Tracking the Mode of Operation of Multi-Function Radars

TRACKING THE MODE OF OPERATION OF MULTI-FUNCTION RADARS

By

I.ARASARATNAM, B.Sc.Eng. (Hons.)
University of Moratuwa, Sri Lanka, 2003

A Thesis
Submitted to the School of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree
Master of Applied Science

McMaster University
February 2006

MASTER OF APPLIED SCIENCE (2006) MCMASTER UNIVERSITY

(Electrical and Computer Engineering) Hamilton, Ontario, Canada

TITLE: **Tracking the Mode of Operation of Multi-Function Radars**

AUTHOR: I.Arasaratnam
B.Sc.Eng. (Hons.)
University of Moratuwa
Sri Lanka, 2003

SUPERVISORS: Professor Simon Haykin & Professor Thia Kirubarajan

NUMBER OF PAGES: xii, 93

Abstract

One of the important objectives of a Radar Warning Receiver (RWR) aboard a tactical aircraft is to evaluate the level of threat posed by hostile radars in an extremely complex Electronic Warfare (EW) environment in reliable, robust and timely manner. For the RWR objective to be achieved, it passively collects electromagnetic signals emitted from potentially hostile radars. One class of such radar systems is the Multi-Function Radar (MFR) which presents a serious threat from the stand point of a RWR. MFRs perform multiple functions simultaneously employing complex hierarchical signal architecture. The purpose of this paper is to uncover the evolution of the operational mode (radar function) from the view point of a target carrying the RWR when provided with noisy observations and some prior knowledge about how the observed radar functions. The RWR estimates the radar's threat which is directly dependant on its current mode of operation. This paper presents a grid filter approach to estimate operational mode probabilities accurately with the aid of pre-trained Observable Operator Models (OOMs) and Hidden Markov Models (HMMs). Subsequently, the current mode of operation of a radar is estimated in the *maximum a posteriori* (MAP) sense. Practicality of this novel approach is tested for an EW scenario in this paper by means of a hypothetical MFR example. Finally, we conclude that the OOM-based grid filter tracks the mode of operation of a MFR more accurately than the corresponding HMM-based grid filter.

To my family

and the people from whom I learned the most.

Acknowledgments

I'm greatly indebted to my supervisors, Professor Haykin and Professor Kirubarajan for providing me the support and guidance to pursue this research work. Their insightful discussions and critical comments contributed enormously towards my understanding of this project.

I'm also thankful to Dr. Fred Dilkes, from Defence Research & Development Canada (DRDC) for his involvement and assistance in this work. His thorough revision of the thesis helped me improve the presentation of this work.

I'd like to thank Cosmin Caroiu and Terry Greenlay for their assistance in solving computer bugs. Also, I thank Lola Brooks, Cheryl Gies and Halen Jachana for their constant help and support.

I'm also grateful to the fellow graduate students in Estimation, Tracking and Fusion Lab (ETF lab.) and Communication Research Lab (CRL) for their help whenever needed.

Last but by no means Least, I'm deeply thankful to my family for their endless support, love, patience and understanding. Especially, my brother Lathan Arasaratnam helped me a lot in proof reading.

The financial assistance provided by Natural Science & Engineering Research Council (NSERC) of Canada and DRDC (Ottawa) is also deeply appreciated.

List of Contributions

1. I. Arasaratnam, "Tracking the Mode of Operation of Multi-Function Radars - OOM approach," Tech. Report, DRDC (Ottawa), Aug. 2005.

2. I. Arasaratnam, S. Haykin, T. Kirubarajan, F. Dilkes, "Tracking the Mode of Operation of Multi-Function Radars," Accepted for presentation in the *Proc. IEEE Inter. Radar Conf. IEEE/IRC'2006*, NY, USA, April 2006.

3. I. Arasaratnam, S. Haykin, T. Kirubarajan, F. Dilkes, "Observable Operator Model-Based Grid filter for Estimating the Mode of Multi-Function Radars," To be submitted for a publication in *IEEE Trans. on Aero. and Elect. Sys.*

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Electronic Warfare

Electronic Warfare (EW) [16] is a military action involving the enemy's electromagnetic emissions in all parts of the electro magnetic spectrum in order to provide intelligence on enemy's order of battle intention, capabilities and to use counter measures to deny the effective use of communication and weapon system while protecting one's own effective use of same spectrum. The EW is one of the dynamically changing fields due to continually changing threats in the battlefield. The proliferation of sophisticated radars in military applications is one reason for such threats. In order to operate against the hostile radars' action, aircraft and ships need to carry Radar Warning Receivers (RWRs) coupled with a defensive capability in the form of a self-protection jammer and decoys.

EW can be organized into four major categories:

1. Signals Intelligence (SIGINT)

2. Electronic Warfare Support Measures (ESMs)

3. Electronic Counter Measures (ECMs)

4. Electronic Counter Counter Measures (ECCMs)

Signals intelligence (SIGINT) basically refers to the data gleaned after the analysis of electro magnetic data radiated by sources such as radars weapon systems etc. Signals intelligence can be either communication data (COMmunication INTelligence or COMINT in short) or non-communication data (ELectronic INTelligence or ELINT in short). COMINT includes the intelligence obtained by interception, processing and analysis of the communication of foreign goverments or groups excluding radio and television broadcasts. ELINT contains radar characteristics such as its Pulse Repetition Frequency (PRF), pulse width, carrier frequency, modulation etc. SIGINT is collected during peace time and may be used during war time.

All the intelligence data are kept in an ELINT library in various formats. With the help of the ELINT library, the ESM system performs tasks such as interception, identification, analysis threat assessment and location of the unknown sources of radiation. One example of such an ESM system is the Radar Warning Receiver (RWR) which passively collects potentially hostile radar signals and analyzes their relative threat with the aid of the ELINT library. The ECM data processing presents one of the most complex and time-domain critical problems for current technology. The output of the ESM system in the form warnings and commands is used to initiate the ECM. The ECM system takes actions to prevent or reduce the enemy's effective use of electro magnetic spectrum (e.g., Jamming using spot noise or barrage noise). On the radar side, the radar takes actions collectively known as Electronic Counter Counter Measures (ECCMs) to dilute the effect of ECM.

1.2 Multi-Function Radars

Multi-Function Radar (MFR), as its name suggests, is capable of performing multiple tasks related to several targets simultaneously. These tasks extend from generalized ones such as search, acquisition, tracking and target-illumination to counter attacks such as initiation of ECCM, missile launching etc. The advances in software-controlled military equipment, cognitive radio, MIMO antenna technology and solid state electronics have made MFR even more sophisticated. To carry out multiple tasks, the MFR uses a highly sophisticated pulse-to-pulse (PPI) scheduling algorithm resulting in complex signal architecture. Another aspect of a MFR is its agility as it has the ability to perform entire functions from a single aperture.

Based on the current function a MFR performs, it controls its beam position and shape using an electronically controlled phased array antenna. The antenna system allows it to point the radiation beam accurately and steer it rapidly. The antenna system may typically contain 4-6 array faces in order to provide hemispherical coverage. The array face consists of a number of antenna elements distributed in a planner surface. Each element at least consists of a phase shifter and a radiating element. The resultant radiation beam is formed by superposition of electromagnetic waves with different phases. The antenna elements are grouped into so called subarray sets. The subarray steering is generally used to minimize the number of ports in mono-pulse network. It also allows large arrays to support wide-bandwidth wave forms [16]. The current EW system therefore needs to be developed in a way that it could react faster with greater accuracy to an extremely complex electromagnetic environment.

1.3 ELINT Library

As mentioned Section 1.1, ELINT refers to information or knowledge gleaned from analysis of the electromagnetic signals transmitted by a radar system or any other non-communication transmitter. ELINT plays a key role in maintaining defensive capabilities and preventing surprises and thus has a great value in EW. The information gathered by the interceptors are compiled by the intelligence service and the radar characteristics, capabilities and applications are preserved in the form of recordings and photographs in a place called ELINT library [23].

The data available in the ELINT library may have inherent uncertainties. The radar signal received at a single ELINT station at any one time provides a glimpse of radar's characteristics. It may therefore require thousands of isolated encounters to provide a reasonable portrait of the radar. In addition to the collection strategies, the optimal design of an ELINT database is also important in EW. The process of designing an optimal structure for the ELINT database is a highly sophisticated task. In the literature, there are several modelling techniques available (e.g., Syntactic modelling [21], Relational Database modelling [13], etc.). Each approach has its own merits. On the receipt of a new information, the ELINT library is also updated by the ESM system. In short, the success on the modern battlefield is heavily dependant on an ELINT library.

1.4 Data Processing in ESM System

As shown in Figure 1.1, three major subsystems have been identified in an automated ESM system. Since the ESM sytem is automated, rapidity in response to radar emitter threats, is enhanced. One of the challenging signal processing jobs in the subsystem 1 of Figure 1.1 is the pulse train deinterleaving. Due to signal density and parametric overlap, the pulse train of more than one emitters transmitted over the same channel needs to be separated for source identification and operational mode/threat estimation and this operation is known as pulse train *deinterleaving*. Figure 1.2 illustrates how a pulse deinterleaver deinterleaves the pooled pulse trains belonging to different radars. A Modern deinterleaver exploits the complete ELINT data (e.g., signal characteristics such as pulse width, radio frequency, etc.) in pulse deinterleaving which yields more efficient results than exploiting the Pulse Time of Arrival (ToA) solely.

In subsystem 2 of Figure 1.1, features of interleaved pulses such as pulse amplitude, carrier frequency, pulse width, Pulse Repetition Interval (PRI), angle of arrival etc. are extracted and compared against the parameters of radars recorded in the ELINT library. The gap between the reality and the available ELINT is one of the major challenges that the EW community has to overcome at this stage.

Subsystem 3 is dedicated specially for radars which exhibit a hierarchical signal architecture. The output of subsystem 3 is finally displayed to the operator in various formats and (s)he takes the final decision over the next tactic.

1.5 Problem Statement

The EW problem may, in general, be viewed in two ways:

- From a radar's view point, the main focus is to detect the target, track and identify its critical parameters. For the radar's objective to be achieved, it needs to perform satisfactorily even in the presence of ECM while exhibiting a low probability of intercept. A low probability of intercept is achieved by employing the techniques such as random frequency hopping, chirped and direct sequence spread spectrum transmission, dynamic transmit power control, infrequent scanning and antennas with suppressed side-lobes.

- From the point of view of a Radar Warning Receiver (RWR) aboard a tactical aircraft, it needs to generate the warnings to the aircraft due to threats posed by potentially hostile radars in a reliable, robust and timely manner. The warnings generated by a RWR facilitate the aircraft or ship to control its behavior based on what is going on within the hostile Radar. The design of the RWR is therefore heavily influenced by the design of the radar systems. For the objective of a RWR to be achieved, it has to perform the following two tasks:

  1. Classification of observed radars
  2. Estimation of the mode of operation of the observed radar.

In this investigation, we assume that we have classified the radars responsible for the emission successfully (e.g., based on the radar signal characteristics). We therefore focus our attention on the operational mode estimation problem from the point of view of a RWR aborad a tactical airborne platform.

1.6 Motivation

The level of threat posed by a hostile radar directly depends on its current mode of operation. For example, the level of threat posed by the radar becomes higher and higher as the radar mode evolves from search to target acquisition and then to track mode which may be the precursor to the missile engagement. Estimation of the current operational mode therefore gives clues about the level of instantaneous threat posed by potentially hostile radar so that the aircraft under scrutiny can prepare for the next tactic in advance. For example, a target can deploy counter measures or perform evasive maneuvers if it becomes engaged.

In this investigation, a grid filter which exploits likelihood estimates of the pre-trained Observable Operator Models (OOMs), is applied to track the current mode of operation of a MFR, hopefully, accurately.

1.7 Summary

This chapter presents a brief account of various elements of electronic warfare that are basic building blocks of an automated ESM system. One example of such ESM system is the Radar Warning Receiver (RWR) which tracks the operational mode of hostile radars and generates warnings. An aircraft or ship carrying the RWR in turn takes counter-actions against the hostile radar. Also, this chapter describes the research question of how a RWR can track the mode of operation of MFRs and the motivation behind this research.

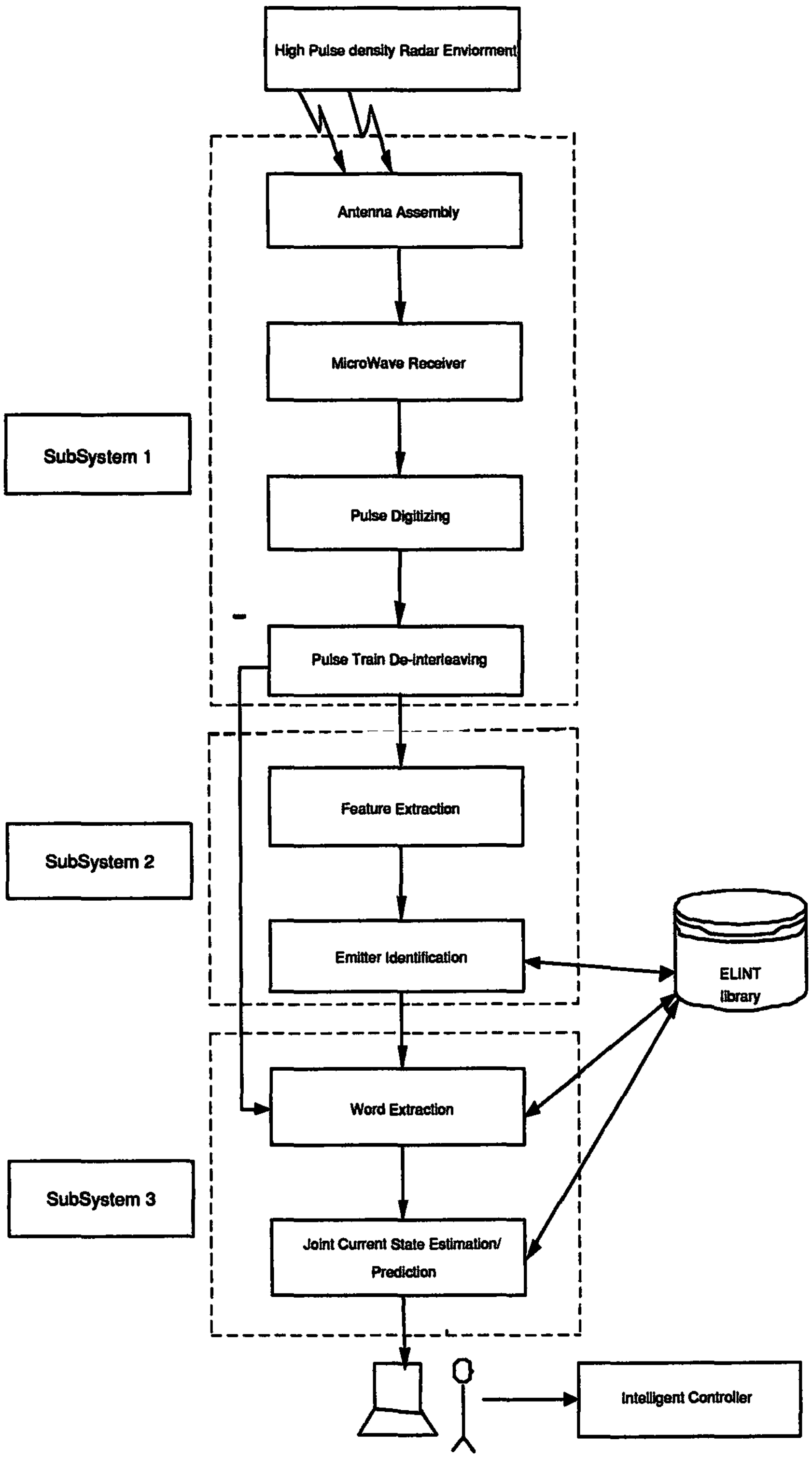

Figure 1.1: Data Processing in ESM System

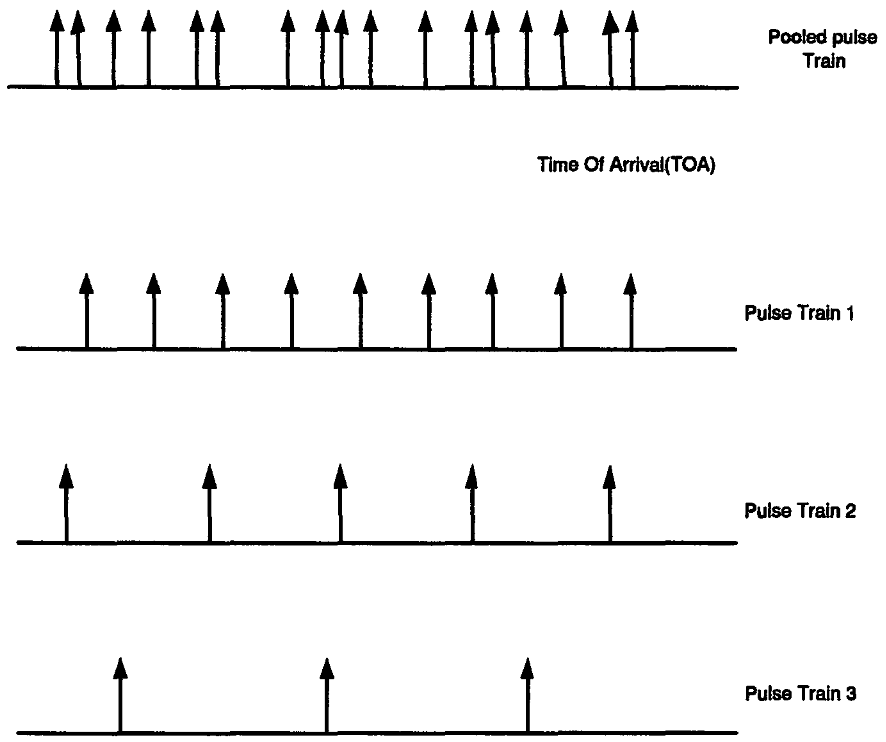

Figure 1.2: Illustration of the output of the deinterleaver after receiving a pooled pulse train in a multi-radar environment

Chapter 2

Signal Architecture of MFRs

MFRs exhibit a highly sophisticated pulse structure. Therefore the signals of modern radars, especially MFRs can no longer be viewed as a stable sequence of pulses. Pulse Repetition Intervals (PRIs) are typically scheduled by intricate radar control software. In addition, multiple independent operations performed by MFRs are often multiplexed in time domain. Since signal processing at pulse level seems complicated and tedious, we exploit the signal arrangement by creating a layered signal architecture and the mode estimation of MFRs is performed at a higher level of the signal hierarchy. Throughout the thesis, we will consider the sanitized version of the signal structure of the MFR called Mercury[†].

2.1 Layered Signal Architecture

Since the focus of this investigation is on word level processing which follows the pulse level processing, the term *word* is fundamental to much of what follows in

[0†]The Mercury data structure is a sanitized version of an actual MFR system, which is described in detail in the PhD thesis of N. Visnevski [21].

the report. Basically a single *word* is made up of pulse sequences which are arranged into groups according to specific pattern. Figure 2.1 shows an example of a syntactic hierarchical signal architecture. As illustrated in Figure 2.1, a single word is made up of 5 sections (A-E). Sections A, C and E are *dead time zones*. No pulse is emitted in the duration of the so-called dead time. Section B is known as *Pulse Doppler Sequence*. Pulse Doppler Sequence consists of a certain number of pulses with a constant pulse-to-pulse interval (PPI). Each word has its own number of pulses and a characteristic PPI. Section D, known as *Synchronization burst sequence* has a fixed number of pulses for all words in the alphabet. The radar has an alphabet of 9 words (W1-W9), all of which have similar pulse envelope.

A group of words make up a *phrase*. In fact the phrase structure is specific to the radar of interest. For the Mercury type MFRs, a phrase is composed of 4 words. Though each phrase is in general, associated with single task such as search, locate, track etc., it is not always the case. The mapping between a task and some phrases is many-to-many. For example, as shown in Table 2.1, [W1 W2 W4 W5] is encountered only in search mode where as [W6 W6 W6 W6] appears in acquisition, non-adaptive track and track maintenance. This relationship makes the mode estimation job even more complicated.

A *clause* is composed of a certain number of phrases. The number of phrases in a clause, determines the number of tasks a radar can engage in simultaneously. As shown in Figure 2.1, the radar can simultaneously engage in 5 different tasks. A clause is the product of time-division multiplexing in MFR operation.

In this investigation, the mode estimation is performed at word level. Signal processing at word level has the following advantages over pulse level:

1. Since the a priori knowledge about a radar signal structure is incorporated in the word extraction process, word level processing is less prone to errors than the signal processing at pulse level. This can also be associated with the well known fact in machine learning that it is always preferable to encode an input sequence with less number of bits for learning a hidden process.

2. Word level processing is less complex than pulse level.

3. It is also possible to analyze the task performed by the MFR for each target under its scrutiny independently.

The third beneficial factor can be further explained with the aid of Figure 2.2. In reality, word sequence is arranged so that the clauses follow each other sequentially. For instance, right after the last word symbol of the last phrase in the first clause, the first symbol of the first phrase of the second clause follows. For the simplicity of analysis, the arrangement of the clauses can be viewed in a way that they are stacked one after the other in a 2-D table format. Now the task performed on each target can be separated out by considering only the vertical alignment of phrases. In Figure 2.2, separation is shown by dotted lines.

2.2 Mode Evolution of a Typical MFR

Figure 2.3 illustrates how a typical radar jumps from one mode to another in an ordered manner. Mode-transition follows the first-order Markov chain which tells that the state at time step n is dependant only on the state at time step $n-1$.

2.3 Summary

This chapter presents the hierarchical structure of the MFR signals. This structure includes three basic blocks namely word, phrase and clause. We take the advantage of this layered structure in estimating the operational mode of a MFR. Also, this chapter gives a brief account of how the modes of a typical MFR evolves from one to another.

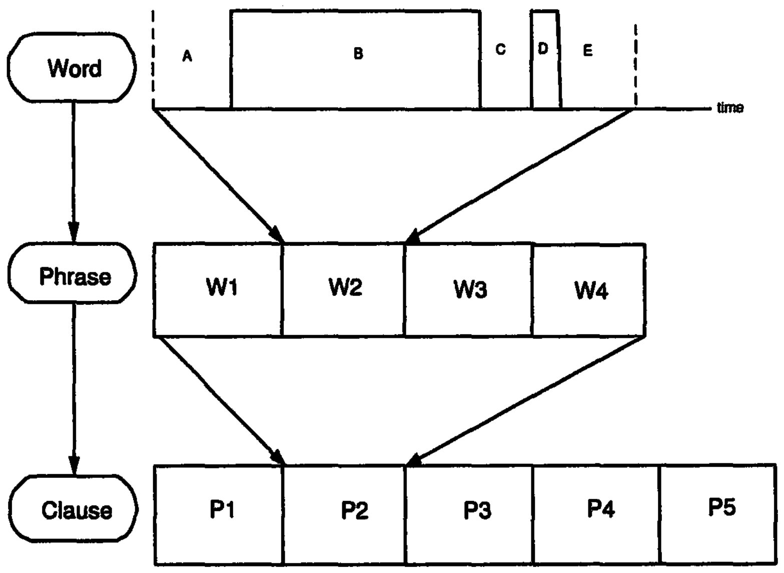

Figure 2.1: Hierarchical Radar Signal Architecture

Functional State	Phrase Content	Functional State	Phrase Content
Four-Word search	$[W_1 W_2 W_4 W_5]$	TM	$[W_1 W_7 W_7 W_7]$
	$[W_2 W_4 W_5 W_1]$	(Track-	$[W_2 W_7 W_7 W_7]$
	$[W_4 W_5 W_1 W_2]$	Maintenance)	$[W_3 W_7 W_7 W_7]$
			$[W_5 W_7 W_7 W_7]$
Three-word Search	$[W_1 W_3 W_5 W_1]$		$[W_6 W_7 W_7 W_7]$
	$[W_3 W_5 W_1 W_3]$		$[W_1 W_8 W_8 W_8]$
	$[W_5 W_1 W_3 W_5]$		$[W_2 W_8 W_8 W_8]$
			$[W_3 W_8 W_8 W_8]$
Acquisition	$[W_1 W_1 W_1 W_1]$		$[W_4 W_8 W_8 W_8]$
	$[W_2 W_2 W_2 W_2]$		$[W_5 W_8 W_8 W_8]$
			$[W_4 W_9 W_9 W_9]$
NAT	$[W_1 W_6 W_6 W_6]$		$[W_5 W_9 W_9 W_9]$
(Non-Adaptive	$[W_2 W_6 W_6 W_6]$		$[W_6 W_9 W_9 W_9]$
Track)/TM	$[W_3 W_6 W_6 W_6]$		$[W_7 W_9 W_9 W_9]$
			$[W_9 W_9 W_9 W_9]$
Range Resolution	$[W_7 W_6 W_6 W_6]$		
	$[W_8 W_6 W_6 W_6]$	Acq.,NAT or TM	$[W_6 W_6 W_6 W_6]$
	$[W_9 W_6 W_6 W_6]$		

Table 2.1: List of a typical MFR phrase combinations according to the functional state of the radar which has an alphabet of size 9 (W1-W9)

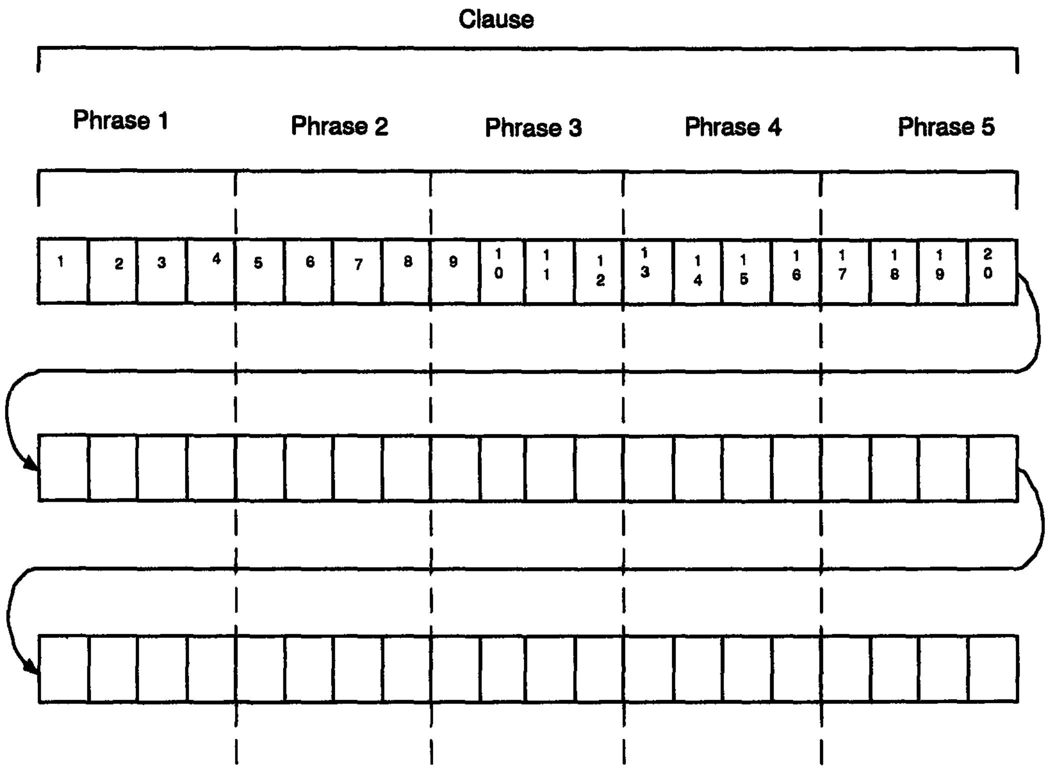

Figure 2.2: Output sequence for the MFR of signal structure of Figure 2.1

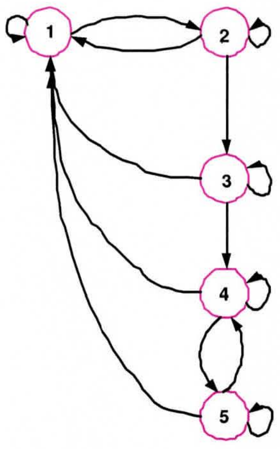

Figure 2.3: Operational Mode Transition of a typical MFR: 1-Search (Sea), 2-Acquisition (Acq), 3-Non Adaptive Track (NAT), 4-Range Resolution (RR), 5-Track Maintenance (TM).

Chapter 3

Syntactic Model-Based Mode Estimation

3.1 Introduction

In this approach, tracking the mode of operation of the MFR is rooted in the theory of formal stochastic language and discrete event system theory. In the development of this approach, the MFR is viewed as an abstract discrete event system that broadcasts messages using an alphabet of fixed size [21]. This is indeed the very approach taken in the theory of formal languages and computational linguistics. A widely known and powerful model for formal language is *grammar* (or syntax) [4]. Grammar is the set of rules of a language and represents the infinite number of strings in a more efficient way. The radar language can therefore be modelled by the stochastic grammar and processed based on the theory of syntax analysis. This approach is known as *syntactic modelling*. Syntactic modelling has widely been used in:

1. Pattern recognition applications
2. Natural language and speech processing
3. Compilers for the computer languages
4. Bio informatics and genomic sequencing

3.2 Grammar

There are two broad classes of grammar: Deterministic and Stochastic Grammar. Since grammar is fundamental to explain the syntactic modelling, let us see its definition.

Definition A deterministic grammar G is a quadruple

$$G = (\Sigma, V, P, S_0) \tag{3.1}$$

where:

Σ is the alphabet(the set of terminal symbols of the grammar)
V is the set of non-terminal symbols (variables) of the grammer
P is the finite set of production rules
S_0 is the starting non-terminal symbol.
In general, P is defined as

$$P : \Sigma^* \times V^* \longrightarrow \Sigma^* \times V^* \tag{3.2}$$

where Σ^* operation is called Kleene (reflexive or transitive) closure which is used to construct the infinite number of strings by concatenating them together.

For example, if $\Sigma = \{a,b,c\}$ then $\Sigma^* = \{\varepsilon,a,b,c,aa,ab,ac,\ldots\}$, where ε refers to the empty string.

By imposing certain restrictions on the definition of the grammar, Chomsky has defined four types of grammars: Regular Grammar (RG), Context-Free Grammar (CFG), Context-Sensitive Grammar (CSG) and Unrestricted Grammar (UG). The syntactic model for the MFR is based on CFG [21]. CFG has the production rule of the form of $S \rightarrow \beta$, where the left hand side of the production rule must contain only the non-terminals whereas the the right hand side can be any strings. Any interested reader may refer [4] for detailed description of grammar.

A number of practical applications contain uncertainties that are often represented by probability distributions. For instance, radar signals are typically observed in the noisy environment where the signal inference may cause observation sparseness. These factors require the extension of the concept described above into the domain of stochastic grammar which is defined as follows:

Definition A stochastic grammar G_s is a five-tuple

$$G_s = (\Sigma, V, P, P_s, S_0) \tag{3.3}$$

where P_s is the probability distributions over the set of production rules P and the rest of the notations have the meaning as defined earlier.

3.3 Finite State Automata (FSA)

Definition A Finite State Automata(FSA) is a five-tuple

$$\Lambda = (Q, \Sigma, \delta, S_0, F) \tag{3.4}$$

where:

Q is the set of states of the FSA

Σ is the set of input symbols of the FSA

δ is the transition function of the FSA

S_0 is the initial state of FSA

F is the set of final (accepting) states of the FSA ($F \subset Q$)

Note 1: FSA can be shown to be equivalent to regular languages, regular grammars and regular expressions.
Note 2: Chomsky has proved that the CFG needs to satisfy the non-self embedding property in order to generate finite state language [2].

Having defined the CFG for the radar language, the finite state automata (FSA) based on this grammar is subsequently developed. In fact, FSA is equivalent to Hidden Markov Model (HMM) with some structural constraints [18, 20]. In essence, the relationship between the formal stochastic language and the HMM therefore provides a technique for MFR signal processing.

Σ-Alphabet/Set of terminal symbols	$\Sigma = \{a, b\}$
V-Variables/Set of non-terminal symbols	$V = \{S_0, S_1\}$
P-Production rules	$S_0 \rightarrow aS_1$ $S_1 \rightarrow bS_0/a$
S_0-Starting non terminal symbol	S_0

Table 3.2: Example for a Grammar

3.4 Toy Example

The following example clearly illustrates the essence of syntactic modelling. The *Syntactic model* is a quadruple grammar defined as in eq (3.1). Consider the example for a grammar as shown in Table 3.2. Based on this grammar, the following sequence can be generated:

$S_0 \Rightarrow aS_1 \Rightarrow aa$

$S_0 \Rightarrow aS_1 \Rightarrow abS_0 \Rightarrow abaS_1 \Rightarrow abaa$

$S_0 \Rightarrow aS_1 \Rightarrow abS_0 \Rightarrow abaS_1 \Rightarrow ababS_0 \Rightarrow ababaS_1 \Rightarrow ababaa$

$S_0 \Rightarrow \ldots\ldots\ldots \Rightarrow abab\ldots aa$

This example shows how a simple grammar can generate strings with each string viewed as a sequence of observations emitted from some radar. It is noteworthy that the syntactic model is stochastic by virtue of the fact that grammatical production rules may themselves be probabilistic. Let us define the probability of the production rule as shown below:

$S_0 \rightarrow aS_1$ with probability, 1

$$S_1 \rightarrow bS_0 \text{ with probability, } (p)$$

$$S_1 \rightarrow a \text{ with probability, } (1-p)$$

Now we are in a position to define the HMM, λ for this grammar as follows:

$$\lambda = \{M, O, \omega_0\} \tag{3.5}$$

where

M - Markov matrix or state transition matrix

O - Observation probability matrix or Emission probability matrix

ω_0 - Initial state probability vector.

For the above mentioned example,

$$M = \begin{pmatrix} 0 & 1 \\ p & 1-p \end{pmatrix}$$

$$O = \begin{pmatrix} 1 & 0 \\ 1-p & p \end{pmatrix}$$

$$\omega_0^T = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

In this context, Expectation Maximization (EM) algorithm may be subsequently used to tune stochastic parameters of the HMM. Availability of the HMM, in turn facilitates the final step known as *HMM filtering* for mode estimation. HMM filtering algorithm strives to produce the most accurate instantaneous state estimate by minimizing the symbol error probability.

The model although provides a procedure for compact representation of ELINT data while facilitating to estimate the state of the radar, in our opinion, following are the limitations associated with the syntactic approach.

- The modelling of MFR is based on the context free grammar, which needs to satisfy the non-self embedding property in order to generate Finite State Automata (FSA). Therefore, there is no guarantee that all the radar grammars would pass this test.
- If a realistic radar is considered, the number of states in the word level HMM is quite large. Consequently, it becomes more expensive in terms of computational time and memory resource.
- Many of the intermediate states in HMM seem dummy. They do not actually refer to operational modes of a radar. As a result, clustering of these dummy HMM states need to be performed to estimate the state of the radar at any given time.
- HMM filtering algorithm is valid for a stationary process. However, an input for the HMM filter is collected from the radar environment which is non-stationary.

In brief, all these issues related to syntactic model-based radar mode estimation lead to the necessity for a more sensible design approach which is conceptually transparent simple albeit reliable and robust. The next chapter describes a novel approach that solves the above-mentioned issues in an efficient manner.

3.5 Summary

MFRs can be considered as a stochastic discrete event system that are communicating information using some stochastic formal languages. These languages can be modelled by grammars that can be derived based on available electronic intelligence about the radar of interest. This MFR modelling approach is refereed to as syntactic modelling. The applicability of this approach is explained using a toy example. Finally, this chapter presents some limitations associated with this approach.

Chapter 4

Introduction to OOMs

The Observable Operator Model (OOM) is a mathematical tool for modelling a stationary time series or symbolic process [10]. For example, an OOM can learn the probability distribution of an unknown symbol generator from its training data. As a method of choice for describing an unknown distribution of stochastic process, the OOM is comparable to higher order Markov chain, Hidden Markov Model (HMM), stochastic grammar or even stochastic Turing machine. A growing interest in OOM has recently been witnessed in the field of optimal decision-making and action-selection for autonomous agents. In the context of this work, OOM plays a key role in capturing the true data generation mechanism of a MFR.

In this chapter, we present the mathematical formulation of OOM. This chapter is in fact, a summary of results of the original work carried out by Herbert Jaeger [7–10], the inventor of OOM.

Definition A m-dimensional OOM is a triple, $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$, where $\omega_0 \in \Re^m$ and $\tau_a : \Re^m \rightarrow \Re^m$ are linear maps represented by matrices, satisfying

1. $\mathbf{1}\omega_0 = 1$.
2. $\mu = \sum_{a\in\Sigma}\tau_a$ has column sum equal to 1, where Σ is the alphabet.
3. $\forall a_0 ... a_r$ it holds that $\mathbf{1}\tau_{\bar{a}}\omega_0 \geq 0$, where $\tau_{\bar{a}} \equiv \tau_{a_r a_{r-1} ... a_0} \equiv \tau_{a_r}\tau_{a_{r-1}} \cdots \tau_{a_0}$.

'm' in this definition refers to the dimension of the vector space spanned by the prediction function or predictor space. In other words, $\Re^m$ is the domain of the operators. τ_a is the operator which is indexed over the output symbol 'a' of the stochastic process and ω_0 is the initial state vector. From the first and second conditions of the above definition, we see that the state vector components and the entries of μ can take negative values while satisfying the condition that column sum is equal to 1. Such relaxations in freedom of sign have played a key role in developing a more efficient OOM learning algorithm than HMM's EM algorithm [10].

Note: We shall use $\underline{1} = (1, ...1) \in \Re^m$, a row vector consisting of all 1's throughout the thesis. Further, we shall denote the Kleene closure operation over Σ as Σ^*, which includes an infinite set of all finite strings formed by concatenation of symbols from Σ and the empty string.

Proposition 4.0.1 *Let $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$ be an OOM. Let $\Omega = \Sigma^*$ and Ψ be the σ-Algebra generated by all finite-length initial events on Ω. Then a numerical function*

$$P_o(\bar{a}) \quad = \quad \mathit{1}\tau_{\bar{a}}\omega_0 \qquad (4.6)$$

can be uniquely extended to a probability measure P on (Ω, Ψ) defining a discrete time, finite valued stochastic process $(\Omega, \Psi, P, (Y_n)_n \in \mathbb{N})$.

Why *Operators* are named as *Observable Operators*?

An OOM describes the change of our knowledge about the future of a stochastic system by means of linear operators chosen from a finite set of operators on the basis of current observations. In other words, the stochastic trajectories in OOM are conceived as sequence of operators. As shown in Figure 4.1, an observed piece of trajectory $\ldots, a_1 a_2 a_3, \ldots$ would correspond to a concatenation of operators $\ldots, \tau_{a_1}(\tau_{a_2}(\tau_{a_3})), \ldots$. The fact that the one-to-one relationship between the selection of operator and the observation symbol has led to the naming of operators as observable operators.

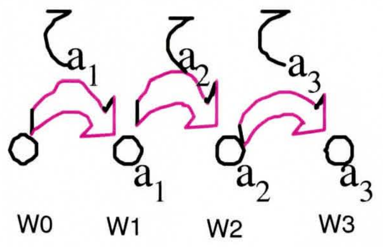

Figure 4.1: Stochastic Trajectory as a Series of Operators

4.1 HMMs and OOMs

HMM basically specifies a distribution of a discrete-time, discrete-valued stochastic observation process $(Y_n)_{n \in N}$, where the random variables Y_n have outcomes in an alphabet $\Sigma = \{a^1 \ldots, a^\alpha\}$. Assume a state process $(X_m)_{m \in \mathbb{N}}$ having the Markov chain with finite number of hidden states $\{s_1, \ldots, s_m\}$. The state transition probability from s_j at time $(n-1)$ to s_i at time n is denoted by the $(i, j)^{th}$ element of the

Markov matrix M of size $m \times m$. To characterize the HMM completely, we need to specify three components: initial distribution ω_0, Markov matrix M and the observation matrix $O_a, \forall a \in \Sigma$. For every $a \in \Sigma$, we collect the emission probabilities $P(Y_n = a_i / X_m = s_j)$ in a diagonal *observation matrix* O_a of size $m \times m$. This section presents how the HMM describing a stochastic process can be associated with an equivalent OOM.

The process described by the HMM is stationary if ω_0 is an invariant distribution of the Markov chain, as shown by

$$M^T \omega_o = \omega_o \tag{4.7}$$

Let the operator indexed over the symbol a taken from the alphabet Σ, be $\tau_a = M^T O_a$. Then the probability to observe the sequence $a_o ... a_r$ can be obtained by

$$P(a_o ... a_r) = 1 \tau_{a_r} ... \tau_{a_o} \omega_o \tag{4.8}$$

M can be recovered from the operators τ_a by observing that

$$\begin{aligned} \sum_{a \in \Sigma} \tau_a &= M^T \sum_{a \in \Sigma} O_a \\ &= M^T \mathbf{id} \\ &= M^T \end{aligned}$$

We can therefore rewrite the HMM with a structure $(\mathbb{R}^m, (\tau_a)_{a \in \Sigma}, \omega_0)$, where $\mathbb{R}^m$ is the domain of the operators.

For example, consider a HMM with two hidden states s_1 and s_2 and the two outcomes $\Sigma = \{a, b\}$. The parameters needed to describe the HMM completely is shown in Figure 4.2. As just outlined, the HMM with a modified structure i.e., (M, O_a, O_b, ω_0) becomes

$$M = \begin{pmatrix} 1/4 & 3/4 \\ 1 & 0 \end{pmatrix}, O_a = \begin{pmatrix} 1/2 & 0 \\ 0 & 1/5 \end{pmatrix}, O_b = \begin{pmatrix} 1/2 & 0 \\ 0 & 4/5 \end{pmatrix}, \text{and}$$

$$\omega_0 = \begin{pmatrix} 2/3 \\ 1/3 \end{pmatrix}, \text{respectively.}$$

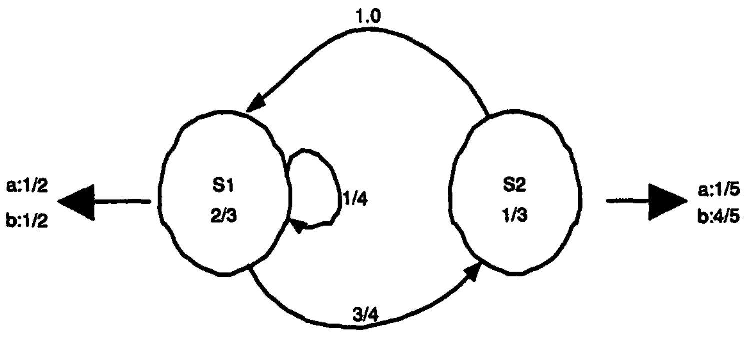

Figure 4.2: Hidden Markov Model

Since $M^T\omega_0 \neq \omega_0$, this example describes a non-stationary process. Similar to OOM structure, the HMM can now be rewritten as $(\Re^m, \tau_a, \tau_b, \omega_0)$, where

$$\begin{aligned} \tau_a &= M^T O_a \\ &= \begin{pmatrix} 1/8 & 1/5 \\ 3/8 & 0 \end{pmatrix} \end{aligned}$$

Similarly,

$$\tau_b = \begin{pmatrix} 1/8 & 4/5 \\ 3/8 & 0 \end{pmatrix}$$

At this point, one can perceive how the OOM components such as m, ω_0 and matrix μ^T in a kind of abstract definition of an OOM are associated with number of states, the state probability vector and Markov matrix or state transition matrix M of a corresponding hidden Markov model respectively. However, one can interpret an OOM by doing the following:

1. Relax the requirement that the transpose of the Markov matrix M^T be the stochastic matrix, to a weaker condition that the column of M^T each sums to 1.

2. Relax the requirement such that ω_0 merely needs to satisfy the component sum equal to 1. In other words, ω_0 is allowed to assume negative values.

The mathematical construction of an OOM can therefore be considered as a generalization of HMM. This is one of the virtues of OOM. Since the learning in OOM is rooted in the efficiency sharpening (ES) principle, OOM learning algorithm is known as the ES algorithm. The ES learning algorithm will be discussed in Chapter 4 in detail. The following are some other virtues of modelling and learning with OOMs:

- **Speed of Convergence**- for a given stochastic process, OOM/ES learning yields a model estimate in a fraction of the computational time than that of HMM/EM algorithm [10]. Typically, OOM/ES learning algorithm requires not more than five ES iterations to converge to a reasonably good model, whereas HMM/EM algorithm converges to the target model in more than one hundred iterations in general. Also, it should be noted that the computational load of one ES iteration is comparable to one EM iteration.

- **Accuracy**- the model obtained via the ES learning algorithm is markedly more accurate than the corresponding model obtained via HMM/EM algorithm. This has been proved after testing over a number of standard data sets [10]. We also carried out an experiment to verify this in Chapter 5. With large data sets, HMM/EM algorithm often gets trapped in one of the suboptimal maxima of the likelihood function. In order to get a reasonably good model, a good guess of initialization is crucial in HMM/EM algorithm.

- **Expressness**- for the same level of modelling accuracy, OOM assumes less dimension than HMM. Another aspect of the enhanced expressiveness of OOM is that the class of processes that have finite dimensional OOM properly includes the process characterized by finite dimensional HMMs. There are certain linear dependant processes that can be captured by OOMs whereas HMMs cannot be employed (e.g., the probability clock [10] where the outcome probability fluctuates in time).

- **Tractability**- since OOM is expressed in terms of linear algebra which is one of the the well established fields in applied mathematics, OOM can be interpreted transparently.

There are two limitations associated with an OOM. One is the negativity issue associated with some (rare) model predicted probabilities. Another one is the instability problem associated with larger model dimension. Fortunately, there exist heuristic counter measures for both of these issues.

4.2 OOM as a Generative Model

When an OOM is said to be a generative model, it means that the OOM includes all the formalized and compressed description of the probability distribution of all possible realizations. Consider an OOM, $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$ where, $\Sigma = \{a^1 ..., a^\alpha\}$. Now we will see how the task of producing observations $a_1, a_2 ...$ is performed at times $n = 1, 2, ...$ such that

1. at time $n = 1$ the probability of producing a is equal to $P(Y_1 = a)$
2. and at every time step $n > 1$, the probability of producing a (after $a_1, a_2, \dots, a_n$ have already been produced) is equal to $P(Y_{n+1} = a|Y_1 = a_1, ..., Y_n = a_n)$

The entire generation procedure can be summarized as follows:

1. State vector initialization: put $\omega = \omega_o$
2. Assume that at time n, the state vector ω_n has been computed. Now determine the probability vector $\mathbf{p}$ of the $n+1^{th}$ symbol as

$$\mathbf{p} = S\omega_n \tag{4.9}$$

where, $S = \begin{pmatrix} 1\tau_{a^1} \\ \vdots \\ 1\tau_{a^\alpha} \end{pmatrix}$ and choose a_n according to the **p**-vector

3. Update the state vector by $\omega_{n+1} = \tau_{a_{n+1}}\omega_n / 1\tau_{a_{n+1}}\omega_n$.

At each instant of time n, the OOM passes through a certain stochastic trajectory. From the third step, the trajectory can be a expressed as a vector called state vector ω_{n+1} which is obtained by ω_n and the new observation obtained at time $(n+1)$. Figure 4.3 illustrates how the symbols are generated from an OOM of dimension 2 with an alphabet $\Sigma = \{a, b\}$ for one time step. At time $n = 0$, let the state vector be ω_0. At time $n = 1$, the symbol a or b can be chosen according to p-vector. Assume that the symbol a is chosen. Therefore, the state vector ω_1 at time $n = 1$ is obtained by applying the operator τ_a and renormalizing to component sum 1.

4.3 Reverse OOMs

For an OOM $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$ with an induced probability distribution P_A, its *reverse* OOM A^r is characterized by a probability distribution P_{A^r} satisfying,

$$\forall a_1a_2\ldots a_n \in \Sigma^* : \; P_A(a_1a_2\ldots a_n) \; = \; P_{A^r}(a_na_{n-1}\ldots a_1) \qquad (4.10)$$

A reverse OOM can be computed from the forward OOM by observing the following proposition.

Proposition 4.3.1 *If $A = (\Re^m, (\tau_a)_{a\in O}, \omega_0)$ is an OOM for a stationary process, ω_0 has no zero entry, then $A^r = (\Re^m, (D\tau_a^T D^{-1})_{a\in O}, \omega_0)$ is a reverse OOM to A,*

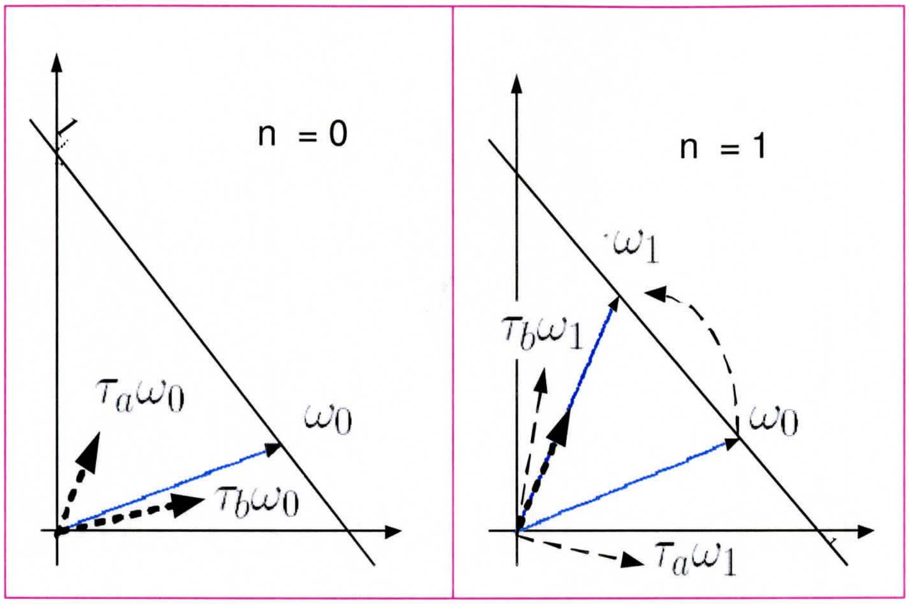

Figure 4.3: OOM as a Sequence Generator

where $D = diag(\omega_0)$ is a diagonal matrix with ω_0 on its diagonal.

For proof, see [8].

From the above proposition, it is observed that the process dimension is the same for forward and reverse cases. Also, we will see how an important type of characterizer is obtained from the reverse OOM when we discuss the OOM/ES learning Algorithm.

4.4 Equivalent OOMs

Two OOMs are defined to be equivalent precisely if and only if they induce the same probability distribution for any finite sequences. The following proposition provides a necessary and sufficient condition for two OOMs to be equivalent.

Proposition 4.4.1 *Given two OOMs $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$ and $A' = (\Re^m, (\tau'_a)_{a\in\Sigma}, \omega'_0)$, A and A' are equivalent by describing the same stochastic process if and only if there exists a bijective linear map $\rho : \Re^m \to \Re^m$ satisfying the following conditions:*

- $\rho(\omega_0) = \omega'_0$
- $\tau'_a = \rho\tau_a\rho^{-1}$ *for all* $a \in \Sigma$
- $1\upsilon = 1\rho\upsilon$ *for all (column) vectors* $\upsilon \in \Re^m$

4.5 Interpretable OOMs

Within the equivalence class of OOM, there is a proper subset of minimal dimensional OOMs. The states of the such minimal dimensional OOMs can be interpreted as future distributions of trajectories. This class of OOMs are collectively known as *interpretable OOMs*. Interpretable OOM is pivotal for OOM learning algorithm. It has the following three properties:

1. Its state vectors are probability vectors.
2. The components of its state vector provide probabilities of a certain well-defined future events (*characteristic events*) as described below.
3. It can constructively be obtained through a learning algorithm which is to be discussed in Chapter 5.

4.5.1 Characteristic Events

For a discrete time stochastic process Y_n with the alphabet Σ, k-event B is defined as a nonempty subset of Σ^k. In mathematical notation, it is represented as

$$B \subset \Sigma^k \tag{4.11}$$

$P((Y_{n+1}, Y_{n+2}, \ldots, Y_{n+k}) \in B_i)$ denotes the probability of observing the process trajectory passing through B_i in the time window of $[n+1, n+k]$. For brevity, we shall use $(P(B))$ instead of $P((Y_{n+1}, Y_{n+2}, \ldots, Y_{n+k}) \in B_i)$ throughout the thesis.

Definition Let $(Y_n)_{n\geq 0}$ be a m-dimensional stationary process with observables from an alphabet Σ. Let, for sufficiently large k, $\Sigma^k = B_1 \cup \ldots \cup B_m$ be the partition of the set of strings of length k into m disjoint, non-empty sets B_i. Then this partition is called a set of characteristic events $B_i (i = 1, \ldots, m)$, if some sequences $\bar{a}_1 \ldots, \bar{a}_m$ exist such that the matrix $V = (P(B_i|\bar{a}_j))_{1\leq i,j\leq m}$ is nonsingular.

Hereby $P(B_i|\bar{a}_j)$ we mean $\sum_{\bar{b}\in B_i} P(\bar{b}|\bar{a}_j)$. We will see the importance of the invertibility of V when we discuss about the OOM learning algorithm. Now let us define interpretable OOMs.

Definition Let $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$ be a finite dimensional OOM and let $(B_i), (i = 1, \ldots, m)$ be the characteristic events of A. Then A is called *interpretable* with respect to the characteristic events $(B_i), (j = 1, \ldots, m)$ if

$$P(B_i|\omega_n) = (\omega_n)_i, \quad \forall\, n \in N, i \in \{1, \ldots, m\} \tag{4.12}$$

where $P(B_i|\omega_n)$ denotes the probability of observing $(X_{n+1}, ..., X_{n+k}) \in B_i$ given that the OOM was in state ω_n at time n. Further, $(\omega_n)_i$ denotes the i-th component of the state vector ω_n. Since $B_1 \cup \ldots \cup B_m$ is an exhaustive and disjoint partitioning of Σ^k, it follows that $\sum_i P(B_i|\omega_n) = 1$ and hence $\mathbf{1}\omega_n = 1$. In other words, ω_n is a probability vector.

Proposition 4.5.1 *In an interpretable OOM (interpretable with respect to Characteristic events $B_1 \ldots, B_m$), it holds that*

1. $\omega_o = (P(B_1), \ldots P(B_m))^T$
2. $\tau_{\bar{a}}\omega_0 = (P[\bar{a}B_1], \ldots\ldots\ldots, P[\bar{a}B_m])^T$

where $P(\bar{a}B) = \sum_{\bar{b}\in B} P(\bar{a}\bar{b})$.

4.6 Fingerprint Plots

The state dynamics of the an interpretable OOM can be graphically represented in a standard fashion. The graphical representation allows one to visually compare the dynamic process expressed by two stochastic generators. Figure 4.4 shows the fingerprint plots of states obtained from generating runs of a 3-dimensional OOM over an observation alphabet of size 3. Interpretable states, being probability vectors are non negative and thus lie in the intersection of the positive orthant of $\Re^3$ with the hyperplane $H = \{x \in \Re^3 | \mathbf{1}x = 1\}$. This intersection is the triangular surface. Its corners mark the three unit vectors of $\Re^3$. Figure 4.4 is, in fact, the projection of a 3-dimensional plot over the 2-dimensional planar surface. The states are colored with one of the three colors depending upon which of the three operators was used to produce this state. When plotting the states of interpretable OOMs with dimension $m > 3$, one can join some of the characteristic events until three merged events are left. State vectors can then be plotted on a 2-dimensional triangular canvass in a way similar to the one obtained in Figure 4.4. Plotting of state fingerprints is an indispensable means to sharpen one's intuitions about the objects one is dealing with. It is noteworthy that there exists a similar technique to graphically represent the states of HMMs.

4.7 Summary

This chapter presents the basic structure of the OOM and how the well known stochastic modelling technique called Hidden Markov Model (HMM) falls into the special case of the OOM. Different classes of the OOM are also presented in this chapter. The interpretable OOMs play a key role in the development of OOM learning algorithm. The state dynamics of a well defined interpretable OOM can be represented in a graphical form using finger print plots.

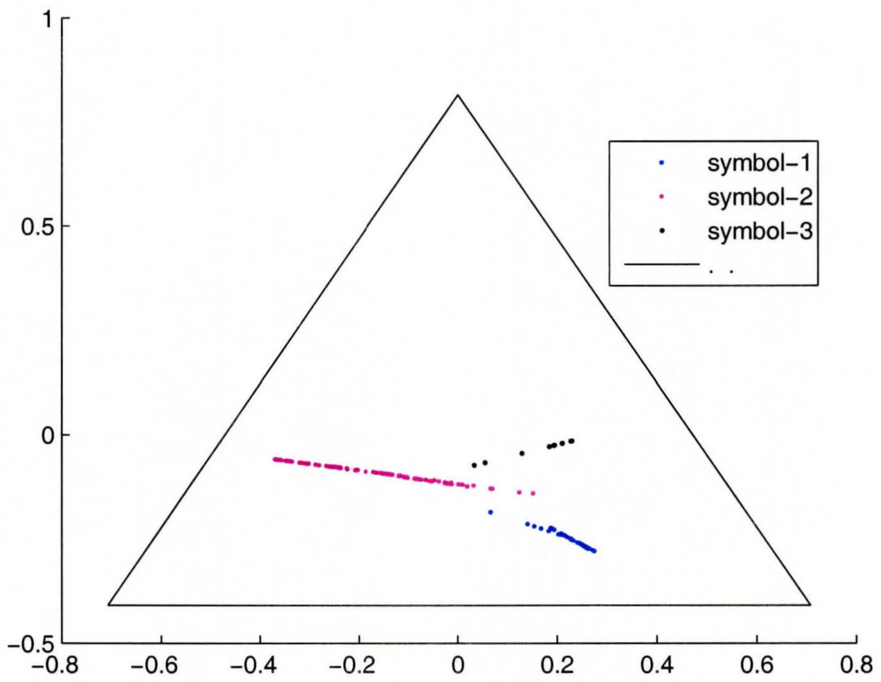

Figure 4.4: Fingerprint Plots

Chapter 5

Learning OOMs

The HMM/EM algorithm [10] finds the local maximum of a likelihood function when it is trained on a data set of interest. But with a large amount of data, there can be many maxima in the likelihood function and the EM algorithm may fail to find the most optimal one. Selection of a reasonably good initial guess for the starting point is therefore crucial in obtaining an optimal solution. To eliminate this limitation in HMM, the Observable Operator Model (OOM) can be employed as an alternative to HMM. Though the OOM is structurally similar to HMM, it is radically different in learning operation. The OOM allows negative entries in its state vector and operators. The relaxation in sign thus, allows us to develop a better learning algorithm than HMM/EM learning algorithm. Given the OOM $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$, OOM learning means computing the estimates for operators $(\tau_a)_{a\in\Sigma}$ from the the sample of finite length produced by an unknown stationary process.

This chapter is broadly categorized into two parts: The first part presents the basic version of OOM learning algorithm and its asymptotic correctness. The second part presents an advanced version of the OOM learning based on the *Efficiency sharpening* (ES) principle.

5.1 Basic Learning Algorithm

As mentioned in Chapter 4, interpretable OOMs are pivotal for the basic learning algorithm. In deriving the learning algorithm, it is assumed that the training sequence s is produced by a stationary process which can be modelled by some OOM, $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$ of minimal dimension m. If we assume that the OOM A is interpretable with respect to characteristic events $B_1, \ldots, B_m$, then the argument value pair for the operator $(\tau_a)_{a\in\Sigma}$ can be obtained from Proposition 3.6.1 as follows:

$$\begin{aligned}\tau_a((P(\bar{a}B_1)\cdots P(\bar{a}B_m))^T) &= \tau_a(\tau_{\bar{a}}\omega_0)\\ &= \tau_{\bar{a}a}\omega_0\\ &= (P(\bar{a}B_1)\cdots P(\bar{a}aB_m))^T \qquad (5.13)\end{aligned}$$

Probabilities of the kind $P(\bar{a}B_i)$ that make up the argument value pairs in eq (5.15) can be estimated from the training sequence s through the relative frequencies $\hat{P}_s$ of the event $\bar{a}B_i$ as follows:

$$\hat{P}_s(\bar{a}B_i) = \frac{\text{number of occurrences of words } \bar{a}\bar{b} \text{ within } s}{N - |\bar{a}B_i| + 1} \qquad (5.14)$$

where, $\bar{b} \in B_i$ and $|\bar{a}|$ = length of the string $\bar{a}$.

A linear operator on $\Re^m$ is determined by m-argument value pairs provided that arguments are linearly independent. It should be noted that similar to characteristic events, indicative events can also be thought of as rasters through which the training sequence is exploited. In other words, indicative events are also defined by partitioning of sequence space.The only difference is that the indicative events are perceived as events which describe the past process trajectories whereas characteristic events describe the future. Considering this fact, we denote A_i for an indicative event and B_i for a characteristic event throughout the thesis. Now the procedure for the learning algorithm can be summarized in the following three steps:

- Step 1: Choose characteristic events $B_1, \ldots, B_m$ and *indicative sequences* $\bar{a}_1, \ldots, \bar{a}_m$ such that the matrix $\hat{V} = (\hat{P}_s(\bar{a}_j B_i))_{i,j=1,\cdots,m}$ is invertible (this matrix contains m linearly independent argument vectors for the operators τ_a in its columns)
- Step 2: For each $a \in O$, collect the corresponding value vectors in a matrix $\hat{W}_a = (\hat{P}_s(\bar{a}_j a B_i))_{i,j=1,\cdots,m}$
- Step 3: Obtain an estimate for τ_a by,

$$\hat{\tau}_a = \hat{W}_a \hat{V}^{-1} \tag{5.15}$$

It is interesting to note that the statistical efficiency can be improved by increased exploitation of the given training data. Indicative sequences for instance, can be substituted with indicative events A_j that partition the Σ^k into m non-empty disjoint subsets. To simplify the counting further, $\hat{V}$ and $\hat{W}_a$ can be substituted with V^{raw} and W_a^{raw} as follows:

$$V^{raw} = (\text{count the number of events } A_j B_i \text{ in } s)_{i,j=1,\ldots m} \quad (5.16)$$

$$W_a^{raw} = (\text{count the number of events } A_j a B_i \text{ in } s)_{i,j=1,\ldots m} \quad (5.17)$$

It can be proved that $W_a^{raw}(V^{raw})^{-1} \approx \hat{W}_a(\hat{V})^{-1}$

5.1.1 Toy Example

Consider a generator of binary symbols 'a' and 'b' and the training sequence s of length 20 is given as '*abbbaaaabaabbbabbbbb*'. Assume that the estimated model dimension is 2; characteristic events $B_1 = \{a\}$ and $B_2 = \{b\}$ and indicative events $A_1 = \{a\}$ and $A_2 = \{b\}$. First we estimate the initial state vector ω_0 by putting

$$\begin{aligned}\hat{\omega}_0 &= \begin{pmatrix} \#a/N & \#b/N \end{pmatrix} \\ &= \begin{pmatrix} 8/20 & 12/20 \end{pmatrix}\end{aligned}$$

Using eqs (5.18) and (5.17), V^{raw} and W_a can be calculated by a single sweep of the inspection window of certain length over s as follows:

$$\begin{aligned}V^{raw} &= \begin{pmatrix} \#aa & \#ba \\ \#ab & \#bb \end{pmatrix} = \begin{pmatrix} 4 & 3 \\ 4 & 8 \end{pmatrix} \\ W_a^{raw} &= \begin{pmatrix} \#aaa & \#baa \\ \#aab & \#bab \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 1 \end{pmatrix} \\ \text{Hence, } \hat{\tau}_a &= W_a^{raw}(V^{raw})^{-1} = \begin{pmatrix} 0.4 & 0.1 \\ 0.6 & -0.1 \end{pmatrix}\end{aligned}$$

Similarly,

$$\hat{\tau}_b = \begin{pmatrix} 0.0 & 0.25 \\ 0.2 & 0.55 \end{pmatrix}$$

It is interesting to note that $\hat{\tau}_a$ is no longer a stochastic matrix as it takes a negative entry.

5.2 Asymptotic Property

OOMs are a special class of Linear Dependant Processes (LDPs) [8]. This section paraphrases the discussion of the basic OOM learning algorithm given in [8] with special emphasis to the aspect of asymptotical correctness.

Lemma 5.2.1 *Let Y_n be a stationary ergodic LDP of finite dimension with distribution P taking values in a finite alphabet Σ. Let $s \in \Sigma^*$ be a realization of $(Y_n), A_i \subset \Sigma^k, B_j \subset \Sigma^l$ and $\hat{P}_{s[1:n]}(A_i)$ and $\hat{P}_{s[1:n]}(B_j)$ the frequencies of A_i and B_j in the initial string $s[1 : n] \in \Sigma^n$ of the realization s. Let $V = (P(B_j A_i))$, $\hat{V} = \hat{P}_{s[1:n]}(B_j A_i)$, $W_a = P(B_j a A_i)$ and $\hat{W}_a = \hat{P}_{s[1:n]}(B_j a A_i)$. As $n \to \infty$, it holds*

1. $\hat{P}_{s[1:n]}(A_i) \to P(A_i)$
2. $\hat{V} \to V$
3. $\forall a \in \Sigma : \hat{W}_a \to W_a$

Corollary 5.2.2 *Assume the situation as in lemma(5.2.1). As $n \to \infty$, it holds*

1. $||V - \hat{V}|| \rightarrow 0$

2. $\forall a \in \Sigma : ||W_a - \hat{W}_a|| \rightarrow 0$

where $||.||$ *denotes the matrix 2-norm.*

Theorem 5.2.3 *Let Y_n be a stationary ergodic LDP of dimension m on a finite alphabet Σ. Let $s \in \Sigma^*$ be a single realization of (Y_n). Assume $(A_i)_i$ and $(B_j)_j$ are indicative and characteristic events of Y_n and A denote the concrete OOM of Y_n which is interpretable with respect to $(A_i)_i$. Assume that $\hat{A}_n(s)$ is the OOM estimated from the initial strings $s_{[1:n]}$. By applying the basic learning algorithm using $(A_i)_i$ and $(B_j)_j$, it holds*

$$\hat{A}_n(s) \rightarrow A \text{ as } n \rightarrow \infty \text{ in some matrix norm sense}$$

5.2.1 Properties of the Basic Learning Algorithm

We have seen that the asymptotic correctness of the basic learning algorithm in the previous section given the training sequence of infinite length. This section summarizes the properties of the basic learning algorithm.

1. Conceptually transparent and computationally cheap.

2. Complexity: Run time complexity is $O(N + m^3)$ and the space complexity is $O(N \log(m) + |\Sigma| m^2)$ where N is the length of the sample, m is the model dimension and $|\Sigma|$ is the length of the alphabet.

3. Event selection: The statistical efficiency of the basic learning algorithm heavily depends on the selection of indicative and characteristic events. In other words, we have to choose the characteristic and indicative events such

that counting matrix V has to be a matrix with condition number closer to 1.

4. Counting statistics: Small portion of the counting statistics, typically twice the length of characteristic events are entered into the estimation algorithm. Much information contained in the training data is thus ignored.

As just outlined, the major challenge in the basic version of algorithm is to find good indicative and characteristic events so that the estimated OOM could capture the *m significant dimension* of the underlying dynamics. In other words, our task is not to find the true dimension but the dimension which captures the data such that the data is neither over-fitted nor underexploited. In order to achieve this objective, the counting matrix V^{raw} has to be selected with the numerical rank closer to 1. In other words, the smallest singular value of V^{raw} should be significantly larger than 0 to become the full ranked matrix. The question now is how to find V^{raw} with numerical rank closer to 1 with less computational overhead.

Applying a small perturbation in V^{raw}, we can show that

$$num.rank(V^{raw}) = m \iff \sigma_{min}(V^{raw}) > \varepsilon.||V^{raw}||_\infty \tag{5.18}$$

where,

σ_{min} is the smallest singular value of V

ε is the average relative error of matrix entries $(||V^{raw} - V^{raw}_{perturbed}||)$

and $||.||_\infty$ refers to the infinity norm.

Moreover it can be shown that

$$\varepsilon = \frac{1}{m^2}\sum_{i,j}\frac{1}{A_iA_j}.(\#(A_iA_j)(1-\#(A_iA_j)/N))^{1/n} \qquad (5.19)$$

under the assumption that $\#(A_iA_j)$ is binomially distributed. It should be noted that $|\Sigma|^k \geq m$ often works well in practice, where $|\Sigma|$ is the length of the alphabet and k is the event length.

5.3 Learning Algorithm Based on Efficiency Sharpening Principle

The Efficiency Sharpening (ES) method is iterative. In each iteration, the model estimated in the previous round is used to construct an estimator with a better statistical efficiency than the previous one. This has led to the naming *Efficiency Sharpening*. In this framework, though the characteristic and indicative events are used in the initial model, the subsequent models obtained iteratively exploit only the states of the previous model as replacement to characteristic and indicative events. The influence on the choice of good characteristic and indicative events is thus eliminated in ES based learning algorithm. The computational load is however, comparable to HMM/EM iteration, typically 2-5 iterations are needed to get a reasonably good estimated model. Moreover, the accuracy of OOM models (training and test likelihoods) is superior to HMM, except on data sets that have been generated by HMMs in the first place. This chapter presents a number of relevant theorems and propositions required to establish the learning algorithm. This chapter also discusses the two versions of OOM/ES learning algorithm - Poor

Man's version and the version based on suffix tree.

As a generalization of characteristic events, the term *characterizers* is introduced in the following definition. We will see how various types of characterizers can be generated, their properties and the role they play in ES based learning algorithm.

Definition let $A = (\Re^n, (\tau_a)_{a\in\Sigma}, \omega_0)$ be a (not necessarily minimal-dimensional) OOM of an m-dimensional process (Y_n). Let $k \in \mathbb{N}$. A function $c : O^k \rightarrow \{r \in \Re^n | \mathbf{1}r = 1\}$ is a characterizer of A (of length k) if

$$\forall \bar{a} \in \Sigma^*: \omega_{\bar{a}} = \sum_{\bar{b}\in\Sigma^k} P(\bar{b}|\bar{a})c(\bar{b}) \tag{5.20}$$

We will identify c with a matrix $C = [c(\bar{b}_1), ..., c(\bar{b}_\kappa)]$ in the following, where $\bar{b}_1, ..., \bar{b}_\kappa$ is the alphabetical enumeration of O^k.

Proposition 5.3.1 *Let κ and $\bar{b}_1, ...\bar{b}_\kappa$ be as in the above definition. Given an m-dimensional process Y_n, a $n \times \kappa$matrix C each column sum of which is equal to 1 is a characterizer of some m-dimensional OOM for Y_n if and only if there exist m sequences $\bar{a}_j$ such that the $n \times m$ product matrix $W = CV$of C and the $\kappa \times m$ matrix $V = P(\bar{b}_i|\bar{a}_j)$ has rank m.*

Now consider two equivalent minimal-dimensional OOMs A and A' which are related by $\tau'_a = \rho\tau_a\rho^{-1}$, then we find

Proposition 5.3.2 *if C is a characterizer of A, then $\rho \circ C$ is a characterizer of A'.*

Proposition 5.3.3 *Let C_0 be a characterizer of length k of a minimal dimensional OOM A. Let κ and V be as in the Proposition 5.3.1. Then C is another characterizer of length k of A if and only if it can be written as $C = C_0 + D$, where*

$$D = [d_1, \ldots, d_{m-1}, \sum_{i=1,\ldots,m-1} (-d_i)]^T \tag{5.21}$$

where d_i is any vectors from $ker(V^T)$.

Proposition 5.3.4 *Let the dimension of (Y_n) be m and let $A^r = (\Re^m, (\tau_a^r)_{a\in\Sigma}, \omega_0)$ be a reverse OOM for (Y_n) that was derived from a forward OOM $A = (\Re^m, (\tau_a)_{a\in\Sigma}, \omega_0)$. Let k_0 be the characterizing length of (Y_n) and $k \geq k_0$. Then the following two statements hold:*

1. *$C = [\omega^r_{\bar{b}_1}, \cdots, \omega^r_{\bar{b}_\kappa}]$ is a characterizer of an OOM A' for (Y_n).*
2. *The states of $\omega'_{\bar{a}}$ of A' are related to states $\omega_{\bar{a}}$ of A by the transformation $\omega'_{\bar{a}} = \rho.\omega_{\bar{a}}$, where $\rho = C\pi_A$. In addition, if $\omega_0 = (1/m, \ldots, 1/m)^T$, then $\rho = R^T R$. The matrix π_A and R are given as*

$$\pi_A = \begin{pmatrix} 1\tau_{\bar{b}_1} \\ \vdots \\ 1\tau\bar{b}_\kappa \end{pmatrix} \tag{5.22}$$

$$R = \pi_A diag((mP(\bar{b}_1))^{-1/2}, \cdots, (mP(\bar{b}_1))^{-1/2}) \tag{5.23}$$

All the above propositions state various means of obtaining characterizers and their relevant properties. The following definition states how the statistical model variance can be expressed in terms of characterizers that are pivotal for the ES principle.

Definition Consider a m-dimensional process (Y_n) with characterizing length k_0. Let $k \geq k_0$and $(\bar{a}_j)_{j=1,\ldots,\kappa} = (\bar{b}_j)_{j=1,\ldots,\kappa}$, where $\bar{a}_j$ and $\bar{b}_j$ are the alphabetical enumerations of Σ^k. Let $\underline{C}$ be any $\kappa \times \kappa$ matrix and c_j denote the j^{th} column of $\underline{C}$. The mean square error is then given as

$$\xi_{\underline{C}} = \sum_{\bar{a}_j,\bar{b}_i \in \Sigma^k} P(\bar{a}_j\bar{b}_i) \| \begin{pmatrix} P(\bar{b}_1|\bar{a}_j) \\ \vdots \\ P(\bar{b}_\kappa|\bar{a}_j) \end{pmatrix} - \underline{c}_j \|^2 \qquad (5.24)$$

Proposition 5.3.5 *Let $\underline{U} = (P(\bar{b}_i|\bar{a}_j))_{1 \leq i,j \leq \kappa}$ be the matrix of forward conditional probabilities and $\underline{U}^r = (P^r(\bar{a}_i|\bar{b}_j))_{1 \leq i,j \leq \kappa}$ be the matrix of reverse conditional probabilities. Then*

$$\arg\min_{\underline{C}} \xi_{\underline{C}} = \underline{UU}^r \qquad (5.25)$$

5.3.1 Poor Man's Version of ES Principle

This learning algorithm is rooted in the estimates $\hat{V}$ and $\hat{W}_a$. The model variance across different training sequence (i.e., the statistical efficiency of the estimator) hinges among factors such as the condition of $\hat{V}$, variance of the estimates $\hat{V}$ and $\hat{W}_a$ etc. The initial model is obtained using the basic learning algorithm. According to Proposition 5.3.5, model variance is minimized when we use the characterizer $C = \underline{UU}^r$. It is proved in [8] that $\hat{V}^{(n)}$ and $\hat{W}_a^{(n)}$ can be written in a recursive manner as follows:

$$\hat{V}^{(n)} = \hat{C}^{(n)}\hat{V}^{(n-1)} \tag{5.26}$$

$$\hat{W}_a^{(n)} = \hat{C}^{(n)}\hat{W}_a^{(n-1)} \tag{5.27}$$

Hence the estimates for the operators can be obtained using eq (5.17). The iterative procedure to find a new model estimate is terminated when the training log-likelihood of models appear to settle on a plateau. Although the poor man's strategy is simple and computationally inexpensive, the only limitation is that the indicative sequences $(\bar{a}_i)_{1\leq\kappa} = O^k$ do not adapt to the training sequence. For example, some $\bar{a}_i$ may not occur in the training sequence and some may occur only a few times. This phenomena leads to poor estimates of probability through relative frequencies.

5.3.2 Suffix Tree-Based ES Principle

This second version is rooted in a data structure called suffix tree (ST). Using ST representation of the training sequence, one can exploit characteristic or indicative events of all possible lengths simultaneously. The subsequence counting statistics are stored in the nodes of the ST. Instead of obtaining the operator $\hat{\tau}_a$ from m-argument value pairs contained in the counting matrices V^{raw} and W_a^{raw}, all the counting values are exploited to obtain argument vectors. It should be noted that the number of used argument value pairs is in the order of the training data size. In short, ST is exploited to represent the following:

1. training sample
2. partitioning of the characteristic and indicative events

3. for counting statistics.

The following subsection therefore, recapitulates the basics of ST.

Suffix Tree:

Suffix Tree (ST) is a data structure that exposes the internal structure of a symbol sequence in a deep way. It turns out that all the possible substrings found inside a string. Since ST belongs to the member of a *Trie* family, we will first see the *suffix trie*. In fact, the word 'trie' comes from the word 'retrieval'. A trie is a k-ary position tree. It is constructed from input strings. That is, the input is a set of n strings called $s_1, s_2, \ldots, s_n$, where each s_i consists of symbols belonging to a finite alphabet and has a unique terminal symbol (also known as sentinel symbol) which we call $. Figure 5.1 illustrates how a suffix trie is constructed from the string *GOOGOL*.

String-G O O G O L $
Indexing 1 2 3 4 5 6 7

To obtain the suffix tree, we now replace every substring by a pair of indices (a, b), where 'a' is the the index representing the beginning of the string and 'b' is index for the end of the string. (e.g., (3,7) for OGOL$, see Figure 5.3.)

Common Applications of ST:

1. To perform the basic operations in a large text (e.g., searching, insertion and deletion of the text found in a dictionary).
2. To store the data in a more compressed way.

3. To search the pattern in a picture file.

Note:- Section 5.3.2 presents only a glimpse of how the so-called suffix tree data structure is exploited in OOM learning algorithm. Any interested reader may refer [10] to find more details in depth.

5.4 Simulation

This section describes a computer simulation carried out with the purpose of comparing the performance of OOM/ES learning algorithm against the HMM/EM learning algorithm. It includes two different results obtained from the computer simulation using Poor Man version of OOM/ES algorithm and HMM/EM algorithm. In this simulation, a randomly created HMM with 4 states and 3 output symbols were used to generate a symbol sequence of 10,000 steps. The generated symbol sequence was then partitioned into 2 portions: a training sequence of 8,000 steps long and a test sequence of 2,000 steps long. One hundred such train/test pairs were used to train and test 100 OOMs of dimension 3. For comparison purpose, 100 HMMs with 3 states (same dimension like OOM) were trained with EM algorithm. The EM algorithm was run for at most 100 iterations and stopped earlier when the ratio between two successive training log-likelihoods sank below 1×10^{-5}. It should be noted that the log-likelihood obtained via HMM/EM algorithm at the end of the hundredth iteration is plotted in Figure 5.4.

The observations collected from Figure 5.4 are summarized as follows:

- On average, the initial model estimate with the basic algorithm seems satisfactory.

- With the first iteration of OOM/ES algorithm, we witness a rapid in training and test likelihood contributing to model improvement. This is due to a significant decrease in condition number of counting matrices.

- A closer inspection of the individual OOM learning runs (not shown here) shows that the curve is not always monotonically increasing. There can be a drop of training log-likelihood in the first iteration followed by a jump towards the plateau level only in the second iteration. This clearly shows a distinguishing feature of OOM/ES algorithm. In the case of HMM/EM algorithm, the likelihood development is not bumpy as it always tries to minimize the training error. In contrast, OMM/ES principle tries to find an estimator with a better statistical efficiency at each iterative step and the estimator with higher statistical efficiency does not always mean a model with higher likelihood.

- When compared to HMM/EM learning algorithm, the training log-likelihood of OOM is higher by approximately 0.5-1.0%. This reflects the greater expressiveness of OOMs. Since the log-likelihood function in general has a rugged surface, the HMM/EM algorithm often gets trapped in one of the suboptimal local maxima. This is not the case for OOM/ES.

- OOM test likelihood (OOM-test) is slightly lower than the corresponding HMM one (HMM-test). This aspect can be attributed to the fact that HMMs have a built-in bias for this particular kind of data.

5.5 Summary

This chapter presents three different versions of the OOM learning algorithm namely, the Basic version, ES/Poor man version, and ES/Suffix tree version. The basic version of the learning algorithm is rooted in the counting statistics of a training sequence. Though it is simple and transparent, the statistical efficiency crucially depends on the selection of indicative and characteristic events. In order to circumvent this limitation, ES principle is introduced. Finally, two different ES principle based learning algorithms are presented. Computer simulation is carried out, based on OOM/ES-Poor Man version to validate relative merits of OOMs over HMMs.

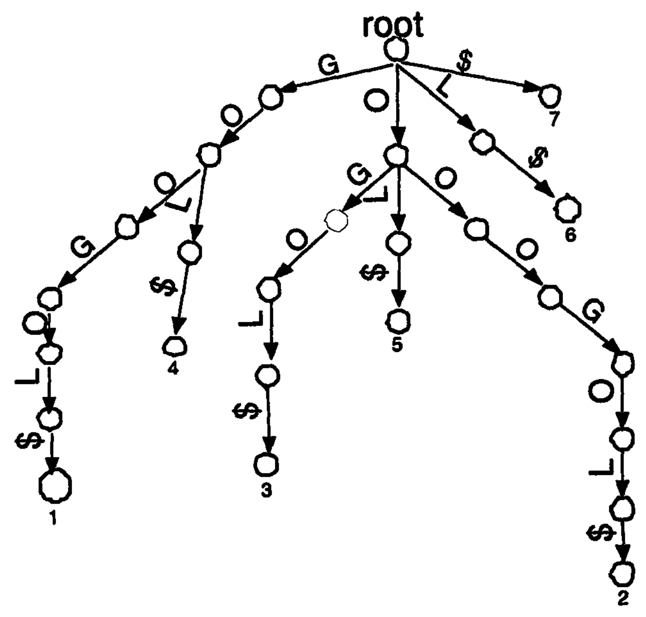

Figure 5.1: Suffix Trie of the Text *GOOGOL$*

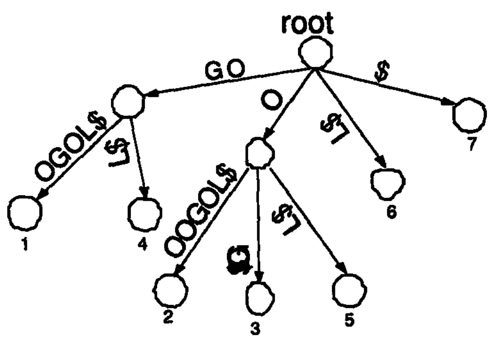

Figure 5.2: Compact Trie of Suffixes of the Text *GOOGOL$*

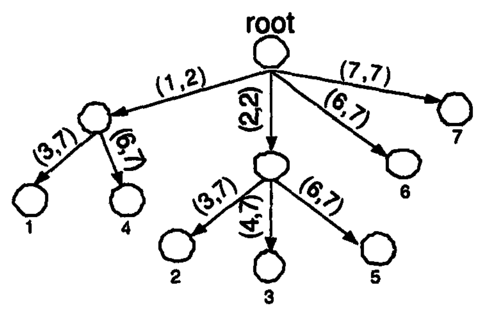

Figure 5.3: Suffix Tree

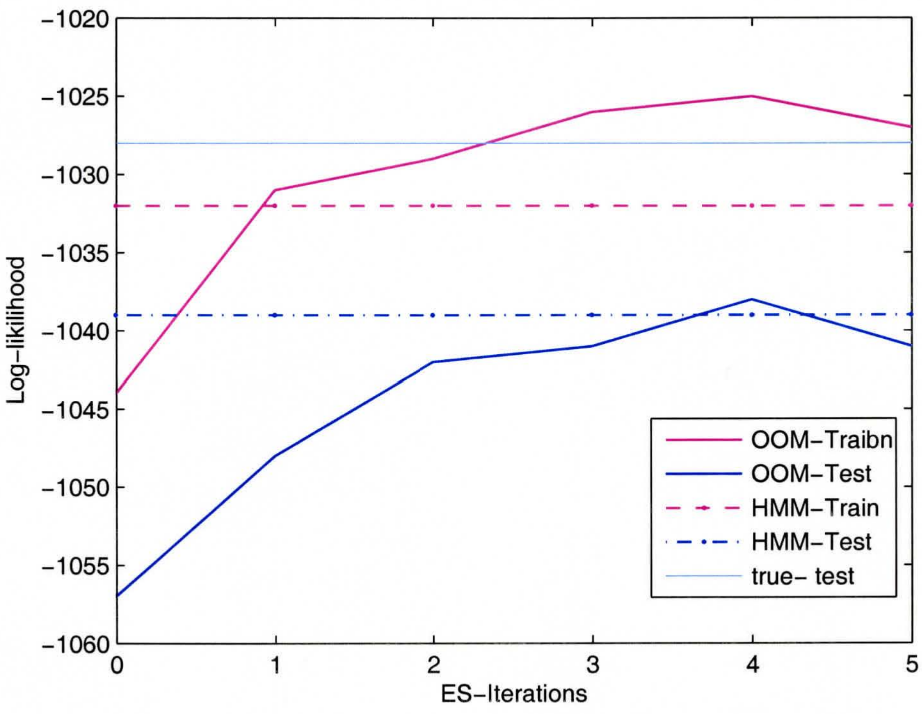

Figure 5.4: Comparison of HMM/EM against OOM/ES learning algorithm

Chapter 6

Grid Filter-Based Mode Estimation

Since mode estimation of MFR is formulated in grid-based filtering frame work, this chapter begins with an introduction to the grid-based filtering approach. The solution of a grid filter is optimal when the state space is discrete and has finite number of states. Section 6.2 describes how mode estimation of a Multi-Function Radar (MFR) is formulated as a filtering problem which exploits the pre-trained mode specific OOMs in its calculation. It also presents how one can estimate the design parameters involved in this filtering approach. Practicality of the proposed approach is finally tested for an electronic warfare scenario by means of the sanitized version of Mercury type MFR example having the signal structure as described in Chapter 2.

6.1 Introduction to Grid Filters

In order to analyze and make inference about a dynamic system, at least two models are required: First, the system model describing the evolution of the state with time and second, the measurement model relating to the noisy measurement to the state. In general, these two models are available in probabilistic form. The probabilistic state space model and the requirement for the updating of information on receipt of new measurement are ideally suited for the Bayesian approach [15].

In the Bayesian approach to dynamic state estimation, a complete solution to the estimation problem is given by the posterior probability density function, as it embodies all the statistical information. One can obtain the optimal estimate with respect to any criterion from the posterior density along with the measure of accuracy. Unlike the parameter estimation, state estimation in general, requires the estimate for the state at each sampling time. In this case a recursive filtering approach is appropriate. In a recursive filter approach, one needs not to store all the past history of data. It includes two steps:

- Prediction
- Update

In the Bayesian filtering frame work, the update operation on the receipt of new measurement is achieved through Bayes' theorem.

6.1.1 Algorithm for the Grid Filter

In a grid filtering approach, the continuous state space is decomposed into particular number of cells. In other words, grid points represent regions of continuous space. In literature [15], a grid filter is also known as a HMM filter. This section presents the grid based filtering algorithm which follows the generalized filtering frame work.

To define the dynamical system, consider the system model evolution of the state sequence $\{x_k, k \in \mathbb{N}\}$ of a system defined by

$$x_k = f_k(x_{k-1}, v_{k-1}) \tag{6.28}$$

where

$f_k : \Re^{n_x} \times \Re^{n_v} \longrightarrow \Re^{n_x}$ is a non-linear function of x_{k-1}

$\{v_{k-1}, k \in \mathbb{N}\}$ is an identically and independently distributed (i.i.d) process noise sequence

n_x & n_v are dimension of state and process noise vectors respectively.

The measurement model of the system is defined by

$$y_k = h_k(x_k, n_k) \tag{6.29}$$

where

$h_k : \Re^{n_x} \times \Re^{n_n} \longrightarrow \Re^{n_y}$ is possibly a non-linear function

$\{n_k, k \in \mathbb{N}\}$ is an i.i.d measurement noise sequence

n_y & n_n are dimensions of measurement and measurement noise vectors respectively.

The objective of the filter is to estimate some degree of belief in the state x_k, given all the available information (measurement) $Y^k = \{y_i | i = 1 \ldots k\}$. In other words, it is required to calculate the posterior density function $p(x_k|Y^k)$. Assuming that the prior $p(x_0)$ is given, $p(x_k|Y^k)$ can be sequentially obtained in two stages: prediction and update.

The prediction stage uses system model to predict the state pdf one time step ahead. Assuming that the required pdf $p(x_{k-1}|Y^{k-1})$ at time $(k-1)$ is available, prediction distribution is given by the Chapman-Kolmogorov equation as follows:

$$p(x_k|Y^{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|Y^{k-1})dx_{k-1} \tag{6.30}$$

The update operation uses the latest measurement y_k to modify the predictive distribution and this can be achieved via Bayes' rule.

$$p(x_k|Y^k) = \frac{p(y_k|x_k)p(x_k|Y^{k-1})}{p(y_k|Y^{k-1})} \tag{6.31}$$

where the normalization constant $p(y_k|Y^{k-1}) = \int p(y_k|x_k)p(x_k|Y^{k-1})dx_k$.

The recursive relations eq (6.30) and eq (6.31) form the basis for a recursive propagation of the posterior probability density function or posterior pdf in short. Next, let us see how this recursive relationship can be extended to state space model which satisfies the following two conditions:

- State space model is discrete
- Number of states in the state space is finite

Suppose the state space consists of discrete states $x_i, i = 1, \ldots r$. For each state x_i, let the conditional probability of the state given measurement up to time step

$(k-1)$ be denoted by $\omega^i_{k-1|k-1}$, where

$$\omega^i_{k-1|k-1} = p(x(k-1) = x_i | Y^{k-1}) \tag{6.32}$$

Substitution of eq (6.32) into eq (6.30) and eq (6.31) yields the prediction and update equation at time step k respectively as follows.

$$p(x_k|Y^{k-1}) = \sum_{i=1}^{r} \omega^i_{k|k-1} \delta(x_k - x_i) \tag{6.33}$$

$$p(x_k|Y^k) = \sum_{i=1}^{r} \omega^i_{k|k} \delta(x_k - x_i) \tag{6.34}$$

where

$$\omega^i_{k|k-1} = p(x(k) = x_i|Y^{k-1}) = \sum_{j=1}^{r} \omega^j_{k-1|k-1} p(x(k) = x_i | x(k-1) = x_j) \tag{6.35}$$

$$\omega^i_{k|k} = p(x(k) = x_i|Y^k) = \frac{\omega^i_{k|k-1} p(y_k|x(k) = x_i)}{\sum_{j=1}^{r} \omega^j_{k|k-1} p(y_k|x(k) = x_i)}, i = 1, \ldots \tag{6.36}$$

Eqs (6.33)-(6.36) assume that $p(x(k) = x_i|x(k-1) = x_j)$ and $p(y_k|x(k) = x_i)$ for all $i, j = 1, \ldots, r$ at every time step are known but do not constrain the particular form of these discrete densities. Again, this is the optimal solution if the assumptions hold.

6.2 Problem Formulation

Let the discrete variable $M(k)$ be the mode of operation (state) at time step k. $M(k)$ is in effect during the period starting at time $(k-1)^+$ and ending at k. Such systems are called jump linear systems. The mode jump process is assumed to be

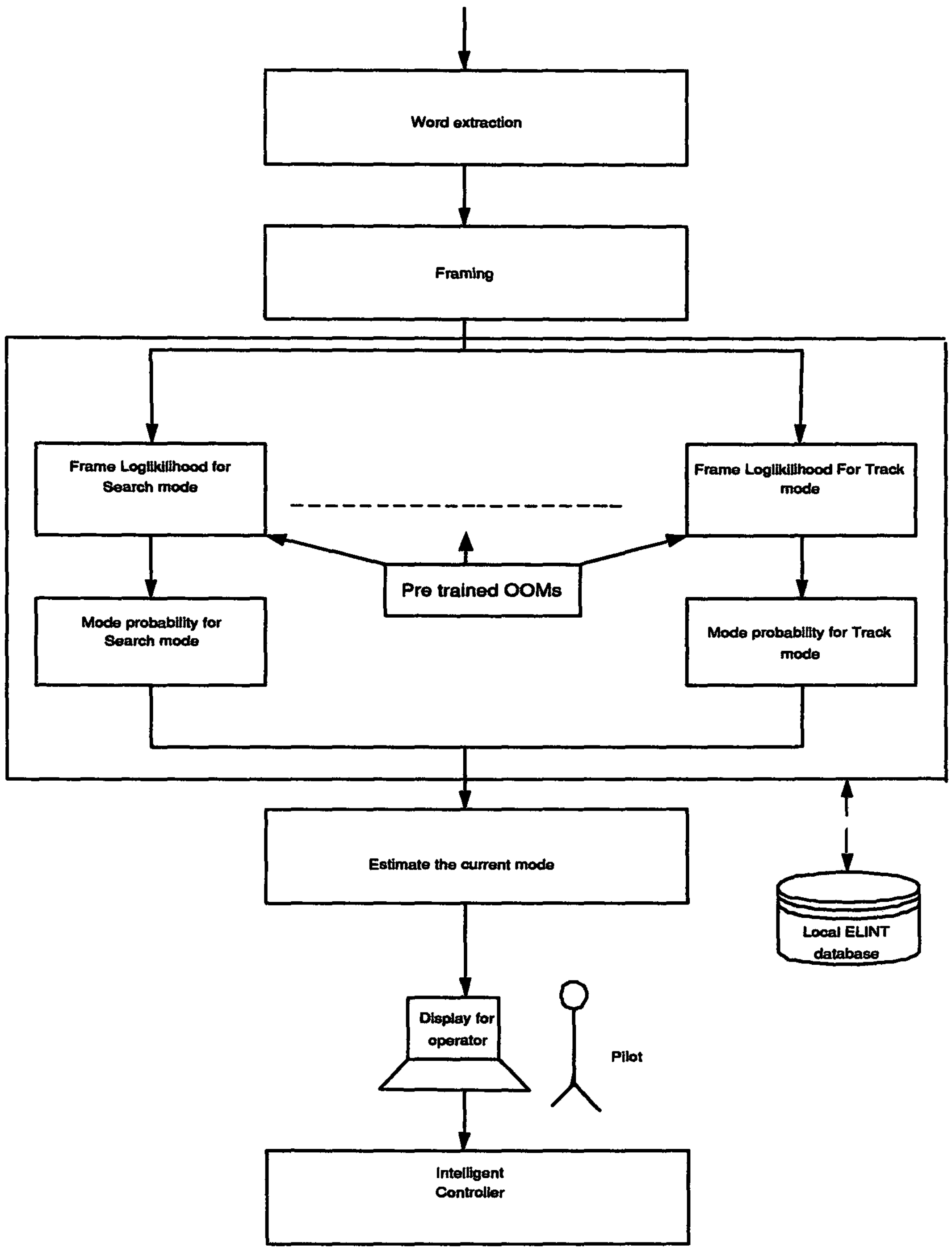
Word extraction
Framing
Frame Loglikilihood for Search mode
Frame Loglikilihood For Track mode
Pre trained OOMs
Mode probability for Search mode
Mode probability for Track mode
Estimate the current mode
Local ELINT database
Display for operator
Pilot
Intelligent Controller

Figure 6.1: Pictorial Representation of Mode Estimation

left-continuous (i.e., the impact of new mode starts at k^+). The mode at time k is assumed to be among the possible r models as follows:

$$M(k) \in \{M_i\}_{i=1}^{r} \tag{6.37}$$

The prior probability that M_i is correct or the system is in mode i is given as

$$p(M(0) = M_i | Y^0) \;\; = \;\; \mu_i(0) \tag{6.38}$$

where,

Y^0 is the prior information at time 0 and

$$\sum_{i=1}^{r} \mu_i(0) \;\; = \;\; 1 \tag{6.39}$$

It is assumed that the mode switching or the mode jump process is a Markovian process (Markov chain) with the known mode transition probabilities (MTPs) as follows:

$$p_{ij} \;\; = \;\; P(M(k) = M_j | M(k-1) = M_i) \tag{6.40}$$

Also the mode transition probability will be assumed to be time invariant. In other words, it is a homogenous Markov chain. It should be noted that the state machine of a MFR is also a time homogenous Markov chain. The following section describes the estimation algorithm where x_k in the grid filter frame work is substituted by M_k.

The mode estimation algorithm includes the following three steps:

- step 1: Mode likelihood estimation.
- step 2: Mode probability update.
- step 3: Current mode estimation.

6.2.1 Mode Likelihood Estimation

As depicted in Figure 6.1, the stream of words coming out of the word extractor is fed to the framing module. The framing module breaks the sequence into non overlapping frames of fixed size. The choice of the frame size is a design parameter. The symbolic sequence received at the framing module may contain errors in the form of mismatched or missing words. The frame likelihood calculation is carried out on a frame-by-frame basis. Framing of the word sequence is motivated by the following factors:

1. Processing of a word sequence frame by frame helps detect the mode jump in a timely manner. It is noteworthy that the frames with smaller sizes increase the chance of on-time detectability at the expense of computational overhead.

2. Since the likelihood is calculated on the basis of frames, error growth of mismatched model can be kept under control. This, in turn, allows soft switching when one of the mismatched models in the past is selected as the matched one at present.

3. OOM/ES learning algorithm was originally developed for a stationary process. The radar environment is non-stationary. Since the likelihood calculation is performed on frames that occupy short periods of time, the application of OOM/ES seems valid even in a non-stationary environment.

Now let us see why and how the mode likelihood is calculated. Baysian based filtering normally requires two defining equations as described in Section 6.1 :(i). the state equation:- In the context of this work the state refers to the current mode of operation of the MFR, M_k and the state equation describes the nature of mode switching of MFR state machine which is indeed a Markovian process as described in Chapter 2. (ii). a measurement equation:- In this context, measurements are word sequences (frames of words) coming out from the pre-processing stage. In the MFR mode estimation problem, we assume that we do not have an access to the optimal measurement equation and the measurement equation cannot be therefore explicitly written as shown in eq (6.29). Consequently, the likelihood estimate at k or $p(y_k|M(k) = M_i), i = 1, \dots, r$ is unavailable for the update step of the grid filter. To tackle this issue, each operating regime (mode) of a MFR is modelled by a distinct observable operator model (OOM). The application of the OOM over the other stochastic modelling techniques is motivated by virtues of OOM as mentioned in Chapter 4. Assume that we have a set of training sequences available for each mode of operation. In order to estimate the likelihood, we perform the following:

1. For each mode of operation M_i, we build an OOM $(\Re^m, (\tau_a)_{a \in \Sigma}, \omega_0)$ and estimate the model parameters that optimize the likelihood of the training set of observation for the i^{th} mode. In other words, the log-likelihood is used as a metric to obtain the optimal value for a model parameter. OOMs

obtained at the end of this step are called *mode-specific pre-trained OOMs*.

2. During the actual operation, word sequences (frames) are recognized (which mode those frames come from) by employing these mode-specific pre-trained OOMs.

This is the very approach taken in the isolated word HMM recognizer [14]. The i^{th} mode-likelihood estimate for the frame received at time step k, $\Omega_i(k)$ is evaluated as follows:

$$\begin{aligned} \Omega_i(k) &= p(y_k | M(k) = M_i) \\ &= \mathbf{1}[\tau_{y(k)}\omega_0]_i \quad i = 1, \ldots, r \end{aligned} \tag{6.41}$$

The maximum likelihood (ML) estimate, $M^{ML}(k)$ is given as

$$M^{ML}(k) = \arg \max_{M_1 \leq M_i \leq M_r} \Omega_i(k) \tag{6.42}$$

where τ and ω_0 are the operator and the initial state vector of Operator Observable Model (OOM) associated with mode i respectively. y_k represents the frame packed with a fixed number of extracted words at time k. In computer simulation, pre-trained HMMs are also employed for the mode estimation with a purpose of comparing the performance of OOMs.

6.2.2 Mode Probability Update

As shown in Figures 6.3 and 6.6, the ML estimate does not always seem reliable. Therefore, we use a grid filter to obtain a more refined version of the likelihood estimate by including the mode switching process of the MFR as a prior knowledge. The output of the grid filter or the mode probability is in fact a more smoother version of the likelihood estimate (see Figures 6.3, 6.5). The prediction and update equation of a grid filter in a mode estimation frame work are given as follows:

$$\omega^i_{k|k-1} = p(M(k) = M_i|Y^{k-1}) = \sum_{j=1}^{r} \omega^j_{k-1|k-1} p_{ij} \tag{6.43}$$

$$\omega^i_{k|k} = p(M(k) = M_i|Y^k) = \frac{\omega^i_{k|k-1}\Omega_i(k)}{\sum_{j=1}^{r} \omega^j_{k|k-1}\Omega_j(k)} \tag{6.44}$$

The mode likelihood $\Omega_i(k)$ is obtained from the first step of the estimation algorithm and the mode transition probability $p_{ij} = p(M(k) = M_i|M(k-1) = M_j)$ is assumed to be known.Therefore, each mode probability, $\omega^i_{k|k}$ $\{i = 1, \ldots, r\}$ can be computed now.

6.2.3 Current Mode Estimation

Finally, the current mode at time k is estimated in the maximum a posteriori (MAP) sense as follows:

$$M^{MAP}(k) = \arg \max_{M_1 \leq M_i \leq M_r} \omega^i_{k|k} \tag{6.45}$$

Figure 6.2 illustrates how this grid filter scheme operates for the time step k.

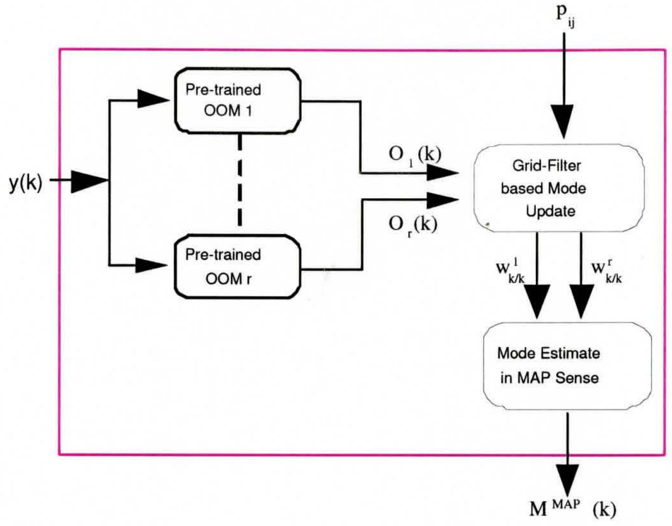

Figure 6.2: MFR mode Estimator for r models (for the time step *k*)

6.3 Structure of the Mode Transition Probability Matrix

In the previous section, the mode transition probability (MTP) which governs the mode jumps, is assumed to be completely known even though it depends critically on the unknown control inputs. MTP is therefore, a design parameter to be selected in the design process of the algorithm. Fortunately, the performance of the filter is not very sensitive to the choice of MTP provided that it is not too far off from the real value. It is noteworthy that the relationship between the average dwelling period in a state (also known as mean sojourn time) for a Markov chain and the MTP gives a good guess for the selection of MTP. Subsection 6.3.1 describes how one can roughly choose the TPM in an off-line manner.

6.3.1 Mean Sojourn Time in a State for a Markov Chain

Given that the model is in a known state (say M_i), *mean sojourn time* ($\bar{\tau}_i$) refers to the expected number of sampling periods the state i has been chosen consecutively. Let the probability that it stays in that state for exactly time τ (in units of sampling period) be $p_i(\tau)$. It can be shown that

$$p_i(\tau) = (p_{ii})^{(\tau-1)}(1-p_{ii}) \tag{6.46}$$

The quantity $p_i(\tau)$ is the (discrete) probability density function of duration τ in state i. The exponential duration density is the characteristic of the state duration in a Markov chain. Based on $p_i(\tau)$, we can readily calculate the expected number of observation (or duration) in a state conditioned on starting in that state as,

$$\begin{aligned} E(\tau_i) = \bar{\tau}_i &= \sum_{\tau=1}^{\infty} \tau p_i(\tau) \\ &= \sum_{\tau=1}^{\infty} \tau (p_{ii})^{\tau-1}(1-p_{ii}) \\ &\approx \frac{1}{1-p_{ii}} \\ \text{Hence, } p_{ii} &= 1 - \frac{1}{\bar{\tau}_i} \end{aligned} \tag{6.47}$$

The above equation which is valid for small values of $\frac{1}{\bar{\tau}_i}$, may lead to unrealistic values for p_{ii} as $\frac{1}{\bar{\tau}_i}$ approaches to 1 in MFR mode estimation problem. Therefore, a lower limit for p_{ii} has been used as follows:

$$p_{ii} = max\{l_i, 1 - \frac{1}{\bar{\tau}_i}\} \tag{6.48}$$

where l_i is the lower limit for the mode i.

The transition probability p_{ij} for $i \neq j$ is selected using the following identity,

$$\sum_{j \neq i} p_{ij} = 1 - p_{ii} \tag{6.49}$$

The probability mass $(1 - p_{ii})$ is apportioned to various possible jumps according to designer's intuition. In addition to intuition, some prior knowledge applies. Some transitions can be assigned zero probability on account of precluded mode changes as in Figure 2.3 and the transition matrix as shown in eq 6.50. In fact, the performance of grid-based filter is not very sensitive to the choice of transition probability. Nevertheless, a reasonably good choice of MTP leads to the accurate and timely detection of mode jump [1, 12].

6.4 Simulation

The computer experiment simulates the following scenario:
The ES system is fitted on an aircraft that approaches a MFR on the ground. MFR has five different radar states denoted as Search (Sea), Acquisition (Acq), Non-Adaptive Track (NAT), Range Resolution (RR) and Track Maintenance (TM)as described in Section 2.2. In the computer simulation, extracted words were fed to the mode estimation module to detect the accurate mode. Mode estimation was carried out considering four cases: Maximum Likelihood (ML) and Maximum A Posteriori (MAP) estimation based on OOMs and HMMs. ML estimate is an unfiltered estimate obtained directly from the first step of the estimation algorithm. On the other hand, MAP estimate includes all three steps of the estimation algorithm. MAP estimate is a filtered version of the likelihood estimate.

In the simulation, the frame length was taken to be twenty words, which is equal to 5 phrases as per the model described in Chapter 2. Taking into account the state transition of the radar (see Figure 2.3), the mode transition probability matrix was defined to be as follows:

$$[p_{ij}] = \begin{pmatrix} 0.80 & 0.20 & 0 & 0 & 0 \\ 0.01 & 0.80 & 0.19 & 0 & 0 \\ 0.01 & 0 & 0.80 & 0.19 & 0 \\ 0.01 & 0 & 0 & 0.80 & 0.19 \\ 0.02 & 0 & 0 & 0.03 & 0.95 \end{pmatrix} \qquad (6.50)$$

The MFR was assumed to produce the observation as those in Table 2.1. The probability to generate a phrase among the available phrases for the particular mode was defined to be uniform.

The results of the computer simulation are shown in Figures 6.3, 6.4, 6.5, 6.6, 6.7 and 6.8. *Red* line in Figures 6.3 and 6.6 shows the actual mode evolution of the radar. The initial state of the radar is search. As the aircraft approaches the radar, it enters the detection zone and the radar initiates the track for that target. The regime of this operation is referred to as target acquisition. After some time, it initiates non-adaptive tracking to track the target of interest. After the non adaptive tracking, it enters into the range resolution mode in order to resolve the range ambiguity. Finally the radar establishes the track for that target. This is known as track maintenance or continuation. Occasionally, it performs a range verification function on the track (see Figure 6.3). As the aircraft flies over and away from the radar, the distance becomes too long for the radar to continue tracking. The target track is finally abandoned and the radar switches back to search mode. The estimation results are summarized in Section 6.5.

6.5 Results

Our findings are collected from Figures 6.3, 6.4, 6.5, 6.6, 6.7 and 6.8. Here is a summary of observations of interest:

- ML estimator often makes jumps from one mode to another. In other words, ML estimate results seem unreliable.
- In MAP estimate, there exists a latency associated with the new mode onset time up to some extent.
- MAP estimator fails to detect the mode where the radar stays only for a period less than 8 phrase time units.
- In OOM case, the computation time for the likelihood estimate was 20 seconds on average when implemented on 3.07GHz Intel Pentium IV processor using MATLAB whereas it was 70-75 seconds in HMM case.
- The OOM-based grid filter tracks the mode evolution of the MFR more reliably than the corresponding HMM-based grid filter.

6.6 Summary

In this chapter, we have discussed a recursive estimator approach which assumes that the system to be in one of a finite number of modes at a time. Each model (mode) yields a model conditioned likelihood estimate evaluated over a sequence of fixed size known as frame. The likelihood estimation is performed using mode specific pre-trained OOMs and HMMs. A mode probability calculator updates the probability of each mode based on the likelihood function of each model and the prior mode probability. The mode estimator finally decides the current mode by taking a mode with the highest mode probability at that time. Computer simulations confirms that the MAP estimator significantly outperforms the ML estimator. Moreover it is observed from the simulations that the performance of OOM is significantly better than that of a HMM given the same scenario. This can be attributed to the fact that the models obtained via the OOM/ES learning algorithm is markedly more accurate than the corresponding models obtained via HMM/EM algorithm. The OOM based grid filter, in turn, yields more accurate mode estimate than a HMM based grid filter.

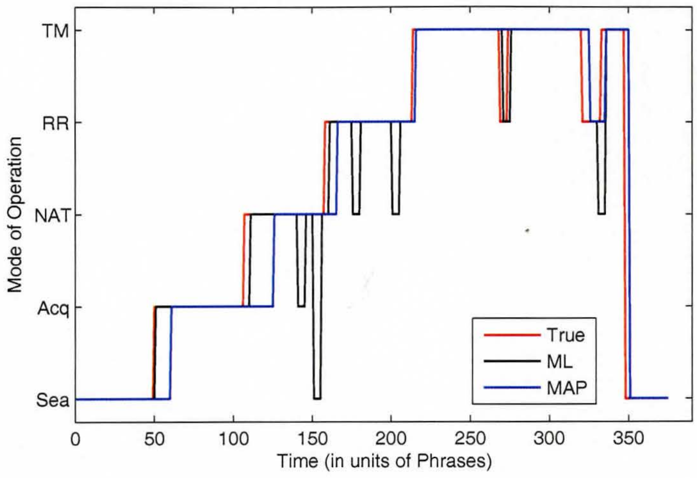

Figure 6.3: OOM-Based ML & MAP Estimates $\forall s$ True Mode Evolution

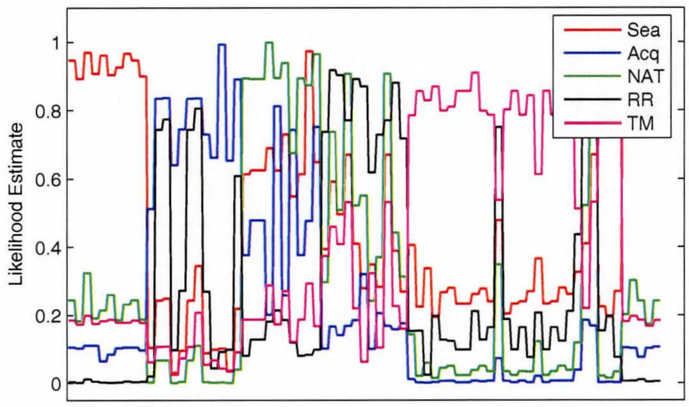

Figure 6.4: OOM-Based Likelihood Estimate for Five different Modes

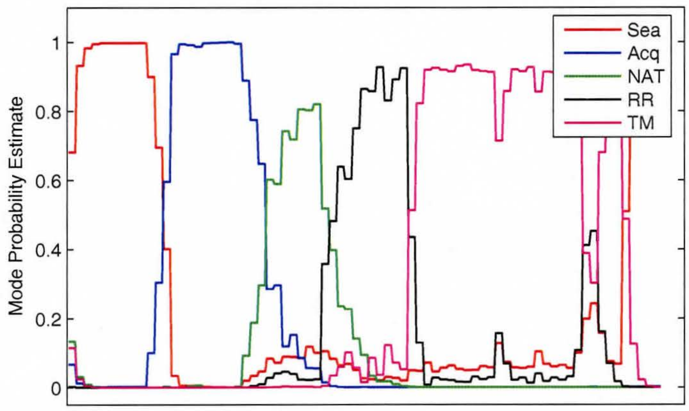

Figure 6.5: OOM-Based Mode Probability Estimate for Five different Modes

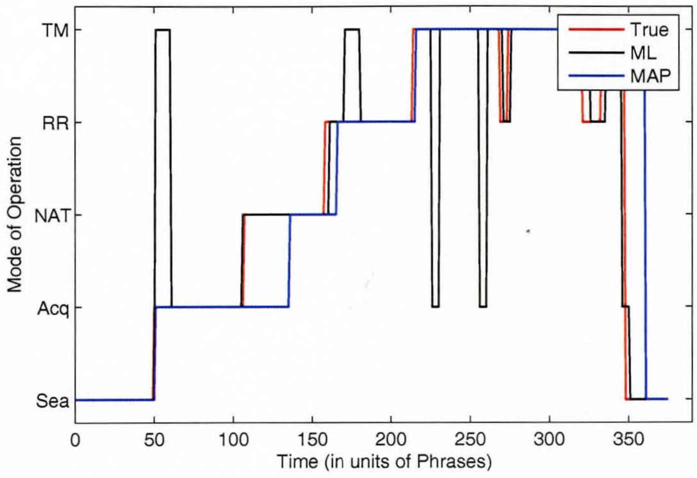

Figure 6.6: HMM-Based ML & MAP Estimates $\forall s$ True Mode Evolution

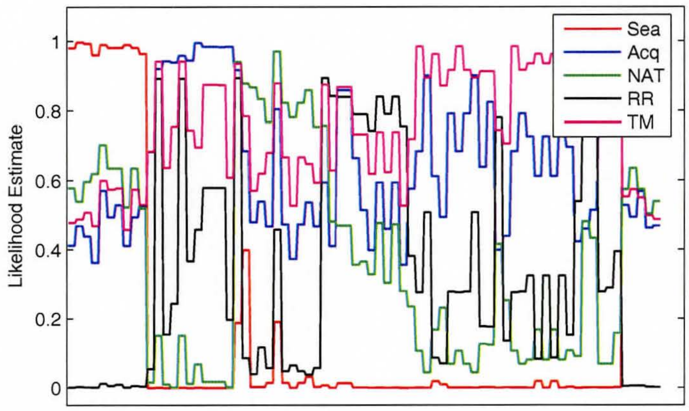

Figure 6.7: HMM-Based Likelihood Estimate for Five different Modes

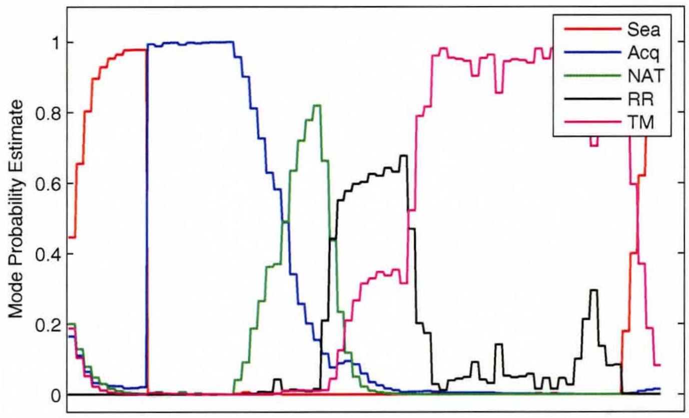

Figure 6.8: HMM-Based Mode Probability Estimate for Five different Modes

Chapter 7

Discussion

This thesis has undertaken a detailed examination of the problem of estimating the mode of operation of MFR. In this chapter, we conclude the thesis by summarizing the main results of our investigation and highlighting the most important contributions. Finally, some recommendations for potential future research work that can be carried out on the MFR problem are presented.

7.1 Concluding Remarks

This thesis presents a novel way of tracking the mode of operation of Multi-Function Radar (MFR). Since radar broadcasts a very complex electro magnetic pulse sequence to perform multiple functions in time multiplexed manner, a layered signal architecture is proposed to keep the complexity of signal architecture manageable. The mode estimation is then performed at a higher level of the layered structure known as word level. This investigation also presents two types of stochastic modelling techniques known as Observable Operator Models (OOMs) and Hidden Markov Models (HMMs) to capture the stochastic nature of the evolution of the radar signature. The computer simulation shows that OOMs outperform HMMs by estimating more accurate mode likelihood in addition to other benefits of an OOM.

Following the mode likelihood estimation, the mode probability is calculated using a grid filter. The filtered output based on MAP decision criteria is then compared to the unfiltered output (or the estimate based on ML). On average, performance based on MAP criteria seems better than ML when one includes prior knowledge about the system in the estimation algorithm. In the context of MFR's mode estimation, mode transition probability (MTP) is included as the known a priori.

The limitation associated with the MAP estimate is that it suffers from a latency issue to some extent. It detects the onset of mode jump after some delay and this may prohibit the aircraft operator from deploying counter measures such as jamming and evasive maneuvers in timely manner. In order to mitigate this problem, two solutions are proposed:

1. Processing the frames of reduced length at the expense of increased computational overhead.

2. On-line (adaptive) tuning of the parameters of MTP –The determination of MTP amounts to identifying a Markov transition law that best fits the unknown truth similar to the process noise covariance Q in Kalman filtering. Fortunately, the performance of the Grid filter is not very sensitive to the choice of MTP, provided that it is not too far off [12].

7.2 Summary of Contributions

1. Multiple-Model Approach:

   To track the mode of operation of MFR, we have proposed multiple models, in which each model is assigned to a specific mode of operation. It is a well-known fact in control theory that the view of a large system as a collection of interacting small subsystems helps keep the tendency for the complexity to grow under control. For this reason, mode-specific OOMs are employed in estimating the operational mode of a MFR.

2. OOMs for modelling:

   Similar to HMMs, OOMs are known to be a proper subset of a wider class of stochastic process known as *linear dependent process*. OOM/ES learning algorithm in the context of MFR, offers specific benefits over HMMs. Efficiency sharpening (ES) algorithm is more accurate, compact and less computationally intensive and all these factors help to detect the mode of operation of a MFR in an accurate and timely manner [10].

3. Operational mode estimation at word level:

   In the development of MFR model, electromagnetic pulses emitted from a MFR are viewed as a sequence of symbols belonging to an alphabet of fixed size. This, in turn, helps to construct a hierarchical signal architecture providing means of clear separation of low-level elements of radar signals such as individual pulses from the higher level elements such as radar words and elements of radar state dynamics. In fact, pulse-level state estimation is more computationally demanding and erroneous than that of word-level.

4. Grid filter for tracking the mode of a MFR:

   In order to track the mode evolution of a MFR accurately, the grid filter is applied. It provides an optimal recursion for the mode probability when the number of modes of a MFR is finite. In recursive estimation, newly received data are sequentially processed so that it is not necessary to store the complete data set or to reprocess it if a new measurement becomes available [15].

7.3 Recommendations for Future Research

1. Information Gap and Robustification:

   In the context of MFR, there exists a considerable amount of reality gap between the available information about the radar and the radar in reality. In other words, ELINT library is an incomplete representation of a real radar. The radar model we develop should therefore be robust so that it will be flexible in adjusting the parameters efficiently when faced with unexpected circumstances or surprises. *Theory of Info-Gap* may be useful to tackle this problem.

2. OOM-Based Pulse Processing:

   Accuracy of word processing heavily relies on radar pulse processing and the completeness of ELINT records. For radars with static word template structure, pulse processing falls in the class of a pattern-recognition problem. This problem can also be addressed by OOMs by converting the Time of Arrivals (TOAs) into binary sequences [19].

3. Intelligent Target Controller Design:

   As discussed in the previous chapters, the aircraft takes action in response to noisy delayed version of MFR's signal. In the case of the unmanned aircraft, it initiates counter measures with the guidance of embedded intelligent controller. Also, the aircraft should be aware of the potential actions taken by the radar when it initiates its own action. In other words, the target should be aware of the consequences of its own actions. In general, the consequences of the aircraft action are highly uncertain, so in one way or an other, they must be internally represented to the aircraft in some manner. Either Input-output OOMs (IO-OOMs) or Partially Observable Markov Decision Process (POMDP) can be employed. Bellman's dynamic programming which provides a mathematical basis for determining optimal control policy for a target to take optimal control. Reinforcement learning in the form Q-learning or TD-Learning provides a computationally tractable solution in real time to Bellman's dynamic programming [3].

4. Radar Classification:

   From the target's view point, the primary objective of the target (acting as a passive sensor observing the EW scene) is to control its own behaviour in the light of emissions produced by the MFR and thereby take the appropriate action to escape the missile engagement from MFR. For this objective to be achieved, the target has to classify the radar responsible for the emissions. For the target to be aware that it has converged onto the correct class of MFR, it needs to minimize the error produced in predicting future values of the radar state as the state evolves. Signal parameters might help to classify the radar responsible for the emissions to some extent. However, unlike conventional radars, a MFR exploits sophisticated signal architecture. Sophisticated modelling technique can be considered in this regard.

Appendix A

Word Extraction Schemes

After the successful identification of a radar class which is responsible for the emissions, ESM system initiates the word extraction process over the deinterleaved pulse train with the aid of ELINT data. In reality, ESM system has to account for the following problems in extracting word symbols from noisy pulse stream:

- Electromagnetic channel is non-stationary ,characteristics of which may be unknown.
- Pulse originating from one radar may leak into another radar's pulse train during the deinterleaving process
- Synchronization is difficult to achieve as the target-emitter environment is not co-operative
- Received sequence of pulses may suffer from quantization distortions

The filtering approach for a mode estimation which follows the word extraction relies on the sequence of the radar words. Word extraction is performed at the pulse-level component of a hierarchical radar model. This chapter presents two different types of word extraction algorithm though they are conceptually similar. They are

- Event-based scoring
- HMM-based Viterbi scoring.

A.1 Event-Based Scoring Algorithm

The score of the sequence with respect to the HMM is defined in [14] as

$$P(O|\lambda_k) = \sum_{i=1}^{M_k} \alpha_L(i) \tag{A.51}$$

where

$O = \{O_1, O_2, O_3, \ldots\}$

λ_k is the HMM for the word template

Forward variable, $\alpha_L(i)$ is defined as,

$$\alpha_l(i) = P(O_1, O_2, O_3, \ldots, O_L, S_i|\lambda_k) \tag{A.52}$$

S_i is the i^{th} state of the HMM.

For the case of static word-to-pulse mapping, we may find an alternative to the HMM λ_k that is easier to use in a score evaluation procedure. We define the model for the k^{th}word template ξ_k simply as a vector of pulse of TOAs of the word k as follows:

$$\xi_k = [t_1, t_2, ..., t_{N_k}]^T \tag{A.53}$$

where,

t_i is the TOA of the i^{th} pulse

N_k is the total number of pulses in the k^{th} radar word.

There are four possible events which contribute to the probabilistic scoring. They are spurious, missing, split and matching pulse events with the probability p_{e_1}, p_{e_2}, p_{e_3},and p_{e_4} respectively. The total probabilistic score in terms of these events is defined as,

$$P(O|\xi_k) = \prod_{i=1}^{N} p_{e_i} \tag{A.54}$$

where,

$O = [\tilde{t}_1, \tilde{t}_2, \ldots]^T$ is the TOAs of pulse sequence in the reconstructed pulse domain

N = The number pulses being processed

$p_{e_1} = p_{spur}$

$p_{e_2} = (1 - p_{spur})p_{miss}$

$p_{e_3} = (1 - p_{spur})(1 - p_{miss})p_i$

$p_{e_4} = (1 - p_{spur})(1 - p_{miss})(1 - p_i)$

$p_i = \Delta\tau_i / T_{obs}$

In this investigation, we consider that the pulses emitted by a radar, could be lost with a probability p_{miss} and spurious pulses could be introduced with a probability p_{spur}. Here p_{miss} refers to the probability that no pulse is detected at a particular sampling instant given that no one should have occurred in that instant and p_{spur} is the probability that a pulse is detected at some sampling instant, given that no such pulse was emitted. The quantization distortion and phase distortions are due to to details of a specific hardware implementation of ESM system. The pulse sequence quantization process is in general controlled by the observer clock described by T_{obs}. The theoretical *Synchronized pulse* quantization model is defined as,

$$n_i = \lfloor \frac{t_i}{T_{obs}} \rfloor \tag{A.55}$$

where,

t_i is the relative TOA of the received pulse

n_i is the associated quantization index

$n = \lfloor x \rfloor$ denotes that n assumes the floor value of x.

In practice, the pulse model should include a uniformly distributed random phase $\varphi \in [0, T_{obs})$ to account for the asynchronous nature. The modified index $n_i(\varphi)$ is therefore defined as follows:

$$n_i(\varphi) = \lfloor \frac{t_i + \varphi}{T_{obs}} \rfloor = \begin{cases} n_i, & \text{with } 1 - p_i; \\ n_i + 1, & \text{with } p_i. \end{cases} \tag{A.56}$$

where the pulse splitting probability p_i is defined as

$$p_i = \frac{t_i}{T_{obs}} - n_i(\varphi) \tag{A.57}$$

From(A.57), we witness that lower the local clock period is, the lower the p_i

is. It is interesting to note that this algorithm is independent of local clock period T_{obs}. Moreover, it is conceptually transparent and efficient, especially for a static word template. The run time complexity is $O[N]$ which is computationally cheaper than HMM/Viterbi Scoring which requires the computation of order(N^2). Figure A.2 shows the simulation output of this algorithm.

A.2 HMM-Based Viterbi Scoring

In this approach, channel is assumed to be binary. Leading edge of the incoming noisy sequence is detected and its time of arrival is recorded. Subsequently, it is transformed into binary observation sequence .This is referred to as *quantization process*. After quantization, the hidden Markov model for the incoming word symbol is constructed for the word template. Now the problem of determining the radar token from the noisy and corrupted pulse sequence is equivalent to the problem of scoring the pulse sequence in Viterbi sense.

The strength of the Viterbi algorithm for the radar pulse processing and word extraction is in its ability to process highly structured pulse patterns. This algorithm does not even require a one-to-one mapping between the radar tokens and their pulse representations. The serious limitation of this algorithm is in the dependance of the number of states, M_k of a HMM word template on the quantization realization. Any interested reader may refer [21] for a detailed description of this approach.

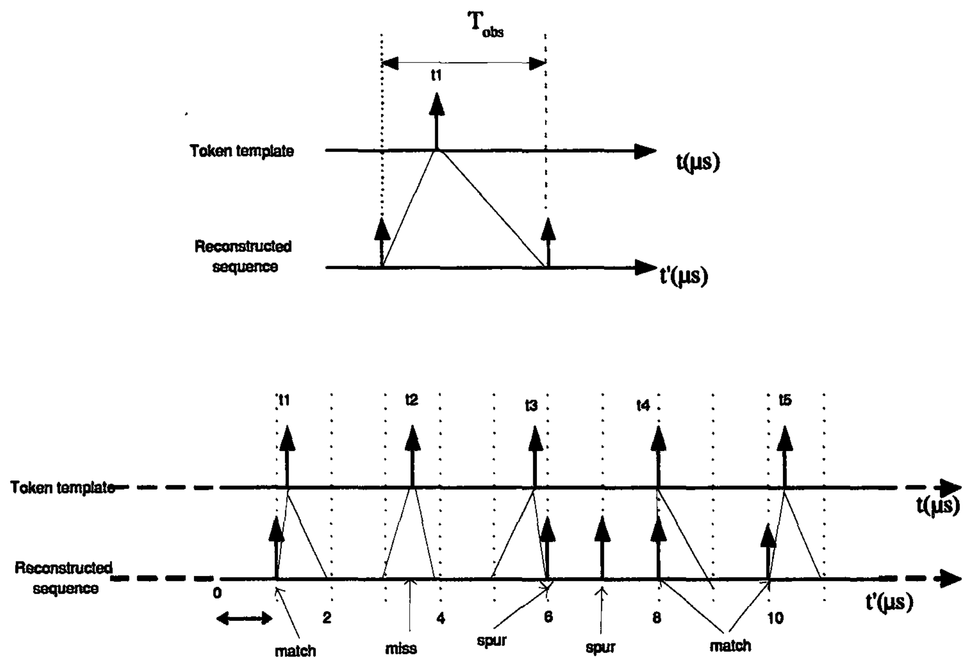

Figure A.1: Event-based Pulse Train Analysis Principle

Figure A.1 provides a geometric interpretation of the event-based analysis algorithm. Reconstructed pulses are expected to appear in either lower left or right vertices of the triangle in the figure shown at the top. Otherwise the pulse should be declared missing. The figure drawn at the bottom shows the operation of the algorithm using the specific example of a reconstructed pulse sequence. The template has a constant $PRI = 2.25\mu s$ pulse sequence of five pulses $\xi = [t_1, \ldots, t_5]^T$ with the first pulse $t_1 = 1.25\mu s$. $T_{obs} = 1\mu s$ and the reconstructed observed sequence also contains five pulses $O = [1, 6, 7, 8, 10]^T$. Six events are considered in this example. The first pulse was found when it was expected and thus it matched the word template. Similarly, how each pulse was declared, is shown in Figure A.1. The total score can therefore, be calculated using (A.54).

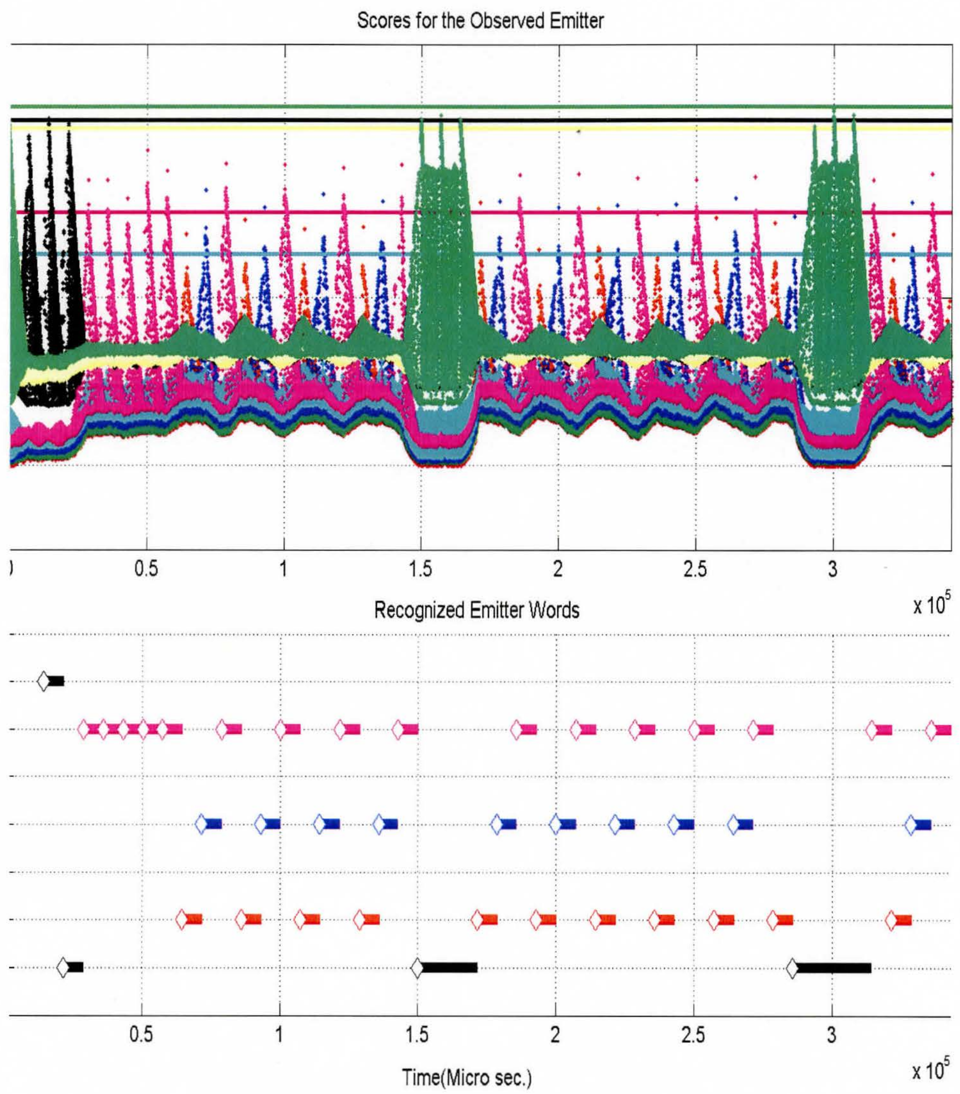

Figure A.2: Simulation results showing Event-based Scoring and Extracted Word Sequences

Bibliography

[1] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. Wiley, NY, 2001. ISBN. 0-471-41655-X.

[2] N. Chomsky, "Three-models for the Description of Language," *IRE Trans. on Info. Theory*, vol. 2, no. 3, pp. 113-124, Sept. 1956.

[3] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Prentice Hall, NJ, 1999. ISBN. 0-13-273350-1.

[4] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, MA, 2002. ISBN. 0-201-44124-1.

[5] A. L. Garcia, *Probability and Random Processes for Electrical Engineering*. Addison Wesley, MA, 1989. ISBN. 0-201-12906-X.

[6] N. Gershenfeld, *The Nature of Mathematical Modeling*. Cambridge University Press, UK, 1999. ISBN. 0-521-57095-6.

[7] H. Jaeger, "Discrete-time, Discrete-valued Observable Operator Models: a Tutorial," Tech. Report 42, GMD, Sankt Augustin, 1998.

[8] H. Jaeger, "Observable Operator Models for Discrete Stochastic Time Series," *Neural Computation* 12 (2000), no. 6, 1371-1398.

[9] H. Jaeger, "Modeling and learning continuous-valued stochastic processes with OOMs," Tech. Report 102, GMD, Sankt Augustin, 2001.

[10] H. Jaeger, M. Zhao, K. Krettzhmar, T. Obesrtein, D. Popovinci and A. Kolling, "Learning Observable Operator Models via the ES Algorithm," A Chapter in the book *New Directions in Statistical Signal Processing: from Systems to Brains.* To be published by MIT Press, editors: S.Haykin, J. C. Principe, T. Sejnowski, and J. McWhirter.

[11] F. L. Lewis, *Optimal Estimation: with an Introduction to Stochastic Control Theory.* Wiley, NY, 1986. ISBN. 0-471-83741-1.

[12] X. R. Li, Y. Zhang, and X. Zhi," Multiple-model estimation with variable structure IV: Design and evaluation of model-group switching algorithm," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 35, no. 1, pp. 242–254, Jan. 1999.

[13] J. Matuszewski and L. Paradowski,"The knowledge based approach for emitter identification," *Microwaves and Radar, 1998. MIKON '98., 12th International Conf.*, pp. 810-814, May 1998.

[14] L. R. Rabiner," A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proc. of the IEEE* , 77(2):257286, Feb. 1989.

[15] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Applications*. Artech House, UK, 2004. ISBN. 1-58053-631-X.

[16] C. Schleher, *Introduction to Electronic Warfare*. Artech House, UK, 1986. ISBN. 0-89006-142-4.

[17] M. I. Skolnik, *Introduction to Radar Systems*. McGraw-Hill, NY, 2002. ISBN. 0-072-88138-0.

[18] N. Visnevski, V. Krishnamurthy, S. Haykin, B. Currie, F. Dilkes, and P. Lavoie, "Multi-Fuction Radar Emitter Modelling:A Stochastic Discrete Event system Approach," *Proc. of the IEEE Conf. on Decision and Control(CDC'03)*, pp:6295-6300, Hawai, USA, Dec.2003.

[19] N. Visnevski, S. Haykin, V. Krishnamurthy, F. Dilkes, and P. Lavoie, "Hidden Markov Model for Radar Pulse train analysis in electronic Warfare" *Proc. of the IEEE Conf. on Acoustics, Speech and Signal Processing (ICASSP'05)*, Philadelphia, PA, USA, Mar. 2005.

[20] N. Visnevski, F. Dilkes, S. Haykin and V. Krishnamurthy," Non-self embeeding context-free grammers for multi-function radar modeling-Electronic Warfare application" *Proc. of the IEEE International Radar Conf. (IRC'05)*, Arlington, VA, USA, May 2005.

[21] N. Visnevski, "Syntactic Modelling of Multi Function Radar," PhD dissertation, Dept. of Electrical and Computer Eng., McMaster University, Canada, 2005.

[22] T. Oberstein, "Efficient Training of OOMs using Context Graphs," Masters thesis, Institute for Autonomous Intelligent System, Fraunhofer AiS, 2002.

[23] R. G. Wiley, *Electronic Intelligence: The analysis of Radar Signal.* Artech House, MA, 1993. ISBN. 089-006-5926.

[24] W. Zhu and J. G. Frias, "Stochastic context-free grammars and Hidden Markov Models for modeling of bursty channels," *IEEE Trans. Veh. Technol.*, 53(3):666676, May 2004.