12-15-2007

When Decision Meets Estimation: Theory and Applications

Ming Yang
University of New Orleans

When Decision Meets Estimation: Theory and Applications

A Dissertation

Submitted to the Graduate Faculty of the
University of New Orleans
in partial fulfillment of the
requirement for the degree of

Doctor of Philosophy
in
Engineering and Applied Science

by

Ming Yang

B.S., Peking University, 1997
M.S., Institute of Acoustics, Chinese Academy of Sciences, 2000

December 2007

Acknowledgment

This research was supported in part by ARO grant W911NF-04-1-0274, NASA/LEQSF grant (2001-4)-01, Navy through Planning Systems Contract # N68335-05-C-0382, High-Performance Networking Program of the Office of Science, U. S. Department of Energy under Contract DE-AC05-00OR22725 with UT-Battelle, LLC. and DoD DURIP program via grant W911NF-05-1-0107.

I would like to offer my deepest gratitude to my major advisor, Dr. X. Rong Li, for his continuous encouragement, timely help, and insightful suggestions during the past more than five years.

I would also thank Dr. Huimin Chen, Dr. Jing Deng, Dr. Vesselin Jilkov, and Dr. Tumulesh K.S. Solanky for serving on my thesis committee, and for their constructive and valuable comments on the dissertation. I also appreciate Dr. Stephen Lipp and Dr. Dongmin Wei for serving on my doctoral qualifying exam. Special thanks go to Dr. Chen for his long-term collaboration, advice and friendship.

In addition, I want to acknowledge all my friends and members in the Information and Systems Laboratory, especially Peng Zhang, Keshu Zhang, Zhanlue Zhao, Anwer Bash, Trang Nguyen, Ryan Pitre, Zhansheng Duan, and Sowmya Bandarupalli. With their collaboration and support, I enjoyed my research work in this pleasant working environment. Furthermore, I would like to thank all the members and staff of the Department of Electrical Engineering at the University of New Orleans for their support. Many thanks go to the University of New Orleans and Louisiana State University Systems, for their efforts made to help reconstruct my research after Hurricane Katrina.

Last but not least at all, I want to thank my family for their unselfish, endless support and love. Especially for my wife, Jifeng Ru, without her as my companion in these years, I could not imagine that I am able to accomplish this long journey. I dedicate this thesis to her.

Abstract

In many practical problems, both decision and estimation are involved. This dissertation intends to study the relationship between decision and estimation in these problems, so that more accurate inference methods can be developed.

Hybrid estimation is an important formulation that deals with state estimation and model structure identification simultaneously. Multiple-model (MM) methods are the most widely-used tool for hybrid estimation. A novel approach to predict the Internet end-to-end delay using MM methods is proposed. Based on preliminary analysis of the collected end-to-end delay data, we propose an off-line model set design procedure using vector quantization (VQ) and short-term time series analysis so that MM methods can be applied to predict on-line measurement data. Experimental results show that the proposed MM predictor outperforms two widely used adaptive filters in terms of prediction accuracy and robustness.

Although hybrid estimation can identify model structure, it mainly focuses on the estimation part. When decision and estimation are of (nearly) equal importance, a joint solution is preferred. By noticing the resemblance, a new Bayes risk is generalized from those of decision and estimation, respectively. Based on this generalized Bayes risk, a novel, integrated solution to decision and estimation is introduced. Our study tries to give a more systematic view on the joint decision and estimation (JDE) problem, which we believe the work in various fields, such as target tracking, communications, time series modeling, will benefit greatly from. We apply this integrated Bayes solution to joint target tracking and classification, a very important topic in target inference, with simplified measurement models. The results of this new approach are compared with two conventional strategies.

At last, a surveillance testbed is being built for such purposes as algorithm development and performance evaluation. We try to use the testbed to bridge the gap between theory and practice. In the dissertation, an overview as well as the architecture of the testbed is given and one case study is presented. The testbed is capable to serve the tasks with decision and/or estimation aspects, and is helpful for the development of the JDE algorithms.

Keywords: Multiple Model, Prediction of Internet End-to-End Delay, Joint Decision and Estimation, Joint Tracking and Classification, Surveillance Testbed, Wireless Sensor Network

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Many statistical inference problems in engineering can be categorized into two classes: decision and estimation. Essentially, *estimation* is used to determine a point in a *continuous* space whereas the selection of one from among *discrete* alternatives is the task of *decision*. In practice, plenty of problems in communications and radar systems have to face both aspects. However, much of past work treated them as two independent events and handled separately.

The objective of this dissertation work is trying to investigate the relationship and interaction between decision and estimation in the problems where both operations are involved. Furthermore, the purpose of the other task in the dissertation, constructing a ground vehicle surveillance testbed with multi-sensors, is two-fold: On the one hand, existing and proposed algorithms can be implemented and evaluated based on the testbed; on the other, the challenges and practical constraints brought up from the implementation might provide

more insights into the problem or directions of the research, which eventually will help the development of the testbed.

1.2 Hybrid Systems and Hybrid Estimation

1.2.1 Multiple-Model Methods

In the traditional viewpoint, estimation is concerned with the *parameters* of a mathematical model, or the *state* of a system, or a *signal* as a stochastic process, etc. In these cases, although the parameters/state/signal are uncertain, the structure of the model/system/process is always assumed known. If the estimation has to be done in the presence of structural uncertainty (unknown structure or random structural change), we may think the system to be estimated has both continuous- and discrete-valued state variables. Such a system is defined as a *hybrid system* [50]. Similarly, the associated estimation, e.g., the estimation subject to structural uncertainty, may be called *hybrid estimation* in the sense that it deals with continuous- and discrete-valued uncertainty simultaneously.

The basic idea of the *multiple-model* (MM) approach to hybrid estimation is as follows: Assume a set of models as possible candidates of the true mode; run a bank of filters, each based on a certain model in the set; and generate the overall estimates by a process based on the results of these filters [55]. Besides the classical state estimation, the MM approach also identifies the system structure by "choosing" the model(s) in effect time by time. The prevailing MM formulation can be viewed as a procedure of *model average*, e.g., instead of *model selection* (choosing the "correct" model), where a compound nonlinear model is

constructed from a set of candidate (linear or nonlinear) models.

1.2.2 Predicting Internet End-to-End Packet Delay

End-to-end packet delay of the Internet is the packet transmission delay along a *path*. An accurate end-to-end delay prediction is helpful for protocol design (e.g., [43]), network monitoring and tomography [83]. More specifically, the predicted delays can be used to dynamically determine the packet size and sending rate, to choose the optimal path (with minimal delay), and to ensure end-to-end quality of service (QoS) (e.g., [75]). Moreover, delay prediction is widely used in many realtime network applications, such as adaptive playout buffering for multimedia [32, 80], performance enhancement for VoIP applications [46], synchronization and delay deduction in video-conferencing [29], and distributed gaming.

The core work in a model-based approach to a prediction problem is to come up with the statistical relationship between the past/current and future observations. The Internet, with its distributed structure, is hard to be described by any single linear time invariant model due to its nonlinear and time varying nature, which is verified by our preliminary data analysis. Previous work based on system identification and time series analysis relies on the linear time invariance (LTI) assumption, which is not quite suitable for capturing the dynamics of the Internet. Considering the phenomena of path switching, traffic splitting and merging, etc. (for details, see [6, 18]), it is reasonable to use a set of models to represent the possible system structures due to different traffic behavior patterns and/or routing paths. In the MM framework, these system behavior patterns are referred to as system *modes*.

We develop a novel approach to predict the Internet end-to-end delay using the MM

methods [91]. The MM approach was originally designed for *state estimation*. Here it is modified for time series prediction. The major task in application of the MM methods lies in the design of the model set, intended to cover all traffic delay patterns at different times. Two key techniques are employed in the proposed model set design procedure: (a) using *short-term time series analysis*, various Auto-Regressive (AR) models are obtained; (b) via *vector quantization* (VQ), quantized AR models are used to summarize the dynamics of the system in different modes. With such a model set, the MM methods can be applied to online delay prediction. Compared with two predictors using single model based adaptive filters, the proposed MM approach improves performance in both prediction accuracy and robustness.

1.3 Joint Decision and Estimation

Although *hybrid estimation* applies decision theory in the solution, it focuses on the estimation part. For the model identification part, as we mentioned above, it is more like a model average in the prevailing MM algorithms. In many practical problems, the "hard" decision has to be made, which means there is *one and only one* "correct" model (hypothesis). Other than the composite hypothesis testing, we are also interested in an accurate parameter estimation.

For instance, a major surveillance task is to do inference of the moving targets (e.g., vehicles, people, etc.) in the surveillance region. Here *target inference* refers not only to target localization and tracking, but also to target detection and recognition. In addition, how to combine the sensed/processed data from different sensors can also be cast into a

target inference problem such as track-to-track association (determine the originality of the track) and track-to-track fusion (obtain the estimate of the common origin based on data from multiple sources). In all of these problems, decision and/or estimation (filtering) are the key elements [11, 12]. They are usually coupled, e.g., local track estimates will affect the decision on whether they have a common origin; decision on target type will affect target motion model to estimate the position and velocity.

Conventional solutions to *joint decision and estimation (JDE)* problems mainly focus on solving the problem one at a time, viz., "decision-then-estimation" or "estimation-then-decision." The "decision-then-estimation" strategy tries to first make the best decision and then do estimation based on the decision made as if it were always correct. It does not account for the possible decision errors in the estimation; on the other hand, decision is made regardless of the results of estimation. Alternatively, the problem can also be solved by "estimation-then-decision." The idea has been widely used in composite hypothesis testing, leading to the so-called *generalized likelihood ratio test (GLRT)*. The decision is made by replacing the uncertain term with the maximum likelihood estimate. Usually, the decision results using GLRT are suboptimal [25].

Due to the drawbacks in the existing approaches, we would like to have a balanced point of view on both decision and estimation rather than put more emphasis on one of them. Middleton and Esposito [66] were the first who tried to obtain a coupled design of simultaneous detection and estimation in a Bayesian framework. Fredriksen et al. [34] extended their work to multiple hypotheses case. This pioneering work was inspired by the signal extraction problem [65], and their problem setup was limited to such a prototype model. Based on a more general JDE problem formulation [54], we will focus on the design

of its Bayes risk to make a tradeoff between decision and estimation errors. This approach is optimal in the sense that the cost of decision and estimation is minimized jointly.

Joint target tracking and classification is of great importance in both ground and airborne surveillance systems. Evidentally, in many situations, keeping the track and identifying the type of the target are fundamentally linked. While many attempts at this problem have been made, there is a lack of a systematic theory to handle the problem in an efficient and unified manner. In this dissertation, the target tracking problem is solved jointly with target ID classification via the above JDE framework and the results are compared with the conventional approaches.

1.4 Thesis Outline

This thesis contains six chapters that are organized as below:

Chapter 1 presents the motivation and background of this research work.

Chapter 2 studies the packet delay of Internet end-to-end delay. Based on the analysis of the Internet end-to-end behavior and properties, a multiple-model predictor is proposed and the performance is compared with two widely-used adaptive filters.

Chapter 3 presents a general formulation of the joint decision and estimation framework [54]. A solution based on a generalized Bayes decision and estimation cost is given. A comprehensive performance index [53] is employed to evaluate the performance of both operations at the same time.

Chapter 4 applies the JDE framework of Chapter 3 to solve a joint target tracking and classification problem. The related background is introduced and measurement models for

sensors of different types are assumed. The performance of the JDE solution is compared with those of other strategies. Since there are two aspects involved, following the conventional idea, performance of decision is evaluated by probability of correct decision and estimation by root mean square errors, respectively. Meanwhile, to provide an overall impression of the performance, the comprehensive performance index is also applied in the comparison.

Chapter 5 presents the development of a ground vehicle surveillance testbed with different type of sensors. The sensors incorporated (and to be incorporated) include digital cameras, wireless cameras, Micaz motes from Crossbow, speed/range radars/scanners, and wired video cameras. The scenarios based on the testbed can be designed for different purposes, e.g., target localization/tracking, maneuver onset time detection, and image processing. One objective of the testbed is to help algorithm development in the JDE framework.

Chapter 6 summarizes the research work in the thesis and provides some further research directions.

Chapter 2

MM Prediction of Internet End-to-End Packet Delay

2.1 Introduction

It is well known that the current Internet operates on a "best effort" principle [7], which neither guarantees the quality of service (QoS), nor allocates and reserves bandwidth effectively. Even after the introduction of *IPv6* (Internet Protocol version 6), and reservation protocols such as *RSVP* (Resource Reservation Protocol), the situation has not been improved significantly, mainly due to the pervasive and enduring heterogeneity of the network. Therefore, applications running over the Internet have to adapt to a network that they cannot control.

In order to tailor an application which generates traffic load to the network and to react promptly to the changes in traffic conditions, the dynamics and statistics of end-to-end traffic streams should be studied carefully. In the current Internet, the congestion control mechanism in Transmission Control Protocol (TCP) uses packet loss as an indication

of congestion, i.e., at least one resource within the network is overloaded if one or more packets are lost. As pointed out in [61], there is a key limitation of this approach: high utilization can be achieved only with full queues, i.e., when the network operates at the boundary of congestion. However, from statistical results (e.g., refer to `www.caida.org`), 90% of the Internet traffic is TCP based; and most TCP applications have short durations but require low latency, whereas a few long-duration TCP applications which can tolerate latency generate most of the traffic. By controlling the network around the status with full queues, short-duration connections (with low-latency tolerance) will suffer unnecessary losses and queueing delays. Moreover, using loss as a congestion indicator will bring unexpected performance degradation when losses are due to other reasons (e.g., power supply, receiver sensitivity). In wireless links, this is the most likely case. In view of the above limitations, delay is considered an important complementary measure.

2.2 Problem Description

Consider a logical network shown in Figure 2.1 [89] where each *node* represents a host, a router, an application (e.g., a printer) or even a subnet. A direct connection between two nodes is called a *link*. The links are bidirectional. Note that each logical link can have a chain of physical links connected by intermediate routers. In this setting, the Internet can be modeled as an undirected graph $\mathcal{G} = (V, L)$ with nodes V and undirected links L. Such a definition of $\mathcal{G}$ implies that the locations of the nodes do not need to be considered. Each connection between two nodes without a loop is called a *path*. The nodes at the two ends of a path are *external* (or *end*) nodes and the rest of the nodes along the path are *internal*

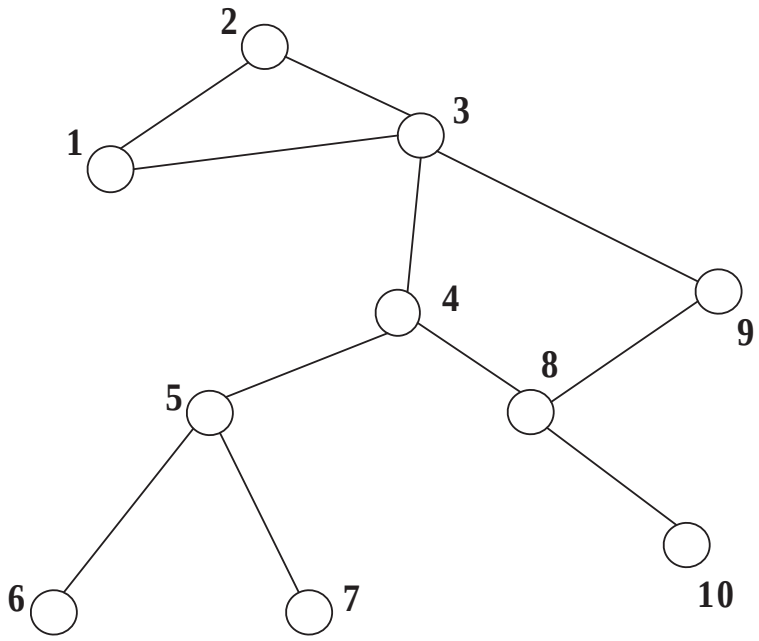


Figure 2.1: A logical network

(or *intermediate*) ones. For instance, in the path $(1, 2, 3, 4, 5, 6)$ of Figure 2.1, 1 and 6 are end nodes, whereas $2, 3, 4, 5$ are intermediate nodes. An end-to-end delay in the graph is the packet transmission delay over a path.

2.2.1 End-to-End Delay of the Internet

The concept of "end-to-end" is used as a relative comparison with "hop-by-hop." Data transmission seldom occurs only between two adjacent nodes, but via a path which may include many intermediate nodes. End-to-end delay is the sum of delays experienced at each hop from the source to the destination. The delay at each intermediate node has two components: a fixed delay which includes the transmission at the sender node and the propagation over the link to the next node, and a variable delay which includes the processing

and queueing at the sender node. The propagation delay is the delay in transmitting the data packet along a physical link. In the literature, Round Trip Time (RTT) is often used to study the Internet dynamics (e.g., [70, 72]), which requires measurements only at one end. Alternatively, One-way Transmission Time (OTT) needs the collaboration at receiver to obtain the measurements. In [28] and [73], the authors found that the mean OTTs cannot be accurately approximated by dividing RTTs in half, i.e., the variations in the OTTs are often asymmetric. However because there is no guarantee to always find collaborative nodes on the Internet, RTT is still widely used in the experimental study.

2.2.2 Introduction to Prediction Theory

If a subspace $\mathbb{M}$ of a Hilbert space $\mathbb{H}$ is used to denote the information about the past of a system, a mapping that maps to an element of $\mathbb{M}$ is called a *predictor* [74]. Therefore $\mathbb{M}$ can be viewed as the space of allowable predictors for the future among which we are to find the "best" one.

If $\hat{X}$ is the predictor for a random variable X ($X \in \mathbb{H}$) which represents the future, the prediction error is $X - \hat{X}$, and it is desirable to make this error as small as possible using certain error metric. Since $X - \hat{X}$ is a random quantity and not observable in general, it is natural to pick $\hat{X}$ so that $X - \hat{X}$ is small on the average. This can be done, for example, by choosing $\hat{X}$ so that either $\Pr\{|X - \hat{X}| \geq \epsilon\}$ or $E[|X - \hat{X}|^p]$ is small, for an appropriate value of ϵ or p.

In practice, it is desirable to develop a prediction theory that leads to simple prediction formulas and requires less statistical information than the full distribution of the past. The

linear prediction or the Kolmogorov-Wiener prediction theory (refer to Chapter 12 of [52]) provides such a setting in which only the knowledge of the past and the first two moments of the distribution of the past are required. In fact for the normal (Gaussian) distribution, the first two moments can fully determine the distribution.

It is clear that such a prediction problem is closely related to a (quality) control problem because if we can predict how a process will behave, we can adjust the process so that the achieved values are, in some sense, as close to the target value as possible. In many applications, the purpose of predicting the Internet end-to-end packet delay is to design a working mechanism so that the Internet can work more stably and more efficiently. In particular, by more accurate prediction, delay-based bandwidth allocation and congestion control can provide further improvements to QoS in heterogeneous networks.

2.2.3 Internet End-to-End Delay Prediction: Relevant Issues

Probing Strategy

Probing strategies are an important topic in the research of computer networks. Specifically, as an active approach to directly measure the Internet dynamics, the properties of probing packets should be considered. A probing packet is a unit of data sent across a network to gather information about the internal network or its end users. "Packet" is a generic term used to describe a unit of data at any layer of the Open System Interconnection (OSI) protocol stack, but here it is recommended to only describe application layer data units. There are three major factors [89] that should be considered when a probing strategy is being designed: packet size, inter-departure time, and data/packet transmission protocols.

Many software tools for actively or passively measuring network performance have been developed by researchers in the fields of computer science and networking. For example, netstat, being an important statistical tool for TCP/IP network connections, can give the current summary of the packets sent/received (with the -e option) for different protocols (with the -s option), e.g., IP, ICMP, TCP, UDP. For active probing tools, ping, pathchar [2], NetDyn [4], traceroute are all widely used for different purposes. The CAIDA (Cooperative Association for Internet Data Analysis) website [2] provides a survey of available network measurements tools.

One important issue ought to be mentioned here is that these tools usually need cooperation along the routing path, e.g., response from the nodes among the path. Furthermore, several special assumptions have to hold, e.g., symmetric routing path (forward and reverse), store-and-forward routers, nonexistence of firewalls. Being a decentralized system, the Internet has to face some uncooperative administrations and these tools might not be applicable. For example, after being attacked by the Internet worm MSBlast.D in August 2003, the servers at the University of New Orleans increased the security level of the firewall and disabled traceroute outside the firewall. In such cases, large-scale inference and Internet tomography methods [22, 30] have their special value since they can deal with uncooperative networks.

Network Tomography

Loosely speaking, *network tomography* refers to estimation of network performance parameters based on measurements from a limited subset of its nodes. [30] provides an introduction to the work in this field.

Many network tomography problems can be roughly approximated by the following linear model [30]

$$y = A\theta + \varepsilon \tag{2.1}$$

where y is a vector of measurements, e.g., packet counts or end-to-end delays; A is a routing matrix, θ is a vector of packet parameters, e.g., mean delays over a link, or the origin-destination traffic vector; ε is a noise term.

In the recent literature, network tomography has been divided into three classes: 1) the estimation of path-level network parameters from measurements made on individual links, which is so-called origin-destination (or source-destination) tomography (y in (2.1) is not known precisely), see, e.g., [84], 2) the estimation of link-level network parameters from path-level measurements (θ is not known precisely), see, e.g., [31, 83], and 3) topology identification (A is not known precisely), see, e.g., [33]. For the recent advanced topics in this area, see [22] for more details.

2.3 Existing Work

In the end-to-end delay prediction problem, similar to traffic prediction, there are two fundamental issues involved: one is the prediction method, the other is the prediction interval [68, 77]. Here the prediction interval refers to how far into the future can the network packet delay be predicted with a certain confidence (subject to a certain error constraint). The choice of a prediction method is a trade-off between the prediction error and cost. For real-time applications an effective prediction requires a low cost, otherwise the network will suffer from a heavy burden caused by extra computation and network resources.

2.3.1 Queueing Network Modeling

A queueing system can be described as customers arriving for service, waiting for service if it is not available immediately, and leaving the system after being served [36]. The term *customer* here is used in a general sense and does not imply necessarily a human customer. For example, in network modeling, a customer is a packet waiting in line to be processed. Such a basic system with a single queue can be illustrated by Figure 2.2. Using the shorthand notation introduced by Kendall (for details, refer to [36]), this is a G/G/1 queueing model.

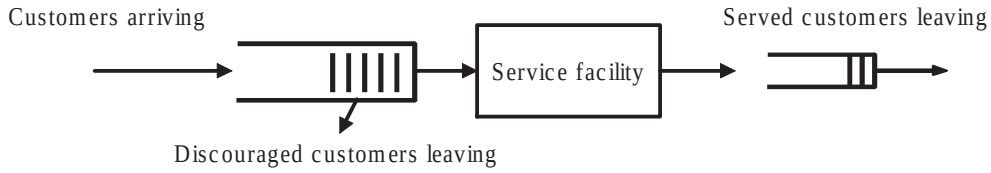


Figure 2.2: A typical queueing process

Queueing theory was developed to provide models to predict the behavior of systems that attempt to provide service for randomly arising demands. Telephone traffic load analysis is one of the earliest problem studied by it.

Queueing theory has been extensively used as a powerful tool to analyze computer and communication networks for a long time. Kleinrock [45] derived an expression for the mean end-to-end delay based on the queueing model at each link with certain assumptions which have been summarized in [86]. When these assumptions hold, queueing analysis will get several product-form solutions, e.g., the queue size distribution for an entire network of queues is equal to the product of the queue size distribution of the individual queues. With this result, several performance measures such as the average delay (queue size) and delay

distribution can be easily calculated.

Queueing network theory can be applied if the distribution at each individual link is known. This assumption might hold in a small-scale network with a few interconnected servers, but usually not for large-scale networks. Even though the distribution of each link is available, the computational cost will grow dramatically as the size of network increases. In addition, the product-form solution does not characterize some features of the real-life network such as the correlations introduced when traffic streams merge and split, the regulation of traffic by routing and flow control mechanisms, or the packet losses due to buffer overflow [18]. Due to these limitations to obtain the dynamic behavior of the networks by queueing theory, we will focus on simulation or measurement based approaches instead of such an analytical method.

2.3.2 System Identification Approach

System identification (SI) is used for building dynamic mathematical models based on the observations of the systems [60]. The three essentials in constructing models from data are: observations, candidate model sets and evaluation criteria.

Basically, for a dynamic system, we choose the observation signals we are interested in as *outputs*, the external signals which can be controlled as *inputs*, and the others as *disturbances*. Figure 2.3 is a general framework of a dynamic system [60], where y denotes the output, u the input, and w the disturbance. If we regard the Internet as a dynamic system, obviously, end-to-end delays (or round-trip times) should be our output y. The input u could be the sending rate (or other parameters, e.g., interdepature times) of the probing packets, which

describes the effect of the probing packets. And the disturbance w accounts for effects from other traffic (i.e., packets coming from other hosts), usually modeled as white Gaussian noise (WGN).

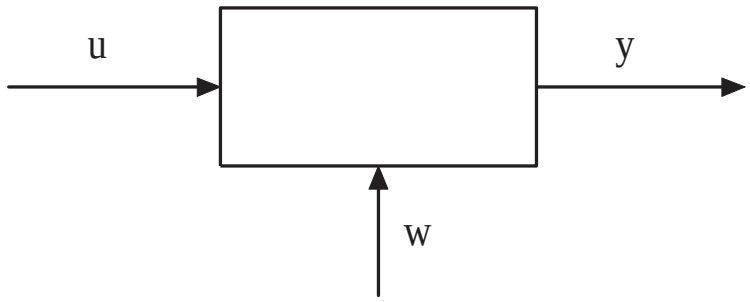


Figure 2.3: A system - output y, input u, disturbance w

Then we select a candidate model set and we determine the "most" suitable one in the set for the system based on the observed data. By choosing a model fitting criterion, we can compare the models in the set to find the one that best fit the measured data. At last we do *model validation*, usually based on "fresh" (new) data, to decide whether the "best" model is good enough for fitting the new observations for our purpose.

For either prediction or control purpose, the core work is to identify a model for the system (or process) given the observations on the input and output of the system. Although recently there has been an increased interest in time-varying and non-linear systems, much of the literature assumes that the system can be adequately approximated over the range of interest by a linear model whose parameters do not change with time. For instance, in [70] and [69], the authors used Auto-Regressive eXogenous (ARX) models for the delay dynamics.

As shown in Figure 2.4, a discrete-time ARX model is defined as

$$A(q)y(k) = B(q)u(k) + e(k) \tag{2.2}$$

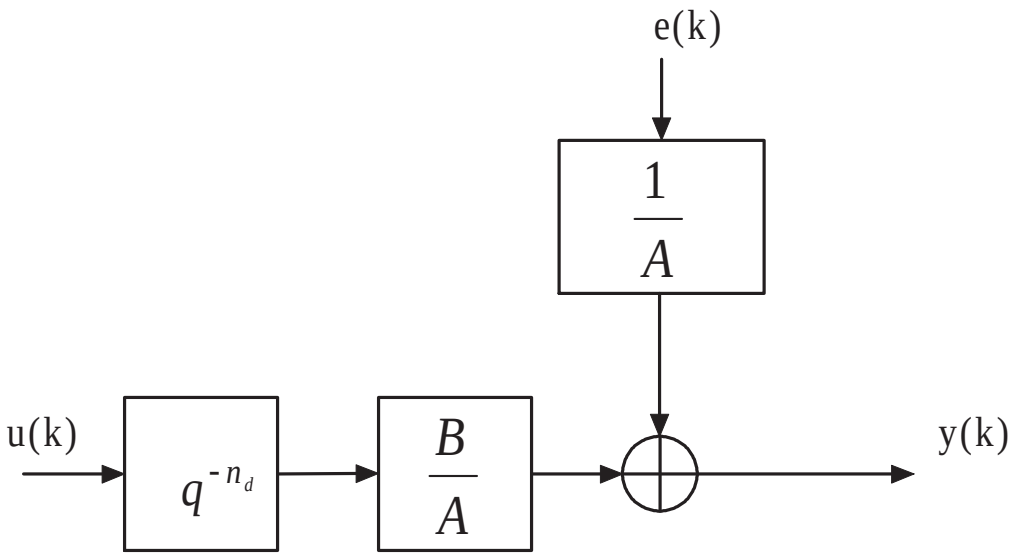


Figure 2.4: ARX model structure

where $A(q)$ and $B(q)$ are given by

$$A(q) = 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a}$$

$$B(q) = 1 + b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b}$$

Here $e(k)$ is unmeasurable disturbance (i.e., white noise), and q^{-1} is the delay operator; i.e., $q^{-1}f(k) = f(k-1)$. The numbers n_a and n_b are the orders of polynomials. The number n_d corresponds to delays from the input to the output. In such a case, the adjustable parameter is

$$\theta = [a_1, a_2, ..., a_{n_q}, b_1, b_2, ..., b_{n_b}]^T \tag{2.3}$$

The "AR" in ARX models stands for the autoregressive part $A(q)$ and $e(k)$, and "X"denotes the external input $B(q)u(k)$.

There are two basic methods for model fitting on observations [60]. One is the so-called *Prediction-Error Identification Method* (PEM), which minimizes the prediction error series.

It contains famous methods such as Least Squares (LS), Maximum Likelihood Estimation (MLE), and Maximum A Posteriori (MAP) estimation, MMSE, etc. To implement these methods, *adaptive filtering* techniques can be applied. The other one is the *Correlation Approach*. Its major difference from PEM is that it does not assume prediction error is independent with the past data. Its typical methods include Instrumental-Variable method (IV) and rational transfer function model, etc. We can regard PEM as a special case of the Correlation Approach. For the problem addressed here, PEM will be emphasized.

2.3.3 Time Series Approach

A time series $y(t)$ is a set of observations ordered sequentially in time [63]. A series of T observations can be viewed as a random process of the variables $y_1, y_2, ..., y_T$, sampled at equidistant time intervals $t_1, t_2, ..., t_T$. In fact, a time series can also be treated as the output of a dynamic system whose external input cannot be observed [60].

There are two major aspects to the study of time series – analysis and modeling. The aim of analysis is to summarize the properties of a series and to characterize its salient features. This can be done either in time domain or in frequency domain. The two forms of analysis are complementary rather than competitive: The same information is processed in different ways, which provide different insights into the essence of the time series.

The main reason for modeling a time series is to predict future output values. The most distinct feature of time series modeling is that there is no attempt to build a relationship between the observations and other variables. In other words, it just uses its "own" past to predict future.

In time series approach, ARMA (Auto-Regressive Moving Average) models are widely used for prediction purpose. Most time series data of Internet end-to-end delay are non-stationary. However, ARMA models are used for stationary time series. This does not present an insolvable problem, since there are several methods which allow us to transform a non-stationary series into a stationary one. In most practical cases first and second-order differencings are sufficient to remove any kind of trend existing in a time series [63]. The ARIMA (Autoregressive Integrated Moving Average) methodology, developed by Box and Jenkins, is based on such an idea [19]. An ARMA model can be viewed as a special case of ARIMA models. From an engineer's point of view, differencings in ARIMA models act as high-pass filters on the trended data.

In addition to non-stationarity in the mean of time series, we can also have non-stationarity in variance. The latter can become stationary by transforming the data into a logarithmic scale or a fraction of a power (e.g., square root).

Recent studies have revealed a fractal-like structure of delay sequences, which may not be well suited to ARMA models [49]. In [49], the authors propose a delay-boundary prediction algorithm based on a deviation-lag function (DLF) to characterize the end-to-end delay variations. Preliminary experiments show that it has an significantly increased prediction accuracy than Jacobson's algorithm in [43], which is based on an ARMA model.

The MSEs of predictions in the ARIMA models tend to increase rapidly as the prediction interval becomes greater [39]. Thus the main value of such models is for short-term forecasting (prediction). In fact, because of routing behavior, competing traffic, and available bandwidth etc., end-to-end delays are quite dynamic and the prediction interval cannot be too large.

State-space models for a time series problem is usually arrived at through a structural analysis of its components that make up the series. These components may include trend, seasonal, cycle, together with explanatory variables, interventions, outliers and missing values. By determining the *state* of the system, the useful information for prediction can be summarized efficiently. In contrast, the ARIMA modeling is a passive *black box* approach in which model identification relies solely on the data without prior information of the system that generated the data. Which tool is more suitable for our purpose? From a control engineer's viewpoint, state-space models have more structural advantages than the ARIMA framework. But in the study of the Internet end-to-end delay, unfortunately, there is very little information to build up the state of the system due to the complexity of the networks. A trade-off between these two schemes is to find the best fit time series model by the ARIMA methods first, and then convert to the state-space representation so that prediction and control can be done more efficiently.

2.3.4 Learning and Prediction

So far the methods above are all model-based. However in many applications the principle or knowledge upon which scientific models can be built up is not available, or the systems under study are too complex to be mathematically described. In such cases, a *learning machine* can be used to do prediction. Loosely speaking, a learning machine tries to build up an unknown mapping of the system from its observed samples. Here we use the term "learning" to emphasize its data-driven nature. Artificial neural networks (NNs) are one of the most representative methods in machine learning.

NNs are computing architectures that consist of massive parallel interconnections of simple neural processors. In fact, the NN is more like a computing technique than a new model.

The NN approach can be used in system identification and prediction, that is, we can use NNs to substitute for other forms of dynamic functions (e.g., time series). For a non-linear function, it is more useful: due to the strong adaptive learning ability of NNs, usually we can obtain a good result. NNs can also be used to construct an input-output structure. For the situation where suitable models are not available, the NN approach is invaluable since it performs a task that many other approaches can not. In such a sense, we can regard the NN method as a "blind" model.

In [72], Parlos presented an empirical approach for the identification of the end-to-end delay and round trip time dynamics using recurrent NNs. Similar to the SI approach in [70], by using the packet inter-departure time as the input and the end-to-end delay variation and round trip time variation as the output, a SISO system was built for the Internet delay dynamics. The predictors were designed for multi-step-ahead accuracy within a finite horizon.

In [9] and [72], the authors claimed that their dynamic recurrent NNs as semi-parametric approximators for modeling complex systems. We regard NNs as a model-free method or a "universal" solution in the sense that it does not require any special structure of the system. Besides the computational complexity, the main drawback of this method stems from that the prior knowledge of the system structure is totally ignored.

2.4 Preliminary Data Analysis

In the literature, Round Trip Time (RTT) has often been used to study the Internet end-to-end delay dynamics (e.g., [70, 72]), which requires measurements only at one end. In [43], Jacobson designed his congestion avoidance algorithm by modeling each RTT sequence based on a certain ARMA model. Following a similar idea, we constructed an experiment to get the Internet end-to-end delay data (RTTs) and did some preliminary data analysis.

2.4.1 Data Collection

In this study we sent probing packets using Internet Control Message Protocol (ICMP) instead of TCP in [43]. More specifically, as in the ping program, the source host sends out a series of ICMP Echo Requests to the destination host, and the destination host returns ICMP Echo Reply messages. Here an ICMP Echo message is regarded as a probing packet. The original ping program sends packets with fixed time interval (one second). We modified it so that variable inter-departure times can be obtained [88]. The reason that we chose ICMP rather than TCP is the following: TCP has an embedded congestion control mechanism so that the packet inter-departure time, which is usually considered as the system input, is not independent, identically distributed (i.i.d.) in time. However this independence assumption is required in most system identification techniques. On the other hand, User Datagram Protocol (UDP) and ICMP have no feedback-based control. The packet inter-departure time of UDP and ICMP can be freely adjusted by the end users.

In our experiment, we used a common source host in the Electrical Engineering Department of the University of New Orleans. Eight different destination hosts were chosen, which

included 2 inside the LAN (without switch and with one single switch) and 6 outside the LAN (with multiple routers/switches). For each destination, we collected the RTTs as a time series using our modified ping program. Two different probing packet sizes (512 bytes and 1024 bytes) were used in the experiment, that is, for each destination there are two sets of time series data. The timeout as well as inter-departure time were set to 0.5 second. The data collection for each path lasted around 24 hours. Figure 2.5 shows one set of time series data collected from a remote destination (www.yahoo.com). Note that an RTT value of zero indicates where the packet was lost. The plot actually follows the people working pattern: the dark part (with longer delays by average) corresponds to the daytime; the flat part corresponds to the nighttime.

2.4.2 Packet Loss

Although this topic is beyond the scope of this study, as a widely-used indication of congestion, packet loss may still provide some useful information for our study. Table 2.1 [90] lists the loss rate of each set of data. The common source host is Fusion1.uno.edu. Note that the loss rate in the source-destination (SD) pair F is much higher than the others. However, this does not necessarily indicate the paths between those two hosts were more congested. It is most likely because more probing packets were cut off by this destination host as a preventive mechanism to the packet flooding from one source. This phenomenon indicates that google.com had a more secure network firewall setting than the other ones at the time the experiment was performed. The sizes of probing packets in our experiment are much larger than normal ones (32 bytes by default) since larger packets carry out more information

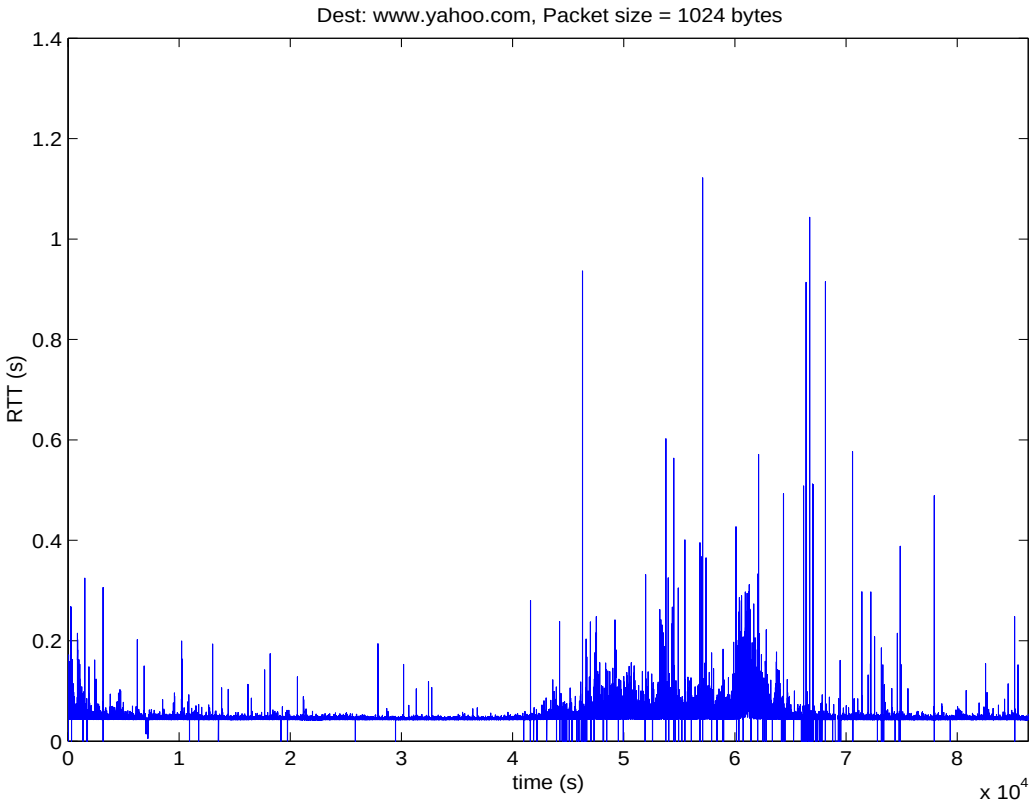


Figure 2.5: A set of end-to-end delay data: the RTT sequence collected at the same host from one particular destination

about delay (RTT might not reflect traffic delay if the packet size is too small). On the other hand, the probing packets should not be too large, otherwise they may significantly affect the traffic.

2.4.3 Round Trip Times

In raw data (refer to Figure 2.5), some RTTs are equal to zero, which correspond to the cases that the packets are lost. In our time series analysis, these points were treated as missing data and skipped. By doing this, the delay properties will not be impacted since packet delay and loss are two separate issues.

Table 2.1: Packet loss rate

Index	Target	$bytes = 512$	$bytes = 1024$
A	enee613-2000.uno.edu	< 0.0001	< 0.0001
B	www.sina.com	0.0273	0.0325
C	www.utd.edu	0.0104	0.0112
D	www.yahoo.com	0.0174	0.0176
E	www.uno.edu	0.0020	0.0017
F	www.google.com	0.1160	0.1698
G	www.wenxuecity.com	0.0081	0.0080
H	216.107.90.145	0.0110	0.0111

Test for Stationarity

A random process is (wide sense) *stationary* if it has a constant mean and the autocorrelation coefficient Function (ACF) depends only on time difference $\tau = \Delta t$ but not the absolute time t. In time series analysis, the stationarity property leads to great simplification. Therefore the first step in time series analysis is to check whether the data are stationary or not. There are two general classes of approaches for stationarity testing, parametric and nonparametric. Being a class of totally data-driven approaches, nonparametric tests are more applicable for our case than parametric ones which often require certain model assumption. On the other hand, nonparametric tests require more data than parametric ones to achieve the same statistical decision at the same confidence level.

The sample ACF is an important parametric tool to assess the stationarity of the process.

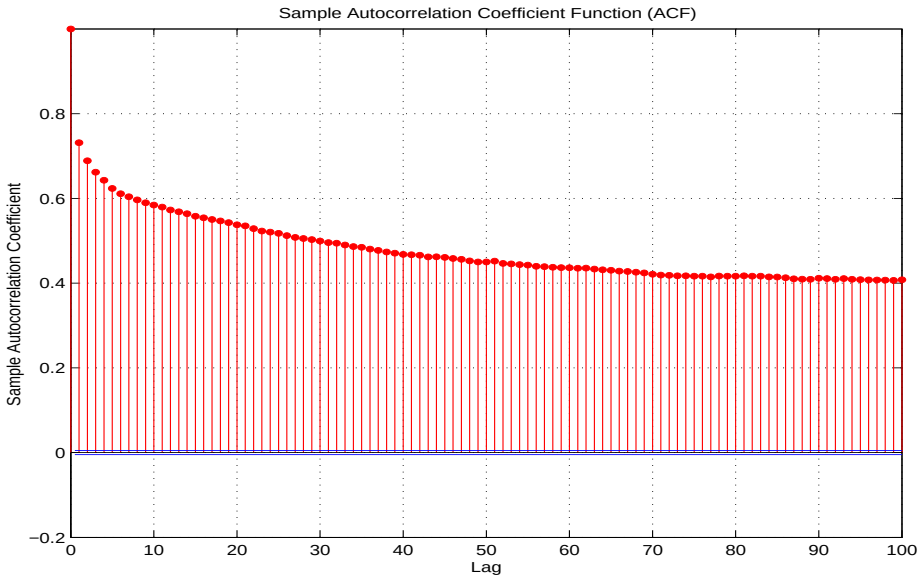


Figure 2.6: The sample ACFs of an RTT time series for the whole sequence

The sample autocovariance function (ACVF) is given by

$$c_\tau = \frac{1}{K}\sum_{i=1}^{K-\tau}(z_i - \bar{z})(z_{i+\tau} - \bar{z}), \quad \tau = 1, 2, ... \tag{2.4}$$

where K is the length of the data $\langle z_k \rangle$, $\bar{z}$ is the sample mean of the $\langle z_k \rangle$. The sample ACF is

$$r_\tau = c_\tau / c_0 \tag{2.5}$$

where $c_0 = \frac{1}{K}\sqrt{\sum_{i=1}^{K} z_i^2 \cdot \sum_{i=1}^{K-\tau} z_{i+\tau}^2}$. Figure 2.6 and 2.7 shows the sample ACF plots of a time series: Figure 2.6 was computed for the whole sequence, whereas Figure 2.7 was computed for a short time interval (60 samples). In the plots, the area between two parallel straight lines is the 95% confidence interval within which the sample values are thought close enough to zero. Typically, the sample ACF will either cut off or die down quickly if the time series is stationary. In Figure 2.6, we found that the sample ACF decays very slow, which suggests that the process is non-stationary on the mean level; however in Figure 2.7, the sample ACF cuts off at lag 3, which suggests that the process is stationary in a short term.

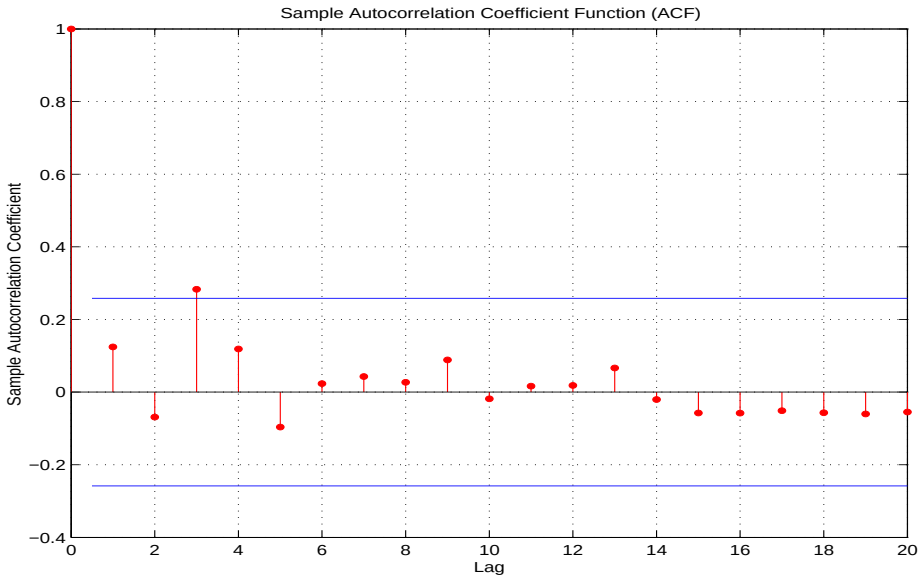


Figure 2.7: The sample ACFs of an RTT time series for a short time interval (60 samples)

However, such ACF analyses highly depend on experience. For example, under what rate the decay can be viewed as quick is not well defined. There is a lack of quantitative ways to make decision based on ACF plots. Therefore an alternative approach is preferred.

We applied [90] a widely-used nonparametric test, *runs test* [15], to check the stationarity of the time series. Consider a sequence of K observations of a random variable where each observation is classified into one of two mutually exclusive categories, denoted by plus (+) or minus (−). A most straightforward example is to flip a coin. A *run* is defined as "a sequence of identical observations that is followed and preceded by a different observation or no observation at all" [15]. For example, the sequence "+++−−+−++−−−+−++−+−−" has 12 runs. The runs test is to check whether the observations are independent with each other by counting the number of runs in the sequence and comparing to the desired level of significance under the hypothesis of independence. The sample significance intervals under

different observation sizes are given in Table A.6 of [15]. The runs test can be used to test stationarity as follows.

1. Divide the sequence into time intervals of equal length.

2. Compute a mean value for each interval.

3. Count the number of runs that the mean value in every interval is above (+) or below (−) the median of the whole sequence.

4. Compare the result to the known sample significance interval: if it is inside the interval, the time series is stationary; otherwise non-stationary.

When the runs test was applied to the whole sequence of a time series (e.g., data from yahoo.com), we divided the sequence into 100 equal intervals and the number of runs $r = 10$. Since the 95% significance interval for 100 is $[42, 59]$, the time series is **non-stationary**. To check the stationarity in shorter time ranges, we cut the whole sequence of the time series into segments with certain lengths K', and treat them as different realizations of a random process to perform runs test one by one. When the frequency f with which the number of runs r falls inside the significance interval is high enough (say, over 90%), the time series is considered stationary with time range K'.

Table 2.2: Runs test in short time ranges

Time range (samples)	30	60	90	120	$\cdots$
Frequency (%)	94.96	91.92	88.38	83.12	$\cdots$

Table 2.2 shows the testing results in different time ranges. For each runs test, the data segment was divided into 10 equal length intervals and the corresponding 95% significance interval is $[3, 8]$. From Table 2.2 we can see that f increases as the time range K' decreases. Although the time series is non-stationary in long-range, when the time range is smaller than 60 samples (i.e., half a minute), each data segment can be viewed as stationary since $f > 90\%$. In other words, the time series is short-term stationary. Based on this understanding, the time series analysis will be performed in each segment with a time range around 60 samples.

Model Order Selection in Short-Term Time Series Analysis

For the sake of the simplicity and extensity, we performed AR modeling for each data segment. One primary work in AR modeling is order selection. This problem can be formulated as a testing problem of multiple composite hypotheses. Let $\mathcal{H}_n$ denote the hypothesis that the model order is n, where n is upper bounded by N, i.e., $n \in [1, N]$. We assume that the hypotheses $\{\mathcal{H}_n\}$ are mutually exclusive meaning that only one of them can be true at one time. Note that the hypotheses $\{\mathcal{H}_n\}$ are *nested* since AR(n) includes AR(m) as a special case if $n > m$. If we only perform likelihood ratio test (or MAP with equiprobable prior), the trend of the order selection is to pick the one with the highest order, especially when the truth is not in the hypotheses. Hence there should be one penalty term which accounts for the model complexity in a "fair" model order selection rule.

Let z^K denote a vector of K independent observations, and θ_n the AR parameter vector, i.e., $\theta_n = [\ a_1 \quad \dots \quad a_n\ ]'$. Most commonly used model selection criteria can be written in a general form

$$\xi_n = -2 \log f_n(z^K | \hat{\theta}_n) + d_n(z^K) \tag{2.6}$$

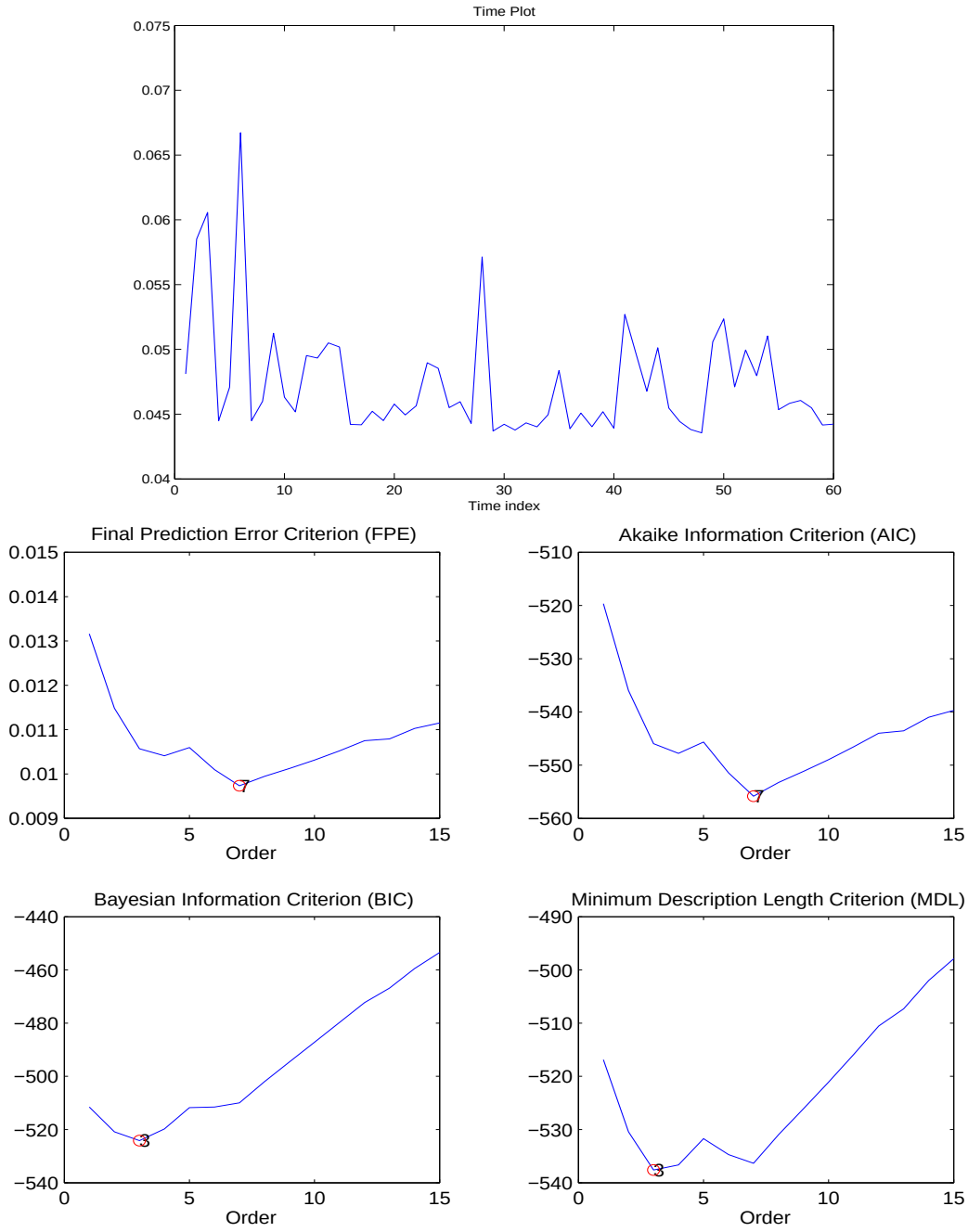


Figure 2.8: A model selection example

minimized among N hypotheses. Here log denotes the natural logarithm and $\hat{\theta}_n$ is the maximum likelihood (ML) estimate

$$\hat{\theta}_n = \arg\max_{\theta_n} f_n(z^K|\theta_n) \tag{2.7}$$

The first term of ξ_n in (2.6) is a natural extension of the generalized likelihood test (GLRT) to deal with multiple hypotheses testing. The reason we add a multiplier 2 is to cancel the factor 1/2 under Gaussian assumption. Due to the limited data and the model nesting issue, if only this term is used for model selection, we will usually have an overfit. Thus the second term $d_n(z^K)$ is used as a penalty function that varies for different criteria. The Akaike information criterion (AIC) uses $d_n(z^K) = 2n$ [5]. The Bayesian information criterion (BIC, also known as SBC – Schwartz's Bayesian Criterion) uses $d_n(z^K) = n \log K$ [78]. The minimum description length (MDL) criterion uses $d_n(z^K) = 2\log(\int f_n(z^K|\hat{\theta}_n)dz^K)$ which is interpreted as part of the normalized maximum likelihood (NML) [76]. Akaike's final prediction error (FPE) criterion uses $d_n(z^K) = 2K \log[(K+n)/(K-n)]$ [60] [1].

In general, the penalty terms have the order: FPE $\approx$ AIC $<$ BIC $\approx$ MDL. When observation number K goes to infinity, BIC (and MDL) can select the best model asymptotically with probability 1 given that the truth is in the model set [78]. FPE and AIC do not have this asymptotic property; they always overfit the data, especially when the sample size is small. Figure 2.8 shows an example in which we used different criteria to decide the AR order of a segment from the time series data.

We applied the above model selection criteria to the whole time series data to take a glance at the model structure. To satisfy the stationarity constraint, the time series data

[1] Rigorously speaking, this is the form of log(FPE). Originally, FPE was defined using prediction error variance although we can also interpret it by the idea of GLRT.

have to be segmented into small pieces to perform analysis. The segment size was chosen to have 60 samples (i.e., half a minute). Sliding windows were used in segmentation to reduce the impact due to high frequency components. A sliding window is like a stencil that you move along a data stream, exposing only a fixed number of data points at one time. Here the window size was 60 samples. The window moving step (sliding factor) was 30 samples, that is, there were $60 - 30 = 30$ samples overlapping between the adjacent segments.

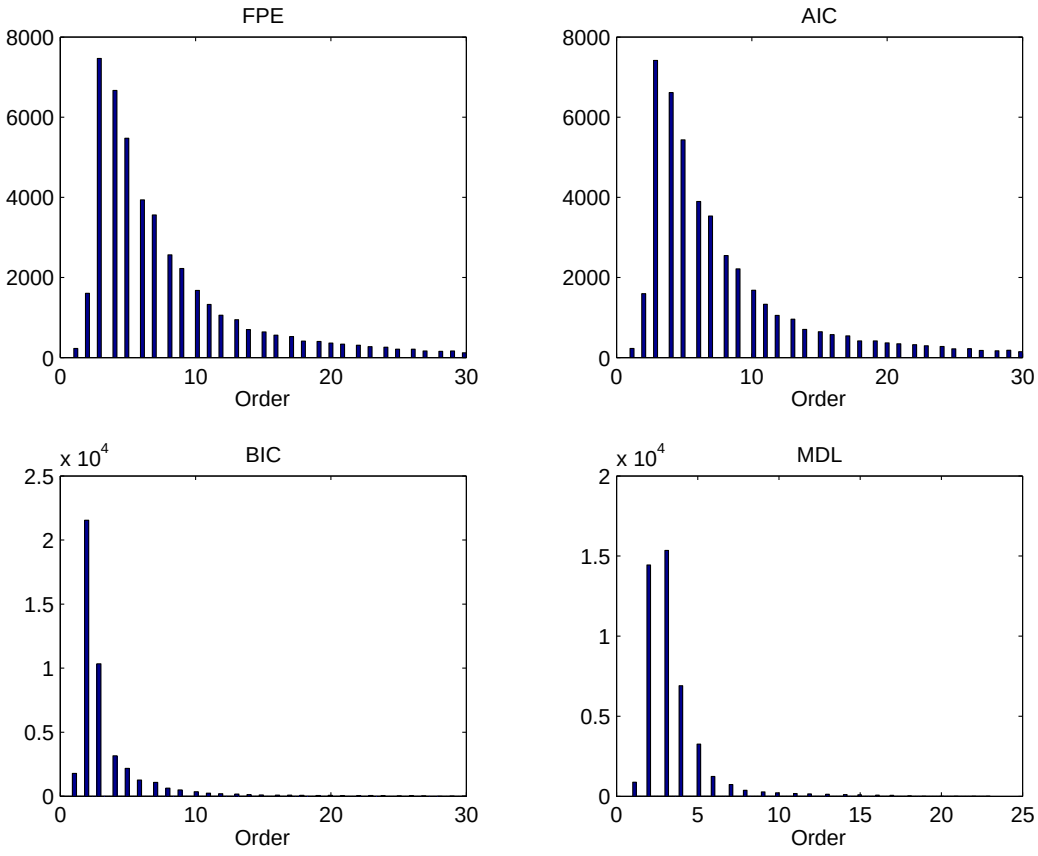


Figure 2.9: The histograms of model selection results

Figure 2.9 shows the histograms of model selection results of the full dataset. Since FPE and AIC are not consistent, we only refer to the results from BIC and MDL. According to these two criteria, about 90% of the time series segments can be approximated by AR

models with order 4 or lower. Therefore the orders of AR models in the short-term time series analysis were chosen as 4 or lower. For simplicity, the orders of all the AR models can be further approximated to be 4 (AR models of a lower order can be viewed as special cases of AR(4) models) to accomplish some batch work.

2.5 The Multiple-Model Approach

Based on the preliminary data analysis, we concluded that the Internet end-to-end data can be analyzed by AR modeling in each short data segment [90]. However, even though the structure (order) of each model can be chosen as the same, the parameters in different segments are usually different. How to use these different AR models to do prediction? Bayesian framework is a natural choice. More specifically, the multiple-model (MM) approach is particularly suitable to this situation since it could solve the model selection problem and parameter estimation problem jointly.

2.5.1 Multiple-Model Predictor

In the MM methods, a set $\mathbb{M}$ of models is assumed to represent the possible system behavior patterns or structures (system modes) which may jump at unknown times; a bank of filters runs in parallel at every time, each based on a particular model, to obtain model-conditioned estimates; the jumps in the system mode can be modeled as switchings between the assumed models. For simplicity, the transition between system modes are usually assumed to follow a Markov chain. Finally, the overall estimate is given by *fusion* of the estimates from these filters.

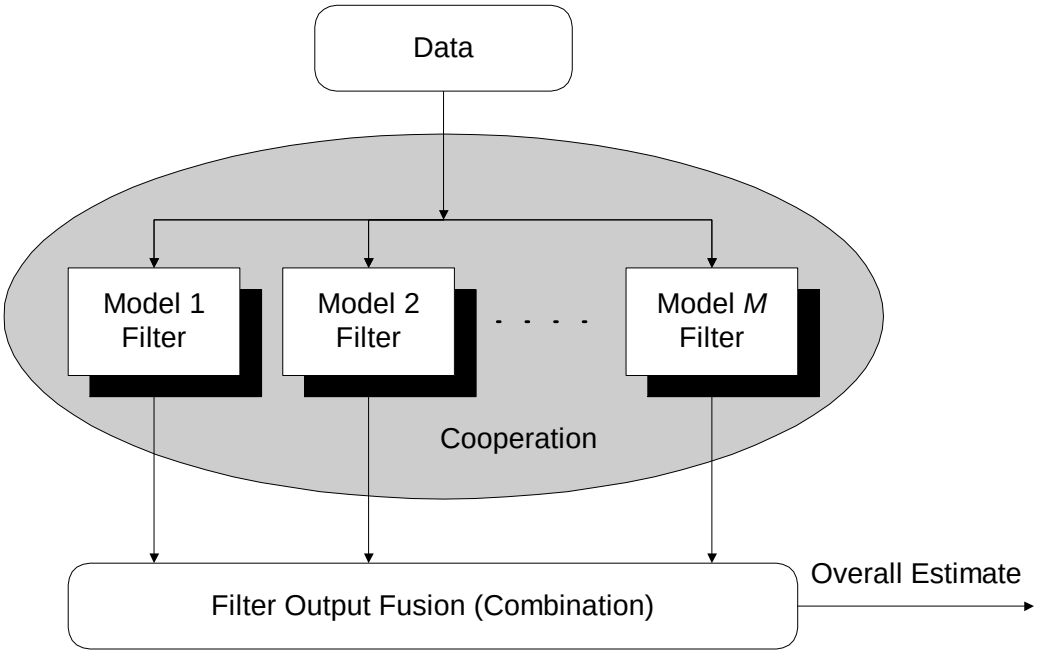


Figure 2.10: General structure of multiple-model methods

A general structure of multiple-model methods is given in Figure 2.10. Development of an MM predictor consists of the following steps: model set design, filter selection, cooperation strategy development, and estimate output fusion. Here "cooperation" means any actions taken among the filters to achieve better performance, such as individualized reconditioning of each filter (e.g., in the interacting MM (IMM) algorithm), interactive iterations and competitions (e.g., in EM based algorithms), etc. The final step, estimate fusion, can be achieved by a procedure based on hard decision or soft decision (e.g, weighted sum). A more detailed description of the MM methods can be found in [51, 55].

In the early work of the MM approach, each filter from the model set operates independently without any interaction with one another, i.e., no cooperation, because it was assumed that the mode does not jump. This type of MM estimator is referred to as the autonomous MM (AMM) [55] estimator. Many applications of AMM estimators can be found in the lit-

erature under various other names, such as "multiple model adaptive estimator/filter" [14], "multi-model partitioning algorithm" [47], etc. However, as a consequence of its underlying assumption, this method is not effective in handling systems with *frequent* mode jumps (which is likely in the Internet end-to-end measurements) because without any interaction among filters it will take a considerable amount of time for the overall estimate to converge to the true mode. Unlike AMM, the IMM estimator assumes that the system mode is a Markov (or semi-Markov) process and thus is allowed to jump between members of a set. In the IMM algorithm, each filter uses a weighted sum of the most recent estimates from every filter as its input which usually differs from one to another. By such a cooperation, IMM can capture the mode transition faster than AMM.

Most existing MM algorithms are built on state-space models and a Kalman filter (KF) is run for each model (for a nonlinear case, an extended KF (EKF) can be used). Note that based on the same idea MM algorithms can also be derived using filters in other forms (e.g., ARMA). The KF was picked mainly because of its simplicity among its peers and capability of on-line recursion.

We apply the IMM algorithm to predict the Internet end-to-end delay. Figure 2.11 shows the block diagram of the IMM algorithm. Note that each filter input matching the corresponding mode is obtained through a mixture of all filter estimates at the previous time. This operation is what "interacting" stands for. A complete cycle of the IMM scheme with Kalman filter as its mode-matched filter is summarized in Table II of [55]. The most widely used minimum mean square error (MMSE) linear predictor is applied in our approach. Let $z^k \triangleq \{z_1, ..., z_k\}$ be the end-to-end delay measurement sequence up to time k, l the time lag,

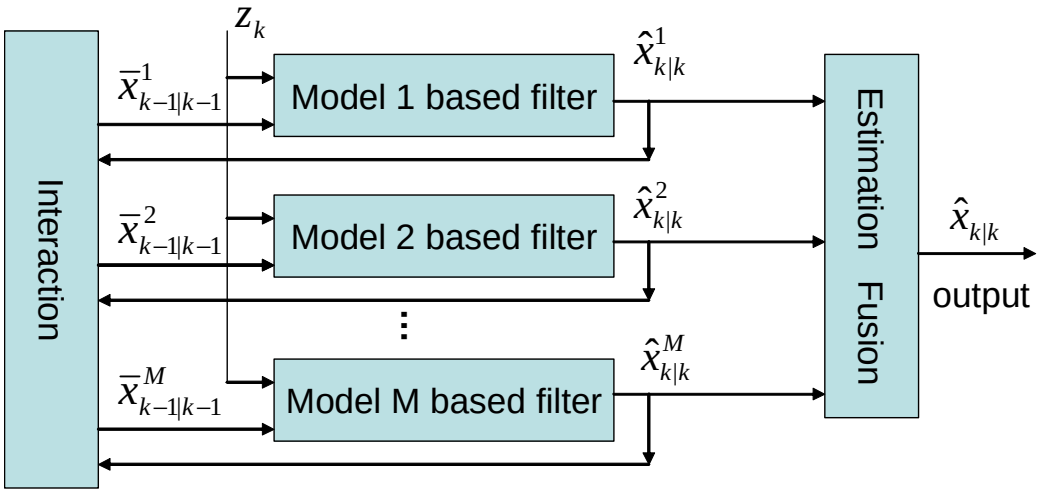


Figure 2.11: The block diagram of the IMM algorithm

then the MMSE linear predictor is

$$\hat{z}_{k+l|k} \triangleq E[z_{k+l}|z^k], \quad P_{k+l|k} \triangleq E[(\hat{z}_{k+l|k} - z_{k+l})(\hat{z}_{k+l|k} - z_{k+l})'] \tag{2.8}$$

where $P_{k+l|k}$ is the prediction error covariance matrix.

The model set is constructed from different AR models (to be presented later). We choose AR models since they have a simple structure and can approximate well a large class of short-term stationary data. To implement the IMM algorithm, the above models should be converted to the state-space representation [88]. Consider an AR(p) model

$$z_k + a_1 z_{k-1} + \cdots + a_p z_{k-p} = b_0 \omega_k \tag{2.9}$$

where $\langle \omega_k \rangle$ is a white noise sequence. Let $\mathbb{M}$ denote the set of all M designed models and j a generic model in it. Then each AR(p) model can be represented by

$$x_{k+1} = F_k^j x_k + G_k^j \omega_k^j \tag{2.10}$$

$$z_k = H_k x_k + \upsilon_k^j \tag{2.11}$$

where x is the state vector; z is the measurement vector; $\omega_k \sim \mathcal{N}(0, Q)$ and $\upsilon_k \sim \mathcal{N}(0, R)$ are independent process and measurement noise, respectively; and the initial state $x_0 \sim \mathcal{N}(\overline{x}_0, P_0)$ is independent of ω_k and υ_k. The model matrices (F, G and H) can be determined in the *observable canonical form*:

$$F = \begin{bmatrix} -a_1 & 1 & 0 & \cdots & 0 \\ -a_2 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_{p-1} & 0 & 0 & \cdots & 1 \\ -a_p & 0 & 0 & \cdots & 0 \end{bmatrix}, \; G = - \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{p-1} \\ a_p \end{bmatrix} b_0 \tag{2.12}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

It is assumed that the system mode sequence is a first order Markov chain with transition probabilities $\pi_{ij} = P\{m_{k+1}^j | m_k^i\}$, where m_k^i denotes that the ith model is in effect at time k. In MM, (posterior) model probabilities provide a meaningful measure of how likely each mode is at a given time. It can be used as a measure for detection of process jumps by comparing some preset thresholds. An l-step ahead end-to-end delay predictor is

$$\widehat{z}_{k+l|k} = \sum_{j=1}^{n} \widehat{z}_{k+l|k}^j \mu_{k+l|k}^j \tag{2.13}$$

$$P_{k+l|k} = \sum_{j=1}^{n} \mu_{k+l|k}^j [P_{k+l|k}^j + (\hat{z}_{k+l|k}^j - \hat{z}_{k+l|k})(\hat{z}_{k+l|k}^j - \hat{z}_{k+l|k})'] \tag{2.14}$$

$$\mu_{k+\eta|k}^j = \sum_{i=1}^{n} \pi_{ij} \mu_{k+\eta-1|k}^i \tag{2.15}$$

where $\widehat{z}_{k+l|k}^j$ is the predicted delay estimate from the jth elemental filter, $\mu_{k+l|k}^j$ is the corresponding model probability which can be obtained by running (2.15) from $\eta = 1$ to l, $\mu_{k|k}^i$

is the i-th model probability of the state estimate $\hat{x}_k$, and n is the number of models in the model set *in effect* at time k. (2.13) uses the total expectation theorem to obtain the overall estimate $\widehat{z}_{k+l|k}$ based on all predicted measurement estimates.

2.5.2 Model Set Design

Model set design is the most important part in the implementation of the MM methods. The performance of MM methods depends largely on the set of models chosen for the problem. A theoretical discussion of model set design can be found in [57, 55]. In this paper, based on preliminary data analysis, an off-line VQ-based method is proposed, which can be viewed as a special case of the clustering method in [57].

Vector Quantization and Clustering

Vector quantization (VQ) is a (usually lossy) data compression method based on the principle of block coding [62]. A VQ is nothing more than an approximator. The idea is similar to that of "rounding-off" (say, to the nearest integer). It is more effective than scalar quantization (can achieve less than 1 bit/parameter).

VQ has deep roots in the clustering algorithms. *Clustering* is an example of unsupervised learning. Usually, the number and form of classes $\{C_i\}$ are unknown, and available data samples $\{x_i\}$ are unlabeled. VQ can be viewed as a special case of clustering (K-means [27] clustering). From this point of view, each individual cluster centroid in VQ is called a *codeword*. The set of cluster centroids is called a *codebook*. Figure 2.12 shows a VQ example in a 2-D space.

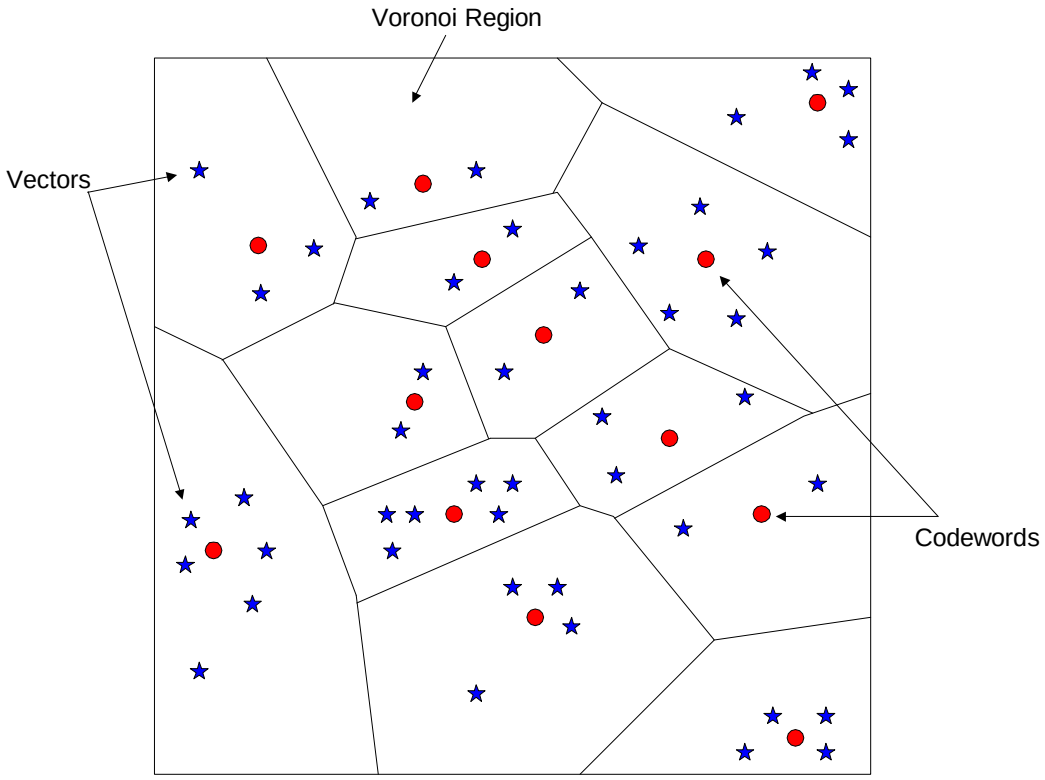


Figure 2.12: Codewords (clustering center) in 2-dimensional space, where the Voronoi regions (nearest neighbor regions) are separated with boundary lines

The LBG Algorithm

VQ can be formulated as an optimization problem. There are two optimality conditions that must be satisfied:

- Nearest neighbor condition: usually used with a Euclidean distance metric

$$d(x, y) = ||x - y||^2 = (x - y)'(x - y) \tag{2.16}$$

- Centroid condition: The codeword should be the average of all those training vectors in the Voronoi region (i.e., nearest neighbor region).

In 1980, Linde, Buzo, and Gray (LBG) proposed a VQ design algorithm based on a training sequence [58]. The LBG VQ design algorithm is an iterative algorithm which alternatively solves the above two optimality criteria. In this method, an initial codeword is set as the average of the entire training sequence. At the beginning, the initial codeword is split into two. The iterative algorithm is run with these two vectors as the initial codebook. After iteration, the two obtained codewords are splitted into four and the process is repeated until the desired number of codewords is obtained. Further details of the LBG algorithm can be found in [58].

Model Set Design Procedure

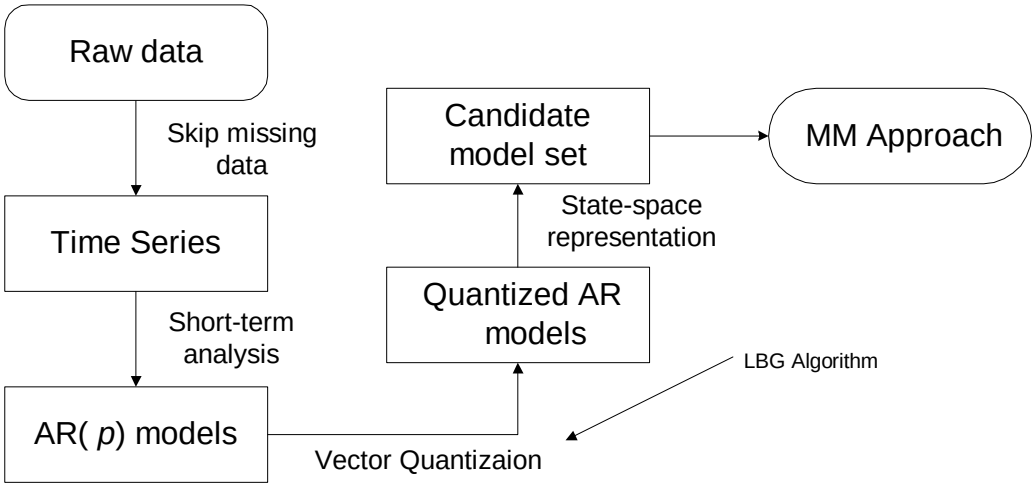


Figure 2.13: Model set design diagram

Figure 2.13 is a diagram of the model set design procedure. The same data as described in Section 2.4.1 were used to implement our MM methods. We cut each RTT sequence into two parts: one for the *training* model set here, the other for the *testing*/validation (in the next section). The ratio of the length of the training dataset to that of the testing dataset is 8:3.

Raw data in the diagram were chosen from the training dataset with 8 different destinations. By the same segmentation scheme in the preliminary data analysis, the training data were segmented into equal-length (60 samples) pieces. After segmentation, we did short-term analysis for each segment to get an AR model. Since there are too many models of different parameters obtained from the training set, the VQ method was used to select the candidate models. As discussed in Section 2.4.1, the order of AR models was chosen fixed at 4. In fact, this setting is just for simplicity, since after clustering, there is no guarantee that lower order AR models can be obtained although they are more suitable in many segments (refer to Figure 2.9). The parameters of each AR(4) model can be viewed as a 5-dimensional vector to apply VQ techniques. The LBG algorithm was used here. The size of the model set, i.e., the size of the codebook, was chosen to be 8. To implement the existing MM algorithms, the quantized models are converted to the state space.

2.6 Numerical Results

Adaptive filtering is a well known technique to estimate the signal with unknown parameters which are usually constant or slow time-varying. In general, a linear adaptive predictor is designed by a transversal filter where tap weights are continuously adapted based on data. Adaptive filtering is powerful to handle parametric changes in the stationary process, but when the parameter variation rate is high it may break down. In fact this case can be viewed as a system with structural uncertainty. Considering the unclear structure of the process corresponding to the Internet end-to-end delay, it would be meaningful to compare the performance between the linear adaptive predictors and the MM based predictors.

Least Mean Square (LMS), Recursive Least Square (RLS) are two most widely used linear adaptive filters. Here they are used as the baseline solutions whose performance is compared with that of the MM methods. The root-mean-square error (RMSE) of the end-to-end delay prediction was used as the performance measure. It is defined as

$$\text{RMSE}(\widehat{z}_{k+l|k}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(z^i_{k+l} - \widehat{z}^i_{k+l|k})^2} \tag{2.17}$$

where N is the number of Monte Carlo runs, k is the time index, l is the prediction steps, and the superscript i stands for quantities on the i-th run.

2.6.1 Synthetic Data

Figures 2.6.1 and 2.15 show the simulation results using AMM and IMM respectively. The time series in the simulation were made up of three different order AR models: AR(1), AR(2), and AR(3). More specifically, the data sequence was generated by five segments: AR(1)-AR(2)-AR(1)-AR(3)-AR(1) in order, which are shown in the figures. All data plots are averages over 100 Monte Carlo runs. The transition probabilities in the IMM were chosen as $\pi_{ii} = 0.90$, $\pi_{ij} = 0.05$, $i, j = 1, 2, 3$, $i \neq j$, which means each model was viewed equally. Comparing the results of the two methods, we found that using IMM the model switchings in the sequence can be accurately tracked, whereas using AMM the delays in time to catch the model changes are relatively long. This phenomenon will be more apparent when model switchings happen more often. In the real data we collected, abrupt peaks were observed very frequently, which suggests that IMM should be a better choice than AMM.

Figure 2.16 compared the RMSEs of IMM with those of LMS and RLS. Since the order of AR models is up to 3, the tap size n of adaptive filters was chosen as 3. For LMS, step

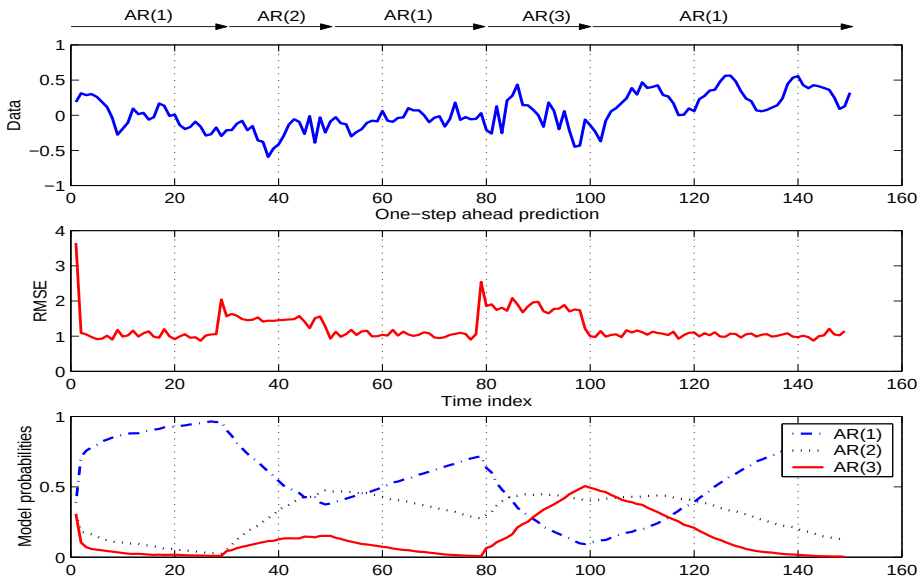


Figure 2.14: Prediction using AMM with AR models of different orders

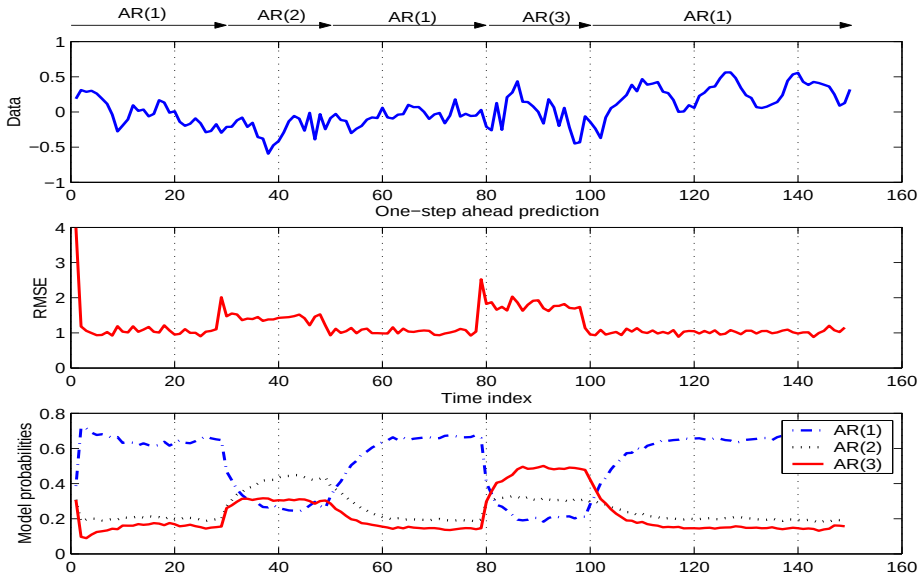


Figure 2.15: Prediction using IMM with AR models of different orders

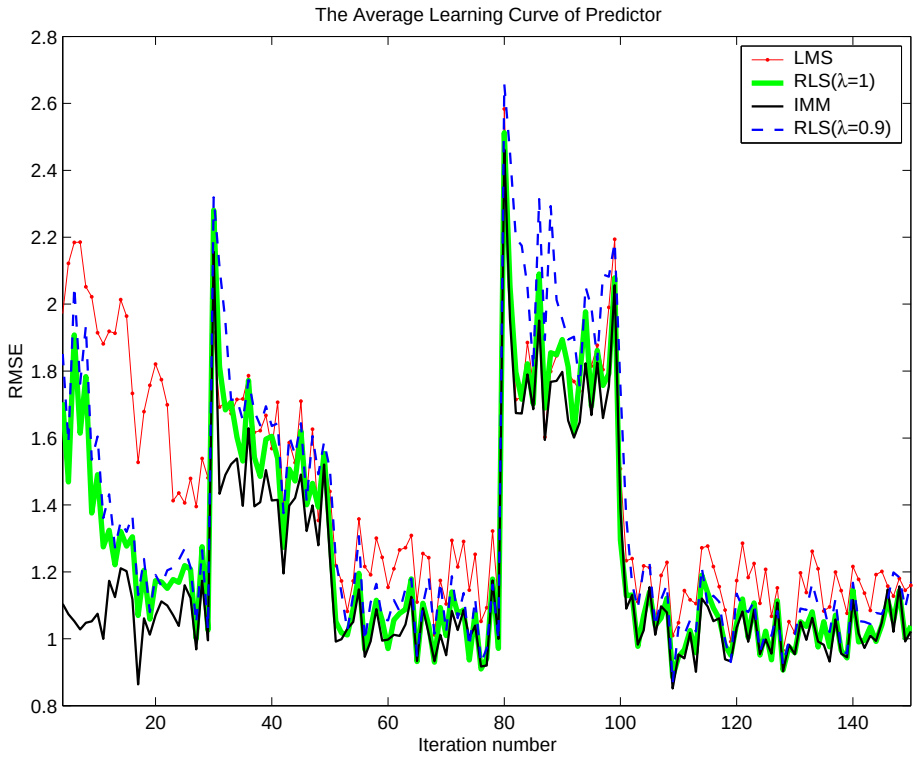


Figure 2.16: Performance comparison between IMM and adaptive filters (synthetic data)

size $\mu = 0.0035$ (larger μ, e.g., 0.005, will make the filter diverge); For RLS, initial constant $\delta = 0.1$, forgetting factor $\lambda \leq 1$. Note that $\lambda = 1$ corresponds to infinite memory. In Figure 2.16, case $\lambda = 0.9$ (only 10 samples in memory) has also been checked. The performance of IMM is always better than that of LMS and RLS. A short memory will make the performance of RLS worse.

2.6.2 Measured Data

The data in our scenario were chosen from the testing dataset obtained in Section 2.5.2. To be more general, multi-step ahead prediction was performed in the following examples. The tap size n of adaptive filters was chosen as 4. For LMS, step size $\mu = 0.1$; for RLS, initial

constant $\delta = 0.1$, forgetting factor $\lambda = 1$. For IMM, the transition probabilities were simply designed as $\pi_{ii} = 0.93$, $\pi_{ij} = 0.01$, $i \neq j$.

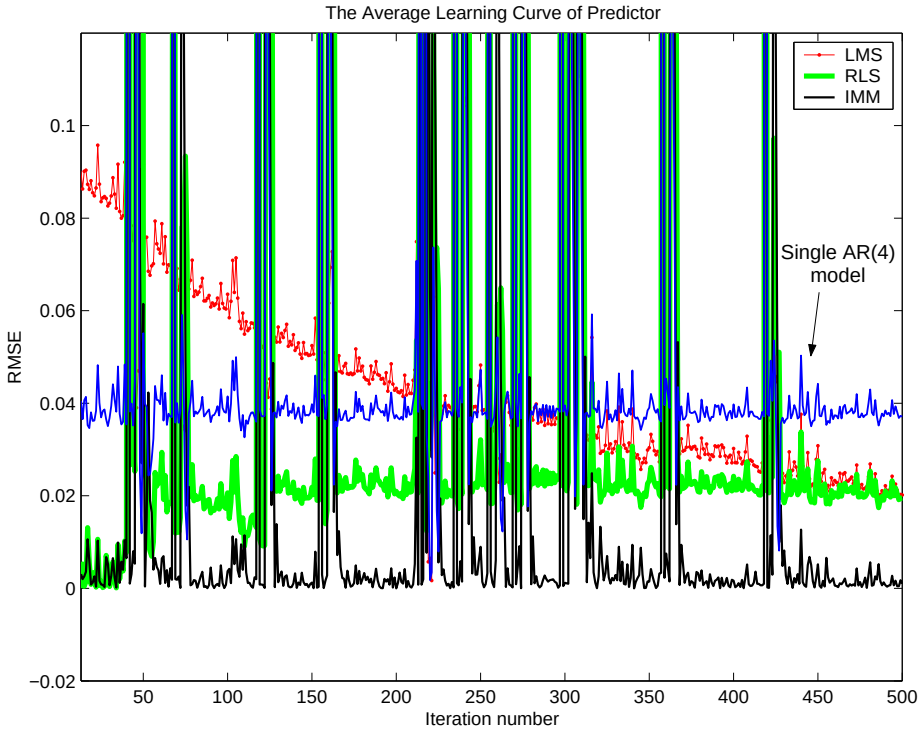


Figure 2.17: Performance comparison between IMM and adaptive filters (measured data)

The prediction results by different predictors are compared in Figure 2.17, which was zoomed in to show the difference of the flat parts of the prediction errors. Note that for the peaks, the prediction errors has the order IMM $<$ RLS $<$ LMS on average. The prediction interval l was chosen as 5. The prediction errors of a single AR(4) model are also shown in the figure as a baseline solution. Table 2.3 gives the average RMSE over time for different prediction intervals l, which also shows that IMM significantly outperforms LMS and RLS on average.

Note that to get the performance shown here, the parameters of LMS and RLS have to

Table 2.3: Average RMSE comparison

	$l = 1$	$l = 2$	$l = 5$	$l = 10$
LMS	0.0657	0.0642	0.0647	0.0668
RLS	0.0584	0.0570	0.0562	0.0586
IMM	0.0458	0.0471	0.0422	0.0454

be chosen carefully; otherwise they could be much worse. For example, the learning curve of LMS here converges very slowly; however, a larger μ will make the prediction diverge, and a smaller μ will decrease the convergence rate further. IMM is not so sensitive to the parameter design. In this sense, IMM is more robust than LMS and RLS.

2.7 Discussion and Conclusions

In this chapter, a novel approach to model the Internet end-to-end delay dynamics using MM methods has been proposed. Although each model is LTI, the MM method provides a non-stationary, nonlinear solution. It turned out that the proposed MM method performs better for prediction, in a highly non-stationary and nonlinear case, than two well known adaptive filters, namely, LMS and RLS.

A crucial part in the application of MM methods is model set design, which affects the performance of the MM methods most. In our implementation, all the AR models were chosen to have order 4 for simplicity. In fact, based on the model selection results in the preliminary data analysis, the orders of different AR models may be different between segments. Moreover, the structure of the model set is also fixed in the sense that none of the

members in the set changes over time. Considering the effectiveness, it is desirable to have models with less overlap in the model set, so that more behavior patterns can be covered. However, from a simulation study, we noticed that the overlap of the AR models with the same order might be large since they have the same model structure. A probably more reasonable design procedure is to include AR models of different orders and/or a model set with a variable structure.

Chapter 3

Joint Decision and Estimation

3.1 Introduction

3.1.1 Statistical Decision

Everyday people may have to make decisions. In statistics, a decision is made based on given hypotheses with a selected decision criterion [13]. There are quite a lot of applications in engineering problems. For instance, in radar systems, after the reflected radio waveforms are observed on the receiver, a decision is needed to make whether a target is present or absent, which is so-called single target detection. In digital communications, when the digits reach the receiver after passing through the communication channels, the measurements are the original signals corrupted by the noise during the transmission, including channel (e.g., medium, shape, path) noise, thermal noise, and receiver measurement noise. Upon receiving the observed data, we need to provide a rule to decide whether there is a signal, and/or each digit is zero or one (for a binary signal). In speech processing, from the received voice

waveforms, to make a decision which speaker is among a group is also an important problem (speaker identification). All the above examples can be formulated within statistical decision theory.

3.1.2 Parameter Estimation

Literally, according to Wikipedia, "Estimation is the calculated approximation of a result which is usable even if input data may be incomplete, uncertain, or noisy." In real engineering problems, if we consider the unknown quantity to be estimated, – let us call it *estimatee* [52], there are two types of estimatees involved in estimation. If the estimatee is time-invariant or slow-varying, it is usually called parameter estimation; if the estimatee is rapid-varying, it is called process (or state) estimation (since we are talking about statistical random process). Another widely-used alias for process estimation is "*filtering*."

The applications of estimation are also ubiquitous. Let us consider the same areas in the previous examples: In radar systems, besides determining the presence/absence of the target, at most time we also want to know relatively precisely the location, velocity, acceleration, etc. and other parameters based the observed data. In digital communications, estimating channel parameters are very important to ensure the robustness and security. In speech processing, an accurate estimate of speaker's pitch acts as the leading role in most tasks.

In this dissertation, we would like to emphasize more on the time-invariant or slow-varying, that is, the so-called parameter estimation. The results in the later example is based on the time-invariant assumption. Nevertheless, we want to point out that, the general formula presented here are not limited to parameter estimation, but also applicable to process

estimation. This scalability issue is important since in some areas, for instance, target tracking, process estimation is more important.

3.1.3 Joint Decision and Estimation

What is the difference and relationship between decision and estimation? Let us give a more rigorous definition firstly. Estimation is an operation to select a point from a continuous space. And decision is an operation to select one out of a set of discrete alternatives [12]. In other words, they both try to make the "best choice," one in a continuous space and the other in a discrete space. Decision can be viewed as a special case of estimation since usually we may consider a discrete space as a subspace of a continuous space. Of course people can argue the other way around. In fact, the estimation can be done in a discrete valued-case, the output is not necessarily one of the candidate values but some of them with probabilities. From this point of view, these two operations are highly related and they have an overlap.

In practice, many engineering problems have to be solved by using both techniques. The MM predictor proposed in the previous chapter can be regarded as such an example, since to provide the prediction output, it not only employs the estimation algorithm (Kalman filtering) but also exploits the model structure which is among a discrete candidate space. However, as a hybrid *estimation* technique, the MM method has a strong favor on the estimation part. The decision made in the algorithm provides users freedom to adaptively adjust the model parameters so that the estimation output will be more accurate than that from a time-invariant estimator. Although the estimation has an impact on the decision, the final goal is still estimation. Moreover, the decision result in the MM method may not be one

of the candidate values, which is so-called "hard decision," but some values obtained from different candidate values with conditional probabilities (let us call it "soft decision"). What is the difference between these two concepts? Consider a binary case with two candidates "0" and "1". In the hard decision, the decision output is either "0" or "1"; whereas in the soft decision, the output is some value in between. As we argued before, we can even think the soft decision as one possible way to achieve estimation. The soft decision is very important in the MM method, especially useful when the true model is not included in the candidate model set (for instance, in Internet end-to-end delay modeling, it might be the case). However, in some other engineering problems, the soft decision is unacceptable, and in this chapter we will concentrate on how to solve the problems with such requirement (hard decision) in an optimal way.

When a problem involves both decision and estimation, and the qualities of them affect each other, we call it a *joint decision and estimation* (*JDE*) problem. Let us start with a ground target tracking and recognition task to illustrate the idea. Referring to Figure 3.1, In Figure 3.1, we would like to recognize and track two possibly crossing targets (a tank and a truck) simultaneously using data transmitted from multiple sensors (e.g., cameras, radar onboard the aircraft and wireless acoustic sensors deployed in the field). The recognition and tracking are joint here, in the sense that, for example, we may want to destroy the tank, but not the truck, and therefore an accurate track (i.e., time series of state estimates) with a wrong recognition result is a disaster, while a poor track with a correct recognition would lead to a miss. Unfortunately, following the conventional strategy, possible decision errors are not accounted for in estimation; on the other hand, decision is done regardless of the result of estimation that it will lead to (maybe very poor).

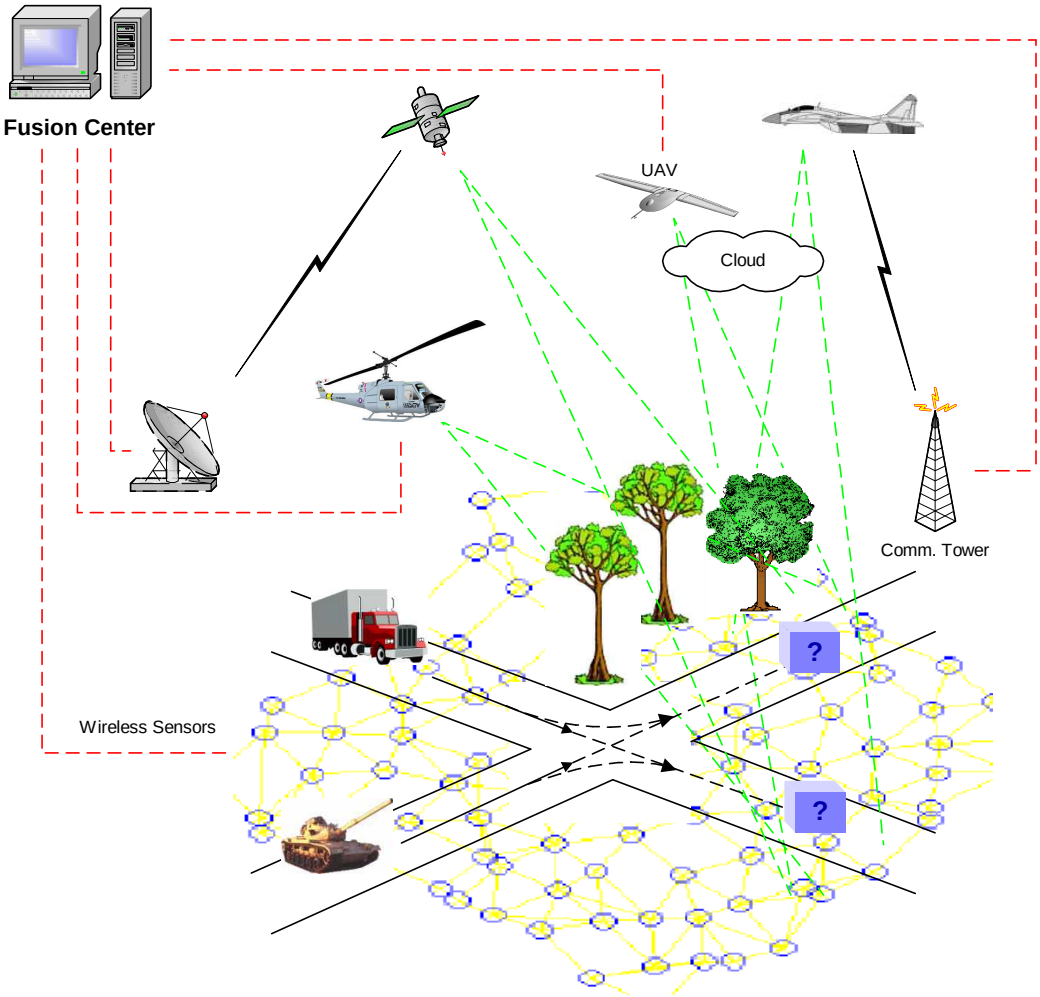


Figure 3.1: Joint tracking and recognition of crossing targets

Conventional solutions to JDE problems usually follow the strategy which can be characterized as "decision-then-estimation." This strategy tries to first make the best decision and then do estimation based on the decision made as if it were correct. It does not account for the possible decision errors in the subsequent estimation; on the other hand, decision is made regardless of the result of estimation. The problem can also be solved by "estimation-then-decision." The idea has been widely used in composite hypothesis testing, such as the so-called *generalized likelihood ratio test (GLRT)*. The decision is made by replacing the

uncertain term with the maximum likelihood estimate. Usually, the decision results using GLRT are suboptimal [25]. Due to the drawbacks in these two approaches, a joint solution is preferred.

3.1.4 Existing Work

In 1968, Middleton and Esposito [66] proposed an integrated framework to solve decision and estimation jointly. They called the approach "simultaneous signal detection and estimation." In their terms, detection and estimation are both "decision" problems since they both try to make the right decision among their problem regions. Therefore they only talked about "decision theory" which emphasizes the connection between decision and estimation. We also want to highlight the difference of the two operations, unless stated otherwise, the word "decision" is reserved for the discrete case only. From this point of view, they were handling a JDE problem and the general model they considered (they called "signal extraction") is shown in Figure 3.2.

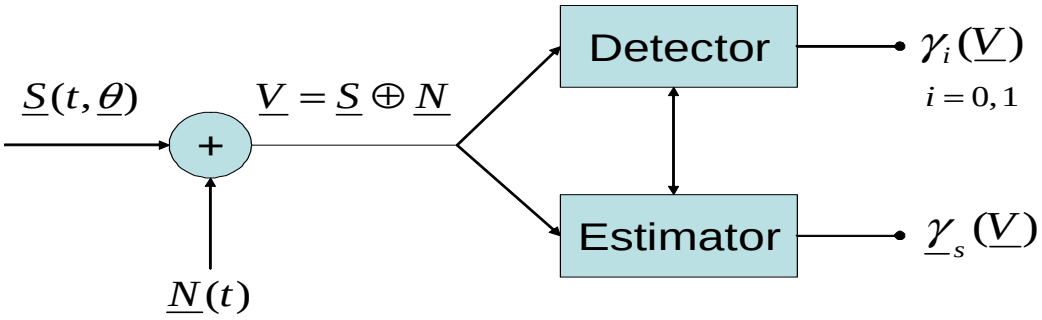


Figure 3.2: A general model of detection-estimation problem

The input to the system is either an arbitrary signal $\mathbf{S}(t, \boldsymbol{\theta})$ corrupted in an arbitrary fashion by a noise process $N(t)$ (hypothesis $\mathcal{H}_1$), or the noise process alone (hypothesis $\mathcal{H}_0$).

In general, the purpose of the system depicted in Figure 3.2 is to provide a double judgement (or in their word, "decision") at its output: a detection as to the presence or absence of the signal and, possibly, an estimate of the signal waveform $\mathbf{S}$, or of the signal parameters $\boldsymbol{\theta}$. This model includes, among other cases: 1) the usual detection problem when the estimator is not present; 2) the usual estimation problem when the signal is present with probability 1 in the observation interval and, therefore, no detection is necessary; and 3) a new type of estimation problem when there is no detection operation involved, but an estimate has to be made without certainty as to the presence of the signal. The main purpose of their approach is to improve the estimate accuracy: comparing with conventional approach, it accounts for possible decision errors, and decision is also affected by estimation. Their solution is based on a Bayesian setting. Since they considered the two operations identical in principle, they used a unified Bayes risk or cost $R(\sigma, \delta)$ in their solution.

$$R(\sigma,\delta) = \int_{\Omega} \begin{Bmatrix} \sigma(\mathbf{S}) \\ \sigma(\boldsymbol{\theta}) \end{Bmatrix} \begin{matrix} d\mathbf{S} \\ d\boldsymbol{\theta} \end{matrix} \int_{\Gamma} F_n(\mathbf{V}|\mathbf{S}(\boldsymbol{\theta}))d\mathbf{V} \int_{\Delta} \delta(\boldsymbol{\gamma}|\mathbf{V}) \begin{Bmatrix} C(\mathbf{S},\boldsymbol{\gamma}) \\ C(\boldsymbol{\theta},\boldsymbol{\gamma}) \end{Bmatrix} d\boldsymbol{\gamma} \tag{3.1}$$

where σ is a priori PDF of $\mathbf{S}$ or $\boldsymbol{\theta}$, $F_n(\mathbf{V}|\mathbf{S}(\boldsymbol{\theta}))$ conditional PDF of data $\mathbf{V}$ given $\mathbf{S}$; $C(\mathbf{S},\boldsymbol{\gamma})$, $C(\boldsymbol{\theta},\boldsymbol{\gamma})$ are cost functions relating decisions $\boldsymbol{\gamma} \triangleq (\gamma_1, \gamma_2, ..., \gamma_M)$; $\boldsymbol{\gamma}$ in detection is in a discrete set of values (e.g., $0, 1$ in ON-OFF case), $\boldsymbol{\gamma}$ in estimation is in a continuum of values; $\delta(\boldsymbol{\gamma}|\mathbf{V})$ is a decision rule; Ω, Γ, Δ, are spaces for ($\mathbf{S}$ or $\boldsymbol{\theta}$), data $\mathbf{V}$, decisions $\boldsymbol{\gamma}$, respectively (here the decision is following their definition). The optimization is done in a two-stage procedure: Initially they assume that the estimator γ is assigned and determine the best detection rule δ^* (as a function of γ); then find the best estimator γ^* that further minimizes the average risk. This approach has been applied to some real engineering problems, e.g., matched field processing [81]. Following a similar idea and system model, Fredriksen et al. [34]

provided the solution to the multiple hypotheses case. The assignments of the average Bayes risk were also given for their example. As pointed out in [85], their solutions, as well as that presented in [85], are all estimation-orientated. In [44], both estimation-orientated and decision-orientated solutions to a specified example were provided.

Most applications of the existing work are in signal reception. Amplitude estimation is one of them. It is very useful in many engineering problems: to decide the threshold in feedback links, to determine signal-to-noise ratio in communication channels, to design the gate sizes in tracking radars, and to verify and identify signals in pattern recognition, and other applications related with energy levels.

One major limitation of these existing work is that their model is not so general, in the sense that the hypotheses were not constructed in a balanced way. The H_0 is kind of special: It doesn't need estimation since $\theta = 0$; if we consider multiple hypotheses case, this problem becomes more obvious. If the application is only limited to signal reception, the idea is natural, but not in many other areas. Another critique is that there is a lack of efficient ways to evaluate the JDE performance of the algorithms.

3.2 Bayesian Decision

In the statistical terminology, decision is known as the problem of *hypothesis testing*. Following the definition in [21], "A hypothesis is a statement about a population parameter." To avoid confusion, here we assume that accepting one hypothesis is equivalent to rejecting the alternatives. The Bayesian approach to statistics has a fundamental difference from the classical approach. In the classical approach the parameter is thought to be unknown,

but fixed. In the Bayesian approach the parameter is considered a quantity described in a probabilistic setting which is called the *prior* distribution. The prior is subjective in general, based on user's belief, and given before the data. The prior is updated (using Bayes rule) by data information. The updated prior is called the *posterior* distribution.

In the Bayesian setting, the optimal decision is to minimize the following Bayes risk [52]:

$$\bar{R}_D = \sum_{i,j} c_{ij} P\{\text{“}H_i\text{”}|H_j\} P\{H_j\} \tag{3.2}$$

where $P\{H_j\}$ is the prior distribution of H_j, c_{ij} is the cost of “H_i” when H_j is true. Here “H_i” means that the decision is on H_i. When $i \neq j$, it is the of the cost of making an incorrect decision; while c_{ii} is the cost of a correct decision on H_i. In practice, c_{ij} are usually selected more or less subjectively and varies in different situations. But generally speaking, in principle, the assignment will satisfy the following condition

$$c_{ij} \geq c_{ii} \qquad i \neq j$$

meaning that a correct decision is never more costly than the corresponding incorrect decision.

The optimal Bayes decision decides on H_i if its posterior cost

$$C_i(z) = \sum_j c_{ij} P\{H_j|z\} \tag{3.3}$$

is the smallest, that is

$$C_i(z) \leq C_k(z) \qquad \forall k \tag{3.4}$$

where z denotes the observations. Consider the binary case, and assume equal prior, the test is simplified as

$$\frac{f(z|H_1)}{f(z|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} \frac{c_{10} - c_{00}}{c_{01} - c_{11}} \tag{3.5}$$

which is known as likelihood ratio test (LRT).

3.3 Bayesian Estimation

Similarly, an optimal Bayes estimator can be regarded as a function of observations z that minimizes the Bayes (estimation) risk $\bar{R}_E = E\left[C(\tilde{x})\right]$ [52], that is

$$\hat{x} = \arg\min_{\hat{x}(z)} E\left[C(\tilde{x})\right] \tag{3.6}$$

where $\bar{R}_E$ is the expectation of a *cost function* $C(\tilde{x})$ of the estimation error $\tilde{x} = x - \hat{x}$. Usually, the cost function should satisfy a set of admissibility conditions, including being symmetric about the origin, positive (semi)definite, and nondecreasing, as well as being convex. The most widely used Bayes risk for estimation is $\bar{R}_E = E[\tilde{x}'\tilde{x}]$, known as *mean-square error* (MSE). The corresponding optimal estimator is the *posterior mean*, $\hat{x} = E[x|z]$, which is known as *minimum mean-square error* (MMSE) estimator. When estimatee x and observations z are jointly normal distributed, the optimal MMSE estimate is

$$E[x|z] = E[x] + C_{xz}C_z^{-1}(z - E[z]) \tag{3.7}$$

and the conditional covariance (MSE)

$$\mathrm{MSE}(\hat{x}|z) = C_x - C_{xz}C_z^{-1}C_{zx} \tag{3.8}$$

where covariance matrix $C_{ab} = E\left[(a - E[a])(b - E[b])'\right]$, $C_a = C_{aa}$, and here a, b denote two arbitrary random variables.

Note that in the case that the $C(\tilde{x})$ is not chosen as $\tilde{x}'\tilde{x}$, but if the posterior density $f(x|z)$ is symmetric about its mean and $C(\tilde{x})$ is symmetric and convex, we still have the optimal estimate $\hat{x} = E[x|z]$.

3.4 Composite Hypothesis Testing

The reason that we want to mention this topic is that this is a famous problem involving both decision and estimation in statistics. If the parameters in the hypotheses are all known, we call the test simple; if there are unknown parameters present in the hypotheses, the test is called composite. Using our taxonomy, most tests for composite hypotheses belong to the "estimation-then-decision" strategy.

There are two widely-used techniques to solve this problem [54]. One is the so-called *generalized likelihood ratio test* (GLRT). In this method, the hypothesized set Θ_i of possible parameter values is lumped into (or represented by) a single value $\hat{\theta} = \hat{\theta}^{\mathrm{ML}} = \arg\max_{\theta\in\Theta_i} f_i(z|\theta)$, that is, the maximum likelihood estimate of θ over Θ_i. In other words, the original composite distribution $f_i(z|\theta), \theta \in \Theta_i$ is replaced by the *most likely distribution* $f_i(z) = f_i(z|\hat{\theta}^{\mathrm{ML}})$, since $f_i(z|\hat{\theta}^{\mathrm{ML}}) = \max_{\theta\in\Theta_i} f_i(z|\theta)$. The test is then based on the simple hypothesis $H_i : z \sim f_i(z|\hat{\theta}^{\mathrm{ML}})$.

The other method is based on the Bayes setting. If the parameter θ in the hypothesized distribution is (assumed) random with known prior distribution $f(\theta)$, a natural way to make the test simple is to perform *marginalization*; that is, replace the original composite distribution $f_i(z|\theta), \theta \in \Theta_i$ by its marginal distribution $f_i(z)$ via marginalization:

$$f_i(z) = E[f_i(z|\theta)|\theta \in \Theta_i] = \int_{\theta\in\Theta_i} f_i(z|\theta) f(\theta) d\theta$$

that is, by its *average distribution* $f_i(z)$. The test is then based on the simple hypothesis $H_i : z \sim f_i(z)$. In the literature, sometimes this approach was still called GLRT (for instance, in [65]), we think that it is more appropriate to name it *marginalized likelihood ratio test* (MLRT).

3.5 General Formulation

In [54], a novel approach to the JDE problems was proposed. We will summarize the framework in the following two subsections.

Consider a JDE problem in which x is the quantity to be estimated and the decision involves M candidates: $D_1, \ldots, D_M$. The complete Bayesian solution to the problem of estimating x using data z is its posterior density $f(x|z)$. By the same token, a complete Bayesian solution to the problem of deciding candidates $D_1, \ldots, D_M$ is the set of posterior probabilities $\{P\{D_1|z\}, \ldots, P\{D_M|z\}\}$. We call this set of probabilities *soft decisions*. In the same spirit, when x and D are not independent, we may be interested in inferring them *jointly* in terms of the set of functions $\{P[(D,x)_1|z], \ldots, P[(D,x)_M|z]\}$, where $P[(D,x)_i|z]$ is the posterior mixed "probability-density." Here the notation $(D,x)_i \triangleq (x(D_i), D_i(x))$ is used to emphasize the mutual dependence of x and D.

If x is not dependent on D, then $P[(D,x)_i|z] = P\{D_i(x)|x,z\}f(x|z)$, which is the case when x is an unknown parameter common to all models D_i, as in classification of a signal or target where the hypotheses have a common unknown parameter. This suggests an estimation-then-decision method. If D_i does not depend on x, then $P[(D,x)_i|z] = f(x|z,D_i)P\{D_i|z\}$. An example of this case is when D_i is the ith model of a system with the state x. This justifies a decision-then-estimation strategy.

The general case in which different D may involve different x has numerous inference examples. For target inference, this includes fusion of tracks (x_i) that may correspond to different targets (D_i), joint tracking (x_i) and classification (D_i) of targets, and tracking (x_i) an unknown number (D_i) of targets. Because of more inter-correlations are introduced,

usually the factorization cannot be performed as the previous case. A joint solution is expected to achieve better inference quality.

The JDE framework in [54] is highly related to traditional Bayes decision and estimation. Based on the analogy of the solutions to decision and estimation, the approach to JDE problem is to minimize the following Bayes risk for joint decision and estimation

$$\bar{R} = \sum_{i=1}^{M} \sum_{j=1}^{N} (\alpha_{ij} c_{ij} + \beta_{ij} E[C(\tilde{x})|D_i, H_j]) P\{D_i, H_j\} \tag{3.9}$$

where D_i stands for the i-th decision, which is equivalent to the event $\{z \in \mathcal{D}_i\}$, here $\mathcal{D}_i$ is the decision region for D_i in the data space; c_{ij} is the cost of decision D_i when hypothesis H_j is true; $\tilde{x} = x - \hat{x}$ is the estimation error; $C(\tilde{x})$ is the estimation cost function, which is a convex function of $\tilde{x}$, e.g., $\tilde{x}'\tilde{x}$; $E[C(\tilde{x})|D_i, H_j]$ is the expected cost conditioned on the case that the decision is made on D_i but H_j is true; α_{ij} and β_{ij} are relative weights of decision and estimation costs, which can be chosen to fit different requirements given by practical problems. When $(\alpha_{ij}, \beta_{ij}) = (1, 0)$ and $(0, 1)$, the joint risk $\bar{R}$ reduces to conventional Bayes risk for decision and estimation, respectively. When $(\alpha_{ij}, \beta_{ij}) = (1, 1)$, $\bar{R}$ is the sum of these two conventional Bayes risks. When $(\alpha_{ij}, \beta_{ij}) = (0, c_{ij})$, $\bar{R}$ is the weighted sum of the product of decision and estimation costs.

This new Bayes risk $\bar{R}$ is a generalization of the traditional Bayes risk for decision, $R_D = \sum_{i,j} c_{ij} P\{\text{“}H_i\text{”}|H_j\} P\{H_j\}$, and the traditional Bayes risk for estimation, $R_E = E[C(\tilde{x})]$, respectively. In general, $D_i \neq$ “H_i” because in this new formulation there need not be a one-to-one correspondence between the set of decision regions $\{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ and the set of hypotheses $\{H_1, \ldots, H_N\}$. As such, the decision part of this framework is more general than *hypothesis* testing, which is limited to $D_i =$ “H_i”.

By an appropriate choice of α_{ij} and β_{ij}, this new framework is suitable for all three classes of JDE problems: 1) decision and estimation are virtually equally important; 2) decision-orientated (decision is primary and estimation is secondary, e.g., composite hypothesis testing); and 3) estimation-orientated (estimation is primary and decision is secondary, e.g., hybrid estimation). In other words, the relative weight of decision and estimation in a JDE problem can be captured by the relative magnitudes of α_{ij} and β_{ij}.

Eq. (3.9) provides a basis for an integrated approach to JDE. Effective and efficient Bayes JDE procedures can be developed in this framework.

3.6 Solution

3.6.1 Decision Part

For any given $E[C(\tilde{x})|D_i, H_j]$, to minimize $\bar{R}$ in (3.9), the *optimal decision* D is

$$D = D_i \qquad \text{if } C_i(z) \leq C_k(z), \forall k \tag{3.10}$$

where the *posterior cost* is given by

$$C_i(z) = \sum_{j=1}^{N} (\alpha_{ij} c_{ij} + \beta_{ij} E[C(\tilde{x})|D_i, H_j]) P\{H_j|z\}$$

3.6.2 Estimation Part

Given any set of decision regions $\{\mathcal{D}_1, ..., \mathcal{D}_M\}$ of the data space, the *optimal estimator* for (3.9) with $C(\tilde{x}) = \tilde{x}'\tilde{x}$ is the following generalized conditional mean

$$\hat{x} = \sum_{i,j} E[x|z, D_i, H_j] \bar{P}\{D_i, H_j|z\} = \sum_{i,j} \hat{x}_{ij} \bar{P}\{D_i, H_j|z\} \tag{3.11}$$

where

$\hat{x}_{ij} = E[x|z, D_i, H_j] = E[x|z, H_j] \triangleq \hat{x}_j$ if $z \in \mathcal{D}_i$ and undefined otherwise

$$\bar{P}\{D_i, H_j|z\} = \frac{\beta_{ij} P\{D_i, H_j|z\}}{\sum_{k,l} \beta_{kl} P\{D_k, H_l|z\}} = \frac{\beta_{ij} P\{D_i|H_j, z\} P\{H_j|z\}}{\sum_{k,l} P\{D_k|H_l, z\} P\{H_l|z\}} = \frac{\beta_{ij} 1(z; \mathcal{D}_i) P\{H_j|z\}}{\sum_{k,l} \beta_{kl} 1(z; \mathcal{D}_k) P\{H_l|z\}}$$

where $1(z; \mathcal{D}_i) = \begin{cases} 1 & z \in \mathcal{D}_i \\ 0 & \text{else} \end{cases}$

A Special Case:

If $z \in \mathcal{D}_i$ implies $z \notin \mathcal{D}_k$, $\forall k \neq i$ (e.g., when $\{\mathcal{D}_1, ..., \mathcal{D}_M\}$ forms a partition of the observation space), then

$$1(z; \mathcal{D}_i) = \begin{cases} 1 & z \in \mathcal{D}_i \\ 0 & z \in \mathcal{D}_k, \forall k \neq i \end{cases}$$

$\Rightarrow$

$$\sum_{k,l} \beta_{kl} P\{D_k, H_l|z\} = \sum_l P\{H_l|z\} \sum_k \beta_{kl} 1(z; \mathcal{D}_l) = 1(z; \mathcal{D}_i) \sum_l \beta_{il} P\{H_l|z\}$$

$\Rightarrow$

$$\begin{aligned} \hat{x} &= \sum_i \sum_j \hat{x}_{ij} \frac{1(z; \mathcal{D}_i) \beta_{ij} P\{H_j|z\}}{1(z; \mathcal{D}_i) \sum_l \beta_{il} P\{H_l|z\}} \\ &= \sum_i 1(z; \mathcal{D}_i) \sum_j E[x|z, H_j] \frac{\beta_{ij} P\{H_j|z\}}{\sum_l \beta_{il} P\{H_l|z\}} \end{aligned}$$

thus the above optimal estimator is simplified by

$$\hat{x} = \sum_i 1(z; \mathcal{D}_i) \check{x}_i \tag{3.12}$$

where

$$\breve{x}_i = \bar{E}[x|z, D_i] \triangleq \sum_j E[x|z, D_i, H_j]\bar{P}\{H_j|z, D_i\} = \begin{cases} \sum_j \hat{x}_j \bar{P}\{H_j|z\} & z \in \mathcal{D}_i \\ \text{undefined} & \text{else} \end{cases}$$

$$\bar{P}\{H_j|z, D_i\} = \frac{\beta_{ij} P\{H_j|z, D_i\}}{\sum_{kl} \beta_{kl} P\{H_l|z, D_k\}} = \begin{cases} \frac{\beta_{ij} P\{H_j|z\}}{\sum_l \beta_{il} P\{H_l|z\}} & z \in \mathcal{D}_i \\ \text{undefined} & \text{else} \end{cases}$$

and

$$P\{H_j|z, D_i\} = \begin{cases} P\{H_j|z\} & z \in \mathcal{D}_i \\ \text{undefined} & \text{else} \end{cases}$$

is the posterior probability of H_j under the ith decision.

3.6.3 A JDE Algorithm

The optimal joint decision-estimate $(D, \hat{x})$ is the combination of the above optimal decision and optimal estimate. It can be obtained by the following iterative algorithm, which always converges.

1. Start from an arbitrary initial decision set $\{D_i\}_{r=0}$ as the initial partition $\{\mathcal{D}_i\}$, where r denotes the iteration index.

2. Estimation update: based on $\{D_i\}_r$, calculate $\hat{x}_{ij}$ and $\bar{P}\{D_i, H_j|z\}$ to obtain the optimal estimate $\hat{x}$ with the corresponding conditional mean square error.

3. Decision update: calculate $C_i(z)$ for each D_i, then $\bar{R}_r$ is obtained.

4. Compare $\bar{R}_r$ and $\bar{R}_{r-1}$. If the difference ratio $\left|\bar{R}_r - \bar{R}_{r-1}\right| / \bar{R}_{r-1}$ is small enough, stop the iteration and D_i with the smallest posterior cost at iteration r is taken as the

optimal decision and the corresponding $\hat{x}$ is taken as the optimal estimate; otherwise change decision regions to $\{\mathcal{D}_i\}_r$ and repeat step 2, 3, 4.

The convergency was proved and the simplification of this algorithm was discussed in [54].

3.6.4 Remarks

Although a composite hypothesis testing problem is often solved by estimation first and then decision, a JDE problem is traditionally solved in two steps: hard decision followed by estimation. Assume there is no decision error, namely, $P\{H_j|D_i, z\} = \delta_{i-j}$ if $z \in \mathcal{D}_i$ and undefined otherwise, where δ_{i-j} is the Kronecker delta: $\delta_{i-j} = 1$ if $i = j$ and 0 otherwise. Then the optimal estimator in Eq. (3.12) reduces to

$$\hat{x} = \sum_i 1(z; \mathcal{D}_i) E[x|z, H_i] \tag{3.13}$$

which is actually in the traditional "decision-then-estimation" manner. Eq. (3.13) is the condition mean based on a single "correct" decision, whereas Eq. (3.12) is a weighted sum of the conditional means under different decisions. In other words, Eq. (3.13) makes a hard decision, and Eq. (3.12) draws a soft decision which takes possible decision errors into account.

Compared with the existing works [34, 26, 23, 41], the approach in [54] is not only providing a systematically integrated framework in theory, but also coming up with an iterative algorithm to implement with proven convergence. Comparing with [34], this approach is more general: the decision regions and hypothesis set are not necessarily one-to-one correspondent, which is the more likely case in reality. This approach is also different from *hybrid*

estimation [50] which is widely used in target tracking. In target tracking, the state estimate with target motion uncertainty is treated as a hybrid estimation problem, which is highly nonlinear with both continuous and discrete uncertainties. By combining the estimation results from different (discrete known) motion models, target motion can usually be well tracked. Although this approach implicitly improves estimation results by decision (mainly "soft," e.g., choose different weight on the output of different models), it is still an estimation technique. With the framework proposed in [54], the impacts from decision and estimation on each other are simultaneously addressed.

3.7 Performance Evaluation

The discussion in this subsection is originally from [53].

While many practical problems involve JDE, their solutions are evaluated so far only in terms of decision performance and estimation performance, separately. We are not aware of any measure, comprehensive or not, for evaluating *joint* decision and estimation performance. In this section, we propose a systematic method and measure for evaluating a JDE algorithm comprehensively based on statistical distance between the original data and the mock data generated by the JDE algorithm.

A large class of JDE problems can be formulated as follows. The ground truth is that the observation data z has a distribution $F(z|s,x)$; that is, $z \sim F(z|s,x)$, where x is to be estimated and s is unknown with discrete (or finitely many possible) values. Probably more often, the exact form of $F(z|s,x)$ is not known; rather, it is known that $z = h(s,x,v)$, where $v \sim F(v|s,x)$ with known $F(v|s,x)$. A JDE problem consists of two subproblems:

decide on the s value and estimate x. With the assumption that $s \in \{1, 2, \ldots, N\}$, it can be formulated as estimating x and testing the hypotheses:

$$H_1 \quad \text{vs.} \quad H_2 \quad \text{vs.} \quad \cdots \quad \text{vs.} \quad H_N$$

where $H_i : z \sim F_i(z|x)$ and $F_i(z|x) = F(z|s, x)|_{s=i}$. Let $\xi = (s, x)$. Then the solution to a JDE problem is $\hat{\xi} = (d, \hat{x})$, where $\hat{x}$ is the estimate of x and $d \in \{1, 2, \ldots, N\}$ is the decision.

The basic idea of our method for evaluating JDE performance is to measure some distance between the true distribution $F(z|s, x)$ and the distribution $F_d(z|\hat{x})$ corresponding to the JDE result if they are available, such as in a simulation-based evaluation study, or some statistical distance between the original data z and the mock data $\hat{z}$ generated by the JDE algorithm if the distributions are not available. Here the mock data $\hat{z}$ is generated randomly by the JDE algorithm with the result $(d, \hat{x})$ via either the distribution $F_d(z|\hat{x})$ or the data model that converts a JDE result to the data, such as $\hat{z} = h(d, \hat{x}, v)$ with $v \sim F(v|d, \hat{x})$.

More specifically, let

$$\rho(z, \hat{z}) = \int\int\int \Delta[f(z|s, x), f_d(z|\hat{x})] dF(z, s, x)$$

where $\Delta[f(z|s, x), f_d(z|\hat{x})]$ is the "distance" between $f(z|s, x)$ and $f_d(z|\hat{x})$.

The above metric can be evaluated via Monte Carlo methods as follows:

If both s and x are random, generate $\xi_i \sim f(\xi)$, $i = 1, \ldots, N_i$, where $f(\xi)$ is the prior distribution of ξ for performance evaluation determined by the evaluator, which could differ from the one used in the JDE algorithm.

- For each pair (s_i, x_i), generate $z_{ij} \sim f(z|s_i, x_i)$, $j = 1, \ldots, N_j$, each may be a vector consisting of multiple pieces of data.

- For each z_{ij}, obtain $\hat{\xi}_{ij} = (d_{ij}, \hat{x}_{ij}) = g(z_{ij})$ by the JDE to be evaluated.

- Let $\rho(z_i, \hat{z}_{ij}) = \Delta[f(z|\xi_i), f_{d_{ij}}(z|\hat{x}_{ij})]$, where $\Delta[f(z|\xi_i), f_{d_{ij}}(z|\hat{x}_{ij})]$ is the "distance" between $f(z|\xi_i)$ and $f_{d_{ij}}(z|\hat{x}_{ij})$. Then, compute the final performance metric

$$\begin{aligned}\rho(z, \hat{z}) &= \int\int \Delta[f(z|s, x), f_d(z|\hat{x})]dF(z, \xi) \\ &\approx \frac{1}{N_i N_j}\sum_{i=1}^{N_i}\sum_{j=1}^{N_j}\rho(z_i, \hat{z}_{ij})\end{aligned}$$

  where $\rho(z_i, \hat{z}_{ij}) = \Delta[f(z|\xi_i), f_{d_{ij}}(z|\hat{x}_{ij})]$ is the "distance" between $f(z|\xi_i)$ and $f_{d_{ij}}(z|\hat{x}_{ij})$. If we use the *total variation distance*

$$\begin{aligned}\Delta_{tv}(F_t, F_d) &= \frac{1}{2}\int |f_t(z) - f_d(z)|dz \\ &= \frac{1}{2}\sum_i |p_t(z_i) - p_d(z_i)|\end{aligned}$$

  where p_t and p_d are pmfs. $\rho(z, \hat{z})$ will have a simple form when pmfs are involved:

$$\rho_{tv}(z, \hat{z}) \approx \frac{1}{2N_i N_j}\sum_{i=1}^{N_i}\sum_{j=1}^{N_j}\sum_k |p(z_{ik}) - \hat{p}(\hat{z}_{ijk})|$$

Remark 1 *The randomness assumption of s and x is not necessary. In [53], the cases that s, x under different assumptions were discussed in details. However, in our later implementation in the dissertation, s and x are assumed random because of the Bayesian framework. The other case studies are omitted here since they are out of the scope of discussion.*

Chapter 4

Joint Target Tracking and Classification in JDE Framework

4.1 Introduction

Typical target inference problems involving both decision and estimation include:

- Joint target classification (or recognition) and tracking, including possibly crossing or closely-spaced targets.
- Integrated track fusion, which handles track-to-track association and track-to-track fusion jointly. It should be capable of fusing tracks with non-kinematic attributes, including target type or class, track quality measures, and target discriminative features.
- Target tracking in the presence of various measurement-origin uncertainties due to clutter, target-sensor geometry, sensor resolution, etc.

Target tracking and classification (recognition) are both important problems in surveillance systems. Many applications can be found in aircraft control, ground transportation management, and building safety and alarm systems. Generally speaking, these two operations are coupled. But in most current approaches, they are handled independently due to the different sensor measurements and available techniques. For instance, in many target tracking problems, tracking algorithms are mainly based on kinematic sensing devices (e.g., radar, sonar) and associated models. On the other hand, target classification is usually handled using the data from identity or attribute sensing devices (e.g., electronic support measure (ESM), high resolution radar; in the wireless sensor networks community, features extracted from acoustic, passive infrared, and seismic modalities are widely used for identity purpose). To overcome the potential drawbacks, how to utilize the coupling between the two operations more effectively has received increasing attention in recent years (see, e.g., [8, 17, 23, 25, 35, 40, 41, 42, 48, 64, 67, 71, 79]).

In the literature, efforts were made to construct a single unified framework to handle the two problems jointly. For instance, target dynamics (class-dependent kinematic models) were exploited to help classification by Jacobs and O'Sullivan [42]. In [23], the coupling between tracking state and target identity was taken into account to improve tracking result based on IMM filter [55]. However as a hybrid estimation techniques, this approach emphasizes more on estimation and no hard decision is made (sometimes it is needed). Another example using hybrid estimation techniques has been described in [17]. In [8, 41] sequential Monte Carlo algorithms and particle filtering were utilized to provide integrated solutions. Similarly, those approaches also pay more attention on the estimation.

These existing work either only considers one-way dependence (decision to estimation or vice versa), or emphasizes one aspect of the two at a time even if the two-way dependence is considered. Here we provide a JDE solution following the framework in [54] to a simple example in target tracking and classification to illustrate how the data are fully utilized in a joint manner, and both tasks are performed in a relatively more balanced way. We consider a simple yet representative JTC example, where three types of data are available: the first type is useful for both tracking and classification; the second is particularly good for classification but not directly useful for tracking, the third is particularly good for tracking but not directly useful for classification. We present optimal decision, optimal estimation, decision-then-estimation, estimation-then-decision, and our proposed JDE solution in the Bayesian setting for this example. In this way, we demonstrate how the proposed JDE solution work and contrast its performance with the existing methods.

While the JDE solution proposed in [54] is general, it relies on several design parameters. Only simple guidelines have been presented in [54]. Here a case study is conducted concerning these design parameters that make a trade-off between decision and estimation performance.

Also, we adopt a general method of evaluating performance of joint decision and estimation in a comprehensive way [56]. To our knowledge, it is the only method, be it comprehensive or not, available that evaluates joint decision and estimation performance.

4.2 JDE Solution to JTC problem

4.2.1 Problem Formulation

Consider three types of data z_1, z_2, z_3 having the same size from three different sensors that are perfectly synchronized. For simplicity, our presentation below is for the case in which each piece of data is a scalar quantity, but the approach works for the general vector case without difficulty. The first type is obtained from infrared imagers (or other energy-selective sensing devices) modeled as

$$z_{1j} = \theta x + v_j, \qquad j = 1, ..., n \tag{4.1}$$

where $v_j \sim \mathcal{N}(0, \sigma_v^2)$ is i.i.d. Gaussian noise; x denotes the target state, which has a normal prior $\mathcal{N}(\bar{x}, \sigma_x^2)$ and is independent of v; and the modulation term θ has two possible values,

$$\begin{aligned} H_0 : &\quad \theta = \theta_0 \\ H_1 : &\quad \theta = \theta_1 \end{aligned} \tag{4.2}$$

which correspond to two possible classes of objects, e.g., humans and moving vehicles, respectively, since different classes of objects have different infrared features, assumed to be reflected in amplitude modulation.

The second type of data is obtained from ESM sensors (or some devices based on image, acoustic, or seismic features). The readings are in identity type and used to indicate different

target classes with certain probabilities:

$$\begin{aligned} &P\{z_{2j} = \theta_0 | \theta = \theta_0\} = 1 - p_0 \\ &P\{z_{2j} = \theta_1 | \theta = \theta_0\} = p_0 \\ &P\{z_{2j} = \theta_0 | \theta = \theta_1\} = 1 - p_1 \\ &P\{z_{2j} = \theta_1 | \theta = \theta_1\} = p_1 \end{aligned} \qquad j = 1, ..., n \tag{4.3}$$

Assume $z_{21}, \ldots, z_{2n}$ are independent.

The third type of data is obtained from kinematic sensing devices, say, radar, modeled as

$$z_{3j} = x + w_j, \qquad j = 1, ..., n \tag{4.4}$$

where $w_j \sim \mathcal{N}(0, \sigma_w^2)$ is independent Gaussian noise.

The objective is not only to track the moving target x (with continuous uncertainty) but also to determine the target type θ. Note that type 2 and type 3 data do not provide direct information for tracking and classification, respectively. As a result, it is difficult to use type 2 data for tracking or use type 3 data for classification without joint decision and estimation.

4.2.2 Conditional Independence

It is assumed that measurement errors from different sensors are independent and thus the three types of data are conditionally independent; that is,

$$\begin{aligned} f(z_1, z_2, z_3 | H_i, x) &= f(z_1 | H_i, x) f(z_2 | H_i, x) f(z_3 | H_i, x) \\ &= f(z_1 | H_i, x) f(z_2 | H_i) f(z_3 | x) \end{aligned}$$

It turns out that type 1 and type 2 data are independent conditioned on the hypothesis (without x):

$$f(z_1, z_2|H_i) = f(z_2|H_i)f(z_1|H_i)$$

which follows from

$$\begin{aligned} f(z_1, z_2|H_i) &= \int f(z_1, z_2|H_i, x)f(x|H_i)dx \\ &= \int f(z_1|H_i, x)f(z_2|H_i)f(x)dx \\ &= f(z_2|H_i)f(z_1|H_i) \end{aligned} \tag{4.5}$$

Similar conditional independence holds between type 3 and type 2 data and between type 2 data and type 1 and type 3 data

$$f(z_2, z_3|H_i) = f(z_2|H_i)f(z_3|H_i) \tag{4.6}$$

$$f(z_1, z_2, z_3|H_i) = f(z_2|H_i)f(z_1, z_3|H_i) \tag{4.7}$$

but in general type 1 and type 3 data are not independent conditioned on the hypothesis (without x)

$$\begin{aligned} f(z_1, z_3|H_i) &= \int f(z_1|H_i, x)f(z_3|x)f(x)dx \\ &= \int \mathcal{N}(z_1; \theta_i x\mathbf{1}, \sigma_v^2 I)\mathcal{N}(z_3; x\mathbf{1}, \sigma_w^2 I)\mathcal{N}(x; \bar{x}, \sigma_x^2)dx \end{aligned} \tag{4.8}$$

where $\mathbf{1} = \mathbf{1}_n = [\overbrace{1, \ldots, 1}^{n}]'$.

4.2.3 Likelihood Functions

Based on the data models, it can be derived (see Appendix A) that

$$\begin{aligned} f(z_1|H_i) &= \int f(z_1|H_i, x) f(x) dx \\ &= \int \mathcal{N}(z_1; \theta_i x \mathbf{1}, \sigma_v^2 I) \mathcal{N}(x; \bar{x}, \sigma_x^2) dx \\ &= c \exp\left[-\frac{1}{2\sigma_v^2} \sum_{j=1}^{n} (\hat{z}_1 - z_{1j})^2 - \frac{(\bar{z}_i - \bar{x})^2}{2(\sigma_{zi}^2 + \sigma_x^2)} \right] \end{aligned} \tag{4.9}$$

$$\begin{aligned} f(z_3|H_i) &= \int f(z_3|x) f(x) dx \\ &= \int \mathcal{N}(z_3; x \mathbf{1}, \sigma_w^2 I) \mathcal{N}(x; \bar{x}, \sigma_x^2) dx \\ &= c' \exp\left[-\frac{1}{2\sigma_w^2} \sum_{j=1}^{n} (\hat{z}_3 - z_{3j})^2 - \frac{(\hat{z}_3 - \bar{x})^2}{2(\sigma_w^2/n + \sigma_x^2)} \right] \end{aligned} \tag{4.10}$$

$$\begin{aligned} f(z_1, z_3|H_i) = c'' \exp\Bigg[&-\frac{1}{2\sigma_v^2} \sum_{j=1}^{n} (\hat{z}_1 - z_{1j})^2 \\ &- \frac{1}{2\sigma_w^2} \sum_{j=1}^{n} (\hat{z}_3 - z_{3j})^2 \\ &- \frac{\sigma_w^2(\bar{x} - \bar{z}_i)^2 + \sigma_{zi}^2(\bar{x} - \hat{z}_3)^2 + \sigma_x^2(\bar{z}_i - \hat{z}_3)^2}{2\left(\sigma_w^2\sigma_{zi}^2 + \sigma_x^2\sigma_w^2 + \sigma_x^2\sigma_{zi}^2\right)} \Bigg] \end{aligned} \tag{4.11}$$

where $\mathcal{N}(y; \bar{y}, \sigma_y^2)$ stands for the pdf of a Gaussian variable y with mean $\bar{y}$ and variance σ_y^2, $\hat{z}_1 = \frac{1}{n}\sum_{j=1}^{n} z_{1j}$ and $\hat{z}_3 = \frac{1}{n}\sum_{j=1}^{n} z_{3j}$.

Also, we clearly have

$$\begin{aligned} f(z_2|H_i) &= f(z_{21}, \ldots, z_{2n}|H_i) \\ &= \prod_{j=1}^{n} [p_i \delta_{z_{2j} - \theta_1} + (1 - p_i) \delta_{z_{2jk} - \theta_0}] \end{aligned} \tag{4.12}$$

4.2.4 Classification by Bayesian Decision

The optimal Bayes decision minimizes the so-called Bayes risk

$$\bar{R}_D = \sum_{i,j} c_{ij} P\{\text{“}H_i\text{”}|H_j\} P\{H_j\}$$

which is a special case of (3.9). It decides on the hypothesis H_i with the smallest posterior cost, that is, $C_i(z) \leq C_l(z), \forall l$, where $C_k(z) = \sum_j c_{kj} P\{H_j|z\}$ and $z = (z_1, z_2, z_3)$.

In our example, we choose $c_{00} = c_{11} = 0$, $c_{01} = c_{10} = 1$. With this choice, the Bayes risk $\bar{R}_D$ becomes the probability of decision error

$$P_e = \sum_{i \neq j} P\{\text{“}H_i\text{”}|H_j\} P\{H_j\}$$

and thus the optimal Bayes decision becomes the minimum decision-error decision. It is well known that this amounts to maximum a posteriori (MAP) decision, which decides on the hypothesis having the maximum posterior probability.

It follows from Bayes' theorem that the posterior probabilities are

$$\begin{aligned} P\{H_0|z\} &= \frac{P\{H_0\} f(z|H_0)}{P\{H_0\} f(z|H_0) + P\{H_1\} f(z|H_1)} \\ P\{H_1|z\} &= 1 - P\{H_0|z\} \end{aligned} \tag{4.13}$$

While type 3 data z_3 is potentially helpful for decision (through some kind of estimation), it is not clear how it should be used in a purely decision setting. For example, if type 3 data is independent of the other types of data conditioned on x, as is the case for our problem, the likelihood ratio of H_1 to H_0 conditioned on x remains unchanged with or without type 3 data. As a result, in an implementation of the optimal Bayes decision, type 3 data is often ignored in practice (although it can actually be used to help decision) and only the first two types of data (z_1 and z_2) are used. Now let $z = (z_1, z_2)$ be the data used for decision.

Since each element in z_2 is Bernoulli distributed, to make decision on θ is actually to infer Bernoulli parameter p. It is known that $\hat{z}_2 = \frac{1}{n}\sum_{j=1}^{n} z_{2j}$ is a sufficient statistic for p [21]. Note that $n\hat{z}_2$ counts the number γ of z_{2j}s that equals 1. Thus, $n\hat{z}_2$ has a binomial (n, p) distribution:

$$P\{\hat{z}_2 = \gamma/n|H_i\} = \begin{pmatrix} n \\ \\ \gamma \end{pmatrix} p_i^{\gamma}(1-p_i)^{n-\gamma}$$

Thus, it follows from the conditional independence of type 1 and type 2 data that

$$\begin{aligned} f(z|H_i) &= f(z_1|H_i)f(z_2|H_i) = f(z_1|H_i)P\{\hat{z}_2 = \gamma/n|H_i\} \\ &= c\exp\left[-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(\hat{z}_1 - z_{1j})^2 - \frac{(\bar{z}_i - \bar{x})^2}{2(\sigma_{zi}^2 + \sigma_x^2)}\right] \\ &\cdot \begin{pmatrix} n \\ \\ \gamma \end{pmatrix} p_i^{\gamma}(1-p_i)^{n-\gamma} \end{aligned}$$

where $\sigma_{zi}^2 = \sigma_v^2/(n\theta_i^2)$ and $\bar{z}_i = \hat{z}_1/\theta_i$. By assuming equal priors of both hypotheses, e.g., $P\{H_0\} = P\{H_1\}$, from (4.13) it follows

$$\begin{aligned} P\{H_0|z\} &= \left[1 + \left(\frac{p_1}{p_0}\right)^{\gamma}\left(\frac{1-p_1}{1-p_0}\right)^{n-\gamma}\exp\left\{\frac{1}{2}S\right\}\right]^{-1} \\ P\{H_1|z\} &= 1 - P\{H_0|z\} \end{aligned} \tag{4.14}$$

where

$$S = \frac{(\bar{z}_0 - \bar{x})^2}{\sigma_{z0}^2 + \sigma_x^2} - \frac{(\bar{z}_1 - \bar{x})^2}{\sigma_{z1}^2 + \sigma_x^2}$$

The MAP decision decides on H_i if $P\{H_i|z\} > P\{H_j|z\}$. As such, the decision rule is

$$S + 2\ln\left[\left(\frac{p_1}{p_0}\right)^{\gamma}\left(\frac{1-p_1}{1-p_0}\right)^{n-\gamma}\right] \underset{H_1}{\overset{H_0}{\lessgtr}} 2\ln\frac{c_{10} - c_{00}}{c_{01} - c_{11}} \tag{4.15}$$

4.2.5 Tracking by Bayesian Estimation

The optimal Bayes estimator is the conditional mean $\hat{x} = E[x|z]$. By the total expectation theorem,

$$\hat{x} = E[x|z] = \sum_{i=0,1} E[x|z, H_i]P\{H_i|z\} = \sum_{i=0,1} \hat{x}_i P\{H_i|z\}$$

$$\text{MSE}(\hat{x}|z) = \sum_{i=0,1} \left[\text{MSE}(\hat{x}_i|z, H_i) + (\hat{x} - \hat{x}_i)(\hat{x} - \hat{x}_i)'\right] P\{H_i|z\}$$

By a similar argument as for the Bayesian decision, in Bayesian estimation, only the first type of data z_1 and third type of data z_3 are used directly and thus in the practical implementation, $z = [z_1', z_3']'$. Under each hypothesis data are normal distributed and from linear models $z = \mathbf{H}x + \upsilon$, where

$$H_0 : \quad \mathbf{H} = \mathbf{H}_0 = [\theta_0 \mathbf{1}', \mathbf{1}']'$$

$$H_1 : \quad \mathbf{H} = \mathbf{H}_1 = [\theta_1 \mathbf{1}', \mathbf{1}']'$$

and $\upsilon = [v', w']'$ with $R = \text{cov}(\upsilon) = \text{diag}(\sigma_v^2 I_n, \sigma_w^2 I_n)$. It is well known that for Gaussian distributions, the conditional mean and its MSE matrix are given by, under H_i,

$$\hat{x}_i = E[x|z, H_i] = \bar{x} + C_{xzi} C_{zi}^{-1} (z - \bar{z}_i)$$

$$\text{MSE}(\hat{x}_i|z, H_i) = C_x - C_{xzi} C_{zi}^{-1} C_{xzi}'$$

where

$$C_{xzi} = \sigma_x^2 \mathbf{H}_i' = \sigma_x^2 [\theta_i \mathbf{1}', \mathbf{1}']$$

$$C_{zi} = \mathbf{H}_i C_x \mathbf{H}_i' + R$$

$$= \sigma_x^2 [\theta_i \mathbf{1}', \mathbf{1}']' [\theta_i \mathbf{1}', \mathbf{1}'] + \text{diag}(\sigma_v^2 I_n, \sigma_w^2 I_n)$$

$$\bar{z}_i = [\theta_i \mathbf{1}', \mathbf{1}']', \quad C_x = \sigma_x^2$$

For equal prior probabilities of hypotheses, we have

$$\begin{aligned} P\{H_0|z\} &= \frac{P\{H_0\}f(z|H_0)}{P\{H_0\}f(z|H_0) + P\{H_1\}f(z|H_1)} \\ &= \frac{f(z|H_0)}{f(z|H_0) + f(z|H_1)} \\ P\{H_1|z\} &= 1 - P\{H_0|z\} \end{aligned}$$

where $f(z|H_i)$ was given by (4.11).

4.2.6 Classification before Tracking (Decision then Estimation)

In this traditional approach to JDE, a decision (target classification) is first made concerning the hypotheses H_0 and H_1, as in Sec. 4.2.4, and then the target state x is estimated, as in Sec. 4.2.5. More specifically, the data space is partitioned as $\{\mathcal{D}_0, \mathcal{D}_1\}$ by the decision rule first and the target state estimator is

$$\hat{x} = \sum_i 1(z; \mathcal{D}_i) E[x|z, H_i]$$

In other words, if the decision is "H_i", which follows Sec. 4.2.4, then the estimate is $\hat{x}_i = E[x|z, H_i]$ and $\text{MSE}(\hat{x}_i|z, H_i)$, given in Sec. 4.2.5.

4.2.7 Tracking before Classification (Estimation then Decision)

In this approach to JDE, the target state x is estimated first, as in Sec. 4.2.5, and then a decision (target classification) is made concerning the hypotheses H_0 and H_1, as in Sec.

4.2.4. More specifically, let $\hat{x}$ be the Bayes target state estimator of Sec. 4.2.5. Then

$$\begin{aligned} f(z|H_i) := f(z|H_i, \hat{x}) &= f(z_1|H_i, \hat{x}) f(z_2|H_i, \hat{x}) \\ &= \mathcal{N}(z_1; \theta_i \hat{x}\mathbf{1}, I) P\{\hat{z}_2 = \gamma/n|H_i\} \\ &= \frac{1}{(2\pi\sigma_v^2)^{n/2}} \exp\left\{-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(z_{1j} - \theta_i\hat{x})^2\right\} \\ &\cdot \begin{pmatrix} n \\ \\ \gamma \end{pmatrix} p_i^{\gamma}(1-p_i)^{n-\gamma} \end{aligned}$$

The Bayes test then decides on H_1 if

$$\frac{f(z|\hat{x}, H_1)}{f(z|\hat{x}, H_0)} > \lambda = \frac{(c_{10} - c_{00})P\{H_0\}}{(c_{01} - c_{11})P\{H_1\}}$$

Our choice $(c_{00}, c_{01}, c_{10}, c_{11}) = (0, 1, 1, 0)$ and equal prior probabilities of hypotheses $P\{H_0\} = P\{H_1\}$ lead to $\lambda = 1$. Then the test can be simplified as

$$\begin{aligned} n\,(\theta_1 - \theta_0)\,\hat{x}[\hat{z}_1 - (\theta_1 + \theta_0)\,\hat{x}/2] \\ + \ln\left[\left(\frac{p_1}{p_0}\right)^{\gamma}\left(\frac{1-p_1}{1-p_0}\right)^{n-\gamma}\right] \underset{H_1}{\overset{H_0}{\lessgtr}} 0 \end{aligned}$$

where the sample mean $\hat{z}_1 = \sum_{j=1}^{n} z_{1j}$.

The procedure described here is not the same as the widely-used GLRT since the estimate is an MMSE instead of MLE. However this is a common process in radar systems: perform tracking algorithms (say, IMM filter) then use the tracking results to help determine the target ID.

4.2.8 Joint Tracking and Classification

The first type of data can serve both tracking and classification purposes directly. The second type is useful for decision but has no direct impact on estimation; the third type is useful for

estimation but has no direct impact on decision. As a result, in the usual implementations of the traditional Bayes decision, Bayes estimation, and the two-stage approaches, one type of data is not used directly for either classification (decision) or tracking (estimation). However, our proposed joint approach uses all data without difficulty. Its performance is generally superior since more information is used.

The joint solution can be achieved by iteration. Although the iteration may start from any decision/estimation results, we choose the one with smaller JDE cost from the conventional solutions. For the choice of JDE cost weights $\{\alpha_{ij}, \beta_{ij}\}$ in (3.9), the values of α_{ij} will modify the decision cost in details, and generally speaking, we would like to choose $\beta_{01}, \beta_{10} < \beta_{00}, \beta_{11}$, which intends to take the estimation error costs into account (otherwise the generalized Bayes risk will be dominated by the cost associated with decision errors).

For simplicity, here our JDE algorithm starts from the Bayes decision results: If the decision is "H_i", i.e., $z \in \mathcal{D}_i$, from (3.12), we have

$$\begin{aligned}\hat{x} = \check{x}_i &= \bar{E}[x|z, D_i] \\ &= \sum_j E[x|z, D_i, H_j]\bar{P}\{H_j|z, D_i\} = \sum_j \hat{x}_j \bar{P}\{H_j|z\}\end{aligned}$$

where

$$\bar{P}\{H_j|z\} = \frac{\beta_{ij}P\{H_j|z\}}{\beta_{i0}P\{H_0|z\} + \beta_{i1}P\{H_1|z\}}, z \in \mathcal{D}_i$$

Then

$$\begin{aligned}\check{x}_i &= \hat{x}_0\bar{P}\{H_0|z\} + \hat{x}_1\bar{P}\{H_1|z\} \\ &= \frac{\hat{x}_0\beta_{i0}P\{H_0|z\} + \hat{x}_1\beta_{i1}P\{H_1|z\}}{\beta_{i0}P\{H_0|z\} + \beta_{i1}P\{H_1|z\}}\end{aligned}$$

To calculate the posterior JDE cost

$$R(z) = \sum_i \sum_j (c_{ij} + \beta_{ij} E[\tilde{x}'\tilde{x}|D_i, H_j, z]) P\{D_i, H_j|z\}$$

the key is to obtain the part imported by estimation. As derived in [54],

$$\begin{aligned} \text{mse}(\hat{x}|D_i, H_j, z) &\triangleq E[\tilde{x}'\tilde{x}|D_i, H_j, z] \\ &= \text{mse}(\hat{x}_{ij}|z, D_i, H_j) + (\hat{x}_{ij} - \hat{x})'(\hat{x}_{ij} - \hat{x}) \end{aligned}$$

where

$$\begin{aligned} \hat{x}_{ij} &= E[x|z, D_i, H_j] = E[x|z, H_j] \qquad \text{if } z \in \mathcal{D}_i \\ \text{mse}(\hat{x}_{ij}|z, D_i, H_j) &= \text{mse}(\hat{x}_{ij}|z, H_j) \qquad \text{if } z \in \mathcal{D}_i \end{aligned}$$

and $\hat{x}$ is obtained by (3.12). Since under $z \in \mathcal{D}_i$

$$\hat{x}_{ij} - \hat{x} = \hat{x}_j - \sum_i 1(z; \mathcal{D}_i)\check{x}_i = \hat{x}_j - \check{x}_i$$

we have

$$\begin{aligned} &\mathcal{E}_{ij} \triangleq \text{mse}(\hat{x}|D_i, H_j) \\ &= E[\text{mse}(\hat{x}_{ij}|z, D_i, H_j)|D_i, H_j] \\ &+ E[(\hat{x}_{ij} - \hat{x})'(\hat{x}_{ij} - \hat{x})|D_i, H_j] \\ &= \text{mse}(\hat{x}_{ij}|D_i, H_j) + E\left[(\hat{x}_j - \check{x}_i)'(\hat{x}_j - \check{x}_i)|D_i, H_j\right] \\ &= \text{mse}(\hat{x}_j|D_i, H_j) + E\left[(\hat{x}_j - \check{x}_i)'(\hat{x}_j - \check{x}_i)|D_i, H_j\right] \end{aligned}$$

For each decision region,

$$\begin{aligned} \text{mse}(\hat{x}_{ij}|z, D_i, H_j) &= \text{mse}(\hat{x}_j|z, H_j) \\ &= \sigma_x^2 - (\sigma_x^2)^2 \mathbf{H}_j' C_{zj}^{-1} \mathbf{H}_j, \text{ if } z \in \mathcal{D}_i \end{aligned}$$

Since it does not depend on observations, we have

$$\begin{aligned}\mathrm{mse}(\hat{x}_{ij}|D_i, H_j) &= E[\mathrm{mse}(\hat{x}_{ij}|z, D_i, H_j)] \\ &= \mathrm{mse}(\hat{x}_j|z, H_j)\end{aligned}$$

Note that under $z \in \mathcal{D}_i$

$$\begin{aligned}\hat{x}_j - \check{x}_i &= \hat{x}_j - \sum_k \hat{x}_k \frac{\beta_{ik} P\{H_k|z\}}{\sum_l \beta_{il} P\{H_l|z\}} \\ &= \frac{\sum_k (\hat{x}_j - \hat{x}_k)\beta_{ik} P\{H_k|z\}}{\sum_l \beta_{il} P\{H_l|z\}}\end{aligned}$$

Therefore

$$\begin{aligned}\tilde{\mathcal{E}}_{ij} &\triangleq E\left[(\hat{x}_j - \check{x}_i)'(\hat{x}_j - \check{x}_i)\,|D_i, H_j\right] \\ &= \int_{z\in\mathcal{D}_i} (\hat{x}_j - \check{x}_i)'(\hat{x}_j - \check{x}_i)\, dF(z|H_j) \\ &= E\left[\frac{\tilde{y}_i'\tilde{y}_i}{(\sum_l \beta_{il} P\{H_l|z\})^2}|D_i, H_j\right]\end{aligned}$$

where $\tilde{y}_i = \sum_k (\hat{x}_j - \hat{x}_k)\beta_{ik} P\{H_k|z\}$. It can be obtained by the Monte-Carlo method numerically

$$\tilde{\mathcal{E}}_{ij} \approx \frac{1}{L_i} \sum_{k=1}^{L_i} \left[\hat{x}_j(z_k^{(i)}) - \check{x}_i(z_k^{(i)})\right]' \left[\hat{x}_j(z_k^{(i)}) - \check{x}_i(z_k^{(i)})\right]$$

where L and L_i are the measurement counts in the Monte Carlo simulation: Generate data points $z_1, \ldots, z_L$ with the distribution $F(z|H_j)$ in the measurement space. Use the decision part to collect all the points $z_1^{(i)}, \ldots, z_{L_i}^{(i)}$ in $\mathcal{D}_i$, where $\sum_i L_i = L$.

Let $c'_{ij} = \alpha_{ij} c_{ij} + \beta_{ij} \mathcal{E}_{ij}$. If $c'_{10} > c'_{00}$ and $c'_{01} > c'_{11}$, for the next iteration, decide on H_1 if

$$S + 2\ln\left[\left(\frac{p_1}{p_0}\right)^{\gamma}\left(\frac{1-p_1}{1-p_0}\right)^{n-\gamma}\right] > 2\ln\frac{c'_{10} - c'_{00}}{c'_{01} - c'_{11}}$$

otherwise the decision rule is to decide on H_1 if

$$(c'_{01} - c'_{11}) \left(\frac{p_1}{p_0}\right)^{\gamma} \left(\frac{1-p_1}{1-p_0}\right)^{n-\gamma} e^{\frac{1}{2}S} > (c'_{10} - c'_{00})$$

The most straightforward stopping criterion is to check the difference of $\bar{R}$ in two adjacent iterations, which is not easy to calculate. Alternatively, we may stop the iteration if $\max_{i,j} |\mathcal{E}_{ij}^{(k)} - \mathcal{E}_{ij}^{(k+1)}|$ is below a threshold and there is no change in the decision (i.e., $z \in (\mathcal{D}_i^{(k)} \cap \mathcal{D}_i^{(k+1)})$).

4.2.9 Performance Evaluation

In this example,

$$\begin{aligned}
f(z_1|\theta, x) &= \mathcal{N}(z_1; \theta_i x\mathbf{1}, \sigma_v^2 I) \\
f(z_2|\theta, x) &= \prod_{j=1}^{n} [p_i \delta_{z_{2j}-\theta_1} + (1-p_i)\delta_{z_{2jk}-\theta_0}] \\
f(z_3|\theta, x) &= \mathcal{N}(z_3; x\mathbf{1}, \sigma_w^2 I) \\
f(z_1, z_2, z_3|\theta, x) &= f(z_1|\theta, x) f(z_2|\theta, x) f(z_3|\theta, x)
\end{aligned}$$

Let

$$\hat{f}(z|\hat{\theta}, \hat{x}) = f(z|\theta, x)|_{(\theta,x)=(\hat{\theta},\hat{x})}$$

Considering pmf, we have

$$\begin{aligned}
g(z|\theta_i, x) &= f_1(z_1|\theta, x) p_2(z_2|\theta, x) f_3(z_3|\theta, x) \\
&= \mathcal{N}(z_1; \theta_i x\mathbf{1}, \sigma_v^2 I) \mathcal{N}(z_3; x\mathbf{1}, \sigma_w^2 I) \\
&\cdot \prod_{j=1}^{n} [p_i \delta_{z_{2j}-\theta_1} + (1-p_i)\delta_{z_{2jk}-\theta_0}]
\end{aligned}$$

and

$$\hat{g}(z|\hat{\theta},\hat{x}) = \mathcal{N}(z_1;\hat{\theta}\hat{x}\mathbf{1},\sigma_v^2 I)\mathcal{N}(z_3;x,\sigma_w^2) \cdot \prod_{j=1}^{n}[\hat{p}_i\delta_{z_{2j}-\theta_1} + (1-\hat{p}_i)\delta_{z_{2jk}-\theta_0}]$$

4.3 Remarks

The results in the previous section are all derived using density functions which involve relatively tedious math. An alternative and better way is the following.

Under each hypothesis, type 1 and type 3 data, $z = [z_1', z_3']'$, follow a linear model $z = \mathbf{H}x + \upsilon$, where

$$H_0: \quad \mathbf{H} = [\theta_0\mathbf{1}', \mathbf{1}']'$$

$$H_1: \quad \mathbf{H} = [\theta_1\mathbf{1}', \mathbf{1}']'$$

and $\upsilon = [v', w']'$ with $R = \text{cov}(\upsilon) = \text{diag}(\sigma_v^2 I_n, \sigma_w^2 I_n)$. Under each hypothesis, x and z are jointly Gaussian because they are two weighted sums of jointly Gaussian random variables x and υ (since they are independent Gaussian). As such, z is Gaussian. Under H_i,

$$E[[z_1', z_3']'|H_i] = [\theta_i\mathbf{1}', \mathbf{1}']'\bar{x} = [\bar{z}_i', \bar{z}_3']'$$

$$\text{cov}([z_1', z_3']'|H_i) = [\theta_i\mathbf{1}', \mathbf{1}']'\sigma_x^2[\theta_i\mathbf{1}', \mathbf{1}'] + R$$

$$= \begin{bmatrix} \theta_i^2\sigma_x^2\mathbf{1}\mathbf{1}' + \sigma_v^2 I_n & \theta_i^2\sigma_x^2\mathbf{1}\mathbf{1}' \\ \theta_i^2\sigma_x^2\mathbf{1}\mathbf{1}' & \sigma_x^2\mathbf{1}\mathbf{1}' + \sigma_w^2 I_n \end{bmatrix}$$

$$= \begin{bmatrix} C_i & \theta_i^2\sigma_x^2\mathbf{1}\mathbf{1}' \\ \theta_i^2\sigma_x^2\mathbf{1}\mathbf{1}' & C_3 \end{bmatrix} = C_{i3}$$

where $\bar{z}_3 = \bar{x}\mathbf{1}$, $C_3 = \sigma_x^2\mathbf{1}\mathbf{1}' + \sigma_w^2 I_n$, and, for $i = 0, 1$,

$$\bar{z}_i = \theta_i\bar{x}\mathbf{1}, \quad C_i = \theta_i^2\sigma_x^2\mathbf{1}\mathbf{1}' + \sigma_v^2 I_n$$

It thus follows that

$$f(z_1, z_3|H_i) = \mathcal{N}([z_1', z_3']'; [\bar{z}_i', \bar{z}_3']', C_{i3}) \tag{4.16}$$

$$f(z_1|H_i) = \mathcal{N}(z_1; \bar{z}_i, C_i) \tag{4.17}$$

$$f(z_3|H_i) = \mathcal{N}(z_3; \bar{z}_3, C_3) \tag{4.18}$$

Note that clearly $f(z_1, z_3|H_i) \neq f(z_1|H_i)f(z_3|H_i)$.

Based on the above likelihood functions, the corresponding solutions can be derived more easily than using the method in the previous section. The details can be found in [56].

4.4 Simulation Results

In our simulation example, the following parameter values were used

$$\theta_0 = 1, \theta_1 = 2, p_0 = 0, p_1 = 0.65,$$

$$\bar{x} = 1, \sigma_x^2 = 0.5^2, \sigma_v^2 = 1, \sigma_w^2 = 0.5$$

Let data length $n = 10$, the simulation results are based on $M = 500$ Monte Carlo runs.

4.4.1 Scenario 1: Data generated from H_0

In this case, the simulated data is generated from H_0 (humans). The inference results are listed in Table 4.1.

To obtain the above results, the weights of JDE cost are chosen as $\alpha_{ij} = \alpha = 1$, $i, j = 0, 1$, and $\beta_{01} = \beta_{10}$, $\beta_{00} = \beta_{11}$. We add the constraint $\sum_i \beta_{ij} = B$ for comparison purpose. Notice that the maximum possible mse (all classification results are incorrect) is around 0.35, to balance the decision and estimation impact, we chose $B = 3$.

In the upper part of table, the RMSE is the root-mean-square errors of tracking the target state, and P_C is the probability of correct classification. Using the above two conventional performance metrics (RMSE and P_C) can only evaluate one aspect of the problem at a time. But how about the overall performance? The proposed JDE performance metric ρ can give us a quantitative measure. Let $N_i = 200$, $N_j = 5$, the results are shown in the lower part in Table 4.1.

Consider tracking errors. In the ideal case (always identify the target correctly) the root-mean square error is $\text{RMSE}_{\text{ideal}} = 0.1767$; for tracking without classification or tracking before classification, $\text{RMSE}_{\text{E}} = \text{RMSE}_{\text{E}\to\text{D}} = 0.2532$, which equals RMSE_{JDE} when $\beta_{ij}/\beta_{ii} = 1$, as they should be; for classification before tracking, $\text{RMSE}_{\text{D}\to\text{E}} = 0.2403$. This indicates that decision (classification) helps estimation (tracking) noticeably.

Consider classification performance now. For classification without tracking or perform classification before tracking, the probability of correct classification $P_C = 0.798$, which equals $(P_C)_{\text{JDE}}$ when $\beta_{ij}/\beta_{ii} = 1$, as they should be; for tracking before classification, $P_C = 0.894$. This indicates that estimation (tracking) also helps decision (classification).

These results for tracking and classification verify that in the H_0 case for this example, performing either decision or estimation will help the other. However, it is hard to compare the overall performance of the different strategies since none of the strategies is always better than the others in terms of both decision and estimation results. The JDE performance index ρ is extremely useful in such a case since we would like to have an overall rating output.

We calculated the ρ values for the two conventional strategies: For classification before tracking, $\rho_{\text{D}\to\text{E}} = 0.9976 \times 10^{-2}$; for tracking before classification, $\rho_{\text{E}\to\text{D}} = 1.0105 \times 10^{-2}$. As a distance measure, a smaller ρ value indicates a better performance. Therefore we conclude

Table 4.1: Simulation results in JDE solutions (truth is H_0)

β_{ij}/β_{ii}	0	10^{-3}	.01	.1	.5	1	2	10	∞
RMSE	0.1864	0.1865	0.1877	0.2042	0.2386	0.2532	0.2680	0.3030	0.3210
P_{D}	0.886	0.886	0.886	0.872	0.826	0.798	0.790	0.786	0.782
$\rho(\times 10^{-2})$	0.8327	0.8331	0.8401	0.8979	1.0700	1.1873	1.3114	1.5671	1.7051

that decision then estimation is indeed better than estimation then decision in terms of JDE performance.

By checking the ρ values of the proposed JDE solution listed in the table, we can see clearly that, the JDE solution with $\beta_{ij}/\beta_{ii} = 0$ which has the smallest ρ value outperforms the two existing strategies, although its decision result is worse than the tracking before classification strategy. This weight choice agrees with our intuition: β_{01}, β_{10} should be smaller than β_{00}, β_{11}.

Table 4.2: Simulation results in JDE solutions (truth is H_1)

β_{ij}/β_{ii}	0	10^{-3}	.01	.1	.5	1	2	10	∞
RMSE	0.1372	0.1372	0.1368	0.1365	0.1415	0.1470	0.1543	0.1769	0.3311
P_{D}	0.904	0.902	0.904	0.900	0.890	0.884	0.880	0.862	0.856
$\rho(\times 10^{-2})$	0.7809	0.7809	0.7810	0.7950	0.8391	0.8872	0.9496	1.1166	1.9436

4.4.2 Scenario 2: Data generated from H_1

In this case, the simulated data is generated from H_1 (moving vehicles). The inference results are listed in Table 4.2.

The weights of JDE cost are the same as those in Scenario 1. For the tracking errors, $\text{RMSE}_{\text{ideal}} = 0.1280$; $\text{RMSE}_{\text{E}} = \text{RMSE}_{\text{E}\to\text{D}} = 0.1470$; and $\text{RMSE}_{\text{D}\to\text{E}} = 0.1493$. This time, decision (classification) does not help estimation (tracking). For the classification results, for classification without tracking or classification before tracking, $P_C = 0.884$; for tracking before classification, $P_C = 0.882$. Estimation (tracking) does not help decision (classification) either. This indicates that in this case, the conventional strategies cannot utilize the coupling between decision and estimation well to improve the inference results.

By checking the upper part of the table, we found that the JDE solution with $\beta_{ij}/\beta_{ii} = 0$ outperforms the decision-then-estimation and estimation-then-decision in decision and estimation performance concurrently. Then we check the comprehensive performance index ρ: For classification before tracking, $\rho_{\text{D}\to\text{E}} = 0.8122 \times 10^{-2}$; for tracking before classification, $\rho_{\text{E}\to\text{D}} = 0.8740 \times 10^{-2}$. By comparing the corresponding ρ values of the JDE solution listed in the table, we can also draw the same conclusion that the JDE solution with $\beta_{ij}/\beta_{ii} = 0$ is better than the two existing strategies in terms of the overall performance.

4.5 Conclusions and Discussion

In this chapter, we applied the new proposed JDE framework on a joint target tracking and classification problem. The performance of the JDE solution was compared with two other existing strategies using both conventional methods and a joint performance index proposed

by us. With an appropriate weight choice of the JDE cost, the JDE solution outperforms the other two strategies by overall evaluation. Note that the weight choice of the JDE cost are problem dependent and subject to change based on the user's preference. For instance, if we want to pay more attention on estimation part, we should increase the values of β_{ij}s (or equivalently decrease α_{ij}s) to achieve better performance on estimation; and vice versa. For simplicity, the idea was illustrated by batch processing. The solution to the dynamic case is still under investigation.

Chapter 5

Vehicle Surveillance Testbed

5.1 Introduction

Many existing surveillance systems can be categorized into two classes: those using fixed (usually wired) sensing devices to cover a surveillance area [24, 38] and those using massively deployed wireless sensors to collaboratively collect data, communicate, and monitor the scene [92]. In recent years, advances in hardware and low-level software have made the second approach more powerful and cost effective than before [93]. The wired sensors have relatively high sensing accuracy with stable power supply, so they have been widely used in conventional radar and camera-based surveillance systems. However, due to the high cost and some physical/geometric restrictions on the deployment, those sensors are more or less fixed at certain locations. Therefore, data collected by them are usually limited in spatial coverage and nonadaptive to environmental change. On the other hand, deploying a large number of wireless sensors (due to the low cost and flexibility) will provide a large amount of data at the price of adding a significant portion of data processing to the system. The

bottleneck of a sensor network surveillance system mainly resides in the limited energy for each sensing node to perform sensing, data processing and communication for a long period of time. A more complete view of the sensing field can be obtained by exploiting the data from both wired and wireless sensors.

Deploying wireless sensor nodes to cover a large area uniformly is usually not a good choice. It is not only inefficient, but also hard to achieve accurate and robust performance: although each sensor node has low cost and low energy consumption, the total cost for collaborative sensing and data processing can be prohibitive. In addition, how to organize and maintain sensors into collaborative groups (decide which sensors should be invoked and how to propagate information to the appropriate nodes, etc.) in a large area is challenging largely due to the limited computaional/communication capability of the sensor nodes. In regard to the pros and cons of both approaches, a natural choice of system design is to have a combination of fixed (mostly wired) devices which persistently cover the region of interest and wireless sensors which can provide more detailed information as needed. To illustrate this, we are in the process of developing a surveillance testbed with a combination of both wired and wireless sensors to obtain multiresolution sensory data from the surveillance area. In the testbed, the targets of interest are remotely-controlled vehicles; sensor types include radar (range/speed), video cameras (wired/wireless), and many small, wireless acoustic/seismic/image sensors (Mica motes [3]) deployed around the indoor or outdoor areas.

One major surveillance task is to detect and track moving targets (e.g., vehicles, people, etc.). We refer to this task as *target inference* which includes not only target localization and tracking, but also target detection and recognition. Since data are from multiple

sensing devices, how to combine the sensed/processed data for the same sources is also an important problem. Such issues arise in track-to-track association (determining the origin of the track) and track-to-track fusion (obtaining the estimate based on data from multiple sources). In all these problems, decision and/or estimation (filtering) are the key elements [11, 12]. They are usually coupled, e.g., local track estimates will affect the decision on whether they are from a common origin; decision on target type will affect target motion model being used to estimate the position and velocity. Previous work mainly focuses on solving one problem conditioned on a solution of the other: "decision-then-estimation" or "estimation-then-decision." One of the major objectives of the development of our ground vehicle surveillance testbed, is to provide a practical base for the Bayes approach [54], which we discussed in previous two chapters, to the *joint decision and estimation (JDE)* problem. We will focus on its application to target inference problem and study the tradeoff between decision and estimation errors by designing different scenarios using the testbed. We will discuss how such a joint solution differs from the conventional approaches and its impact on the target inference with sensors of different types.

In a surveillance system using both wired and wireless sensors, how should networked sensors be integrated? This is a typical problem of information fusion. In simple words, *information fusion* is to combine information (data, decisions, estimates, identities, votes, etc.) from multiple sources (sensors or data processing nodes, etc.) to achieve better inference than could be achieved by the use of a single source [37]. In our experimental study, we first use only wireless sensors to do the inference, and high-resolution video cameras to construct the ground truth. We plan to evaluate the performance of the wireless sensors by comparing their inference results with the "ground truth" obtained by the video cameras. It is an

intermediate step in the development of the testbed. At this stage, we try to quantitatively analyze the capability and limitation of both types of sensing devices. The analysis results will be used as a guideline for parameter design in the integrated target inference, which is the next task of our testbed development.

5.2 Sensor Fusion with Practical Constraints

When different sensors/processors carry out JDE with local data, some inference results, e.g., state estimates, target types, may correspond to the same target, which leads to a (decision or estimation) fusion problem. Fusion techniques with various practical constraints have to be considered in real target inference problems. For a JDE problem, the data need be sent from each local processor to the fusion center. They include $\{\hat{x}_k, \text{mse}_k, D_k\}$, where $\hat{x}_k$ and mse_k are the estimate and mean square error, respectively, D_k is the decision result which could be hard decision (the hypothesis chosen) or soft decision (e.g., a probabilistic choice), all at a local processor k. The corresponding weight $\{\alpha_k, \beta_k\}$ chosen at each local processor should also be transmitted if the requirements of the task need to be tuned in real-time.

Compared with the conventional approaches, in which different local processors (for decision or estimation, separately) only need the information either for decision or estimation, the JDE solution requires more data flow at a time in the whole network. Finding efficient way to transmit data among local processors and the fusion center is another major task for the testbed development.

5.2.1 Data Fusion among Sensors of Different Types

Data fusion in a surveillance system using wireless sensor networks and one with video cameras (wired and wireless) poses quite a few challenges in both decision fusion and estimation fusion. Existing works [20, 59] mainly focused on how to organize distributed sensors (informative ones) into collaborative group, then perform localization techniques to initiate the tracks, and run filters to maintain them. This approach usually requires the geographic information of the deployed sensors, measurement synchronization, and relatively accurate resolution of sensors. Moreover, to achieve better performance, the computational load of the sensor management nodes will be fairly high due to the nonlinear nature of target motion [12]. Unlike previous work in the literature [20, 59, 93, 16], we would like to treat the measurements from networked wireless sensors as a type of data different from conventional sensor data. These data will be processed cooperatively with the conventional sensor data, rather than do the inference independently. Due to low sensor resolution and poor image registration, we do not use this type of data to track targets directly. Instead, for instance, they are used for feature extraction. By examining the spectrum of acoustic signals from wireless sensors, different types of vehicles can be discriminated. Without the proposed integrated JDE framework, this data can only help classification or recognition. Through our JDE framework, the resulting solution will have the potential to improve the performance of decision and estimation simultaneously.

5.2.2 Hierarchical Fusion

A large-scale target inference task can not be carried out in a fully centralized manner, which is inefficient, costly and vulnerable to the failure of the central node. On the other hand, it is also uncommon to implement the system in a "fully" distributed manner which will lead to a lot of data redundancy, heavy computational load and communication cost, especially when one faces geometry/physical constraints/limited regions of interest. In regard to the distributed nature of the data (from sensor networks), an on-demand procedure to determine the infrastructure of information fusion is preferred [10].

To be more specific, we expect the fully-developed testbed to work in this manner: The high-resolution video cameras consistently monitor the interested field, and the networked sensors remain in the energy-saving mode most of the time. The sensing nodes are divided into groups. Once any target is detected by one or more sensing devices in one group, the adjacent sensors in the same group will be invoked. The data collected at the nodes in the group will be sent to a local management unit. After distributed signal processing at each local management unit, the processed outputs (e.g., the probabilistic decision results, local estimates) will be collected at the fusion center. This stage-by-stage data processing procedure is what here "hierarchical fusion" stands for. By hierarchical data processing and fusion, we emphasize not only utilizing different types of sensing devices in an effective way but also choosing an appropriate architecture of the fusion procedure. The inference results will be improved in terms of accuracy, response time, and target acquisition range.

5.3 Target Surveillance Testbed with Networked Sensors

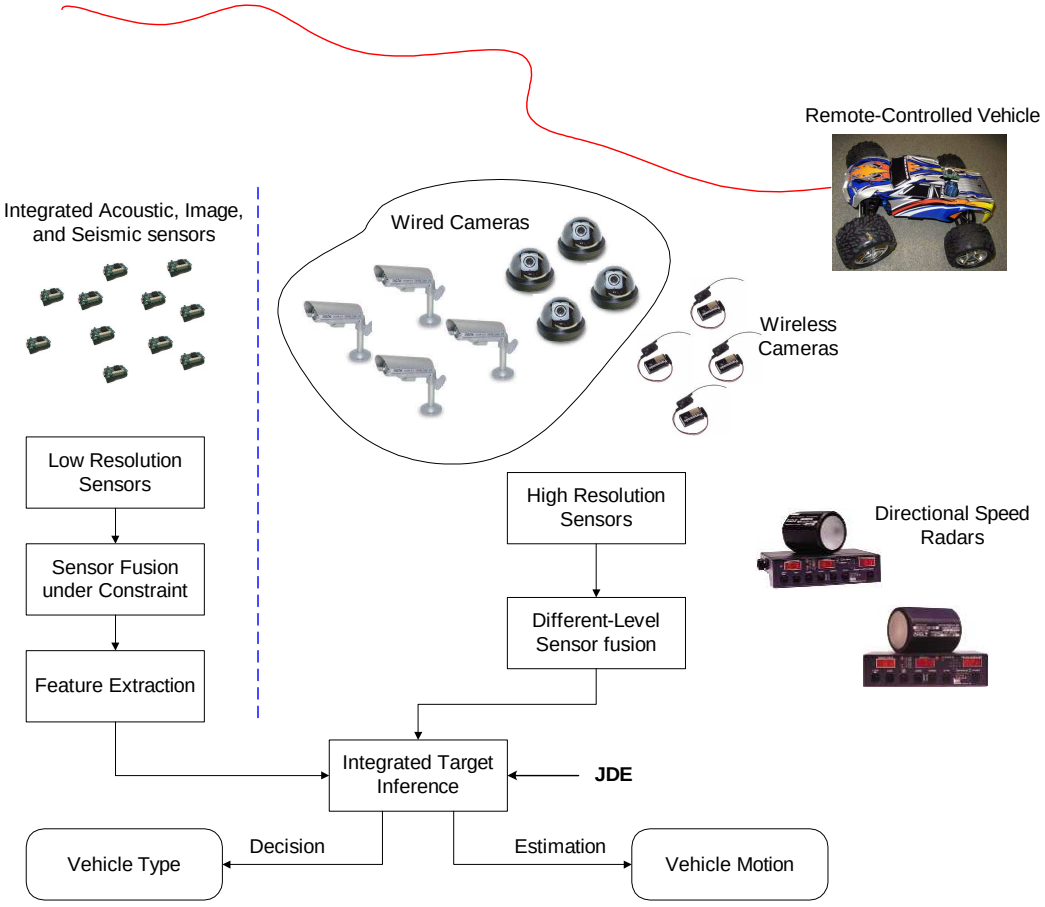


Figure 5.1: JDE with integrated target inference testbed

To address the challenges described in the previous sections, we have developed a target surveillance testbed for small-scale illustration of the automated vehicle detection and tracking using various algorithms. The testbed will allow the research in different areas including target information processing, integrated sensing and data fusion, network control, communication and computing systems. A unique feature of the testbed is the integration

of sensing and data processing in a dynamic network environment with multiple moving vehicles performing cooperative tasks. The equipment being used to develop the testbed system includes:

- Remote control vehicles: each of them works at one of 6 different control channels; the size is 1:10 comparing with real ones.
- Two desktop computers and two laptops: for collecting data and doing local processing.
- Two work stations: located in different rooms and used as fusion centers.
- Wireless sensor nodes and developing kits (Micaz Mote-Kit2400 [3]): the programming board is connected to a desktop via a serial-USB convertor.
- Overhead cameras and a surveillance center (currently built on a high performance desktop).
- Wireless cameras: the video streams are collected via TV tuner cards.
- Wireless routers (802.11b/g).
- Two speed radars: the effective range is up to 250 feet for the vehicles we use; the speed range is 1 to 100 mph; the radar data are sent to a desktop via serial (RS-232) ports.

As one possible configuration of the testbed, Figure 5.1 shows how the JDE framework is applied to the scenarios designed upon the target surveillance testbed with networked sensors. One or more remotely-controlled vehicles will be monitored by different sensors. The wired video cameras are mounted on fixed locations, recording video streams and directly

communicating to a desktop. The wireless video cameras are placed on fixed locations that are not convenient for wiring. The video data from wireless cameras are collected by a multi-channel wireless receiver connected to the desktop. The two speed radars (collecting two dimensional readings) are connected via serial ports of a PC. These sensors are used as high resolution sensors for inference. The Mica motes with acoustic sensors are used as low resolution sensors to be deployed into the experimental area. After fusing data under limited communication constraints (e.g., maximum rate of 76.8 kbps for each Mica mote), data of different types can be used in the JDE framework for integrated target inference. The final output of the system includes both the vehicle types and the fused state estimates related to the vehicle motion. With different maneuver motion scenarios, we want to illustrate how data of different types can be fully exploited through this integrated approach. For instance, Mica motes (only for decision without JDE) can help motion estimation, and speed radars (only for estimation without JDE) can improve the accuracy of target recognition.

Figure 5.3 illustrates a sample indoor placement: A vehicle moves along a straight line. On its path, there is a set of Micaz motes deployed to obtain the environment observations. Based on the sensory data, target detection, localization and tracking can be performed depending on the task requirement. In Figure 5.2, to obtain the details of vehicle dynamics, two Micaz motes were attached on the top of the vehicle.

5.4 Experimental Results

In this section, we present an illustrative scenario of vehicle detection and tracking with multiple sensors.

Figure 5.2: A moving vehicle with motes on top

5.4.1 Hardware Description

Micaz Motes

In these experiments, we used Micaz motes to construct the wireless sensor network. The Micaz has been developed by the researchers in the University of California, Berkeley and released by Crossbow, Inc. [3]. It is an open hardware and software platform for environment sensing and with support for plug-in sensor boards (see Figure 5.4). It is a 2.4 GHz, IEEE 802.15.4 compliant, Mote module used for enabling low-power, wireless, sensor networks. With enhancement on the overall functionality of Crossbow's Mica family of wireless sensor networking products, the Micaz Mote features several new capabilities [3]:

- IEEE 802.15.4/ZigBee compliant RF transceiver
- 2.4 to 2.4835 GHz ISM band

Figure 5.3: A vehicle moves along a straight line

- Direct sequence spread spectrum radio
- 250 kbps data rate.

Before the experiments, the Micaz Motes had been programmed with TinyOS firmware and ready to collect data periodically.

MTS310CA/MTS300CA Sensor Board

The sensor board attached on each Micaz is MTS310CA or MTS300CA, which are flexible sensor boards with a variety of sensing modalities. These modalities can be exploited in developing sensor networks for a variety of applications including vehicle detection, low-

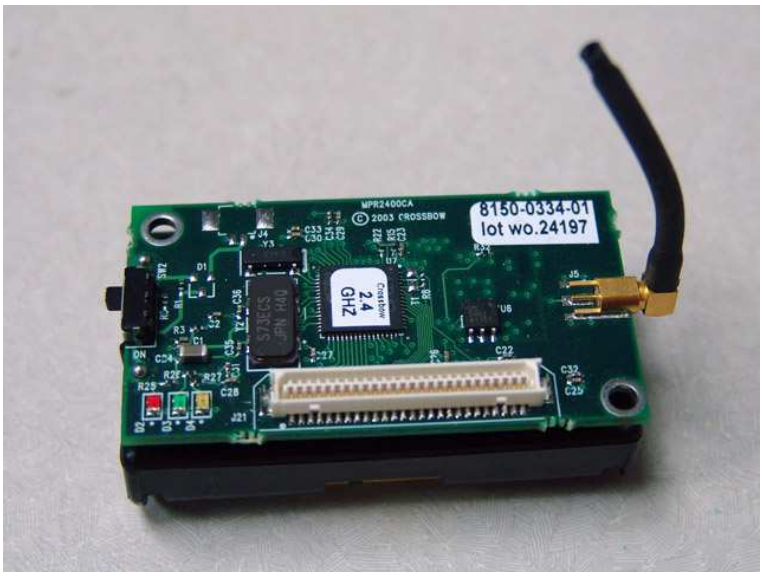


Figure 5.4: A Micaz mote

performance seismic sensing, movement, acoustic ranging, robotics, and other applications [3]. The detailed sensing modalities are illustrated in Figure 5.5.

Note the sounder is not a sensor, but an output. It is useful in unmanned safety and security systems.

The MTS310CA/MTS300CA sensor board has the following modalities:

- Microphone
- Light and Temperature
- 2-Axis Magnetometer (only for MTS310CA)
- 2-Axis Accelerometer (only for MTS310CA)

The sensor board is connected to the Micaz via the standard 51-pin expansion connector (refer to Figure 5.6). We only used microphone and light sensors for the current setting.

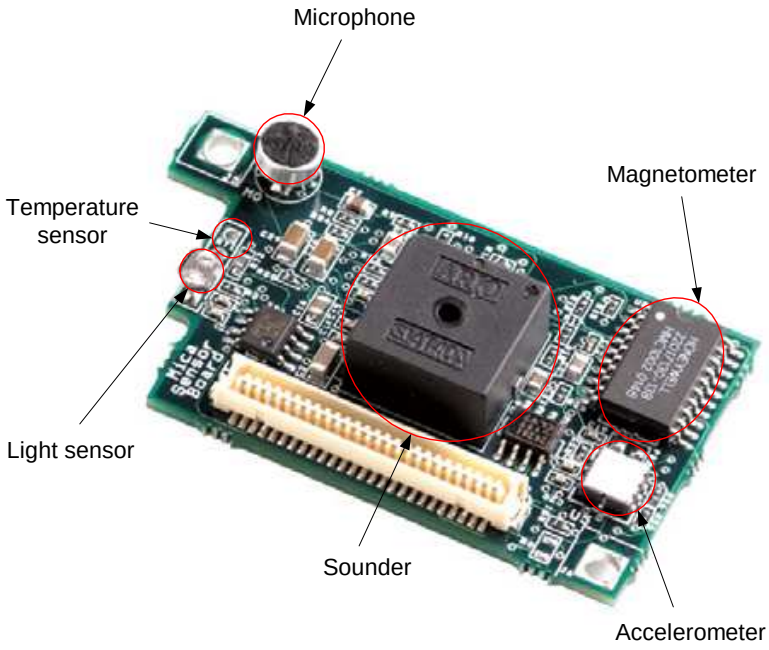


Figure 5.5: MTS310CA sensor board

The microphone works as a tone detector. And the light sensor is a simple CdSe photocell.

Wireless Sensor Network Gateway

The MIB510 gateway allows for the aggregation of sensor network data on a PC. In addition to data transfer, the MIB510 also provides an RS-232 serial programming interface. In our implementation, we used a serial-USB adapter to connect to a laptop. With an onboard processor, it is capable of programming Micaz and Mica2DOT processor radio boards. In programming, the USB port is identified as a com port. The data collection can be invoked by programming with TinyOS. There is also a GUI tool, MoteView (refer to Figure 5.7), available from Crossbow Inc. The Micaz motes can also be programmed using the MoteConfig in the GUI tool (refer to Figure 5.8) instead of typing in command line in TinyOS.

All the visualization tools in MoteView require being connected to a database. This

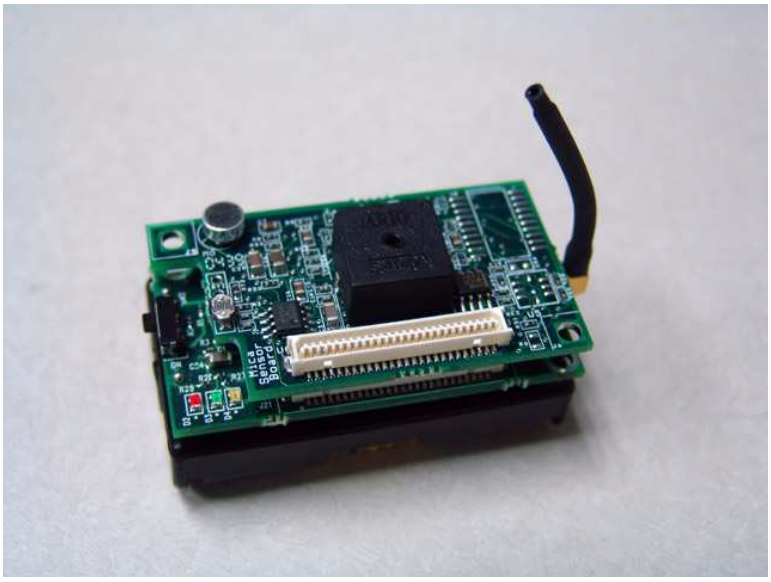


Figure 5.6: Micaz mote with sensor board attached

database can reside on the PC ("localhost"), or a remote server. The database used in our experiments is PostgreSQL 8.0. The tables stored in the database were dumped to text files, then further convert to excel files.

Figure 5.9 shows how the gateway connects other components: Besides the AC adapter and RS-232/serial cable (or serial-USB adapter), one Micaz mote, which was labeled as 0, attached on the top via the connector and acts as the base station. On the other side, a sensor board can also be attached.

Video Cameras

The video surveillance systems in our experiments was an EZWatch Pro system customized by Automated Video Systems [1]. The video cameras in use were SONY BU 581SRW color bullet cameras (see Figure 5.10).

The specifications of the video cameras are listed in Table 5.1. All the video cameras

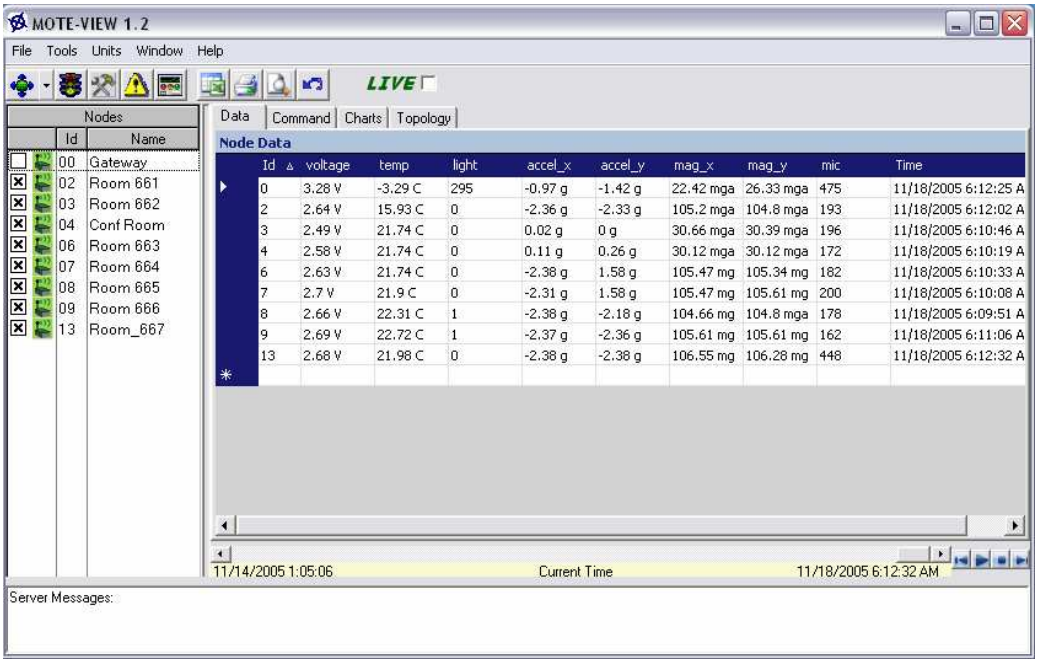


Figure 5.7: MoteView GUI tool

were connected to a desktop PC. Since they were all using the local time on the PC, there is no synchronization problem involved when collaboration is needed.

The trajectories obtained by video cameras are used to create the ground truth. The objective of the scenario is to detect the presence of the vehicle and find the locations at different time instants by using the measurements from wireless sensors. The ground truth obtained by the camera measurements will be used in performance evaluation.

5.4.2 Scenario Setup

The experiment was carried out in a dark room. The coverage area of all the sensing devices is about 20 ft $\times$ 4 ft. Five sensors are placed 4 feet apart along both sides of the coverage area. The placement of the sensors is shown in Figure 5.11. A remotely controlled vehicle with

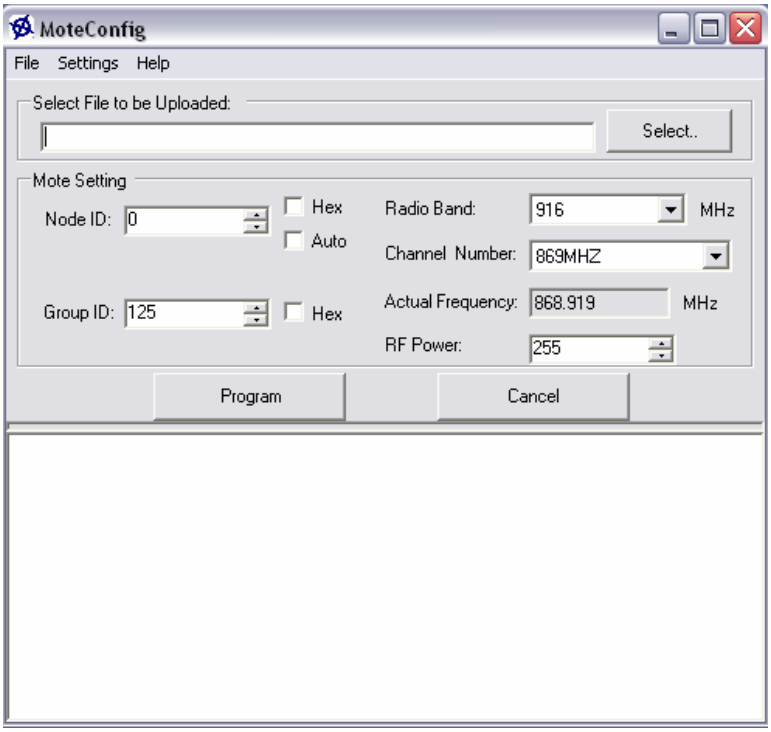
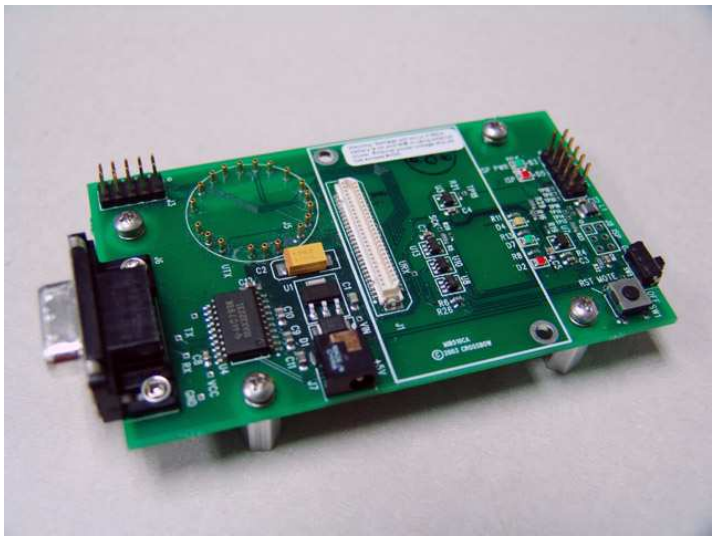


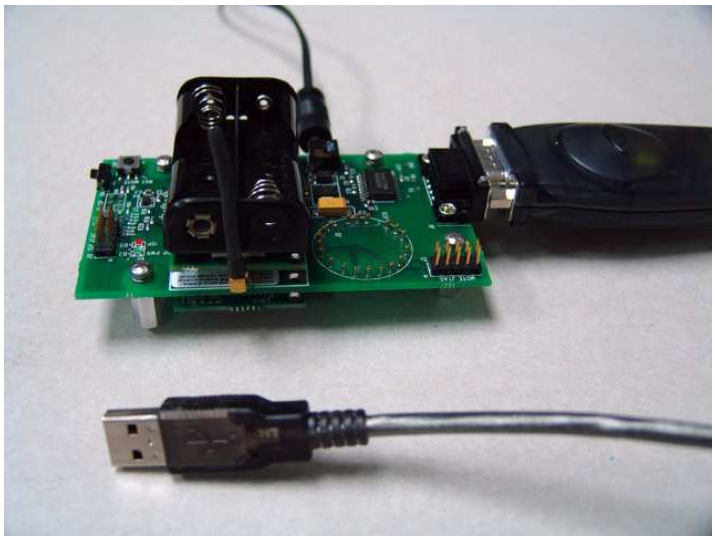
Figure 5.8: MoteConfig in MoteView GUI tool

two sensors and a flash light onboard was used. Three video cameras and a digital camera are used to provide the scenes of surveillance area and possibly the position information of the vehicle. The laptop enables the processing of the sensor programming board which is collecting sensing data from each Micaz mote. The three video cameras are placed at 0ft, 15ft and 23ft along x-axis respectively.

We intended to create a simple motion at a nearly constant velocity. However since the motion of the vehicle is controlled by a remote controller, it depends heavily on the operation of the remote controller with timely acceleration and deceleration. During the experiment, the vehicle moved roughly along straight lines both in the forward and the backward directions. The duration of the experiment is approximately 52.9 seconds.



(A) MIB510CA Programming Board



(B) MIB510CA with Micaz and sensor board

Figure 5.9: MIB510 serial gateway

Figure 5.10: BU 581SRW - SONY CCD bullet camera

5.4.3 Preliminary Sensor Data Processing

We used the light and acoustic sensors on the Micaz motes to detect the presence of the vehicle and estimate the locations periodically. Without any object passing by, the reading of each sensor (light or acoustic) is around a base value which can be treated as constant. Because of our objective of the scenario, we care more about the relative change in the reading, and therefore we did not perform strict calibration for light and acoustic sensors. When the vehicle with the flashlight passes the sensor (see Figure 5.12), with fairly high probability there is a significant change in the sensor reading. If this value exceeds a threshold value chosen as 20 in analog-to-digital converter (ADC) readings in the scenario, then we declare the vehicle's presence. Unfortunately, the sensor readings are not always accurate — there were cases of missed detection and false alarms.

Figure 5.13 shows all the measurements of one particular light sensor placed along the coverage area during the experiment. We can observe that there is one peak over the thresh-

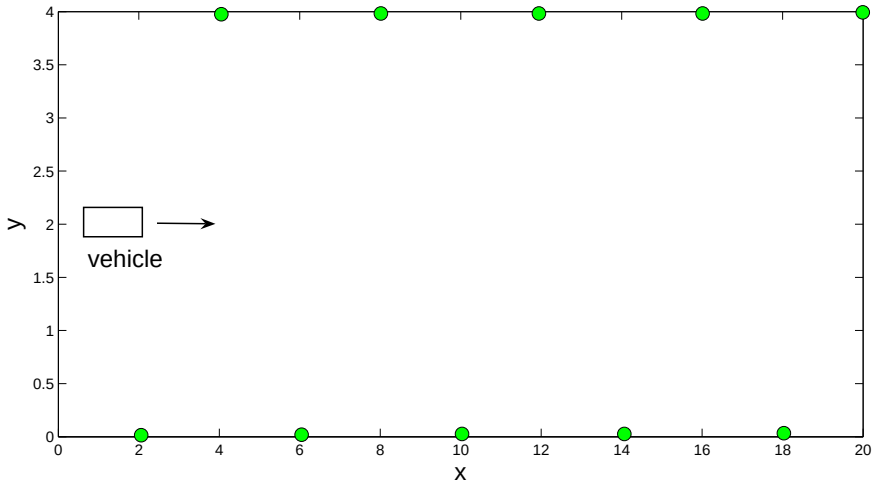


Figure 5.11: Sensor placement (units in feet)

old, which clearly indicates the vehicle's presence in the range of the sensor. However in some cases, the peak is either not large enough or is comparable with the noise level. In fact, sometimes there is no peak at all even when the vehicle passes by. By observing these phenomena, it is reasonable to construct a probabilistic measurement model which is a function of the sensing range. The parameters of the models need to be determined by a further study.

In our experiments, we also mounted two Micaz motes onboard to gain the knowledge of the true accelerations invoked by the remote controller, which is helpful to create the ground truth. However, the result is not so attractive: There is quite noticeable time delay between the sensor reading and the truth, and the delay time is unpredictable. This is mainly due to the limit of the transmission rate of the sensors. Meanwhile, there is also no clear way how the temperature sensor can serve our purpose. Therefore in our implementation, only the light and acoustic sensors are emphasized.

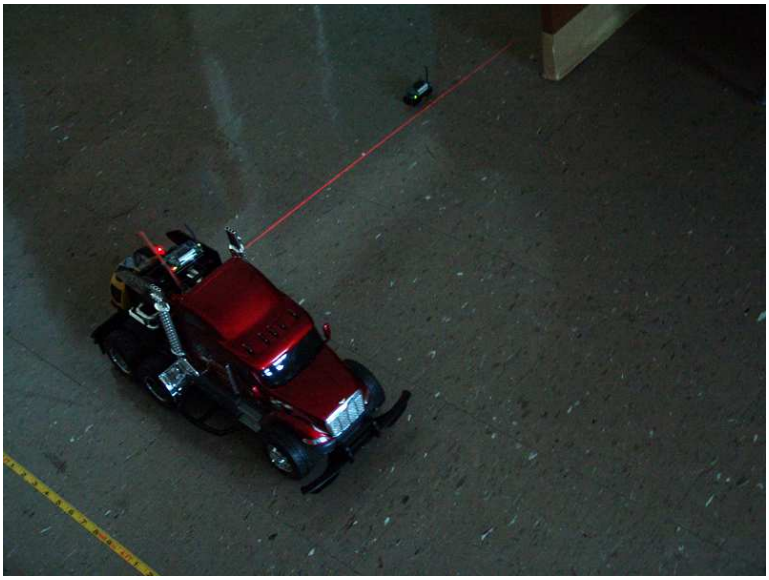


Figure 5.12: Vehicle passes by a mote node

5.4.4 Camera Calibration

Camera calibration is a process to estimate the intrinsic and/or extrinsic parameters of a camera given the values of the reference points in both the *image coordinates* and the *world coordinates*. For a single camera, a calibration procedure can be found in [82]. However, this method require many reference points (6 at least, the more the better, in practice people may use hundreds of reference points to achieve a very accurate result). In practice, if there is more than one camera in use, the required number of reference points can be significantly reduced. This is an interesting open topic in image processing and also important to the development of the testbed. However this is beyond the scope of the dissertation, and we will not go into technical details here.

In our experiment setup, we have three different cameras positioned at different locations, covering partially the scene for the duration of the experiment. From the video sequence

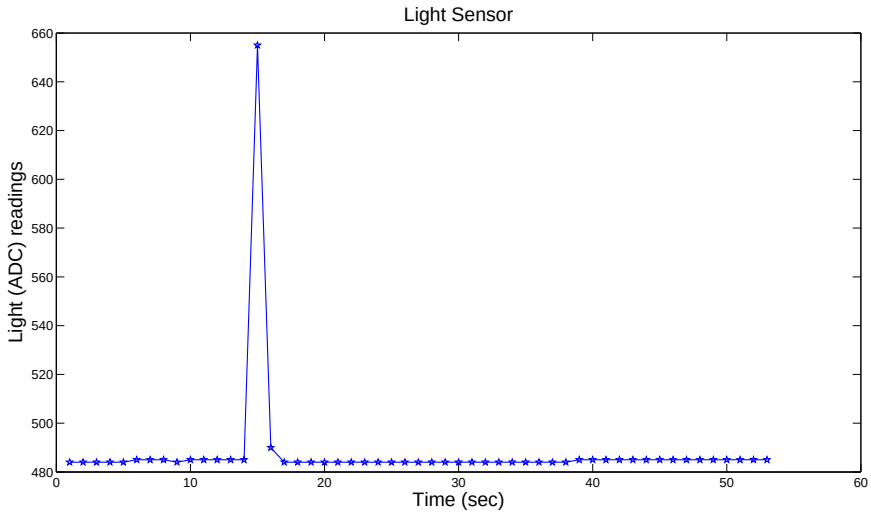


Figure 5.13: Measurements from a Light sensor for the entire experiment

obtained by each video camera, a set of image coordinates representing the centroids of the moving vehicle were calculated. The sampling rate is once per second.

The location and orientation of the three cameras with respect to the real world coordinate are measured prior to the experiments and they were assumed known perfectly. Two reference points with known image and world coordinates with respect to all three cameras are used to calculate their *intrinsic* and *extrinsic* parameters, which are used to obtain the mapping between the *image coordinates* and the *world coordinates*.

The plots in Figure 5.14 shows the trajectories formed by the image coordinates of the vehicle centroid for each camera. We can observe from the three plots that the vehicle moves more or less in a straight line.

Camera calibration can be performed individually following the procedure described in [82]. Figure 5.15 shows the calibration results for each individual camera. Note that not all 3 cameras can see the vehicle all the time.

Figure 5.14: Plots of vehicle centroid as observed from three cameras

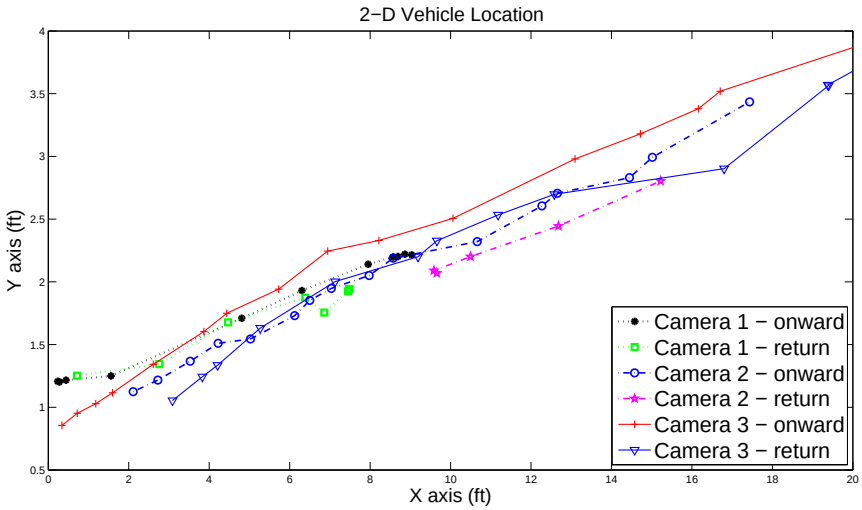


Figure 5.15: Calibration results for each individual camera

As we see from the above figure, the cameras are not very well calibrated using the single camera calibration method because the reference points are few.

To improve the accuracy of the estimates of the vehicles in the world coordinates from three cameras, we could use an iterative procedure to calibrate them. The basic idea of the procedure is described as follows.

Start from one arbitrary camera, calibrated individually using single camera calibration method with the two known reference points. This result is used to obtain the world coordinate values of several points. Then these points are used as additional reference points to calibrate other cameras. After the unknown parameters of the other cameras are obtained by each single camera calibration, the world coordinates of these points from different cameras are compared. If the result are quite different, the new world coordinates of the additional reference points are used to calibrate the previous camera. The values of camera parameters will be refined by iteratively repeating the above steps. The iterative procedure stops

when the corresponding values are close enough (by comparing the difference with a preset threshold). There is a multiple camera calibration example of implementing this idea in [87]. Figure 5.16 shows the calibration results for all three cameras.

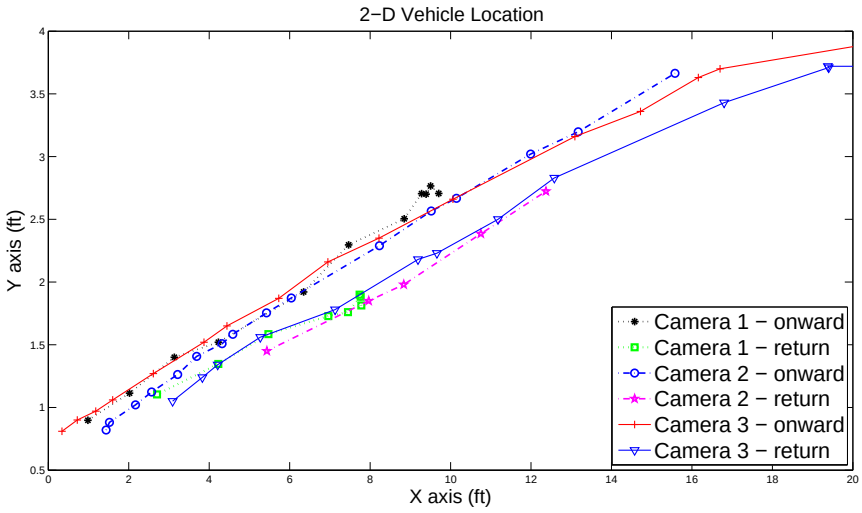


Figure 5.16: Calibration results for multiple cameras

To check the accuracy of the calibration, we compared the sample standard deviation $\hat{\sigma}$ of each estimated location for the cases before and after the calibration: Before calibration, the averaged $\hat{\sigma}$, denoted as $\bar{\sigma}$, is 0.84ft; after calibration, $\bar{\sigma} = 0.42$ft. If we look into the details for both axes, $\bar{\sigma}_x = 0.83$ft and $\bar{\sigma}_y = 0.10$ft before calibration, $\bar{\sigma}_x = 0.41$ft and $\bar{\sigma}_y = 0.08$ft after calibration. From Figure 5.16, we observed that after calibrating the cameras iteratively using the other calibrated cameras in a few steps, we still do not have a single trajectory that can be used to represent the ground truth of the vehicle's motion, which actually means the iteration should go on. To make it simple, we take the arithmetic average of the estimates from the three cameras at all the common time instants. The trajectory of the ground truth obtained after combining the estimates from the three cameras is shown in Figure 5.17,

where we use a 4th degree polynomial equation that best fits the given data set, i.e. the estimates of the vehicle location, to obtain a smooth representation of the vehicle's motion. The trajectory shown in the figure is close to a straight line, which agrees with what we observed.

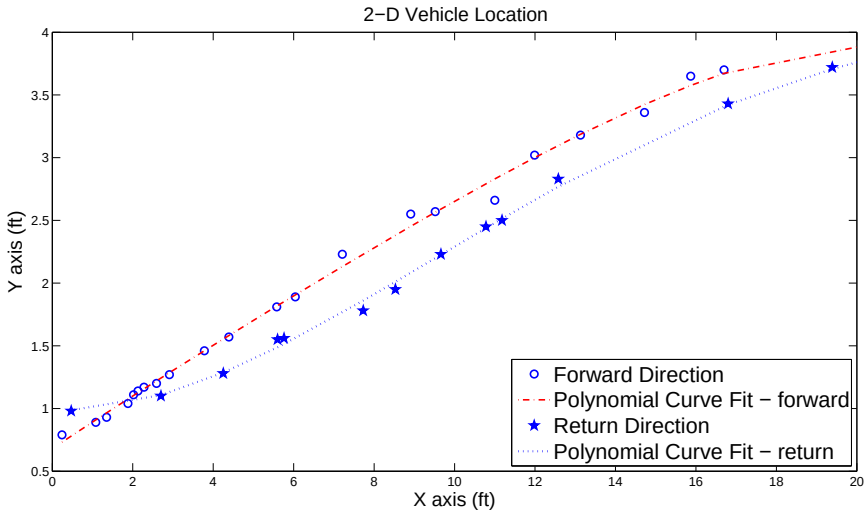


Figure 5.17: Vehicle locations estimated by video cameras

5.4.5 Localization by Wireless Sensors

As we show in the Figure 5.11, the coverage area has 10 sensors placed on both sides and at approximately 4 feet apart. Significant changes in the light and acoustic sensor readings are observed when the vehicle passed by the sensors correspondingly. Such changes in the sensor readings indicate the time instants (both forward and reverse directions) when the vehicle passes by the sensors. The vehicle locations can be estimated from the locations of the Micaz motes which have a significant increase in sensor readings (mic, light, etc) corresponding to the vehicle's presence in the vicinity of all the sensors.

Once we have the approximate time instants for both forward and return paths, the next step would be to describe the motion of the vehicle, and hence the location of the vehicle, at any particular time instant. With the available distance and time instants, one possible way to describe the vehicle motion in both forward and reverse directions is to fit to the existing data in the least-squares sense, as applied in [87].

The curve fitting process fits equations of approximating curves to the raw field data. Nevertheless, for a given set of data, the fitted curves of a given type are generally not unique. For simplicity, a curve with a minimal deviation from all data points is desired. This best-fitted curve can be obtained by the method of least squares. Detailed fitting results by the least squares were presented in [87]. The estimation error is about 0.77 ft on average. This error is relatively large compared with the sensing range (20 ft $\times$ 4 ft). One major reason is that the data rate is quite limited.

5.4.6 Remarks

As a testbed serving both decision and estimation purposes, only the estimation operation was described so far. Here the decision part is to determine whether there is a target present. This decision is coupled with the estimation presented in the previous subsection. Therefore we can treat these two parts as a JDE problem. In our implementation, we were following a conventional strategy, namely, decision-then-estimation, to solve this joint problem. The localization was done by assuming we always make the correct decision, i.e., the locations of the vehicle were estimated based on the known sensor locations once the target was declared present. The observations used for decision are the light readings. When one reading is over

the threshold, we declared there was a target present. The constant velocity model of the vehicle was assumed.

For the detection part, as we mentioned in the preliminary analysis, the peak in the sensor readings (if any) usually has a quite noticeable delay after the vehicle passes by. Apparently, only depending on these wireless sensors, the inference results are not quite satisfactory. The bottleneck here is the communication constraint of the mote sensors. There are two ways to improve: 1) Cooperate with high precision sensing devices by fusion techniques; 2) Implement the JDE framework to exploit the data for both purposes interactively and more effectively. In [56], we provided a solution to a joint target tracking and classification (JTC) example based on the JDE framework proposed in [54]. The assignment of the weight $\{\alpha_{ij}, \beta_{ij}\}$ in the Bayes risk was discussed by comparing the results in different situations of the example. Besides the conventional probability of correct decision and root mean squares errors for decision and estimation respectively, a comprehensive performance index for JDE was used to evaluate the performance of the JDE solution. Here we need to point out that our current decision-then-estimation strategy in the testbed implementation is a special case of the JDE setting. If $\beta_{ij} \equiv$ constant, the estimation will not have any impact on decision and the JDE solution degrades to this decision-then-estimation strategy. One hidden problem in [56] is that if we want to implement the algorithm on our testbed, we may need a new version of the comprehensive JDE performance index to evaluate the results, since the Monte Carlo method used there is time-consuming and may not be suitable for a real-time implementation. In [56], a general guideline for the weight assignment for the JTC problem was given based on the simulation results, which agrees with our intuition. If we can draw the same conclusion based on the testbed implementation, it is beneficial for our

theoretical development.

Our current solution is still centralized in nature. However if we treat the existing placement as a group of the system, it is possible to extend our work to a hierarchical setting.

Another problem is that the measurement model of the sensors used here is over simplified. A more meaningful probabilistic model, rather than the current energy-based binary indicator, is needed to improve the inference results. For instance, in [56], one data model with confusion matrix was assumed in the problem setting. We expect that through further experimental study based on JDE, more realistic measurement models and better design of the algorithms can be achieved. These are several important directions for the future development of our testbed.

5.5 Discussion and Conclusions

In this chapter, we have presented the development of a surveillance testbed with networked sensors. Based on this testbed, we have illustrated how to detect a moving vehicle and find its trajectory in the sensing field based on the change of lighting and acoustic readings from Micaz motes. In order to obtain the ground truth with better accuracy for algorithm comparison, three video sequences were collected via high-resolution video cameras. Each camera was calibrated with limited reference points. Although the presented example scenario only addressed data processing in different data types, the usage of the testbed is not limited to such simple problem settings. For instance, fusion with limited communication constraints can be addressed. The readings of the acoustic sensors used in the experiments are quite rough: they only act as tone detectors and the resolution is quite poor, sometimes even

when the vehicle passes by their adjacent area there are no readings at all. The sampling rate of the sensors is also too low compared with that of traditional devices (say, radars). By introducing more accurate and diverse sensing devices into the testbed, we should be able to attack more realistic problems in target inference. Because of the specific interest in the JDE problem setting, in the next step we will focus on the development of the JDE scenarios using the testbed. Another important future topic is to develop meaningful performance metrics to evaluate the outputs from various algorithms based on the moving vehicle scenarios.

Table 5.1: BU 581SRW - SONY CCD bullet camera

Image Sensor	Sharp 1/3" Color CCD Sensor
Horizontal Effective Pixels	768 pixel(s)
Vertical Effective Pixels	492 pixel(s)
CCD Size	1/3 inch
CCD Type	Color
Resolution	480 TV Line
Power Type	DC
Power Source	12V
Power Consumption	120 mA
Scanning Frequency	15.734Khz (H), 59.94Hz (V)
Scanning System	2:1 Interlace
Video Output	BNC / 75 Ohm
Lens Type	Standard
Lens	4mm Fixed Lens (85 degree view angle)
Diameter	30 mm
Height	100 mm
Weight	290 g (approx.)

Chapter 6

Summary and Future Work

The dissertation mainly focuses on solving statistical inference problems with both decision and estimation components in engineering applications.

A novel approach to model the Internet end-to-end delay dynamics using MM methods has been proposed. Although each model is LTI, overall the MM method provides a non-stationary, nonlinear solution. The proposed MM method performs better for prediction, in a highly non-stationary and nonlinear case (which corresponds to the Internet traffic at daytime), than two well known adaptive filters, namely, LMS and RLS.

An integrated approach to JDE based on proposed generalized Bayes risks has been studied in detail. Our goal is to provide a general systemic solution to this type of problems. One example in joint target tracking and classification was solved using the integrated approach. The performance of the inference results was evaluated using the conventional indices and a newly proposed comprehensive index for the JDE problem. A vehicle surveillance testbed with networked sensors for integrated target inference is being developed to build a connection between theory and practice.

In my future research work, there are several possible directions worth pursuing:

- Further extension of the JTC example. This topic is two fold: one is to apply our JDE framework in a similar setting to other signal processing and communication problems (for instance, user detection and parameter estimation, signal extraction and system identification); the other is to consider the dynamic case.

- Model selection. This dissertation started from an estimation-oriented application, and later mainly focused on a more or less balanced Bayes framework. One important part of the general JDE problems, inference with decision in primary place (or estimation-oriented inference), has not been studied in detail. It will be interesting to see what role the JDE Bayes risk can play in this type of questions. As one of the most well-known questions in this category, hopefully model selection can give us deeper understanding by applying this generalized risk.

- JDE performance evaluation in practical problems. As we addressed in the previous chapter, it will be beneficial for both theoretical development and experimental design of the testbed.

Appendix A

Likelihood Functions in JTC Example

Since

$$\begin{aligned} f(z_1|H_i) &= \int f(z_1|H_i, x) f(x) dx \\ &= \int \mathcal{N}(z_1; \theta_i x \mathbf{1}, \sigma_v^2 I) \mathcal{N}(x; \bar{x}, \sigma_x^2) dx \end{aligned}$$

where

$$\begin{aligned} \mathcal{N}(z_1; \theta_i x \mathbf{1}, \sigma_v^2 I) &= \frac{1}{(2\sigma_v^2)^{n/2}} \exp\left[-\frac{1}{2\sigma_v^2} \sum_{j=1}^{n} (z_{1j} - \theta_i x)^2 \right] \\ \mathcal{N}(x; \bar{x}, \sigma_x^2) &= \frac{1}{(2\sigma_x^2)^{1/2}} \exp\left[-\frac{(x - \bar{x})^2}{2\sigma_x^2} \right] \end{aligned}$$

therefore

$$\begin{aligned} -2 \ln \mathcal{N}(z_1; \theta_i x \mathbf{1}, \sigma_v^2 I) &= \frac{\theta_i^2}{\sigma_v^2} \sum_{j=1}^{n} (z_{1j}/\theta_i - x)^2 + c \\ &= \frac{1}{n\sigma_{zi}^2} \sum_{j=1}^{n} \left[(x - \bar{z}_i) + (\bar{z}_i - z_{1j}/\theta_i) \right]^2 + c \\ &= \frac{1}{\sigma_{zi}^2} \left[(x - \bar{z}_i)^2 + \frac{1}{n} \sum_{j=1}^{n} (\bar{z}_i - z_{1j}/\theta_i)^2 \right] + c \end{aligned}$$

$$-2\ln\mathcal{N}(x;\bar{x},\sigma_x^2)=\frac{1}{\sigma_x^2}(x-\bar{x})^2+\bar{c}$$

where c is a function of σ_v and $\bar{c}$ is a function of σ_x. Ignoring these two additive constants that do not depend on x or z,

$$\begin{aligned}
&-2\ln\left[\mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)\right]\\
&=\frac{1}{\sigma_x^2}(x-\bar{x})^2+\frac{1}{\sigma_{zi}^2}\left[(x-\bar{z}_i)^2+\frac{1}{n}\sum_{j=1}^{n}(\bar{z}_i-z_{1j}/\theta_i)^2\right]\\
&=\frac{1}{\sigma_x^2}(x^2-2x\bar{x}+\bar{x}^2)+\frac{1}{\sigma_{zi}^2}\left(x^2-2x\bar{z}_i+\bar{z}_i^2\right)+\frac{1}{n\sigma_{zi}^2}\sum_{j=1}^{n}(\bar{z}_i-z_{1j}/\theta_i)^2\\
&=\frac{1}{\sigma_x^2}(x^2-2x\bar{x}+\bar{x}^2)+\frac{1}{\sigma_{zi}^2}\left(x^2-2x\bar{z}_i+\bar{z}_i^2\right)+\frac{1}{\sigma_v^2}\sum_{j=1}^{n}(\theta_i\bar{z}_i-z_{1j})^2\\
&=\left(\frac{1}{\sigma_x^2}+\frac{1}{\sigma_{zi}^2}\right)x^2-2x\left(\frac{\bar{x}}{\sigma_x^2}+\frac{\bar{z}_i}{\sigma_{zi}^2}\right)+\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{1}{\sigma_v^2}\sum_{j=1}^{n}(\theta_i\bar{z}_i-z_{1j})^2 \qquad \text{(A.1)}
\end{aligned}$$

We define

$$\begin{aligned}
\sigma_{xi}^2&=\left(\frac{1}{\sigma_x^2}+\frac{1}{\sigma_{zi}^2}\right)^{-1}=\frac{\sigma_x^2\sigma_{zi}^2}{\sigma_x^2+\sigma_{zi}^2}\\
\hat{x}&=\sigma_{xi}^2\left(\frac{\bar{x}}{\sigma_x^2}+\frac{\bar{z}_i}{\sigma_{zi}^2}\right)
\end{aligned}$$

To complete square for first two terms in Eq. (A.1) we have

$$\left(\sigma_{xi}^2\right)^{-1}\left[x^2-2x\hat{x}+\hat{x}^2\right]-\left(\sigma_{xi}^2\right)^{-1}\hat{x}^2$$

therefore

$$\begin{aligned}
&-2\ln\mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)\\
&=\frac{(x-\hat{x})^2}{\sigma_{xi}^2}+\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}-\frac{\hat{x}^2}{\sigma_{xi}^2}+\frac{1}{\sigma_v^2}\sum_{j=1}^{n}(\theta_i\bar{z}_i-z_{1j})^2
\end{aligned}$$

After integration, the first term become constant, then

$$\int \mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)dx = c\exp\left[-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(\theta_i\bar{z}_i - z_{1j})^2 - \frac{1}{2}\left(\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}-\frac{\hat{x}^2}{\sigma_{xi}^2}\right)\right]$$
$$= c\exp\left[-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(\hat{z}_1 - z_{1j})^2 - \frac{1}{2}\left(\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}-\frac{\hat{x}^2}{\sigma_{xi}^2}\right)\right]$$

since

$$\begin{aligned}
\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}-\frac{\hat{x}^2}{\sigma_{xi}^2} &= \frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}-\left(\frac{\bar{x}}{\sigma_x^2}+\frac{\bar{z}_i}{\sigma_{zi}^2}\right)^2\frac{\sigma_x^2\sigma_{zi}^2}{\sigma_x^2+\sigma_{zi}^2}\\
&= \frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}-\frac{\left(\sigma_{zi}^2\bar{x}+\sigma_x^2\bar{z}_i\right)^2}{\left(\sigma_x^2+\sigma_{zi}^2\right)\sigma_x^2\sigma_{zi}^2}\\
&= \frac{\bar{x}^2\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)+\bar{z}_i^2\sigma_x^2\left(\sigma_x^2+\sigma_{zi}^2\right)-\left(\sigma_{zi}^2\bar{x}+\sigma_x^2\bar{z}_i\right)^2}{\sigma_x^2\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)}\\
&= \frac{\bar{x}^2\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)+\bar{z}_i^2\sigma_x^2\left(\sigma_x^2+\sigma_{zi}^2\right)-\left(\sigma_{zi}^2\right)^2\bar{x}^2-\left(\sigma_x^2\right)^2\bar{z}_i^2-2\sigma_{zi}^2\bar{x}\sigma_x^2\bar{z}_i}{\sigma_x^2\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)}\\
&= \frac{\bar{x}^2\left[\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)-\left(\sigma_{zi}^2\right)^2\right]+\bar{z}_i^2\left[\sigma_x^2\left(\sigma_x^2+\sigma_{zi}^2\right)-\left(\sigma_x^2\right)^2\right]-2\sigma_{zi}^2\bar{x}\sigma_x^2\bar{z}_i}{\sigma_x^2\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)}\\
&= \frac{\bar{x}^2\sigma_{zi}^2\sigma_x^2+\bar{z}_i^2\sigma_x^2\sigma_{zi}^2-2\sigma_{zi}^2\sigma_x^2\bar{x}\bar{z}_i}{\sigma_x^2\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)} = \sigma_{zi}^2\sigma_x^2\frac{\bar{x}^2+\bar{z}_i^2-2\bar{x}\bar{z}_i}{\sigma_x^2\sigma_{zi}^2\left(\sigma_x^2+\sigma_{zi}^2\right)}\\
&= \frac{\left(\bar{x}-\bar{z}_i\right)^2}{\sigma_x^2+\sigma_{zi}^2}
\end{aligned}$$

then the first likelihood is

$$\begin{aligned}
f(z_1|H_i) &= \int \mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)dx\\
&= c\exp\left[-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(\hat{z}_1 - z_{1j})^2 - \frac{\left(\bar{x}-\bar{z}_i\right)^2}{2\left(\sigma_x^2+\sigma_{zi}^2\right)}\right] \qquad \text{(A.2)}
\end{aligned}$$

Similarly, for the second likelihood,

$$f(z_3|H_i) = \int \mathcal{N}(z_3;x\mathbf{1},\sigma_w^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)dx$$

and

$$
\begin{aligned}
-2\ln\mathcal{N}(z_3; x\mathbf{1}, \sigma_w^2 I) &= \frac{1}{\sigma_w^2}\sum_{j=1}^{n}(z_{3j}-x)^2 + c' \\
&= \frac{1}{\sigma_w^2}\sum_{j=1}^{n}\left[(z_{3j}-\hat{z}_3)+(\hat{z}_3-x)\right]^2 + c' \\
&= \frac{1}{\sigma_w^2/n}\left[(x-\hat{z}_3)^2 + \frac{1}{n}\sum_{j=1}^{n}(z_{3j}-\hat{z}_3)^2\right] + c'
\end{aligned}
$$

where c' is a function of σ_w. Then ignoring the additive constants

$$
\begin{aligned}
&-2\ln\left[\mathcal{N}(z_3; x\mathbf{1}, \sigma_w^2 I)\mathcal{N}(x; \bar{x}, \sigma_x^2)\right] \\
&= \frac{1}{\sigma_x^2}(x-\bar{x})^2 + \frac{n}{\sigma_w^2}\left[(x-\hat{z}_3)^2 + \frac{1}{n}\sum_{j=1}^{n}(z_{3j}-\hat{z}_3)^2\right] \\
&= \frac{1}{\sigma_x^2}(x^2 - 2x\bar{x} + \bar{x}^2) + \frac{1}{\sigma_w^2/n}\left(x^2 - 2x\hat{z}_3 + \hat{z}_3^2\right) + \frac{1}{\sigma_w^2}\sum_{j=1}^{n}(z_{3j}-\hat{z}_3)^2 \\
&= \left(\frac{1}{\sigma_x^2} + \frac{1}{\sigma_w^2/n}\right)x^2 - 2x\left(\frac{\bar{x}}{\sigma_x^2} + \frac{\hat{z}_3}{\sigma_w^2/n}\right) + \frac{\bar{x}^2}{\sigma_x^2} + \frac{\hat{z}_3}{\sigma_w^2/n} + \frac{1}{\sigma_w^2}\sum_{j=1}^{n}(z_{3j}-\hat{z}_3)^2
\end{aligned}
\tag{A.3}
$$

After observing the similarity between Eq. (A.1) and Eq. (A.3), we replace σ_{zi}^2 in Eq. (A.1) by σ_w^2/n, $\bar{z}_i$ by $\hat{z}_3$, then the second likelihood is

$$
\begin{aligned}
f(z_3|H_i) &= \int \mathcal{N}(z_3; x\mathbf{1}, \sigma_w^2 I)\mathcal{N}(x; \bar{x}, \sigma_x^2)dx \\
&= c' \exp\left[-\frac{1}{2\sigma_w^2}\sum_{j=1}^{n}(\hat{z}_3 - z_{3j})^2 - \frac{(\hat{z}_3-\bar{x})^2}{2(\sigma_w^2/n + \sigma_x^2)}\right]
\end{aligned}
\tag{A.4}
$$

For the third likelihood

$$
f(z_1, z_3|H_i) = \int \mathcal{N}(z_1; \theta_i x\mathbf{1}, \sigma_v^2 I)\mathcal{N}(z_3; x\mathbf{1}, \sigma_w^2 I)\mathcal{N}(x; \bar{x}, \sigma_x^2)dx
$$

since

$$
\begin{aligned}
&-2\ln\mathcal{N}(z_1; \theta_i x\mathbf{1}, \sigma_v^2 I) \\
&= \frac{1}{\sigma_{zi}^2}\left[(x-\bar{z}_i)^2 + \frac{1}{n}\sum_{j=1}^{n}(\bar{z}_i - z_{1j}/\theta_i)^2\right] + c
\end{aligned}
$$

$$-2\ln\mathcal{N}(x;\bar{x},\sigma_x^2)=\frac{1}{\sigma_x^2}(x-\bar{x})^2+\bar{c}$$

$$-2\ln\mathcal{N}(z_3;x\mathbf{1},\sigma_w^2 I)=\frac{1}{\sigma_w^2}\sum_{j=1}^{n}(z_{3j}-x)^2+c'$$

Ignoring the additive constants that do not depend on x or z,

$$\begin{aligned}
&-2\ln\mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2 I)\mathcal{N}(z_3;x\mathbf{1},\sigma_w^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)\\
&=\frac{1}{\sigma_x^2}(x-\bar{x})^2+\frac{1}{\sigma_{zi}^2}\left[(x-\bar{z}_i)^2+\frac{1}{n}\sum_{j=1}^{n}(\bar{z}_i-z_{1j}/\theta_i)^2\right]+\frac{1}{\sigma_w^2}\sum_{j=1}^{n}\left[(x-\hat{z}_3)^2+(\hat{z}_3-z_{3j})^2\right]\\
&=\frac{1}{\sigma_x^2}(x^2-2x\bar{x}+\bar{x}^2)+\frac{1}{\sigma_{zi}^2}\left(x^2-2x\bar{z}_i+\bar{z}_i^2\right)+\frac{1}{n\sigma_{zi}^2}\sum_{j=1}^{n}(\bar{z}_i-z_{1j}/\theta_i)^2\\
&+\frac{1}{\sigma_w^2}(x^2-2x\hat{z}_3+\hat{z}_3^2)+\frac{1}{\sigma_w^2}\sum_{j=1}^{n}(\hat{z}_3-z_{3j})^2
\end{aligned}$$

Complete square for x

$$\begin{aligned}
&\frac{1}{\sigma_x^2}(x^2-2x\bar{x}+\bar{x}^2)+\frac{1}{\sigma_{zi}^2}\left(x^2-2x\bar{z}_i+\bar{z}_i^2\right)+\frac{1}{\sigma_w^2}(x^2-2x\hat{z}_3+\hat{z}_3^2)\\
&=\left(\frac{1}{\sigma_x^2}+\frac{1}{\sigma_{zi}^2}+\frac{1}{\sigma_w^2}\right)x^2-2x\left(\frac{\bar{x}}{\sigma_x^2}+\frac{\bar{z}_i}{\sigma_{zi}^2}+\frac{\hat{z}_3}{\sigma_w^2}\right)+\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}\\
&=\left(\sigma_{xi}^2\right)^{-1}\left[x^2-2x\hat{x}+\hat{x}^2\right]-\left(\sigma_{xi}^2\right)^{-1}\hat{x}^2+\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}
\end{aligned}$$

then

$$\begin{aligned}
&-2\ln\mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)\\
&=\frac{(x-\hat{x})^2}{\sigma_{xi}^2}+\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}-\frac{\hat{x}^2}{\sigma_{xi}^2}+\frac{1}{\sigma_w^2}\sum_{j=1}^{n}(\hat{z}_3-z_{3j})^2+\frac{1}{\sigma_v^2}\sum_{j=1}^{n}(\theta_i\bar{z}_i-z_{1j})^2
\end{aligned}$$

We define

$$\sigma_{xi}^2=\left(\frac{1}{\sigma_x^2}+\frac{1}{\sigma_{zi}^2}+\frac{1}{\sigma_w^2}\right)^{-1}$$

$$\hat{x}=\sigma_{xi}^2\left(\frac{\bar{x}}{\sigma_x^2}+\frac{\bar{z}_i}{\sigma_{zi}^2}+\frac{\hat{z}_3}{\sigma_w^2}\right)$$

After integration, the first term become constant, then

$$
\begin{aligned}
&\int \mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2 I)\mathcal{N}(z_3;x\mathbf{1},\sigma_w^2 I)\mathcal{N}(x;\bar{x},\sigma_x^2)dx \\
&= c'' \exp\left[-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(\theta_i\bar{z}_i - z_{1j})^2 - \frac{1}{2\sigma_w^2}\sum_{j=1}^{n}(\hat{z}_3 - z_{3j})^2 - \frac{1}{2}\left(\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}-\frac{\hat{x}^2}{\sigma_{xi}^2}\right)\right] \\
&= c'' \exp\left[-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(\hat{z}_1 - z_{1j})^2 - \frac{1}{2\sigma_w^2}\sum_{j=1}^{n}(\hat{z}_3 - z_{3j})^2 - \frac{1}{2}\left(\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}-\frac{\hat{x}^2}{\sigma_{xi}^2}\right)\right]
\end{aligned}
$$

where c'' is a function of σ_v and σ_w. Since

$$
\begin{aligned}
&\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}-\frac{\hat{x}^2}{\sigma_{xi}^2} \\
&= \frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}-\left(\frac{\bar{x}}{\sigma_x^2}+\frac{\bar{z}_i}{\sigma_{zi}^2}+\frac{\hat{z}_3}{\sigma_w^2}\right)^2\left(\frac{1}{\sigma_x^2}+\frac{1}{\sigma_{zi}^2}+\frac{1}{\sigma_w^2}\right)^{-1} \\
&= \frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}-\frac{\sigma_x^2\sigma_w^2\sigma_{zi}^2}{\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2}\cdot\frac{\left(\sigma_w^2\sigma_{zi}^2\bar{x}+\sigma_x^2\sigma_w^2\bar{z}_i+\sigma_x^2\sigma_{zi}^2\hat{z}_3\right)^2}{\left(\sigma_x^2\sigma_w^2\sigma_{zi}^2\right)^2} \\
&= \frac{\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\left(\sigma_w^2\sigma_{zi}^2\bar{x}^2+\sigma_x^2\sigma_w^2\bar{z}_i^2+\sigma_x^2\sigma_{zi}^2\hat{z}_3^2\right)-\left(\sigma_w^2\sigma_{zi}^2\bar{x}+\sigma_x^2\sigma_w^2\bar{z}_i+\sigma_x^2\sigma_{zi}^2\hat{z}_3\right)^2}{\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\sigma_x^2\sigma_w^2\sigma_{zi}^2}
\end{aligned}
$$

notice that

$$
\begin{aligned}
\left(\sigma_w^2\sigma_{zi}^2\bar{x}+\sigma_x^2\sigma_w^2\bar{z}_i+\sigma_x^2\sigma_{zi}^2\hat{z}_3\right)^2 &= \left(\sigma_w^2\sigma_{zi}^2\right)^2\bar{x}^2+\left(\sigma_x^2\sigma_w^2\right)^2\bar{z}_i^2+\left(\sigma_x^2\sigma_{zi}^2\right)^2\hat{z}_3^2 \\
&\quad + 2\sigma_w^2\sigma_{zi}^2\bar{x}\sigma_x^2\sigma_w^2\bar{z}_i + 2\sigma_x^2\sigma_w^2\bar{z}_i\sigma_x^2\sigma_{zi}^2\hat{z}_3 + 2\sigma_w^2\sigma_{zi}^2\bar{x}\sigma_x^2\sigma_{zi}^2\hat{z}_3
\end{aligned}
$$

and

$$
\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\sigma_w^2\sigma_{zi}^2\bar{x}^2-\left(\sigma_w^2\sigma_{zi}^2\right)^2\bar{x}^2 = \sigma_w^2\sigma_{zi}^2\left(\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\bar{x}^2
$$

$$
\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\sigma_x^2\sigma_w^2\bar{z}_i^2-\left(\sigma_x^2\sigma_w^2\right)^2\bar{z}_i^2 = \sigma_x^2\sigma_w^2\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_{zi}^2\right)\bar{z}_i^2
$$

$$
\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\sigma_x^2\sigma_{zi}^2\hat{z}_3^2-\left(\sigma_x^2\sigma_{zi}^2\right)^2\hat{z}_3^2 = \sigma_x^2\sigma_{zi}^2\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2\right)\hat{z}_3^2
$$

and for the numerator

$$\sigma_w^2\sigma_{zi}^2\left(\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\bar{x}^2+\sigma_x^2\sigma_w^2\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_{zi}^2\right)\bar{z}_i^2+\sigma_x^2\sigma_{zi}^2\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2\right)\hat{z}_3^2$$

$$-2\sigma_w^2\sigma_{zi}^2\bar{x}\sigma_x^2\sigma_w^2\bar{z}_i-2\sigma_x^2\sigma_w^2\bar{z}_i\sigma_x^2\sigma_{zi}^2\hat{z}_3-2\sigma_w^2\sigma_{zi}^2\bar{x}\sigma_x^2\sigma_{zi}^2\hat{z}_3$$

$$=\sigma_w^2\sigma_{zi}^2\sigma_x^2\sigma_w^2\bar{x}^2+\sigma_x^2\sigma_w^2\sigma_w^2\sigma_{zi}^2\bar{z}_i^2-2\sigma_w^2\sigma_{zi}^2\bar{x}\sigma_x^2\sigma_w^2\bar{z}_i$$

$$+\sigma_w^2\sigma_{zi}^2\sigma_x^2\sigma_{zi}^2\bar{x}^2+\sigma_x^2\sigma_{zi}^2\sigma_w^2\sigma_{zi}^2\hat{z}_3^2-2\sigma_w^2\sigma_{zi}^2\bar{x}\sigma_x^2\sigma_{zi}^2\hat{z}_3$$

$$+\sigma_x^2\sigma_w^2\sigma_x^2\sigma_{zi}^2\bar{z}_i^2+\sigma_x^2\sigma_{zi}^2\sigma_x^2\sigma_w^2\hat{z}_3^2-2\sigma_x^2\sigma_w^2\bar{z}_i\sigma_x^2\sigma_{zi}^2\hat{z}_3$$

$$=\sigma_w^2\sigma_{zi}^2\sigma_x^2\sigma_w^2(\bar{x}-\bar{z}_i)^2+\sigma_w^2\sigma_{zi}^2\sigma_x^2\sigma_{zi}^2(\bar{x}-\hat{z}_3)^2+\sigma_x^2\sigma_w^2\sigma_x^2\sigma_{zi}^2(\bar{z}_i-\hat{z}_3)^2$$

therefore

$$\begin{aligned}
&\frac{\bar{x}^2}{\sigma_x^2}+\frac{\bar{z}_i^2}{\sigma_{zi}^2}+\frac{\hat{z}_3^2}{\sigma_w^2}-\frac{\hat{x}^2}{\sigma_{xi}^2}\\
&=\frac{\sigma_w^2\sigma_{zi}^2\sigma_x^2\sigma_w^2(\bar{x}-\bar{z}_i)^2+\sigma_w^2\sigma_{zi}^2\sigma_x^2\sigma_{zi}^2(\bar{x}-\hat{z}_3)^2+\sigma_x^2\sigma_w^2\sigma_x^2\sigma_{zi}^2(\bar{z}_i-\hat{z}_3)^2}{\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)\sigma_x^2\sigma_w^2\sigma_{zi}^2}\\
&=\frac{\sigma_w^2(\bar{x}-\bar{z}_i)^2+\sigma_{zi}^2(\bar{x}-\hat{z}_3)^2+\sigma_x^2(\bar{z}_i-\hat{z}_3)^2}{(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2)}
\end{aligned}$$

the third likelihood is

$$\begin{aligned}
f(z_1,z_3|H_i)&=\int\mathcal{N}(z_1;\theta_i x\mathbf{1},\sigma_v^2I)\mathcal{N}(z_3;x\mathbf{1},\sigma_w^2I)\mathcal{N}(x;\bar{x},\sigma_x^2)dx\\
&=c''\exp\left[-\frac{1}{2\sigma_v^2}\sum_{j=1}^{n}(\hat{z}_1-z_{1j})^2-\frac{1}{2\sigma_w^2}\sum_{j=1}^{n}(\hat{z}_3-z_{3j})^2\right.\\
&\left.-\frac{\sigma_w^2(\bar{x}-\bar{z}_i)^2+\sigma_{zi}^2(\bar{x}-\hat{z}_3)^2+\sigma_x^2(\bar{z}_i-\hat{z}_3)^2}{2\left(\sigma_w^2\sigma_{zi}^2+\sigma_x^2\sigma_w^2+\sigma_x^2\sigma_{zi}^2\right)}\right]
\end{aligned}\tag{A.5}$$

Bibliography

[1] Automated Video Systems. http://www.ezwatchstore.com/.

[2] CAIDA: Cooperative Association for Internet Data Analysis. Available: http://www.caida.org/Tools/.

[3] Crossbow technology INC. http://www.xbow.com/.

[4] NetDyn: Monitoring tool used. Available: http://www.cs.umd.edu/ suman/netdyn/.

[5] H. Akaike. A new look at the statistical model identification. *IEEE Tran. on Automatic Control*, 19:716–723, 1974.

[6] M. Allman and V. Paxson. On estimating end-to-end network path properties. In *SIGCOMM*, pages 263–274, 1999.

[7] J. Andren, M. Hilding, and D. Veitch. Understanding end-to-end Internet traffic dynamics. In *Proc. IEEE GLOBECOM*, pages 1118–1122, Sydney, Australia, Nov. 1998.

[8] D. Angelova and L. Mihaylova. Sequential Monte Carlo algorithms for joint target tracking and classification using kinematic radar information. In P. Svensson and J. Schubert, editors, *Proceedings of the Seventh International Conference on Information Fusion*,

volume II, pages 709–716, Mountain View, CA, Jun 2004. International Society of Information Fusion.

[9] A. F. Atiya and A. G. Parlos. New results on recurrent networking training: Unifying the algorithms and accelerating convergence. *IEEE Trans. on Neural Networks*, 11(3):697–709, May 2000.

[10] Y. Bar-Shalom. On hierarchical tracking for the real world. *IEEE Trans. on Aerospace and Electronic Systems*, 42(3):846–850, July 2006.

[11] Y. Bar-Shalom and X. R. Li. *Multitarget-Multisensor Tracking: Principles and Techniques*. YBS, Storrs, CT, 1995.

[12] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation: Theory, Algorithms, and Software*. Wiley, New York, 2001.

[13] M. Barkat. *Signal Detection and Estimation*. Artech House, 2005.

[14] G. Beligiannis, L. Skarlas, and S. Likothanassis. A generic applied evolutionary hybrid technique. *IEEE Signal Processing Magazine*, 21(3):28– 38, May 2004.

[15] J. S. Bendat and A. G. Piersol. *Random Data: Analysis and Measurement Procedures*. John Wiley & Sons, Inc., 1971.

[16] P. K. Biswas and S. Phoha. A sensor network test-bed for an integrated target surveillance experiment. In *Proc. of the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*, Tampa, FL, Nov. 2004.

[17] Y. Boers and H. Driessen. Integrated tracking and classification: An application of hybrid state estimation. In *Proc. of SPIE Signal and Data Processing of Small Targets*, volume 4473, pages 198–209, 2001.

[18] J. Bolot. Characterizing end-to-end packet delay and loss in the Internet. *Journal of High Speed Networks*, 2(3):305–323, Dec. 1993.

[19] G. E. P. Box and G. M. Jenkins. *Time-Series Analysis: Forecasting and Control*. Holden Day, San Francisco, 1976.

[20] R. R. Brooks, P. Ramanathan, and A. M. Sayeed. Distributed target classification and tracking in sensor networks. *Proc. of the IEEE*, 91(8):1163 – 1171, Aug. 2003.

[21] G. Casella and R. L. Berger. *Statistical Inference*. Duxbury Press, Belmont, CA, 1990.

[22] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu. Network tomography: Recent developments. Statistical Science, Mar. 2003.

[23] S. Challa and G. W. Pulford. Joint target tracking and classification using radar and ESM sensors. *IEEE Trans. on Aerospace and Electronic Systems*, 37(3):1039–1055, 2001.

[24] S. Chaudhuri and D. R. Taur. High-resolution slow-motion sequencing: How to generate a slow-motion sequence from a bit stream. *IEEE Signal Processing Magazine*, 22(2):16–24, Mar. 2005.

[25] H. Chen, X. R. Li, and Y. Bar-Shalom. On joint track initiation and parameter estimation under measurement origin uncertainty. *IEEE Trans. on Aerospace and Electronic Systems*, 40(2):675–694, Apr. 2004.

[26] R. Chen, X. Wang, and J. S. Liu. Adaptive joint detection and decoding in flat-fading channels via mixture kalman filtering. *IEEE Trans. on Information Theory*, 46(6):2079–2094, Sep. 2000.

[27] V. Cherkassky and F. Mulier. *Learning from Data : Concepts, Theory, and Methods*. Wiley-Interscience, March 1998.

[28] K. Claffy, G. Polyzos, and H-W. Braun. Measurement Considerations for Assessing Unidirectional Latencies. *Internetworking: Research and Experience*, 4(3):121–132, Sep. 1993.

[29] D. D. Clark. Supporting realtime applications in an integrated services packet network: Architecture and mechanism. In *Proc. ACM Sigcomm'92*, pages 14–26, Baltimore, MD, Aug. 1992.

[30] M. Coates, A. O. Hero III, R. Nowak, and B. Yu. Internet tomography. *IEEE Signal Processing Magazine*, pages 47–65, May 2002.

[31] M. J. Coates and R. D. Nowak. Sequential Monte Carlo inference of internal delays in nonstationary data networks. *IEEE Trans. on Signal Processing*, 50(2):366–376, Feb. 2002.

[32] P. De Leon and C. J. Sreenan. An adaptive predictor for media playout buffering. In *Proc. ICASSP'99*, pages 3097–3100, Phoenix, AZ, Mar. 1999.

[33] N. G. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Multicast topology inference from measured end-to-end loss. *IEEE Trans. on Information Theory*, 48:26–45, Jan. 2002.

[34] A. Fredriksen, D. Middleton, and V. VandeLinde. Simultaneous signal detection and estimation under multiple hypotheses. *IEEE Trans. on Information Theory*, 18(5):607–614, Sep. 1972.

[35] N. Gordon, S. Maskell, and T. Kirubarajan. Efficient particle filters for joint tracking and classification. In *Proc. of SPIE Signal and Data Processing of Small Targets*, pages 439–449, 2002.

[36] D. Gross and C. M. Harris. *Fundamentals of Queueing Theory.* John Wiley & Sons, New York, NY, Third edition, 1998.

[37] D. L. Hall and J. Llinas. An introduction to multisensor data fusion. *Proc. of IEEE*, 85(1):6–23, Jan. 1997.

[38] A. Hampapur, L. Brown, J. Connell, A. Ekin, N. Haas, M. Lu, H. Merkl, and S. Pankanti. Smart video surveillance: Exploring the concept of multiscale spatiotemporal tracking. *IEEE Signal Processing Magazine*, 22(2):38–51, Mar. 2005.

[39] A. C. Harvey. *Time Series Models.* The MIT Press, Cambridge, Massachusetts, second edition, 1992.

[40] S. Herman and P. Moulin. A particle filtering approach to FM-band passive radar tracking and automatic target recognition. In *Proc. of the IEEE Aerospace Conference*, volume 4, pages 1789–1808, 2002.

[41] S. M. Herman. *A Particle Filtering Approach to Joint Passive Radar Tracking and Target Classification*. PhD thesis, Univ. of Illinois at Urbana-Champaign, 2002.

[42] S. P. Jacobs and J. A. O'Sullivan. High resolution radar models for joint tracking and recognition. In *IEEE International Radar Conference*, pages 99–104, Syracuse, NY, May 1997.

[43] V. Jacobson. Congestion avoidence and control. In *Proc. ACM Sigcomm'88*, pages 314–329, Stanford, CA, Aug. 1988.

[44] T. T. Kadota. Optimal, causal, simultaneous detection and estimation of random signal fields in a gaussian noise field. *IEEE Trans. On Information Theory*, IT-24(3):297–308, May 1978.

[45] L. Kleinrock. *Computer Applications*, volume 2 of *Queueing Systems*. Wiley-Interscience, New York, 1976.

[46] A. Kos, B. Klepec, and S. Tomazic. Techniques for performance improvement of VoIP applications. In *Proc. of 11th IEEE Mediterranean MELECON 2002*, pages 250–254, May 2002.

[47] D. G. Lainiotis. Partitioning: A unifying framework for adaptive systems I: Estimation. *Proc. IEEE*, 64(8):1126–1143, 1976.

[48] A. D. Lanterman. Tracking and recognition of airborne targets via commercial television and FM radio signals. In *Proc. of SPIE 3692 Conference on Acquisition, Tracking, and Pointing XIII*, Orlando, FL, Apr. 1999.

[49] Q. Li and D. L. Millis. Jitter-basd delay-boundary prediction of wide-area networks. *IEEE Trans. on Networking*, 9(5):578–590, Oct. 2001.

[50] X. R. Li. Hybrid estimation techniques. In C. T. Leondes, editor, *Control and Dynamic Systems: Advances in Theory and Applications*, volume 76, pages 213–287, San Diego, 1996. Academic Press.

[51] X. R. Li. Engineer's guide to variable-structure multiple-model estimation for tracking. In Y. Bar-Shalom and W.D. Blair, editors, *Multitarget-Multisensor Tracking: Applications and Advances*, volume 3, pages 499–567, Boston, 2000. Artech House.

[52] X. R. Li. *Applied Estimation and Filtering.* University of New Orleans, New Orleans, LA, 2002.

[53] X. R. Li. Information and Systems Laboratory internal seminar, University of New Orleans, 2006.

[54] X. R. Li. Optimal Bayes joint decision and estimation. In *Proc. of the 10th Int. Conf. on Information Fusion*, Québec, 2007. to appear.

[55] X. R. Li and V. P. Jilkov. Survey of maneuvering target tracking – Part V: Multiple-model methods. *IEEE Trans. on Aerospace and Electronic Systems*, 41(4):1255–1321, Oct. 2005. Available: `http://ece.engr.uno.edu/isl/MTTSurveys.htm`.

[56] X. R. Li, M. Yang, and J.-F. Ru. Joint tracking and classification based on Bayes joint decision and estimation. In *Proc. of the 10th Int. Conf. on Information Fusion*, Québec, 2007. to appear.

[57] X. R. Li, Z.-L. Zhao, and X. B. Li. General model-set design methods for multiple-model approach. *IEEE Trans. on Automatic Control*, 50(9):1260–1276, Sep. 2005.

[58] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Trans. on Communications*, pages 702–710, Jan. 1980.

[59] J. J. Liu, J. Liu, J. Reich, P. Cheung, and F. Zhao. Distributed group management for track initiation and maintenance in target localization applications. In *Proc. of 2nd International Workshop on Information Processing in Sensor Networks (IPSN'03)*, Apr. 2003.

[60] L. Ljung. *System Identification – theory for the user*. N.J.: Prentice Hall, Englewood Cliffs, 1987.

[61] S. H. Low, F. Paganini, and J. C. Doyle. Internet congestion control. *IEEE Control Systems Magazine*, 22(1):28–43, Feb. 2002.

[62] D. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[63] S. Makridakis. A survey of time series. *Int. Stat. Rev.*, 44(1):29–70, 1976.

[64] W. Mei, G.-L. Shan, and X. R. Li. An efficient bayesian algorithm for joint target tracking and classification. In *Proc. of the Seventh International Conference on Signal Processing*, Beijing, China, Aug. 31 - Sept. 4 2004.

[65] D. Middleton. *An Introduction to Statistical Communication Theory*. McGraw-Hill, New York, 1960.

[66] D. Middleton and R. Esposito. Simultaneous optimum detection and estimation of signals in noise. *IEEE Trans. on Information Theory*, IT-14(3):434–444, May 1968.

[67] M. I. Miller, A. Srivastava, and U. Grenander. Conditional-mean estimation via jump-diffusion process in multiple target tracking/recognition. *IEEE Trans. on Signal Processing*, 43(11):2678–2690, 1995.

[68] D. Morato, J. Aracil, L. A. Diez, M. Izal, and E. Magana. On linear prediction of Internet traffic for packet and burst switching networks. In *Proc. Of ICCCN 2001*, pages 138 –143, Oct. 2001.

[69] H. Ohsaki, M. Morita, and M. Murata. On modeling round-trip time dynamics of the Internet using system identification. In *The 16th International Conference on Information Networking (ICOIN-16)*, Jan. 2002.

[70] H. Ohsaki, M. Murata, and H. Miyahara. Modeling end-to-end packet delay dynamics of the Internet using system identification. In *Proceedings of the International Teletraffic Congress 17*, pages 1027–1038, Dec. 2001.

[71] J. A. O'Sullivan, S. P. Jacobs, M. I. Miller, and D. L. Synder. A likelihood-based approach to joint target tracking and identification. In *Proc. of Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, pages 290–294, Nov. 1993.

[72] A. G. Parlos. Identification of the Internet end-to-end delay dynamics using multi-step neuro-predictors. In *Proc. Of the 2002 Int. Joint Conf. On Neural Networks, IJCNN '02*, Honolulu, HI, May 2002.

[73] V. Paxson. End-to-end Internet packet dynamics. *IEEE/ACM Transactions on Networking*, 7(3):277–292, 1999.

[74] M. Pourahmadi. *Foundations of Time Series Analysis and Prediction Theory*. John Wiley & Sons, Inc., 2001.

[75] N. S. V. Rao. Overlay networks of in-situ instruments for probabilistic guarantees on message delays in wide-area networks. *IEEE Journal on Selected Areas in Communications*, 22(1):79–90, Jan. 2004.

[76] J. Rissanen. Universal coding, information, prediction, and estimation. *IEEE Trans. on Information Theory*, (4):629–636, July 1984.

[77] A. Sang and S.-Q. Li. A predictability analysis of network traffic. In *INFOCOM (1)*, pages 342–351, 2000.

[78] G. Schwartz. Estimating the dimension of a model. *Annals of statistics*, 6(461-464), 1978.

[79] P. Smets and B. Ristic. Kalman filter and joint tracking and classification in the TBM framework. In P. Svensson and J. Schubert, editors, *Proceedings of the Seventh International Conference on Information Fusion*, volume I, pages 46–53, Mountain View, CA, Jun 2004. International Society of Information Fusion.

[80] C. J. Sreenan, J.-C. Chen, P. Agrawal, and B. Narendan. Delay deduction techniques for playout buffering. *IEEE Trans. on Multimedia*, 2(2):88–100, June 2000.

[81] E. J. Sullivan and D. Middleton. Estimation and detection in matched field processing. In *Proc. of IEEE ICASSP*, volume III, pages 70–74, 1993.

[82] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, Mar. 1998.

[83] Y. Tsang, M. Coates, and R. D. Nowak. Network delay tomography. *IEEE Trans. on Signal Processing*, 51(8):2125– 2136, Aug. 2003.

[84] Y. Vardi. Network tomography: Estimating source-destination traffic intensities from link data. *J. Amer. Stat. Assoc.*, 91(433):365–377, 1996.

[85] P. K. Varshney and A. H. Haddad. A receiver with memory for fading channel. *IEEE trans. on Communications*, COM-26(2):278–283, Feb. 1978.

[86] J. W. Wong. Queueing network modeling of computer communication networks. *Computing Surveys*, 10(3):343–351, Sept. 1978.

[87] M. Yang, H. Chen, S. Bandarupalli, and X. R. Li. A surveillance testbed with networked sensors for integrated target inference. In *Proc. of IEEE TridentCom 2007*, Orlando, 2007. to appear.

[88] M. Yang and X. R. Li. Predicting end-to-end delay of the Internet using time series analysis. Technical report, University of New Orleans, Lakefront, Nov. 2003. Available: `http://ece.engr.uno.edu/isl/Yang/`.

[89] M. Yang, X. R. Li, H. Chen, and N. S. V. Rao. Predicting Internet end-to-end delay: An overview. In *Proc. of 36th IEEE Southeastern Symposium on Systems Theory*, pages 210–214, Atlanta, Mar. 2004.

[90] M. Yang, J.-F. Ru, H. Chen, A. Bashi, X. R. Li, and N. S. V. Rao. Predicting Internet end-to-end delay: A statistical study. *Annual Review of Communications*, Vol. 58:665–677, 2005.

[91] M. Yang, J.-F. Ru, X. R. Li, H. Chen, and A. Bashi. Predicting Internet end-to-end delay: A multiple-model approach. In *Proc. of IEEE INFOCOM 2005*, volume 4, pages 2815–2819, Miami, Mar. 2005.

[92] F. Zhao and L. Guibas. *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann Publishers, May 2004.

[93] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich. Collaborative signal and information processing: An information directed approach. *Proc. of the IEEE*, 91(8):1199–1209, 2003.

VITA

Ming Yang received the B.S. degree from Peking University, Beijing, China, in 1997, and M.S. degree from the Institute of Acoustics, Chinese Academy of Sciences (IAAS), Beijing, China, in 2000, both in Electrical Engineering.

From Oct. 1996 to Oct. 2000, he was a research assistant in the Speech & Communication Laboratory, IAAS. His research was mainly on speech recognition, speech synthesis, speech coding and communications, including algorithm development and DSP real-time implementation. Since Aug. 2001 he has been a research assistant in University of New Orleans. His current work area is data fusion and its applications in communications. He has authored and coauthored 10 journal and conference proceedings papers.