

Cellular Diagnostic Systems Using Hidden Markov Models

Maruf Mohammad

Dissertation submitted to the faculty of the
Virginia Polytechnic Institute & State University
in partial fulfillment of the requirements of the degree

Doctor of Philosophy
in
Electrical Engineering

Approved:

Dr. William H. Tranter, Chairman

Dr. R. Michael Buehrer

Dr. Jeffrey H. Reed

Dr. Sedki M. Riad

Dr. James Holub

October 19, 2006
Blacksburg, Virginia

Keywords: Hidden Markov models, Kullback-Leibler divergence, 1xEV-DV, Cellular diagnostics, Dropped call prediction, Coverage problem, Position location.

Copyright 2006, Maruf Mohammad

Cellular Diagnostic Systems Using Hidden Markov Models

Maruf Mohammad

(Abstract)

Radio frequency system optimization and troubleshooting remains one of the most challenging aspects of working in a cellular network. To stay competitive, cellular providers continually monitor the performance of their networks and use this information to determine where to improve or expand services. As a result, operators are saddled with the task of wading through overwhelmingly large amounts of data in order to troubleshoot system problems. Part of the difficulty of this task is that for many complicated problems such as hand-off failure, clues about the cause of the failure are hidden deep within the statistics of underlying dynamic physical phenomena like fading, shadowing, and interference. In this research we propose that Hidden Markov Models (HMMs) be used as a method to infer signature statistics about the nature and sources of faults in a cellular system by fitting models to various time-series data measured throughout the network. By including HMMs in the network management tool, a provider can explore the statistical relationships between channel dynamics endemic to a cell and its resulting performance.

This research effort also includes a new distance measure between a pair of HMMs that approximates the Kullback-Leibler divergence (KLD). Since there is no closed-form solution to calculate the KLD between the HMMs, the proposed analytical expression is very useful in classification and identification problems. A novel HMM based position location technique has been introduced that may be very useful for applications involving cognitive radios.

Acknowledgements

I am grateful to Allah Almighty for being able to pursue a doctoral degree. I would like to express my gratitude to my advisor, Dr. William Tranter, for his advice, tutelage, and encouragement. I wish to extend special thanks to Dr. Buehrer and Dr. Jeffrey Reed for not only serving on my advising committee, but also for being great mentors both in and out of the classroom. My sincere appreciation goes to Dr. Sedki Riad and Dr. James Holub for their support.

I also wish to express my gratitude to my friends and colleagues at the Mobile and Portable Radio Research Group (MPRG). Special thanks to James Hicks for his help in writing the proposal and his advice in shaping my Ph.D. research. I would like to thank the outstanding staff at MPRG for making an atmosphere that is conducive to great research work. Special thanks to Steve Magee of Motorola for the University Partnership in Research (UPR) Fellowship which sponsored this research work. It was a great experience working with Steve and his colleagues at the Networks sector of Motorola.

Finally, on a personal note, I would like to thank my parents, my loving wife, Farhana, and my little daughter, Aishah, who have supported me with encouragement and love, and have always been a source of inspiration for me, and to whom I dedicate this thesis.

Contents

Chapter 1: Introduction.....	1
1.1 Motivation.....	1
1.2 Contributions.....	3
1.3 Disclosures, Papers and Reports	5
1.4 Overview of the Dissertation	7
 Chapter 2: Fundamentals of Hidden Markov Models.....	 9
2.1 State and Transition	9
2.2 Markov Process.....	10
2.2.1 Example of a Simple Markov Model.....	13
2.3 Hidden Markov Model.....	14
2.3.1 HMM Parameters.....	15
2.3.2 Generation of Output Sequences	17
2.3.3 Structures of HMMs	17
2.3.4 Gilbert Model.....	19
2.3.5 Fritchman's State Partitioned Model	20
2.4 Semi-Hidden Markov Processes	23
2.5 Block Equivalent Hidden Markov Models	25
2.6 Validation of Hidden Markov Models.....	26
2.7 Conclusion	28
 Chapter 3: Hidden Markov Model Parameter Estimation	 29
3.1 Minimization Algorithm	29

3.1.1	Expectation-Maximization (EM) Algorithm	31
3.1.1.1	Convergence of EM Algorithm	33
3.1.2	Phase-Type (PH) Distribution.....	34
3.2	Training Algorithms for Hidden Markov Models	37
3.2.1	Baum-Welch Algorithm (BWA).....	38
3.2.1.1	Scaling.....	45
3.2.1.2	Convergence	47
3.2.1.3	Initialization	48
3.2.1.4	Complexity and Memory Requirements	49
3.2.2	Segmental K- Means Algorithm	50
3.2.3	Genetic Algorithm	51
3.2.3.1	Encoding Mechanism.....	52
3.2.3.2	Fitness Value Evaluation	52
3.2.3.3	Selection Mechanism	53
3.2.3.4	Crossover and Mutation.....	53
3.3	Multiple Observations Training of the HMM.....	54
3.3.1	Independent Observation Sequences	56
3.3.2	Uniformly Dependent Observation Sequences	57
3.3.3	Ensemble Average of Independent Observation Sequences.....	57
3.4	Conclusion	58
Chapter 4:	Distance Measures and Parametric Analysis of HMM	59
4.1	Notion of Distance Measure	59
4.1.1	Kullback-Leibler Divergence.....	61
4.1.1.1	Properties of KLD.....	61
4.2	KLD Between Markov Models.....	62
4.2.1	Numerical Example	65
4.3	Distance Measures Between Hidden Markov Models.....	66
4.3.1	Upper Bound of KLD	67
4.3.2	Probabilistic Distance Measure.....	68
4.3.3	Heuristic Distance Measure	69

4.3.4	Numerical Examples	71
4.4	Parametric Analysis of HMM.....	73
4.4.1	Augmented Matrix	75
4.4.2	Residual Matrix.....	76
4.4.3	Example	77
4.5	Conclusion	78
 Chapter 5: 1xEV-DV – An Overview of Cellular Voice and Packet Data System		80
5.1	Cellular Standards	80
5.1.1	First Generation	81
5.1.2	Second Generation	81
5.1.3	Migration to Third Generation.....	82
5.1.4	Third Generation.....	83
5.1.5	Third Generation Evolutions.....	84
5.2	cdma2000 1xEV-DV	85
5.2.1	Air Interface Layer Architecture.....	86
5.2.2	Link Adaptation	88
5.2.3	Forward EV-DV Channels.....	89
5.2.4	Reverse EV-DV Channels	90
5.2.5	Key 1xEV-DV Features.....	91
5.3	1xEV-DV System Simulation.....	91
5.3.1	Simulation Architecture	92
5.3.1.1	Session Control	92
5.3.1.2	Mobility Manager	93
5.3.1.3	Bearer Traffic Model	94
5.3.1.4	Packet Control Function	95
5.3.1.5	Selection and Distribution Unit	96
5.3.1.6	BTS Cell and Sector.....	96
5.3.1.7	Mobile Station.....	97
5.3.2	System Level Performance	97
5.4	Conclusion	102

Chapter 6: Cellular Diagnosis Using HMMs	103
6.1 Modeling Frame Error Statistics	103
6.1.1 Validation of HMM	106
6.2 Frame Error Patterns in a Dropped Call	108
6.2.1 Dropped Call Prediction Using HMMs	110
6.2.2 Burst Error Patterns in Dropped Calls	113
6.3 Common Network Problems in CDMA	114
6.3.1 Hidden Markov Modeling of Coverage Related Dropped Calls	118
6.3.1.1 Optimization of Coverage Model	119
6.3.2 Identification of Coverage Problems Using HMM.....	120
6.3.3 Non-Coverage Related Problems.....	125
6.4 Conclusion	126
 Chapter 7: HMM-Based Position Location	 127
7.1 Traditional Position Location Techniques	127
7.1.1 Angle of Arrival Based Location Estimation.....	128
7.1.2 Time of Arrival Based Location Estimation.....	129
7.1.3 Time Difference of Arrival Based Location Estimation.....	130
7.1.4 Hybrid Techniques.....	131
7.2 HMM-Based Position Location	131
7.2.1 System Architecture.....	132
7.2.2 Feature Extraction.....	133
7.2.3 Vector Quantization.....	135
7.2.4 HMM-Based Classification	137
7.2.5 Experimental Results	139
7.2.6 Effect of Codebook Size	143
7.3 Conclusion	144
 Chapter 8: Conclusions.....	 145
8.1 Review of the Dissertation.....	145
8.2 Future Research Issues.....	147

Appendix A: List of Acronyms	150
Appendix B: Derivation of Log-likelihood Function	155
Appendix C: Hypothesis Test for Markov Chains.....	157
C.1 Hypothesis Testing for Markov Chain.....	160
C.2 Numerical Examples	162
Bibliography.....	165
Vita.....	173

List of Tables

Table 5.1: FTP Traffic model parameters.....	95
Table 5.2: System simulation parameters.	99
Table 6.1: Impact of number of states on coverage model, λ_{cov}	120
Table 7.1: Confusion matrix for HMM-based position location scheme for Rx 1.1 – Rx 1.4.....	140
Table 7.2: Confusion matrix for correlation-based position location for Rx 1.1 – Rx 1.4.	140
Table 7.3: Confusion matrix for HMM-based position location scheme for Rx 5.2 – Rx 5.4.....	142
Table 7.4: Impact of codebook size on classification accuracy.....	143
Table C.1: Summary of hypothesis testing.	158

List of Figures

Figure 2.1: State transition probability model for N -state Markov chain.....	11
Figure 2.2: An N -state urn and ball model for HMM	15
Figure 2.3: Structures of HMMs. (a) Ergodic. (b) Left-right model.....	18
Figure 2.4: Illustration of Gilbert model.....	20
Figure 2.5: Fritchman's state partitioned model with $N-1$ good states and 1 bad state..	21
Figure 2.6: Matrix precomputation for observation sequence.	26
Figure 3.1: Block diagram of the EM algorithm.....	32
Figure 3.2: Block diagram of HMM training.....	38
Figure 3.3: Illustration of forward-backward procedure. (a) Forward variables. (b) Backward variables. (c) Computation lattice.....	41
Figure 3.4: Illustration of the joint event that the system is in state s_i at time t and state s_j at time $t+1$	43
Figure 3.5: Convergence of the BWA is affected by the initial parameter set.	48
Figure 3.6: Block diagram of segmental K -means algorithm.	50
Figure 3.7: Genetic representation of HMM parameters.	52
Figure 3.8: Illustration of three point crossover.....	53
Figure 4.1: KLD rate measurement results for two Markov models given by (4.25).	66
Figure 4.2: Divergence rate measurement results for two HMM's given by (4.44).....	72
Figure 4.3: Divergence rate measurement results for two HMM's given by (4.45).....	73
Figure 5.1: Comparing peak data rates for different CDMA standards.....	86
Figure 5.2: cdma2000 1xEV-DV architecture.	87
Figure 5.3: Forward link power allocation in 1xEV-DV.....	89

Figure 5.4: Network Extensible Simulator (Nextsim) architecture.	93
Figure 5.5: A typical FTP session.....	95
Figure 5.6: System level simulation layout.....	98
Figure 5.7: System level performance of Nextsim. (a) FER (b) Transmit Gain (TxGa) adjust. (c) Sector power distribution. (d) Mobile transmit power (TxPo) distribution. (e) Distribution of active set pilots in PSMM.	101
Figure 6.1: Training of HMM by frame error statistics.	104
Figure 6.2: Log-likelihood function for HMM given by (6.2).	105
Figure 6.3: Comparison of original and HMM generated short term FER.....	106
Figure 6.4: Comparison of error free intervals for the original and HMM generated sequences.	107
Figure 6.5: Comparison of error burst lengths for the original and HMM generated sequences.	108
Figure 6.6: Timeline of a dropped call in CDMA.	108
Figure 6.7: Frame error pattern in a dropped call.	109
Figure 6.8: Training of HMMs for call drop prediction.	110
Figure 6.9: Evolution of divergence as link quality changes.....	111
Figure 6.10: CDF of divergence at the time of call drop and 4 sec. before drop.....	112
Figure 6.11: Hypothesis testing to predict dropped calls.....	113
Figure 6.12: Segmentation of error burst in a dropped call.	114
Figure 6.13: Hidden Markov modeling of error sequence in a coverage related dropped call.....	119
Figure 6.14: Identification of coverage problem by hypothesis testing for the first segment.	123
Figure 6.15: Probability density function of divergence between coverage related and non-coverage related dropped calls for the second segment.	124
Figure 7.1: AoA based position location.	128
Figure 7.2: Example of ToA based position location using three receivers.	129
Figure 7.3: Hyperbolic position location scheme.	130
Figure 7.4: Training of HMM for a given location.....	133
Figure 7.5: Testing of HMM-based location classifier.	134

Figure 7.6: An example power delay profile (PDP).	135
Figure 7.7: An example vector quantization in a two dimensional space. A codeword (circle) is obtained by satisfying (7.3) and (7.5), and represents all the training vectors in the corresponding partition.	137
Figure 7.8: Measurement Locations 1.1-1.4, 4th Floor, Durham Hall, Virginia Tech. The transmitter is located in Room 475, Receivers 1.1 and 1.2 are located in Room 471; Receiver 1.3 is in the conference room in the 476 computer lab, and Receiver 1.4 is located in the hallway adjacent to 475.	141
Figure 7.9: Measurement Locations 5.1 – 5.4, 4th Floor, Durham Hall, Virginia Tech. The Transmitter is located next to the receptionist’s desk in Room 432. Receiver location 5.1 is in room 433, Receiver locations 5.2 and 5.3 are in the atrium next to the elevators, and Receiver location 5.4 is in the hallway next to 432.	142
Figure C.1: Hypothesis testing and significance level.	159
Figure C.2: Hypothesis testing for 1st order Markov chain with $s = 3$	163
Figure C.3: Hypothesis testing for 2nd order Markov chain with $s = 4$	164

Chapter 1: Introduction

A hidden Markov model (HMM) is a bivariate parametric process, in which the underlying process is a finite-state Markov chain. The second process is a sequence of conditionally independent random variables given the Markov chain. At any time instant, the distribution of each random variable depends only on the Markov chain. The underlying Markov process is “hidden” and only the random variables are directly observable. HMMs were first introduced in full generality in 1966 by Baum and Petrie [5], who referred them as *probabilistic functions of Markov chains*. Since then HMMs have been used in disciplines as diverse as economics, biology, information theory, statistics, signal processing, pattern recognition, and wireless communications. Relevant examples of HMM applications include the modeling of wireless channels [8], the utterance of phoneme sequences in speech recognition [1], and bit error sequences in digital communication systems with memory [22]. In this chapter we will discuss the scope of this research effort, original contributions, and the overview of the dissertation.

1.1 Motivation

To stay competitive, cellular providers continually monitor the performance of their networks and use this information to determine where to improve or expand services. For example, they must decide how to load their network, when to split or sub-divide cells, modify antennas patterns, upgrade antenna arrays, split sectors, change handoff parameters, upgrade base-band processing algorithms, and re-allocate carrier frequencies. In addition to the myriad available propagation prediction tools and simulation models, cellular vendors bundle software with their Base Transceiver Station (BTS) products that

help a provider measure and manage the performance of a cell. The software package records both problem indicator statistics and diagnostic data. Problem indicators alert the operator to the existence of system problems such as hand-off failures, excessive frame-error rates (FER) and call-drop rates. Among the parameters measured at the BTS are: forward and reverse audio quality events (event triggered when the measured FER exceeds an acceptable level), percentage and number of blocked calls, percentage and number of dropped calls, percentage and the number of hand-off failures (both hard and soft). Each base-station also records diagnostic data that can help the operator track down problem sources. These diagnostics include the relative strength of all forward traffic channels, reverse traffic channels and pilot channels. These also include reverse noise rise, the number of handoffs that have occurred, and the number of calls terminated because of radio frequency (RF) loss. Perhaps the most important diagnostic is the time-series record of pilot strength measurement message (PSMM) reported by each mobile during a call. However, when a problem occurs, providers are saddled with the task of wading through a large amount of data in order to trouble-shoot their system. Part of the difficulty of this task is that for many complicated problems such as hand-off failure, clues about the cause of the failure are hidden deep within the statistics of the underlying dynamic physical phenomena like fading, shadowing, and interference. Information about these phenomena might be extracted from recorded time-series data. In addition to the numerous types of diagnostic time-series, frame error sequences, sequences of zeros and ones indicating a successful or unsuccessful frame transmission, respectively, are also of interest. Such sequences are estimated in most cellular standards with a cyclic redundancy check (CRC). A need becomes apparent for a statistical analysis tool that a provider can use to describe and troubleshoot the dynamic behavior of the system. Such a tool should be easy enough for any RF engineer to use, but should be flexible enough to describe a variety of phenomena. Hidden Markov Models (HMMs) [7] emerge as a convenient and tractable mathematical tool to describe the dynamic behavior of complicated random time-series.

Many time-series have intuitively appealing statistical features whose details are difficult to describe. For instance, consider an observed frame error sequence. It is well known that often times in a wireless channel bit and frame errors occur in bursts. These

burst errors may be caused by fading, shadowing, interference, noise, inadequate power control, inadequate packet scheduling (as in the case of 1xEV-DO or 1xEV-DV), or any combination of the above. For instance, errors are much more likely to occur when a received signal undergoes a deep fade than when the fading profile reaches its peak. The pseudo-periodic nature of slow fading processes (whose period is greater than a frame length) induces a pseudo-periodic nature in the frame-error process, manifesting itself in the form of error bursts. Now, by laboriously examining a given frame-error sequence, it is possible to recognize a bursty frame error process from an error process without bursts. However, describing the details of the burst-error process or defining the statistical features that distinguish one burst error process from another can be difficult without HMMs. HMMs have become a popular model of many naturally occurring phenomena because of their ability to capture a wide range of statistical features, and they provide a compact description of the phenomena of interest. While HMMs have been widely used as a simulation tool [7], [8], their use as a cellular network measurement and diagnostic tool has not been considered. In this dissertation we propose that HMMs be used as a method to infer signature statistics about the nature and sources of faults in a cellular network. Since HMMs can serve as a compact description of a random process, they may be used to compare and contrast the behavior of channels and error processes experienced at the BTS in a large network.

1.2 Contributions

The main contributions of this research work are as follows:

- Proposed a new distance measure between hidden Markov models
- Compared proposed distance measure against other methods
- Model optimization based on parametric analysis
- Synthesize error patterns in dropped calls using hidden Markov models
- Link quality analysis and prediction of dropped calls using HMMs
- Coverage problem identification using feature extraction from call statistics
- Position location using hidden Markov models

We now describe these contributions.

Proposed a new distance measure for hidden Markov models

Kullback-Leibler divergence (KLD) is often used as a distance measure to compare different HMMs in estimation and classification problems. Since there is no closed-form solution, the only way to compute the KLD is through expensive numerical computations. We proposed a new distance measure that approximates the KLD with minimal computational burden.

Compared proposed distance measure against other methods

The performance of the proposed distance measure has been evaluated by comparing against other published distance measures [57]-[59]. Unlike some of the distance measures [61], [62], the proposed distance measure is not limited to any specific topology or application.

Applied parametric analysis to optimize model structure

Parametric analysis of HMM has recently been proposed for pattern recognition applications [67]. Usually in communication applications, model optimization (i.e., number of states, etc.) is done heuristically. We extend the concept presented in [67] for our application.

Synthesize error patterns in dropped calls using hidden Markov models

Even though HMMs have been widely used to model bursty error patterns [22], [77], it has not been exactly applied to model dropped calls. In a dropped call error bursts are time-varying, which results in a non-stationary HMM. By partitioning the error bursts into smaller segments, we propose a new method to synthesize the whole error pattern in a dropped call.

Link quality analysis and prediction of dropped calls using hidden Markov models

HMMs have been proposed to analyze link quality in a cellular network. By having a model for a good link quality call, the network operator can use the proposed method to monitor the link quality over the life time of a call. This concept is then extended to predict dropped calls in terms of a hypothesis test.

Coverage problem identification using feature extraction from call statistics

Wading through the call statistics and analyzing the relevant call detail log (CDL) information, we found that the HMM has the potential to identify coverage problem. Due

to their signature statistics in the frame error sequences, coverage related dropped calls can be classified from other dropped calls.

Position location using hidden Markov models

Position location has become a problem of growing importance. In this dissertation, position location has been shown as a novel application of HMM based techniques. Unlike existing methods, the proposed technique does not depend on the line of sight reference and performs well for the non-line of sight and multipath indoor environments.

1.3 Disclosures, Papers and Reports

- **M. Mohammad**, J. Reed, and W. Tranter, "Position location using hidden Markov models," U.S. Patent Serial No. 60/812,634.
- **M. Mohammad**, "Prediction of dropped calls in a cellular network," Motorola Def. Pub., <http://ip.com/pubView/IPCOM000140987D>, Sept. 2006.
- **M. Mohammad**, "Identification of coverage related dropped calls," Motorola Def. Pub., <http://ip.com/pubView/IPCOM000139509D>, Aug. 2006.
- **M. Mohammad** and R. M. Buehrer, "The effects of ordering criteria in linear successive interference cancellation in CDMA systems," to be submitted to *Trans. on Wireless Communications*.
- **M. Mohammad** and W. Tranter, "Comparing distance measures for hidden Markov models," *Proc. of the Southeast Conf.*, pp. 256-260, Apr. 2006.
- **M. Mohammad** and W. Tranter, "Blind acquisition of short burst with per-survivor processing," to be published, *Trans. on Wireless Communications*.
- **M. Mohammad** and R. M. Buehrer, "On the impact of SNR estimation error on adaptive modulation," *IEEE Comm. Letts.*, vol. 9, pp. 490-492, Jun. 2005.
- **M. Mohammad**, "System level problems in CDMA and HMM," Tech. Report, CDMA System Division, Motorola, May 2005.
- **M. Mohammad** and W. Tranter, "A novel divergence measure for hidden Markov models," *Proc. of the Southeast Conf.*, pp. 240-243, Apr. 2005.

- **M. Mohammad** and W. Tranter, “Blind acquisition of short burst with per-survivor processing,” *Proc. of Southeast Conf.*, pp. 244-250, Apr. 2005.
- S. Al-Ghadhban, **M. Mohammad**, R. M. Buehrer, and B. Woerner, “Performance evaluation of decoding algorithms for multi-layered STBC-OFDM systems,” *Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 1208-1212, Nov. 2004.
- **M. Mohammad**, S. Al-Ghadhban, B. Woerner, and W. Tranter, “Spatial sequence estimation based decoding algorithm for V-BLAST,” *Proc. Vehicular Technology Conf.*, vol. 3, pp. 1875-1879, Sept. 2004.
- S. Al-Ghadhban, **M. Mohammad**, and B. Woerner, “Iterative spatial sequence estimator for multi-group space-time trellis coded systems,” *Proc. Vehicular Technology Conf.*, vol. 2, pp. 1353-1357, Sept. 2004.
- **M. Mohammad**, “System level performance evaluation of reverse channel quality indicator channel in 1xEV-DV,” Tech. Report, Network Advance Technology, Motorola, Aug. 2004.
- **M. Mohammad**, “Impact of TxPA and scheduler load averaging on system performance in 1xEV-DV,” Tech. Report, Network Advanced Technology, Motorola, Aug. 2004.
- **M. Mohammad**, “Hypothesis testing for Markov chain,” Tech. Report, CDMA System Division, Motorola, May 2004.
- **M. Mohammad**, S. Al-Ghadhban, B. Woerner, and W. Tranter, “Comparing decoding algorithms for multi-layered space-time block codes,” *Proc. of the Southeast Conf.*, pp. 147-152, Mar. 2004.
- **M. Mohammad**, “Approximating the distribution of a sum of squared correlated Gaussian random variables with arbitrary mean and variances,” Tech. Report, CDMA System Division, Motorola, Dec. 2003.

1.4 Overview of the Dissertation

In Chapter 2 the basic concepts related to Markov processes and hidden Markov models (HMMs) are presented. The transition probabilities, the state transition matrix and their properties are discussed in the framework of Markov processes. Different topologies of HMM's are also presented. One of the limitations of the Markov chain is its inability to model non-geometric distributions. Unlike Markov chains, semi hidden Markov models explicitly incorporate the state distributions which include gamma or Rayleigh distribution. Block equivalent representations of the HMM are discussed, which imposes a block diagonal structure on the HMM to facilitate faster parameter estimation. Validation methods for HMMs are also discussed.

Chapter 3 discusses the model structure and parameter estimation procedures. The parameter estimation procedures are complex and are usually formulated to optimally fit the empirical data. We present several methods for approximating the hidden Markov model (HMM) parameters from given experimental observations. Three different types of algorithms namely, the Baum-Welch algorithm (BWA), the segmental K -means algorithm, and the genetic algorithm are discussed. The BWA is presented as a special case of the EM algorithm with a detailed discussion of different aspects of the algorithm. Multiple observation sequence training for BWA is also presented. This is particularly useful for reliable parameter estimation when the sequence length is small.

In Chapter 4 distance measures between a pair of Markov and hidden Markov models are analyzed. Distance measure is often used to compare different HMMs and has widespread application in estimation and classification problems. The notion of Kullback-Liebler divergence (KLD) as a distance measure is presented. Unlike Markov models, there is no closed form solution to compute the KLD rate between HMMs. A novel heuristic divergence measure is proposed that closely approximates the KLD rate between HMMs. The proposed method is compared against other distance measures for HMMs. From the matrix theory perspective, the HMM parameters are analyzed to optimize the model structure.

Chapter 5 presents a brief overview of the evolution of cellular standards. The main focus of this chapter is cdma2000 1xEV-DV, a Code Division Multiple Access (CDMA)

based third generation standard. The key aspects of 1xEV-DV are highlighted. We also introduce Motorola's 1xEV-DV simulator, which is extensively used in this research. The system level performance of 1xEV-DV is also illustrated.

In Chapter 6 the application of HMM as a cellular diagnostic tool is analyzed. Even though HMMs have been widely used to simulate wireless channels [8], [22], it has not been applied to cellular diagnostic systems. The novel use of the HMM as a dropped call prediction tool is introduced in this chapter. A segmentation approach to synthesize the call statistics in a dropped call is presented. By exploiting the signature statistics in the frame error sequences, we show that the coverage related dropped calls can be identified from other dropped calls.

Chapter 7 presents a novel position location algorithm based on hidden Markov models. Most of the traditional position location techniques depend on the triangulation method that requires the line of sight (LoS) reference for acceptable accuracy. However, in many practical applications especially in indoor environments, the LoS reference is not present. As a result, the triangulation method fails in such scenarios. The proposed HMM based position location technique overcomes this problem as it exploits the location specific channel statistics and does not depend on the LoS reference.

Finally, Chapter 8 concludes with the future research directions.

Chapter 2: Fundamentals of Hidden Markov Models

In this chapter the basic concepts related to Markov processes and hidden Markov models (HMMs) are presented. The notions of state and state transitions are introduced first. Then the transition probabilities, the state transition matrix and their representations are discussed in the framework of Markov processes. Later, Markov processes are extended to hidden Markov models. Different types and structures of HMM's are also discussed. Unlike Markov chains, semi hidden Markov models explicitly incorporate the state distributions, which can be used to model non-geometric interval distributions. Long observation periods are required to accurately estimate the small valued parameters of the HMM. Though small, these values are critical for evaluating the system performance. However, arbitrary structure of the HMM can be very inefficient in this case. The block equivalent representation of the HMM is discussed, which imposes a block diagonal structure on the HMM to facilitate faster parameter estimation. Once the model parameters are estimated, the accuracy must be tested to validate the model. At the end of the chapter a few validation techniques are presented.

2.1 State and Transition

Often times a physical system is described in terms of a number of variables that can completely describe the system. For example, an electric gadget can be generally specified by the voltage, current and temperature it can tolerate. These variables are critical to the functionality of the system and are called *state* variables. When all values of the state variables are known, the system is said to be fully defined. In this case, the

system can be described as a function of its states at any instant. If the process is dynamic the system moves from one state to another, resulting a sequence of *transitions* of states over the lifetime of the process.

Let us consider an example in which there are N urns and a ball in a room. Let us assume that the ball can only be placed in one of the urns and not on the floor. The ball moves from one urn to another according to some random process. If q_t denotes the urn where the ball is placed after the t -th move, then the process is said to be in *state* q_t at time index t . Since the ball can randomly move to any one of the N urns, q_t can take on any values in the range $[s_1, s_2, \dots, s_N]$ at any time index $t = 1, 2, 3, \dots$. The trajectory of the process $\mathbf{Q} = q_1, q_2, q_3, \dots, q_n$ represents the *state sequence*. Since the state sequence is randomly selected, the statistical behavior of the process can be defined by the conditional probability of any urn being the next state given all the previous urns. Mathematically this is given by

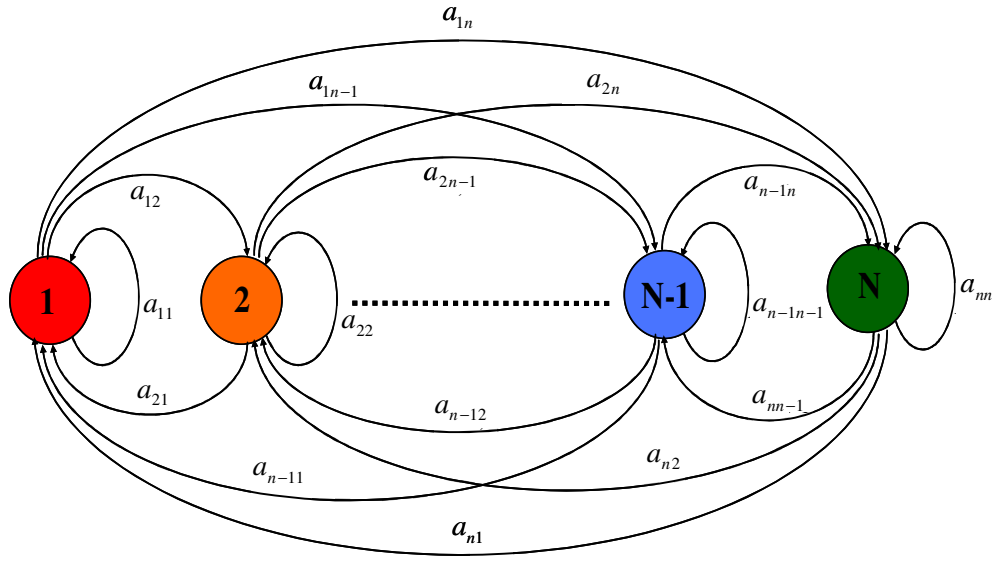
$$P(q_{t+1} = s_j | q_t = s_i, q_{t-1} = s_k, \dots, q_0 = s_m) \quad (2.1)$$

Since the probability in (2.1) depends on all the previous urns, it becomes difficult to analyze the process analytically. However, by applying certain assumptions (2.1) can be simplified significantly, as we will find in the next section.

2.2 Markov Process

Markov processes are an important class of stochastic processes [1]-[3]. This class of processes has some special properties that make them analytically tractable. With states and transitions formally defined, let us consider a system with N distinct states, $\mathbf{S} = [s_1, s_2, \dots, s_N]$, as illustrated in Figure 2.1. At regularly spaced discrete time instants, $t = 1, 2, 3, \dots$ ¹, the system undergoes a sequence of states, $\mathbf{Q} = q_1, q_2, q_3, \dots, q_n$, depending on the probabilities associated with the corresponding states. Even though a full probabilistic description usually requires specification of all the previous states, using the

¹ Only discrete Markov processes are considered here.

Figure 2.1: State transition probability model for N -state Markov chain.

Markovian assumption, this can be limited to last k states only. A Markov process is governed by the *Markovian assumption*, which states that the future behavior of the process depends *only* on the last k states. Since all the previous states before the last k states are not significant in determining the future state of the system, this is also referred to as the k -th order Markov chain. For the special case of $k = 1$, the chain reduces to the *first-order* Markov chain where the probabilistic description is truncated to the future and present state. In equation form we have

$$P(q_{t+1} = s_j | q_t = s_i, q_{t-1} = s_k, \dots, q_1 = s_m) = P(q_{t+1} = s_j | q_t = s_i), \quad \forall t \quad (2.2)$$

The assumption in (2.2) greatly simplifies the behavior of the process and makes the problem mathematically tractable. The Markovian assumption is a very strong assumption and it is difficult to prove that any physical system is absolutely Markovian or non-Markovian. Yet the Markov process is an extremely useful model for wide classes of systems as it can describe complex models with manageable complexity. Also, any k -th order Markov chain can be represented by a first-order Markov chain with higher number of states. Typically the number of states in the first order chain grows exponentially with memory k . To be specific, an N -state k -th order Markov chain requires N^k states in order to make the first-order representation valid.

In this thesis, we only consider *homogeneous* chains in which the probabilities in (2.2) do not depend on the time index t , thereby leading to a set of transition probabilities given by

$$a_{ij} = P(q_{t+1} = s_j | q_t = s_i), \quad 1 \leq i, j \leq N \quad (2.3)$$

The transition probabilities must satisfy the stochastic constraints

$$0 \leq a_{ij} \leq 1, \quad 1 \leq i, j \leq N \quad (2.4)$$

and

$$\sum_{j=1}^N a_{ij} = 1, \quad i = 1, 2, \dots, N \quad (2.5)$$

Any joint distribution of a first-order N -state homogenous Markov chain is compactly represented by an $N \times N$ transition probability matrix, $\mathbf{A}$, given by

$$\mathbf{A}_{N \times N} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix} \quad (2.6)$$

Since the elements of $\mathbf{A}$ must satisfy the constraints imposed by (2.4) and (2.5), it is called a *stochastic matrix* [4]. Also due to the constraint in (2.5), only $N \times (N-1)$ parameters are needed to fully define the N -state Markov process.

The initial probability distribution of the system is given by

$$\boldsymbol{\pi} = [\pi_i], \quad 1 \leq i \leq N \quad (2.7)$$

where

$$\pi_i = P(q_1 = s_i), \quad 0 \leq \pi_i \leq 1, \quad \sum_{i=1}^N \pi_i = 1 \quad (2.8)$$

The chain starts from state q_1 with the probability π_{q_1} . It then moves to state q_2 with the probability $a_{q_1 q_2}$. Next it transfers to state q_3 with the probability $a_{q_2 q_3}$ and so on. The joint probability of the sequence of states $\mathbf{Q} = q_1, q_2, q_3, \dots, q_n$ can be expressed as

$$\begin{aligned} P(q_1 = s_m, q_2 = s_k, \dots, q_{t-1} = s_i, q_t = s_j) &= \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \dots a_{q_{t-1} q_t} \\ &= \pi_{q_1} \prod_{v=2}^t a_{q_{v-1} q_v} \end{aligned} \quad (2.9)$$

Since the state transition probability matrix, $\mathbf{A}$, is a stochastic matrix we have

$$\mathbf{A}\mathbf{1} = \mathbf{1} \quad (2.10)$$

where $\mathbf{1}$ is a $N \times 1$ column vector of all ones. Note that $\mathbf{1}$ is an eigenvector of $\mathbf{A}$ corresponding to the eigenvalue $\sigma = 1$. This vector is also called a *right invariant vector* as the multiplication in (2.10) does not change $\mathbf{1}$.

If π_t denotes the state probability distribution at time t , by definition of the state transition matrix, the state probability distribution at time $t+1$ is given by

$$\pi_{t+1} = \pi_t \mathbf{A} \quad (2.11)$$

Similarly, the state distribution at time $t+2$ is

$$\pi_{t+2} = \pi_{t+1} \mathbf{A} = (\pi_t \mathbf{A}) \mathbf{A} = \pi_t \mathbf{A}^2 \quad (2.12)$$

And in general,

$$\pi_{t+n} = \pi_t \mathbf{A}^n \quad (2.13)$$

where $\mathbf{A}^n$ denotes the n -step transition matrix.

The Markov chain is called *stationary* if there exists a unique vector $\mathbf{p} = [p_1, p_2, \dots, p_N]$ such that [4]

$$\lim_{n \rightarrow \infty} \pi \mathbf{A}^n = \mathbf{p} \quad (2.14)$$

$$\mathbf{p} \mathbf{A} = \mathbf{p}$$

where π is the initial state distribution given by (2.7) and (2.8). Note that $\mathbf{p}$ is the left eigenvector of $\mathbf{A}$ corresponding to the eigenvalue $\sigma = 1$. The vector $\mathbf{p}$ is also called the *steady-state* distribution or stationary vector of $\mathbf{A}$. It can be easily shown that for sufficiently large n , the rows of $\mathbf{A}^n$ also equal $\mathbf{p}$.

2.2.1 Example of a Simple Markov Model

Let us consider a simple three-state Markov model for weather. We assume a day is observed as one of the following: rainy, cloudy or sunny, where they are represented by state 1, state 2 and state 3, respectively, in the model. On any given day the weather is characterized by the transition probability matrix given by

$$\mathbf{A} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix} \quad (2.15)$$

Given that the weather on the first day is sunny, the probability that the weather for the next seven days will be “sun-sun-rain-rain-sun-cloudy-sun” is given by

$$\begin{aligned} P(s_3, s_3, s_3, s_1, s_1, s_3, s_2, s_3 | \mathbf{A}) &= \pi_3 a_{33} a_{33} a_{31} a_{11} a_{13} a_{32} a_{23} \\ &= 1 \times 0.8 \times 0.8 \times 0.1 \times 0.4 \times 0.3 \times 0.1 \times 0.2 \\ &= 1.5236 \times 10^{-4} \end{aligned} \quad (2.16)$$

Given the model is in a known state, i , the probability that it stays in that state for exactly d consecutive days is given by

$$P_i(d) = a_{ii}^{d-1} (1 - a_{ii}) \quad (2.17)$$

where $P_i(d)$ is the discrete probability density function of duration d in state i . Note that the Markov model imposes a geometric distribution over the state duration. This is one of the limitations of the Markov model when modeling non-geometric interval distributions. The expected number of consecutive days in state i is given by

$$\bar{d}_i = \frac{1}{1 - a_{ii}} \quad (2.18)$$

So the expected number of sunny days given the model $\mathbf{A}$ is 5.

By solving (2.14), the steady state distribution or stationary vector of $\mathbf{A}$ is given by

$$\mathbf{p} = [0.18 \quad 0.27 \quad 0.55] \quad (2.19)$$

2.3 Hidden Markov Model

In the Markov model discussed so far, each state corresponds to an observable physical event. To be specific, in the ball and urn example in Section 2.1, the states are defined by the urns occupied by the ball, which are directly observable. In other words, the observation symbols are deterministic functions of the state sequences. However, in many practical applications this model is too restrictive and the notion of the Markov model is extended to include the case in which the observation is a probabilistic function of the state. The resulting model is called a *hidden* Markov model (HMM) or probabilistic function of Markov chains [5]-[9]. Unlike Markov processes, there is no one-to-one mapping between the observations and the state sequences in hidden Markov models. The HMM is a *doubly stochastic* process with an underlying stochastic process (state sequence) that is not directly observable, but can only be observed through another

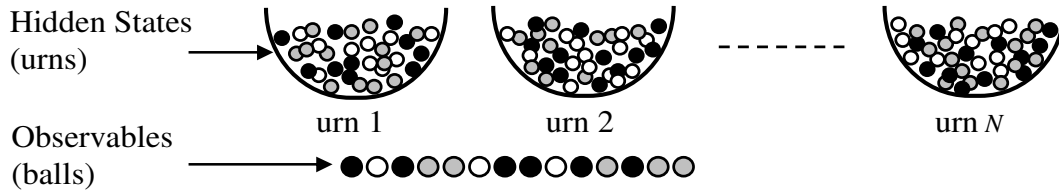


Figure 2.2: An N -state urn and ball model for HMM.

stochastic process that results in the observation symbol sequence. So upon observation of a symbol, the state that produces the symbol can not be identified. Since the states are *hidden* the model is called hidden Markov model.

Let us consider a simple example, illustrated in Figure 2.2, to explain the concept of hidden Markov models (HMMs). We assume there are N urns in a room. Each urn contains a large number of colored balls. In total, there are M distinct colors of the balls. Let us assume a stochastic process where first an urn is chosen at random. Next, a ball is selected at random from this urn and its color is recorded as the observation. The ball is then put back in the urn and a new urn is chosen according to some random process and the random ball selection process is repeated. As is obvious from this example, the urn and color of the ball are analogous to the state and observation symbol of an HMM. The choice of urns is dictated by the state transition matrix of the HMM. Note that by looking at the color of the ball there is no way of knowing the urn from which it came since each urn contains balls of every color. In other words, the states are hidden.

2.3.1 HMM Parameters

A hidden Markov model is a collection of finite states connected by transitions. Each state is characterized by two sets of probabilities: a state transition probability based on the first order Markov chain and a discrete² output probability distribution that defines the conditional probability of emitting an output symbol, chosen from a finite alphabet, given the state of the model. The elements of an HMM can be formally defined as follows:

² Only discrete output probability distributions are considered in this thesis as they are computationally more efficient. Also, they don't require any assumption on the underlying distribution of the observations.

1) Number of states

The number of states in the model is denoted by N . Although the states are hidden, in most of the applications there is a physical meaning related to the states. The individual states are defined as $\mathbf{S} = [s_1, s_2, \dots, s_N]$ and the state at time index t is given by q_t , where $q_t \in \mathbf{S}$.

2) Number of observation symbols

The discrete alphabet size or the number of distinct observation symbols per state is given by M . The observation symbols correspond to the physical output of the system. The individual symbols are defined as $\mathbf{V} = [v_1, v_2, \dots, v_M]$.

3) State transition probability

The state transition probability matrix is given by $\mathbf{A} = \{a_{ij}\}, 1 \leq i, j \leq N$, where a_{ij} , as before, is defined as the elements of a first order Markov chain

$$\begin{aligned} a_{ij} &= P(q_{t+1} = s_j | q_t = s_i) \\ 0 \leq a_{ij} &\leq 1, \quad \sum_{j=1}^N a_{ij} = 1 \end{aligned} \quad (2.20)$$

4) Observation symbol probability

The observation symbol probability matrix is given by $\mathbf{B} = \{b_j(l)\}, 1 \leq j \leq N, 1 \leq l \leq M$, where

$$\begin{aligned} b_j(l) &= P(v_l \text{ at } t | q_t = s_j) \\ 0 \leq b_j(l) &\leq 1, \quad \sum_{l=1}^M b_j(l) = 1 \end{aligned} \quad (2.21)$$

Note that $\mathbf{B}$ is also a stochastic matrix.

5) Initial state distribution

The initial state distribution is given by $\boldsymbol{\pi} = \{\pi_i\}, 1 \leq i \leq N$, where

$$\begin{aligned} \pi_i &= P(q_1 = s_i) \\ 0 \leq \pi_i &\leq 1, \quad \sum_{i=1}^N \pi_i = 1 \end{aligned} \quad (2.22)$$

For convenience, an HMM is often denoted by the compact triplet

$$\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}) \quad (2.23)$$

2.3.2 Generation of Output Sequences

Given the model, $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, the HMM can be used as an observation sequence generative model. Let T be the length of the observation sequence to be generated and $o_t \in \mathbf{V}$ be the observation symbol at time instant t . The observation sequence can be generated using the following steps:

- 1) Choose an initial state $q_1 = s_i$ by generating a uniformly distributed random number in the range $(0,1]$ and comparing it against the initial state distribution $\boldsymbol{\pi}$.
- 2) The time index is set to $t = 1$.
- 3) Choose $o_t = v_l$ by generating a uniformly distributed random number in the range $(0,1]$ and comparing against symbol probability distribution in state s_i , $b_i(l)$.
- 4) The next state $q_{t+1} = s_j$ is determined by generating another uniformly distributed random number in the range $(0,1]$ and comparing it against the state transition probability distribution for state s_i , a_{ij} .
- 5) Set $t = t + 1$.
- 6) If $t > T$ terminate the process, otherwise return to step 3.

2.3.3 Structures of HMMs

Although there are many conceivable variations and combinations possible, in this section we will discuss two of the most widely used structures of HMMs, namely the ergodic structure and the left-right structure. An ergodic or fully connected HMM is one in which every state of the model can be reached from every other state of the model in a finite number of steps. As shown in Figure 2.3(a), this topology of model has the property that every element of the transition probability matrix, $a_{ij}, 1 \leq i, j \leq N$, is non-zero.

In certain applications, i.e., speech, the signal properties change over time. In order to accurately model this signal pattern over time, the elements of the transition matrix are constrained such that as time increases the state index increases or stays the same. This type of structure is called a Bakis model [10], [11] or a *left-right* model as the state

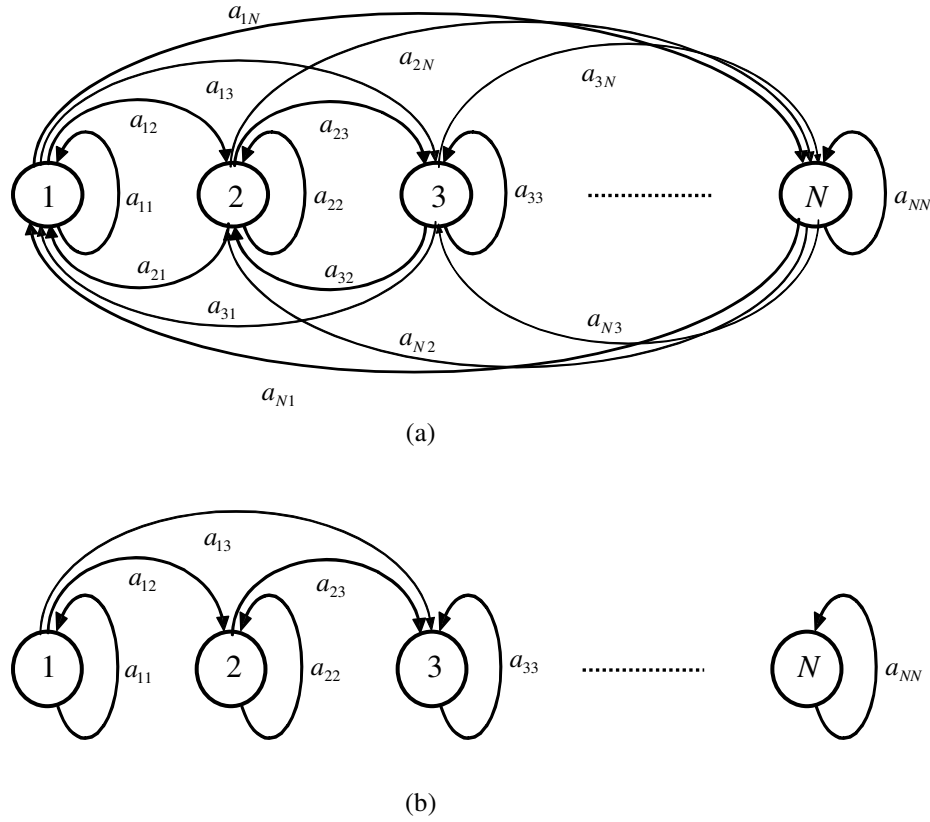


Figure 2.3: Structures of HMMs. (a) Ergodic. (b) Left-right model.

sequence over time moves from left-to-right. In a left-right model, the elements of the transition matrix must be such that no transitions are allowed to states whose indices are lower than the current state. Specifically

$$a_{ij} = 0, \quad j < i \quad (2.24)$$

Also, the initial state distribution has the property

$$\pi_i = \begin{cases} 1, & i = 1 \\ 0, & i \neq 1 \end{cases} \quad (2.25)$$

Based on the application, additional constraints can be placed on the state transition probabilities of a left-right model. In order to make sure that large changes in the state indices do not occur, the following condition may be used

$$a_{ij} = 0, \quad j > i + \delta \quad (2.26)$$

As shown in Figure 2.3(b), $\delta = 2$ and no jumps of more than 2 states are allowed in this particular left-right model. The corresponding state transition matrix has the following form

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & \cdots & 0 \\ 0 & a_{22} & a_{23} & a_{24} & \cdots & 0 \\ 0 & 0 & a_{33} & a_{34} & \cdots & 0 \\ 0 & 0 & 0 & a_{44} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & a_{NN} \end{bmatrix} \quad (2.27)$$

It is obvious from (2.27) that the transition probabilities for the last state are specified as

$$\begin{aligned} a_{NN} &= 1 \\ a_{Ni} &= 0, \quad i < N \end{aligned} \quad (2.28)$$

Note that even though certain applications impose specific constraints on the HMM structure, it does not affect the model parameter estimation process. As we will see in Chapter 3, the parameters that are set to zero initially, remain at zero throughout the estimation procedure.

2.3.4 Gilbert Model

The characterization of wireless channels using discrete time, finite state HMMs was initiated by Gilbert [12]. The Gilbert model is a two state model having a good state, G , and a bad state, B , with the state transition probability matrix

$$\mathbf{A} = \begin{bmatrix} Q & P \\ p & q \end{bmatrix} \quad (2.29)$$

The model state transition diagram is shown in Figure 2.4. There are no errors in state G , while errors occur in state B with the probability h . The observation symbol probability matrix is therefore given by

$$\mathbf{B} = \begin{bmatrix} 1 & 1-h \\ 0 & h \end{bmatrix} \quad (2.30)$$

Gilbert used this model to calculate the capacity of a channel with burst errors. Since the probability of changing states is made very small, the model can describe the bursty nature of errors in the wireless channel. The generated error process is determined by the transition probabilities of the Markov chain, which are determined from some statistics of the original error sequence of the channel. However, one of the primary limitations of

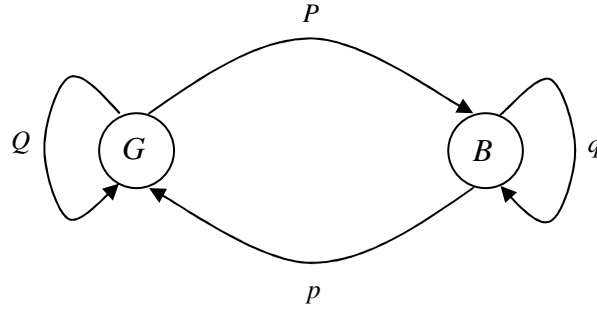


Figure 2.4: Illustration of Gilbert model.

Gilbert's model is it can not describe more complex error processes with its simple geometric distribution assumption for the run lengths of both states.

Elliot [13] proposed a modification to the Gilbert model by replacing the error-free good state by another state with a very small error probability so that the $\mathbf{B}$ matrix in (2.30) changes to

$$\mathbf{B} = \begin{bmatrix} 1-g & 1-h \\ g & h \end{bmatrix} \quad (2.31)$$

where g is the small error probability for the good state, G . This allows the resulting error process to have a small background error probability even when the channel is in good state. Even though this model improves on the Gilbert model, it still falls short of accurately characterizing real channel error profiles.

2.3.5 Fritchman's State Partitioned Model

In order to overcome the shortcoming of the Gilbert and Gilbert-Elliot models, many more complex models have been proposed [8], [14]-[16]. The most popular generalizations of the Gilbert model are single error state models, in which there is more than one good state and one bad state. A particular case of the single error state model, in which transitions are not allowed between any two good states, is called Fritchman's state partitioned model [16]. The state transition diagram for this model is illustrated in Figure 2.5.

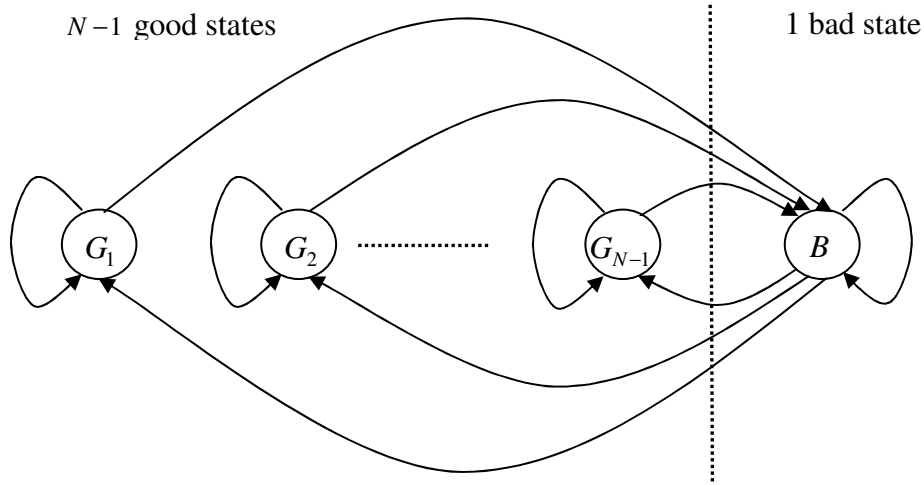


Figure 2.5: Fritchman's state partitioned model with $N-1$ good states and 1 bad state.

Fritchman proposed a Markov chain having N states. These N states are divided into two groups. The first group consists of k good states, while the second group has $N-k$ bad states. The good state represents the error free transmission while errors occur in bad states with probability one. This eliminates the need to estimate the elements of $\mathbf{B}$, which consists of only ones and zeros. The state transition matrix for Fritchman model with three good states and one bad state is given by

$$\mathbf{A} = \begin{bmatrix} a_{11} & 0 & 0 & a_{14} \\ 0 & a_{22} & 0 & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad (2.32)$$

Note that for the good states, only those elements are non-zero that represent self-transition, where 4-th state is the error or bad state. The simplified observation symbol probability matrix $\mathbf{B}$ is given by

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.33)$$

Usually for the Fritchman model, $\mathbf{A}$ can be partitioned as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{GG} & \mathbf{A}_{GB} \\ \mathbf{A}_{BG} & \mathbf{A}_{BB} \end{bmatrix} \quad (2.34)$$

where the submatrices represent the transition probabilities between various good and bad states. For the Fritchman model, burst error distribution can be analytically derived in

terms of the model parameters. The analytical results can then be used to estimate the model parameters from the empirical error burst distribution.

Fritchman showed that the error-free interval distribution between errors can be expressed by the sum of k exponentials, while the distribution of error intervals between good frames is given by the sum of $N - k$ exponentials. Let the notation $(0^m | 1)$ denote the event of observing m or more consecutive error free transmissions following an error, and $(1^m | 0)$ represent the event of observing m or more consecutive errors following a good period. The probability of occurrence of these two event are given by

$$P(0^m | 1) = \sum_{i=1}^k f_i \sigma_i^{m-1} \quad (2.35)$$

and

$$P(1^m | 0) = \sum_{i=k+1}^N f_i \sigma_i^{m-1} \quad (2.36)$$

where σ_i , $i = 1, 2, \dots, k$, and σ_i , $i = k+1, k+2, \dots, N$, are the eigenvalues of $\mathbf{A}_{GG}$ and $\mathbf{A}_{BB}$, respectively, and the corresponding values of f_i are functions of a_{ii} .

The Fritchman model can be interpreted as equivalent to a Markov process with a state transition probability matrix given by

$$\tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_{GG} & \mathbf{A}_{GB} \\ \mathbf{A}_{BG} & \mathbf{A}_{BB} \end{bmatrix} \quad (2.37)$$

where $\mathbf{A}_{GG}$ and $\mathbf{A}_{BB}$ are diagonal matrices given by

$$\mathbf{A}_{GG} = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_k \end{bmatrix} \quad (2.38)$$

and

$$\mathbf{A}_{BB} = \begin{bmatrix} \sigma_{k+1} & 0 & \cdots & 0 \\ 0 & \sigma_{k+2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_N \end{bmatrix} \quad (2.39)$$

Note that in the equivalent model, there are no transitions within the set of k good states and no transitions within the set of $N - k$ bad states.

Several researchers have investigated the behavior of a simplified version of the Fritchman model (SFM) that has only bad state, i.e., $N - k = 1$. It has also been shown that the Fritchman model is unique only when there is only one bad state [17], since usually the distributions $P(0^m | 1)$ and $P(1^m | 0)$ do not specify the statistical dependence of the error free runs and the error bursts. For a single error state model, $P(0^m | 1)$ reduces to

$$P(0^m | 1) = \sum_{\mu=1}^{N-1} \frac{a_{N\mu} (a_{\mu\mu})^m}{a_{\mu\mu}}, \quad m \geq 1 \quad (2.40)$$

With the simplified error-free distribution given in (2.40), it is relatively easy to estimate the parameters of the Fritchman model. First, $P(0^m | 1)$ is calculated from the empirical or simulated error sequences. Then using curve fitting techniques, $P(0^m | 1)$ can be expressed in terms of weighted summation of $N - 1$ exponentials as shown in (2.35). The transition probabilities can then be obtained by equating similar terms between the expansion and (2.40).

Nearly all applications of generative Markov models in modeling real communication channels reported in the literature are based on the SFM, mainly due to its relative simplicity [18]-[20]. Even though Fritchman's model is applicable to discrete channels with simple burst error distributions, it may not be adequate to characterize very complex burst error patterns that require more than one error state in the model. Also, this model assumes that the intervals between consecutive errors are independent and identically distributed, which is not satisfied by most empirical data. To model the interval dependence, it is necessary to use more than one bad state. Furthermore, the curve fitting technique involves quite a bit of trial and error.

2.4 Semi-Hidden Markov Processes

Semi-hidden Markov processes are a special class of hidden Markov processes. In a semi-hidden Markov process, successive state occupancies are described by the transition probabilities of a Markov chain, while the state duration is determined by a random

number (integer-valued) that depends only on the current and future state [8], [17], [21]. At transition times, there is little difference between a semi-hidden Markov process and a Markov process. A semi-hidden Markov process is also called a *Markov renewal process*. Due to the structure of the transition matrix, semi-hidden Markov models (SHMM) significantly reduce computational complexity.

Let us assume there are N states in an SHMM, where the process is currently in state i . Based on the state transition probability a_{ij} , it determines the next state j . However, after j has been selected and before making the transition to state j , the process holds for a time τ_{ij} in state i . The parameter τ_{ij} is called the *holding time*, which is a positive, integer-valued random variable. Usually the holding time for a given current and future state is expressed in terms of the holding time mass function for the corresponding transition. Thus the probability distribution of τ_{ij} is given by

$$P(\tau_{ij} = m) = h_{ij}(m), \quad 1 \leq i, j \leq N, \quad m = 1, 2, \dots \quad (2.41)$$

All holding time mass functions, h_{ij} , are compactly expressed by the state duration matrix,

$\mathbf{H}$, where $\mathbf{H} = \{h_{ij}\}_{N \times N}$, $1 \leq i, j \leq N$.

If the state holding time distribution depends only on the current state

$$h_{ij}(m) = h_i(m), \quad 1 \leq i, j \leq N \quad (2.42)$$

then the process is called *autonomous*.

An SHMM can therefore be described by the initial state distribution, $\boldsymbol{\pi}$, the transition probability matrix, $\mathbf{A}$, the state duration distribution matrix, $\mathbf{H}$, and the output symbol probability matrix, $\mathbf{B}$. By changing the matrix $\mathbf{H}$, it is possible to model a wide variety of error sources using SHMM. For example, non-geometric distributions, i.e., normal, gamma, etc., can be used to explicitly define the state holding times. This overcomes the limitation of Markov models, where the state duration is limited by geometric distributions. Note that if the state holding times are made exactly one time unit in length the SHMM reduces to the simple HMM. Specifically, the holding time is given by

$$h_{ij}(m) = \delta(m-1), \quad 1 \leq i, j \leq N, \quad m = 0, 1, 2, \dots \quad (2.43)$$

where δ denotes the delta function.

2.5 Block Equivalent Hidden Markov Models

Often in modeling channel error statistics, long observation periods are required for accurate estimation of the HMM parameters. The transition matrix elements have large dynamic range and the reliable estimates of the small probability values are critical for evaluating the system performance. However, computation increases as long observation sequences are used for parameter estimation. Using a block matrix version, Turin proposed a method that reduces the overall computational burden. [22].

In this method, the N states of the system can be partitioned into to subsets N_0 and N_1 , corresponding to good and bad states, respectively. The transition probability matrix has the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{00} & \mathbf{A}_{01} \\ \mathbf{A}_{10} & \mathbf{A}_{11} \end{bmatrix} \quad (2.44)$$

By splitting the states this way, the errors can be made deterministic rather than a probabilistic function of states. Note that the computational complexity can be reduced by precomputing and reusing the powers of submatrices involved in the computation. For example, using *fast exponentiation* [25], $\mathbf{A}^8$ can be calculated using only 3 matrix multiplications³, i.e., $\mathbf{A}^2 = \mathbf{A}\mathbf{A}$, $\mathbf{A}^4 = \mathbf{A}^2\mathbf{A}^2$ and $\mathbf{A}^8 = \mathbf{A}^4\mathbf{A}^4$. Figure 2.6 illustrates the matrix computation for a specific sequence.

Another modification, proposed in [23], is based on the fact that for a general HMM there is a statistically equivalent Fritchman like model with k good state and $N - k$ bad states. The “block diagonal” transition matrix for this method is given by

$$\mathbf{A} = \begin{bmatrix} \mathbf{\Lambda}_{00} & \mathbf{A}_{01} \\ \mathbf{A}_{10} & \mathbf{\Lambda}_{11} \end{bmatrix} \quad (2.45)$$

where $\mathbf{\Lambda}_{00}$ and $\mathbf{\Lambda}_{11}$ are diagonal matrices.

With this model, the channel remains in the same state during an error burst and changes state only at the end of a burst. In an HMM with arbitrary structure, the parameters are updated at the end of every observation symbol, while in this modification they are updated only at the beginning of each error burst. Note that the computations

³ If the matrix exponent is not a power of two, faster computation can still be done using recursive methods. For details the reader is referred to [25].

during long interval distributions now involve raising the power of diagonal matrix rather than raising the power of an arbitrary matrix. Also, matrix multiplications occur only at the transition from one burst to another. This makes the overall computations very efficient.

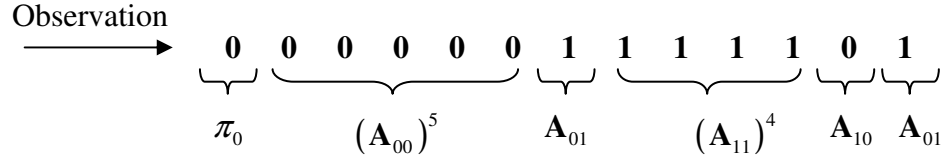


Figure 2.6: Matrix precomputation for observation sequence.

2.6 Validation of Hidden Markov Models

Once we have a particular HMM, its accuracy must be tested to validate the model. Depending on the applications, the validation of the HMM may involve different diverse techniques. In this section we only concentrate on HMMs that are used to model wireless communication channels.

The HMM replaces the waveform channel with a *discrete* channel model that probabilistically describes the error generation mechanism. From the error sequences, either obtained from the real channel or from computer simulation of the overall communication link, the HMM can be developed that is capable of generating error sequences with that same statistical distributions. There is a high level of abstraction involved in emulating the wireless channel with HMM, which may imply a loss of accuracy. The model therefore must be validated before it is used as a representation of the actual system.

The validation of HMMs can be done by computing the following statistics:

- 1) Scalar statistics i.e., bit error rate (BER) or frame error rate (FER)
- 2) Interval distribution statistics

For example, let us assume that the wireless channel is characterized by the following HMM for a given set of physical parameters, i.e., SNR, velocity, channel etc.

$$\begin{aligned}
\mathbf{A} &= \begin{bmatrix} 0.9159 & 0.0352 & 0.0489 \\ 0.1519 & 0.7434 & 0.1047 \\ 0.3059 & 0.2654 & 0.4287 \end{bmatrix} \\
\mathbf{B} &= \begin{bmatrix} 0.9827 & 0.9400 & 0.3640 \\ 0.0173 & 0.0600 & 0.6360 \end{bmatrix} \\
\boldsymbol{\pi} &= [1 \ 0 \ 0]
\end{aligned} \tag{2.46}$$

The first (second) row of $\mathbf{B}$ defines the condition probabilities for the observation of a good (bad) frame. Using (2.14), the steady state distribution of the transition matrix $\mathbf{A}$ is given by

$$\mathbf{p} = [0.7066 \ 0.1968 \ 0.0966] \tag{2.47}$$

The probability of frame error for this HMM can be calculated as

$$\begin{aligned}
P_{FE} &= \sum_{i=1}^N p_i \mathbf{B}(2, i) \\
&= 0.09
\end{aligned} \tag{2.48}$$

Assuming that the model is accurate, the FER predicted by the model should be very close to the actual FER of the real system.

Since in the real world error sequences are in general complex with error bursts and error free intervals having certain statistical characteristics, only first order statistics i.e., FER can not reliably test the model. In fact two frame error sequences can have the same FER with completely different error distributions. Thus, the error distributions also need to be investigated for a comprehensive validation. Based on the model in (2.46), the output sequences are generated as discussed in Section 2.3.2. The following statistics are computed from the generated sequences [24]:

- a) Distribution of error free intervals
- b) Distribution of error burst lengths

Similar quantities are calculated from the frame error sequences and compared against the model generated ones. If the model closely approximates these distributions then the model is assumed to be sufficiently accurate to model the wireless channel.

2.7 Conclusion

In this chapter the basic theory of Markov chains and the hidden Markov model was discussed. Error profiles in real world communication systems have complex statistics and the Markov model can not adequately emulate these systems due to the underlying simple geometric distributions. Markov models are therefore extended to more complex hidden Markov models. The HMM is a convenient and analytically tractable tool, capable of describing the statistical behavior of complicated random time-series. Various structures of HMMs were illustrated that trade-off complexity and performance. Based on the statistics in the observed sequence, a particular structure may be imposed on the model for efficient modeling of the underlying system. With the fundamental concepts and definitions introduced in this chapter, the more complex HMM parameter estimation algorithms will be discussed in the next chapter.

Chapter 3: Hidden Markov Model Parameter Estimation

In order to simulate the hidden Markov model (HMM), the model structure and parameter values must be estimated from simulated data or measured observations. The model parameter estimation procedures are complex and are usually formulated to optimally fit the empirical data. In this chapter, several methods for approximating the hidden Markov model (HMM) parameters from given experimental observations are described. We start with the iterative expectation maximization (EM) algorithm which is the generic version of the statistical EM algorithm. In order to fit the HMM to a stochastic process, three different types of algorithms namely, the Baum-Welch algorithm (BWA), the segmental K -means algorithm and the genetic algorithm are discussed. The BWA is presented as a special case of the EM algorithm with a detailed discussion of different aspects of the algorithm. The phase type distribution, the simplest case of the BWA, is also explained as a modeling tool for long interval distributions. When only a small number of observations for a given state are allowed, multiple observation sequence training is required for reliable parameter estimation. Different types of algorithms based on multiple observation sequences are introduced.

3.1 Minimization Algorithm

In order to approximate a stochastic process with a given process, we must first define a quality metric that measures the closeness between the two processes. Let us assume $F(x)$ to be the cumulative probability distribution of the discrete random variable x for

the original stochastic process and $G(x, \tau)$ be the corresponding cumulative distribution for its model, where τ is a model parameter. The approximation quality measure $L(\tau)$ is given by [25]

$$L(\tau) = \Delta(F(x), G(x, \tau)) \quad (3.1)$$

Usually the closeness criterion is given in terms of *divergence*¹ [26]. The optimal fit is achieved by minimizing $L(\tau)$ with respect to τ such that

$$\hat{\tau} = \arg \min_{\tau} L(\tau), \quad \tau \in \Omega \quad (3.2)$$

where Ω denotes the parameter space. In most cases it is almost impossible to solve this problem analytically. Iteratively, a sequence of parameters $\tau_0, \tau_1, \tau_2, \dots$ is obtained that minimizes the function $L(\tau)$. Using the gradient method, the first-order approximation is given by

$$L(\tau) \approx L(\tau_p) + \frac{\partial L(\tau_p)}{\partial \tau_p} (\tau - \tau_p) \quad (3.3)$$

It is obvious from (3.3) that the direction of the function's fastest decrease at the point τ_p is opposite to its gradient $\partial L(\tau_p) / \partial \tau_p$. So we can write

$$\tau'_{p+1} = \tau'_p - \lambda_p \frac{\partial L(\tau_p)}{\partial \tau_p}, \quad p = 0, 1, 2, \dots \quad (3.4)$$

where p denotes the iterations. If λ_p is sufficiently small, $L(\tau_p) \geq L(\tau_{p+1})$ and the approximation improves after every iteration. However, it is difficult to find an optimal λ_p at every iteration. Better results can be achieved by using the second-order approximation given by [25]

$$L(\tau) \approx L(\tau_p) + \frac{\partial L(\tau_p)}{\partial \tau_p} (\tau - \tau_p) + (\tau - \tau_p)' \frac{\partial^2 L(\tau_p)}{\partial \tau_p^2} (\tau - \tau_p) \quad (3.5)$$

An iterative algorithm can now be developed by approximating $L(\tau)$ with some function $Q(\tau, \tau_p)$, in the vicinity of τ_p , with respect to τ where

$$\tau_{p+1} = \arg \min_{\tau} Q(\tau, \tau_p) \quad (3.6)$$

The closeness of approximation or the *residual* $H(\tau, \tau_p)$ is defined as

¹ A formal and detailed description of divergence is given in Chapter 4.

$$H(\tau, \tau_p) = L(\tau) - Q(\tau, \tau_p) \quad (3.7)$$

Intuitively, for good approximation, the residual $H(\tau, \tau_p)$ must be much smaller than $Q(\tau, \tau_p)$. However, the same is not true for their increments and it cannot be guaranteed that both $L(\tau)$ and $Q(\tau, \tau_p)$ decrease at the same time, although $L(\tau)$ decreases when $H(\tau, \tau_p)$ and $Q(\tau, \tau_p)$ decrease. Also, if $H(\tau, \tau_p)$ always decreases then a decrease of $Q(\tau, \tau_p)$ causes a decrease of $L(\tau)$. Thus the parameter sequence, given by (3.6), decrease $L(\tau)$. In the next section we discuss the expectation-maximization algorithm, which is a special case of this minimization algorithm.

3.1.1 Expectation-Maximization (EM) Algorithm

The expectation-maximization (EM) algorithm [27], [28] estimates model parameters iteratively after starting from some initial guess. As the name suggests, the EM algorithm consists of two major steps; an expectation step, followed by a maximization step. The expectation is with respect to the unknown underlying variables, using the current estimate of the parameters conditioned upon the observations. The maximization step provides a new estimate of the parameters based on the assumption that the expectation step is correct. These two steps are iterated until the algorithm achieves convergence. The concept is illustrated in Figure 3.1.

Let us consider the likelihood function given by

$$\begin{aligned} L(\tau) &= \sum_x f(x) \log(g(x, \tau)) \\ &= E[\log(g(x, \tau)) | f(x)] \end{aligned} \quad (3.8)$$

where $f(x)$ and $g(x, \tau)$ are the probability distributions of the original process and the model, respectively; and $E[\cdot]$ denotes the expectation operation.

Now let us assume that we have a function $\kappa(z; x, \tau) > 0$ whose sum over z is a constant. The function $\kappa(z; x, \tau)$ can be interpreted as a probability distribution of z by normalizing the function and is given by

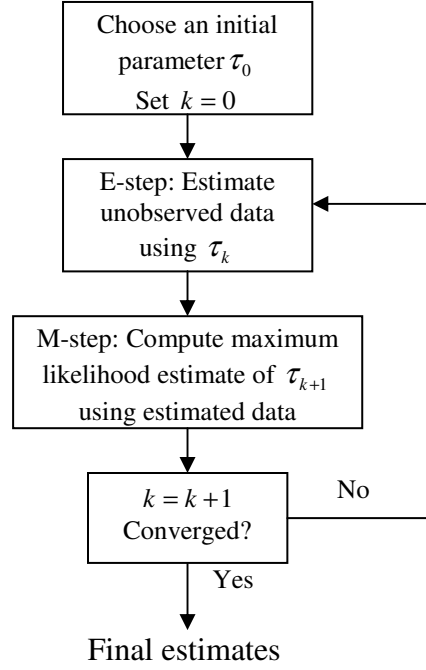


Figure 3.1: Block diagram of the EM algorithm.

$$\sum_z \kappa(z; x, \tau) = 1 \quad (3.9)$$

$\kappa(z; x, \tau)$ is called the *auxiliary* probability distribution. Different auxiliary functions result in different EM algorithms. However, auxiliary functions are selected in such a way that the maximization problem can be solved analytically.

Let us denote a new function

$$\Psi(z; x, \tau) = g(x, \tau) \kappa(z; x, \tau) \quad (3.10)$$

or

$$\log g(x, \tau) = \Psi(z; x, \tau) - \log \kappa(z; x, \tau) \quad (3.11)$$

Multiplying both sides of (3.11) with $r(z; x, \tau) = f(x) \kappa(z; x, \tau)$ and summing with respect to z and x yields [25]

$$\sum_x f(x) \log g(x, \tau) = \sum_x \sum_z \log \Psi(z; x, \tau) r(z; x, \tau_p) - \sum_x \sum_z \log \kappa(z; x, \tau) r(z; x, \tau_p) \quad (3.12)$$

which can be re-written in the following form

$$L(\tau) = Q(\tau, \tau_p) + H(\tau, \tau_p) \quad (3.13)$$

where

$$Q(\tau, \tau_p) = \sum_x \sum_z \log \Psi(z; x, \tau) r(z; x, \tau_p) = E[\log \Psi(z; x, \tau) | r(z; x, \tau_p)] \quad (3.14)$$

$$H(\tau, \tau_p) = -\sum_x \sum_z \log \kappa(z; x, \tau) r(z; x, \tau_p) = -E[\log \kappa(z; x, \tau) | r(z; x, \tau_p)] \quad (3.15)$$

Using *Jensen's inequality* [29] it can be shown that

$$E[\log \kappa(z; x, \tau) | \kappa(z; x, \tau_p)] \leq E[\log \kappa(z; x, \tau_p) | \kappa(z; x, \tau_p)] \quad (3.16)$$

Multiplying both the sides of (3.16) by $-f(x)$ and summing, we obtain that

$H(\tau, \tau_p) \geq H(\tau_p, \tau_p)$, which proves that $H(\tau_p, \tau_p)$ is a global minimum of $H(\tau, \tau_p)$. The

likelihood maximum can therefore be obtained using (3.6) and we have

$$\hat{\tau} = \arg \max_{\tau} E[\log g(x, \tau) | f(x)] \quad (3.17)$$

Equation (3.17) can be solved iteratively by the EM algorithm as

$$\tau_{p+1} = \arg \max_{\tau} E[\log \Psi(z; x, \tau) | r(z; x, \tau_p)], \quad p = 0, 1, 2, \dots \quad (3.18)$$

The expectation and maximization steps can be summarized as

Expectation step: Calculate $E[\log \Psi(z; x, \tau) | r(z; x, \tau_p)]$

Maximization step: Find a maximum of $E[\log \Psi(z; x, \tau) | r(z; x, \tau_p)]$

Usually the likelihood function $E[\log g(x, \tau) | f(x)]$ increases significantly in the first iteration while the subsequent iterations provide smaller increments. Since the likelihood function is monotonically increasing and is bounded by $E[\log f(x) | f(x)]$, it converges to some finite value. This limiting point could be a local maxima or a *saddle point*. The saddle point, however, is not stable and small rounding errors usually help the algorithm to get out of the local maxima.

3.1.1.1 Convergence of EM Algorithm

Like any other iterative algorithm, convergence of the EM algorithm is an important factor. The conditions of convergence can be formulated in terms of $L(\tau)$ [30]. If a maximization function converges in the limit, the function may attain a local maxima or it can converge to the saddle point. Based on applications, convergence may not be an

issue as most of the time local maxima provide acceptable results. It takes only several iterations of the algorithm to improve on the initial approximation significantly.

The other important factor is the speed of convergence. Even though the EM algorithm is robust, it is slow. However, the speed of convergence depends on the selection of the auxiliary function $\kappa(z; x, \tau)$ and $\Psi(z; x, \tau)$. If $\Psi(z; x, \tau) = g(x; \tau)$, the algorithm becomes faster and it takes only one iteration to solve the problem. Usually a simple form of $\Psi(z; x, \tau)$ is selected so that the maximization step is analytically tractable. Unfortunately this may result in a very slow algorithm. In order to accelerate the convergence, the EM algorithm can be combined with faster but less robust algorithms [25]. For example, after several iterations with the EM algorithm, the Newton-Raphson method can be applied for faster convergence. Although the gain in the convergence speed comes at the cost of increased complexity.

3.1.2 Phase-Type (PH) Distribution

In this section a special case of the EM algorithm, called phase-type (PH) distribution, is described to approximate a given probability distribution $f(x)$. The phase-type distribution is important in modeling the same observations value (state) such as error-free interval in the frame error sequence of a communication link or to estimate the fade duration in a quantized fading envelope. The phase-type distribution is a special case of the matrix-geometric distribution and can be expressed as [25]

$$g(x, \tau) = \pi \mathbf{A}^{x-1} \mathbf{b}, \quad x = 1, 2, \dots \quad (3.19)$$

where x is the state duration, $g(x, \tau)$ is the probability of observing x , $\pi = \{\pi_i\}, 1 \leq i \leq u$, is a row vector, $\mathbf{b} = \{b_i\}, 1 \leq i \leq u$, is a column vector, and $\mathbf{A} = \{a_{ij}\}, 1 \leq i, j \leq u$, is a square matrix such that

$$\mathbf{P} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \pi & 0 \end{bmatrix} \quad (3.20)$$

is a stochastic matrix [4]. The vectors π and $\mathbf{b}$ define the probability of being in one of the u states for the given observation value. The matrix $\mathbf{A}$ describes the transition

probabilities between u states. Since $\mathbf{P}$ is stochastic, it follows that $\boldsymbol{\pi}\mathbf{1}=\mathbf{1}$ and $\mathbf{b}=(\mathbf{I}-\mathbf{A})\mathbf{1}$, where $\mathbf{1}=[1,1,\dots,1]'$ and $\mathbf{I}$ is the identity matrix.

The matrix product in (3.19) can be written as

$$g(x, \tau) = \sum_z \pi_{i_0} a_{i_0 i_1} \cdots a_{i_{x-2} i_{x-1}} a_{i_{x-1}, u+1} \quad (3.21)$$

where $a_{i, u+1} = b_i$ and $z = \mathbf{i}_0^{x-1}$. The auxiliary function is defined as

$$\Psi(z; x, \tau) = \pi_{i_0} a_{i_0 i_1} \cdots a_{i_{x-2} i_{x-1}} a_{i_{x-1}, u+1} \quad (3.22)$$

so that

$$g(x, \tau) = \sum_z \Psi(z; x, \tau) \quad (3.23)$$

Dividing both sides of (3.23) by $g(x, \tau)$ and defining $\kappa(z; x, \tau) = \Psi(z; x, \tau) / g(x, \tau)$, $\kappa(z; x, \tau)$ can be interpreted as a probability distribution.

After some mathematical manipulation it can be shown that

$$\sum_{z, i_0=i} \Psi(z; x, \tau) = \sum_{\mathbf{i}_1^{x-1}} a_{i_0 i_1} \cdots a_{i_{x-2} i_{x-1}} a_{i_{x-1}, u+1} \quad (3.24)$$

is the i -th element of the matrix $\mathbf{A}^{x-1} \mathbf{b}$.

The *forward* and *backward* variables are defined as

$$\begin{aligned} \alpha_i(x, \tau) &= \boldsymbol{\pi} \mathbf{A}^x \mathbf{e}_i^t \\ \beta_i(x, \tau) &= \mathbf{e}_i \mathbf{A}^{x-1} \mathbf{b} \end{aligned} \quad (3.25)$$

where $1 \leq i \leq u$ and $\mathbf{e}_i$ is a $1 \times u$ row vector whose i -th element is a one and the rest of the elements are zero. Since

$$g(x, \tau) = \sum_{i=1}^u \pi_i \beta_i(x, \tau) = \boldsymbol{\pi} \mathbf{A}^{x-1} \mathbf{b} \quad (3.26)$$

it can be shown that

$$\pi_{i, p+1} = \pi_{i, p} \sum_x \beta_i(x, \tau_p) \rho(x, \tau_p) \quad (3.27)$$

where p is the iteration index and $\rho(x, \tau_p) = f(x) / g(x, \tau_p)$. Similarly for $j = 1, 2, \dots, u$ it can be shown that

$$\sum_{z, i_{k-1}=i, i_k=j} \Psi(z; x, \tau) = \alpha_i(k-1, \tau) a_{ij} \beta_j(x-k, \tau) \quad (3.28)$$

The algorithm for fitting the PH distribution to $f(x)$ can be derived as [25]

$$\pi_{i,p+1} = \pi_{i,p} \sum_x \beta_i(x, \tau_p) \rho(x, \tau_p) \quad (3.29)$$

$$a_{ij,p+1} = a_{ij,p} \frac{\sum_x \rho(x, \tau_p) \sum_{k=1}^{x-1} \alpha_i(k-1, \tau_p) \beta_j(x-k, \tau_p)}{\sum_x \rho(x, \tau_p) \sum_{k=1}^x \alpha_i(k-1, \tau_p) \beta_i(x-k+1, \tau_p)} \quad (3.30)$$

$$b_{i,p+1} = b_{i,p} \frac{\sum_x \rho(x, \tau_p) \alpha_i(x-1, \tau_p)}{\sum_x \rho(x, \tau_p) \sum_{k=1}^x \alpha_i(k-1, \tau_p) \beta_i(x-k+1, \tau_p)} \quad (3.31)$$

Note that the value of $\mathbf{b}$ can also be found from the normalization condition

$$\mathbf{b} = (\mathbf{I} - \mathbf{A})\mathbf{1} \quad (3.32)$$

Let the sequence, $\mathbf{x}_1^T$, be the collection of some PH distributed variable, where T denotes the length of the sequence. Replacing $f(x)$ with $1/T$, the statistical version of the EM algorithm can be rewritten as

$$\pi_{i,p+1} = \frac{\pi_{i,p}}{T} \sum_{m=1}^T \beta_i(x_m, \tau_p) / g(x_m, \tau_p) \quad (3.33)$$

$$a_{ij,p+1} = a_{ij,p} \frac{\sum_{m=1}^T \sum_{k=1}^{x_m-1} \alpha_i(k-1, \tau_p) \beta_j(x_m-k, \tau_p) / g(x_m, \tau_p)}{\sum_{m=1}^T \sum_{k=1}^{x_m} \alpha_i(k-1, \tau_p) \beta_i(x_m-k+1, \tau_p) / g(x_m, \tau_p)} \quad (3.34)$$

$$b_{i,p+1} = b_{i,p} \frac{\sum_{m=1}^T \alpha_i(x_m-1, \tau_p) / g(x_m, \tau_p)}{\sum_{m=1}^T \sum_{k=1}^{x_m} \alpha_i(k-1, \tau_p) \beta_i(x_m-k+1, \tau_p) / g(x_m, \tau_p)} \quad (3.35)$$

The phase-type distributed parameters are initialized such that $\pi_{i,0} = 1/u$, $a_{ii,0} > a_{ii\pm 1,0} > a_{ii\pm 2,0} > \dots$, and the initial values of the vector $\mathbf{b}$ are obtained from (3.32). Also, the elements initialized to zero don't change their values through the iteration steps.

Since $\lim_{x \rightarrow \infty} \alpha_i(x, \tau) = 0$, $1 \leq i \leq u$, for large T the dynamic range of $\alpha_i(x, \tau)$ is small.

This causes *underflow* problem which leads to loss of accuracy. It is therefore imperative to normalize the forward variables and the procedure is called *scaling*. We will discuss

scaling in details in a later section. With scaling incorporated, (3.34) and (3.35) can be written as [25]

$$\begin{aligned} a_{ij,P+1} &= C_i \sum_{m=1}^T A_{ij}(x_m, \tau_P) / g(x_m, \tau_P) \\ b_{i,P+1} &= C_i \sum_{m=1}^T \alpha_i(x_m - 1, \tau_P) / g(x_m, \tau_P) \end{aligned} \quad (3.36)$$

where C_i is the normalization factor given by

$$C_i^{-1} = \sum_{m=1}^T \left[A_{ij}(x_m, \tau_P) / g(x_m, \tau_P) + \alpha_i(x_m - 1, \tau_P) / g(x_m, \tau_P) \right] \quad (3.37)$$

and

$$A_{ij}(x, \tau) = a_{ij} \boldsymbol{\pi} \sum_{k=1}^{x-1} \mathbf{A}^{k-1} \mathbf{U}_{ij} \mathbf{A}^{x-k-1} \mathbf{b} \quad (3.38)$$

with $\mathbf{U}_{ij} = \mathbf{e}_i' \mathbf{e}_j$ is the matrix whose ij -th element is one and the rest are zeros.

3.2 Training Algorithms for Hidden Markov Models

Hidden Markov models (HMM) are described by the compact triplet $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ where $\mathbf{A}$ is the state transition probability matrix, $\mathbf{B}$ is the observation probability matrix and $\boldsymbol{\pi}$ is the initial state distribution². There are several algorithms to optimize the model parameters based on the observation data or *training sequence*. The objective of the training algorithm is to optimally adapt the model parameters to the observed sequence which results in an accurate modeling of the underlying statistics in the training data. Since the model parameters are updated using recursive algorithms, the number of iterations must be fixed before training starts. As shown in Figure 3.2, the model structure and dimension are also set as an input to the training algorithm. The most commonly used HMM training algorithms are

- a) Baum-Welch Algorithm
- b) Segmental K -Means Algorithm
- c) Genetic Algorithm

² The model parameters $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ are defined in Chapter 2.

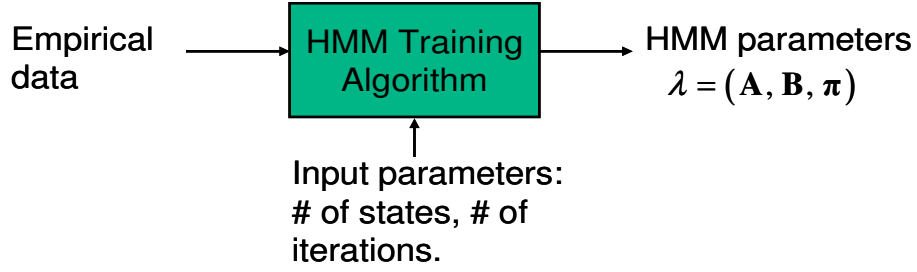


Figure 3.2: Block diagram of HMM training.

Each algorithm has its relative advantages/disadvantages and a performance/complexity trade-off must be made based on the application. The algorithms are introduced next with particular emphasis on the Baum-Welch algorithm.

3.2.1 Baum-Welch Algorithm (BWA)

The Baum-Welch algorithm (BWA), developed by Leonard Baum and Lloyd Welch, optimizes the parameters of a finite-state machine (FSM) model using empirical observations [1], [7], [31]. Given an observation sequence, the algorithm estimates the HMM parameters through an iterative procedure to maximize the probability that the model will regenerate the observation sequence.

Let $\mathbf{O} = o_1, o_2, \dots, o_T$ be an observation sequence where $o_k \in \mathbf{V}$ is the observation symbol at time k that can take on one of the M values from the set $\mathbf{V} = [v_1, v_2, \dots, v_M]$ and T is the observation sequence length. In order to calculate the probability of the observation sequence, $\mathbf{O}$, given the model, $\lambda (= \mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, we must enumerate over every possible state sequence, $\mathbf{Q} = q_1, q_2, \dots, q_T$, where each state in the model can take one of the N values, $q_k \in \mathbf{S}$, and $\mathbf{S} = [s_1, s_2, \dots, s_N]$. The probability of the observation sequence, $\mathbf{O}$, for the state sequence, $\mathbf{Q}$, is given by

$$P(\mathbf{O} | \mathbf{Q}, \lambda) = \prod_{t=1}^T P(o_t | q_t, \lambda) \quad (3.39)$$

If statistical independence of observation is assumed then

$$P(\mathbf{O} | \mathbf{Q}, \lambda) = b_{q_1}(o_1) \cdot b_{q_2}(o_2) \cdots b_{q_T}(o_T) \quad (3.40)$$

The probability of the state sequence, $\mathbf{Q}$, given the model, λ , is expressed as

$$P(\mathbf{Q} | \lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T} \quad (3.41)$$

Given the model, λ , the joint probability of the state sequence $\mathbf{Q}$ and the observation sequence $\mathbf{O}$ can be calculated by multiplying (3.40) and (3.41)

$$P(\mathbf{O}, \mathbf{Q} | \lambda) = \pi_{q_1} b_{q_1}(o_1) \prod_{k=2}^T a_{q_{k-1} q_k} b_{q_k}(o_k) \quad (3.42)$$

The probability of $\mathbf{O}$, given the model, λ , is obtained by summing the joint probability over all possible state sequences. This gives

$$\begin{aligned} P(\mathbf{O} | \lambda) &= \sum_{\mathbf{Q}} P(\mathbf{O}, \mathbf{Q} | \lambda) \\ &= \sum_{q_1} \sum_{q_2} \cdots \sum_{q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \cdots a_{q_{T-1} q_T} b_{q_T}(o_T) \end{aligned} \quad (3.43)$$

The above equation can be interpreted in the following way. At the beginning ($t = 1$), the model is in state q_1 with probability π_{q_1} , and generates observation o_1 with probability $b_{q_1}(o_1)$. At $t = 2$, the model makes a transition to state q_2 with probability $a_{q_1 q_2}$ and generates observation o_2 with probability $b_{q_2}(o_2)$. The process continues with more and more terms until the observation sequence ends.

According to the definition in (3.43), the calculation of $P(\mathbf{O} | \lambda)$ requires on the order of $2T \cdot N^T$ calculations. At each time instant, there are N possible states which results in N^T possible state sequences. For all these state sequences, the exact number of computations is $(2T-1)N^T$ multiplications and N^T-1 additions. This level of complexity is obviously not practical since even for small value of N and T ; e.g. for $N = 5$, and $T = 100$, the number of computations are $\approx 10^{72}$!

A more efficient algorithm is required that does not increase the computation exponentially. The *forward-backward* procedure is one such algorithm. As the name suggests, the algorithm recursively computes two sets of probabilities in either direction – *forward* (from start to the end of the data) and *backward* (from end to the start of the data). The *forward variable*, $\alpha_t(i)$, is defined as the probability of partial observation sequence $o_1, o_2, \dots, o_t$ (until time t) and state $q_t = s_i$ at time t , given the model, λ .

$$\alpha_t(i) = P(o_1, o_2, \dots, o_t, q_t = s_i | \lambda) \quad (3.44)$$

Calculation of the forward variables involves three steps: initialization, induction and termination.

Initialization:

$$\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N \quad (3.45)$$

Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(o_{t+1}), \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (3.46)$$

Termination:

$$P(\mathbf{O} | \lambda) = \sum_{i=1}^N \alpha_T(i) \quad (3.47)$$

The first step initializes the forward variables as the joint probability of initial observation o_1 and state $q_t = s_i$. The induction step is illustrated in Figure 3.3(a). As seen from the Figure 3.3(a), at time instant $t+1$, the state s_j can be reached from N possible states, s_i , $1 \leq i \leq N$, at time t (assuming each a_{ij} is non-zero). The product $\alpha_t(i) a_{ij}$ is the probability of the joint event that $o_1, o_2, \dots, o_t$ are observed and state s_j is reached at time $t+1$ through previous state s_i at time t . Summing the product over all possible states, s_i , $1 \leq i \leq N$, at time t , gives the probability of s_j at time $t+1$ with all the previous partial observations taken into account. Now $\alpha_{t+1}(j)$ can be computed by multiplying the summed quantity by the probability $b_j(o_{t+1})$. The calculation in (3.46) is enumerated for all states and for all observations.

Similarly, the *backward variable*, $\beta_t(i)$, is defined as the probability of partial observation sequence $o_{t+1}, o_{t+2}, \dots, o_T$ (starting from time $t+1$) given the state $q_t = s_i$ at time t , and the model, λ .

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \dots, o_T | q_t = s_i, \lambda) \quad (3.48)$$

Calculation of the backward variables involves two steps; initialization and induction.

Initialization:

$$\beta_T(i) = 1, \quad 1 \leq i \leq N \quad (3.49)$$

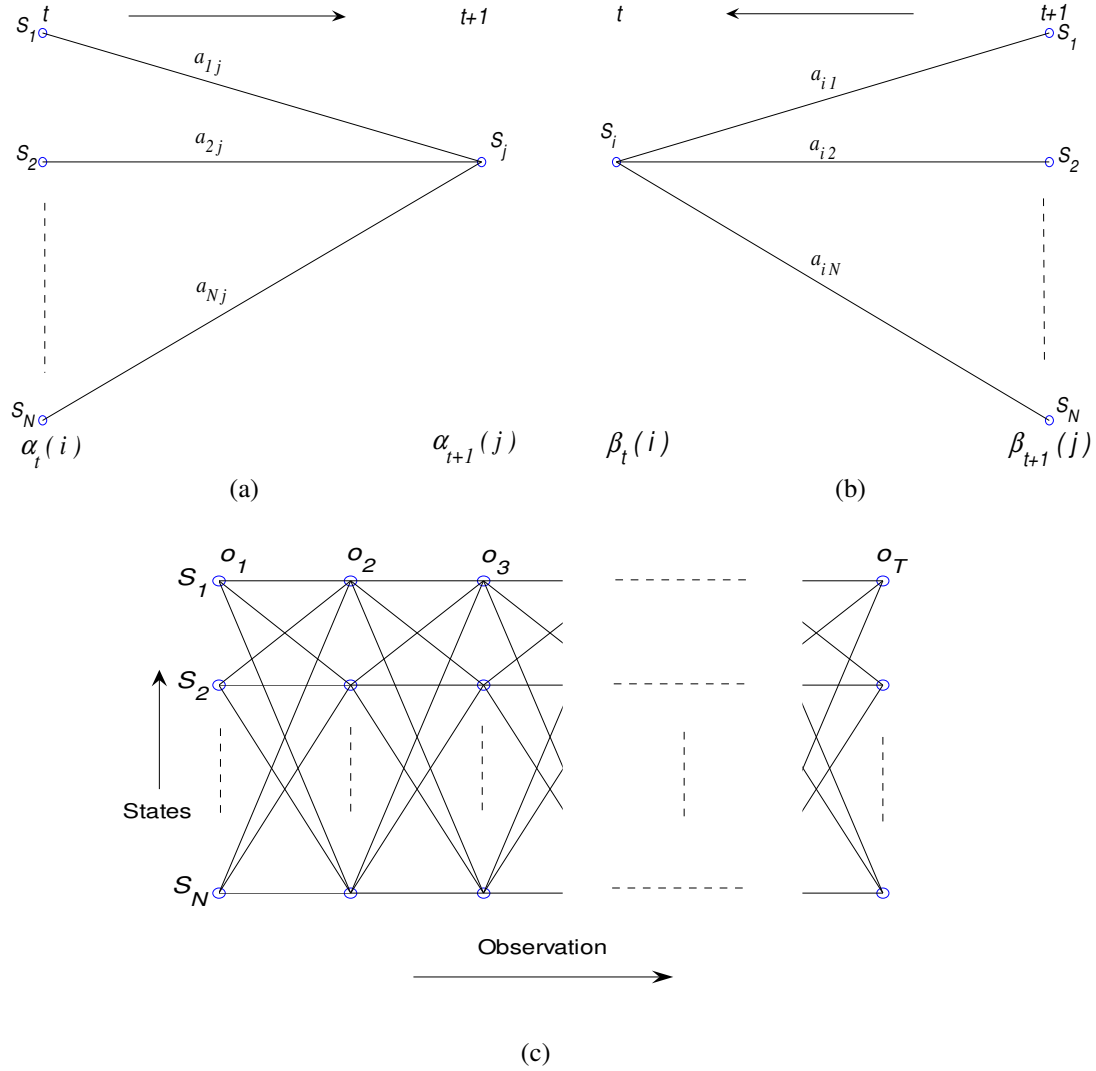


Figure 3.3: Illustration of forward-backward procedure. (a) Forward variables. (b) Backward variables. (c) Computation lattice.

Induction:

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (3.50)$$

The first step initializes (arbitrarily) the backward variables to be 1 for all states. The second step is depicted in Figure 3.3(b). In order to calculate the backward variable at state $q_t = s_i$ and time t , and account for the partial observation sequence from time $t+1$ to T , several probabilities must be taken into account. The first of these is the transition probability term a_{ij} , for all possible s_j (again assuming each a_{ij} is non-zero). The second

is the observation o_{t+1} in state j , which is denoted by $b_j(o_{t+1})$. The third term, $\beta_{t+1}(j)$, accounts for the remaining partial observation sequence corresponding to each s_j .

Based on the forward and backward variables, two more probability terms are defined that help us estimate the HMM parameters. The first variable is $\gamma_t(i)$, which defines the probability of being in state s_i at time t , given the model, λ , and the whole observation sequence, $\mathbf{O}$. Thus

$$\gamma_t(i) = P(q_t = s_i | \mathbf{O}, \lambda) \quad (3.51)$$

In terms of forward and backward variables

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^N \alpha_t(i)\beta_t(i)} \quad (3.52)$$

The partial observation sequence $o_1, o_2, \dots, o_t$ is accounted for by $\alpha_t(i)$ while the rest of the observation sequence $o_{t+1}, o_{t+2}, \dots, o_T$ is accounted for by $\beta_t(i)$. The denominator makes $\gamma_t(i)$ a probability measure so that

$$\sum_{i=1}^N \gamma_t(i) = 1 \quad (3.53)$$

The second probability term, denoted by $\xi_t(i, j)$, defines the probability of being in state s_i at time t and s_j at time $t+1$, given the model λ and the whole observation sequence $\mathbf{O}$.

$$\xi_t(i, j) = P(q_t = s_i, q_{t+1} = s_j | \mathbf{O}, \lambda) \quad (3.54)$$

The calculation of $\xi_t(i, j)$ is illustrated in Figure 3.4. In terms of forward and backward variables we have

$$\xi_t(i, j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{P(\mathbf{O} | \lambda)} \quad (3.55)$$

where $P(\mathbf{O} | \lambda)$ can be expressed as

$$\begin{aligned} P(\mathbf{O} | \lambda) &= \sum_{i=1}^N \alpha_t(i)\beta_t(i) \\ &= \sum_{i=1}^N \sum_{j=1}^N \alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j) \end{aligned} \quad (3.56)$$

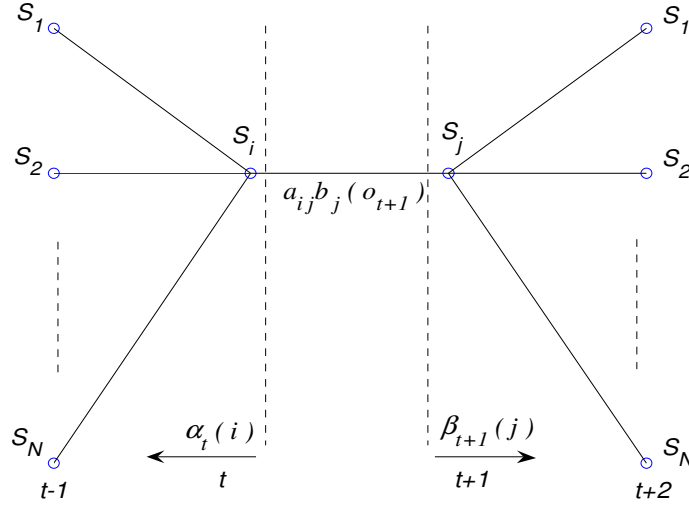


Figure 3.4: Illustration of the joint event that the system is in state s_i at time t and state s_j at time $t+1$.

In (3.56), the first t observations are accounted for by $\alpha_t(i)$, ending in state s_i at time t . In the first line of (3.56) the rest of the observations are accounted for by $\beta_t(i)$, while in the second line this is re-written as sum of $a_{ij}b_j(o_{t+1})$ over all j to account for the transition to s_j at time $t+1$ with the occurrence of observation o_{t+1} . The remaining part of the observation sequence is accounted for by $\beta_{t+1}(j)$. Finally (3.55) can be written as

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \quad (3.57)$$

The numerator term in (3.56) can be defined as $P(q_t = s_i, q_{t+1} = s_j, \mathbf{O} | \lambda)$. The denominator normalizes $\xi_t(i, j)$, so that it sums to unity.

The term $\gamma_t(i)$, as defined previously, is the probability of being in state s_i at time t given the observation and the model. So $\gamma_t(i)$ can be related to $\xi_t(i, j)$ as

$$\gamma_t(i) = \sum_{j=1}^N \xi_t(i, j) \quad (3.58)$$

The sum of $\gamma_t(i)$ over time t can be viewed as the expected number of times state s_i is visited (excluding time instant T). Similarly, the summation of $\xi_t(i, j)$ over t (again

excluding time instant T) can be interpreted as the expected number of transitions from state s_i to state s_j . Thus

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from } s_i \quad (3.59)$$

$$\sum_{t=1}^{T-1} \xi_t(i, j) = \text{expected number of transitions from } s_i \text{ to } s_j \quad (3.60)$$

In addition to the forward and backward variables, the above formulas also count event occurrences which can be used in HMM parameter estimation. Usually the HMM model parameters $\lambda (= \mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ are estimated using

$$\bar{\pi}_i = \text{expected number of times in state } s_i \text{ at time } t=1, \gamma_1(i) \quad (3.61)$$

$$\begin{aligned} \bar{a}_{ij} &= \frac{\text{expected number of transitions from state } s_i \text{ to state } s_j}{\text{expected number of transitions from state } s_i} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \end{aligned} \quad (3.62)$$

$$\begin{aligned} \bar{b}_j(k) &= \frac{\text{expected number of time in state } s_j \text{ and observing symbol } v_k}{\text{expected number of times in state } s_j} \\ &= \frac{\sum_{t=1}^T \gamma_t(i)}{\sum_{t=1}^T \gamma_t(j)} \end{aligned} \quad (3.63)$$

Using (3.61)-(3.63), the HMM parameters are *reestimated* and the new model is defined as $\bar{\lambda} (= \bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\boldsymbol{\pi}})$. Baum and his colleagues have proven that either the initial model λ defines a critical point in the likelihood function, i.e., $\lambda = \bar{\lambda}$ or the new model $\bar{\lambda}$ is more likely in the sense that $P(\mathbf{O} | \bar{\lambda}) > P(\mathbf{O} | \lambda)$. Based on this likelihood criteria, $\bar{\lambda}$ replaces λ in each iteration and the reestimation calculation are repeated. After each iteration, the probability of $\mathbf{O}$ being observed by the estimated model increases until some limiting point is reached. The final result of this estimation procedure is called a maximum likelihood estimate of the HMM.

The reestimation formulas of (3.61)-(3.63) can be derived directly by considering Baum's auxiliary function given by [31]

$$\mathbf{Q}(\lambda, \bar{\lambda}) = \sum_{\mathbf{Q}} P(\mathbf{Q} | \mathbf{O}, \lambda) \log [P(\mathbf{O}, \mathbf{Q} | \bar{\lambda})] \quad (3.64)$$

where $\bar{\lambda}$ is the auxiliary variable corresponding to λ .

It can be shown that the Baum-Welch reestimation techniques are essentially similar to the Expectation-Maximization algorithm discussed before. For example,

Expectation step: Calculate $\mathbf{Q}(\lambda, \bar{\lambda})$

Maximization step: Find $\max_{\bar{\lambda}} [\mathbf{Q}(\lambda, \bar{\lambda})]$

Note that the element of $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\pi}$, that are initialized to zero do not change their original values in the reestimation step. Also, the reestimation procedure inherently preserves the stochastic constraints of the HMM parameters. Specifically,

$$\begin{aligned} \sum_{i=1}^N \bar{\pi}_i &= 1 \\ \sum_{i=1}^N a_{ij} &= 1, \quad 1 \leq j \leq N \\ \sum_{k=1}^M b_j(k) &= 1, \quad 1 \leq j \leq N \end{aligned} \quad (3.65)$$

3.2.1.1 Scaling

The calculation of the forward and backward variables involves multiplication of fractions. Since the forward and backward variables are calculated recursively, as t increases, they tend to zero exponentially. Therefore for a large observation sequence, the dynamic range of these variables will exceed the precision of most machines. Thus, the forward and backward variables must be *scaled* properly to prevent numerical underflow. The basic idea is to multiply $\alpha_t(i)$ by a coefficient that is independent of i , so that the scaled $\alpha_t(i)$ remains within the dynamic range of the computer for the entire observation sequence. The backward variables are also scaled by the same coefficient for each t . The scaling constant is defined as

$$C_t = \frac{1}{\sum_{i=1}^N \alpha_t(i)} \quad (3.66)$$

The scaled values of $\alpha_{t+1}(j)$, denoted by $\hat{\alpha}_{t+1}(j)$, are given by

$$\hat{\alpha}_{t+1}(j) = \frac{\sum_{i=1}^N \hat{\alpha}_t(i) a_{ij} b_j(o_{t+1})}{\sum_{j=1}^N \sum_{i=1}^N \hat{\alpha}_t(i) a_{ij} b_j(o_{t+1})}, \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (3.67)$$

By induction $\hat{\alpha}_t(i)$ can be expressed as

$$\hat{\alpha}_{t+1}(j) = \left(\prod_{\tau=1}^{T-1} C_\tau \right) \hat{\alpha}_t(j), \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (3.68)$$

Using (3.66) and (3.55), $\alpha_t(j)$ can be rewritten as

$$\hat{\alpha}_t(j) = \frac{\alpha_t(j)}{\sum_{j=1}^N \alpha_t(j)}, \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (3.69)$$

The backward variables are also scaled by the same coefficients. The scaled values of $\beta_t(i)$, denoted by $\hat{\beta}_t(i)$, are given by

$$\hat{\beta}_t(j) = C_t \beta_t(j), \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (3.70)$$

Similar to (3.66), the backward variables can be recursively computed as

$$\hat{\beta}_{t+1}(j) = \left(\prod_{s=t+1}^T C_s \right) \beta_{t+1}(j) = D_{t+1} \beta_{t+1}(j), \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (3.71)$$

Using (3.66) and (3.69), the reestimation formula in (3.62) can be written in terms of the scaled forward and backward variables as

$$\begin{aligned} \bar{a}_{ij} &= \frac{\sum_{t=1}^{T-1} \hat{\alpha}_t(i) a_{ij} b_j(o_{t+1}) \hat{\beta}_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^N \hat{\alpha}_t(i) a_{ij} b_j(o_{t+1}) \hat{\beta}_{t+1}(j)} \\ &= \frac{\sum_{t=1}^{T-1} C_t \alpha_t(i) a_{ij} b_j(o_{t+1}) D_{t+1} \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^N C_t \alpha_t(i) a_{ij} b_j(o_{t+1}) D_{t+1} \beta_{t+1}(j)} \end{aligned} \quad (3.72)$$

The term $C_t D_{t+1}$ is independent of t and has the general form

$$C_t D_{t+1} = \prod_{s=1}^t C_s \prod_{s=t+1}^T C_s = \prod_{s=1}^T C_s = C_T \quad (3.73)$$

The term $C_t D_{t+1}$ cancels out of both the numerator and the denominator of (3.70) resulting in the original reestimation given by (3.62). In a similar way, it can be shown that scaling does not change the parameter reestimation of $\boldsymbol{\pi}$ and $\mathbf{B}$, given by (3.61) and (3.63), respectively.

However, the computation of $P(\mathbf{O} | \lambda)$ changes due to scaling. Unlike (3.47), $P(\mathbf{O} | \lambda)$ cannot be calculated by summing up the scaled forward variables, $\hat{\alpha}_t(i)$. Instead, $P(\mathbf{O} | \lambda)$ can be expressed in terms of the scaling constant as

$$P(\mathbf{O} | \lambda) = \frac{1}{\sum_{t=1}^T C_t} \quad (3.74)$$

For a long observation sequence, $P(\mathbf{O} | \lambda)$ is very small and is usually determined as

$$\log[P(\mathbf{O} | \lambda)] = -\sum_{t=1}^T \log C_t \quad (3.75)$$

where the base of the logarithm is arbitrary.

3.2.1.2 Convergence

Since the BWA is iterative, the number of iterations must be determined for a required level of accuracy. Like the EM algorithm, it takes the BWA only several iterations to improve on the initial approximation. One of the ways to determine the convergence of the model is to monitor the value of $\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}$ at the end of each iteration. Based on the values of $\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}$ a stopping criteria can be set to terminate the algorithm once the values no longer change, within the given accuracy, from iteration to iteration. Also, with prior knowledge, a fixed number of iterations can be performed for acceptable accuracy.

Another commonly used method for tracking the convergence is to observe the evolution of $\log[P(\mathbf{O} | \lambda)]$. This is referred to as *the log-likelihood function* and can be

calculated using the forward algorithm as shown in (3.75). Since the BWA is guaranteed to convergence to a local maxima, if the successive values of $\log[P(\mathbf{O}|\lambda)]$ differ very little from iteration to iteration or remain virtually constant over a few iterations, the algorithm may be terminated.

3.2.1.3 Initialization

Theoretically, the reestimation procedure calculates the HMM parameters in such a way that at least the local maximum of the likelihood function is achieved. In practical problems of interest, the optimization surface is very complex and usually has many local maxima. However, by choosing a prudent initial parameter set, the algorithm can be led to higher local maxima or, in the best case, a global maxima. The initialization also affects the convergence rate of the algorithm. This fact is illustrated in Figure 3.5, where initial parameter set λ_2 achieves global maxima while λ_1 fails to do so because the algorithm is trapped in the local maxima range. One possible solution to overcome local maxima is to perturb the model parameters with random noise.

However, there is no straightforward way to initialize parameters that guarantee global maximization. It has been reported that random or uniform initialization of $\mathbf{A}$ and $\boldsymbol{\pi}$ produces acceptable reestimation in almost all cases [7]. For the parameters of $\mathbf{B}$, the

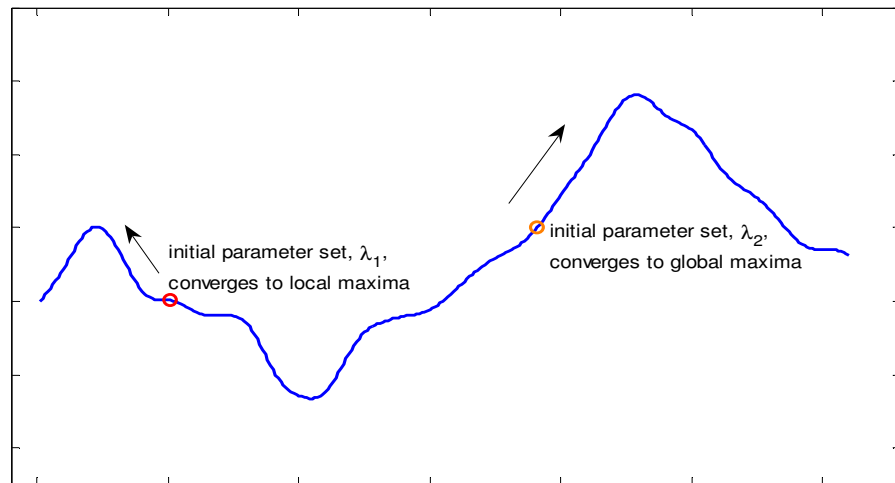


Figure 3.5: Convergence of the BWA is affected by the initial parameter set.

initial estimates can be obtained by segmenting the observation sequence into states or by using the segmental K -means algorithm [32], which will be introduced in a later section.. Note that the estimates of $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\pi}$ for a given set of observation sequences, are not unique unless the initial parameters of $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\pi}$ are identical. Since the observation sequence is a function of both $\mathbf{A}$ and $\mathbf{B}$, various combination of $\mathbf{A}$ and $\mathbf{B}$ may produce *statistically equivalent* result and a specific realization depends on the given initial parameter. In Chapter 4, statistically equivalent HMM's will be formally defined.

3.2.1.4 Complexity and Memory Requirements

The computational complexity of the BWA is mostly dominated by the forward and backward procedures. Exact calculations show that there are $N(N+1)(T-1)+N$ multiplications and $N(N-1)(T-1)$ additions required to calculate the forward variables, $\alpha_t(i)$, which is of the order of $O(N^2T)$. Similarly, the complexity for backward variables, $\beta_t(i)$, is also $O(N^2T)$. The other two probability terms $\gamma_t(i)$ and $\xi_t(i, j)$ also have the complexity of the order of $O(N^2T)$. A detailed calculation of the BWA indicates that the *space complexity*, which is a direct function of the number of observation symbols M , is $O(N(N+M+TN))$ and the time complexity is $O(N(1+T(M+N)))$ [33]. However, in most of the practical applications $T \gg M$, which reduces the space and time complexity for the BWA to $O(N^2T)$ for each iteration.

Once the HMM parameters are updated, it takes very little memory space to store them. To be specific, the total memory requirement for the HMM is $x(NN+NM+N)$, where x is the number of bit used to store each HMM parameter. For example, if a 32-bit floating point format is used to store the parameters of an HMM with 5 states and 2 observation symbols, then the total memory required is 1280 bits.

3.2.2 Segmental K - Means Algorithm

The intensive computations in the forward-backward recursion can be avoided by using another training algorithm called segmental K -means algorithm. As the name suggests, this is an algorithm for estimating the HMM parameters by embedding the well-known K -means iterative procedure for clustering data [9], [32]. Figure 3.6 illustrates the algorithm. First, the model parameters are initialized randomly or based on any available model that is appropriate to the training sequences. Once the model is initialized, the set of observation sequences is segmented into states based on the initial model. The segmentation is obtained by finding the optimum state sequence. Using the Viterbi Algorithm (VA), the most likely path can be easily obtained via trace back [34], [35]. Segmenting each of the training sequences results in a maximum likelihood estimate of the set of observations for each state. Each of the observation vectors within a state can be coded using the M -codeword codebook. The updated estimate of the $b_j(k)$ parameter is given by [9]

$$b_j(k) = \frac{\text{number of vectors with codebook index } k \text{ in state } j}{\text{number of vectors in state } j} \quad (3.76)$$

Based on the state segmentation, the state transition probabilities are given by

$$a_{ij} = \frac{\text{number of transition from state } i \text{ to state } j}{\text{number of transitions from state } i} \quad (3.77)$$

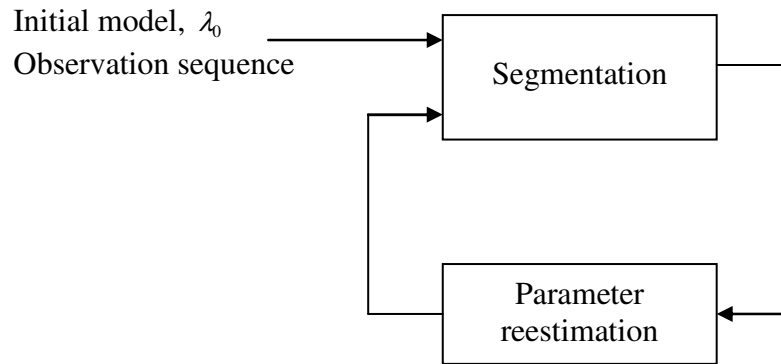


Figure 3.6: Block diagram of segmental K -means algorithm.

An updated model $\bar{\lambda}$ is obtained from the new model parameters. The resulting model is compared to the previous one. If the model distance exceeds a threshold, then the new

model replaces the old model and the recursive parameter update is repeated. If the distance falls below the threshold, then convergence is assumed and the model parameters are saved as the final model parameter.

3.2.3 Genetic Algorithm

The training algorithms discussed in the previous sections may not achieve global parameter optimization. The BWA is guaranteed to achieve only a local maxima as the convergence is heavily dependent on the initial value. To overcome this problem, a stochastic search method called Genetic Algorithm (GA) can be used as a global optimization tool for training the HMM parameters [36]-[38].

The gene is a biological concept. The genetic algorithm has been used as a function optimizer, which has been shown to be an effective optimization tool. The GA mimics natural evolution and performs global searching within the defined search space. The GA is manipulated on a population of genetic representations of solutions. An encoding mechanism is employed to encode the potential solutions. Following initialization, the fitness value of each of the solutions is evaluated and the solutions are ranked based on that value. The selection process facilitates fitter solutions to have a greater chance of survival in subsequent generations while weaker solutions gradually die out. *Crossover* and *mutation* are two natural genetic operators which simulate genetic recombination and variation, respectively. The crossover exchanges portions of the encoded solutions to generate new solutions, while mutation randomly alters portions of the encoded solutions. Mutation provides the GA the global search capability as it regain missed information that has not been generated during the previous generations.

The GA can be used to train the HMM parameters as the reestimation of the parameters may be considered as a global optimum search in the parameter space. It allows the HMM to escape from the local maxima to obtain a global maxima or at least a more optimized local maxima. The GA, as it applies to HMM training, can be described with the following steps:

3.2.3.1 Encoding Mechanism

In HMM training, first the model parameters are encoded into a string of real numbers. As shown in Figure 3.7, all HMM parameters, $a_{ij}, b_j(k), \pi_i, 1 \leq i, j \leq N, 1 \leq k \leq M$, are arranged in a line to construct a chromosome, where individual parameters acts as a gene. For an initial population $p(0) = [C_1, C_2, \dots, C_S]$, each chromosome $C_i, 1 \leq i \leq S$, represents an HMM. Based on the configuration of the model, if any parameter is zero it need not be encoded as gene of the chromosome. If all the HMM parameters are non-zero the total number of genes in the chromosome is $NN + NM + N$.

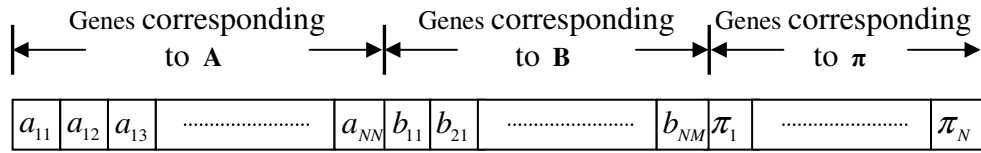


Figure 3.7: Genetic representation of HMM parameters.

3.2.3.2 Fitness Value Evaluation

The fitness value of each chromosome is calculated in terms of the log likelihood function. Using the forward algorithm and (3.75), the log likelihood function for each chromosome can be evaluated. The fitness value usually transforms the log likelihood value into a measure of relative performance. The fitness value for chromosome C_i is given by [39]

$$f_i = \frac{\rho_i}{\sum_{j=1}^S \rho_j}, \quad 1 \leq i \leq S \quad (3.78)$$

where ρ_i is the reciprocal of the log likelihood function $\log[P(\mathbf{O}|\lambda_i)]$, λ_i is the corresponding HMM representation of chromosome C_i , and S is the total number of chromosomes in the population.

3.2.3.3 Selection Mechanism

Many selection procedures employ the *roulette wheel* mechanism to probabilistically select individual chromosomes based on some measure of their performance. In a roulette wheel, each chromosome is allocated a sector of the wheel with an angle subtended by the sector at the center of the wheel, which is equal to $2\pi \times f_i$. A chromosome is selected as an offspring if a randomly generated number in the range $(0, 2\pi]$ falls into the sector corresponding to that chromosome.

3.2.3.4 Crossover and Mutation

Once selected, the chromosomes are passed through the crossover and mutation procedure to create the next generation. Two parent chromosomes from the population are randomly chosen and a random number in the range $(0, 1]$ is generated to compare with the crossover rate. If the random number is greater than the crossover rate, no crossover takes place and the two parents enter as offspring for the next generation. Otherwise crossover points are randomly selected and the portions of the chromosomes are exchanged between the parents. The number of crossover points is a user defined parameter which is set beforehand. Figure 3.8 shows three point crossovers between two chromosomes.

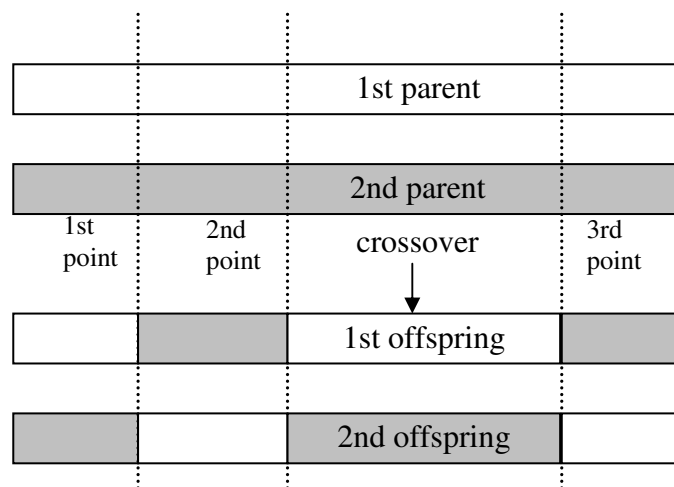


Figure 3.8: Illustration of three point crossover.

In the mutation process, one chromosome is randomly chosen from the population to carry out the mutation operation. Like in crossover, a random number in the range $(0,1]$ is generated and compared against the mutation rate to determine if mutation can be carried out. If the decision is true the genes equal to the number of mutation points are randomly selected and replaced by real random numbers in the range $(0,1]$. Since the probability values change after crossover and mutation, they must be normalized to meet the stochastic constraint given by (3.65).

Out of the three training algorithms for HMM, the BWA has been preferred for the following reasons. The BWA adapts the model parameters to maximize the probability of the observed sequence by summing up the probability over all possible state sequences. The BWA is superior to the segmental K -means algorithm as it is a full likelihood approach and produces asymptotically unbiased and efficient estimates of the model parameters [40]. However, the segmental K -means algorithm requires less computation since it makes use of the efficient Viterbi algorithm. When compared against the genetic algorithm, the BWA is computationally efficient. Even though the GA is a global optimization tool, it is not guaranteed that the HMM parameters will achieve global maxima. Furthermore, it is hard to compare two HMMs trained by the GA since, due to the randomness in the model, two observation sequences from the same training set may produce two disparate models. With the same initial conditions, the BWA produces equivalent models for all observation sequences in the same training set.

3.3 Multiple Observations Training of the HMM

Often times a single observation sequence is not sufficient to train the HMM parameters. This is primarily due to the fact that a single observation sequence may not contain all of the statistical features that the model is expected to learn. Also, if the transition probability of any state is very small, there are only a small number of occurrences of that state. Hence, in order to have sufficient data for a reliable estimate of all model parameters, multiple observation sequence training must be done.

Let us consider a set of K observation sequences given by

$$\mathbf{O} = [\mathbf{O}^1, \mathbf{O}^2, \dots, \mathbf{O}^K] \quad (3.79)$$

where

$$\mathbf{O}^k = o_1^k, o_2^k, \dots, o_{T_k}^k, \quad 1 \leq k \leq K \quad (3.80)$$

is the k -th individual observation sequence. The probability of the entire set $\mathbf{O}$, given the model, λ , can be expressed as [41]

$$\begin{aligned} P(\mathbf{O} | \lambda) &= P(\mathbf{O}^1 | \lambda) P(\mathbf{O}^2 | \mathbf{O}^1, \lambda) \dots P(\mathbf{O}^K | \mathbf{O}^{K-1} \dots \mathbf{O}^1, \lambda) \\ P(\mathbf{O} | \lambda) &= P(\mathbf{O}^2 | \lambda) P(\mathbf{O}^3 | \mathbf{O}^2, \lambda) \dots P(\mathbf{O}^1 | \mathbf{O}^K \dots \mathbf{O}^2, \lambda) \\ &\vdots \\ P(\mathbf{O} | \lambda) &= P(\mathbf{O}^K | \lambda) P(\mathbf{O}^1 | \mathbf{O}^K, \lambda) \dots P(\mathbf{O}^{K-1} | \mathbf{O}^K \mathbf{O}^{K-2} \dots \mathbf{O}^1, \lambda) \end{aligned} \quad (3.81)$$

Based on the probabilities in (3.81), the multiple observation probability can be written in a compact form as

$$P(\mathbf{O} | \lambda) = \sum_{k=1}^K w_k P(\mathbf{O}^k | \lambda) \quad (3.82)$$

where

$$\begin{aligned} w_1 &= \frac{1}{K} P(\mathbf{O}^2 | \mathbf{O}^1, \lambda) \dots P(\mathbf{O}^K | \mathbf{O}^{K-1} \dots \mathbf{O}^1, \lambda) \\ w_2 &= \frac{1}{K} P(\mathbf{O}^3 | \mathbf{O}^2, \lambda) \dots P(\mathbf{O}^1 | \mathbf{O}^K \dots \mathbf{O}^2, \lambda) \\ &\vdots \\ w_K &= \frac{1}{K} P(\mathbf{O}^1 | \mathbf{O}^K, \lambda) \dots P(\mathbf{O}^{K-1} | \mathbf{O}^K \mathbf{O}^{K-2} \dots \mathbf{O}^1, \lambda) \end{aligned} \quad (3.83)$$

are weighting factors. The weighing factors take into account the independence/dependence relationship among the observation sequences.

For multiple observation sequences, the auxiliary function given by (3.64), can be modified as

$$\mathbf{Q}(\lambda, \bar{\lambda}) = \sum_{k=1}^K w_k \mathbf{Q}_k(\lambda, \bar{\lambda}) \quad (3.84)$$

where, as before, $\bar{\lambda}$ is the auxiliary variable corresponding to λ . Baum's auxiliary functions related to individual observations are given by

$$\mathbf{Q}_k(\lambda, \bar{\lambda}) = \sum_{\mathbf{Q}} P(\mathbf{Q} | \mathbf{O}^k, \lambda) \log [P(\mathbf{O}^k, \mathbf{Q} | \bar{\lambda})] \quad (3.85)$$

Using the Lagrange multiplier method an objective function can be defined as [31]

$$F(\bar{\lambda}) = Q(\lambda, \bar{\lambda}) + \sum_{i=1}^N c_{ai} \left[1 - \sum_{j=1}^N \bar{a}_{ij} \right] + \sum_{j=1}^M c_{bj} \left[1 - \sum_{k=1}^M \bar{b}_j(k) \right] + c_{\pi} \left[1 - \sum_{i=1}^N \bar{\pi}_i \right] \quad (3.86)$$

where c_{ai} , c_{bj} and c_{π} are Lagrange multipliers. Differentiating the objective function with respect to individual parameters, the following modified reestimation formulas are obtained:

$$\bar{a}_{ij} = \frac{\sum_{k=1}^K w_k P(\mathbf{O}^k | \lambda) \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) a_{ij} b_j(o_{t+1}^k) \hat{\beta}_{t+1}^k(j)}{\sum_{k=1}^K w_k P(\mathbf{O}^k | \lambda) \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}, \quad 1 \leq i, j \leq N \quad (3.87)$$

$$\bar{b}_j(l) = \frac{\sum_{k=1}^K w_k P(\mathbf{O}^k | \lambda) \sum_{\substack{t=1 \\ s.t. o_t^k = v_l}}^{T_k} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}{\sum_{k=1}^K w_k P(\mathbf{O}^k | \lambda) \sum_{t=1}^{T_k} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}, \quad 1 \leq j \leq N, 1 \leq l \leq M \quad (3.88)$$

$$\bar{\pi}_i = \frac{\sum_{k=1}^K w_k P(\mathbf{O}^k | \lambda) \hat{\alpha}_1^k(i) \hat{\beta}_1^k(i)}{\sum_{k=1}^K w_k P(\mathbf{O}^k | \lambda)}, \quad 1 \leq i \leq N \quad (3.89)$$

3.3.1 Independent Observation Sequences

If each observation sequence is assumed independent of every other observation sequences then we have

$$P(\mathbf{O} | \lambda) = \prod_{k=1}^K P(\mathbf{O}^k | \lambda) \quad (3.90)$$

The individual weighting factors can therefore be reduced as

$$w_k = \frac{1}{K} P(\mathbf{O} | \lambda) / P(\mathbf{O}^k | \lambda) \quad (3.91)$$

Substituting the weighting factors in (3.87)-(3.89), the following reestimation formulas are obtained

$$\bar{a}_{ij} = \frac{\sum_{k=1}^K \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) a_{ij} b_j(o_{t+1}^k) \hat{\beta}_{t+1}^k(j)}{\sum_{k=1}^K \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}, \quad 1 \leq i, j \leq N \quad (3.92)$$

$$\bar{b}_j(l) = \frac{\sum_{k=1}^K \sum_{t=1}^{T_k} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}{\sum_{k=1}^K \sum_{t=1}^{T_k} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}, \quad 1 \leq j \leq N, 1 \leq l \leq M \quad (3.93)$$

$$\bar{\pi}_i = \frac{1}{K} \sum_{k=1}^K \hat{\alpha}_1^k(i) \hat{\beta}_1^k(i), \quad 1 \leq i \leq N \quad (3.94)$$

3.3.2 Uniformly Dependent Observation Sequences

All of the weighting factors become equal if the individual observation sequences are uniformly dependent on one another,

$$w_k = \text{const}, \quad 1 \leq k \leq K \quad (3.95)$$

Substituting the weighting factors in (3.87)-(3.89), the reestimation formulas are given by

$$\bar{a}_{ij} = \frac{\sum_{k=1}^K P(\mathbf{O}^k | \lambda) \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) a_{ij} b_j(o_{t+1}^k) \hat{\beta}_{t+1}^k(j)}{\sum_{k=1}^K P(\mathbf{O}^k | \lambda) \sum_{t=1}^{T_k-1} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}, \quad 1 \leq i, j \leq N \quad (3.96)$$

$$\bar{b}_j(l) = \frac{\sum_{k=1}^K P(\mathbf{O}^k | \lambda) \sum_{t=1}^{T_k} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}{\sum_{k=1}^K P(\mathbf{O}^k | \lambda) \sum_{t=1}^{T_k} \hat{\alpha}_t^k(i) \hat{\beta}_t^k(i)}, \quad 1 \leq j \leq N, 1 \leq l \leq M \quad (3.97)$$

$$\bar{\pi}_i = \frac{\sum_{k=1}^K P(\mathbf{O}^k | \lambda) \hat{\alpha}_1^k(i) \hat{\beta}_1^k(i)}{\sum_{k=1}^K P(\mathbf{O}^k | \lambda)}, \quad 1 \leq i \leq N \quad (3.98)$$

3.3.3 Ensemble Average of Independent Observation Sequences

In this method, one model is trained for each of the K observation sequences. Since the observation sequences are mutually independent, there are K independent model estimates from the observation sequence set. By taking a simple weighted average of the model parameters, the reestimation formulas are obtained as [42]

$$\bar{a}_{ij} = \sum_{k=1}^K \frac{w_k}{N_a} a_{ij}^k \quad (3.99)$$

$$\bar{b}_j(l) = \sum_{k=1}^K \frac{w_k}{N_b} b_j^k(l) \quad (3.100)$$

$$\bar{\pi}_i = \sum_{k=1}^K \frac{w_k}{N_\pi} \pi_i^k \quad (3.101)$$

where N_a, N_b, N_π are normalizing constants to meet the stochastic constraints. With $w_k = 1, 1 \leq k \leq K$, direct ensemble averaging across the observation sequences results.

3.4 Conclusion

In this chapter the HMM parameter estimation algorithms were discussed. The most popular and widely used Baum-Welch algorithm was discussed in detail. The relative advantages and disadvantages of the BWA was contrasted with other training algorithms, i.e., the segmental K -mean algorithm and the genetic algorithm. Since the BWA is a better fit both in terms of model accuracy and when comparing different HMMs, we exclusively used it for HMM training. Note that irrespective of the algorithm used, the HMM parameters must be reestimated once the underlying physical system changes. For example, in modeling a radio channel, if the SNR at the receiver input changes, a new set of HMM parameters must be estimated to emulate the changed statistics. In the next chapter the notion of divergence will be introduced as a distance measure between two HMMs. Also, the HMM parameters will be analyzed from a matrix theory perspective to optimize the model structure.

Chapter 4: Distance Measures and Parametric Analysis of HMM

In estimation and classification problems, it is often required to compare different hidden Markov models. Usually a dissimilarity measure, commonly referred to as distance or divergence, is formulated between two HMMs. In this chapter we discuss different distance measures between HMMs in considerable detail. We start with a simple geometrically motivated Euclidean distance measure between two HMMs. Next more intricate divergence measures are introduced that calculate statistical distance in the probability space. For stationary Markov chains, the analytic expression for the Kullback-Leibler divergence (KLD) rate is compared against the simulation results. Unlike Markov models, there is no closed form solution to compute the KLD rate between HMMs. A novel heuristic divergence measure is proposed that closely approximates the KLD rate between HMMs. The proposed method is compared against other distance measures for HMMs. Later in the chapter, the HMM parameters are investigated to improve its performance. From the matrix theory perspective, the HMM parameters are analyzed to optimize the model structure.

4.1 Notion of Distance Measure

The need for comparing different HMMs often arises in a variety of contexts. For example, in an automatic speech recognition system, distinct sets of HMMs are used to classify multilingual phoneme [43]. Other applications include vocabulary selection [44], monitoring of the training procedure, and evaluation of the re-estimation process [45]. A common way to accomplish this comparison is to introduce a suitable *distance* measure

between the HMMs. A distance measure for HMMs is a function which assigns any pair of models, $\lambda (= \mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ and $\lambda' (= \mathbf{A}', \mathbf{B}', \boldsymbol{\pi}')$, a positive real number $D(\lambda, \lambda')$. A simple distance measure for discrete HMMs is the sum of the Euclidean distances between the pair of transition probability matrices, $(\mathbf{A}, \mathbf{A}')$, and the observation symbol probability matrices, $(\mathbf{B}, \mathbf{B}')$. Specifically

$$D_{ec}(\lambda, \lambda') = \sqrt{\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^N |a_{ij} - a'_{ij}|^2} + \sqrt{\frac{1}{N} \sum_{j=1}^N \sum_{l=1}^M |b_j(l) - b'_j(l)|^2} \quad (4.1)$$

where the number of states, N , and the number of observation symbols per state, M , in both the model are assumed to be equal.

However, the simplistic distance measure in (4.1) does not take into account the probabilistic interaction between $\mathbf{A}$ and $\mathbf{B}$, and thus ignores the characterization of a hidden Markov process. As a result it is possible to find a pair of models λ and λ' , where $D_{ec}(\lambda, \lambda')$ is non-zero but the probability of generating a given sequence for the two models are exactly same. For example, let us consider two HMMs given by

$$\begin{aligned} \boldsymbol{\pi} &= [1.0 \quad 0.0 \quad 0.0], & \boldsymbol{\pi}' &= [1.0 \quad 0.0 \quad 0.0] \\ \mathbf{A} &= \begin{bmatrix} 0.4 & 0.0 & 0.6 \\ 0.0 & 0.4 & 0.6 \\ 0.5 & 0.1 & 0.4 \end{bmatrix}, & \mathbf{A}' &= \begin{bmatrix} 0.2 & 0.2 & 0.6 \\ 0.3 & 0.1 & 0.6 \\ 0.0 & 0.6 & 0.4 \end{bmatrix} \\ \mathbf{B} &= \begin{bmatrix} 1.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix}, & \mathbf{B}' &= \begin{bmatrix} 1.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \end{bmatrix} \end{aligned} \quad (4.2)$$

It is easy to verify that even though $D_{ec}(\lambda, \lambda')$ is not zero, these two models generate any given sequence with the same probability. This fact raises an important issue in the Markov literature, called the *identifiability problem* [46], [47], where two ostensibly different Markov chains generate the same stochastic process. Two HMMs λ and λ' are said to be *equivalent* if and only if the probability of generating any sequence for the two models are identical. In other words

$$P(\mathbf{O} | \lambda) = P(\mathbf{O} | \lambda'), \quad \forall \mathbf{O} \quad (4.3)$$

Obviously in order to identify equivalent HMMs, we need to formulate a dissimilarity measure that represents distance in the probability space rather than in the Cartesian space. Divergence is one such probabilistic distance which will be introduced next.

4.1.1 Kullback-Leibler Divergence

A different distance measure that overcomes the drawback in (4.1) is known as Kullback-Leibler divergence (KLD) [26], [29]. The KLD or *relative entropy* measures the average discrimination information between two hypotheses modeled as random variables. These hypotheses are often represented by their respective probability density functions. The KLD of a probability density function, $F_1(z)$, with respect to another probability density function, $F_2(z)$, $D_{KL}(F_1 \parallel F_2)$, is defined as

$$D_{KL}(F_1 \parallel F_2) = \int F_1(z) \log \frac{F_1(z)}{F_2(z)} dz \quad (4.4)$$

where the base of the logarithm is arbitrary. Also, to ensure the existence of the integral, the two density functions are assumed to be continuous with respect to one another. If the random variables are defined on the *discrete space*, the probability density functions and the integral in (4.4) are replaced by the respective probability mass functions, f_1 and f_2 , defined on χ , and summation, respectively. This gives

$$D_{KL}(f_1 \parallel f_2) = \sum_{i \in \chi} f_{1,i} \log \frac{f_{1,i}}{f_{2,i}} \quad (4.5)$$

The KLD is a member of a larger class of distributional distances known as the *Ali-Silvey distances* [48]. The KLD is also known as *I-divergence*. Throughout this thesis, in the formulation of the KLD, we use the conventions $0/0 = 1$ and $0 \log 0 = 0$.

4.1.1.1 Properties of KLD

The KLD has been widely used in the fields of information theory and statistical signal processing for nearly 50 years. For example, the KLD has been applied in approximating probability distributions [49], signal processing [50], pattern recognition [51], hypothesis testing [52], [53]. Now we will discuss some of the important properties of the KLD.

Additivity: For independent observations x_1 and x_2 , the KLD between the joint distributions $f_1(x_1, x_2)$ and $f_2(x_1, x_2)$ is equal to the sum of the KLD between the marginal distributions. So that

$$D_{KL}(f_1(x_1, x_2) \| f_2(x_1, x_2)) = D_{KL}(f_1(x_1) \| f_2(x_1)) + D_{KL}(f_1(x_2) \| f_2(x_2)) \quad (4.6)$$

This property simply derives from the fact that for independent observations the logarithm of a product equals the sum of logarithms.

Non-negativity: Based on the *log-sum inequality* [64] and exploiting the fact that $\log t$ is a monotonically increasing function of t , we have

$$D_{KL}(f_1(x) \| f_2(x)) \geq 0 \quad (4.7)$$

where the equality holds *if and only if* $f_1(x) = f_2(x)$ for all x .

Convexity: The KLD is convex, i.e., if (f_1, g_1) and (f_2, g_2) are two pairs of probability density functions then

$$D(\lambda f_1 + (1-\lambda)f_2 \| \lambda g_1 + (1-\lambda)g_2) \leq \lambda D(f_1 \| g_1) + (1-\lambda)D(f_2 \| g_2), \quad \forall \lambda \in [0,1] \quad (4.8)$$

Non-symmetric: The KLD is not symmetric in general, i.e.,

$$D_{KL}(f_1 \| f_2) \neq D_{KL}(f_2 \| f_1) \quad (4.9)$$

This implies that the first distribution may match the second distribution rather well, while the contrary may not be true. However, a symmetric version of the KLD, $D_{KL(S)}$, can be calculated in the following way [45]

$$D_{KL(S)}(f_1 \| f_2) = \frac{D_{KL}(f_1 \| f_2) + D_{KL}(f_2 \| f_1)}{2} \quad (4.10)$$

4.2 KLD Between Markov Models

The probability functions in Markov models are not easy to compute and requires numerical methods to compute the Kullback-Leibler divergence given by (4.5). Recently, by using the theory of non-negative matrices and Perron-Frobenius theory [54], the KLD between two Markov models has been computed analytically [55]. Let $\mathbf{Q} = q_1, q_2, q_3, \dots, q_T$ be the state sequence of a first-order time-invariant Markov chain with finite alphabet $\mathbf{S} = [s_1, s_2, \dots, s_N]$. In other words, there are N states in the model with $q_t \in \mathbf{S}$ for any t . The state transition probabilities are compactly defined by the transition matrix, $\mathbf{A} = \{a_{ij}\}, 1 \leq i, j \leq N$. The probability of the state sequence, $\mathbf{Q}$, under the model $\mathbf{A}$ is

$$\begin{aligned}
P(\mathbf{Q} | \mathbf{A}) &= \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \dots a_{q_{T-1} q_T} \\
&= \pi_{q_1} \prod_{v=2}^T a_{q_{v-1} q_v}
\end{aligned} \tag{4.11}$$

where $\pi = \{\pi_i\}, 1 \leq i \leq N$, is the initial state distribution. Similarly, the probability of state sequence, $\mathbf{Q}$, under another model with the same dimension $\mathbf{A}' = \{a'_{ij}\}$, and initial state distribution $\pi' = \{\pi'_i\}, 1 \leq i, j \leq N$ can be calculated. Now the KLD between the two Markov chains is given by

$$D_{KL}(\mathbf{A} \parallel \mathbf{A}') = \sum_{\mathbf{Q} \in \mathbf{S}^T} P(\mathbf{Q} | \mathbf{A}) \log \frac{P(\mathbf{Q} | \mathbf{A})}{P(\mathbf{Q} | \mathbf{A}')} \tag{4.12}$$

where $\mathbf{Q}$ is the state sequence of length T defined on $\mathbf{S}^T$.

Since the divergence between the two Markov models depends on length T , the *divergence rate* is a better discriminating measure. The KLD rate, as a limit of the time-averaged value of D_{KL} , is given by

$$\bar{D}_{KL}(\mathbf{A} \parallel \mathbf{A}') = \lim_{T \rightarrow \infty} \frac{1}{T} D_{KL}(\mathbf{A} \parallel \mathbf{A}') \tag{4.13}$$

The existence of the KLD rate between two *ergodic* Markov chains has been proven in [56].

Let us assume that $\mathbf{A}$ and π are *continuous* with respect to $\mathbf{A}'$ and π' , respectively. This condition implies that $a'_{ij} = 0 \Rightarrow a_{ij} = 0$ and $\pi'_i = 0 \Rightarrow \pi_i = 0$, $1 \leq i, j \leq N$, which is denoted by $\mathbf{A} \ll \mathbf{A}'$.

Now we define

$$m(q_2 | q_1 = s_i) \triangleq \sum_{j \in \mathbf{S}} a_{ij} \log \frac{a_{ij}}{a'_{ij}}, \quad 1 \leq i, j \leq N \tag{4.14}$$

Note that $m(q_2 | q_1 = s_i)$ is well defined for all s_i since we assumed $\mathbf{A} \ll \mathbf{A}'$. After expanding the logarithm in (4.12), we have

$$D_{KL}(\mathbf{A} \parallel \mathbf{A}') = \pi(\mathbf{I} + \mathbf{A} + \dots + \mathbf{A}^{T-2}) \mathbf{W} + \sum_{q_1 \in \mathbf{S}} \pi_{q_1} \log \frac{\pi_{q_1}}{\pi'_{q_1}} \tag{4.15}$$

where $\mathbf{I}$ is the identity matrix and

$$\mathbf{W} = [m(q_2 | q_1 = s_1), m(q_2 | q_1 = s_2), \dots, m(q_2 | q_1 = s_N)]^t \tag{4.16}$$

in which $[\cdot]^t$ denotes the transposition of the quantity in brackets.

By combining (4.13) and (4.15), and assuming $\mathbf{A}$ is *irreducible*¹, the KLD rate is given by

$$\bar{D}_{KL}(\mathbf{A} \parallel \mathbf{A}') = \boldsymbol{\pi} \mathbf{L} \mathbf{W} \quad (4.17)$$

where

$$\mathbf{L} = \mathbf{d} \mathbf{c} = \begin{bmatrix} p_1 & p_2 & \cdots & p_N \\ p_1 & p_2 & \cdots & p_N \\ \vdots & \vdots & \ddots & \vdots \\ p_1 & p_2 & \cdots & p_N \end{bmatrix} \quad (4.18)$$

in which $\mathbf{c}$ and $\mathbf{d}$ are left and right eigenvector of $\mathbf{A}$ associated with eigenvalue of 1, respectively, and $\mathbf{p} = [p_1, p_2, \dots, p_N]$ is the stationary vector of $\mathbf{A}$, defined by (2.14). Exploiting the steady-state property of the Markov chain, the KLD rate in (4.17) reduces to the following compact form

$$\bar{D}_{KL}(\mathbf{A} \parallel \mathbf{A}') = \mathbf{p} \mathbf{W} \quad (4.19)$$

Now we relax the first-order assumption on the Markov process $\mathbf{Q} = q_1, q_2, q_3, \dots, q_T$. Let us assume that $\mathbf{Q}$ has arbitrary order k where $P_{\mathbf{A}_k}$ and $P_{\mathbf{A}'_k}$ are probability distributions for $\mathbf{Q}$. Now $\mathbf{R} = r_1, r_2, \dots$ can be defined as the process obtained by k -step blocking of the Markov chain. To be specific,

$$r_n = [q_n, q_{n+1}, \dots, q_{n+k-1}] \quad (4.20)$$

Note that $\mathbf{R}$ is a first-order Markov chain with N^k states. Now let $\boldsymbol{\pi} = [\pi_1, \pi_2, \dots, \pi_{N^k}]$ and $\boldsymbol{\pi}' = [\pi'_1, \pi'_2, \dots, \pi'_{N^k}]$ denote the initial state distribution, and $\mathbf{A} = \{a_{ij}\}$ and $\mathbf{A}' = \{a'_{ij}\}$, $1 \leq i, j \leq N^k$, denote the probability transition matrices for $\mathbf{R}$. It can be shown that the KLD rate for the first order Markov process $\mathbf{S}$ and the k -th order Markov process $\mathbf{R}$ is the same [55]. So the KLD rate is given by

$$\bar{D}_{KL}(\mathbf{A} \parallel \mathbf{A}') = \frac{1}{T} \boldsymbol{\pi} (\mathbf{I} + \mathbf{A} + \cdots + \mathbf{A}^{T-2}) \mathbf{W} + \frac{1}{T} \sum_{q_1 \in \mathbf{S}} \pi_{q_1} \log \frac{\pi_{q_1}}{\pi'_{q_1}} \quad (4.21)$$

where

¹ $\mathbf{A}$ is called *reducible* if, by permutation of its rows and columns, it can be transformed into $\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & 0 \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$. Otherwise, $\mathbf{A}$ is called *irreducible*.

$$\mathbf{W} = \left[m(q_2 | q_1 = s_1), m(q_2 | q_1 = s_2), \dots, m(q_2 | q_1 = s_{N^k}) \right]^T \quad (4.22)$$

4.2.1 Numerical Example

In this section the numerical evaluation of the Kullback-Leibler divergence rate between two Markov chains is compared against the analytic formula. In order to compute the summation in (4.12) numerically, usually the Monte-Carlo method is used. This can be done by rewriting (4.13) as

$$\bar{D}_{KL}(\mathbf{A} \parallel \mathbf{A}') = \frac{1}{T} E_{\mathbf{A}} \left[\log(P(\mathbf{Q} | \mathbf{A})) - \log(P(\mathbf{Q} | \mathbf{A}')) \right] \quad (4.23)$$

where $E_{\mathbf{A}}[\cdot]$ is the expectation operator with respect to $\mathbf{A}$.

Now a set of sample state sequences $\mathbf{Q}^1, \mathbf{Q}^2, \dots, \mathbf{Q}^L$ of length T can be randomly and independently generated based on the initial distribution $\boldsymbol{\pi}$ and the Markov chain $\mathbf{A}$. The KLD rate can be approximated as

$$\bar{D}_{KL}(\mathbf{A} \parallel \mathbf{A}') \approx \frac{1}{T} \frac{1}{L} \sum_{i=1}^L \left[\log(P(\mathbf{Q}^i | \mathbf{A})) - \log(P(\mathbf{Q}^i | \mathbf{A}')) \right] \quad (4.24)$$

where $P(\mathbf{Q}^i | \mathbf{A})$ is calculated using (4.11).

Basically $\bar{D}_{KL}(\mathbf{A} \parallel \mathbf{A}')$ is a measure of how well model λ matches the observation sequences generated by it, relative to how well model λ' matches the same sequences. Usually for large L the right hand side of (4.24) converges to $\bar{D}_{KL}(\mathbf{A}' \parallel \mathbf{A}')$.

Let us consider two first order Markov chains given by

$$\begin{aligned} \boldsymbol{\pi} &= [0.5 \quad 0.5], & \boldsymbol{\pi}' &= [0.5 \quad 0.5] \\ \mathbf{A} &= \begin{bmatrix} 0.9 & 0.1 \\ 0.8 & 0.2 \end{bmatrix}, & \mathbf{A}' &= \begin{bmatrix} 0.8 & 0.2 \\ 0.4 & 0.6 \end{bmatrix} \end{aligned} \quad (4.25)$$

For the Monte-Carlo method, for the arbitrarily selected sequence lengths, the KLD rate is calculated by averaging over 1000 independent trials. This is illustrated in Figure 4.1. Using the analytic expression in (4.19), the KLD rate between two Markov models was found to be 0.0481. As expected, as T increases, the Monte-Carlo estimation of divergence rate converges to the analytic results. The natural logarithm was used for both analytic and simulation results.

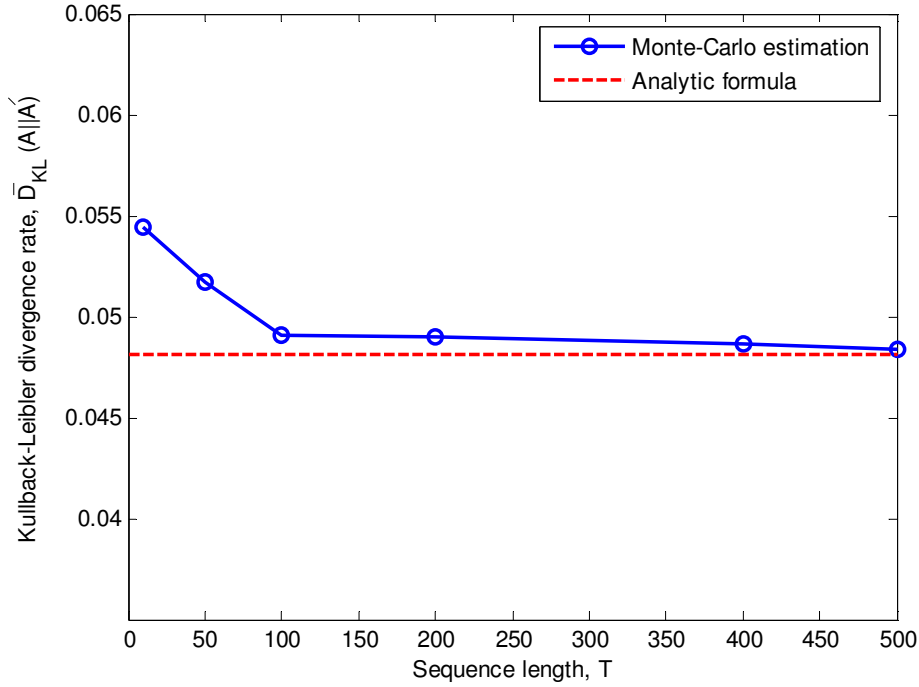


Figure 4.1: KLD rate measurement results for two Markov models given by (4.25).

4.3 Distance Measures Between Hidden Markov Models

For discrete hidden Markov models, the probability mass functions in (4.5) are very complex and requires iterative forward-backward algorithm [31] to compute it. Unlike finite state Markov chains, there is no closed form analytical expression in terms of the model parameters to evaluate the KLD rate between two HMM's.

Let $\mathbf{O} = o_1, o_2, \dots, o_T$ be an observation sequence where $o_t \in \mathbf{V}$, $\mathbf{V} = [v_1, v_2, \dots, v_M]$, represents the observation symbol at time t and let $\mathbf{Q} = q_1, q_2, \dots, q_T$ be the corresponding *hidden* state sequence with $q_t \in \mathbf{S}$, $\mathbf{S} = [s_1, s_2, \dots, s_N]$, be the state at time t . Given the HMM, $\lambda (= \mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, the probability of $\mathbf{O}$, is obtained as

$$P(\mathbf{O} | \lambda) = \sum_{q_1} \sum_{q_2} \dots \sum_{q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \dots a_{q_{T-1} q_T} b_{q_T}(o_T) \quad (4.26)$$

Commonly, the Monte-Carlo method is used to numerically approximate the KLD rate between HMMs [45]. This approach is similar to (4.24), where a set of observation sequences $\mathbf{O}^1, \mathbf{O}^2, \dots, \mathbf{O}^L$ are randomly generated based on $\mathbf{A}$ and $\mathbf{B}$ matrices. Then for

each of the sequences, the probability in (4.26) is calculated using the forward algorithm in terms of the *scaling* constants². Now the KLD rate between the two HMMs of compatible dimension, $\lambda(=\mathbf{A},\mathbf{B},\boldsymbol{\pi})$ and $\lambda'(=\mathbf{A}',\mathbf{B}',\boldsymbol{\pi}')$, is approximated as

$$\bar{D}_{KL}(\lambda \parallel \lambda') \approx \frac{1}{T} \frac{1}{L} \sum_{i=1}^L \left[\log(P(\mathbf{O}^i \mid \lambda)) - \log(P(\mathbf{O}^i \mid \lambda')) \right] \quad (4.27)$$

As discussed before, for an accurate approximation, L must be very large, which may be prohibitively intensive for certain applications. In order to avoid expensive computations, researchers have proposed different distance measures that analytically calculate the dissimilarity between two HMMs [57]-[63]. We will discuss three such distance measures and compare them against the Monte-Carlo estimation to evaluate their performances. Unlike in [61]-[63], these measures are not limited to any particular application or specific topology i.e., left-right model.

4.3.1 Upper Bound of KLD

In [57], Do provides an analytical formula to calculate the upper bound (UB) for the KLD rate between two stationary HMMs. Using the log-sum inequality [64] and the backward algorithm, a simple expression was derived to compute the upper bound.

Let us consider two probability mass functions given by $\mathbf{w}=[w_1, w_2, \dots, w_N]$ and $\mathbf{f}=[f_1, f_2, \dots, f_N]$. The KLD between two mixture densities $\sum_{i=1}^N w_i f_i$ and $\sum_{i=1}^N w'_i f'_i$ is upper bounded by [64]

$$D\left(\sum_{i=1}^N w_i f_i \parallel \sum_{i=1}^N w'_i f'_i\right) \leq D(\mathbf{w} \parallel \mathbf{w}') + \sum_{i=1}^N w_i D(f_i \parallel f'_i) \quad (4.28)$$

where $D(\mathbf{w} \parallel \mathbf{w}') = \sum_{i=1}^N w_i \log \frac{w_i}{w'_i}$. The equality of (4.28) holds only when

$$\frac{w_i f_i}{\sum_{i=1}^N w_i f_i} = \frac{w'_i f'_i}{\sum_{i=1}^N w'_i f'_i} \text{ for all } i.$$

Now let us define the conditional probability of the partial observation sequence $o_{t+1}, o_{t+2}, \dots, o_T$ given the state $q_t = s_i$ at time t , and the model λ , as³

² For details, the reader is referred to Chapter 3, Section 3.2.1.

$$\beta_t(i) = P(o_t, o_{t+1}, \dots, o_T | q_t = s_i, \lambda), \quad 1 \leq i \leq N \quad (4.29)$$

Using (4.28) and the chain rule for KLD, we have [57]

$$D(\beta_t(i) \parallel \beta'_t(i)) = D(\mathbf{b}_i \parallel \mathbf{b}'_i) + D(\mathbf{a}_i \parallel \mathbf{a}'_i) + \sum_{j=1}^N a_{ij} D(\beta_{t+1}(j) \parallel \beta'_{t+1}(j)) \quad (4.30)$$

where $\mathbf{a}_i = [a_{i1}, a_{i2}, \dots, a_{iN}]$ and $\mathbf{b}_i = [b_i(1), b_i(2), \dots, b_i(M)]$.

In the backward algorithm, by applying the termination condition ($t = 1$), we have

$$D_{KL}(\lambda \parallel \lambda') \leq D(\boldsymbol{\pi} \parallel \boldsymbol{\pi}') + \sum_{j=1}^N \pi_j D(\beta_1(j) \parallel \beta'_1(j)) \quad (4.31)$$

Using (4.31), (4.30) and assuming the model λ is stationary, the upper bound for the KLD rate is obtained as

$$\bar{D}_{KL}(\lambda \parallel \lambda') \leq \sum_{i=1}^N p_i (D(\mathbf{a}_i \parallel \mathbf{a}'_i) + D(\mathbf{b}_i \parallel \mathbf{b}'_i)) \quad (4.32)$$

where $\mathbf{p} = [p_1, p_2, \dots, p_N]$ is the stationary distribution of $\mathbf{A}$.

In [58], the authors proposed a distance measure between HMMs that is based on the optimum state sequence, obtained via the Viterbi algorithm [34]. The proposed distance measure in [58] has the same expression as in (4.32). Even though the upper bound given by (4.32) is very simple to calculate, the tightness of the bound was not quantified in [57]. In a later section, we will verify the tightness of the bound using numerical examples.

4.3.2 Probabilistic Distance Measure

Recently, using the probability measure transformation technique, a probabilistic distance measure between discrete time, finite state, HMMs has been proposed [59]. By restricting the probability distributions associated to the HMMs to the sub- σ -algebra generated by the state and observation sequence, a recursive expression was derived to calculate the dissimilarity between HMMs.

Let us consider two HMMs $\lambda (= \mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ and $\lambda' (= \mathbf{A}', \mathbf{B}', \boldsymbol{\pi}')$. We assume that $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\pi}$ are *continuous* with respect to $\mathbf{A}'$, $\mathbf{B}'$ and $\boldsymbol{\pi}'$, respectively. In other words

³ Note that there is a slight difference between the backward variables of the BWA and the variable defined in (4.29).

$a'_{ij} = 0 \Rightarrow a_{ij} = 0$, $b'_j(l) = 0 \Rightarrow b_j(l) = 0$ and $\pi'_i = 0 \Rightarrow \pi_i = 0$, which is compactly denoted by $\lambda \ll \lambda'$. The difference in HMMs, called the *probabilistic distance*, is expressed in terms of the information states as

$$\bar{D}_{prob}(\lambda \parallel \lambda') = \lim_{t \rightarrow \infty} \frac{1}{t} \log \frac{\sum_{i=1}^N \mu_t(i)}{\sum_{i=1}^N \mu'_t(i)} \quad (4.33)$$

where t varies over the observation length, $1 \leq t \leq T$, $\mu_t(i)$ and $\mu'_t(i)$ are normalized information states associate with λ and λ' , respectively, given by the following recursions:

$$\mu_t(i) = \sum_{j=1}^N a_{ij} b_j(o_t) \mu_{t-1}(j) \quad (4.34)$$

$$\mu'_t(i) = \sum_{j=1}^N a'_{ij} b'_j(o_t) \mu'_{t-1}(j) \quad (4.35)$$

where $\mu_t(i)$ and $\mu'_t(i)$ are initialized as $\mu_0(i) = \pi_i$ and $\mu'_0(i) = \pi'_i$, respectively.

In order to evaluate $\bar{D}_{prob}(\lambda \parallel \lambda')$ in (4.33), first an output sequence $\mathbf{O} = o_1, o_2, \dots, o_T$ is realized according to the model λ . Next, for this realization, the samples of the information states at each time instant t are computed based on (4.34) and (4.35). Note that T has to be large for $\bar{D}_{prob}(\lambda \parallel \lambda')$ to converge after initial fluctuations. However, unlike in Monte-Carlo method, only one realization of the observation sequence is needed to compute the distance resulting in huge computational savings.

4.3.3 Heuristic Distance Measure

A heuristic distance measure between HMMs, proposed in [60], shows very close agreement with the Kullback-Leibler divergence rate. Let us consider an observation sequence $\mathbf{O} = o_1, o_2, \dots, o_T$ and the underlying state sequence $\mathbf{Q} = q_1, q_2, \dots, q_T$, generated by the HMM parameter set $\lambda (= \mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$. Let λ' be another HMM with $\lambda \ll \lambda'$. The KLD between λ and λ' is given by

$$D_{KL}(\lambda \parallel \lambda') = \sum_{\mathbf{O} \in \mathbf{V}^T} P(\mathbf{O} \mid \lambda) \log \frac{P(\mathbf{O} \mid \lambda)}{P(\mathbf{O} \mid \lambda')} \quad (4.36)$$

where the observation sequences are defined on $\mathbf{V}^T$ and $P(\mathbf{O} \mid \lambda)$ is given by (4.26).

Combining (4.36) and (4.26), the KLD can be rewritten as

$$D_{KL}(\lambda \parallel \lambda') = \sum_{\mathbf{O} \in \mathbf{V}^T} \sum_{q_1} \dots \sum_{q_T} \pi_{q_1} b_{q_1}(o_1) \dots a_{q_{T-1}q_T} b_{q_T}(o_T) \cdot \log \frac{\sum_{q_1} \dots \sum_{q_T} \pi_{q_1} b_{q_1}(o_1) \dots a_{q_{T-1}q_T} b_{q_T}(o_T)}{\sum_{q_1} \dots \sum_{q_T} \pi'_{q_1} b'_{q_1}(o_1) \dots a'_{q_{T-1}q_T} b'_{q_T}(o_T)} \quad (4.37)$$

As obvious from (4.37), there is no easy way to compute the summation inside the logarithm. In fact, the forward-backward algorithm is usually employed to calculate this, which requires considerable computations. In [60], the authors proposed a slightly modified distance measure given by

$$D_{heu}(\lambda \parallel \lambda') = \sum_{\mathbf{O} \in \mathbf{V}^T} \sum_{q_1} \dots \sum_{q_T} \pi_{q_1} b_{q_1}(o_1) \dots a_{q_{T-1}q_T} b_{q_T}(o_T) \cdot \left[\log \frac{\sum_{q_1} \pi_{q_1} b_{q_1}(o_1)}{\sum_{q_1} \pi'_{q_1} b'_{q_1}(o_1)} + \sum_{k=2}^T \log \frac{\sum_{q_k} a_{q_{k-1}q_k} b_{q_k}(o_k)}{\sum_{q_k} a'_{q_{k-1}q_k} b'_{q_k}(o_k)} \right] \quad (4.38)$$

Note the difference between (4.37) and (4.38), where in the latter the dependency of states in the summation inside the logarithm was ignored.

Now we define

$$m(q_2 \mid q_1 = i) \triangleq \sum_{o \in \mathbf{V}} \left[\sum_{j \in \mathbf{S}} a_{ij} b_j(o) \log \frac{\sum_{j \in \mathbf{S}} a_{ij} b_j(o)}{\sum_{j \in \mathbf{S}} a'_{ij} b'_j(o)} \right], \quad 1 \leq i, j \leq N \quad (4.39)$$

Since $\mathbf{A}$ and $\mathbf{B}$ are stochastic matrices, using the same technique as in [55], the heuristic divergence can be expressed as

$$D_{heu}(\lambda \parallel \lambda') = \boldsymbol{\pi} (\mathbf{I} + \mathbf{A} + \dots + \mathbf{A}^{T-2}) \mathbf{W} + \sum_{o_1 \in \mathbf{V}} \sum_{q_1} \pi_{q_1} b_{q_1}(o_1) \log \frac{\sum_{q_1} \pi_{q_1} b_{q_1}(o_1)}{\sum_{q_1} \pi'_{q_1} b'_{q_1}(o_1)} \quad (4.40)$$

where

$$\mathbf{W} = [m(q_2 \mid q_1 = 1), \dots, m(q_2 \mid q_1 = N)]^t \quad (4.41)$$

Now the heuristic divergence rate is given by

$$\bar{D}_{heu}(\lambda \parallel \lambda') = \boldsymbol{\pi} \mathbf{L} \mathbf{W} \quad (4.42)$$

where $\mathbf{L}$ is defined by (4.18).

Assuming the model λ is stationary, the heuristic divergence rate can be rewritten as

$$\bar{D}_{heu}(\lambda \parallel \lambda') = \mathbf{p} \mathbf{W} \quad (4.43)$$

where $\mathbf{p}$ is the stationary vector of $\mathbf{A}$.

Using the results for non-stationary, irreducible transition matrices in [55], the heuristic divergence rate can be extended to discriminate non-stationary HMMs as well. Also note that, when $\mathbf{B}$ and $\mathbf{B}'$ are identity matrices, the HMMs become first-order Markov chains and $\bar{D}_{heu}(\lambda \parallel \lambda')$ reduces to the KLD rate of the first order Markov chain given by (4.19).

4.3.4 Numerical Examples

In this section the performance of different distance measures are compared against the Monte-Carlo simulation method to verify their accuracy. The Monte-Carlo estimation, the upper bound (UB) of the KLD rate, the probabilistic distance and the heuristic distance between HMMs are calculated according to (4.27), (4.32), (4.33) and (4.43), respectively. For all calculations associated with distance measurement, we used the natural logarithm. In the Monte-Carlo method, for each sequence length, the KLD is calculated averaging over 1000 independent trials. In the first example, we consider two 3-state HMM's with 2 observation symbols/state. The parameters of the two models λ and λ' are randomly generated and given by

$$\begin{aligned} \boldsymbol{\pi} &= [0.36 \quad 0.40 \quad 0.24], \quad \boldsymbol{\pi}' = [0.39 \quad 0.23 \quad 0.38] \\ \mathbf{A} &= \begin{bmatrix} 0.13 & 0.37 & 0.5 \\ 0.06 & 0.73 & 0.21 \\ 0.45 & 0.09 & 0.46 \end{bmatrix}, \quad \mathbf{A}' = \begin{bmatrix} 0.60 & 0.17 & 0.24 \\ 0.41 & 0.30 & 0.29 \\ 0.28 & 0.51 & 0.21 \end{bmatrix} \\ \mathbf{B} &= \begin{bmatrix} 0.02 & 0.61 & 0.50 \\ 0.98 & 0.39 & 0.50 \end{bmatrix}, \quad \mathbf{B}' = \begin{bmatrix} 0.96 & 0.69 & 0.57 \\ 0.04 & 0.31 & 0.43 \end{bmatrix} \end{aligned} \quad (4.44)$$

The divergence rate measurement results are shown in Figure 4.2. The results of the Monte-Carlo method converges as the length of the randomly generated sequence becomes sufficiently large ($T > 500$). From Figure 4.2, it is obvious that the UB of the KLD rate is very loose. Both the probabilistic distance and heuristic divergence closely

approximate the Monte-Carlo estimation. With the probabilistic distance, as expected, the length of the sequence has to be large before it converges to Monte-Carlo estimation.

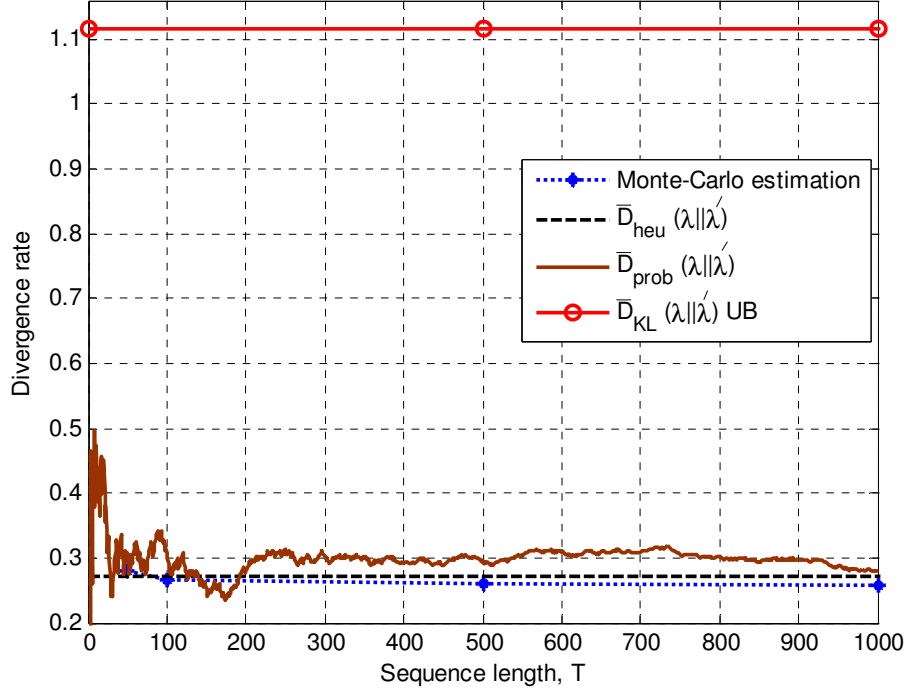


Figure 4.2: Divergence rate measurement results for two HMM's given by (4.44).

For the second example, we consider two 2-state HMM's with 3 observation symbols/state. Once again, we randomly generate the model parameters. They are given by

$$\begin{aligned}
 \boldsymbol{\pi} &= [0.16 \quad 0.84], \quad \boldsymbol{\pi}' = [0.57 \quad 0.43] \\
 \mathbf{A} &= \begin{bmatrix} 0.48 & 0.52 \\ 0.57 & 0.43 \end{bmatrix}, \quad \mathbf{A}' = \begin{bmatrix} 0.46 & 0.54 \\ 0.99 & 0.01 \end{bmatrix} \\
 \mathbf{B} &= \begin{bmatrix} 0.41 & 0.60 \\ 0.22 & 0.07 \\ 0.37 & 0.33 \end{bmatrix}, \quad \mathbf{B}' = \begin{bmatrix} 0.35 & 0.17 \\ 0.21 & 0.25 \\ 0.44 & 0.58 \end{bmatrix}
 \end{aligned} \tag{4.45}$$

The divergence rate results for these two models are shown in Figure 4.3. The results follow the same pattern as before.

Based on these results, we conclude that $\bar{D}_{heu}(\lambda \parallel \lambda')$ provides a good approximation of the KLD rate between HMMs while requiring only $O(N^2)$ computations. This simple analytic formula based on model parameters is attractive

especially in practical applications that require real-time model classification. Unlike the probabilistic distance method, the heuristic formula does not require any realization of the observation sequence. Thus $\bar{D}_{heu}(\lambda \parallel \lambda')$ is a deterministic number and there is no “randomness” involved in the calculations. Also, note that in the formulation of the heuristic expression we did not impose any constraint on the Markov chain structure.

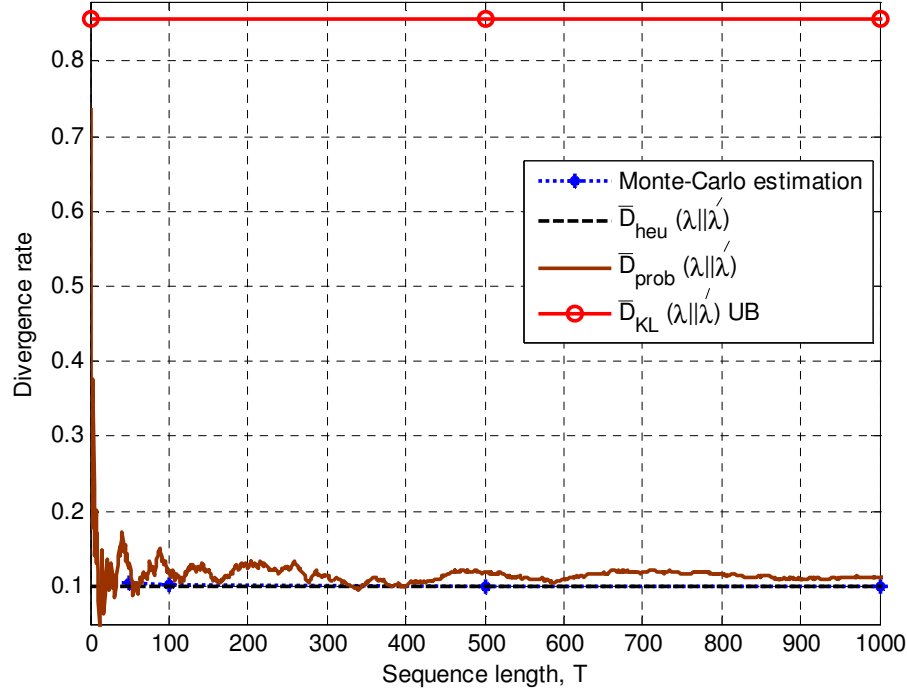


Figure 4.3: Divergence rate measurement results for two HMM's given by (4.45).

4.4 Parametric Analysis of HMM

Often times it is reported that an HMM, defined by a given number of states with a given set of observation symbols, can successfully classify and/or model the observation sequences without explicitly mentioning the model parameters, $\lambda (= \mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$. The number of states and observation symbols may be application specific and are usually determined heuristically. However, as the information regarding the exact model parameters is not available, in most cases it becomes difficult to analyze the performance and efficiency of that particular HMM in question. For example, an HMM can excel in classification simply due to the discriminating observation symbols ($\mathbf{B}$ matrix) or

because of a very strong memory model ($\mathbf{A}$ matrix). Thus parametric analysis of the HMM not only helps us evaluate its performance but also facilitates optimization of the model.

As discussed in Chapter 3, the forward and backward recursions in the Baum-Welch algorithm are at the heart of the HMM parameter estimation procedure. Specifically

$$\alpha_{t+1}(j) = \sum_{i=1}^N \alpha_t(i) a_{ij} b_j(o_{t+1}), \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (4.46)$$

and

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad 1 \leq j \leq N, \quad 1 \leq t \leq T-1 \quad (4.47)$$

In both recursions, the product term $a_{ij} b_j(o_{t+1})$ plays an important part and can be described the by compact product matrix

$$\mathbf{G} = \mathbf{B}\mathbf{A} \quad (4.48)$$

The significance of the product matrix has been noted previously in [66], [67]. In [66], the product matrix was used to define the model equivalence and minimal models.

A couple of observations can be readily made based on the $\mathbf{A}$ and $\mathbf{B}$ matrices. For example, if $\mathbf{A}$ is a rank one matrix then the unique steady-state distribution, $\mathbf{p}$, is the same as the rows of $\mathbf{A}$

$$\sum_{j=1}^N b_j(l) a_{ij} = \sum_{j=1}^N b_j(l) \pi_j \quad (4.49)$$

For a Markov chain, the distance between the current state transition probability and the steady-state probability is usually defined as a function of the left *residual* eigenvalue [68]. The residual eigenvalue is defined as the sum of squared sub dominant eigenvalues of $\mathbf{A}$. Since in a *regular*⁴ Markov chain, there is only one unity eigenvalue, the residual eigenvalue is given by

$$\mu_{res} = \sum_{i=2}^N \mu_i^2 \quad (4.50)$$

⁴ A Markov chain, denoted by $\mathbf{A}$, is called *regular* if its characteristic polynomial $\det(\mathbf{I}z - \mathbf{A})$ has only one root with unity absolute value, $|z| = 1$.

where μ_i is the i -th eigenvalue of $\mathbf{A}$ and $|\mu_1|=1$. In addition, μ_{res} is a measure of the ergodicity of the model, which is directly related to the generation of variable state sequences as a function of the observation symbols.

Similarly, the inference of a specific state from observation symbols decreases as the columns of $\mathbf{B}$ becomes more correlated. Consequently, if $\mathbf{B}$ has only one non-zero eigenvalue, the observations do not provide any information about the state sequence at all! Note that when $\mathbf{B}$ is close to an *identity* matrix there is a strong one-to-one correlation between the states and the observations symbols, which essentially reduces the HMM to an ordinary first order Markov chain. Thus the composition of $\mathbf{A}$ and $\mathbf{B}$ is important in evaluating the overall performance of the HMMs.

4.4.1 Augmented Matrix

Even though the product matrix, $\mathbf{G}$, is important in analyzing the performance of HMMs, we explore the characteristics of a composite matrix that is constructed by augmenting the rows of $\mathbf{A}$ and $\mathbf{B}^t$ [67]

$$\mathbf{C}_{N \times (N+M)} = \mathbf{A} | \mathbf{B}^t \quad (4.51)$$

where $(\cdot)^t$ denotes the transposition of the original quantity.

Unlike the product matrix, $\mathbf{G}$, in the augmented matrix, $\mathbf{C}$, each row provides a complete description of a given state in terms of state transition and observation symbol probabilities. It is intuitive that linearly independent rows of $\mathbf{A}$ and $\mathbf{B}^t$ provide maximum discrimination of states and observation symbols, respectively. In order to minimize the number of states and observation symbol ambiguity, the rows of $\mathbf{C}$ have to be as linearly independent as possible. The dependency between the rows can be quantified in terms the inverse condition number (ICN), γ [67]

$$\gamma = \frac{\sigma_{\min}}{\sigma_{\max}} \quad (4.52)$$

where $\sigma_{\max}$ ($\sigma_{\min}$) is the maximum (minimum) singular value of $\mathbf{C}$. A well-conditioned matrix has unity ICN indicating all states can be realized from the observation sequences.

However, an ill-conditioned matrix with $\gamma=0$ does not provide any predictive information about the state sequence.

4.4.2 Residual Matrix

The ICN of $\mathbf{C}$ indicates how well the observation sequences are mapped into the model space spanned by the model parameter set. In other words, it is an indicator of how well the HMM is likely to span the full dimensionality of the model parameter space. However, it does not locate the problematic state or observation symbol that is instrumental in the rank deficiency of $\mathbf{C}$. Recently, the authors in [67] have proposed a new matrix that plays a key role in identifying the rank deficiency.

Let $\mathbf{C}_{\setminus i}$ be the $(N-1) \times (N+M)$ matrix with i -th row ($\mathbf{u}_i$) removed. Let $\mathbf{Q}_{\setminus i}$ be the projection matrix onto the space orthogonal to the space spanned by the rows of $\mathbf{C}_{\setminus i}$ and is given by [4]

$$\mathbf{Q}_{\setminus i} = \mathbf{I} - \mathbf{C}_{\setminus i}' (\mathbf{C}_{\setminus i}')^\dagger \quad (4.53)$$

where $(\cdot)^\dagger$ denotes the Penrose-Moore pseudo inverse of the original quantity⁵. The projection of the i -th row onto the orthogonal complement of the range space of $\mathbf{C}_{\setminus i}'$ is called the *residual vector* and is given by

$$\mathbf{e}_i = \mathbf{Q}_{\setminus i} \mathbf{u}_i' \quad (4.54)$$

Now the *residual matrix* is defined as

$$\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_N]' \quad (4.55)$$

which defines the distribution of parameter dependency and can be used to isolate the problematic states and observation symbols. For example, if a particular element of $\mathbf{E}$ is close to 0, then the corresponding HMM parameter is linearly dependent on other parameters of the model. On the other hand, if the element is close to the original element, then it is independent of other rows and is important for the model performance.

⁵ Pseudo inverse of a full column rank matrix $\mathbf{X}$ is given by $\mathbf{X}^\dagger = (\mathbf{X}^H \mathbf{X})^{-1} \mathbf{X}^H$, where $\mathbf{X}^H$ denotes the Hermitian of $\mathbf{X}$.

Given the residual matrix, $\mathbf{E} = \{e_{ij}\}_{N \times N+M}$, a compact measure, called the *residual sum vector*, is defined as

$$\mathbf{R} = [\underbrace{r_1, r_2, \dots, r_N}_{\text{associated with states}}, \underbrace{r_{N+1}, \dots, r_{N+M}}_{\text{associated with symbols}}] \quad (4.56)$$

where

$$r_j = \sqrt{\sum_{i=1}^N e_{ij}^2}, \quad 1 \leq j \leq N+M \quad (4.57)$$

A value of r_j close to unity indicates the state or observation symbol j is independent of other states or symbols and hence plays a significant role in the overall performance. However, if r_j is close to zero then either the state or symbol is highly dependent on other parameters or the occurrence of that state or symbol is rare.

4.4.3 Example

In this section we present a numerical example to illustrate the concept of parameter analysis. Let us consider the problem of modeling the frame error statistics for a given wireless channel. We start arbitrarily with a 5-state model, i.e., $N = 5$ and $M = 2$, since the observation symbols in a frame error sequence is either 0 (good frame) or 1 (bad frame). For a given set of channel parameters, after training with the frame error sequence, the following HMM parameter set results:

$$\mathbf{A} = \begin{bmatrix} 0.0410 & 0.0608 & 0.3107 & 0.0001 & 0.5874 \\ 0.2864 & 0.4028 & 0.1199 & 0.0002 & 0.1906 \\ 0.0106 & 0.4397 & 0.1695 & 0.0003 & 0.3799 \\ 0.0001 & 0.0019 & 0.0045 & 0.9932 & 0.0003 \\ 0.3076 & 0.1619 & 0.0431 & 0.0001 & 0.4874 \end{bmatrix} \quad (4.58)$$

$$\mathbf{B} = \begin{bmatrix} 0.9926 & 0.9928 & 0.9737 & 0.4059 & 0.9965 \\ 0.0074 & 0.0072 & 0.0263 & 0.5941 & 0.0035 \end{bmatrix}$$

Since number of observation symbols is fixed in this specific example, we will only investigate the number of states in the model. After forming the augmented matrix and applying (4.52), the inverse condition number (ICN) is calculated to be 0.0562. Obviously this is a low number which indicates possible correlation and redundancies

among the states. To be more specific about the problematic states, we find the residual sum vector using (4.56). This is given by

$$\mathbf{R} = [0.1610 \quad 0.2141 \quad 0.3372 \quad 0.9933 \quad 0.2306 \quad 0.1038 \quad 0.5918] \quad (4.59)$$

As seen from (4.59), the first two states are dependent on other states and may be deleted to optimize the model. After retraining the HMM with a 3-state model we have the following set of parameters:

$$\mathbf{A} = \begin{bmatrix} 0.9935 & 0.0049 & 0.0016 \\ 0.0006 & 0.3544 & 0.6450 \\ 0.0001 & 0.0714 & 0.9285 \end{bmatrix} \quad (4.60)$$

$$\mathbf{B} = \begin{bmatrix} 0.4073 & 0.9175 & 0.9999 \\ 0.5927 & 0.0825 & 0.0001 \end{bmatrix}$$

For this parameter set, the ICN is 0.1401 and the residual sum vector is given by

$$\mathbf{R} = [0.9960 \quad 0.4983 \quad 0.2566 \quad 0.1441 \quad 0.5367] \quad (4.61)$$

It is apparent that after reducing the number of states the ICN has improved and the dependency among different states also decreases. It also implies that a three state model is sufficient to extract the statistical features in the frame error sequences in question and adding more states does not necessarily improve performance.

Since the optimization technique should not alter the statistics of the model, as a sanity check, we need to verify the statistical similarity between the two models given by (4.58) and (4.60). Using (4.27), the Kullback-Leibler divergence (KLD) between the two models was found to be 0.0022. This small divergence number indicates that the second model, though smaller, is statistically equivalent to the first model and is capable of producing similar statistics.

4.5 Conclusion

In this chapter the distance measures for Markov chains and hidden Markov models were discussed. While for Markov models there is an analytical expression to calculate the KLD rate, HMMs do not have any such closed form solution. Different divergence measures for HMMs were compared against the commonly used Monte-Carlo method. The proposed heuristic divergence measure closely approximates the Monte-Carlo

method, while requiring minimal computations. Also it does not require realization of the observation sequence, nor does it impose any structural constraints on the model. The heuristic method is especially attractive in applications that require online calculations of divergence. The distance measures discussed here will be used extensively in the classification problems later on. Using matrix theory, the HMM parameters were analyzed to optimize the model structure. Based on this concept, a diagnostic tool can be developed to verify the quality of the model.

Chapter 5: 1xEV-DV – An Overview of Cellular Voice and Packet Data System

The cellular communications industry has experienced explosive growth over the last decade. The rapid worldwide growth in wireless communications enabled people and businesses to operate efficiently and cost-effectively through the use of always-on, always-connected applications. The widespread success of cellular systems has led to the development of newer techniques and standards to accommodate diverse applications namely voice, high speed data, and multimedia services. This chapter begins with a brief discussion of the evolution of cellular systems. Key features and capabilities of cellular standards from different generations are presented. The important aspects of 1xEV-DV system, a Code Division Multiple Access (CDMA) based third generation standard, are explained. Next, Motorola's 1xEV-DV simulator [77] is introduced which is extensively used in this research. The system level performance of 1xEV-DV is also presented.

5.1 Cellular Standards

In the late 70s, AT&T Bell Laboratories developed the first US cellular telephone system called the first generation (1G) system. Based on the analog technology, 1G system was first employed in Chicago in the 800 MHz band. The second generation (2G) came into existence in the early 90s and was implemented by digital technology. While 1G was dedicated solely to support voice communications, 2G was designed to support voice and low-rate data. Additionally, the intent of 2G and its enhanced version i.e., 2.5G, was to improve voice quality, coverage, and capacity. Third generation (3G) systems mark significant advances, both in applications and capacity, compared to the second

generation standards. The goal of 3G systems was to further increase the system capacity and provide multimedia based services along with voice and high-speed data.

The two most prominent bodies working in the area of 3G standardization are third generation partnership project (3GPP) and third generation partnership project 2 (3GPP2). The standardization work for Universal Mobile Telecommunication System (UMTS) is being carried-out under the supervision of 3GPP. UMTS is an upgrade from 2G Global System for Mobile communications (GSM). UMTS is the likely to dominate Europe, Pacific and some part of Asia, and has been touted as the successor to GSM. On the other hand, the standardization work for Code Division Multiple Access (CDMA) and its evolution is being carried out under the auspices of 3GPP2. CDMA based systems have stronghold in the North American and Asian markets. In this chapter we will focus on the CDMA based standards. For the GSM based standards the reader is referred to [69]-[72].

5.1.1 First Generation

The Advanced Mobile Phone System (AMPS) is the most notable of the first generation systems. Other first generation systems include Narrowband AMPS (NAMPS), Nordic Mobile Telephone System (NMT-900) and Total Access Cellular System (TACS). The following are the important features of this standard

- Employs analog Frequency Modulation (FM) for voice and digital signaling for control information.
- Uses Frequency Division Multiple Access (FDMA) as the multiple access mode.
- Networks connections are based on dedicated circuit-switched techniques.

5.1.2 Second Generation

The evolution toward the second generation cellular systems can be attributed to the rapid growth of cellular subscribers, as 2G promised improved voice quality, more capacity and better coverage. 2G systems also support data services along with voice with very

limited Internet browsing capability. 2G cellular systems include Code Division Multiple Access (CDMA) based IS-95 or cdmaOne, United States Digital Cellular (USDC) standards IS-54 and IS-136, Global System for Mobile communications (GSM), Pacific Digital Cellular (PDC), etc. The key aspects of IS-95 are:

- Uses digital modulation schemes i.e., Binary Phase Shift Keying (BPSK).
- Employs digital compression and error correction coding techniques like convolutional codes.
- Uses Code Division Multiple Access (CDMA) as the multiple access scheme.
- Uses radio channels with 1.25 MHz of bandwidth.
- Uses network connections based on circuit-switched techniques supporting a peak data rate of 14.4 Kbps.

5.1.3 Migration to Third Generation

Since 2G technologies use circuit-switched network connections, the data rates are limited to a single user voice channel, which is on the order of 10 Kbps. This rate is too low for e-mail or Internet browsing applications. In order to make the existing 2G standards compatible with higher data rate applications, new data-oriented standards have been developed that can be overlaid upon the existing 2G systems. These new standards, called 2.5G, support higher data rate communication for e-mail, web browsing, etc. The GSM based 2.5G systems include High Speed Circuit Switched Data (HSCSD), General Packet Radio Service (GPRS), Enhanced Data Rate for GSM Evolution (EDGE). Unlike GSM, there is only one CDMA based 2.5G system – IS-95B.

IS-95B is the evolved version of IS-95A, supported by an enhanced MAC layer, provides higher speed data services. Note that the physical layer is same for both IS-95 and IS-95B. By employing additional Walsh codes and PN sequence masks, a dedicated user may be assigned up to eight code channels simultaneously and in parallel for higher data rates (8×14.4 Kbps). The additional features of IS-95B are

- Supports peak data rate of up to 115 Kbps.

- Introduces mobile assisted handoff where mobile units search for different channels without instruction from the Base Transceiver Station (BTS). Thus handoff procedure is faster and more efficient.

5.1.4 Third Generation

Third generation (3G) cellular standards have been developed promising wireless access in ways that have never been possible before. High speed Internet access, communications using Voice over Internet Protocol (VoIP), multimedia, unparalleled network capacity are the salient features of 3G that mark the advent of a new era in cellular industry. The two most popular 3G standards are GSM based Wideband CDMA (WCDMA) and CDMA based cdma2000 1xRTT¹.

cdma2000 is a natural progression of IS-95. The upgrade from IS-95 to cdma2000 only needs new software and channel cards at the BTS, but does not require any additional Radio Frequency (RF) equipment. The following are the key aspects of this standard [73]-[76]

- Supports data rates of up to 614 Kbps, with typical throughput rates of up to 144 Kbps.
- Introduces Radio Configurations (RC) – physical layer parameters such as data rates, modulation scheme, and spreading gain are compactly characterized by transmission formats.
- Convolutional and Turbo coding techniques used as forward error correction (FEC) scheme.
- Uses Quality and Erasure indicator bits (QIB and EIB) on the reverse power control channel. These inform the BTS about bad frames received at the mobile station (MS).
- Improves battery life by introducing a new paging channel.
- Can accommodate both Time Division Duplexing (TDD) and Frequency Division Duplexing (FDD) modes.

¹ 1x implies one times the original cdmaOne channel bandwidth which is 1.25 MHz and RTT stands for Radio Transmission Technology.

- Uses Quadrature Phase Shift Keying (QPSK) as the modulation technique.

5.1.5 Third Generation Evolutions

Originally developed by Qualcomm, cdma2000 1xEV is an evolutionary advancement of the 3G standard. cdma2000 1xEV provides the option of installing radio channels with *data only* (cdma2000 1xEV-DO) or with *data and voice* (cdma2000 1xEV-DV). cdma2000 1xEV technology can be overlaid upon the existing cdma2000 systems where individual 1.25 MHz channels may be allocated to support high speed packet data within selected cells.

1xEV-DO, also known as High Data Rate (HDR), is the first phase of cdma2000 evolution. 1xEV-DO uses separate channels for voice and data as it avoids difficult load-balancing task. The official air interface standard for 1xEV-DO is specified in IS-856 by 3GPP2. The following are the prominent features of this standard

- Supports data rates of up to 2.45 Mbps in the forward link (BTS to MS) and 153 Kbps in the reverse link (MS to BTS). This type of asymmetric data rates are particularly useful in applications like file downloading from the Internet.
- Not backward-compatible with cdma2000.
- Provides adaptive modulation and coding.
- Based on channel conditions, provide adaptive rate operation.
- Includes two inter-operable modes; an integrated 1x mode optimized for voice and medium data rates, and a 1xEV mode optimized for non real-time high capacity/high speed data and Internet access

The ultimate solution for CDMA relies on multicarrier (MC) technique that joins contiguous cdma2000 channels together for increased bandwidth. cdma2000 3xRTT uses three adjacent 1.25 MHz channels to provide data throughput rate up to 2 Mbps. Likewise three non-adjacent channels may be operated simultaneously and in parallel where no new RF hardware is required at the BTS.

Even though 1xEV-DO is tailored to support high data rates, it requires a different RF channel to carry voice traffic. Also, it does not support real time multimedia services.

Another 3G evolution that overcomes these limitations, called cdma2000 1xEV-DV, which will be introduced next.

5.2 cdma2000 1xEV-DV

1xEV-DV is the second phase of cdma2000 evolution that offers complete backward compatibility with 2G standards. 1xEV-DV systems provide much higher peak data rates compared to 2G standards by employing a number of optimizations in modulation, coding and encryption schemes. Like 1xEV-DO, DV also supports asymmetric data rates. While the peak data rate for 1xEV-DV is 3.09 Mbps in the forward direction, the reverse link can only support 1.85 Mbps. A comparison of peak data rates for different standards is shown in Figure 5.1. Note that actual data rates experienced by users in these systems are usually much lower than peak rates mainly due to fading, presence of other users, interference, etc.

The detailed description of the requirements for 1xEV-DV has been presented in [73]. Some of the major requirements are given here

- At least 2.4 Mbps on the forward channel when serving only packet data traffic for any user in an outdoor, high speed vehicular environment.
- At least 1.25 Mbps on the reverse channel when serving only packet data traffic for any user in an outdoor, high speed vehicular environment.
- 1xEV-DV shall operate with 3xRTT radio configurations.
- Handoff of voice and data services between a 1xEV-DV radio channel and another radio channel that are operating in accordance with the cdma2000 family of specifications.
- If a mobile station does support both 1xEV-DV and 1xEV-DO, then the mobile station shall support handoff of commonly supported data services between the 1xEV-DO and 1xEV-DV radio channels.
- Multiple, concurrent packet data sessions per user.

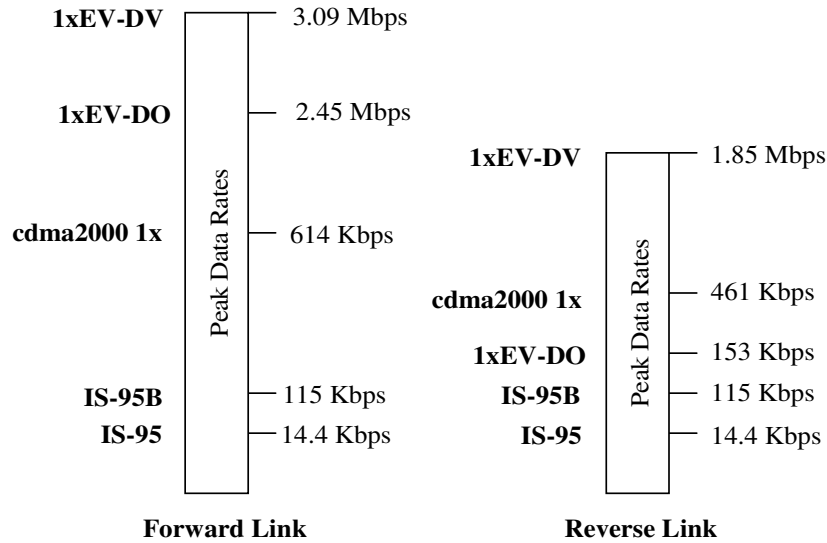


Figure 5.1: Comparing peak data rates for different CDMA standards.

5.2.1 Air Interface Layer Architecture

There are primarily three layers in the 1xEV-DV air interface architecture. These three layers are part of the Open System Interface (OSI) architecture and are shown in Figure 5.2. Layer 1 (L1) is the physical layer which is responsible for over-the-air functions i.e., modulation, coding etc. Layer 2 (L2) consists of Medium Access Control (MAC) and Link Access Control (LAC) sub-layers. The LAC sub-layer implements a data link protocol that provides correct delivery of signaling messages generated by Layer 3 (L3). Its operations include authentication and message integrity, automatic repeat request (ARQ), segmentation and reassembly. The MAC sub-layer implements the medium access protocol and is responsible for the delivery of the LAC protocol data units. It handles best effort delivery of user traffic for data services through Radio Link Protocol (RLP). Layers above the MAC sub-layer consists of several upper layers including a signaling layer that sets up voice and packet data sessions. It also includes data service protocols like Point-to-Point Protocol (PPP), Transmission Control Protocol (TCP) and Internet Protocol (IP).

All the layers mentioned above are present in 2G systems as well. The key addition to the air interface architecture in 1xEV-DV is the Packet Data Channel (PDCH) control

function that handles the Forward and Reverse EV-DV channels. As illustrated in Figure 5.2, this control function resides at layer 2 between the MAC and physical layer and is responsible for controlling the operations of all the new channels introduced in 1xEV-DV systems. If only the services of 1x standard are required, the PDCH control function is bypassed with the rest of the layers remaining the same.

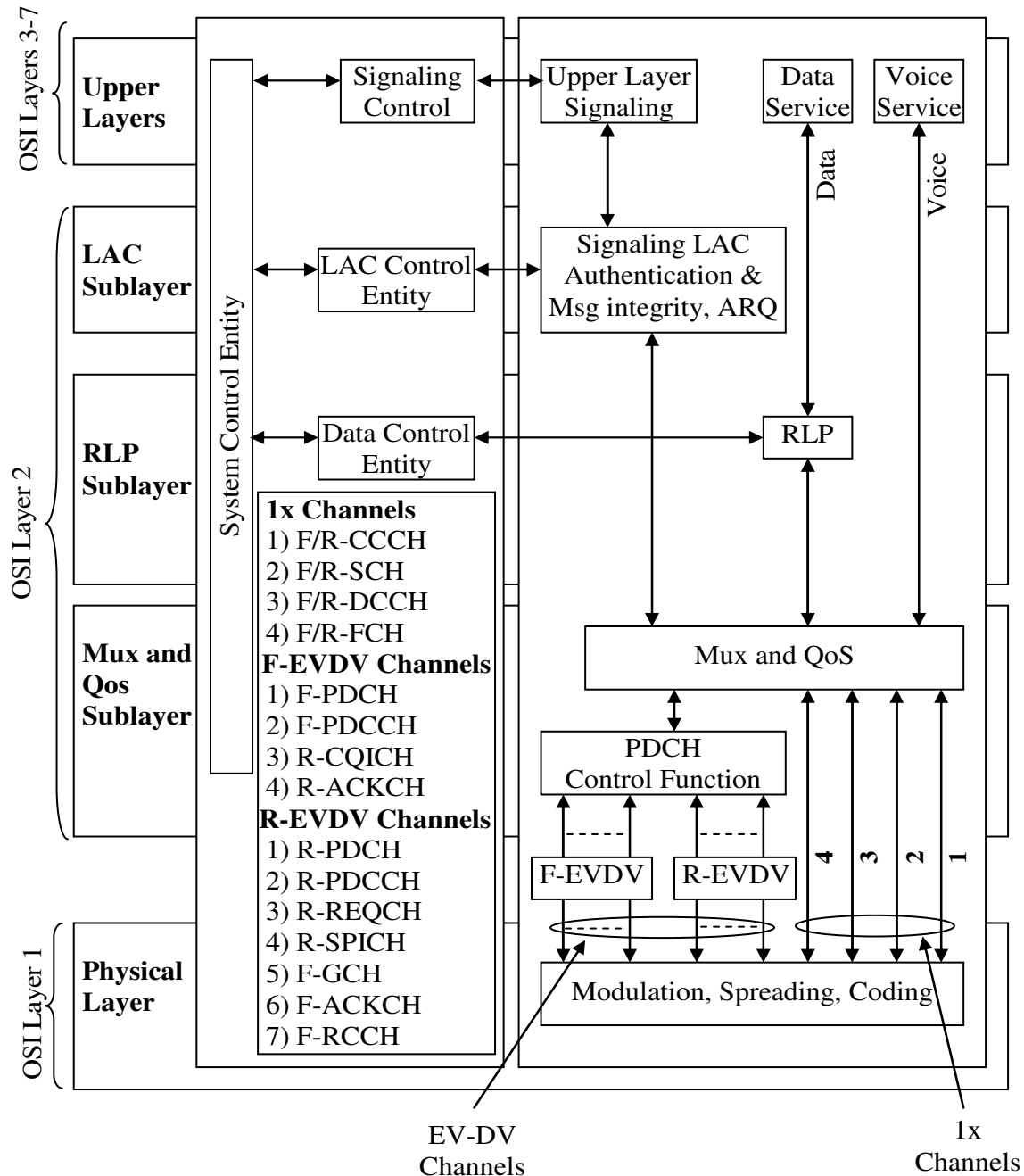


Figure 5.2: cdma2000 1xEV-DV architecture.

5.2.2 Link Adaptation

Due to weather condition, terrain, mobility of the user, etc. the radio environment in a wireless system constantly change. In link adaptation, the receiver reports back the current radio conditions to the transmitter so that the transmitter can exploit the environment by selecting the best set of parameters such as modulation, coding, repetition rate etc. There are mainly two ways a CDMA system can adjust to the changing radio conditions, namely power control and data rate control.

To optimize the system performance, the CDMA system usually adapt the transmit power to keep the inference level as low as possible. For example, in cdma2000 the transmit power is adjusted every 1.25 ms based on the power control commands. However, this type of power control is suitable only for low data rate services like voice. In contrast, 1xEV-DO systems take care of the link adaptation by changing the data rates while keeping the total transmit power fixed. This approach is particularly useful for non-real time data applications. Since 1xEV-DV supports both voice and real-time high rate data services simultaneously, it combines both the above mentioned power control philosophies. For voice and medium data rate services, 1xEV-DV adjusts the transmit power using the power control commands. For high speed data services, 1xEV-DV allocates the remaining available transmit power to a single user at any given time and adjusts the data rate based on the link conditions. The power control scheme in 1xEV-DV is explained in Figure 5.3. Note that the entire Forward Packet Data Channel (F-PDCH) is assigned to only one user using Time Division Multiplexing (TDM) approach. The time periods in the TDM mode are 1.25, 2.5 and 5 ms in the forward direction and 10 ms in the reverse direction. In addition, by using multi-code spreading (taking multiple Walsh codes from the available code space) the throughput to that user is increased.

The BTS receives radio condition reports from all the mobiles that are being served on the PDCH. Based on these reports, the scheduler selects a particular user for a given number of time slots. The modulation and coding schemes are selected adaptively at that time based on some scheduling criteria. QPSK, 8-PSK and 16-QAM are the modulation schemes available in 1xEV-DV forward link while BPSK, QPSK and 8-PSK are available in the reverse link. Also, Turbo encoding is mandatory for both forward and

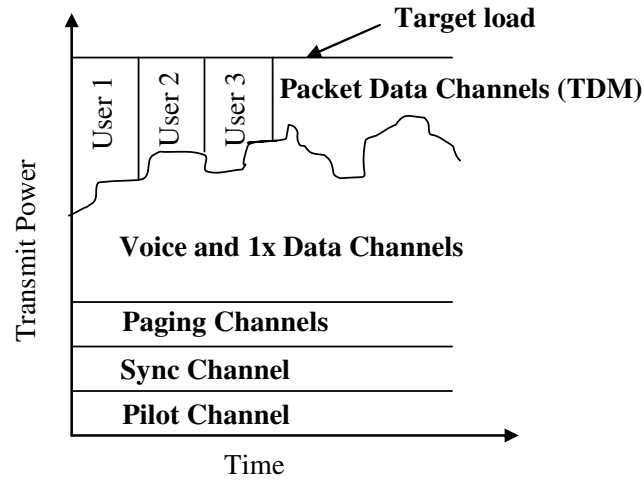


Figure 5.3: Forward link power allocation in 1xEV-DV.

reverse PDCH. It provides better error protection compared to convolutional encoding used in cdma2000.

1xEV-DV systems do not support sort handoffs. Using Fast Cell Site Switching (FCSS), the mobile terminal is always served by the best sector in its *active set*, which is periodically updated by the mobile. The selection of the best sector increases system capacity as well as reduces interference. In both forward and reverse link, Hybrid ARQ (H-ARQ) is used to improve latency of the system. H-ARQ transmits encoded symbols incrementally from the transmitter. The receiver performs incremental redundancy and combines all the symbols and successfully decodes the original bits.

5.2.3 Forward EV-DV Channels

1xEV-DV supports all the forward channels of cdma2000. In addition, there are several new channels added in the forward link to support high data rate services. These channels are as follows:

Forward Packet Data Channel (F-PDCH): F-PDCH is a time division multiplexed channel that is used to send high speed data from the BTS to the MS. A forward CDMA channel may contain up to 2 F-PDCH and each F-PDCH transmits to one specific mobile at a time. Based on the scheduler, the transmission on this channel can have variable time slot of 1.25 ms, 2.5ms or 5ms long.

Forward Packet Data Control Channel (F-PDCCH): F-PDCCH is used to transmit the control information associated with F-PDCH. There are two different types of PDCCHs, namely Forward Primary Packet Data Control Channel (F-PPDCCH) and Forward Secondary Packet Data Control Channel (F-SPDCCH). F-PPDCCH indicates the duration of F-PDCH, while FSPDCCH is used to send the scheduled user's MAC ID (8-bit identifier to uniquely identify the MS on the shared PDCCH), encoder packet size etc.

Reverse Channel Quality Indication Channel (R-CQICH): At every 1.25 ms, R-CQICH carries feedback information from the MS about the channel condition for F-PDCH. BTS uses this information to determine the data rate for that mobile. R-CQICH carries either a full CQI or a differential CQI information. Full CQI information is a signal-to-noise ratio of the pilot, while differential CQI information is a positive (up) or negative (down) increment to the most recently transmitted value.

Reverse Acknowledgement Channel (R-ACKCH): Based on the status of decoding of packets sent on F-PDCH, the mobile sends ACK/NAK over R-ACKCH back to the BTS.

5.2.4 Reverse EV-DV Channels

Like forward link, 1xEV-DV supports all the reverse channels in 1x standards. The additional reverse link channels are as follows:

Reverse Packet Data Channel (R-PDCH): R-PDCH is a time division multiplexed channel that is used to transmit high speed packet data in the reverse link. Unlike F-PDCH, there is only one time slot in the reverse direction which is 10 ms.

Reverse Packet Data Control Channel (R-PDCCH): R-PDCCH is a control channel associated with R-PDCH. It carries all the control information i.e., packet size, to the BTS so that it can decode R-PDCH correctly.

Reverse Request Channel (R-REQCH): R-REQCH carries the request to use R-PDCH from the mobile to the BTS. When the mobile wants to use R-PDCH, it sends the buffer size, channel condition etc. to the BTS through this channel.

Reverse Secondary Pilot Channel (R-SPICH): If the data rate in R-PDCH exceeds a pre-defined rate, mobile uses this channel to transmit a secondary pilot that improves coherent detection.

Forward Grant Channel (F-GCH): When a mobile sends a request to use R-PDCH via R-REQCH, the BTS responds back on F-GCH specifying the allowable data rate for transmission.

Forward Acknowledgement Channel (F-ACKCH): In order to support H-ARQ in the reverse link, the BTS sends ACK/NAK on the F-ACKCH after decoding the R-PDCH.

Forward Rate Control Channel (F-RCCH): The transmission data rates are sent through this channel by the BTS to the mobile using control commands (Up, Down or Hold).

5.2.5 Key 1xEV-DV Features

The following are some of the additional features of 1xEV-DV compared to the 2G standard of cdma2000.

- Forward peak data rate of 3.072 Mbps
- Reverse peak data rate of 451.2 Kbps
- Adaptive modulation (QPSK, 8-PSK, 16-QAM) and rate 1/5 turbo coding in real time to adapt to the time varying RF environment.
- Improved latency with the use of H-ARQ at the physical layer.
- Variable RF frame duration (1.25, 2.5 or 5 ms) for PDCH in the forward link.
- No soft handoff on Forward PDCH. Using FCSS, the best sector always serves the mobile.
- PDCH is shared between users using time division multiplexing with full available power.

5.3 1xEV-DV System Simulation

Evaluating system level performance of radio access network (RAN) has been approached through analytical and simulation methods. Analytical methods become intractable when trying to evaluate intricate real systems, capable of supporting high speed voice/data services. In this section we introduce Motorola's Network Extensible

Simulator (Nextsim), which supports an end-to-end simulation of 1xEV-DV RAN. The simulator includes a physical layer, MAC, LAC, session control etc. It is capable of generating different types of traffic that includes voice and packet data. In this research we extensively used Nextsim to characterize performance of 1xEV-DV. Next the simulation architecture will be discussed.

5.3.1 Simulation Architecture

Determining the system level performance of 1xEV-DV requires a dynamic simulation tool that accurately models protocol execution, feedback loops, signal latency and random packet arrivals in a multipath-fading environment. Nextsim supports voice traffic on the forward and reverse link while data services only on the forward link [77]. Figure 5.4 illustrates different components of Nextsim and the voice and data traffic paths.

Session control is a single entity that marks the start and end of the simulation. The number of BTS cells and MS are user defined parameters. The rest of the entities i.e., Bearer Traffic Model (BTM), Packet Control Function (PCF), Mobility Manager (MM), and Selection and Distribution Unit (SDU) have one-to-one correspondence with MS.

5.3.1.1 Session Control

Session control begins the simulation by starting sessions consisting of starting and ending mobile drops at user defined time intervals and positions. Once a session has been generated, session control finds an available mobile and repositions the mobile in the simulation space. Next, a session is initiated with the user specified traffic type (voice or data) via Bearer Traffic Model for that particular mobile. When the call ends the BTM sends back a message to session control and it is placed into the available pool of mobiles for session regeneration.

In case of a voice call, session control regenerates it unless it is at the end of the simulation time. This is because, unlike data calls, voice calls have user defined call duration distributions. Thus when a voice call is ended, session control repositions the mobile after a specified interval and regenerates the voice call. For data users, if the call

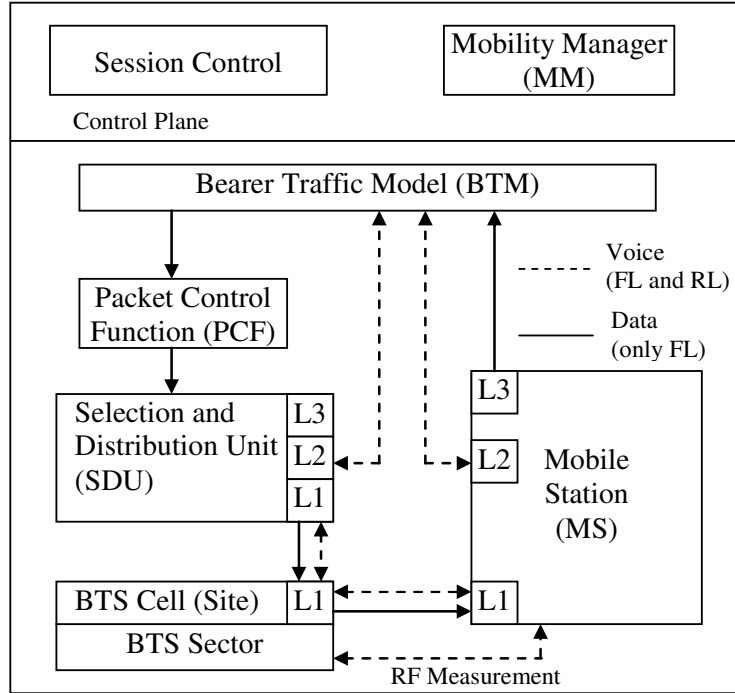


Figure 5.4: Network Extensible Simulator (Nextsim) architecture.

is ended due to abnormal termination i.e., RF loss, Packet Control Function regenerates the call.

5.3.1.2 Mobility Manager

Mobility Manager (MM) controls the mobility aspects of 1xEV-DV simulation. Its primary purpose is to originate call with the MS and perform handoff operations. At the beginning of a call, the MM receives a request from the BTM or PCF to start a call for voice or data user, respectively. Next, the MM sends an origination request to the MS to start the call. Once the MS responds positively, the MM informs the BTM or SDU to start transferring voice or data traffic, respectively.

Whenever the MS detects a pilot that is strong enough to be added to its *active set* or when it detects that one of its active pilots is too weak to be dropped, it sends a Pilot Strength Measurement Message (PSMM)² to the MM. After receiving the PSMM, the MM filters the pilots by carrying out its own validation on the PSMM. Usually, in a real

² PSMM contains a list of active and candidate set pilots with their respective EcIo measurements.

system, this operation is performed by the Transcoder (XC). For pilots requested to be added, the MM sends a request to the corresponding BTS(s) to allocate appropriate resources. Once the resources are allocated, a handoff direction message is sent to the MS. Likewise, for pilots to be dropped, the MM unallocates resources for the corresponding BTS(s).

5.3.1.3 Bearer Traffic Model

The function of generating different types of data and voice traffic is performed by the Bearer Traffic Model (BTM) entity. The data traffic supported by Nextsim includes File Transfer Protocol (FTP), Hyper Text Transfer Protocol (HTTP), Wireless Application Protocol (WAP), Near Real Time Video (NRTV) etc. Next, the voice and data protocols are discussed briefly with emphasis on FTP, since in this research it was used exclusively as the data service.

Voice Traffic Model: Voice traffic is emulated by Markov process that closely represents the characteristics of IS-95 vocoder. The call duration is specified according to a user-defined distribution. Number of frames is calculated based on the call duration and IS-95 frame duration of 20 msec.

FTP Traffic Model: The FTP session is simulated by a sequence of file transfers, separated by reading times. Figure 5.5 illustrates an example FTP session. It uses Transfer Control Protocol (TCP) to model the reading time delay and the slow-start mechanism. Table 5.1 defines the distribution parameters of FTP traffic model.

HTTP Traffic Model: HTTP traffic model simulates a typical web browsing session. The session consists of ON and OFF periods representing web page downloads and the intermediate reading times, respectively. Like FTP, HTTP also uses TCP to emulate the reading time delays.

WAP Traffic Model: The request from the WAP browser is modeled as having a fixed size while the response time of the WAP server is assumed to be exponentially distributed. The response is composed of a number of geometrically distributed objects. Once the last object is received, the exponentially distributed reading time starts, and it ends when the WAP browser generates another request.

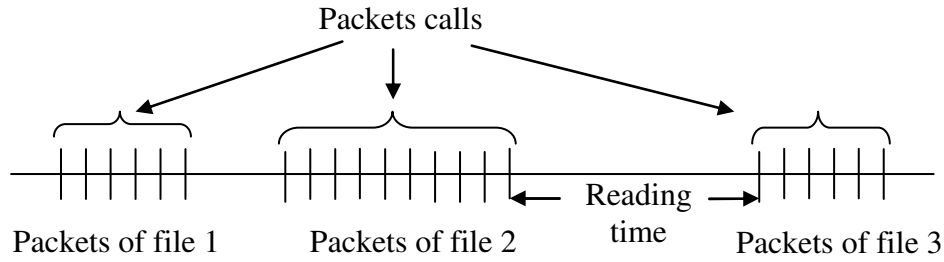


Figure 5.5: A typical FTP session.

Table 5.1: FTP Traffic model parameters.

Parameter	Distribution	Variable	Probability distribution function (pdf)
File Size	Truncated Lognormal	Mean = 2 Mbytes Std. Dev. = 0.722 Mbytes Max. size = 5 Mbytes	$f_x = \frac{1}{\sqrt{2\pi}\sigma_x} \exp\left[-\frac{(\ln x - \mu)^2}{2\sigma^2}\right], x \geq 0$ $\sigma = 0.35, \mu = 14.45$
Reading time	Exponential	Mean = 180 sec.	$f_x = \lambda e^{-\lambda x}, x \geq 0$ $\lambda = 0.006$

NRTV Traffic Model: This model simulates streaming video traffic using a source rate of 32 Kbps. A video session is defined as the entire video streaming call time. Each frame of video data is decomposed into a fixed number of slices. The size of these slices is distributed as truncated Pareto. Encoding delays of the video encoder are also modeled by truncated Pareto distribution.

5.3.1.4 Packet Control Function

Power Control Function (PCF) receives data packets from the BTM and transfers it to the SDU. There is one instance of PCF entity per MS. When the first packet arrives from BTM, PCF request MM to set up the call path. Initially PCF is in dormant state and enqueue all the packets. Once the call path setup is complete, PCF dequeues and transmits all the packets. As long as the PCF is in active mode, it forwards all the packets from BTM to SDU. If the call path is disrupted due to bad RF condition, the PCF goes back to the dormant mode.

5.3.1.5 Selection and Distribution Unit

The selection and Distribution Unit (SDU) provides an interface between the BTS(s) and the network communications (PCF, BTM and MM). SDU receives voice, data and signaling from BTM, PCF and MM, respectively, and forwards them to the BTS(s). It implements Radio Link Protocol (RLP), Link Access Control (LAC), outer loop power control, multiplexer/demultiplexer and selector/distributor.

RLP provides radio link framing and retransmission based error correction for data traffic. LAC performs segmentation and re-assembly and retransmission based error correction for control signals. LAC supports ACK required, fast repeat and un-acknowledged model of operations. When in ACK required mode, LAC guarantees error free delivery of signaling messages. Nextsim supports Fundamental Channel (FCH) for voice, Dedicated Control Channel (DCCH) and F-PDCH for data traffic. Both F/R-FCH channels support 20 ms frames. Data and voice traffic are multiplexed in F-PDCH while voice and signaling are multiplexed in FCH/DCCH. Demultiplexer extracts the appropriate data and signaling and delivers them to the corresponding higher layer entities. SDU implements the reverse link (RL) outer loop power control algorithm for R-DCCH and R-FCH channels. It receives Erasure Indicator Bit (EIB) from each serving BTS every 20 ms. Based on the status of the EIB, the outer loop threshold is updated every 20 ms. The delay in the feedback loop is assumed two Power Control Group (PCG) which is 2.5 ms.

5.3.1.6 BTS Cell and Sector

BTS cell implements MAC layer scheduling, L1 RF channel management, resource management. The BTS sector entity provides the necessary physical layer components such as receive antenna, transmit antenna, Transmit Power Amplifier (TxPA). After receiving resource request from the MM, the BTS dynamically allocates resources among data and voice users. Resources allocated by the BTS include PDCH, MAC scheduler, Walsh code manager. The MAC scheduler interfaces to several resource objects i.e., Walsh code manager, BTS sector TxPA, F-PDCH, F-PDCCH. The sector TxPA and

Walsh code managers are monitored every PCG (1.25 ms) to obtain the available power and code for scheduling assignment.

5.3.1.7 Mobile Station

The mobile station (MS) entity implements layers L1 through L3. L1 channels i.e., voice channels, PDCH, PDCCH etc. are allocated during initialization. All the channels use central signaling system to synchronize RF measurements and L1 framing between the MS and BTS(s). Pilot scanning functionality is provided that contains active/neighbor scanning for handoff management.

5.3.2 System Level Performance

The performance of a cellular network, from the RF perspective, has many dynamic aspects to it. The fact that the system is dynamic with many interrelated issues makes the job of improving the network performance very challenging. To optimize the RF system performance and expedite the troubleshooting process, it is imperative to determine the critical factors, which need to be monitored closely. Often times the link quality³ in a cellular network is quantified in terms of the following five factors:

Frame Error Rate (FER): Frame error rate, calculated as a ratio of bad frames and total number of transmitted frames, indicates the overall link quality. In cdma2000 employing Rate Set 2 (RS2) configuration, mobile transmits back the status of forward link frames every 20ms. Usually in a good link FER is around 1-2%. For poor link FER is 5-10%, while in a dropped call it may be as high as 20%. However this simple scalar statistic does not reveal anything about the physical phenomenon that impacts the link quality.

EcIo: EcIo is the ratio of the pilot being measured and the total power in the channel including the pilot being measured, pilots from other base stations, traffic channel and noise. Measured by the mobile, it is usually expressed in dB. EcIo is usually less than -15dB in a bad quality link. The Pilot Strength Measurement Message (PSMM) sent by the mobile contains all the detected BTS's EcIo information.

³ In this section and in the next Chapter we focus on the forward link (FL) only.

Received Power (RxPo): This metric indicates the total received power by the mobile. Judging by the raw received power only, it is not possible to indicate problems like pilot pollution (interference due to too many pilots with similar E_c/I_o). However, in a coverage hole it is usually lower ($<-100\text{dBm}$) than an interference limited environment ($>-95\text{dBm}$). RxPo of -76 dBm usually represents good link quality.

Transmit Power (TxPo): At the beginning of a call, the transmit power (TxPo) of the mobile is fixed by the “open-loop power control”, which sets it inversely proportional to the received power. Once on the traffic channel (fundamental channel (FCH) in 1x), the MS and BTS are engaged in a “closed-loop power control” which sends power control (PC) commands every 1.25 ms for 1 dB step. For CDMA phones, the maximum transmit power is 23 dBm (200 mW). A TxPo of 0 dBm indicates the mobile is close to the middle of a cell.

Transmit Gain Adjust (TxGa): Transmit gain (TxGa) adjust are the closed-loop power control commands send by the BTS to the mobile. If the interference level increases, TxGa also increases. For a good link, usually TxGa is in between -15 dB and -8 dB .

Given the set of critical parameters, we now present the simulation results from Nextsim simulator. The simulation scenario and the parameters are explained in Figure 5.6 and Table 5.2, respectively. A two-ring model was implemented where the outer ring

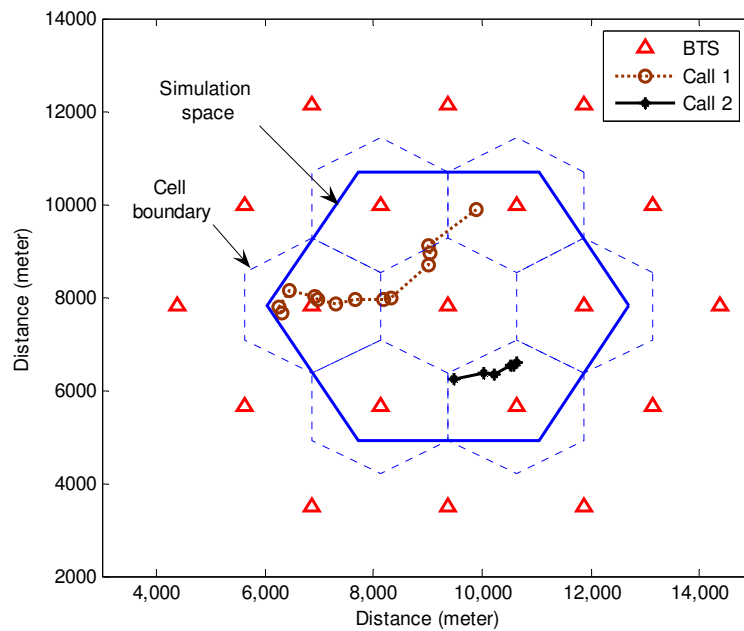


Figure 5.6: System level simulation layout.

Table 5.2: System simulation parameters.

Parameter	Value	Remarks
Number of Cells	19	2-ring, 3-sector system, 57 sectors in total
Antenna Pattern	70 deg (-3 dB)	
Cell Distance	2500 m	
Carrier Frequency	Forward Link = 1950 MHz Reverse Link = 1850 MHz	Frequency Division Duplexing (FDD)
Propagation Model	Standard Hata model extended to three dimensions	Hata model for urban, large city, min. separation 6 m.
Log normal shadowing	Mean 6.8, Standard deviation 6.5	
BTS Antenna Height	32 m	Antenna tilt 0 deg, antenna heading 0, 120, 240 deg. 2 elements for Rx. diversity in RL.
BTS Max. PA Power	20 W	
Pilot Power	2500 mW	
BTS Tx./Rx. Gain	Tx. = 15 dB, Rx. = 15 dB	
BTS Noise Floor	-107 dBm	
BTS Rx. Bandwidth	1.288 MHz	Corresponding to spreading rate of 1.2288 Mcps
MS Antenna Height	1.5 m	Antenna tilt 0 deg, single element 3 finger rake with MRC scheme.
MS Max Power	200 mW	
MS Tx./Rx. Gain	Tx. = 0 dB, Rx. = -11 dB	
MS Noise Floor	-103 dBm	
MS Rx. Bandwidth	1.288 MHz	
Pilot Strength Measurement Message (PSMM) variables	TAdd Threshold = -14 dB TDrop Threshold = -16 dB TComp Threshold = 4 dB TTDrop Threshold = 3 sec.	
Fading Channel Model	Based on Speed Model A: ITU Pedestrian A, 1 finger Jakes, 3kmph, 0.30 probability Model B: ITU Pedestrian B, 3 finger Jakes, 10kmph, 0.30 probability Model C: ITU Vehicular A, 2 finger Jakes, 30kmph, 0.20 probability	Jakes or Rician Model

	Model D: ITU Pedestrian A, 1 finger Jakes, 120kmph, 0.10 probability Model E: Single path Rician (K=10dB), 0mphs (5Hz Doppler), 0.10 probability	
Forward Link Power Control (PC) loop delay	Two power control groups (2.5 ms)	Power control command sent once every 1.25 ms. PC BER = 4%

of BTS represents interference only. A mixed load of 105 voice and 105 data (FTP) users were simulated for average call duration of 200 sec. The fading channel used in the simulation was Model D. The users were randomly dropped within the demarcated simulation space and were not allowed to perform handoff to the outer ring BTS(s). Two example calls are also shown in Figure 5.6, where the PSMM locations were marked.

The simulation results are summarized in Figure 5.7. Frame error rates for both the forward and reverse link are shown in Figure 5.7(a), which includes only the voice users. The simulation time of 200 sec. corresponds to roughly 10000 1x frames (frame length of 20 ms). The average FER in the FL is slightly higher, which is due to the diversity advantage in the RL with two antennas. From Figure 5.7(b), it is apparent that most of the times the closed loop commands are within the normal range with few samples greater than -5 dB that corresponds to interference limited scenarios. The sector power distribution is illustrated in Figure 5.7(c). Note that close to the maximum power limit of the BTS power amplifier (20 watts), the peak demonstrates the TDM based allocation of all the available power to a single data user. The second peak, which is much lower, represents the overhead channels in the sector while the floor is due to voice users.

Distribution of mobile transmit power (TxPo) is depicted in Figure 5.7(d). One interesting observation is that the distribution is skewed to the right of 0 dBm, which roughly points the mobile position to the middle of the cell. The distribution of the number of pilots in the active set of PSMM is illustrated in Figure 5.7(e). Note that even though most of the times there were three active set pilots, in some cases there were four pilots. This extra pilot adds interference, known as *pilot pollution*, since there are only three fingers in the Rake receiver of the mobile.

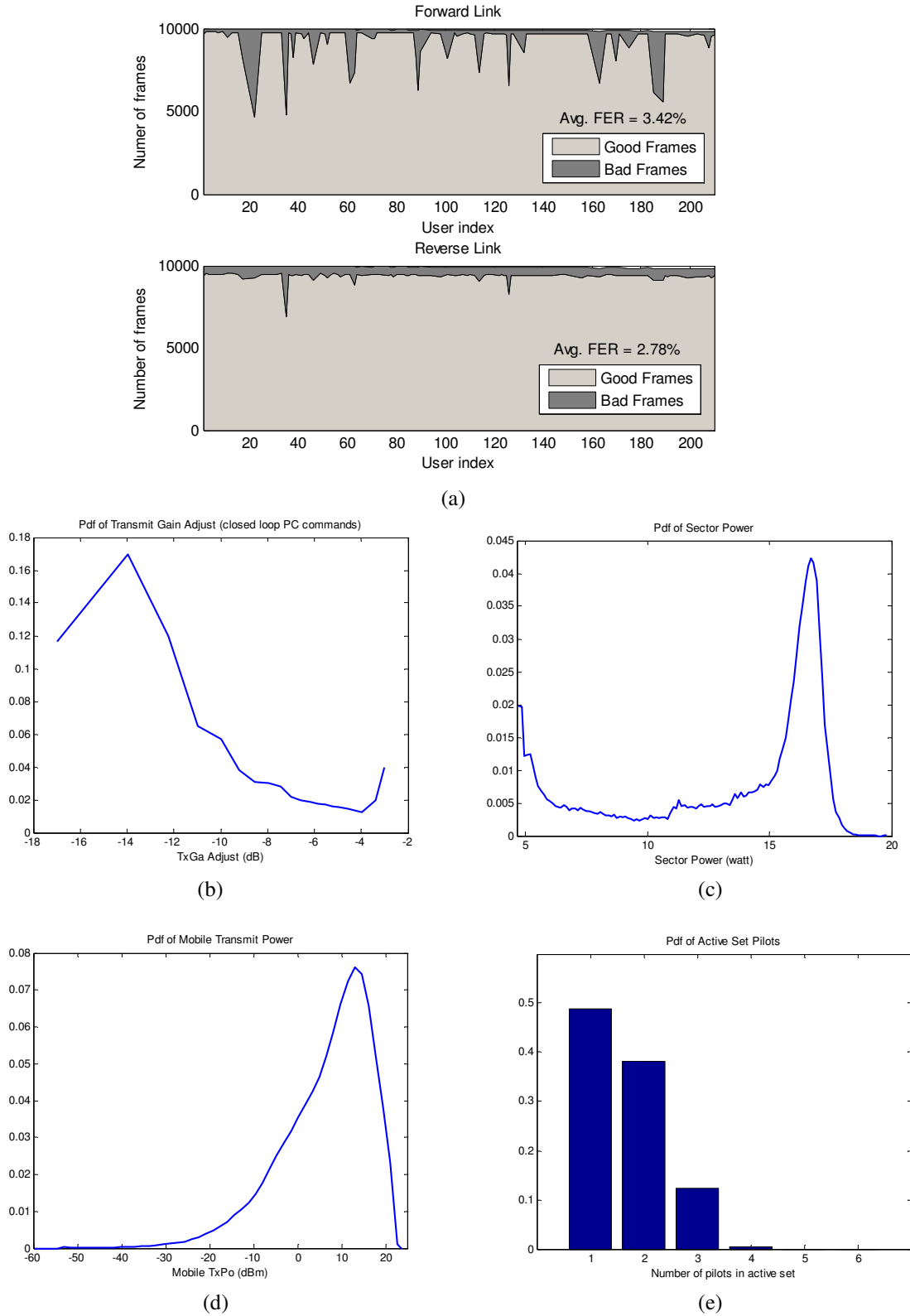


Figure 5.7: System level performance of Nextsim. (a) FER (b) Transmit Gain (TxGa) adjust. (c) Sector power distribution. (d) Mobile transmit power (TxPo) distribution. (e) Distribution of active set pilots in PSMM.

Based on these critical metrics and results, the link quality can be assessed. We will use this analysis and simulation results extensively in diagnosing cellular networks, specifically in predicting link quality, identifying dropped calls, etc. The details will be presented in the next chapter.

5.4 Conclusion

In this chapter the evolution of CDMA standards were discussed. The key aspects of 1xEV-DV systems were highlighted. By adding special features such as adaptive modulation and coding, hybrid ARQ, multi-code spreading and time division multiplexing of the PDCH channel, 1xEV-DV provides a means of leveraging the channel conditions in order to optimize data rates. We also introduced Motorola's Nextsim simulator that emulates the characteristics of 1xEV-DV RAN. The simulation architecture was explained with reference to the corresponding 3GPP2 standard. The system level performances of 1xEV-DV were presented. We will use this simulator in the next chapter to diagnose cellular networks.

Chapter 6: Cellular Diagnosis Using HMMs

In this chapter, hidden Markov models are applied to diagnose the performance of cellular networks. Based on the system level simulation results and critical parameters values discussed in Chapter 5, cellular network performance can be predicted. The impact of these parameter values and the corresponding link quality characterize the call statistics. Our objective is to analyze call statistics using hidden Markov models. Even though hidden Markov models have been widely used to simulate wireless channels [8], [22] and packet data error statistics [78], [79], to the best of the author's knowledge, it has not been extended to cellular diagnostic systems. The application of system diagnosis using HMMs is novel and intriguing. First, a simple model of *good* link quality will be presented. Next, the good link model will be extended to predict the link quality in terms of a hypothesis test. By segmenting error bursts, a novel method for modeling dropped calls will be presented. We will also show that coverage problems have signature statistics which can be classified using HMMs. All simulation results presented in this chapter are based on Motorola's Network Extensible Simulator (Nextsim) [77], which simulates 1xEV-DV radio access network.

6.1 Modeling Frame Error Statistics

In this section we describe the application of HMMs in modeling the “good” link quality (with Frame Error Rate (FER) between 1-2%) from the call statistics collected at the Base Transceiver Station (BTS). In 1x systems employing the Rate Set 2 (RS2) configuration,

mobile transmits 50bps Eraser Indicator Bits (EIB) indicating the status of the frame on the forward link. Even in moderately poor channel conditions the EIB is very reliable and can be considered close to a true frame error sequence. Since the EIB sequence is available at the BTS, it can be used to characterize the forward link. The basic idea is to extract the call statistics by having an HMM trained on the EIB sequence.

As shown in Figure 6.1, to generate an HMM model, first the number of iterations and the model structure is fixed. The Baum-Welch algorithm (BWA) is then trained by the frame error statistics collected from the EIB sequence. The frame error sequence is binary with 1 (0) indicating a frame error (no frame error). Since the observation sequence has only two possible symbols (either 0 or 1), $M = 2$ in this case. We used a three-state model with the parameters initialized as:

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 0.90 & 0.05 & 0.05 \\ 0.10 & 0.80 & 0.10 \\ 0.10 & 0.20 & 0.70 \end{bmatrix} \\ \mathbf{B} &= \begin{bmatrix} 0.90 & 0.80 & 0.70 \\ 0.10 & 0.20 & 0.30 \end{bmatrix} \\ \boldsymbol{\pi} &= [0.40 \quad 0.30 \quad 0.30] \end{aligned} \quad (6.1)$$

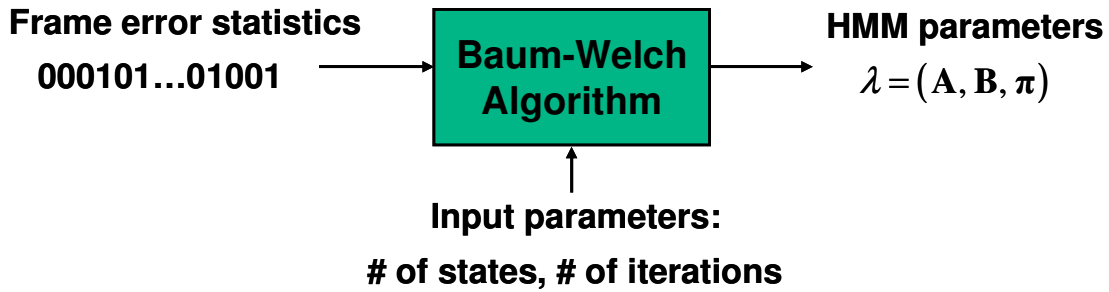


Figure 6.1: Training of HMM by frame error statistics.

Even though model parameters are often initialized randomly, we used some heuristic ideas in the initialization. Since the frame error patterns to be modeled represent good link quality, most of the states were assigned low error probability values. With the highest value of self transition, the first state mostly represents the error free intervals

which dominate the frame error statistics in a good quality link. The third state mostly modeled the error bursts with the highest error probability of 30%. A total of 10,000 frames were used in training the BWA. Since the FER for the training sequence was 0.0196, there were about 200 erroneous frames to model the statistics. After 50 iterations, BWA was terminated and the final model parameters are given by

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 0.9807 & 0.0193 & 0.0000 \\ 0.2007 & 0.7972 & 0.0021 \\ 0.0000 & 0.0101 & 0.9899 \end{bmatrix} \\ \mathbf{B} &= \begin{bmatrix} 0.9941 & 0.9666 & 0.3804 \\ 0.0059 & 0.0334 & 0.6196 \end{bmatrix} \\ \boldsymbol{\pi} &= [0.9983 \quad 0.0017 \quad 0.0000] \end{aligned} \quad (6.2)$$

The HMM defined by (6.2) is referred to as “good” link model (GLM), λ_{GLM} . Figure 6.2 shows the convergence of the BWA where the log-likelihood ratio is plotted against the number of iterations. Note that even though the algorithm was run for 50 iterations, it took the BWA only about 15 iterations to make a good estimate of the model parameters. Also, as expected, the most significant improvement on the initial approximation was made during the first few iterations.

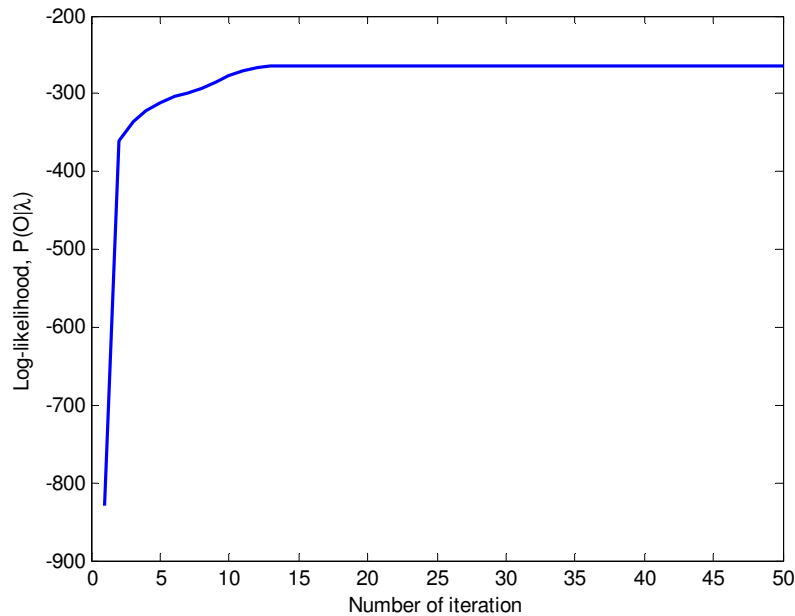


Figure 6.2: Log-likelihood function for HMM given by (6.2).

6.1.1 Validation of HMM

Once the HMM is generated, it must be tested to validate its accuracy. The validation process can be divided into two parts. In the first part, the first order statistic, i.e., the frame error rate, is compared. However, the FER statistic is not sufficient to test the reliability of the model, as two frame error sequences can have the same FER with completely different error distributions. Thus, the error-free intervals and the length of error bursts also need to be compared for a comprehensive model validation.

The FER for the GLM given by (6.2) is calculated in the same way as described in Section 2.6. Using (2.48), it was found that the frame error rate, $P_{FE} = 0.0195$, which is very close to the original FER (0.0196). Apart from the long-term FER, network operators also monitor the short-term FER to track the evolution of the link quality. Usually this is done by calculating the FER over 100 frames (2 sec. time period for 1x systems). To test the short term FER statistics, a hidden Markov chain was employed to emulate the transmission of 10,000 frames. Figure 6.3 compares the short term FER for the original and HMM generated sequences. Note that even though the HMM generated frame error patterns are not *identical*, they are statistically similar to the original.

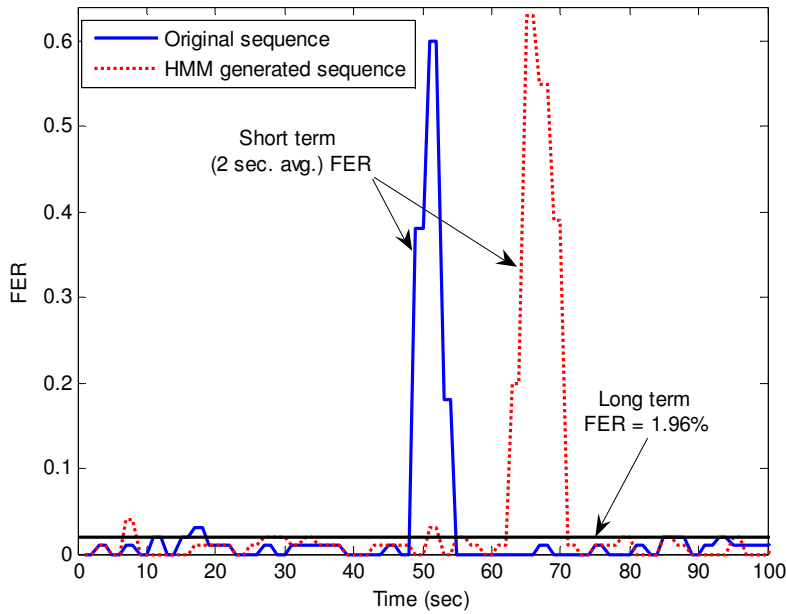


Figure 6.3: Comparison of original and HMM generated short term FER.

In the second part of the validation process, the distributions of the error statistics are examined. Usually the error patterns are analyzed by computing the following:

- 1) Distribution of error-free intervals
- 2) Distribution of error burst lengths

The distribution of error-free intervals and error burst lengths for the original sequence and the HMM generated sequence are depicted in Figure 6.4 and Figure 6.5, respectively, where $P(0m|1)$ ($P(1m|0)$) denotes the probability of m or more consecutive good (bad) frames following a bad (good) frame. The HMM generated sequence has nearly the same distribution as the original sequence. Thus we conclude that the HMM can successfully model the good link quality.

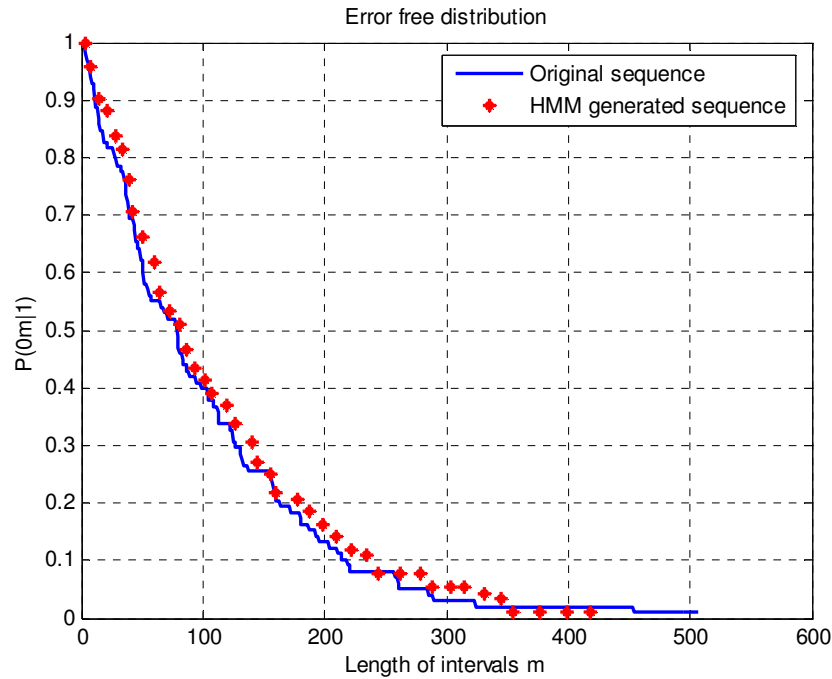


Figure 6.4: Comparison of error free intervals for the original and HMM generated sequences.

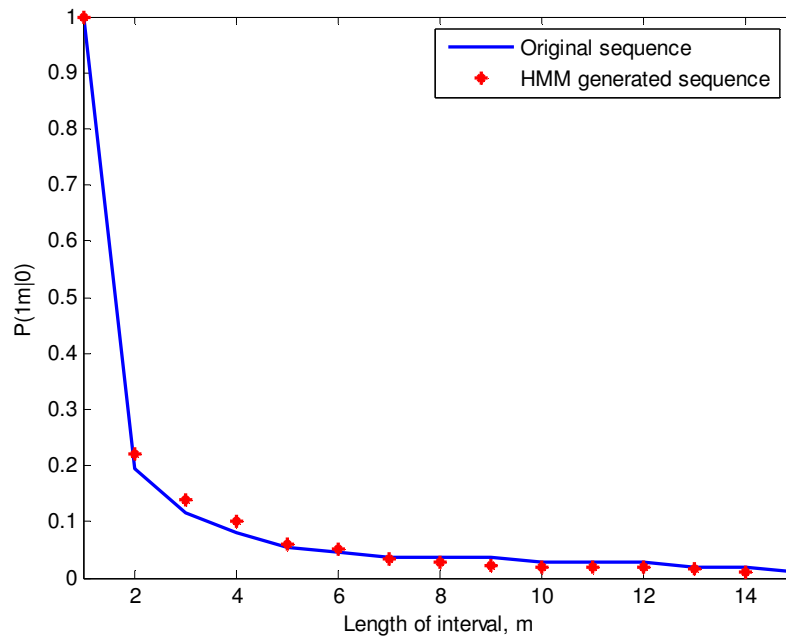


Figure 6.5: Comparison of error burst lengths for the original and HMM generated sequences.

6.2 Frame Error Patterns in a Dropped Call

In this section the frame error statistics in a dropped call are investigated. A call is declared dropped if 500 consecutive frames are in error. In other words, in a dropped call the fade timer expires when there is transmission loss over a time period of 10 sec. the fade timer is reset whenever the mobile receives two consecutive good frames. The timeline of a dropped call is illustrated in Figure 6.6.

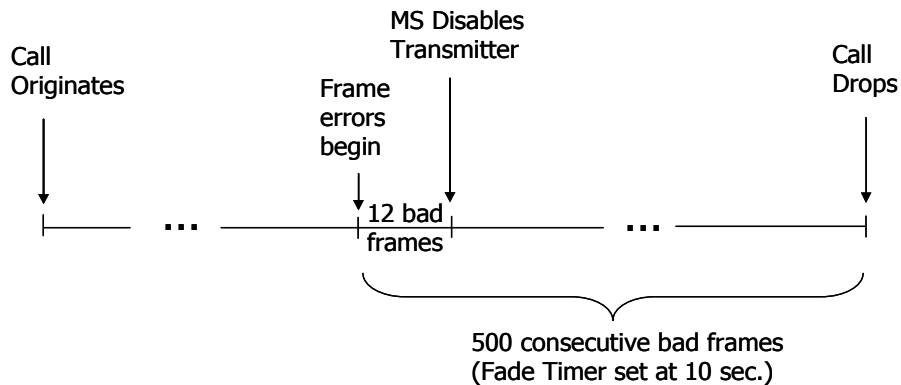


Figure 6.6: Timeline of a dropped call in CDMA.

Figure 6.7 shows one typical example of error patterns in a dropped call. As mentioned before, a '0' represents a correct frame while a '1' represents an erroneous frame. A glance at Figure 6.7 shows that the frame error pattern can be separated into two parts. The first part represents similar statistics as observed in a good link quality. The second part mostly consists of various error bursts depending on the intensity of the error flow and is referred to as the *dense error part*. These two parts can be partitioned based on one parameter, for example, by identifying the position of the frame error sequence where an error-free interval, for the first time, falls below some selected value ρ and never again exceeds that value. Usually, the length of error bursts increases as the call quality degrades with time. We see from Figure 6.7 that the statistics are time-varying over the frame error sequence. Due to the time-varying statistics, if the BWA is trained by the entire frame error sequence, the resulting Markov chain becomes *non-stationary*. However, to overcome this problem, the frame error sequence can be segmented into different parts based on the error intensity and each part can be modeled by a separate HMM. The modeling of error burst will be discussed in a following section.

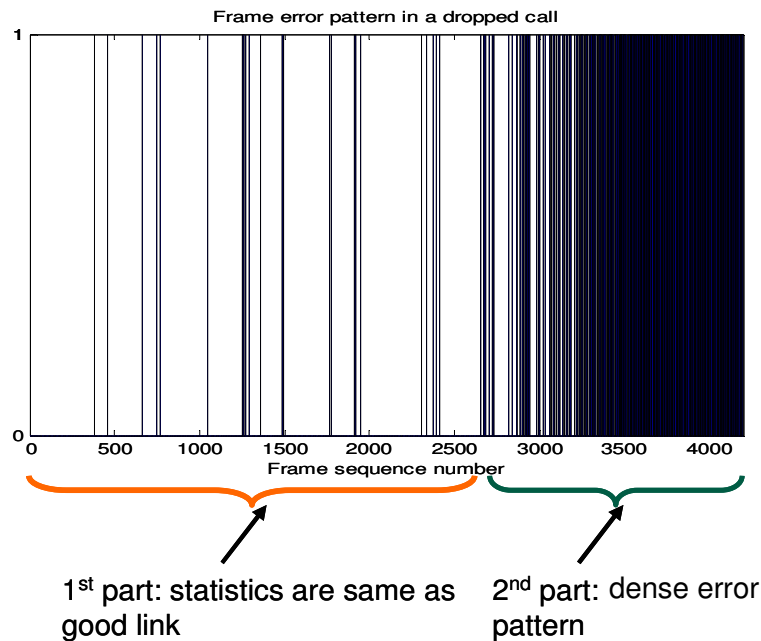


Figure 6.7: Frame error pattern in a dropped call.

6.2.1 Dropped Call Prediction Using HMMs

In a dropped call the intensity of error flow increases gradually over time. As the link condition continues to degrade, more and more consecutive frames are in error resulting in error statistics that change over time as the link quality evolves from good link quality call to dropped call. We propose the use of HMMs to capture the time-varying statistics in the frame error sequence and to predict dropped calls.

The basic idea behind this notion is to track the changes in the HMM as the link quality changes. In order to accomplish this, HMMs are trained on the frame error sequence using the *sliding window* approach. As the name suggests, the length of the training window is kept fixed while it slides forward by a fixed amount for the next training. The length of the window and the incremental step size are set to 1000 and 200 frames, respectively. Even though these values are somewhat arbitrary, a couple of things were factored into their selection. The training window cannot be too long so that unnecessary computations can be avoided. Also, if the training sequence is too short it does not adequately represent the statistics. Since the good link quality results in 1-2% errors, there are at least a few errors in the training window of 1000 frames. On the other hand, a smaller step size means frequent training, which will add to the computational burden. As the objective is to predict the call drops, the step size is upper bounded by 500 frames which is the maximum number of consecutive bad frames to terminate a call. The temporal statistics in the frame error sequence just before a call is dropped is very important in the prediction, and after trying with several values it was found that a step size of 200 frames produced reasonable results. The segmentation of the training window is illustrated in Figure 6.8.

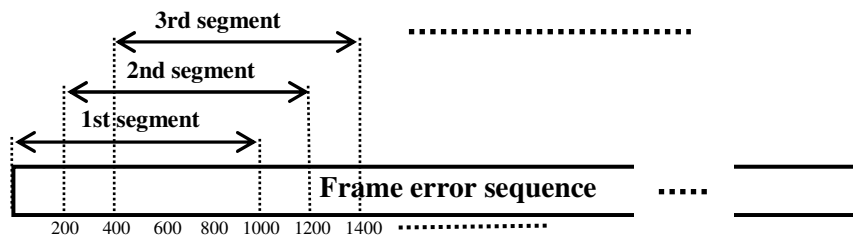


Figure 6.8: Training of HMMs for call drop prediction.

In the call drop prediction, the structure of the HMMs were the same as used for the good link model with three states and were initialized as in (6.1). Once the HMM, λ , is trained on a given window, it is compared to the good link model, λ_{GLM} . The comparison is accomplished by calculating the Kullback-Liebler divergence (KLD) [45] between the two models, which is denoted by $D(\lambda_{GLM} \parallel \lambda)$. Since the convergence of the BWA depends on the initialization, for a fair comparison the same initialization was used. When the link quality is good, the training window has only a few errors resulting in a model, λ , that is similar to λ_{GLM} . Consequently there is little difference between the models in probabilistic measures and the divergence is very small. As the link quality degrades, there are more errors in the training window and the corresponding HMM, λ , looks different from λ_{GLM} and the divergence $D(\lambda_{GLM} \parallel \lambda)$ is large. This is illustrated in Figure 6.9.

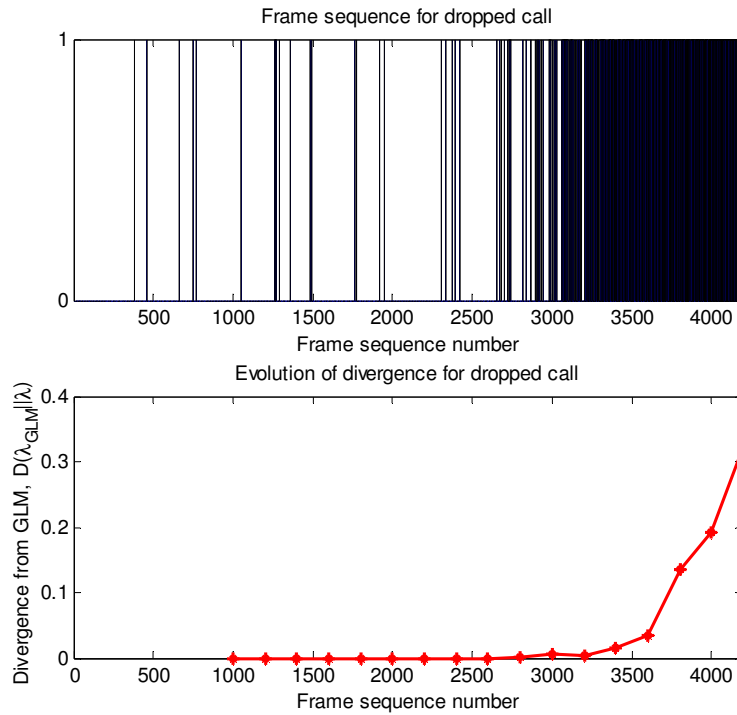


Figure 6.9: Evolution of divergence as link quality changes.

Also, note that in the training of HMMs, we have taken into account the fact that the objective is not to reproduce the frame error statistics in a dropped calls, which

probably would require extensive training and validation of the model itself. Rather the goal is to recognize the temporal changes in the frame error statistics, which can be done reasonably well with HMMs even with limited training.

Figure 6.10 shows the cumulative distribution function (CDF) of divergence between λ_{GLM} and λ , where model λ was generated by training on dropped call frame error sequences. In order to develop the call drop prediction as an alert system and to do the subsequent processing to avert the drop, there has to be a time interval between the actual drop and the time of alert. This time interval must be estimated based on the existing technology and the required processing. In this analysis, the time interval is arbitrarily set at 4 sec. (200 frames).

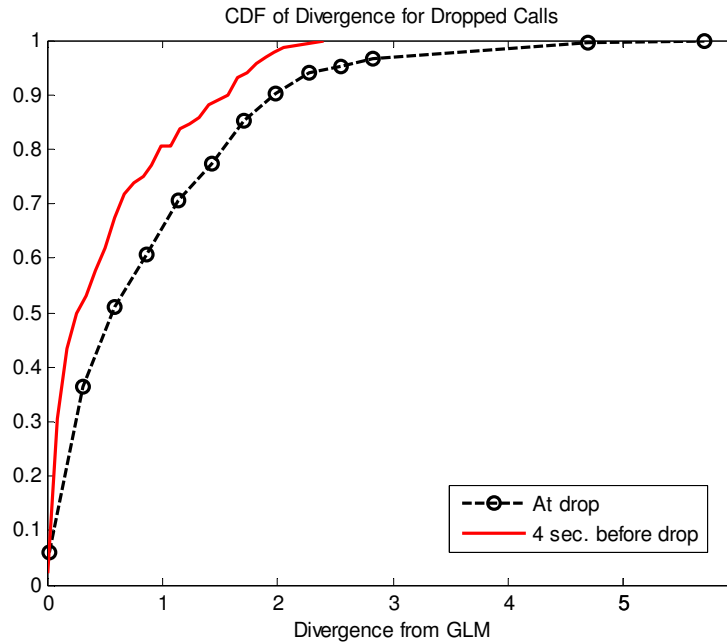


Figure 6.10: CDF of divergence at the time of call drop and 4 sec. before drop.

In Figure 6.11, the divergence of non-dropped calls is also added to calculate the false alarm in the hypothesis test [80]. Note that the long tail of the distribution is due to the calls that were not dropped but had very poor link quality. As shown in Figure 6.11, by setting the divergence threshold equal to 0.07, 70% of the dropped calls can be predicted with 10% risk of false alarm. Also, this threshold gives us 4 sec. worth of time to do the required processing to avert the call being dropped. Obviously if the processing

time is lowered, the divergence threshold for the same number of calls to be alerted will be higher. This will also decrease the false alarm probability.

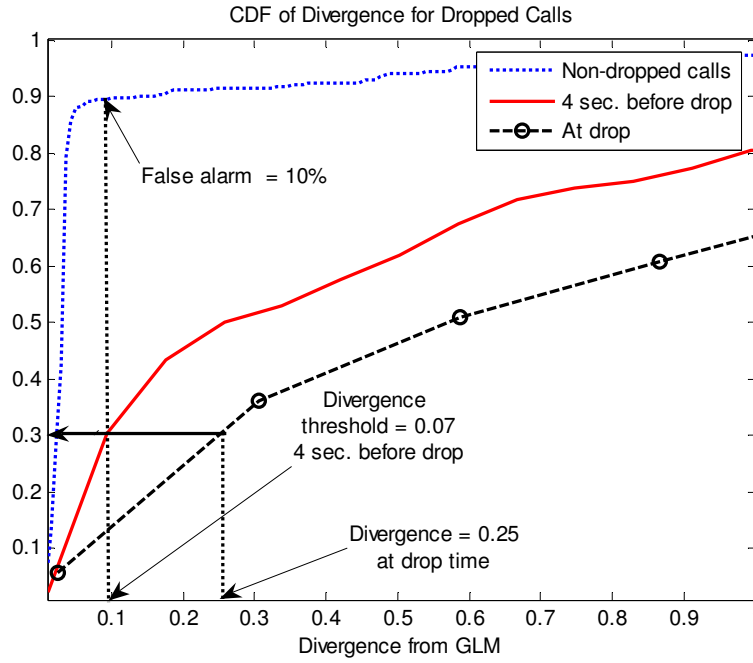


Figure 6.11: Hypothesis testing to predict dropped calls.

6.2.2 Burst Error Patterns in Dropped Calls

As already pointed out in a previous section, the dense error part mostly consists of various error bursts depending on the intensity of error flow. Also, as the call approaches transmission loss, the error bursts become longer and are separated by mostly *non-renewal* error-free intervals. Since the burst statistics vary significantly over the entire error profile, the dense error part is partitioned into segments based on the error flow intensity. The partition can be accomplished by identifying the portion of the dense error part as segment i , in which the burst length never exceeds a selected value τ_i . The segments are contiguous (non-overlapping) and $\tau_1 < \tau_2 < \tau_3 \dots$. Note that the proposed partition scheme defines the end point of the segments only. Since, by definition the segments are adjacent, the start of a succeeding segment follows immediately the end of

the preceding segment with no gap in between. Also, the beginning of the dense error part (defined in a previous section) is always the start of the first segment.

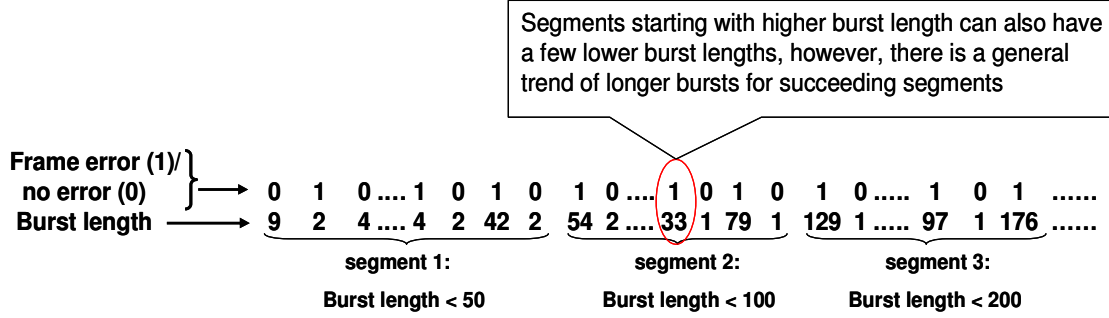


Figure 6.12: Segmentation of error burst in a dropped call.

Figure 6.12 illustrates the idea of segmentation where the first row represents the status of the frame, i.e., ‘1’ for an error and ‘0’ for a correct frame, and the second row denotes the corresponding burst length¹. For example, the sequence starts with 9 good frames followed by 2 bad frames and so on. The partition values were set such that $\tau_1 = 50, \tau_2 = 100, \tau_3 = 200$ for the first three segments. Note that even though generally burst length increases for succeeding segments, occasionally there may be a few bursts of lower lengths in a segment starting with a higher burst length. This is unavoidable, since in a dropped call burst lengths are not always monotonically increasing. Unlike in error pattern generation, where the error bursts are reproduced and randomly placed between good frames [78], in the analysis of dropped calls the temporal relation between the good and bad frames must be preserved. Once the dense error part is partitioned into a given number of segments, a separate HMM is trained by each segment to analyze patterns in that segment. Even though the main objective is to analyze the patterns in each segment, the segmentation approach can be extended to reproduce the whole error sequence by concatenating the HMMs.

¹ This is a compact representation of error sequence which is commonly known as *runlength distribution*.

6.3 Common Network Problems in CDMA

System level performance analysis and troubleshooting is one of the most daunting and rewarding aspects of working in a wireless network. For a given technology and system configuration, there are some basic concepts of system design to optimize network performance. From a system level perspective, the performance of a wireless network has many dynamic aspects to it. The fact that the system is dynamic and many issues are interrelated makes the task of improving the performance of a system very challenging. To improve services, cellular providers have to handle issues related to the addition of new sites, changes to existing sites, new features, maintenance, and customer service. However, when a problem occurs, providers are saddled with the task of wading through a large amount of data in order to troubleshoot their systems. To enhance the system performance and expedite the troubleshooting process, system operators continually monitor several critical system metrics, or key factors. As systems continue to grow in size and complexity, the statistics of key factors become more and more crucial. The particular system metrics used by the operator are often dependent on the underlying system infrastructure. However, irrespective of the wireless technology or infrastructure used, the important metrics in a cellular network include number of dropped calls, blocking (congestion) probability, probability of access failure, frame error rate (FER), and equipment failure. By focusing on these parameters, service providers can proactively handle the issues of troubleshooting.

The dropped call statistic is probably the most discussed and monitored metric in the wireless industry. This is mainly because this metric has a personalized impact on subscribers as most of them experienced the frustration associated with a dropped call. Even though there have been improvements in detecting and isolating dropped calls over the last few years, the proliferation of wireless technology demands better services. Typically in CDMA systems, a dropped call occurs when the BTS can no longer maintain communication with the mobile. The failure of communication between the BTS and the mobile has a multitude of possible causes, including but not limited to,

- 1) Coverage problem

- 2) Hand-off failure
- 3) Pilot pollution
- 4) Loading
- 5) Neighbor list problem
- 6) Cell breathing

Coverage problem occurs due to insufficient signal strength at the mobile and may be caused by the terrain or the existing system configuration. This may happen in a localized area in a cell, called *coverage hole* (a low spot in the road), or at the extreme edges of the coverage area due to poor signal strength. Sometimes adjacent cell configuration may also affect the site's overall coverage map. Real world cell coverage maps are highly irregular, far from the neat hexagonal models found in text books. Usually infrastructure vendors use their internal diagnostic tools to monitor the received signal strength indicator (RSSI) in a drive testing campaign to plot the coverage map. The field data collected from drive testing is compared against predicted models to verify coverage problems.

Even though hand-off failures in CDMA systems are not as prevalent as in other wireless systems, they are still important to be aware of as in most cases the end result is a dropped call. Since all three types of hand-offs, soft, softer, and hard, can occur in a CDMA system, handoff failures can occur between sectors of a cell, between cells, between CDMA carriers, and even in a transition zone from CDMA to AMPS. As a result, the types of hand-off failures and the corresponding root causes cover a wider range of possible issues than most of the other technologies. Hand-off problems may also be triggered by a mismatch in radio configuration (RC) between neighboring cells. The hand-off parameters i.e., T_ADD, T_DROP, T_COMP are also important factors as too low or too high values may create an imbalance in loading between neighboring cells.

Pilot pollution is defined as the presence of too many pilots having significant strengths. It can be caused by a number of factors, including uneven terrain, bridges and overpasses that are covered by too many transmitter stations at the same time and on the same carrier. Besides creating interference, pilot pollution causes additional load on the

network. Since the best server is only of short duration with pilot pollution, hand-offs occur frequently occupying precious radio resources. Since the rake receiver of most of the mobiles has three fingers, if there are more than three pilots of significant power, pilot pollution occurs resulting higher FER and increased dropped call rate. Pilot pollution can be eliminated by adjusting cell size parameters or aligning a few antennas.

Excessive traffic load at the surrounding cells may cause a blocking or dropped call due to hand-off failure. Too many soft or softer hand-offs, the lack of channel elements or Walsh codes can be a direct result of excessive loading and is often manifested as dropped calls. A lack of Walsh codes is not common unless the cell is very small with very high traffic.

Neighbor list problems occur when there are incorrect or missing neighbors in the mobile's BTS list. In this scenario, the mobile receives a high-strength pilot signal that does not appear in its neighbor list causing high interference. The lack of reciprocity in the neighbor list also causes a high dropped call rate in the neighboring cells. Most of the neighbor list problems are manifested as hand-off failures before leading to high FER and dropped calls.

Cell breathing is a common phenomenon in CDMA systems. The constant change of the range of the geographical area covered by a BTS based on its current amount of traffic load is called cell breathing. When a cell becomes heavily loaded, its range shrinks and the traffic is then redirected to a neighboring cell that is more lightly loaded. This phenomenon is also called *load balancing*.

In this chapter, we propose that HMMs be used as a method to infer signature statistics about the nature and sources of dropped calls in a cellular system. The motivation behind this idea is the fact that, in most cases, clues about the cause of a dropped call are hidden deep within the statistics of dynamic physical phenomena like fading, shadowing, and interference. So, different causes of dropped calls may impose different signature patterns on the corresponding frame error sequences. By fitting models to frame error sequence, first, a prototype HMM is created to represent a particular type of dropped call. Next, based on the signature error patterns, the prototype

HMM is used to classify dropped calls. The modeling of coverage related dropped calls is discussed in the next section.

6.3.1 Hidden Markov Modeling of Coverage Related Dropped Calls

In order to analyze the statistics in the error sequence generated in a coverage related dropped call, first the dense error part was identified by finding the position of the error-free interval that never exceeds $\rho = 30$. Next, the dense error part was partitioned into five segments with $\tau_1 = 50, \tau_2 = 100, \tau_3 = 200, \tau_4 = 400, \tau_5 = 500$. For each segment, a separate HMM was trained to capture the corresponding error patterns. In this section we consider the modeling of the first segment only since extension to other segments is straightforward. After segmentation the length of the first segment becomes small (on average only 1000 frames). So just one training segment may not be sufficient to reliably estimate the parameters of the HMM. To overcome this problem, multiple observations training assuming independent observation sequence was used [41]. The first segments from 10 different dropped calls were used in the multiple observation training. Since the dropped calls were collected from different sectors, the frame error sequences were assumed independent. The three-state model parameters were initialized as

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 0.50 & 0.30 & 0.20 \\ 0.10 & 0.60 & 0.30 \\ 0.05 & 0.05 & 0.90 \end{bmatrix} \\ \mathbf{B} &= \begin{bmatrix} 0.70 & 0.30 & 0.10 \\ 0.30 & 0.70 & 0.90 \end{bmatrix} \\ \boldsymbol{\pi} &= [1 \quad 0 \quad 0] \end{aligned} \tag{6.3}$$

The first state represents the error free interval and the last two states model the error bursts. The third state models the longer burst lengths with the highest self transition probability. Since the training sequence consists of mostly error bursts, all the states are associated with high error probability. Also, the initial state is always set to start in the

first state, which is consistent with the formation of the first segment². After 50 iterations of the multiple observations sequence training, the algorithm converged and the final parameters were given by

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} 0.8038 & 0.1866 & 0.0097 \\ 0.2023 & 0.7819 & 0.0158 \\ 0.0059 & 0.0131 & 0.9811 \end{bmatrix} \\ \mathbf{B} &= \begin{bmatrix} 0.9372 & 0.2776 & 0.1182 \\ 0.0628 & 0.7224 & 0.8818 \end{bmatrix} \\ \boldsymbol{\pi} &= [1 \quad 0 \quad 0] \end{aligned} \quad (6.4)$$

The model given by (6.4) is defined as *coverage model*, λ_{cov} . Note that even though the burst length increases over time, the resultant HMM is not a *left-right* model. The *ergodicity* of the model is mainly because of the occasional smaller bursts in between longer bursts. The validation of λ_{cov} is illustrated in Figure 6.13, where the error burst lengths are compared. Even with only three states, the model can reasonably represent the original sequence statistics. Similar models can be generated for other segments. Once all the models are trained, they can be concatenated to synthesize the complete dense error part of the frame error sequence.

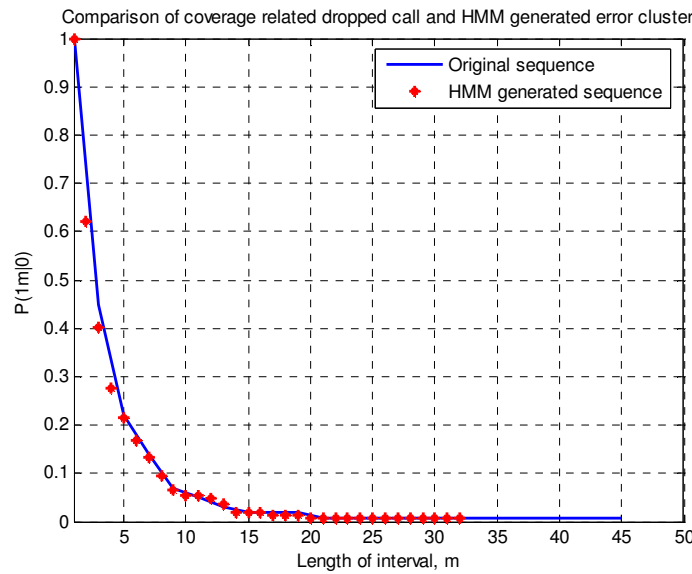


Figure 6.13: Hidden Markov modeling of error sequence in a coverage related dropped call.

² Based on the parsing of the error sequence, the first segment always starts with an error free interval.

6.3.1.1 Optimization of Coverage Model

In the results presented so far, all the proposed HMMs were developed using only three states. While the number of states was chosen arbitrarily, the models produced acceptable results in regenerating the statistical patterns. However, in order to optimize the model structures we need to investigate the impact of additional states. Obviously, as more states are used in the model, the computational complexity increases. However, an increased number of states also provide the models more degrees of freedom in extracting the underlying statistical patterns. So the apparent trade-off between complexity and performance must be considered in designing an HMM.

To analyze the impact of the number of states, the coverage models were developed with $N = 3, 4, 5, 6$ states. For each number of states, the model, λ_{cov} , was randomly initialized and was trained for 50 iterations. The results are summarized in Table 6.1. The performance of the model was determined in terms of the *log-likelihood function*, $\log[P(\mathbf{O}|\lambda_{cov})]$, using (3.75). The log-likelihood function indicates how likely the model is to represent the statistics in the training data. As the number of states is increased it is also possible to have redundant states in the model. Redundant states do not necessarily improve performance, nonetheless increase computational complexity. Based on the parametric analysis of the HMM discussed in section 4.4, the inverse condition number (ICN) and the residual sum vector, $\mathbf{R}$, are also calculated for the models. As observed from Table 6.1, as the model size is increased, the correlation between states increases resulting in lower ICN. The three-state model is found to be the best choice in terms of complexity/performance trade-off.

Table 6.1: Impact of number of states on coverage model, λ_{cov} .

States	$\log P(\mathbf{O} \lambda_{cov})$	ICN	Residual sum vector
3	-254.70	0.4654	$\mathbf{R} = [0.5919 \ 0.7604 \ 1.0537 \ 0.8959 \ 0.6531]$
4	-260.44	0.0254	$\mathbf{R} = [0.1621 \ 0.2876 \ 0.1452 \ 0.1425 \ 0.1107 \ 0.1354]$
5	-256.78	0.0509	$\mathbf{R} = [0.2240 \ 0.3504 \ 0.3394 \ 0.2991 \ 0.2720 \ 0.2378 \ 0.1974]$
6	-256.23	0.0179	$\mathbf{R} = [0.3237 \ 0.8791 \ 0.1731 \ 0.2516 \ 0.3108 \ 0.2894 \ ...$ 0.2014 0.1056]

6.3.2 Identification of Coverage Problems Using HMM

Once the coverage model, λ_{cov} , is generated from the training data set, it can be used to identify coverage related dropped calls. The idea is motivated by the fact that if coverage related dropped calls have different statistics compared to other dropped calls (due to pilot pollution, loading etc.), the HMMs generated from the corresponding error sequences should be different as well. The notion of identification of coverage problem can be explained as follows: once a call is dropped, its error sequence is used to train an HMM, λ_{drop} . If the error statistics in the given dropped call is similar to the ones in the coverage related problem then the models λ_{cov} and λ_{drop} should also be similar. However, if the statistics are different then the models should look dissimilar. As before, the similarity between the models can be measured in the probability space in terms of the Kullback-Liebler divergence (KLD) [45].

The identification of coverage problem can be formulated as a simple hypothesis test. In a hypothesis test, the question of interest is simplified into two competing claims; the null hypothesis, H_0 , against the alternative hypothesis, H_1 . Usually, H_1 is set as the complement of H_0 . In the context of the coverage problem, the null hypothesis is proposed as

$$H_0 : \text{the dropped call is due to coverage problem} \quad (6.5)$$

If the divergence between the two models, $D(\lambda_{cov} \parallel \lambda_{drop})$, is greater than a selected threshold, w , the null hypothesis is rejected, otherwise it is accepted. Like any other hypothesis test, the threshold is set based on the *acceptance region* and *rejection region*. The acceptance region is defined by the probability density function (pdf) of the divergence between the coverage model, λ_{cov} , and models trained from coverage related dropped calls that are *not* included in the training set of the coverage model, λ_{drop_cov} . Intuitively, $D(\lambda_{cov} \parallel \lambda_{drop_cov})$ should be very small reflecting closeness between the models. Since λ_{cov} is trained on a small number of dropped calls, the metric

$D(\lambda_{cov} \parallel \lambda_{drop_cov})$ also evaluates the performance of the model for “unseen” coverage related dropped calls. On the other hand, the rejection region is defined by the pdf of the divergence between λ_{cov} and the models trained from the all but the coverage related dropped calls, $\lambda_{drop_all \setminus cov}$. Unfortunately, since these two regions overlap there are two types of errors associated with the hypothesis testing.

- 1) Type I Error (probability of misdetection):** Type I error occurs when the null hypothesis is rejected even though it is true. The probability of misdetection is given by

$$\int_w^\infty P_{D(\lambda_{cov} \parallel \lambda_{drop_cov})} \quad (6.6)$$

- 2) Type II Error (probability of false alarm):** Type II error occurs when the null hypothesis is not rejected even though it is false. The probability of false alarm is given by

$$\int_0^w P_{D(\lambda_{cov} \parallel \lambda_{drop_all \setminus cov})} \quad (6.7)$$

where p_x denotes the pdf of x .

These two types of errors are inversely proportional and reflect the tradeoff between the risk of detection failure and the unnecessary use of the resources associated with a false alarm.

The algorithm to identify coverage problem is summarized below:

- 1) First, a hidden Markov model (HMM) is trained with the frame error sequences from a set of “known” coverage related dropped calls. This HMM, denoted as λ_{cov} , extracts statistical features from the error sequences.
- 2) Another HMM, λ_{drop_cov} , is trained with the frame error sequence of a coverage related dropped call that is not included in the training set of λ_{cov} .
- 3) Calculate the KLD between λ_{cov} and λ_{drop_cov} .

- 4) Repeat step 2 and 3 for all the coverage related dropped calls that are not included in the training set of λ_{cov} .
- 5) Evaluate the probability distribution of divergence calculated in step 3.
- 6) Repeat step 2 through 5 for all non-coverage related dropped calls which results in a probability distribution of divergence for non-coverage related dropped calls.
- 7) Based on these two distributions, a threshold w , is set and a hypothesis test is carried out. For a given dropped call, if the divergence is greater than w , the call is rejected as a coverage related dropped call and vice versa.

Figure 6.14 illustrates the hypothesis test to identify coverage problem. As explained in a previous section, due to error bursts of different density, the error sequences were first partitioned into different segments. Next, the HMMs λ_{cov} , λ_{drop} , etc. were generated by the training sequences obtained from the first segment only. Each of the pdf was generated from 100 dropped calls statistics. As shown in the figure, for a threshold, $w = 0.028$, the probability of misdetection and false alarm are 8% and 16%, respectively. Based on the application, the value of w can be varied to meet a specific service requirement.

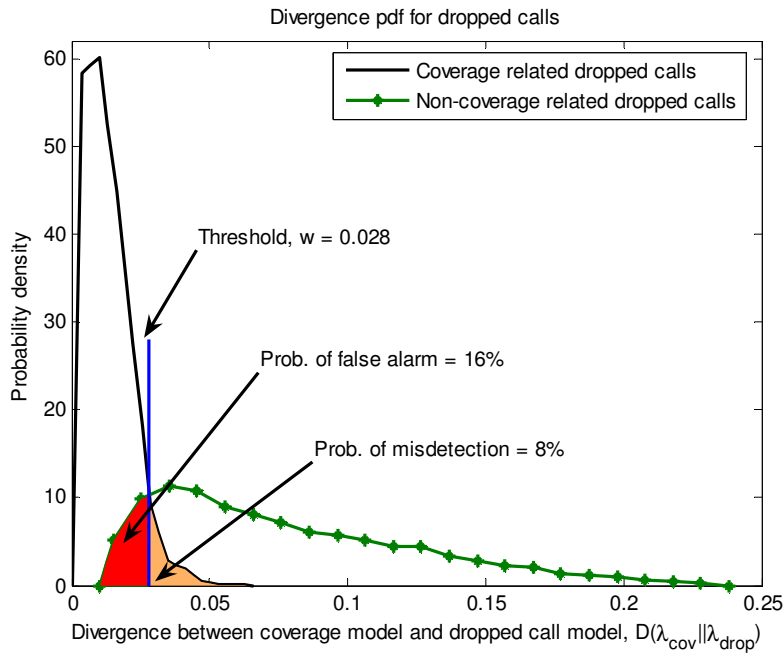


Figure 6.14: Identification of coverage problem by hypothesis testing for the first segment.

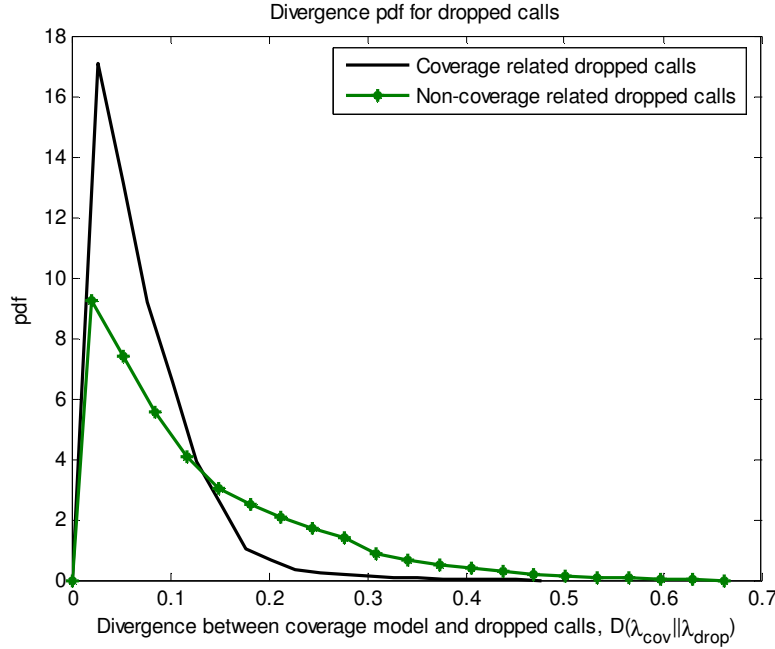


Figure 6.15: Probability density function of divergence between coverage related and non-coverage related dropped calls for the second segment.

When the same hypothesis test is carried out on the second segment, the results were not as good as for the first segment. Figure 6.15 depicts the hypothesis test for the second segment. Similar results were found for the third segment as well. The reason behind this poor performance is that the error bursts in both the coverage and non-coverage related dropped calls have similar patterns over all the segments except the first one. Consequently, the resulting models are similar and the corresponding pdfs overlap for most of the part. However, in the first segment, the error bursts have different density patterns for coverage and non-coverage related dropped calls. Specifically, in coverage related dropped calls, the link quality degrades more gracefully over time producing many short error bursts. In non-coverage related dropped calls, there are *not* many short error bursts due to the rapid degradation in link quality. This is why the hypothesis test performs best over the first segment. Also note that the notion of coverage problem identification can be extended to an alert system. This is because the extraction of distinguishing features between coverage and non-coverage related dropped calls requires only 50 consecutive frame errors, while a call is not declared dropped until it goes through 500 consecutive frame errors. So by implement the algorithm in the BTS, the

users may be notified about the impending coverage hole in real-time. Given the technology and processing power, if the required processing can be accomplished in time, the drop due to coverage problem may even be averted which will greatly improve customer service.

Coverage holes in a cellular network are usually detected by drive testing. While drive testing is expensive, it is neither exhaustive nor flexible -- making the detection of coverage related dropped calls very challenging. Existing methods employ correlation of scalar statistics from the call detail log (CDL) files and provide a coverage map based on past history. Current methods typically involve empirical measurements across the network, which is time consuming. The proposed algorithm employs HMM based techniques to speed up detection, allowing for near real-time identification of coverage holes. Also, note that coverage models can be made robust and flexible by adding new coverage related dropped call statistics.

6.3.3 Non-Coverage Related Problems

The classification of dropped calls due to non-coverage related problems i.e., forward link loading, pilot pollution, hand-off failure and shadowing was also attempted through prototype HMMs. However, unlike coverage problem, non-coverage related problems do not have signature patterns in their frame error sequences that can be exploited to classify them. This fact is somewhat intuitive in the sense that almost all of the non-coverage related problems are due to excessive interference. The source of the interference may be different in different classes of dropped calls but that is not reflected in the error bursts of the frame error sequence. For example, in the case of forward link loading, the interference is due to all other users using the same BTS, while in pilot pollution it is the excess number of neighboring BTSs that causes most of the interference. So the problem of classifying non-coverage related dropped calls comes down to identifying interference sources. Unfortunately, this is not possible by observing error patterns only. Other call statistics like pilot strength measurement message (PSMM) may be used to identify interference sources. The similarity of error bursts in non-coverage related dropped calls

may also be the reason for the single mode distribution of $D(\lambda_{cov} \parallel \lambda_{drop_all \setminus cov})$ in Figure 6.14. On the other hand, coverage problem is not an interference limited scenario; rather it is due to insufficient signal strength and is manifested in the error patterns in the frame error sequence. This is why HMMs can successfully isolate coverage problems from all other non-coverage related dropped calls.

6.4 Conclusion

In this chapter, the application of HMMs in modeling frame error sequences at the system level was discussed. HMMs have been widely used as an error generating source to facilitate simulation of complex communication systems. In this chapter HMMs have been proposed as a predictive tool in network diagnostic systems. HMMs can be used to predict dropped calls and this idea can be integrated into the radio resource management system to improve customer service. Due to time-varying statistics of the error sequence in a dropped call and the resulting non-stationary HMM, a new method was proposed to model the dropped calls in a cellular network. After dividing the frame error sequence into smaller group of segments, the sequence can still be modeled with an HMM. By concatenating the HMMs of different segments, the whole error pattern can be regenerated. Coverage related dropped calls have signature patterns that can be extracted by HMMs and used to identify coverage problems. However, by only observing frame error patterns, it is not possible to classify non-coverage related problems.

Chapter 7: HMM-Based Position Location

Finding the location of people, equipment, mobile terminals and access points in indoor and outdoor environments has become a problem of growing importance. For example, reliable position location is imperative to ensure the safety and efficiency of emergency management workers, i.e., fire fighters, coal miners, etc. Geolocation capabilities are also important in the context of "cognitive" radios as they are expected to be "aware" of their environment. By estimating their locations, cognitive radios have the potential to accommodate high data rate multi-media applications through improved radio resource management and better transmission technologies. Asset management using RF ID tags is another potential application that requires accurate location information. In this chapter, the hidden Markov model (HMM) [7] is shown to be useful as a position location tool. In a wireless system, channel statistics, i.e., power delay profile, delay spread of multipath components are indicative of the corresponding location characteristics. The HMM based technique allows a location to be matched according to its channel profile. The proposed method does not depend on the line of sight (LoS) reference and performs well for environments that suffer from multipath and non-line of sight (NLoS) propagation.

7.1 Traditional Position Location Techniques

Over the years, researches have investigated a number of position location algorithms based on radio frequency (RF) wave propagation [81]. Since an electromagnetic wave at RF is capable of penetrating most objects, position location can be attempted by direct

measurement of radio signals traveling between the transmitter and the receiver. Most common position location algorithms are usually implemented using angle of arrival (AoA), direction of arrival (DoA), or time difference of arrival (TDoA) techniques. It is also possible to combine two or more of these techniques to formulate what is known as a *hybrid* technique. We will briefly discuss these techniques here.

7.1.1 Angle of Arrival Based Location Estimation

The angle of arrival (AoA) method uses antenna-arrays to estimate the direction of the signal of interest. A single AoA measurement restricts the source of the signal to be along a line. Now, if at least two such AoA estimates are available at two different locations, the position of the signal source can be located by the intersection of the two lines. Usually the accuracy of this method can be improved by using multiple AoA estimates. Figure 7.1 illustrates the method of finding the source location by the intersection of AoA from three antenna arrays. By exploiting the phase difference between multiple antennas, the AoA can be estimated.

Although the basic principle of the AoA method is simple, it suffers from a few drawbacks. For accurate AoA estimates, it is important that the signals coming from the source to the antenna arrays have LoS propagation including narrow angle spread. However, this is often not the case in wireless communications, especially in indoor environments. Also, multiple antenna elements are required to implement this technique. Regular calibration of the antenna array is necessary as a slight change in the physical arrangement of the array may result in significant error in the location estimation.

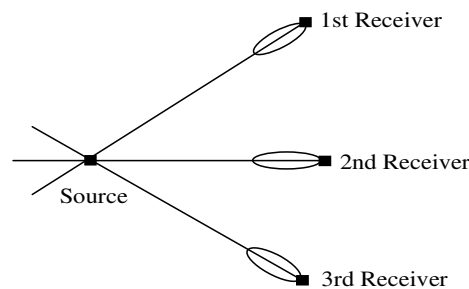


Figure 7.1: AoA based position location.

7.1.2 Time of Arrival Based Location Estimation

In the time of arrival (ToA) technique, the receiver indirectly determines the time taken by the signal to propagate from the source to the receiver. This is usually done by measuring the time in which the source responds to an inquiry or an instruction transmitted to the receiver. The total time elapsed from the instant the instruction is transmitted to the instant the source response is detected includes the sum of the round trip delay and any processing and response delay within the source. If the processing delay within the source unit is known with sufficient accuracy, it is possible to estimate the total round trip delay. Half of that quantity represents an estimate of the signal delay in one direction, which then provides an approximate distance of the source from the receiver. If the source response can be detected at two additional receivers, the triangulation method can be employed to find the position of the source at the intersection of the circles determined by the time delay measurements. Figure 7.2 illustrates the ToA based position location technique where r_1 is the estimated distance of the source from the 1st receiver and so on.

There are several problems associated with this method. Since the ToA technique is based on a time reference, it is highly susceptible to the timing error in the absence of a LoS component as there would be errors due to multiple signal reflections [82]. Moreover, the response delay within the source unit may be difficult to estimate accurately in practice.

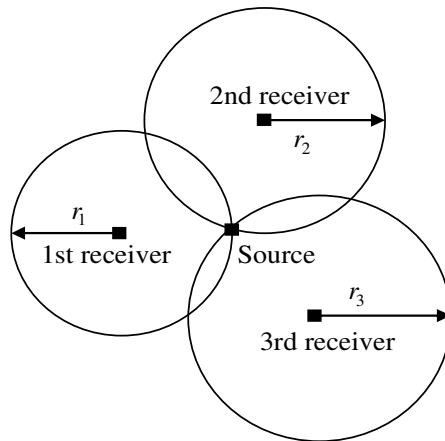


Figure 7.2: Example of ToA based position location using three receivers.

7.1.3 Time Difference of Arrival Based Location Estimation

The time difference of arrival (TDoA) technique is based on the difference in the arrival times of the signal from the source at multiple receivers [83], [84]. This is usually accomplished by taking a snapshot of the signal at multiple receivers over a synchronized time period. For example, if there are two receivers, then two versions of the received signal at those receivers are cross-correlated. The time difference of the signal arriving at the receivers is obtained from the peak of this cross correlation output. Once the time difference is estimated, it can be used to define a hyperbola between the two receivers assuming that the source and the receivers are coplanar. If this procedure is repeated with another receiver along with any of the previously used receivers, another hyperbola results and the intersection of these two hyperbolas estimates the location of the source. As illustrated in Figure 7.3, the TDoA method is also referred to as the *hyperbolic position location method*.

This method has the relative advantage over the ToA method in the sense that the TDoA does not require any time reference to estimate position. However, it requires strict synchronization between receivers. Since any timing bias is cancelled in the differential time measurement, the TDoA methods outperforms the ToA methods in the absence of LoS propagation between the source and the receivers. However, additional timing delay due to NLoS propagation may still result in errors in TDoA based location measurements.

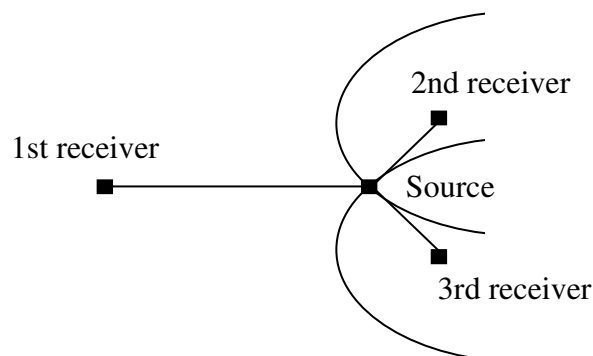


Figure 7.3: Hyperbolic position location scheme.

7.1.4 Hybrid Techniques

In hybrid techniques (HT) [81], two or more of the techniques discussed in the previous sections are combined to create a more reliable position location scheme. For example, when the AoA and ToA methods are combined, it is possible to locate a source even when only one receiver receives the signal with sufficient strength. An AoA or TDoA method fails in such a scenario and the only solution is a hybrid scheme. Similarly, the AoA and the TDoA methods can be combined to form an AoA-TDoA hybrid scheme. With this scheme, multiple receivers receive signals from the source unit and the AoA estimates from each receiver and the TDoA estimates between multiple receivers are combined to determine the target location. Usually, the combination of the AoA and the TDoA method improves the position location accuracy. However, it is important to combine the results from the two methods in a way so that errors from one method do not degrade the overall performance [81].

Even though most of these techniques are based on simple geometric principles, they all require a line of sight (LoS) reference between the transmitter and the receiver to provide acceptable accuracy. However, in wireless communications, especially in indoor environments, radio channels often suffer from multipath and non-line of sight (NLoS) propagation, making the triangulation method used by these techniques hard to employ. The limitation of traditional techniques has motivated us to explore an alternate approach that is based on the location specific channel characteristics. The HMM based position location technique will be discussed in the section.

7.2 HMM-Based Position Location

In this section, we present a novel position location technique based on HMM. For a given location, the received power delay profile (PDP) [85] provides significant information regarding the surrounding environment. For example, the presence or absence of the line of sight (LoS) component, multipath component strengths, number of

multipath components, delay spread etc., can be derived from the power delay profile and are all indicative of the ambient environment. The basic idea of this method is to extract the location dependent features from the PDP and use the HMM to exploit the “signature” statistics to identify locations. Since the statistical features are derived from the PDP, the proposed technique does not depend on the LoS reference and works equally well for all types of indoor and outdoor environments.

Neither the HMM nor its application as a classifier is novel. In fact, similar techniques have been successfully implemented in speech recognition [9]. However, to the best of the authors’ knowledge, this idea has not been extended to position location applications. Several position location schemes have been presented in [86]-[89] using different signature or fingerprint information. For example, in [86] received signal strengths (RSS) have been used as a signature pattern, while channel impulse response (CIR) or angular power profile (APP) has been utilized in [87]-[89]. In most of these methods, an artificial neural network (ANN) has been employed as a pattern matching algorithm. Next, the HMM based classification scheme is described as it applies to position location.

7.2.1 System Architecture

Successful design of a location classifier entails two fundamental steps:

- 1) Extraction of a set of features that ensures highly discriminatory attributes between locations and
- 2) Selection of a suitable classification model.

In the first step, the required features from the PDP are extracted by using the Fourier Transform (FT). Next, vector quantization (VQ) [90] is used to convert the features into a discrete observation sequence. The development of the HMM classifier requires two steps: training and testing.

In the training step, a separate HMM is constructed for each location of interest. As depicted in Figure 7.4, the PDP for the k -th location is converted to a feature vector

which is mapped to a discrete observation sequence, $\mathbf{O}$, by the VQ. After the HMM training, the resultant parameter set $\lambda_k (= \mathbf{A}, \mathbf{B}, \pi)$ compactly characterizes location k . The training step is necessary as it ensures that the location under consideration is best characterized by the corresponding HMM.

In the testing (classification) step, the observation sequence resulting from the PDP of the location to be estimated is matched against all the HMMs previously trained for different locations. Each HMM in the database characterizes a unique location and outputs the probability score of producing the given observation sequence under the corresponding model, $P(\mathbf{O} | \lambda)$. The HMM with the highest probability score estimates the location. The HMM based position classification is illustrated in Figure 7.5.

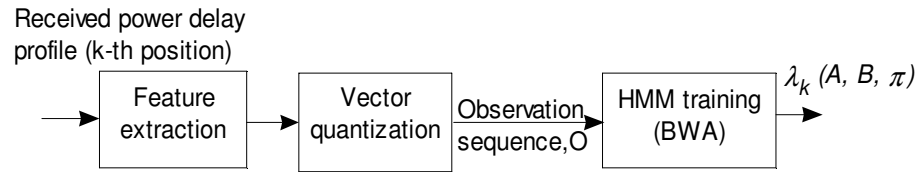


Figure 7.4: Training of HMM for a given location.

7.2.2 Feature Extraction

The location-specific features from the PDP are extracted with the help of the Fourier Transform (FT). The FT converts a time domain waveform into frequency components [91]. For a discrete time signal $x(n)$ of N points, the discrete Fourier transform (DFT) is given by

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j \frac{2\pi nk}{N}}, \quad 0 \leq k \leq N-1 \quad (7.1)$$

where $X(k)$ denotes the k -th spectral sample. For $N = 2^l$, l is an integer, the Fast Fourier Transform (FFT) efficiently implements the DFT computations.

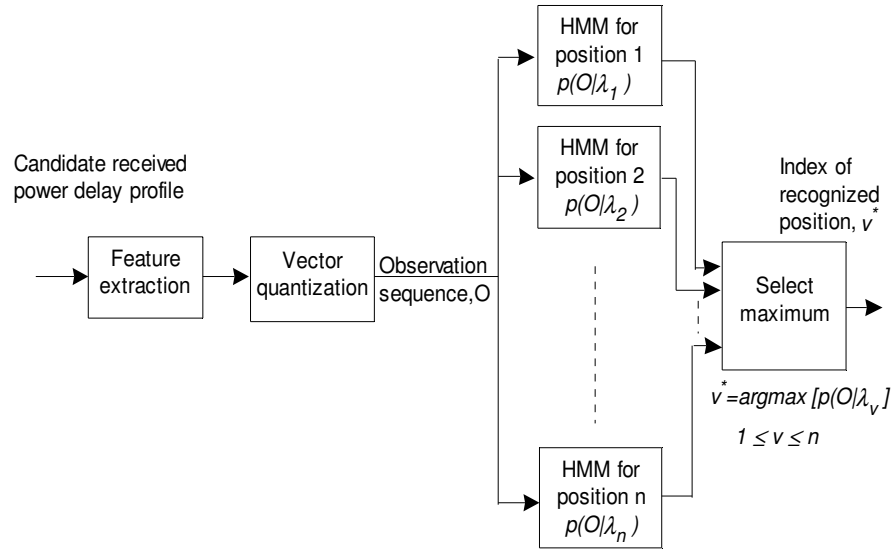


Figure 7.5: Testing of HMM-based location classifier.

In the feature extraction step, first the received PDP is recorded for a given location. Using an appropriate threshold, the noise floor from the PDP is removed in order to differentiate between multipath components and thermal noise. Also, the time scale is adjusted so that the first path starts at time zero. An example PDP is shown in Figure 7.6. The frequency response of the channel is created through the FT of the PDP. Next, three dimensional feature vectors are generated with the magnitude and phase of the frequency response and the excess delay of the received PDP in each direction. The excess delay information is added to achieve additional discrimination in the time domain that is not captured by the FT. The feature vectors encapsulate the time, frequency and phase information from the PDP and preserve the location specific signature statistics. Note that other transformation techniques (i.e., wavelet transformation [92]) may be used to extract the time and frequency domain features as well. Also, if available, additional information (i.e., angle of arrival or received signal strength) may be added as a new dimension to the feature vectors to create unique location-specific attributes.

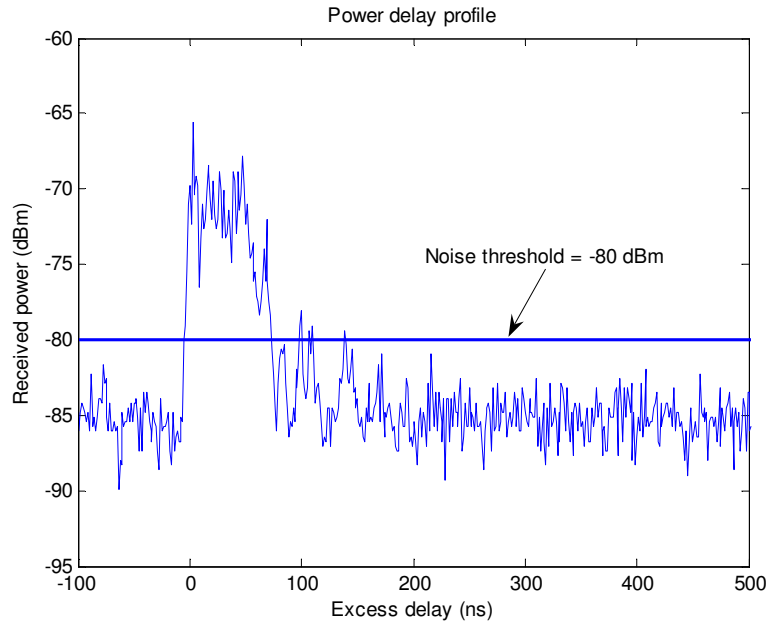


Figure 7.6: An example power delay profile (PDP) (from [95]).

Source: C. R. Anderson, “Design and implementation of an ultrabroadband millimeter-wavelength vector sliding correlator channel sounder and in-building multipath measurements at 2.5 & 60 GHz”, M. Sc. Thesis. © C. R. Anderson, 2002. Used with permission.

7.2.3 Vector Quantization

Once the feature vector is formed, it is used to optimize the HMM parameter set. However, since the discrete observation density is required to train HMMs¹, feature vectors obtained in the previous section cannot be applied directly. In fact, a quantization step is required to map the “continuous” feature vectors into a discrete observation sequence. Vector quantization (VQ) [90] is an efficient way of representing multi-dimensional feature vectors by a small set of vectors, which is referred to as a *codebook*. The VQ process consists of two steps; first, design of the appropriate codebook vectors from a representative training set and, second, using the codebook set, assigning a label to each of the input feature vectors.

In this method, the codebook vectors were determined using an iterative LBG algorithm [93]. The training data set for the algorithm was obtained from the feature

¹ As before, only discrete HMMs are considered here as they are computationally more efficient and don’t require any assumptions on the underlying distribution of the observation symbols.

vectors extracted from the received PDP of several different locations. The LBG algorithm basically partitions the training data set into M disjoint clusters (encoding regions) and represents each cluster by a single vector ($\mathbf{c}_m, 1 \leq m \leq M$). The clustering of training vectors is illustrated in Figure 7.7.

Let us consider a training set of L vectors $\tau = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_L\} \in \mathcal{R}^k$, where $\mathbf{x}$ is a feature vector. The algorithm iteratively finds the codebook vector, $C = \{\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_M\}$, and the partition of space $P = \{S_1, S_2, \dots, S_M\}$, where S_m is the encoding region corresponding to the code vector $\mathbf{c}_m$, so that the average distortion is minimized.

$$D_{avg} = \frac{1}{L \cdot k} \sum_{l=1}^L \|\mathbf{x}_l - Q(\mathbf{x}_l)\|^2 \quad (7.2)$$

where $Q(\mathbf{x}_l) = \mathbf{c}_m$ if $\mathbf{x}_l \in S_m$.

In order to minimize the average distortion, the following two criteria must be satisfied [94]:

1) Nearest neighbor condition: The encoding region S_m should consist of all vectors that are closer to $\mathbf{c}_m$ than any other code vectors.

$$S_m = \{\mathbf{x} : \|\mathbf{x}_l - \mathbf{c}_m\|^2 \leq \|\mathbf{x}_l - \mathbf{c}_{m'}\|^2, \forall m = 1, 2, \dots, M, \forall l = 1, 2, \dots, L\} \quad (7.3)$$

The encoding regions are called *Voronoi* regions which partition the entire space, $\mathcal{R}^k$, so that

$$\bigcup_{m=1}^M S_m = \mathcal{R}^k, \quad \bigcap S_m S_n = \emptyset, \forall m, n, m \neq n \quad (7.4)$$

2) Centroid condition: The code vector $\mathbf{c}_m$ should be an average of all the training vectors that are in the encoding region S_m .

$$\mathbf{c}_m = \frac{\sum_{\mathbf{x}_l \in S_m} \mathbf{x}_l}{\sum_{\mathbf{x}_l \in S_m} 1} \quad (7.5)$$

Once the codebook vectors are obtained from the training algorithm, the continuous feature vectors are mapped to codebook indices. The mapping is done by finding the

index of the codebook vector, i^* , that results in the minimum distance from the feature vector $\mathbf{x}$.

$$i^* = \arg \min_{1 \leq i \leq M} (\|\mathbf{x} - \mathbf{c}_i\|^2) \quad (7.6)$$

Thus, instead of dealing with multidimensional feature vectors, the statistical matching is based on scalar index sequence. The discrete output from the VQ forms the observation sequence for the HMM, making the training of the HMM more efficient. Note that, since the size of the codebook is equivalent to the number of distinct observation symbols in the HMM, a bigger codebook increases computations in the HMM training even though it reduces the distortion error in the VQ.

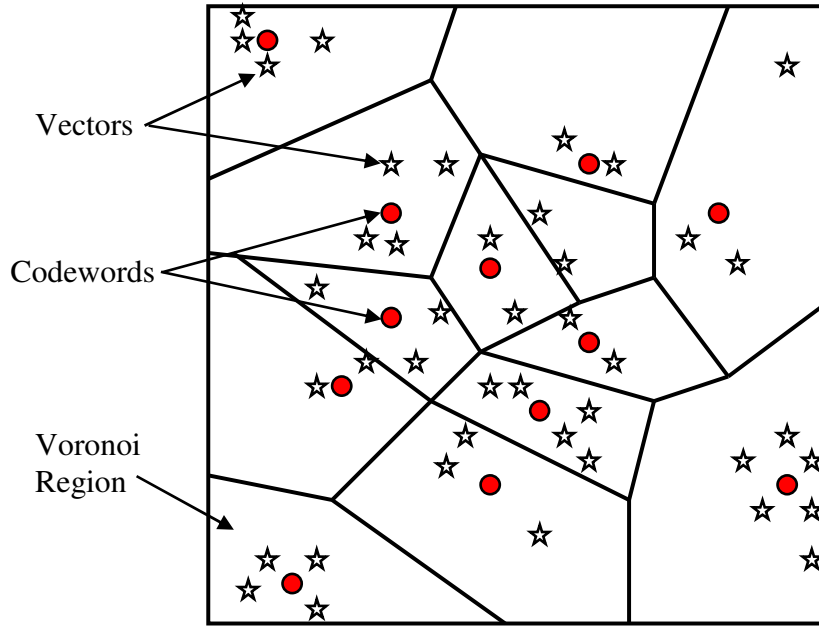


Figure 7.7: An example vector quantization in a two dimensional space. A codeword (circle) is obtained by satisfying (7.3) and (7.5), and represents all the training vectors in the corresponding partition.

7.2.4 HMM-Based Classification

After the feature vectors are formed from the PDP and represented by the codebook index from the VQ, the next step is to extract “signature” patterns from the codebook index sequence. We propose the application of the HMM as a statistical matching tool to

extract patterns since it is a convenient and analytically tractable method to describe the statistical behavior of complicated random time-series. Also, since the HMM is based on probabilistic functions, finding similarity between the test and the reference sequence in the classification stage is very efficient.

Like any other pattern matching algorithm, the implementation of an HMM-based classifier involves two steps; training and detection. Let us assume we have a total of n positions to identify and each position is characterized by a separate HMM. For each of the n positions, we have a training set consisting of an observation sequence, $\mathbf{O} = \{o_1, o_2, \dots, o_T\}$. The observation sequence is generated by the VQ and is representative of the PDP of the corresponding location.

Training: In the training phase, a separate HMM, λ_k , is built for each location, k (Figure 7.4). The objective of the training procedure is to optimize the model parameters $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ for a given training set. In other words, the probability of the observation sequence given the model parameters, $P(\mathbf{O} | \lambda)$, is maximized in the training step. Even though there are several algorithms available for the HMM training, the Baum-Welch algorithm (BWA) [31] is preferred as it is the most common and widely used algorithm.

Detection: Given a bank of HMM models generated for different locations in the training stage, the detection step involves finding the model that is most likely to produce a given observation sequence. As shown in Figure 7.5, after the observation sequence is generated from the PDP via the feature extraction and the VQ steps, the probability function is calculated for all the models, $P(\mathbf{O} | \lambda_v)$, $1 \leq v \leq n$. The model with the highest probability is selected as the best choice to represent the given PDP (location).

$$v^* = \arg \max_{1 \leq v \leq n} (P(\mathbf{O} | \lambda_v)) \quad (7.7)$$

Although there are many conceivable variations and combinations possible, there is no rigorous theoretical approach to determine the optimal topology of the HMM. In most applications, fine-tuning of the topology is performed heuristically based on the training data. In a Bakis model [10] or a *left-right* model, as time increases the states proceed from left to right i.e., $a_{ij} = 0$, when $j < i$. Due to this property, left-right HMMs are

better choices for modeling signals that change over time. So in the position location application, we constrain all the HMMs to have a left-right topology.

7.2.5 Experimental Results

To verify the accuracy of the proposed HMM based approach, the position location scheme was tested on empirical data. A measurement campaign [95] was set up on the fourth floor of Durham Hall at the Virginia Tech campus to investigate the indoor channel profiles. A series of power delay profiles were recorded at various indoor locations at a carrier frequency of 2.5 GHz. All the results presented in this section are based on these power delay profiles. The transmitter and the receiver locations are illustrated in Figure 7.8. Transmitter location 1 (Tx 1) represents the non-line of sight (NLoS) propagation from a room to another room and from a room to a hallway. The transmitter was located in a small laboratory. Receiver locations 1.1-1.3 (Rx 1.1- Rx 1.3) were in adjacent rooms, whereas receiver location 1.4 (Rx 1.4) was in an adjacent hallway. The transmitter and receivers were separated by drywall containing metal studs. Moreover, standard office dry-erase whiteboard was located on the wall separating the transmitter and the receivers at locations Rx 1.1 – Rx. 1.3.

At each receiver location, 40 PDP measurements were recorded with a separation of 30 millimeters (a quarter wavelength) between successive readings. For each location, 20 PDP measurements were used for training the HMM and the rest were used to test the classifier. A four-state left-right HMM was used to represent each receiver location (Rx 1.1-1.4). For the VQ, the codebook size was set to 32, which correspond to 32 unique symbols in the observation sequence of the HMM. The classification result is given in terms of the confusion matrix in Table 7.1. The overall accuracy of the scheme was found to be 95%.

The HMM-based method was compared against the correlation-based position location technique, where 20 PDP measurements for each receiver location were used to create a *template* PDP to represent that location. In the testing phase, the rest of the

measurements were correlated with each template PDP and the template with the highest correlation was declared as the estimated location. Since received power level of the PDP for a specific location is a function of pathloss, a simple time correlation may not be an appropriate metric to distinguish between receiver locations that have similar delay profiles with varying power (for example, Rx 1.1 and Rx 1.2). To avoid such ambiguity, PDPs were correlated based on the squared error normalized by the template PDP values². The normalization ensures that errors in the high energy portion of the PDP are penalized more and thus accounts for the delay distribution. The correlation-based position location results are presented in Table 7.2. The HMM-based position location technique outperforms correlation-based technique.

Table 7.1: Confusion matrix for HMM-based position location scheme for Rx 1.1 – Rx 1.4.

HMM based on Rx location (estimated) →

HMM based on Rx location (true) ↓	Location	Rx 1.1	Rx 1.2	Rx 1.3	Rx 1.4
	Rx 1.1	95%	5%	0%	0%
	Rx 1.2	5%	95%	0%	0%
	Rx 1.3	0%	0%	100%	0%
	Rx 1.4	0%	10%	0%	90%

Table 7.2: Confusion matrix for correlation-based position location for Rx 1.1 – Rx 1.4.

Correlation based Rx location (estimated) →

Correlation based Rx location (true) ↓	Location	Rx 1.1	Rx 1.2	Rx 1.3	Rx 1.4
	Rx 1.1	90%	0%	0%	10%
	Rx 1.2	5%	95%	0%	0%
	Rx 1.3	0%	0%	100%	0%
	Rx 1.4	0%	10%	25%	65%

² The normalized correlation between the template PDP, $\mathbf{x} = [x_1, x_2, \dots, x_L]$, and the candidate PDP, $\mathbf{y} = [y_1, y_2, \dots, y_L]$, was calculated as $\sum_{i=1}^L (x_i - y_i)^2 / x_i$.

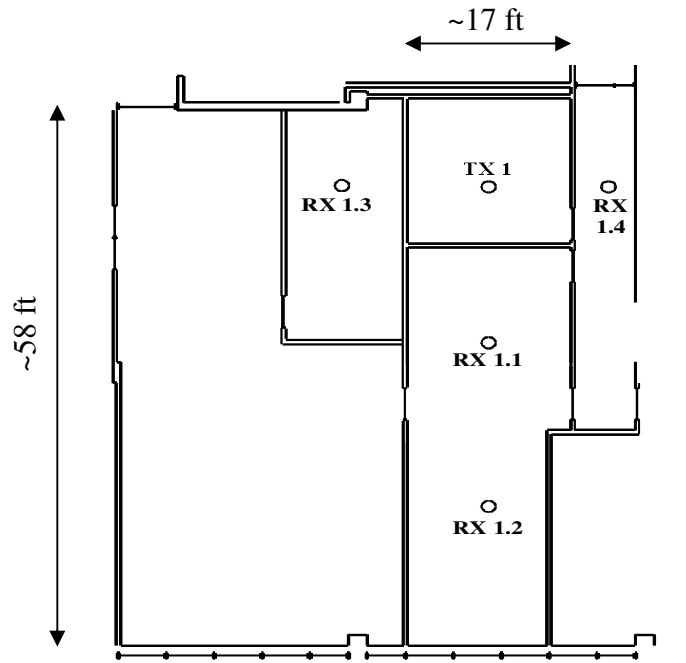


Figure 7.8: Measurement Locations 1.1-1.4, 4th Floor, Durham Hall, Virginia Tech. The transmitter is located in Room 475, Receivers 1.1 and 1.2 are located in Room 471; Receiver 1.3 is in the conference room in the 476 computer lab, and Receiver 1.4 is located in the hallway adjacent to 475 (from [95]). Source: C. R. Anderson, "Design and implementation of an ultrabroadband millimeter-wavelength vector sliding correlator channel sounder and in-building multipath measurements at 2.5 & 60 GHz", M. Sc. Thesis. © C. R. Anderson, 2002. Used with permission.

Similar measurements were taken for transmitter location 5 (Tx 5) and receiver locations 5.1-5.4 (Rx 5.1-Rx 5.4). The floor plan for these locations is illustrated in Figure 7.9. Tx 5 represents a typical office reception area containing a single cubicle, chairs, and office plants. Rx 5.1 represents the NLoS propagation through drywall containing metal studs into an office. Rx 5.2 and Rx 5.3 are on the opposite side of a large clear glass window partially obscured by office plants, while Rx 5.4 is on the opposite side of a large clear glass door. The position location results are given in Table 7.3. Note that the results do not include receiver location 5.1, as the measurement data could be recovered for that location. As before, the HMM based position location tool provides high classification accuracy.

Table 7.3: Confusion matrix for HMM-based position location scheme for Rx 5.2 – Rx 5.4.

		HMM based on Rx location (estimated) →		
		Location	Rx 5.2	Rx 5.3
HMM based on Rx location (true) ↓	Rx 5.2	90%	10%	0%
	Rx 5.3	10%	90%	0%
	Rx 5.4	15%	0%	85%

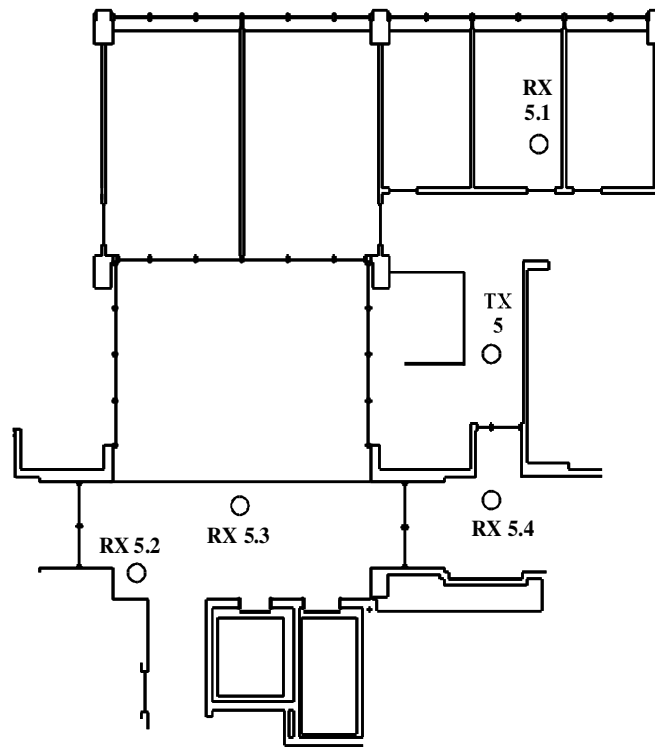


Figure 7.9: Measurement Locations 5.1 – 5.4, 4th Floor, Durham Hall, Virginia Tech. The Transmitter is located next to the receptionist's desk in Room 432. Receiver location 5.1 is in room 433, Receiver locations 5.2 and 5.3 are in the atrium next to the elevators, and Receiver location 5.4 is in the hallway next to 432 (from [95]).

Source: C. R. Anderson, "Design and implementation of an ultrabroadband millimeter-wavelength vector sliding correlator channel sounder and in-building multipath measurements at 2.5 & 60 GHz", M. Sc. Thesis. © C. R. Anderson, 2002. Used with permission.

7.2.6 Effect of Codebook Size

There is a distortion penalty associated with the VQ, since the input vectors in a given partition are represented by corresponding code vector. Obviously, as the number of code vectors (code book size) increases, the quantization error or distortion decreases. However, a large codebook size results in a large number of HMM parameters requiring more memory and computations. The tradeoff between the classification accuracy and the codebook size is illustrated in Table 7.4. As the codebook size increases, the classification accuracy improves. We found that a codebook size of 32 is sufficient to represent the input vectors for this particular application.

Table 7.4: Impact of codebook size on classification accuracy.
HMM based on Rx location (estimated)

HMM based on Rx location (true) ↓	Codebook size	Location	Rx 1.1	Rx 1.2	Rx 1.3	Rx 1.4
	8	Rx 1.1	85%	15%	0%	0%
		Rx 1.2	15%	85%	0%	0%
		Rx 1.3	0%	0%	100%	0%
		Rx 1.4	5%	60%	10%	25%
	16	Rx 1.1	85%	15%	0%	0%
		Rx 1.2	5%	95%	0%	0%
		Rx 1.3	0%	0%	100%	0%
		Rx 1.4	0%	5%	10%	85%
	32	Rx 1.1	95%	5%	0%	0%
		Rx 1.2	5%	95%	0%	0%
		Rx 1.3	0%	0%	100%	0%
		Rx 1.4	0%	10%	0%	90%

The proposed algorithm can be extended to determine a route by pattern matching the sequence of channel profiles. Note that the HMM based technique is based on measurements taken from a single site. However, the extension to multi-site measurements using data fusion has the potential to enhance performance.

The conventional position location algorithms fail to provide acceptable accuracy in environments where there is no line of sight (LoS) component. Since these models are based on the angle of arrival, time of arrival or time difference of arrival, the presence of LoS component is critical in their overall performance. However, the HMM based technique does not depend on the LoS reference and works equally well for all types of indoor and outdoor environments. Unlike complicated pattern matching algorithms, compact models are used to characterize locations which render efficient probability calculations.

7.3 Conclusion

In this chapter, a novel position location application of HMM was introduced. In a wireless system, channel statistics are indicative of the corresponding location characteristics. The HMM based technique allows a location to be estimated based on its channel profile. The conventional position location algorithms fail to provide adequate accuracy in environments where there is no LoS component. Another limitation of existing methods is the requirement of measurements from multiple sites. The proposed method does not depend on the LoS reference and performs well for environments that suffer from multipath and NLoS propagation. Also, the proposed technique can be accomplished based on measurements from a single site making the implementation efficient.

Chapter 8: Conclusions

This chapter provides a summary of the research contributions and accomplishments of this dissertation. The key results from different chapters are discussed briefly. Future research issues are also presented at the end of the chapter.

8.1 Review of the Dissertation

Since its introduction by Baum and Petrie, hidden Markov models (HMMs) have been widely used in diverse applications such as economics, statistics, biology, information theory, pattern recognition, and wireless communications. In this dissertation, HMMs have been proposed as a diagnostic tool to improve cellular services.

HMMs are analytically tractable tools that are capable of extracting statistical features from a complicated random time-series. Unlike simple Markov chains, HMMs can model non-geometric distributions by explicitly incorporating the gamma or Rayleigh distributed state distributions. Like any other learning algorithm, HMM parameters must be estimated with training data in order to model the underlying statistics of an unknown process. There are several methods for approximating the HMM parameters from given experimental observations. Three such algorithms discussed in this dissertation were the Baum-Welch algorithm (BWA), segmental K -means algorithm and the genetic algorithm (GA). Of these three algorithms, the BWA is preferred as it provides a good complexity/performance trade-off. Since the BWA produces equivalent models for all the

observation sequences in the same training set when initialized identically, it becomes easier to compare HMMs. However, model comparison becomes somewhat tricky when the GA is used since the GA introduces randomness in the parameters which may result in disparate models. Also, when compared against the GA, the BWA is computationally efficient.

Distance measure is often used to compare different HMMs and has widespread application in estimation and classification problems. The Kullback-Liebler divergence (KLD) or *relative entropy* is widely used as a distance metric for hidden Markov processes. Unlike finite state Markov models, there is no closed form solution to compute the KLD between HMMs. A novel heuristic divergence measure was proposed that closely approximates the KLD rate between HMMs. The proposed method was compared against other distance measures for HMMs, including a theoretical upper bound. The complexity of a given HMM depends on the number of states and the observation symbols in the model. The model size may be application specific and is usually determined heuristically. Since there is no analytical formulation to optimize the model structure in most cases, it becomes difficult to analyze the performance and efficiency of the model. From the matrix theory perspective, the HMM parameters have been analyzed to optimize the model structure. The correlation between different states can be predicted using the inverse condition number (ICN) of the augmented matrix. Also, by computing the orthogonal complement of the range space of the augmented matrix, the redundant state or observation symbol can be removed from the model.

Even though HMMs have been widely used to simulate wireless channels, it has not been applied to cellular diagnostic systems. A segmentation approach to synthesize call statistics in a dropped call was presented. The novel use of the HMM as a dropped call prediction tool has been introduced in this dissertation. The link quality of a given call can be tracked by monitoring certain channel related statistics. The frame error sequence is one such metric that represents the link quality as the channel statistics evolve over time. HMMs have been shown to model frame error statistics and utilize this information to predict dropped calls using hypothesis tests. By exploiting the signature statistics in the frame error sequences, it has been shown that the coverage related dropped calls can be classified using HMM based hypothesis tests. Unlike coverage

related dropped calls, non-coverage related dropped calls do not have any signature patterns in their frame error sequences. Almost all of the non-coverage related problems are due to excessive interference. The source of the interference may be different in different classes of dropped calls but that is not reflected in the error bursts of the frame error sequences. So the problem of classifying non-coverage related dropped calls is essentially the same as identifying interference sources. Unfortunately, the interference sources cannot be distinguished by observing error patterns only.

Position location capability is imperative to ensure the safety and efficiency of emergency management workers. Estimating location is also important in the context of cognitive radio and asset management. Most of the traditional position location techniques depend on the triangulation method that requires the line of sight (LoS) reference for acceptable accuracy. However, in many practical applications especially in indoor environments, the LoS reference is not present. As a result, the triangulation method fails in such scenarios. A novel position location algorithm based on HMMs was introduced that overcomes this problem. The algorithm exploits the location specific channel statistics and does not depend on the LoS reference. Results from measured data shows that the proposed method provides high accuracy with minimal complexity.

8.2 Future Research Issues

In this dissertation, the notion of applying hidden Markov models (HMMs) as a cellular diagnostic tool has been introduced. HMMs are suitable for extracting statistics from a complicated random time-series. Since in most wireless applications the channel is dynamically changing with time, the relevant statistics can be modeled efficiently by HMMs. In fact, this idea can be extended to many non-cellular applications that require wireless transmissions. Here are some of the suggestions that may be attempted as future research directions:

- Hand-off failure is one of the major causes of dropped calls. In a cellular system, hand-offs are triggered by setting a threshold on the received signal strength or

pilot signal strength. Since signal strengths can fluctuate over a wide range depending on the surrounding location (i.e., the traffic intersection of a busy downtown area), sometimes the base transceiver station (BTS) does not have sufficient time to process a hand-off request and the call is eventually dropped. In order to avoid such a scenario, it may be possible to employ HMMs in predicting hand-offs. For a given route or road section, HMMs can be used to model the evolution of signal strength. Based on the given model for a specific location, the BTS can be informed ahead of time for a hand-off request.

- In Chapter 6, it was concluded that the coverage related dropped calls can be classified using HMM based hypothesis tests. The classification was accomplished based on the signature patterns in the frame error sequences. However, no such patterns were found for non-coverage related dropped calls. Since most of the non-coverage related dropped calls are caused by different interference sources, it is possible to classify them if we can classify the underlying interference sources. For example, interference sources in pilot pollution and forward link loading are different, and may be distinguished by looking at other call statistics like pilot strength measurement message (PSMM). HMMs may be trained on PSMM to identify such scenarios.
- Cellular diagnostic ideas can be extended to other applications like cognitive radios. HMMs can be applied to monitor the link condition based on some channel quality indicator metric, i.e. the signal to noise ratio (SNR), frame error sequence, received signal strength, etc. By tracking the evolution of a given channel quality indicator metric, HMMs may be able to predict the link quality. Based on the channel quality information, modulation/coding schemes, frequency band/time slots may be assigned to meet a given quality of service (QoS) in near-real time.
- In Chapter 7, for the HMM based position location algorithm, the feature vectors were generated by taking the Fourier transform (FT) of the channel power delay profile. Since the FT does not preserve the time information of the signal, an additional dimension in the feature vector was created using the excess delay of

the channel. Instead of appending the excess delay information, the feature vectors may be created by using the wavelet transformation (WT) of the power delay profile. Wavelet analysis is a variable-sized windowing technique. It allows for the use of long time intervals for lower frequency information and shorter intervals for higher frequency information. The WT is therefore capable of revealing aspects of data that other signal analysis techniques may ignore.

- The position location algorithm introduced in Chapter 7 is based on single site, single antenna measurements. While the single site measurement provides relative advantages over traditional position location techniques, it is possible to improve the performance of the algorithm by including multiple site measurements. Multiple site measurements will result in diversity gain, much the same way as space-time coding. The use of an antenna array is another avenue to explore this algorithm further.
- The performance of the HMM based position location technique was analyzed under different code book sizes and number of states. However, due to the lack of measurement data, we could not study the resolution of the model for a given accuracy. Also, the performance of the proposed algorithm may be extended by incorporating information provided by GPS receivers and building maps stored in a central database.

Appendix A: List of Acronyms

1xEV-DO 1x Evolution with Data Only

1xEV-DV 1x Evolution with Data and Voice

3G Third Generation

3GPP Third Generation Partnership Project

3GPP2 Third Generation Partnership Project2

AMPS Advanced Mobile Phone System

ANN Artificial Neural Network

AOA Angle of Arrival

APP Angular Power Profile

ARQ Automatic Repeat Request

BER Bit Error Rate

bps bit per second

BTM Bearer Traffic Model

BTS Base Transceiver Station

BWA Baum-Welch Algorithm

CDF Cumulative Distribution Function

CDL Call Detail Log

CDMA Code Division Multiple Access

CIR Channel Impulse Response

CRC Cyclic Redundancy Check

DCCH Dedicated Control CHannel

DFT Discrete Fourier Transform

DOA Direction of Arrival

EDGE Enhanced Data rate for GSM Evolution

EIB Erasure Indicator Bit

EM Expectation Maximization

F-ACKCH Forward Acknowledgement CHannel

FCH Fundamental CHannel

FCSS Fast Cell Site Switching

FDD Frequency Division Duplexing

FDMA Frequency Division Multiple Access

FEC Forward Error Correction

FER Frame Error Rate

FFT Fast Fourier Transform

F-GCH Forward Grant CHannel

FM Frequency Modulation

F-PDCCH Forward Packet Data Control CHannel

F-PDCH Forward Packet Data CHannel

F-RCCH Forward Rate Control CHannel

FSM Finite State Machine

FT Fourier Transform

FTP File Transfer Protocol

GA Genetic Algorithm

GLM Good Link Model

GPRS General Packet Radio Service

GSM Global System for Mobile communications

H-ARQ Hybrid Automatic Repeat Request

HDR High Data Rate

HMM Hidden Markov Model

HSCSD High Speed Circuit Switched Data

HTTP Hyper Text Transfer Protocol

ICN Inverse Condition Number

IP Internet Protocol

IS Interim Standard

KLD Kullback-Leibler Divergence

LAC Link Access Control

LoS Line of Sight

MAC Medium Access Control

MC Multi-Carrier

MM Mobility Manager

MS Mobile Station

NAMPS Narrowband Advanced Mobile Phone System

Nextsim Network Extensible Simulator

NLoS Non-Line of Sight

NRTV Near Real Time Video

OSI Open System Interface

PCF Packet Control Function

PCG Power Control Group

PDC Pacific Digital Cellular

PDCH Packet Data CHannel

pdf probability density function

PDP Power Delay Profile

PH Phase-Type

PPP Point-to-Point Protocol

PSK Phase Shift Keying

PSMM Pilot Strength Measurement Message

QAM Quadrature Amplitude Modulation

QIB Quality Indicator Bit

QPSK Quadrature Phase Shift Keying

R-ACKCH Reverse Acknowledgement CHannel

RAN Radio Access Network

RC Radio Configuration

R-CQICH Reverse Channel Quality Indication CHannel

RF Radio Frequency

RLP Radio Link Protocol

R-PDCCH Reverse Packet Date Control CHannel

R-PDCH Reverse Packet Date CHannel

R-REQCH Reverse Request CHannel

RS2 Rate Set 2

R-SPICH Reverse Secondary Pilot CHannel

RSSI Received Signal Strength Indicator

RTT Radio Transmission Technology

SDU Selection and Distribution Unit

SFM Simplified version of Fritchman Model

SHMM Semi-Hidden Markov Model

SNR Signal-to-Noise Ratio

TACS Total Access Cellular System

TCP Transmission Control Protocol

TDD Time Division Duplexing

TDM Time Division Multiplexing

TDOA Time Difference of Arrival

TxPA Transmit Power Amplifier

UB Upper Bound

UMTS Universal Mobile Telecommunication System

USDC United States Digital Cellular

VA Viterbi Algorithm

VoIP Voice over Internet Protocol

VQ Vector Quantization

WAP Wireless Application Protocol

WCDMA Wideband Code Division Multiple Access

WT Wavelet Transform

Appendix B: Derivation of Log-likelihood Function

In this appendix, the log-likelihood function of the Baum-Welch algorithm (BWA) is derived. The *forward variable*, $\alpha_t(i)$, is defined as the probability of partial observation sequence $o_1, o_2, \dots, o_t$ (until time t) and state $q_t = s_i$ at time t , given the model, λ .

$$\alpha_t(i) = P(o_1, o_2, \dots, o_t, q_t = s_i \mid \lambda) \quad (\text{B.1})$$

The forward variables are initialized as

$$\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N \quad (\text{B.2})$$

Now the *scaled* forward variable is defined as

$$\hat{\alpha}_1(i) = C_1 \alpha_1(i), \quad 1 \leq i \leq N \quad (\text{B.3})$$

where $C_1 = 1 / \sum_{i=1}^N \alpha_1(i)$ is the scaling constant.

For $t = 2$, the scaled forward variable is given by

$$\begin{aligned} \hat{\alpha}_2(j) &= \frac{\sum_{i=1}^N \hat{\alpha}_1(i) a_{ij} b_j(o_2)}{\sum_{j=1}^N \sum_{i=1}^N \hat{\alpha}_1(i) a_{ij} b_j(o_2)}, \quad 1 \leq j \leq N \\ &= C_1 C_2 \sum_{i=1}^N \alpha_1(i) a_{ij} b_j(o_2) \\ &= \left(\prod_{\tau=1}^2 C_\tau \right) \alpha_2(j) \end{aligned} \quad (\text{B.4})$$

where $\alpha_2(j)$ is the unscaled forward variable and the corresponding scaling constant is

$C_2 = 1 / \sum_{j=1}^N \sum_{i=1}^N \hat{\alpha}_1(i) a_{ij} b_j(o_2)$. Similarly, by induction, it can be shown that

$$\hat{\alpha}_T(i) = \left(\prod_{\tau=1}^T C_\tau \right) \alpha_T(i), \quad 1 \leq i \leq N \quad (\text{B.5})$$

Given the definition in (B.1), $P(\mathbf{O} | \lambda)$ can be obtained in terms of the forward variables as

$$P(\mathbf{O} | \lambda) = \sum_{i=1}^N \alpha_T(i) \quad (\text{B.6})$$

Now by summing over all possible states, (B.5) can be rewritten as

$$1 = \left(\prod_{\tau=1}^T C_\tau \right) P(\mathbf{O} | \lambda) \quad (\text{B.7})$$

where the left hand side equals to unity as the variables are scaled. Since for long observation sequence $P(\mathbf{O} | \lambda)$ is very small, it is usually determined as

$$\log[P(\mathbf{O} | \lambda)] = - \sum_{t=1}^T \log C_t \quad (\text{B.8})$$

where the base of the logarithm is arbitrary.

Appendix C: Hypothesis Test for Markov Chains

Statistical hypothesis testing [80] is one of the basic tools of model building. Setting up and testing hypothesis is an essential part of statistical inference. In hypothesis testing, the question of interest is simplified into two competing claims (hypotheses); the null hypothesis, denoted by H_0 , against the alternative hypothesis, H_1 . The null hypothesis represents a assumption that is to be used as a basis for the argument, while the alternative hypothesis is a statement of what a statistical hypothesis test is set up to establish. For example, in a clinical trial of a new drug, the competing claims may be as follow:

H_0 : There is no difference between the new and current drug.

H_1 : There is difference.

In testing the claims, special consideration is given to the null hypothesis. This is due to the fact that the null hypothesis relates to the statement being tested, whereas the alternative hypothesis relates to the statement to be accepted when the null hypothesis is rejected. Once the test has been carried out, the final conclusion is usually given in terms of the null hypothesis. We either 'reject H_0 in favor of H_1 ' or 'do not reject H_0 '. Note that, if we conclude 'do not reject H_0 ', this does not necessarily mean that the null hypothesis is true, it *only* suggests that there is not sufficient evidence against H_0 in favor of H_1 . However, rejecting the null hypothesis suggests that the alternative hypothesis may be true.

There are two types of errors involved in a hypothesis testing set up. Type I error (false alarm) occurs when the null hypothesis is rejected even though it is in fact true; i.e., H_0 is wrongly rejected. Based on the previous example, type I error would occur if we concluded that the two drugs produced different effects when in fact there was no difference between them.

Type II error (misdetection) occurs when the null hypothesis, H_0 , is not rejected while it is in fact false. For example, a type II error would occur if it was concluded that the two drugs produced the same effect, that is, there is no difference between the two drugs, when in fact they produced different ones. If the Type I error is the chance of crying wolf, then the Type II error is the chance of not seeing a real wolf! Possible results of any hypothesis testing are summarized in Table C.1.

Table C.1: Summary of hypothesis testing.

Decision	Truth	
	H_0	H_1
Reject H_0	Type I error	Right decision
Don't reject H_0	Right decision	Type II error

To test a hypothesis, we need some criteria of closeness, $w(x)$, between the experimental data and the null hypothesis. If the criteria value is greater than some *critical value*, w_{cr} , the null hypothesis is rejected. The *critical value* is a threshold to which the value of the test statistic is compared to determine whether or not the null hypothesis is rejected. The sample space for the test statistic is partitioned into two regions. The *critical region*, or *rejection region*, is a set of values of the test statistic for which the null hypothesis is rejected. It is defined as

$$D_0 = \{\mathbf{x} : w(\mathbf{x}) > w_{cr}\} \quad (\text{C.1})$$

So, if the observed value of the test statistic is a member of the critical region, we conclude 'reject H_0 '. The complement of the rejection region, $\bar{D}_0$, is usually called the *acceptance region*. The probability of Type I error can be computed as,

$$\Pr(\mathbf{x} \in D_0 | H_0) = \alpha \quad (C.2)$$

Similarly the probability of Type II error can be calculated as

$$\Pr(\mathbf{x} \notin D_0 | H_1) = \beta \quad (C.3)$$

The Type I error probability α is also called the *significance level*. A low value of significance level is desirable to protect the null hypothesis. Usually significance level is chosen to be 5%.

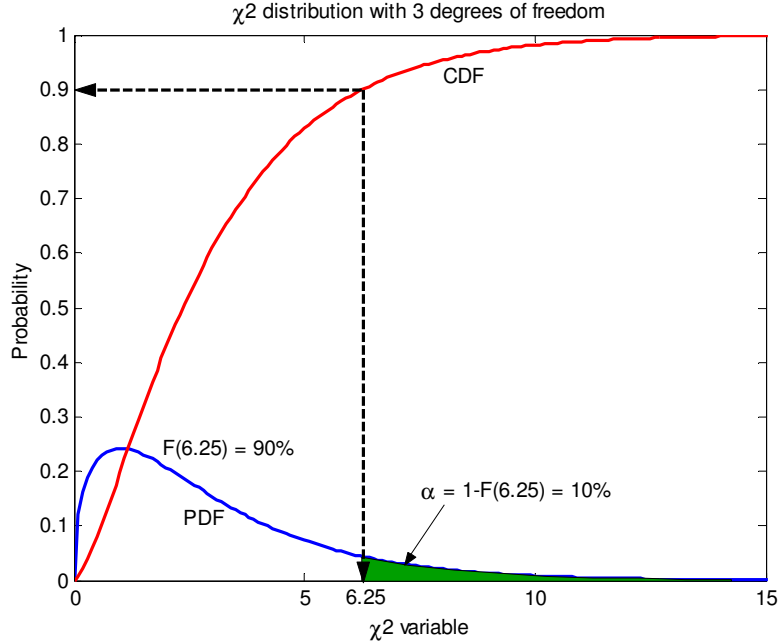


Figure C.1: Hypothesis testing and significance level.

The *power* of a statistical hypothesis test measures the test's ability to reject the null hypothesis when it is actually false, i.e., to make a correct decision. In other words, the power of a hypothesis test is the probability of not committing a Type II error. It is calculated as

$$power = 1 - \beta \quad (C.4)$$

The maximum power a test can have is 1, the minimum is 0. Ideally we want a test to have high power, close to 1. Usually power is chosen to be 0.8. A test's power improves with the sample size. The only way to improve the power of the hypothesis test without increasing the false alarm probability is to use more samples.

Figure C.1 demonstrates how significance level is calculated from probability distribution. Suppose an experiment is run whose result follows a chi square distribution with 3 degrees of freedom. As shown in Figure C.1, 6.25 is the 90th percentile of this distribution, where F denotes the cumulative distribution. We expect to have $\chi^2 \leq 6.25$ 90 times out of 100. For $\alpha=10\%$, if $\chi^2 \leq 6.25$ the null hypothesis is *true*. There is only 10% chance that $\chi^2 > 6.25$. So if $\chi^2 > 6.25$, we can be 90% sure that the null hypothesis is *false*. Thus we can quantify the "doubt" about hypothesis tests in terms of the significance level, α .

Intuitively, we want to define a critical region to minimize the probabilities of both types of errors. However, it is not possible to make these errors arbitrarily small simultaneously since, for any given data set, Type I and Type II errors are inversely related. A Type I error is often considered to be more serious, and therefore more important to avoid, than a Type II error. So the critical region D_0 is selected to minimize the Type II error for some small fixed Type I error. This reflects the best tradeoff between the risk of detection failure and the unnecessary use of the resources associated with a false alarm.

C.1 Hypothesis Testing for Markov Chain

A sequence of random variables $y_1, y_2, y_3, \dots$ is called a finite Markov chain of order k if

$$\Pr(y_t = b_t \mid y_{t-1} = b_{t-1}, \dots, y_1 = b_1) = \Pr(y_t = b_t \mid y_{t-1} = b_{t-1}, \dots, y_{t-k} = b_{t-k}) \quad (\text{C.5})$$

for any $y_t, y_{t-1}, \dots, y_1$ and $t > k$; $b_t, b_{t-1}, \dots, b_1$ are chosen from a finite set Ω , usually $b_t \in \Omega = \{1, 2, \dots, s\}$. In other words the probability that the Markov chain is in state b_t at time instant t depends only on k previous states. Different sequences of $y_{t-1}, y_{t-2}, \dots, y_{t-k}$ constitute states of the Markov chain which are defined as k -tuples $i = (y_{t-1}^i, y_{t-2}^i, \dots, y_{t-k}^i)$. The transition probability from state i to state j is given by

$$p_{ij} = p_{(y_{t-1}^i, y_{t-2}^i, \dots, y_{t-k}^i)(y_t^j, y_{t-1}^j, \dots, y_{t-k+1}^j)} = \begin{cases} p_{y_t, y_{t-1}, \dots, y_{t-k}}, & \text{if } y_{t-1}^i = y_{t-1}^j, \dots, y_{t-k+1}^i = y_{t-k+1}^j \\ 0, & \text{otherwise} \end{cases} \quad (\text{C.6})$$

where $p_{y_t, y_{t-1}, \dots, y_{t-k}}$ is defined as the probability of y_t at time t given that $y_{t-1}, y_{t-2}, \dots, y_{t-k}$ occurred at time $t-1, t-2, \dots, t-k$, respectively; or in other words $p_{y_t, y_{t-1}, \dots, y_{t-k}} = \text{Pr}(y_t | y_{t-1}, y_{t-2}, \dots, y_{t-k})$. These transition probabilities along with the chain order, k , must be estimated from the experimental data.

Hypothesis testing is used to estimate the chain order. To set up the test, we need a null hypothesis which will be tested against an alternative hypothesis. The two hypotheses may be expressed as

H_0 : The Markov chain has order $k-1$, i.e., mathematically

$$H_0 : p_{y_t, y_{t-1}, \dots, y_{t-k}} = p_{y_t, y_{t-1}, \dots, y_{t-k+1}}, \quad \forall y_t, y_{t-1}, \dots, y_{t-k} \quad (\text{C.7})$$

H_1 : The Markov chain has order k , i.e., mathematically

$$H_0 : p_{y_t, y_{t-1}, \dots, y_{t-k}} \neq p_{y_t, y_{t-1}, \dots, y_{t-k+1}}, \quad \forall y_t, y_{t-1}, \dots, y_{t-k} \quad (\text{C.8})$$

Let $y_1, y_2, \dots, y_n$ be the outcome of some experiment where n is the length of the observation sequence. If hypothesis H_0 is true then the likelihood function is given by [80], [96]

$$\begin{aligned} L &= p(y_1, y_2, \dots, y_{k-1}) \prod_{m=k}^{n-k+1} p_{y_{m+k-2}, \dots, y_{m+1}, y_m} \\ &= p(y_1, y_2, \dots, y_{k-1}) \prod_{b_1, \dots, b_k} p_{b_1, b_2, \dots, b_k}^{f(b_1, b_2, \dots, b_k)} \end{aligned} \quad (\text{C.9})$$

where $f(b_1, b_2, \dots, b_k)$ is the frequency of k -tuples $b_1, b_2, \dots, b_k$ in the observed sequence $y_1, y_2, \dots, y_n$. The logarithmic likelihood function is given by

$$\ln L = J_0 + J_1 \quad (\text{C.10})$$

where $J_0 = \ln p(y_1, y_2, \dots, y_{k-1})$ and $J_1 = \sum_{b_1, \dots, b_k} f(b_1, \dots, b_k) \ln p_{b_1, b_2, \dots, b_k}$.

The maximum likelihood parameter estimates $\hat{p}_{b_1, \dots, b_k}$ for large n are those that maximizes J_1 and can be found as [25]

$$\hat{p}_{b_1, \dots, b_k} = \frac{f(b_1, b_2, \dots, b_k)}{f(b_1, b_2, \dots, b_{k-1})} \quad (\text{C.11})$$

It is shown that the hypothesis χ^2 criterion is given by

$$\chi_{hyp}^2 = \sum_{b_1, b_2, \dots, b_{k+1}} f(b_1, b_2, \dots, b_k) \left(\hat{p}_{b_1, b_2, \dots, b_{k+1}} - \hat{p}_{b_2, b_3, \dots, b_{k+1}} \right)^2 / \hat{p}_{b_2, b_3, \dots, b_{k+1}} \quad (\text{C.12})$$

If the hypothesis H_0 is true then this criteria has a limiting χ^2 distribution with N Degrees of Freedom (DoF). The value of N is calculated according to

$$\begin{aligned} N &= \text{number of free parameters in Markov chain of order } k \\ &\quad - \text{number of free parameters in Markov chain of order } k-1 \\ &= s^k (s-1) - s^{k-1} (s-1) = s^{k-1} (s-1)^2 \end{aligned} \quad (\text{C.13})$$

where s is the cardinality of the observation Alphabet $|\Omega|$. The null hypothesis H_0 is rejected if $\chi_{hyp}^2 > \chi_{(1-\alpha), N}^2$, where α is the significance level of the hypothesis test and N is the DoF calculated using (C.13).

C.2 Numerical Examples

In this section a numerical example will be presented applying the hypothesis test discussed in the previous section. Suppose we have a random process generated from 1st order Markov chain. However, in practice, the true order is unknown and must be estimated from empirical data. The number of possible values the observation sequence can take on is fixed to three, i.e. $s = 3$. To set up the hypothesis, first a null hypothesis is assumed that the data is obtained from a 0-th order chain. The null hypothesis is tested against the alternative hypothesis that the chain is of 1st order. By applying (C.12), the hypothesis χ_{hyp}^2 criterion is calculated from the observed sequence. The theoretical χ^2 distribution has a DOF of $s^{k-1} (s-1)^2 = 12$, where the true order is $(k-1) = 1$. If the null hypothesis is true then the hypothesis χ_{hyp}^2 distribution should follow the theoretical χ^2 distribution. The test is illustrated in Figure C.2.

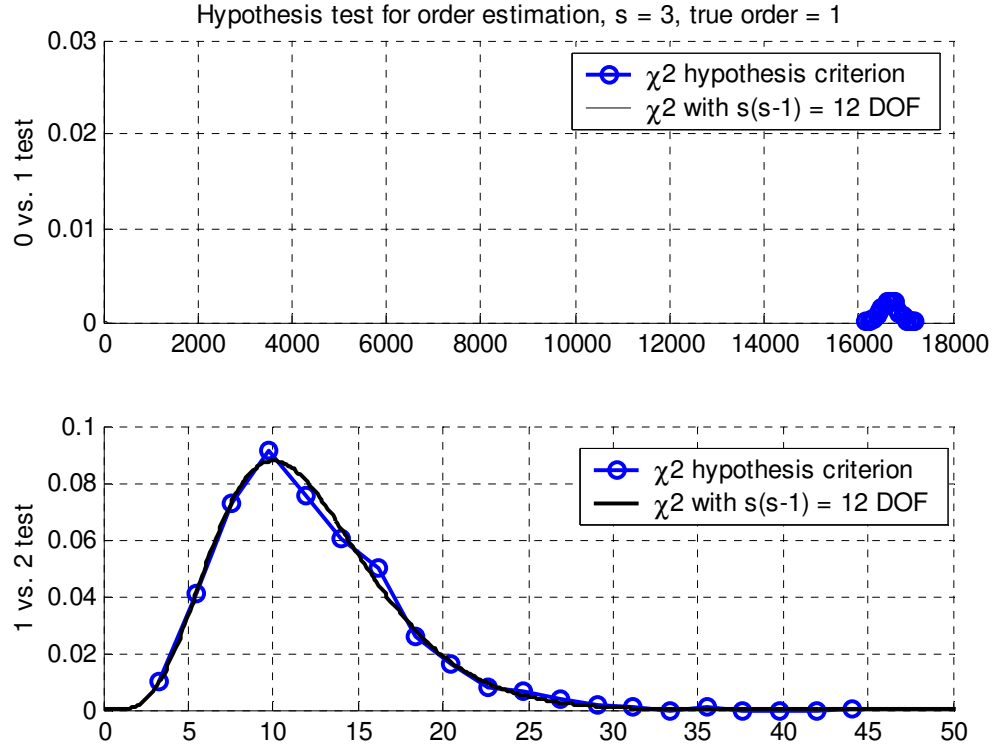


Figure C.2: Hypothesis testing for 1st order Markov chain with $s = 3$.

As seen in Figure C.3, for the test between 0 vs. 1 order test, the χ^2_{hyp} distribution is found to be way above the predicted theoretical χ^2 distribution. So the null hypothesis that the chain is of 0 order is rejected. Now for the next step, the null hypothesis is assumed that the data is from a 1st order Markov chain. This hypothesis is compared against the alternative hypothesis that this is a 2nd order chain. As before the χ^2_{hyp} distribution is fitted against the theoretical χ^2 distribution. It is obvious that there is a close match between these two distributions. So the null hypothesis is accepted. The estimated order is 1 which is indeed the true order. Since a good match is found the test is terminated at this point.

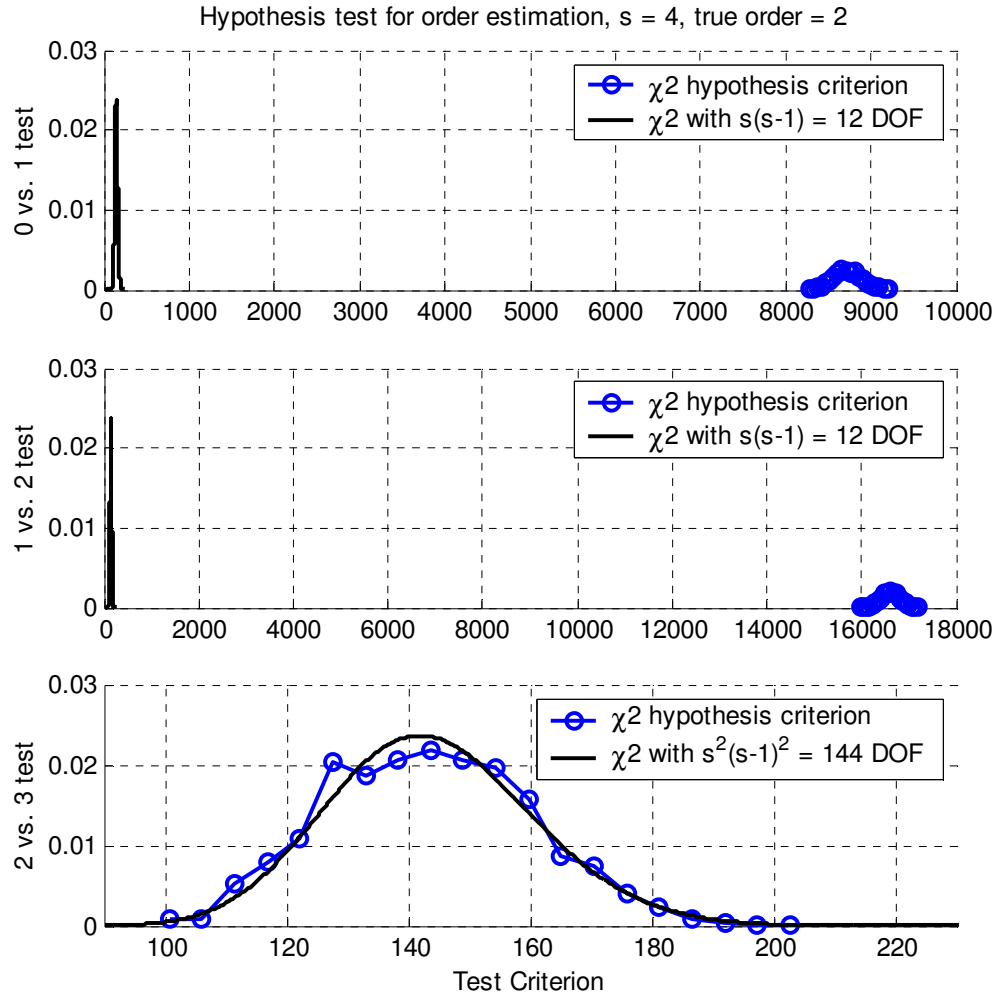


Figure C.3: Hypothesis testing for 2nd order Markov chain with $s = 4$.

In the second example, we consider a 2nd order Markov chain with $s = 4$. The results are depicted in Figure C.3. Note that since the true order is 2, three hypothesis tests were carried out before a close match is found between χ^2_{hyp} distribution and theoretical χ^2 distribution. In this case the DOF for the predicted χ^2 distribution is 144.

Bibliography

- [1] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, Englewood Cliffs, New Jersey, 1993.
- [2] R. A. Howard, *Dynamic Probabilistic Systems, Volume I: Markov Models*. John Wiley and Sons Inc., 1971.
- [3] E. Cinlar, *Introduction to Stochastic Processes*. Prentice Hall, Englewood Cliffs, New Jersey, 1975.
- [4] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 1985.
- [5] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Statist.*, vol. 37, pp. 1554-1563, 1966.
- [6] T. Petrie, "Probabilistic functions of finite state Markov chains," *Ann. Math. Statist.*, vol. 40, no. 1, pp. 97-115, 1969.
- [7] L. Rabiner and B.-H. Juang, "An introduction to hidden Markov models," *IEEE ASSP magazine*, pp 4-16, Jan. 1986.
- [8] W. Turin, *Performance Analysis of Digital Transmission Systems*. New York: Computer Science Press, 1990.
- [9] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286, 1989.
- [10] R. Bakis, "Continuous speech word recognition via centi-second acoustic states," Proc. ASA Meeting, Apr. 1976.
- [11] F. Jelinek, "Continuous speech recognition by statistical methods," *Proc. IEEE*, vol. 64, pp. 532-536, Apr. 1976.
- [12] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell Syst. Tech. J.*, vol. 39, pp. 1253-1266, Sept. 1960.

- [13] E. O. Elliot, "Estimates of error rates for codes on burst-noise channels," *Bell Syst. Tech. J.*, vol. 42, pp.1977-1997, Sept. 1963.
- [14] L. N. Kanal and A. R. K. Sastry, "Models for channels with memory and their applications to error control," *Proc. IEEE*, vol. 66, no. 7, pp. 724-744, July 1978.
- [15] F. Swarts and H. C. Ferreira, "Modeling and performance evaluation of mobile VHF radio channels employing FSK, DPSK, QPSK, and 8-ary PSK as modulation scheme," *Proc. IEEE GLOBECOM '91*, pp. 1935-1939, Dec. 1991.
- [16] B. D. Fritchman, "A binary channel characterization using partitioned Markov chains," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 221-227, Apr. 1967.
- [17] C. Pimentel and I. F. Blake, "Modeling burst channels using partitioned Fritchman's Markov models," *IEEE Trans. Veh. Technol.*, vol. 47, no 3, Aug. 1998.
- [18] T. Sato, K. Tokuda, M. Kawabe, and T. Kato, "Simulation of burst error models and adaptive error control scheme for high speed data transmission over analog cellular systems," *IEEE Trans. Veh. Technol.*, vol. 40, no. 2, pp.443-452, 1991.
- [19] S. Tsai and P. S. Schmied, "Interleaving and error burst distribution," *IEEE Trans. Commun.*, vol. COM-20, no. 3, pp. 291-296, 1972.
- [20] F. Swarts and H. C. Ferreira, "Markov characterization of digital fading mobile VHF channels," *IEEE Trans. Veh. Technol.*, vol. 43, no. 6, pp. 977-985, 1994.
- [21] R. A. Howard, *Dynamic Probabilistic Systems, Volume. II: SemiMarkov and Decision Processes*. John Wiley and Sons Inc., 1971.
- [22] W. Turin and M. M. Sondhi, "Modeling Error Sources in digital channels," *Selected Areas in Communications, IEEE Journal on*, vol. 11, issue 3, pp. 340-347, Apr. 1993.
- [23] S. Sivaprakasam, and K.S. Shanmugan, "An equivalent Markov model for burst errors in digital channels", *IEEE Trans. Commun.*, vol. 43, pp. 1347-1355, Feb. 1995.
- [24] A. Umber and P. Diaz, "A genetic radio channel emulator to evaluate higher layer protocols in a CDMA system," *Proc. on 11th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC 2000)*, pp. 401-405, Sept. 2000.
- [25] W. Turin, *Digital Transmission Systems: Performance Analysis and Modeling*. McGraw-Hill, 1999.
- [26] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, pp. 79-86, 1951.

- [27] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc.*, vol. 76, pp. 341-353, 1977.
- [28] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, pp. 47-60, Nov. 1996.
- [29] S. Kullback, *Information Theory and Statistics*. John Wiley & Sons, New York, 1959.
- [30] W. I. Zangwill, *Nonlinear Programming: A Unified Approach*. Prentice Hall, 1969.
- [31] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Statist.*, vol. 41, no. 1, pp. 164-171, 1970.
- [32] B.-H. Juang and L. Rabiner, "The segmental K -means algorithm for estimating the parameters of hidden Markov models," *IEEE Trans. ASSP*, vol. 38, no. 9, pp. 1639-1641, Sept. 1990.
- [33] G. Florez-Larrahondo, S. Bridges, and E. Hansen, "Incremental estimation of discrete hidden Markov models based on a new backward procedure", *Proc. IAAAI-05*, pp. 758-763, July 2004.
- [34] G. D. Forney, Jr., "The Viterbi algorithm," *Proceedings of the IEEE*, vol. 61, pp. 268-278, Mar. 1973.
- [35] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimal decoding algorithm," *IEEE Trans. Informat. Theory*, vol. IT-13, pp. 260-269, Apr. 1967.
- [36] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.
- [37] C. Chau, S. Kwong, C. Diu, and W. Fahrner, "Optimization of HMM by a genetic algorithm," *Proc. ICASSP-97*, vol. 3, pp. 1727-1730, Apr. 1997.
- [38] F. Sun and G. Hu, "Speech recognition based on genetic algorithm for training HMM," *Electronics Letters*, vol. 34, pp. 1563-1564, Aug. 1998.
- [39] K. J. Won, A. Prügel-Bennett, and A. Krogh, "Training HMM structure with genetic algorithm for biological sequence analysis," *Bioinformatics* 20, pp. 3613-3619, Dec. 2004.
- [40] F. Qin, "Restoration of single-channel currents using the segmental K -means method based on hidden Markov modeling," *Biophys. J.* 86, pp. 1488-1501, Mar. 2004.

- [41] X. Li, M. Parizeau, and R. Plamondon, "Training hidden Markov models with multiple observations – a combinatorial method," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 4, pp. 371-377, Apr. 2000.
- [42] R. Davis, B. Lovell, and T. Caelli, "Improved estimation of hidden Markov model parameters from multiple observation sequences," *International Conference on Pattern Recognition*, vol. 2, pp. 168-171, Aug. 2002.
- [43] J. Kohler, "Multilingual phoneme recognition exploiting acoustic-phonetic similarities of sounds," in *Proceedings of ICSLP*, vol. 4, pp. 2195-2198, Oct. 1996.
- [44] P. Geutner, M. Finke, and A. Waibel, "Selection criteria for hypothesis driven lexical adaptation," in *Proceedings of ICASSP*, vol. 2, pp. 617-620, Mar. 1999.
- [45] B.-H. Juang and L. Rabiner, "A probabilistic distance measure for hidden Markov models," *AT&T Technical Journal*, vol. 64, no. 2, pp. 391-408, Feb. 1985.
- [46] D. Blackwell and L. Koopmans, "On the identifiability problem for function of finite Markov chains," *Ann. Math. Statist.*, vol. 28, pp. 1011-1015, 1957.
- [47] H. Ito, S.-I. Amari, and K. Kobayashi, "Identifiability of hidden Markov information sources and their minimum degrees of freedom," *IEEE Trans. Inform. Theory*, vol. 38, no. 2, pp. 324-333, Mar. 1992.
- [48] S. M. Ali and S. D. Silvey, "A general class of coefficients of divergence of one distribution from another," *J. of the Royal Statistical Society*, 28(1):131–142, 1966.
- [49] C. K. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 462-467, May 1968.
- [50] T. T. Kadota and L. A. Shepp, "On the best finite set of linear observables for discriminating two Gaussian signals," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 278-284, Apr. 1967.
- [51] C. H. Chen, *Statistical Pattern Recognition*. Rochelle Park, NJ: Hayden, 1973.
- [52] J. F. Chamberland and V. V. Veeravalli, "Decentralized detection in sensor networks," *IEEE Trans. Signal Processing*, 51(2):407–416, Feb. 2003.
- [53] D. Kazakos and P. Papantoni-Kazakos, *Detection and Estimation*. Computer Science Press, New York, 1990.
- [54] R. G. Gallager, *Discrete Stochastic Processes*. Boston, MA: Kluwer, 1996.

- [55] Z. Rached, F. Alajaji, and L. L. Campbell, "The Kullback-Leibler divergence rate between Markov sources," *IEEE Trans. Info. Theory*, vol. 50, no. 5, pp. 917-921, May, 2004.
- [56] K. Marton and P. C. Shields, "The positive-divergence and blowing-up properties," *Israel J. Math.*, vol. 86, pp. 331-348, 1994.
- [57] M. N. Do, "Fast approximation of Kullback-Leibler distance for dependence trees and hidden Markov models," *IEEE Signal Proc. Lett.*, vol. 10, no. 4, pp. 115-118, Apr. 2003.
- [58] M. Falkhausen, H. Reininger, and D. Wolf, "Calculation of distance measures between hidden Markov models," in *Proc. of Eurospeech '95*, pp. 1487-1490, 1995.
- [59] L. Xie, V. A. Ugrinovskii, and I. R. Petersen, "Probabilistic distances between finite-state finite-alphabet hidden Markov models," *IEEE Trans. Automatic Control*, vol. 50, no. 4, pp. 505-511, Apr. 2005.
- [60] M. Mohammad and W. Tranter, "A novel divergence measure for hidden Markov models," *Proc. IEEE Southeast Conf.*, pp. 240-243, Apr. 2005.
- [61] M. Vihola, M. Harju, P. Salmela, J. Suontausta and J. Savela, "Two dissimilarity measures for HMMs and their application in phoneme model clustering," in *Proc. ICASSP 2002*, pp. 933-936, May 2002.
- [62] J. Silva and S. Narayanan, "Average divergence distance as a statistical discrimination measure for hidden Markov models," *IEEE Trans. Speech, Audio Processing*, in press.
- [63] N. Vasconcelos, "On the efficient evaluation of probabilistic similarity functions for image retrieval," *IEEE Trans. Inform. Theory*, vol. 50, no.7, pp. 1482-1496, July 2004.
- [64] T. M. Cover and J. A. Thomas, *Element of Information Theory*. Wiley Interscience, New York, NY, 1991.
- [65] Y. Singer and M. K. Warmuth, "Batch and on-line parameter estimation of Gaussian mixtures based on the joint entropy," *Advances in Neural Information Processing Systems 11 (NIPS'98)*, pp. 578-584, 1998.
- [66] V. Balasubramanian, "Equivalence and reduction of hidden Markov models," Tech. Rep. A. I. Technical Report No. 1370, MIT Artificial Laboratory, 1993.

- [67] B. McCane and T. Caelli, "Diagnostic tools for evaluating and updating hidden Markov models," *Pattern Recognition*, vol. 37, no. 7, pp. 1325-1337, July 2004.
- [68] J. Norris, *Markov Chains*. Cambridge University Press, 1997.
- [69] E. Dahlman, B. Gudmundson, M. Nilsson, and J. Skold, "UMTS/IMT-2000 Based on Wideband CDMA," *IEEE Communications Magazine*, vol. 36, no. 9, pp. 70-80, Sept. 1998.
- [70] E. Dahlman, P. Beming, J. Knutsson, F. Ovesjo, M. Persson, and C. Roobol, "WCDMA- The Radio Interface for Future Mobile Multimedia Communications," *IEEE Trans. Veh. Technol.*, vol. 47, no. 4, pp. 1105-1118, Nov. 1998.
- [71] UMTS, "Physical Channels and Mapping of Transport Channels onto Physical Channels (FDD)," 3G TS 25.211 v.3.2.0 (2000-03).
- [72] TIA/EIA IS-856/3GPP2 C.S0024, "CDMA 2000 high rate packet data air interface specification," Std. v.3.0, Dec. 2001.
- [73] 3GPP2 C.S0001, Introduction to cdma2000 Standards for Spread Spectrum Systems, Release C 1.0, May 2002.
- [74] 3GPP2 C.S0002 Physical Layer Standard for cdma2000 Spread Spectrum Systems, Release C 1.0, May 2002.
- [75] 3GPP2 C.S0003 Medium Access Control Standard for cdma2000 Spread Spectrum Systems, Release C 1.0, May 2002.
- [76] 3GPP2 C.S0004, "Signaling Link Access Control Standard for cdma2000 Spread Spectrum Systems," Release C 1.0, May 2002.
- [77] K. Kamdar, R. Nikides, T. Chang, and A. Sinha, "Nextsim: Network extensible simulator," *Motorola S³ Symposium*, July, 2004.
- [78] J. Garcia-Frias and P. Crespo, "Hidden Markov models for burst error characterization in indoor radio channels," *IEEE Trans. Veh. Technol.*, vol. 46, no. 4, Nov. 1997.
- [79] A. Konrad, B. Zhao, A. Joseph, and R. Ludwig, "A Markov-based channel model algorithm for wireless networks," in *Fourth ACM International Workshop on Modeling, Analysis and Simulation of Wireless Mobile Systems (MSWiM)*, pp. 189-199, 2001.
- [80] R. Gray, *Entropy and Information Theory*. Springer-Verlag, 1990.

- [81] T. S. Rappaport, J. H. Reed, and B. D. Woerner, "Position location using wireless communications on highways of the future," *IEEE Communications Magazine*, vol. 34, no. 10, pp. 33-41, Oct. 1996.
- [82] M. P. Wylie, J. Holtzman, "The non-line of sight problem in mobile location estimation," *Int. Conf. Universal Personal Commun.*, pp. 827-831, 1996.
- [83] Y. T. Chan, K. C. Ho, "A simple and efficient estimator for hyperbolic location," *IEEE Trans. Signal Processing*, vol. 42, no. 8, pp 1905-1915, Aug. 1994.
- [84] C.H. Knapp and G.C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-24, no. 4, pp. 320-327, Aug. 1976.
- [85] T. S. Rappaport, *Wireless Communications - Principles and Practice*. Prentice Hall, 1996.
- [86] R. Battiti, A. Villani, and T. L. Nhat, "Neural network models for intelligent network: Deriving the location from signal patterns," *Autonomous Intelligent Networks Systems*, May 2002.
- [87] W. Max and O. Hilsenrath, "Signature matching for location determination in wireless communication systems," US Patent 6,112,095.
- [88] T. Nypan, K. Gade, and T. Maseng, "Location using estimated impulse responses in mobile communication system," *4th Nordic Signal Processing Symposium*, Oct. 2001.
- [89] C. Nerguizian, C. Despins, and S. Affes, "Geolocation in mines with an impulse response fingerprinting technique and neural network," *IEEE Trans. Wireless Commun.*, vol. 5, no. 3, pp. 603-611, Mar. 2006.
- [90] A. Gersho, R. Gray, and R. Gallager, *Vector Quantization and Signal Compression*. Kluwer Academic, 1991.
- [91] J. Proakis and D. Manolakis, *Digital Signal Processing*. 3rd Edition, Prentice Hall, 2000.
- [92] C. Burrus, R. Gopinath, and H. Guo, *Introduction to Wavelets and Wavelets Transforms*. Prentice Hall, 1997.
- [93] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantization design," *IEEE Trans. Commun.*, vol. 28, pp. 84-95, Jan. 1980.
- [94] <http://www.geocities.com/mohamedqasem/vectorquantization/vq.htm>.

- [95] C. R. Anderson, "Design and implementation of an ultrabroadband millimeter-wavelength vector sliding correlator channel sounder and in-building multipath measurements at 2.5 & 60 GHz", *M.Sc. Thesis*, Virginia Tech, 2002.
- [96] M. Bartlett, "The frequency goodness of fit test for probability chains," *Proc. Cambridge Philos. Soc.* Vol. 47, pp. 86-95, 1951.
- [97] P. Billingsley, *Statistical Inference for Markov Processes*. University of Chicago Press, Chicago 1961.

Vita

Maruf Mohammad was born in Dhaka, Bangladesh. He received his Bachelor of Science degree in Electrical Engineering from Bangladesh University of Engineering and Technology (BUET), Bangladesh, in 1999. He joined the graduate program at Virginia Polytechnic Institute and State University in January 2000, and received his Masters degree in 2002. He has been working as a graduate research assistant since 2000 in Mobile and Portable Radio Research Group (MPRG) at Virginia Tech. In 2003, he began pursuing a Doctor of Philosophy degree in Electrical Engineering with MPRG. His research work involved hidden Markov models and their applications in cellular diagnostic systems. He was awarded a University Partnership in Research (UPR) Fellowship by Motorola which supported his doctoral research. In the summer of 2003, he worked as a senior member of technical staff at Cingular Wireless, where he developed link layer simulator for EDGE. The next two summers he interned at Motorola working on the system level simulator for cdma2000 1xEV-DV and UML compliant call processing software.

Maruf is expected to finish his Ph.D. during Fall 2006. His research interests include joint data and channel estimation, equalization, coding, space-time processing, MIMO-OFDM systems, Hidden Markov models. He authored three disclosures and several journal and conference papers. He has also served as a reviewer for the IEEE Transactions of Communications, IEEE Transactions of Communications Letters and a number of conference publications.