1} \cdot \left[ \left( P^x_{k+1|N} \right)^{-1} - \left( P^x_{k+1|k} \right)^{-1} \right]^{-1} \left( \hat{x}_{k+1|N} - \left( P^x_{k+1|k} \right)^{-1} \bar{F}_k \hat{\xi}_{k|k} \right) \right. \]
\[ = \left( \left( P^x_{k+1|N} \right)^{-1} - \left( P^x_{k+1|k} \right)^{-1} + P^x_{k+1|k} \right)^{-1} \cdot \left( \left( P^x_{k+1|N} \right)^{-1} - \left( P^x_{k+1|k} \right)^{-1} \right)^{-1} \left( \hat{x}_{k+1|N} - \bar{F}_k \hat{\xi}_{k|k} \right) \]
\[ = \left( P^x_{k+1|k} \right)^{-1} \left( \hat{x}_{k+1|N} - \bar{F}_k \hat{\xi}_{k|k} \right). \]

Hence,

\[ \hat{\xi}_{k|N} = \hat{\xi}_{k|k} + P^x_{k|k} \bar{F}_k^T \mu_k \]
\[ = \hat{\xi}_{k|k} + P^x_{k|k} \bar{F}_k^T \left( P^x_{k+1|k} \right)^{-1} \left( \hat{x}_{k+1|N} - \bar{F}_k \hat{\xi}_{k|k} \right) \]
\[ = \hat{\xi}_{k|k} + K_k \left( \hat{x}_{k+1|N} - \bar{F}_k \hat{\xi}_{k|k} \right). \]

which can be rewritten as

\[ \hat{\xi}_{k|N} = \hat{\xi}_{k|k} + K_k \left( \hat{x}_{k+1|N} - \bar{F}_k \hat{\xi}_{k|k} \right), \quad (5.27) \]
\[ K_k = P^x_{k|k} \bar{F}_k^T \left( P^x_{k+1|k} \right)^{-1}. \quad (5.28) \]
By (5.11)-(5.12), the error covariance is

$$P_{ξ|N}^k = \left[ \left( P_{ξ|k}^k \right)^{-1} + \tilde{F}_k^T \Delta_k^{-1} \tilde{F}_k \right]^{-1}$$

It is known from LSISF-wDF that $P^x_{k+1|k} = \tilde{F}_k P^ξ_{k|k} \tilde{F}_k^T + Q_k$. Then

$$P_{ξ|N}^k = P_{ξ|k}^k - P_{ξ|k}^k \tilde{F}_k^T \left( \left( P^x_{k+1|N} \right)^{-1} - \left( P^x_{k+1|k} \right)^{-1} \right)^{-1} P^x_{k+1|k} \tilde{F}_k P_{ξ|k}^k$$

$$= P_{ξ|k}^k + P_{ξ|k}^k \tilde{F}_k^T \left( P^x_{k+1|k} \right)^{-1} \left( P^x_{k+1|N} - P^x_{k+1|k} \right) \left( P^x_{k+1|k} \right)^{-1} \tilde{F}_k P_{ξ|k}^k$$

$$= P_{ξ|k}^k + K_k \left( P^x_{k+1|N} - P^x_{k+1|k} \right) K_k^T,$$

which can be written as

$$P_{ξ|k}^k = P_{ξ|k}^k + K_k \left( P^x_{k+1|N} - P^x_{k+1|k} \right) K_k^T. \quad (5.29)$$

The forward filtering corresponding to the smoothing above is the LSISF-wDF algorithm. Together, they will form the LSISS-wDF algorithm, which is given in Table 5.3

**Table 5.3:** The LSISS-wDF algorithm: SISS for linear systems with direct feedthrough.

<table>
<thead>
<tr>
<th>Forward filtering:</th>
</tr>
</thead>
<tbody>
<tr>
<td>for $k = 1$ to $N$ do</td>
</tr>
<tr>
<td>compute filtered input and state estimate via the LSISF-wDF algorithm</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Backward smoothing:</th>
</tr>
</thead>
<tbody>
<tr>
<td>for $k = N - 1$ to $1$ do</td>
</tr>
<tr>
<td>perform joint input and state smoothing via (5.27)-(5.29)</td>
</tr>
<tr>
<td>export $\hat{u}_{k</td>
</tr>
<tr>
<td>compute joint smoothing error covariance via (5.29)</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

Furthermore, let us consider linear systems without direct feedthrough. Specializing ISIIBI-w/oDF to the linear case gives rise to the following input and
state smoother:

\[
\hat{\sigma}_{k|N} = \hat{\xi}_{k|k} + K_k (\hat{x}_{k+1|N} - F_k \hat{x}_{k|k} - G_k \hat{u}_{k|N}), \tag{5.30}
\]

\[
K_k = P_{k|k}^\sigma N^T F_k^T \left( F_k P_{k|k}^\sigma F_k^T + Q_k \right)^{-1}, \tag{5.31}
\]

\[
P_{k|N} = P_{k|k}^\xi + K_k \left( M_k P_{k+1|N} M_k^T - F_k P_{k|k}^\sigma F_k^T - Q_k \right)^{-1} K_k^T, \tag{5.32}
\]

where

\[
M_k = \begin{bmatrix} -G_k & I \end{bmatrix}.
\]

The derivation is straightforward and thus omitted here. The forward filtering that corresponds to the backward smoothing (5.30)-(5.32) is the LSISF-w/oDF algorithm in Chapter 4. The LSISSS-w/oDF algorithm is summarized in Table. 5.4.

Table 5.4: The LSISSS-w/oDF algorithm: SISS for linear systems with direct feedthrough.

<table>
<thead>
<tr>
<th>Forward filtering:</th>
</tr>
</thead>
<tbody>
<tr>
<td>for ( k = 1 ) to ( N ) do</td>
</tr>
<tr>
<td>compute filtered input and state estimate via the LSISF-w/oDF algorithm</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Backward smoothing:</th>
</tr>
</thead>
<tbody>
<tr>
<td>for ( k = N - 1 ) to ( 1 ) do</td>
</tr>
<tr>
<td>perform joint input and state smoothing via (5.27)</td>
</tr>
<tr>
<td>export ( \hat{u}_{k-1</td>
</tr>
<tr>
<td>compute joint smoothing error covariance via (5.28)</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

5.5 Conclusions

Simultaneous input and state estimation or SISE is a challenging issue arising in a broad range of areas and applications. Existing research has been concerned with the filtering case. This chapter, however, has studied the smoothing problem, demanded by the practical application of high-accuracy but non-real-time flow-field reconstruction.
The forward-backward smoothing for SISE was considered for systems with and without direct feedthrough. It was treated from a statistical perspective, with a Bayesian framework constructed in the first place. A MAP based nonlinear smoothing algorithm was then developed within the framework to obtain smoothed input and state estimates. Developed for nonlinear systems, the algorithms were specialized to linear ones. All the algorithms have strong potential to be used in various multidisciplinary applications, in which where the inputs and states need to be estimated simultaneously.

This chapter is based on the following papers that were published or submitted:


Chapter 6

Ensemble Simultaneous Input and State Estimation

6.1 Introduction

In this chapter, we study ensemble-based input and state estimation, with the objective of developing SISF and SISS algorithms for large-scale nonlinear systems. This work is partially inspired by the EnKF. As a sequential Monte Carlo method for Bayesian filtering, the EnKF represents each pdf by a group of point samples with the assumption that the pdf’s involved are Gaussian and performs computation directly on the samples. Such a treatment gives rise to two advantages. First, its conceptual formulation is straightforward, simply based on the Bayesian Gaussian update; second, it has easy implementation due to its derivative-free, relatively efficient computation [105]. An extension of EnKF to smoothing is discussed in [105, 21, 106, 107] and the references therein to improve the estimation performance. Since its advent in [12], the EnKF has gained much popularity and provided significant impetus to the development of solutions to various high-dimensional issues, e.g., ocean, atmospheric, hydrological data assimilation\footnote{Data assimilation, a term in oceanography, meteorology and hydrology, refers to the process of incorporating measurements into the mathematical model to generate estimates of unknown variables or forecasts of future situations.} [12, 21, 105, 108, 109, 110]. Its other applications include highway traffic...
estimation [111], target tracking [112], power systems [113], reservoir engineering [114], etc.

Ensemble SISE remains an open problem yet. Thus we are motivated to extend the notion of EnKF to both SISF and SISS for nonlinear systems with and without direct feedthrough. Based on the thinking of Bayesian estimation, we will construct a series of algorithms, including EnSISF-wDF, EnSISF-w/oDF, EnSISS-wDF and EnSISS-w/oDF in this chapter, which will pave the way for application of SISE to large-scale nonlinear systems.

6.2 Ensemble SISF

6.2.1 Ensemble SISF for Nonlinear Systems with Direct Feedthrough

The Gaussian SISF based on Bayesian paradigms forms the foundation of ensemble SISF. Consider the nonlinear system with direct feedthrough in (4.1). Let \( \xi_k = [u_k^\top x_k^\top]^\top \). Intending to derive the estimate \( \xi_k \) from \( \hat{Y}_k \), we make take the prediction-update approach.

For prediction, it follows from (4.2) that

\[
\hat{x}_{k|k-1} = \int x_k p(x_k | \hat{Y}_{k-1}) dx_k = \int \left[ \int x_k p(x_k | \xi_{k-1}) dx_k \right] p(\xi_{k-1} | \hat{Y}_{k-1}) d\xi_{k-1} = \int f(\xi_{k-1}) p(\xi_{k-1} | \hat{Y}_{k-1}) d\xi_{k-1}.
\] (6.1)

The prediction error covariance is given by

\[
P_{k|k-1} = \int (x_k - \hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1})^\top p(x_k | \hat{Y}_{k-1}) dx_k
\]

\[
= \int x_k x_k^\top p(x_k | \hat{Y}_{k-1}) dx_k - \hat{x}_{k|k-1}^\top \hat{x}_{k|k-1} + Q_{k-1}.
\] (6.2)
To proceed further, we make the following pivotal assumption:

**Assumption 6.1.** \(p \left( \begin{bmatrix} \xi_k^T \\ y_k^T \end{bmatrix} \mid \mathbb{Y}_{k-1} \right)\) is a Gaussian distribution:

\[
p \left( \begin{bmatrix} \xi_k \\
 y_k \end{bmatrix} \mid \mathbb{Y}_{k-1} \right) = G \left( \begin{bmatrix} \xi_k \\
 y_k \end{bmatrix} ; \begin{bmatrix} P_{\xi k|k-1} \\
 P_{y y k|k-1} \end{bmatrix} \right). \tag{6.3}
\]

In (6.3), \(\hat{y}_{k|k-1}\) is prediction of \(y_k\) given \(\mathbb{Y}_{k-1}\):

\[
\hat{y}_{k|k-1} = \int y_k p(y_k \mid \mathbb{Y}_{k-1}) \, dy_k.
\]

Since

\[
p(y_k \mid \mathbb{Y}_{k-1}) = \int p(\xi_k, y_k \mid \mathbb{Y}_{k-1}) \, d\xi_k = \int p(y_k \mid \xi_k)p(\xi_k \mid \mathbb{Y}_{k-1}) \, d\xi_k,
\]

we have

\[
\hat{y}_{k|k-1} = \int \left[ \int y_k p(y_k \mid \xi_k) \, dy_k \right] p(\xi_k \mid \mathbb{Y}_{k-1}) \, d\xi_k = \int h(\xi_k)p(\xi_k \mid \mathbb{Y}_{k-1}) \, d\xi_k, \tag{6.4}
\]

The associated covariance is

\[
P_{y k|k-1} = \int (y_k - \hat{y}_{k|k-1})(y_k - \hat{y}_{k|k-1})^T p(y_k \mid \mathbb{Y}_{k-1}) \, dy_k
\]

\[
= \int h(\xi_k) h^T(\xi_k) p(\xi_k \mid \mathbb{Y}_{k-1}) \, d\xi_k - \hat{y}_{k|k-1} \hat{y}_{k|k-1}^T + R_k, \tag{6.5}
\]

and the cross-covariance is

\[
P_{\xi y k|k-1} = \int (\xi_k - \hat{\xi}_{k|k-1})(y_k - \hat{y}_{k|k-1})^T p(\xi_k, y_k \mid \mathbb{Y}_{k-1}) \, d\xi_k 
\]

\[
= \int \xi_k h(\xi_k)^T p(\xi_k \mid \mathbb{Y}_{k-1}) \, d\xi_k - \hat{\xi}_{k|k-1} \hat{y}_{k|k-1}^T. \tag{6.6}
\]
With Assumption 6.1 and Lemma A.3, we obtain

\[ p(\xi_k | \bar{Y}_k) = G(\xi_k; \bar{\xi}_k, \bar{P}_k^\xi), \]

where

\[ \bar{\xi}_k = \hat{\xi}_{k|k-1} + P^{\xi y}_{k|k-1} \left( P^{y}_{k|k-1} \right)^{-1} (y_k - \hat{y}_{k|k-1}), \]
\[ \bar{P}_k^\xi = P^\xi_{k|k-1} - P^{\xi y}_{k|k-1} \left( P^{y}_{k|k-1} \right)^{-1} \left( P^{\xi y}_{k|k-1} \right)^\top. \]

This leads to the updated estimate of \( \xi_k \), which is expressed as

\[ \hat{\xi}_{k|k} = \bar{\xi}_k, \quad (6.7) \]
\[ P^\xi_{k|k} = \bar{P}_k^\xi. \quad (6.8) \]

One question that would be raised about the above derivation is how to determine \( p(\xi_k | \bar{Y}_{k-1}) \) since \( u_k \) is unknown. Here, we adopt the following assumption:

**Assumption 6.2.** \( \{ u_k \} \) is a white process, independent of \( x_0, \{ w_k \} \) and \( v_k \), and \( p(u_k) \) follows a Gaussian distribution with mean \( \bar{u} \) and covariance \( \bar{P}^u \).

Assumption 6.2 proposes an ‘engineering’ way to deal with the unknown \( u_k \). For completely unknown \( u_k \), \( \bar{u} \) can be randomly assigned a value, and \( \bar{P}^u \) set equal to \( p_0 I \), where \( p_0 \) is a large positive value. If certain \( a \ priori \) information of \( u_k \) is available, one can decide \( \bar{u} \) and \( \bar{P}^u \) via incorporating such knowledge. For instance, \( \bar{u} \) and \( \bar{P}^u \) at time \( k \) can be approximated by the estimates obtained at the previous time instant, \( \hat{u}_{k-1|k-1} \) and \( \bar{P}^u_{k-1|k-1} \) when the change of \( u_k \) with \( k \) is continuous. With this assumption, we express \( \hat{\xi}_{k|k-1} \) and the associated covariance as

\[ \hat{\xi}_{k|k-1} = \begin{bmatrix} \bar{u} \\ \hat{x}_{k|k-1} \end{bmatrix}, \quad (6.9) \]
\[ P^\xi_{k|k-1} = \begin{bmatrix} \bar{P}^u \\ P^{x}_{k|k-1} \end{bmatrix}. \quad (6.10) \]
Now we are in a good position to summarize the Gaussian SISF algorithm, which is given in Table 6.1.

Table 6.1: The GSISF-wDF algorithm: Gaussian SISF for nonlinear systems with direct feedthrough.

| initialize: | $k = 0$, $\hat{\xi}_{0|0} = E(\xi_0)$, $P^\xi_{0|0} = p_0I$, where $p_0 > 0$, typically a large positive value |
| repeat | $k \leftarrow k + 1$ |
| Prediction: | project the state ahead via (6.1) |
| | compute the prediction error covariance via (6.2) |
| Update: | build $\hat{\xi}_{k|k-1}$ via (6.9)-(6.10) based on Assumption 6.2 |
| | predict the output via (6.4) with the associated covariance via (6.5) |
| | compute the cross-covariance via (6.6) |
| | perform state update via (6.7) |
| | compute the estimation error covariance via (6.8) |
| until | no more measurements arrive |

The GSISF-wDF algorithm is conceptually useful and provides a framework for understanding the SISF problem from Bayesian Gaussian perspective. We will turn it into an implementable algorithm using the Monte Carlo-based ensemble approach.

For the prediction in (6.1)-(6.2), let us represent $p(\xi_{k-1}|\bar{Y}_{k-1})$ with an ensemble of samples, $\left\{ \hat{\xi}^i_{k-1}, i = 1, 2, \cdots, N_s \right\}$. Then the samples for $p(x_k|\bar{Y}_{k-1})$ are $\left\{ \hat{x}^i_{k|k-1}, i = 1, 2, \cdots, N_s \right\}$, where

$$\hat{x}^i_{k|k-1} = f \left( \hat{\xi}^i_{k-1|k-1} \right) + w^i_{k-1},$$

(6.11)

where $\left\{ w^i_{k-1} \right\}$ are drawn from the Gaussian distribution with zero mean and covariance $Q_{k-1}$. Based on (6.1)-(6.2) and (2.23), $\hat{x}_{k|k-1}$ and $P^{xx}_{k|k-1}$ can be numerically evaluated by

$$\hat{x}_{k|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{x}^i_{k|k-1},$$

(6.12)
\[ P^x_{k|k-1} = \frac{1}{N} \sum_{i=1}^{N} \hat{x}^i_{k-1} \hat{x}^i_{k-1}^T - \hat{x}_{k|k-1} \hat{x}_{k|k-1}^T. \] (6.13)

We draw some random points, using Assumption 6.2, to characterize \( p(u_k) \), \( \{\bar{u}^i_k, i = 1, 2, \cdots, N_s\} \). They are combined with \( \{\hat{x}^i_{k|k-1}\} \) to build a new ensemble, \( \{\hat{\xi}^i_{k|k-1}\} \) following (6.9). Then \( p(y_k|Y_{k-1}) \) can be represented by \( \{\hat{y}^i_{k|k-1}\} \), where

\[ \hat{y}^i_{k|k-1} = h(\hat{\xi}^i_{k|k-1}) + v^i_k. \] (6.14)

The ensemble \( \{v^i_k\} \) in above is generated by the Gaussian distribution with zero mean and covariance \( R_k \). From (6.4)-(6.5), it follows that

\[ \hat{y}_{k|k-1} = \frac{1}{N} \sum_{i=1}^{N} \hat{y}^i_{k|k-1}, \] (6.15)

\[ P^y_{k|k-1} = \frac{1}{N} \sum_{i=1}^{N} \hat{y}^i_{k|k-1} \hat{y}^i_{k|k-1}^T - \hat{y}_{k|k-1} \hat{y}_{k|k-1}^T. \] (6.16)

Based on (6.6), the cross-covariance between \( \xi_k \) and \( y_k \) is given by

\[ P^{\xi y}_{k|k-1} = \frac{1}{N} \sum_{i=1}^{N} \hat{\xi}^i_{k|k-1} \hat{y}^i_{k|k-1}^T - \hat{\xi}_{k|k-1} \hat{y}_{k|k-1}^T, \] (6.17)

where

\[ \hat{\xi}_{k|k-1} = \frac{1}{N} \sum_{i=1}^{N} \hat{\xi}^i_{k|k-1}. \]

Then the ensemble \( \{\xi^i_{k|k-1}\} \) into \( \{\xi^i_{k}\} \) by (6.7) to represent \( p(x_k|Y_k) \):

\[ \xi^i_{k} = \xi^i_{k|k-1} + P^{\xi y}_{k|k-1} \left( P^y_{k|k-1} \right)^{-1} (y_k - \hat{y}^i_{k|k-1}), \] (6.18)

which indeed update the prediction. Aggregating the updated estimates, we have

\[ \hat{\xi}_{k} = \frac{1}{N} \sum_{i=1}^{N} \xi^i_{k}, \] (6.19)
\[
P_k^{\xi} = \frac{1}{N_s} \sum_{i=1}^{N_s} \xi_{k|k}^i \hat{\xi}_{k|k}^i \hat{\xi}_{k|k}^i \cdot \tag{6.20}
\]

Putting together the above equations, we obtain the EnSISF-wDF algorithm in Table 6.2.

**Table 6.2**: The EnSISF-wDF algorithm: Ensemble SISF for nonlinear systems with direct feedthrough.

| initialize: | \(k = 0\), generate the ensemble of samples \(\{\hat{\xi}_0^i, i = 1, 2, \ldots, N_s\}\) |
| repeat \(k \leftarrow k + 1\) |
| Prediction: | generate the ensemble \(\{w_{k-1}^i\}\) |
| | project the ensemble \(\{\hat{\xi}_{k-1|k}^i\}\) ahead into \(\{\hat{x}_{k|k}^i\}\) via (6.11) |
| | compute the sample mean and covariance of \(\{\hat{x}_{k|k}^i\}\) via (6.12)-(6.13) |
| Update: | generate samples \(\{\hat{u}_k^i\}\) based on Assumption 6.2 |
| | generate the ensemble \(\{v_k^i\}\) |
| | build the ensemble \(\{\hat{\xi}_{k|k}^i\}\) |
| | compute the ensemble \(\{\hat{y}_{k|k}^i\}\), its mean and covariance via (6.14)-(6.16) |
| | compute the cross-covariance via (6.17) |
| | perform the ensemble update to obtain \(\{\hat{\xi}_{k|k}^i\}\) via (6.18) |
| | compute the sample mean of \(\{\hat{\xi}_{k|k}^i\}\) and the covariance via (6.19)-(6.20) |
| export \(\hat{u}_{k|k}\) and \(\hat{x}_{k|k}\) from \(\hat{\xi}_{k|k}\) |
| until no more measurements arrive |

**Remark 6.1.** Development of the EnSISF-wDF algorithm depends on Assumption 6.2. It can be difficult to draw samples for the completely or partially unknown input, which would further limit the estimation performance of EnSISF-wDF. To improve we can introduce an iterative extension of the update procedure. That is, (6.14)-(6.18) are iteratively implemented — the ensemble \(\{\hat{\xi}_{k|k}^i\}\) obtained will be used to replace \(\{\hat{\xi}_{k|k-1}^i\}\) in the next iteration and repeat the update procedure.
6.2.2 Ensemble SISF for Nonlinear Systems without Direct Feedthrough

Consider the nonlinear system without direct feedthrough in (4.19). The problem now is to estimate \( u_{k-1} \) and \( x_k \) or \( \sigma_k \) from \( Y_k \) for each \( k \). To build an ensemble approach, we assume that \( \{ u_k \} \) is modeled as a white Gaussian process. The initial distribution of \( u_k \) has mean \( \bar{u} \) and covariance \( \bar{P}^u \). A joint Gaussian distribution is also assumed for \( \sigma_k \) and \( y_k \) conditioned on \( Y_{k-1} \), that is,

\[
p(\begin{bmatrix} \sigma_k \\ y_k \end{bmatrix} | Y_{k-1}) = \mathcal{G}\left( \begin{bmatrix} \sigma_k \\ y_k \end{bmatrix} ; \begin{bmatrix} \hat{\sigma}_k | k-1 \\ \hat{y}_k | k-1 \end{bmatrix}, \begin{bmatrix} P^{\sigma | k-1} & P^{\sigma y | k-1} \\ P^{\sigma y | k-1} & P^y | k-1 \end{bmatrix} \right),
\]

where \( \hat{x}_{k|k-1} \) in \( \hat{\sigma}_{k|k-1} \) is constructed based on the prior of \( u_k \) and \( \hat{y}_{k|k-1} \) is a mapping of \( \hat{x}_{k|k-1} \).

We follow similar lines to the development of the EnSISF-wDF algorithm to build the algorithm for the system in (4.19). To begin with, suppose that there is an ensemble, \( \{ \hat{x}_{k-1|k-1}^i, i = 1, 2, \cdots, N_s \} \), to represent \( p(x_{k-1} | Y_{k-1}) \) and generate an ensemble of samples for \( u_{k-1} \), \( \{ \bar{u}_{k-1}^i, i = 1, 2, \cdots, N_s \} \), based on its initial distribution. Then the ensemble for \( p(x_k | Y_{k-1}) \), \( \{ \hat{x}_{k|k-1}^i \} \), can be generated with

\[
\hat{x}_{k|k-1}^i = f(\hat{x}_{k-1|k-1}^i, \bar{u}_{k-1}^i) + w_{k-1}^i. \quad (6.21)
\]

It can be merged with the ensemble for \( u_{k-1} \) to form the ensemble for \( \sigma_k \) by

\[
\hat{\sigma}_{k|k-1}^i = \begin{bmatrix} \bar{u}_{k-1}^i \\ \hat{x}_{k|k-1}^i \end{bmatrix}, \quad (6.22)
\]

\[
\hat{\sigma}_{k|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\sigma}_{k|k-1}^i. \quad (6.23)
\]

Meanwhile, let prediction be made about the output to generate the ensemble...
\{ \hat{y}_{k|k-1}^i \} \text{ with }
\hat{y}_{k|k-1}^i = h(\hat{x}_{k|k-1}^i) + v_k^i, \tag{6.24}
\hat{y}_{k|k-1} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{y}_{k|k-1}^i. \tag{6.25}

Its auto-covariance and cross-covariance with \( \sigma \) are expressed as

\[ P_{k|k-1}^{\sigma_k} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{y}_{k|k-1}^i \hat{y}_{k|k-1}^i - \hat{y}_{k|k-1} \hat{y}_{k|k-1}^\top, \tag{6.26} \]
\[ P_{k|k-1}^{\sigma_y} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\sigma}_{k|k-1}^i \hat{y}_{k|k-1}^i - \hat{\sigma}_{k|k-1} \hat{y}_{k|k-1}^\top. \tag{6.27} \]

The a priori ensemble \( \{ \hat{\sigma}_{k|k-1}^i \} \) can be updated to the a posteriori one \( \{ \hat{\sigma}_{k|k}^i \} \) by

\[ \hat{\sigma}_{k|k}^i = \hat{\sigma}_{k|k-1}^i - P_{k|k-1}^{\sigma_y} \left( P_{k|k-1}^{\sigma_k} \right)^{-1} (y_k - \hat{y}_{k|k-1}^i). \tag{6.28} \]

Then the updated estimate of \( \sigma_k \) and the approximate estimation covariance are given by

\[ \hat{\sigma}_{k|k} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\sigma}_{k|k}^i, \tag{6.29} \]
\[ P_{k|k}^{\sigma} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\sigma}_{k|k}^i \hat{\sigma}_{k|k}^\top - \hat{\sigma}_{k|k} \hat{\sigma}_{k|k}^\top. \tag{6.30} \]

Collecting the above equations, we can obtain EnSISF-w/oDF algorithm for the system in (4.19), as shown in Table 6.3.

### 6.3 Ensemble SISS

We study the ensemble approaches for SISS in this section. Forward-backward smoothing is considered here. That is, there are two passes through time. The first one is made of forward filtering from time 1 to \( N \) by running the EnSISS-wDF or EnSISS-w/oDF algorithm. The backward pass is then run from
initialize: $k = 0$, generate the ensemble of samples $\{\hat{x}_{0|0}^i, i = 1, 2, \ldots, N_s\}$

repeat

$k \leftarrow k + 1$

Prediction:

generate the ensemble $\{\bar{u}_{k-1|k-1}^i\}$

generate the ensemble $\{w_{k-1}^i\}$

project the state ensemble ahead to obtain $\{\hat{x}_{k|k-1}^i\}$ via (6.21)

build the ensemble $\{\hat{\sigma}_{k|k-1}^i\}$ and compute its mean via (6.22)-(6.23)

Update:

generate the ensemble $\{v_k^i\}$

compute the ensemble $\{\hat{y}_{k|k-1}^i\}$ and its mean (6.24)-(6.25)

compute the covariance of $\{\hat{y}_{k|k-1}^i\}$ (6.26)

compute the cross-covariance between $\{\hat{\sigma}_{k|k-1}^i\}$ and $\{\hat{y}_{k|k-1}^i\}$ via (6.27)

perform the update to obtain $\{\hat{\xi}_{k|k}^i\}$ via (6.28)

compute the sample mean of $\{\hat{\xi}_{k|k}^i\}$ and its covariance via (6.29)-(6.30)

export $\hat{u}_{k|k}$ and $\hat{x}_{k|k}$ from $\hat{\sigma}_{k|k}$

until no more measurements arrive

$N - 1$ to 1 to derive smoothed estimates.

### 6.3.1 Ensemble SISS for Nonlinear Systems with Direct Feedthrough

Let us consider nonlinear systems with direct feedthrough in (4.1) in the first place. The pdf of interest is $p(\xi_k|Y_N)$, which can be further expressed as

$$p(\xi_k|Y_N) = \int p(\xi_k|x_{k+1}, Y_N)p(x_{k+1}|Y_N)dx_{k+1}$$

$$= \int p(\xi_k|x_{k+1}, Y_k)p(x_{k+1}|Y_N)dx_{k+1}. \quad (6.31)$$

The following Gaussian distribution assumptions are made:
Assumption 6.3. $p \left( \begin{bmatrix} \xi_k^T \\ x_{k+1}^T \end{bmatrix} \mid Y_k \right)$ is a Gaussian distribution:

$$p \left( \begin{bmatrix} \xi_k \\ x_{k+1} \end{bmatrix} \mid Y_k \right) = G \left( \begin{bmatrix} \xi_k \\ x_{k+1} \end{bmatrix} ; \hat{\xi}_{k|k} \hat{x}_{k+1|k}, \begin{bmatrix} P_{k,k+1|k}^\xi \\ P_{k,k+1|k}^x \end{bmatrix} \right). \quad (6.32)$$

Assumption 6.4. $p(x_{k+1} \mid Y_N)$ is a Gaussian distribution with mean $\hat{x}_{k+1|N}$ and covariance $P_{k+1|N}^x$.

Based on Lemma A.3, Assumption 6.32 indicates that $p(\xi_k \mid x_{k+1}, Y_k)$ is a Gaussian distribution. Specifically,

$$p(\xi_k \mid x_{k+1}, Y_k) = G \left( \xi_k, \bar{\xi}_k(x_{k+1}), \bar{P}_k \right), \quad (6.33)$$

where

$$\bar{\xi}_k(x_{k+1}) = \hat{\xi}_{k|k} + P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1} (x_{k+1} - \hat{x}_{k+1|k}),$$

$$\bar{P}_k = P_k^x - P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1} P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1} P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1} P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1} P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1}.$$

Since both $p(\xi_k \mid x_{k+1}, Y_k)$ and $p(x_{k+1} \mid Y_N)$ are Gaussian, the Gaussianity of $p(\xi_k \mid Y_N)$ can be seen from (6.31) and Lemma A.3. It can also be decided using Lemma A.3 that the mean and covariance of $p(\xi_k \mid Y_N)$, or equivalently, the smoothed estimate of $\xi_k$ given $Y_N$ and the smoothing covariance, are given by

$$\hat{\xi}_{k|N} = \hat{\xi}_{k|k} + P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1} (\hat{x}_{k+1|N} - \hat{x}_{k+1|k}), \quad (6.34)$$

$$P_{k|N}^x = P_k^x + P_{k,k+1|k}^x \left( P_{k+1|k}^x \right)^{-1} (P_{k+1|N}^x - P_{k+1|k}^x) \left( P_{k+1|k}^x \right)^{-1} (P_{k+1|N}^x - P_{k+1|k}^x)^\top. \quad (6.35)$$

The essence of the ensemble-based approach is to represent the pdf’s by ensembles of point samples and characterize the forward or backward evolution of the pdf’s through time by the change of the ensemble. Suppose that the ensemble $\{\hat{x}_{k+1|N}^i, i = 1, 2, \cdots, N_s\}$ is available for $p(x_k \mid Y_N)$ and $\{\hat{\xi}_{k|k}^i\}$ and $\{\hat{x}_{k+1|k}^i\}$ have been stored in the filtering process. It is also understood that $\hat{\xi}_{k|k}, \hat{x}_{k+1|k}$ and
and $P^x_{k,k+1|k}$ have been recorded.

The cross-covariance $P^\xi_{k,k+1|k}$ can be calculated using

\[
P^\xi_{k,k+1|k} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\xi}^i_{k|k} \hat{x}^i_{k+1|k} - \hat{\xi}_{k|k} \hat{x}^\top_{k+1|k}.
\] (6.36)

Based on (6.34), the ensemble for $p(\xi_k|Y_N)$, $\{\hat{\xi}^i_{k|N}\}$, will be given by

\[
\hat{\xi}^i_{k|N} = \hat{\xi}^i_{k|k} + P^\xi_{k,k+1|k} (P^x_{k+1|k})^{-1} (\hat{x}^i_{k+1|N} - \hat{x}^i_{k+1|k}).
\] (6.37)

The smoothed estimate can then be obtained via aggregating the ensemble, i.e.,

\[
\hat{\xi}_{k|N} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\xi}^i_{k|N}.
\] (6.38)

The smoothing error covariance is

\[
P^\xi_{k|N} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\xi}^i_{k|N} \hat{\xi}^i_{k|N} - \hat{\xi}_{k|N} \hat{\xi}_{k|N}^\top.
\] (6.39)

The above equations show the scheme for backward recursive updating. They form the EnSISS–wDF algorithm for the system in (4.1), as summarized in Table 6.4.

6.3.2 Ensemble SISS for Nonlinear Systems without Direct Feedthrough

Let us consider the nonlinear systems without direct feedthrough in (4.19). The backward updating of $p(\sigma_k|Y_N)$ is governed by

\[
p(\sigma_k|Y_N) = \int p(\sigma_k|\sigma_{k+1}, Y_N) p(\sigma_{k+1}|Y_N) d\sigma_{k+1}.
\] (6.40)
Table 6.4: The EnSISS-wDF algorithm: ensemble SISS for nonlinear systems with direct feedthrough.

Forward filtering:
for $k = 1$ to $N$
    compute filtered input and state estimate via the EnSISF-wDF algorithm
end for

Backward smoothing:
for $k = N - 1$ to $1$
    compute the cross-covariance between $\{ \hat{\xi}_i^k | k \}$ and $\{ \hat{x}_i^{k+1} | k \}$ via (6.36)
    compute the ensemble $\{ \hat{\xi}_k | N \}$ via (6.37)
    compute the sample mean of $\{ \hat{\xi}_i^k | N \}$ and its covariance via (6.38)-(6.39)
    export $\hat{u}_k | N$ and $\hat{x}_k | N$ from $\hat{\xi}_k | N$
end for

Assumption 6.5. $p \left( \begin{bmatrix} \sigma_k \top & \sigma_{k+1} \top \end{bmatrix} \top | Y_k \right)$ is a Gaussian distribution:

$$p \left( \begin{bmatrix} \sigma_k \\ \sigma_{k+1} \end{bmatrix} | Y_k \right) = G \left( \begin{bmatrix} \sigma_k \\ \sigma_{k+1} \end{bmatrix} ; \begin{bmatrix} \hat{\sigma}_k | k \\ \hat{\sigma}_{k+1} | k \end{bmatrix}, \begin{bmatrix} P^\sigma_{k|k} & P^\sigma_{k,k+1|k} \\ (P^\sigma_{k,k+1|k}) \top & P^\sigma_{k+1|k} \end{bmatrix} \right). \quad (6.41)$$

Assumption 6.6. $p (\sigma_{k+1} | Y_N)$ is a Gaussian distribution with mean $\hat{\sigma}_{k+1|N}$ and covariance $P^\sigma_{k+1|N}$.

By analogy to the development presented in Section 5.3, we have the smoothed estimate

$$\hat{\sigma}_{k|N} = \hat{\sigma}_{k|k} + P^\sigma_{k,k+1|k} (P^\sigma_{k+1|k})^{-1} (\hat{\sigma}_{k+1|N} - \hat{\sigma}_{k+1|k}) , \quad (6.42)$$

with the smoothing covariance

$$P_{k|N} = P^\sigma_{k|k} + P^\sigma_{k,k+1|k} (P^\sigma_{k+1|k})^{-1} (P^\sigma_{k+1|N} - P^\sigma_{k+1|k}) \cdot (P^\sigma_{k+1|k})^{-1} (P^\sigma_{k,k+1|k}) \top . \quad (6.43)$$

With the ensembles $\{ \hat{\sigma}_i^k | k \}$ and $\{ \hat{\sigma}_i^{k+1} | k \}$, the cross-covariance $P^\sigma_{k,k+1|k}$ is
given by

$$P_{k,k+1|k} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\sigma}_k^i \hat{\sigma}_k^i \mathbf{1} - \hat{\sigma}_k \hat{\sigma}_k \mathbf{1}.$$  \hspace{1cm} (6.44)$$

Following (6.42), the ensemble \( \{ \hat{\sigma}_k^i|N \} \) can then be produced to represent \( p(\sigma_k|Y_N) \) through

$$\hat{\sigma}_k^i = \hat{\sigma}_k^i + P_{k,k+1|k} (P_{k+1|k})^{-1} (\hat{\sigma}_k^i|N - \hat{\sigma}_k^i|k).$$  \hspace{1cm} (6.45)$$

We can aggregate the ensemble to generate the sample mean and the covariance, that is,

$$\hat{\sigma}_{k|N} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\sigma}_k^i,$$  \hspace{1cm} (6.46)$$

$$P_{k|N} = \frac{1}{N_s} \sum_{i=1}^{N_s} \hat{\sigma}_k^i \hat{\sigma}_k^i \mathbf{1} - \hat{\sigma}_k \hat{\sigma}_k \mathbf{1}.$$  \hspace{1cm} (6.47)$$

The EnSISS-w/oDF algorithm, which is formally presented in Table 6.5, combines the above equations and provides the means to realize ensemble SISS for the system in (4.19).

Table 6.5: The EnSISS-w/oDF algorithm: ensemble SISS for nonlinear systems with direct feedthrough.

```plaintext
Forward filtering:
for k = 1 to N do
    compute filtered input and state estimate via the EnSISF-w/oDF algorithm
end for

Backward smoothing:
for k = N - 1 to 1 do
    compute the cross-covariance between \( \{ \hat{\sigma}_k^{i|k} \} \) and \( \{ \hat{\sigma}_k^{i|k+1} \} \) via (6.44)
    compute the ensemble \( \{ \hat{\sigma}_k^{i|N} \} \) via (6.45)
    compute the sample mean of \( \{ \hat{\sigma}_k^{i|N} \} \) and its covariance via (6.46)-(6.47)
    export \( \hat{u}_{k-1|N} \) and \( \hat{x}_{k|N} \) from \( \hat{\sigma}_{k|N} \)
end for
```
6.4 Conclusion

We have extended the results in Chapters 4-5 to build ensemble-based SISE approaches. Both filtering and forward-backward smoothing were studied in this chapter. A class of ensemble SISF and SISS algorithms have been proposed for nonlinear systems with and without direct feedthrough. The construction of the algorithms was built upon a series of Gaussian distribution assumptions and the notion of representing the pdf’s by ensembles of sample points. Propagating the ensembles through time, we can obtain consistent ensemble-based description of the pdf’s and determine the best estimates of input and state. The proposed algorithms have the advantage of efficient implementation, which avoids derivation of nonlinear functions and does not require the maintaining of covariance matrices. Hence, they are suitable for large-scale high-dimensional nonlinear systems.

This chapter is based on the following papers that will be submitted:

Chapter 7

Oceanic Flow Field Reconstruction

7.1 Introduction

This chapter studies the application of the SISF and SISS algorithms proposed in Chapters 4 and 5 to oceanic flow field reconstruction. Understanding of ocean flow fields is of extreme importance in oceanography due to the fundamental role of flows in phenomena such as the transportation of nutrients, the motion of biological species in their early life, and the diffusion of contaminants and algal blooms. Technology-enabled ocean observation has been attracting intense interest nowadays [115, 116, 117]. An original ocean observing system is being developed at UC San Diego, which is based on a group of small, inexpensive, buoyancy-controlled drifters [118]. Through the system, three-dimensional flow field in an ocean domain where the drifters are deployed will be reconstructed, with the aim of helping oceanographers monitor flows of nutrients, behavior of animals, coastal circulation and pollution dispersion in the future.

A schematic illustration of the observing system is shown in Fig. 7.1a. The drifters are released at different locations in a region of the ocean, each then traveling in cycling movement patterns of submerging/surfacing (see Fig. 7.1b), with lateral motion driven by ocean currents and vertical motion regulated by buoy-
Figure 7.1: (a) The scenario for flow-field reconstruction based on submersible drifters; (b) the traveling profile (submerging/surfacing) of a drifter.

ancy. While underwater, each drifter stores a time record of its current depth, acceleration and other oceanographic quantities such as temperature and salinity. When it surfaces, it determines its geographical location and then transmits all the data via a communication satellite to a central server, where analysis and computation is performed. The acceleration is measured by on-board accelerometers, and
the position by the satellite-based Global Positioning System (GPS). It should be noted that the position measurements are intermittently available only when the drifter is on surface, because GPS signals are attenuated seriously in underwater circumstances. Yet acceleration measurements are available continuously.

### 7.2 Drifter Dynamics

Due to the independence of perpendicular components of motion, we consider a drifter’s motion along $x$-direction without loss of generality. For a drifter, the flow velocity $v(d_x, z)$ at its $x$-displacement $d_x$ is time-stationary and dependent only on its depth $z$. The dynamics of a drifter within the flow field is described in [119]:

$$m \ddot{d}_x = c \cdot \text{sign} \left( v(d_x, z) - \dot{d}_x \right) \cdot \left( v(d_x, z) - \dot{d}_x \right)^2,$$

(7.1)

where $m$ is the constant rigid mass and $c$ is the drag parameter that quantifies the resistance exercised on the drifter in the flow field. The right hand side of (7.1) represents the drag force that acts opposite to the relative motion of the drifter moving with respect to the surrounding flow.

From (7.1), we define two state variables $x_1 := d_x$ and $x_2 := \dot{d}_x$. Further, $v(d_x, z)$ can be viewed as the unknown external input into the drifter, naturally implying the definition of $u := v(d_x, z)$. Then (7.1) can be rewritten as,

$$\begin{align*}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= \frac{c}{m} \cdot \text{sign} (u - x_2) \cdot (u - x_2)^2.
\end{align*}$$

(7.2)

Its discrete-time representation via finite difference is

$$\begin{align*}
x_{1,k+1} &= x_{1,k} + T \cdot x_{2,k}, \\
x_{2,k+1} &= x_{2,k} + T \cdot \frac{c}{m} \cdot \text{sign} (u_k - x_{2,k}) \cdot (u_k - x_{2,k})^2,
\end{align*}$$

(7.3)

where $u_k := u(kT)$ and $x_{i,k} := x_i(kT)$ for $i = 1, 2$. The above equation can be expressed as

$$x_{k+1} = f(x_k, u_k),$$

(7.4)
where $f$ can be determined from the context.

The motion of the drifter is characterized by an irregularly cycling submerging/surfacing pattern — it submerges and moves underwater for a certain duration, then resurfaces, and repeats the process again. No matter whether it is underwater or on the surface, the depth $z_k := z(kT)$ and acceleration $\ddot{d}_{x,k} := \ddot{d}_x(kT)$ are measurable; however, the position $d_{x,k} := d_x(kT)$ can only be measured when it is at surface. Thus irregularly sampled measurements arise as a result, with the fast

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figures/figure7.2.png}
\caption{(a) depth-width cross section of the density profile; (b) depth-width cross section of the along-front velocity field; (c) drifter traveling in cubic unidirectional flow domain using the drogues (solid circles: drifters; dashed lines: trajectory of a drifter).}
\end{figure}
Figure 7.3: Depth profile of the drifter.

one $\tau_k := \ddot{d}_{x,k}$ and slow one $\eta_k := d_{x,k}$ given by, respectively,

$$
\tau_k = \frac{c}{m} \cdot \text{sign} (u_k - x_{2,k}) \cdot (u_k - x_{2,k})^2, \\
\eta_k = x_{1,k}.
$$

(7.5)

For simplicity of notation, we rewrite (7.5) as

$$
\tau_k = \varphi(x_k, u_k), \\
\eta_k = \phi(x_k).
$$

(7.6)

Combining (7.3) and (7.5), we obtain the state space model to describe the dynamics of the drifter:

$$
S: \left\{ \begin{array}{l}
x_{k+1} = f(x_k, u_k) + w_k, \\
y_k = h(x_k, u_k) + v_k,
\end{array} \right.
$$

(7.7)

Here, when the drifter is underwater, $y_k = \tau_k$ and $h = \varphi$, when at surface, $y_k = [\tau_k^T \eta_k^T]^T$ and $h = [\varphi^T \phi^T]^T$. In addition, $w$ and $v$ are added to account for noise. They are assumed to be white Gaussian and independent of each other.
7.3 Simulation Examples

In this section, we implement the algorithms in Chapters 4, 5 and 6 to the model $S$ in (7.7) to acquire the information estimates of not only the velocities of the flow field (unknown input variable) but also the trajectory and velocity profile of the drifter (state variables).

We consider a unidirectional flow field in a cubic domain, as shown in Fig. 7.2. The cross-sectional view of the considered cubic flow domain is shown in Fig. 7.2. It has dimensions of $(0, 5.0 \times 10^4) \times (0, 80)m$. Fig. 7.2a illustrates the fronts, which are regions of strong horizontal density gradients in the ocean.
They are usually sites of strong currents and eddy formation. Under the combined influence of the density gradient and Coriolis force, an along-front velocity field is generated, as illustrated in Fig. 7.2b, where the velocities are directed downward through the plane of the paper. This flow is assumed to be stationary. Additionally, it is along a specific direction through the entire cross section with a continuous velocity and in parallel streamlines, as shown in Fig. 7.2c.

In simulation, 51 drifters are deployed uniformly in spatial scale along the width direction. Let \( m = 1.5 \text{Kg}, \ c = 0.24 \text{N} \cdot \text{s}^2/\text{m}^2 \) and \( T = 0.01 \text{s} \). Inserting the parameters into the state space drifter model in (7.7), we then apply the ISISF-wDF to each drifter. As aforementioned, the implementation is based on sequential prediction and update, but the update procedure at each time instant is computed iteratively. The number of iterations should be chosen with caution to balance the computational complexity and performance, because a large one induces heavy time consumption and a small one may lead to divergence.

Let us first examine the simulation on the drifter released at the middle point \((2.5 \times 10^4 \text{m})\). It is assumed to follow the depth profile in Fig. 7.3, though the ISISF-wDF algorithm certainly allows for other options. Fig. 7.4 makes comparisons between the actual and estimated values for \( u, x_1 \) and \( x_2 \), respectively.

**Figure 7.5:** Comparison between the true flow field and the estimated one: (a) true; (b) estimated.
It is observed that the estimated values approach fast to the true ones for all the input and state variables. Identical results are obtained for the other drifters, and thus omitted here to save space.

Putting together the input estimation results of all drifters, we can reconstruct the entire flow field. The estimated flow field is shown in Fig. 7.5b. From direct comparison with the true one in Fig. 7.5a, it is seen that the estimation delicately captures the changes of the velocities at the central area, with a high overall accuracy achieved. The simulations show the power of the proposed ISISF-wDF to provide reliable estimates for the challenging problem of flow field estimation using buoyancy-controlled drifters.

Next, we consider a three-dimensional flow field (see Fig. 7.6) with dimensions of \((-1000,1000)m \times (-1000,1000)m \times (0,30)m\). The scale is intentionally narrowed to keep the computational cost affordable, but it does not restrict application of the proposed algorithms to larger flow fields. Its space is almost fully occupied by two eddies, which are centered at \((500,500)m\) and \((-500,-500)m\), respectively. Let 20 drifters be deployed evenly along the line segment from \((-800,-1000)m\) to \((1000,800)m\). We suppose that 10 drifters are released along the diagonal line from left bottom to right top.

The forward ISISF-wDF algorithm is used in the first place to produce
Figure 7.7: Smoothing results for the drifters released at (-400,-600)m: (a) $x_1$ — displacement along $x$-direction; (b) $x_2$ — velocity along $x$-direction; (c) $u$ — flow velocity; (d) trajectory of the drifter, where the circle denotes the location where the drifter is released.

To have a look at estimation for an individual drifter, we consider the one released at (-400,-600)m and its motion in the $x$-direction. The results are shown in Fig. 7.7. Figs. 7.7a-7.7c compares the true values and the smoothed estimates of the $x$-displacement, $x$-velocity and the flow velocity, respectively. Fig. 7.7d shows the smoothed trajectory in comparison to the true one. It is seen that the smoothing algorithm exhibits high accuracy, with the estimated values agreeing...
Figure 7.8: (a) True surficial flow velocity along $x$-direction; (b) smoothed surficial flow velocity along $x$-direction; (c) True surficial flow velocity along $y$-direction; (d) smoothed surficial flow velocity along $y$-direction.

well with the truth.

Further, the estimated inputs of all drifters, which are the smoothed flow velocity data at different locations, are collected together and used to reconstruct the flow field via the tessellation-based linear interpolation. The reconstructed surficial flow velocity fields along $x$-direction and $y$-direction are compared in Fig. 7.8 with the true fields, respectively. Despite some minor differences, the reconstruction is noted to have satisfactory accuracy, showing that the proposed ISISF-wDF and ISISS-wDF algorithms are able to provide reliable input and state estimates.
Figure 7.9: (a) $xy$-trajectories of the drifters in the sea test; (b) $xyz$-trajectories of the drifters in the sea test.
Figure 7.10: (a) Estimated flow velocity along $x$-direction; (b) estimated flow velocity along $y$-direction; (c) estimated flow velocity along $z$-direction.
7.4 Sea Test Example

In this section, we perform some preliminary analysis for data collected from a sea test with the multi-drifter system. The experiment was conducted on October 1, 2014, in which 13 drifters were deployed and traveled for about 5 hours in the ocean. The ocean area covered had dimensions of $500m \times 1200m$ approximately.

The $xy$- and $xyz$- trajectories of the drifters are demonstrated in Fig. 7.9. Rather than following the assumed submerging/surfacing pattern, each drifter submerged and then kept itself at around 10m underwater through the travel. The vertical position was regulated by changing the drifter volume using a mechanical piston. The drifter’s acceleration measurements were unreliable because the accelerometers were heavily influenced by the disturbance. Yet the a priori knowledge that the acceleration was close to zero could be used in the estimation. An improvement imposed on the drifter is that it can determine its underwater position through communication with other drifters. Thus the position measurements were available at each time instant. The sampling was performed every 12 seconds.

We apply the EnSISF-wDF algorithm to each of the 13 drifters to estimate the flow velocities in $x$-, $y$- and $z$-direction over time. The estimation results for the fifth drifter were illustrated in Fig. 7.10. We see that the flow ran slowly overall. The sign change of the $x$- and $y$-velocities match the trajectory shown in the black color in Fig. 7.9a. The estimated $z$-velocity shown in Fig. 7.10c is small in magnitude as well, and the estimation can provide information about internal waves. Internal waves oscillate and propagate within a fluid medium, instead of on its surface, and are of much importance for oceanographic research [120].

7.5 Conclusion

Technology-enabled ocean monitoring has been attracting intense interest for its vital role in understanding the ocean. A major function of an oceanic flow observing system is analysis of the measurement data for reconstruction of flow fields. In this chapter, we consider assimilating the data obtained by a group of
drifters traveling along the flow to reconstruct the flow velocity field and monitor the drifter status. The SISF and SISS algorithms afore proposed were applied to achieve this objective. Their performance is verified through analysis of both simulation and experimental data.

The results presented in this chapter are based on the following papers that were published or submitted:


Chapter 8

Conclusion and Future Work

This dissertation presents studies of theoretical and applied Bayesian estimation. Centering on updating the probabilistic belief on unknown or uncertain quantities using new evidence, Bayesian analysis offers a philosophically meaningful and practically effective methodology to model-based estimation. Many estimation methods have been built or explained along this direction, and their successful application to various real-world issues has been reported in the literature. To expand the existing research, we explored some new developments and applications of Bayesian estimation in the dissertation.

Gaussian state filtering is an important family of Bayesian estimation methods. We designed a new Gaussian filter that is built upon the radial basis function approximation. The EnKF also belongs to Gaussian filtering and likely the most promising method to deal with large-scale nonlinear systems. We modified the standard version by embedding iterative optimization into each recursion, thus counteracting the effect of the system nonlinearities and enhancing the estimation accuracy. The SoC information is highly sought after in the practice of batteries due to operational safety and performance considerations. Yet real-time SoC estimation has remained a pressing challenge. We proposed a filtering-based method for multi-model adaptive SoC estimation, which can act as a reliable SoC gauge even in the presence of unknown model parameters. The great potential of multi-model battery management was also highlighted. We investigated SISE, including SISF and SISS, via taking a Bayesian approach. With the aid of the constructed
Bayesian paradigms, we developed algorithms based on iterative optimization and ensemble Monte Carlo, respectively. It was shown that our algorithms generalizes some existing methods. We further analyzed their asymptotic stability properties of linear SISF algorithms. Application of SISE to oceanic flow field reconstruction was studied. Measurement data collected by a group of drifters traversing an ocean domain was processed by the proposed algorithms to estimate the flow velocity (input) and monitor the each drifter’s underwater motion status (state). This study was part of an effort with Scripps Institution of Oceanography to build an ocean observing system based on the small, inexpensive drifters.

This dissertation establishes some useful principles and tools for Bayesian estimation and demonstrates interesting applications. There are a few potential directions for future research.

First, the core challenge facing nonlinear Bayesian estimation stems from nonlinear transformation of random variables. The question is how to determine the probability distribution of nonlinearly transformed random variables. A full understanding still has a long way to go despite a large amount of research in the past. The available methods either assume certain densities for the unknown quantities, e.g., Gaussian, or uses sample ensembles to characterize the transformation, or only concerns the first- and second-order statistical properties (mean and covariance). However, none of them is widely accepted as the best solution in terms of both accuracy and efficiency. Future work can be devoted to improving the current results and seeking novel treatments.

Second, the joint input and state observability for nonlinear systems is crucial for SISE. However, no investigation has not been reported regarding this problem. It would be of great interest to derive the conditions for the joint observability as future work.

Finally, we are very interested in extending the application range of the algorithms proposed in the dissertation. For example, the flow field reconstruction may benefit from incorporation of an ocean model into the data analysis. In this case, a large state space will arise, so the IEnKF in Chapter 2 and ensemble SISE methods in Chapter 6 can be used. Another promising application is wildfire
spread prediction. Wildfires can cause serious damages, which stimulates real-time forecasting of fire growth in a complex ecological environment. The SISF algorithms can play a role to evaluate the fire status and uncertain disturbances such as the wind or humidity.
Lemma A.1. (Matrix inversion in block form) [121] Let a square matrix $A$ be partitioned into four blocks

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

where $A_{11}$ and $A_{22}$ are square. The inverse of $A$ is given by

$$A^{-1} = \begin{bmatrix} U^{-1} & -U^{-1}A_{12}A_{22}^{-1} \\ -A_{22}^{-1}A_{21}U^{-1} & A_{22}^{-1} + A_{22}^{-1}A_{21}U^{-1}A_{12}A_{22}^{-1} \end{bmatrix},$$

where $U = A_{11} - A_{12}A_{22}^{-1}A_{21}$.

Lemma A.2. (Woodbury matrix identity) [121] Let $A$, $C$ and $C^{-1} + DA^{-1}B$ be nonsingular matrices. Then

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}.$$
then $x$ is conditionally Gaussian with

$$p(x|y) = \mathcal{G}\left(x; x + P^x (P^y)^{-1} (y - \bar{y}), P^z - P^{xy} (P^y)^{-1} (P^{xy})^\top\right).$$

**Lemma A.4.** Let $\bar{X}_1, \bar{X}_2 > 0$. Given two Gaussian functions of $x \in \mathbb{R}^n$, their multiplication leads to another Gaussian function, i.e.,

$$\mathcal{G}(x; \bar{x}_1, \bar{X}_1) \cdot \mathcal{G}(x; \bar{x}_2, \bar{X}_2) = \lambda \cdot \mathcal{G}(x; \bar{x}, \bar{X}),$$

where

$$\bar{X} = \left((\bar{X}_1^{-1} + \bar{X}_2^{-1})^{-1},
\bar{x} = \bar{X}(\bar{X}_1^{-1} \bar{x}_1 + \bar{X}_2^{-1} \bar{x}_2),
\lambda = (2\pi)^{-\frac{n}{2}} |\bar{X}_1|^{-\frac{1}{2}} |\bar{X}_2|^{-\frac{1}{2}} |\bar{X}|\exp\left[-\frac{1}{2} (\bar{x}_1^\top \bar{X}_1^{-1} \bar{x}_1 + \bar{x}_2^\top \bar{X}_2^{-1} \bar{x}_2 - \bar{x}^\top \bar{X}^{-1} \bar{x})\right].$$

**Proof.** The proof is straightforward and thus omitted here. \hfill \Box

**Lemma A.5.** Given $x, y \in \mathbb{R}^n$ and $X, Y > 0$, the following equation holds:

$$\mathcal{G}(x; \bar{x}, \bar{X}) \cdot \mathcal{G}(x; y, Y) \propto \mathcal{G}(x; \bar{x}, \bar{X}) \cdot \mathcal{G}(y; \bar{y}, \bar{Y}),$$

where

$$\bar{X} = \left((\bar{X}^{-1} + Y^{-1})^{-1},
\bar{x} = \bar{X}(\bar{X}^{-1} \bar{x} + Y^{-1} y),
\bar{Y} = \bar{X} + Y,
\bar{y} = \bar{x}.\)$$

**Proof.** It follows from Lemma A.4 that

$$\mathcal{G}(x; \bar{x}, \bar{X}) \cdot \mathcal{G}(x; y, Y) = \lambda \cdot \mathcal{G}(x; \bar{x}, \bar{X}).$$
In above,

\[
\lambda = (2\pi)^{-\frac{n}{2}} |\bar{X}|^{-\frac{1}{2}} |\bar{Y}|^{-\frac{1}{2}} |\bar{X}|^{\frac{1}{2}} \exp \left[ -\frac{1}{2} (\bar{x}^{\top} \bar{X}^{-1} \bar{x} + y^{\top} \bar{Y}^{-1} y - \bar{x}^{\top} \bar{X}^{-1} \bar{x} - y^{\top} \bar{Y}^{-1} y) \right]
\]

\[
= (2\pi)^{-\frac{n}{2}} |X|^{-\frac{1}{2}} |Y|^{-\frac{1}{2}} |X|^{\frac{1}{2}} \exp \left[ -\frac{1}{2} ((y - U \bar{x})(U^{-1} y - U \bar{x})) - \bar{x}^{\top} (U U^{\top}) \bar{x} \right],
\]

where

\[
U = \left( Y^{-1} - Y^{-1} \left( \bar{X}^{-1} + Y^{-1} \right)^{-1} Y^{-1} \right)^{-1}
= \bar{X} + Y,
\]

\[
V = Y^{-1} \left( \bar{X}^{-1} + Y^{-1} \right)^{-1} \bar{X}^{-1}
= (\bar{X} + Y)^{-1},
\]

\[
W = \left( \bar{X}^{-1} - \bar{X}^{-1} \left( \bar{X}^{-1} + Y^{-1} \right)^{-1} \bar{X}^{-1} \right)^{-1}
= \bar{X} + Y.
\]

It is noted that \( U V = I, (V U V)^{-1} = V^{-1} = W \). Hence,

\[
\lambda = |X|^{-\frac{1}{2}} |Y|^{-\frac{1}{2}} |\bar{X}|^{\frac{1}{2}} |X + Y|^{-\frac{1}{2}} \cdot (2\pi)^{-\frac{n}{2}} |\bar{X} + Y|^{-\frac{1}{2}} \cdot \exp \left[ -\frac{1}{2} (y - \bar{x})^{\top} (\bar{X} + Y)^{-1} (y - \bar{x}) \right]
\]

\[
\propto \mathcal{G}(y; \bar{x}, \bar{X} + Y),
\]

which concludes the proof. 

\( \square \)

**Lemma A.6.** For \( x \in \mathbb{R}^n, y \in \mathbb{R}^m, A \in \mathbb{R}^{m \times n} \) and \( \bar{X}, \bar{Y} > 0 \), the following equation holds:

\[
\mathcal{G}(x; \bar{x}, \bar{X}) \cdot \mathcal{G}(Ay; y, \bar{Y}) \propto \mathcal{G}(x; \bar{x}, \bar{X}) \cdot \mathcal{G}(y; \bar{y}, \bar{Y}),
\]
where

\[ \tilde{X} = \left( X^{-1} + A^T Y^{-1} A \right)^{-1}, \]
\[ \tilde{x} = \tilde{X} (\tilde{X}^{-1} \bar{x} + A^T Y^{-1} y), \]
\[ \tilde{Y} = A \bar{X} A^T + Y, \]
\[ \tilde{y} = A \bar{x}. \]

\textbf{Proof.} The proof can be obtained following similar lines to that of Lemma A.5. \qed

\textbf{Lemma A.7.} For \( x \in \mathbb{R}^n, y \in \mathbb{R}^m, z \in \mathbb{R}^l, A \in \mathbb{R}^{l \times n}, B \in \mathbb{R}^{l \times m} \) and \( \bar{X}, Z > 0 \), the following equation holds:

\[ \mathcal{G}(x; \bar{x}, \bar{X}) \cdot \mathcal{G}(z; Ax + By + c, Z) \propto \mathcal{G}(x; \tilde{x}, \tilde{X}) \cdot \mathcal{G}(D \sigma; \tilde{\sigma}, \tilde{\Sigma}), \]

where \( D = [-B \ I], \sigma = [y^\top \ z^\top]^\top, \) and

\[ \tilde{X} = \left( X^{-1} + A^T Z^{-1} A \right)^{-1}, \]
\[ \tilde{x} = \tilde{X} (\tilde{X}^{-1} \bar{x} + A^T Z^{-1} (D \sigma - c)), \]
\[ \tilde{\Sigma} = A \bar{X} A^T + Z, \]
\[ \tilde{\sigma} = A \bar{x} + c. \]

\textbf{Proof.} It is noted that

\[ \mathcal{G}(z; Ax + By + c, Z) = \mathcal{G}(Ax; D \sigma - c, Z). \]

By Lemma A.6, we have

\[ \mathcal{G}(x; \tilde{x}, \tilde{X}) \cdot \mathcal{G}(Ax; D \sigma - c, Z) = \mathcal{G}(x; \bar{x}, \bar{X}) \cdot \mathcal{G}(D \sigma - c; A \bar{x}, A \bar{X} A^T + Z) \]
\[ = \mathcal{G}(x; \tilde{x}, \tilde{X}) \cdot \mathcal{G}(D \sigma; A \bar{x} + c, A \bar{X} A^T + Z) \]
\[ = \mathcal{G}(x; \tilde{x}, \tilde{X}) \cdot \mathcal{G}(D \sigma; \tilde{\sigma}, \tilde{\Sigma}), \]

which concludes the proof. \qed
Lemma A.8. For the minimization problem

\[ \hat{x} = \arg \min_x (x - \bar{x})^\top \bar{X}^{-1} (x - \bar{x}) + (Tx - y)^\top Y^{-1} (Tx - y), \]

the solution is given by

\[ \hat{x} = \bar{x} + \bar{X}^\top (Y + T\bar{X}^\top)^{-1} (y - Tx), \]

with the associated covariance

\[ P = \left( \bar{X}^{-1} + T^\top Y^{-1} T \right)^{-1} = \bar{X} - \bar{X}^\top (Y + T\bar{X}^\top)^{-1} T \bar{X}. \]

Proof. The above minimization problem is equivalent to the standard maximum likelihood estimation. The derivation of its solution is available in [19, Section 5.3]. \qed
Bibliography


