INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6” x 9” black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0800

UMI
Structures For Kalman-Based Detection Over Rayleigh Fading Channels

by

Mohammad Javad Omidi

A thesis submitted in conformity with the requirements for the Degree of Doctor of Philosophy, Graduate Department of Electrical and Computer Engineering, University of Toronto

© Copyright by Mohammad Javad Omidi 1998
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.
In The Name of Allah,

The Gracious, The Merciful.
Abstract

Structures For Kalman-Based Detection
Over Rayleigh Fading Channels

Mohammad Javad Omidi

Doctor of Philosophy, 1998
Department of Electrical and Computer Engineering
University of Toronto

Channel estimation is an essential part of many adaptive detection techniques proposed for data transmission over fading channels. For the frequency selective Rayleigh fading channel an autoregressive representation is proposed based on the fading model parameters. The parameters of this representation are determined based on the fading channel characteristics, making it possible to employ the Kalman filter as the best estimator for the channel impulse response.

For IS-136 formatted data transmission, the Kalman filter is employed with the Viterbi algorithm in a Per-Survivor Processing (PSP) fashion and the overall bit error rate performance is shown to be superior to that of detection techniques using the RLS and LMS estimators. To allow more than one channel estimation per symbol interval, Per-Branch Processing (PBP) method is introduced as a general case of PSP.

The Kalman filter can lead to significant improvement in the receiver bit error rate performance. However, a Kalman filter is a complex algorithm and is sensitive to roundoff errors. Different implementation methods are considered for measurement update and time update equations of the Kalman filter. The unit-Lower-triangular-Diagonal Correction (LDC) algorithm is used for the time update equations, and systolic array structures are proposed for its implementation. For the overall implementation of joint data and channel estimation, parallel structures are proposed to perform both the Viterbi algorithm and channel estimation.

A new approach is proposed for the implementation of the Kalman filter based on differential channel states. This leads to a robust differential Kalman filtering algorithm that can be simplified further to ease the implementation without any major loss in performance.
Acknowledgments

First and foremost, praise and glory to Allah, the almighty creator and the eternal source of wisdom and knowledge.

I wish to express my sincere gratitude to my outstanding supervisors Professor Glenn Gulak and Professor S. Pasupathy. I have been very fortunate to benefit from their technical advice, continuous encouragement, moral and financial support, and the friendship that developed over these years in Toronto.

I extend my appreciation to the members of my defense committee, Professors D. Johns, E. Sousa, R. Iravani, and E. Shwedyk for their careful review of my work and their useful comments. I am grateful to all the teachers who I learned from in my life.

Financial support by the Iranian Ministry of Culture and Higher Education and also by ITRC and NSERC throughout this work is gratefully acknowledged.

Thanks to the people in the Electronics Engineering and Computer Group. I am grateful to all of my friends in LP392, especially Ali Sheikholeslami, for their invaluable friendship and the great time we had. I would like to thank Ms. Jennifer Rodrigues and Ms. Sarah Cherian for their help.

I wish to thank Masoud Hashemi and my other friends in Toronto, whose friendship have been a valuable support for me and my family.

I am very grateful to my parents and my brother for their continued love and constant encouragement in all aspects of my life.

Finally, my love goes to my wife Nooshin and my children Ehsan and Navid. I am very grateful to my wife for her patience, understanding and being supportive of my interest to continue my studies. Many thanks to Ehsan and Navid for making my life full of joy and happiness, and for their patience while I had to work.
# Table of Contents

## CHAPTER 1 Introduction
1.1 Wireless Channel Equalization: An Overview .......................... 3
1.2 Motivations ........................................................................ 8
1.3 Outline of Dissertation ..................................................... 10

## CHAPTER 2 Signal Transmission and the Channel Model .............. 12
2.1 The Signal Model .................................................................. 14
2.2 The Channel Model ............................................................. 16
2.3 The AR Model for the CIR .................................................. 20
2.4 The State Space Model ......................................................... 25
2.5 Parameter Estimation Using The Received Signal .................... 27
2.6 Summary ............................................................................. 28

## CHAPTER 3 Channel Estimation Algorithms .............................. 29
3.1 The Kalman Filter ............................................................... 31
3.2 The RLS Algorithm (Kalman Algorithm) .............................. 35
3.3 The LMS Algorithm ............................................................ 38
3.4 Sub-optimal variants of the Kalman filter .............................. 40
3.5 Summary ............................................................................. 40

## CHAPTER 4 Joint Data Detection and Channel Estimation .......... 42
4.1 Maximum Likelihood Sequence Detection .............................. 44
4.2 The Viterbi Algorithm ........................................................ 45
4.3 Sequence Detection with PSP and PBP .................................. 48
4.4 Simulation Results ............................................................... 56
4.5 Summary ............................................................................. 65
CHAPTER 5  Implementing the Estimation Algorithm  66

5.1  Square-root Filtering .................................................. 68
5.2  Implementation Algorithms for Measurement Update Equations 70
5.3  Implementation Algorithms for Time Update Equations ........ 78
5.4  A Systolic VLSI Structure for the LDC Algorithm .............. 82
5.5  Structures for Viterbi Detection and Channel Estimation ...... 86
5.6  Hardware Implementation of the Estimator ...................... 92
5.7  Summary. .................................................................. 95

CHAPTER 6  Differential Kalman Filtering  96

6.1  The Differential Bases for the Kalman filter .................... 98
6.2  Simplifying the Transition Matrix ................................. 101
6.3  Simplifying the covariance matrix ............................... 105
6.4  Square-root Implementation ........................................ 107
6.5  Performance Results ................................................. 107
6.6  Summary. ................................................................. 108

CHAPTER 7  Conclusions and Future Research  110

7.1  Conclusions ............................................................. 110
7.2  Major Contributions .................................................. 111
7.3  Future Research ....................................................... 112

APPENDIX A ................................................................ 114

APPENDIX B ................................................................ 115

APPENDIX C ................................................................ 116

APPENDIX D ................................................................ 117

APPENDIX E ................................................................ 118

REFERENCES ................................................................ 119
List of Figures

2-1  The signal model for the baseband communication system. .................. 15

2-2  Autocorrelation and power spectral density functions in the flat fading channel. The transmitted signal is a single tone. The operator ' * ' denotes convolution. .......................................................... 17

2-3  Power spectral density of the received band-pass signal for an isotropic scattering channel. ................................................................. 18

2-4  (a) Theoretical spectral density of the complex envelope of the received signal; (b) Fading filter frequency response, |P(f)|; (c) |P(f)|², f_p=f_d/1.03. ... 19

2-5  The two-ray fading channel model .................................................. 21

2-6  (a) The combination of raised-cosine filter f(k) and fading channel c(k,k-j), (b) Time variant impulse response of the fading channel. .............. 22

2-7  Illustration of the channel impulse response as the output of the fading filter. ................................................................................. 23

2-8  (a) The tapped delay line model for multi-ray fading channel. (b) equivalent to (a) with the fading filter moved to the output. ..................... 24

2-9  Linear time varying model of signal transmission over a Rayleigh fading channel. ................................................................. 26

3-1  Comparison of the minimum squared error in channel estimation for the RLS algorithm and the Kalman filter. ........................................ 37

3-2  Comparison of the minimum squared error in channel estimation for the LMS algorithm and the Kalman filter. ..................................... 39

4-1  The trellis diagram for the Viterbi algorithm .............................. 47
The adaptive receiver model for joint data detection and channel estimation. ................................................................. 48

Data flow for the computations required on each branch of the trellis in PBP for three channel estimations per symbol interval. ......................................................... 51

Channel adaptation in PBP fashion. After the survivor path is known, its channel estimate will be used as the initial value for channel estimators on all of the outgoing branches. ......................................................................... 52

(a) Data flow for PBP with one sample per symbol interval, which can be reduced to PSP. (b) Data flow for PSP equivalent to (a). ......................................................... 53

The PBP and PSP algorithms ................................................................................................................................................. 55

Block diagram of the simulation system. ......................................................................................................................................... 57

Simulation results with different estimation methods, when the speed of the vehicle is 100 km/h. ......................................................... 58

Simulation results with different estimation methods, when the speed of the vehicle is 50 km/h. ........................................................................................................ 59

The result of changing the estimation rate. ................................................................................................................................. 60

The effect of error in estimation of Doppler frequency shift, using the Kalman Filter. ....................................................................................................................................... 61

The effect of considering lower order fading filters at the receiver. ................................................................................................. 62

The effects of changing \( \lambda \) in the RLS algorithm on the overall BER performance. ................................................................................................................................. 64

The data structure for the inputs and outputs of the Jover-Kailath algorithm [53]. ......................................................................................................................................... 73

A parallel structure for implementation of the RLS Algorithm. ............................................................................................................. 75

The structure of the processor \( \mu_p \) used in the RLS algorithm implementation found in Fig. 5-2 ......................................................................................................................... 76

The pipeline structure for implementing the RLS channel estimator and the branch metric generator. ......................................................................................................................... 77

A systolic VLSI structure for implementation of the LD correction algorithm. ................................................................................................................................. 83
5-6 Mapping the systolic structure of the correction algorithm to a smaller number of processors. ............................................. 85

5-7 The required PSP computation for one node (state) of the trellis diagram. First, the branch metrics are generated and the survivor path is found, then the estimator will receive the survivor path information to estimate the channel. .......................................................... 87

5-8 A parallel architecture for joint data detection and channel estimation (parallel state-computation structure). .................................................. 89

5-9 A parallel architecture for joint data detection and channel estimation (sequential state-computation structure). .................................................. 91

5-10 The effects of changing the wordlength on estimation methods ($E_b/N_0=15$ dB). Kalman-1 represents the Gram-Schmidt method, Kalman-2 is for the correction method, and Kalman-3 is the direct method. PSP method is employed for detection. .......................................................... 93

5-11 The effects of changing the wordlength on estimation methods ($E_b/N_0=15$ dB). See Fig. 5-10 for details. .................................................. 94

6-1 Variation of the coefficients of matrix $F$ with the vehicle speed and sampling rate. (Rayleigh fading channel, Symbol rate = 25 ksymbols/s, 900 MHz band) .................................................. 103

6-2 Variation of the coefficients of matrix $F$ with the vehicle speed and sampling rate. (Rayleigh fading channel, Symbol rate = 25 ksymbols/s, 900 MHz band) .................................................. 104

6-3 MSE for QPSK modulation with Differential Kalman filter, Differential Kalman filter with rounded $E$ matrix, and Differential Kalman filter with reduced $E_k$. (Vehicle speed = 100 km/h) .................................................. 108

6-4 BER for QPSK modulation with Differential Kalman filter, Differential Kalman filter with rounded $E$ matrix, and Differential Kalman filter with reduced $E_k$. (Vehicle speed = 100 km/h, PSP is used for detection) .......... 109
# Table of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACS</td>
<td>Add Compare Select</td>
<td>46</td>
</tr>
<tr>
<td>AR</td>
<td>Autoregressive</td>
<td>14</td>
</tr>
<tr>
<td>ARMA</td>
<td>Autoregressive Moving Average</td>
<td>8</td>
</tr>
<tr>
<td>ASIC</td>
<td>Application Specific Integrated Circuit</td>
<td>1</td>
</tr>
<tr>
<td>ASK</td>
<td>Amplitude Shift Keying</td>
<td>3</td>
</tr>
<tr>
<td>AWGN</td>
<td>Additive White Gaussian Noise</td>
<td>3</td>
</tr>
<tr>
<td>BER</td>
<td>Bit Error Rate</td>
<td>2</td>
</tr>
<tr>
<td>BM</td>
<td>Branch Metric</td>
<td>46</td>
</tr>
<tr>
<td>BMG</td>
<td>Branch Metric Generator</td>
<td>51</td>
</tr>
<tr>
<td>CIR</td>
<td>Channel Impulse Response</td>
<td>4</td>
</tr>
<tr>
<td>CPM</td>
<td>Continuous Phase Modulation</td>
<td>3</td>
</tr>
<tr>
<td>DFE</td>
<td>Decision Feedback Equalizer</td>
<td>4</td>
</tr>
<tr>
<td>DQPSK</td>
<td>Differentially coded Quadrature Phase-Shift Keying</td>
<td>14</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processing</td>
<td>1</td>
</tr>
<tr>
<td>FLOPS</td>
<td>Floating Point Operations Per Second</td>
<td>1</td>
</tr>
<tr>
<td>ILPF</td>
<td>Ideal Low Pass Filter</td>
<td>15</td>
</tr>
<tr>
<td>ISI</td>
<td>Inter-Symbol Interference</td>
<td>1</td>
</tr>
<tr>
<td>LDC</td>
<td>LD Correction</td>
<td>11</td>
</tr>
<tr>
<td>LDU</td>
<td>unit Lower triangular, Diagonal, unit Upper triangular</td>
<td>70</td>
</tr>
<tr>
<td>LMS</td>
<td>Least Mean Square</td>
<td>2</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>MAP</td>
<td>Maximum A Posteriori</td>
<td>4</td>
</tr>
<tr>
<td>MLSD</td>
<td>Maximum Likelihood Sequence Detector</td>
<td>4</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean Squared Error</td>
<td>5</td>
</tr>
<tr>
<td>PBP</td>
<td>Per-Branch Processing</td>
<td>9</td>
</tr>
<tr>
<td>pdf</td>
<td>Probability Density Function</td>
<td>44</td>
</tr>
<tr>
<td>PSK</td>
<td>Phase Shift Keying</td>
<td>3</td>
</tr>
<tr>
<td>PSP</td>
<td>Per-Survivor Processing</td>
<td>7</td>
</tr>
<tr>
<td>QAM</td>
<td>Quadrature Amplitude Modulation</td>
<td>3</td>
</tr>
<tr>
<td>RLS</td>
<td>Recursive Least Squares</td>
<td>2</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal to Noise Ratio</td>
<td>49</td>
</tr>
<tr>
<td>VLSI</td>
<td>Very Large Scale Integration</td>
<td>1</td>
</tr>
<tr>
<td>WGS</td>
<td>Weighted Gram-Schmidt</td>
<td>11</td>
</tr>
<tr>
<td>WSSUS</td>
<td>Wide-sense stationary uncorrelated scattering</td>
<td>15</td>
</tr>
</tbody>
</table>
# List of Symbols

**Symbols**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_m$</td>
<td>Amplitude</td>
</tr>
<tr>
<td>$A, B, C, D$</td>
<td>coefficients of the fading filter</td>
</tr>
<tr>
<td>${a_i}$</td>
<td>complex data sequence</td>
</tr>
<tr>
<td>$b_k$</td>
<td>transmitted information sequence at the sampling intervals</td>
</tr>
<tr>
<td>$b_{ij}$</td>
<td>branch metric from state $i$ to $j$</td>
</tr>
<tr>
<td>$D$</td>
<td>diagonal factor of the LDU decomposition of covariance matrix $P_k$</td>
</tr>
<tr>
<td>$D_k$</td>
<td>nonzero part of the measurement matrix</td>
</tr>
<tr>
<td>$D_p$</td>
<td>diagonal factor of the LDU decomposition of covariance matrix $P_{k</td>
</tr>
<tr>
<td>$E_b$</td>
<td>average bit energy of the transmitted symbol</td>
</tr>
<tr>
<td>$e_k$</td>
<td>error signal at time $k$</td>
</tr>
<tr>
<td>$F$</td>
<td>state transition matrix</td>
</tr>
<tr>
<td>$f_c$</td>
<td>carrier frequency</td>
</tr>
<tr>
<td>$f_d$</td>
<td>maximum Doppler frequency shift</td>
</tr>
<tr>
<td>$f_p$</td>
<td>peak frequency of the fading filter response</td>
</tr>
<tr>
<td>$f(\cdot)$</td>
<td>impulse response of the shaping filter</td>
</tr>
<tr>
<td>$G$</td>
<td>process noise coupling matrix</td>
</tr>
<tr>
<td>$H_k$</td>
<td>the hypothesized transmitted data sequence</td>
</tr>
<tr>
<td>$h_k$</td>
<td>channel impulse response vector at sampling time $k$</td>
</tr>
<tr>
<td>$\bar{h}_k, \bar{h}_k^*, \bar{h}_l$</td>
<td>first, second and third order difference values of $h_k$</td>
</tr>
<tr>
<td>$h_{k,i}$</td>
<td>channel impulse response at time $k$ due to an impulse applied at time $k-i$</td>
</tr>
</tbody>
</table>
$I$ identity matrix
$K_k$ the Kalman gain
$L$ unit lower triangular factor of the $LDU$ decomposition of $P_k$
$L_p$ unit lower triangular factor of the $LDU$ decomposition of $P_{klk}$
$L$ number of taps in the tapped delay line
$l$ channel memory length
$N_o'$ power density of the additive Gaussian noise
$N_o$ variance of the additive Gaussian noise
$n_k$ sample of the additive Gaussian noise at time $k$
$n_s$ number of samples per symbol interval
$P_k$ a priori error covariance matrix at time $k$
$P_{klk}$ a posteriori error covariance matrix at time $k$
$P(.)$ transfer function of the fading filter
$p(.)$ impulse response of the fading filter
$Q$ covariance matrix of $w_k$
$q$ length of the channel impulse response vector
$q_{ij}$ elements of the covariance matrix $Q$
$R_k$ the variance of the estimation error in estimating $z_k$, obtained in Kalman filtering process
$S(.)$ power spectral density
$S_i$ state number $i$
$s$ number of samples per symbol interval
$T$ differencing matrix
$T$ symbol period at the transmitter
$T_m$ total multipath spread in the fading channel
$T_s$ sampling period at the receiver
$U$ unit upper triangular matrix
$V$ vehicle speed
$W$ bandwidth of the received signal
\( w_k \) \( w_k \) a zero mean white Gaussian process defined in ARMA representation of CIR

\( x_k \) \( x_k \) state vector at time \( k \) for the state space model of the fading channel

\( \hat{x}_k \) \( \hat{x}_k \) a priori state estimate at time \( k \) for the state space model of the fading channel

\( \hat{x}_{k|k} \) \( \hat{x}_{k|k} \) a posteriori state estimate at time \( k \) for the state space model of the fading channel

\( x(\cdot) \) \( x(\cdot) \) Gaussian complex random signal in the fading channel model

\( y(\cdot) \) \( y(\cdot) \) Gaussian complex random signal in the fading channel model

\( Z_k(m) \) \( Z_k(m) \) \{\( z_1, z_2, \ldots, z_k, \{a_m\} \}\), the sequence of observation up to and including time \( k \) and the hypothesized transmitted sequence \{\( a_m \}\)

\( z(\cdot) \) \( z(\cdot) \) received signal at the receiver

\( z_k \) \( z_k \) received signal sample at time \( k \)

\( \alpha_0, \alpha_1, \ldots, \alpha_L \) \( \alpha_0, \alpha_1, \ldots, \alpha_L \) the multiplicative fading signal

\( \alpha_{i,k} \) \( \alpha_{i,k} \) real scalar used in simplification of the covariance matrix

\( \mu_{i,k} \) \( \mu_{i,k} \) real scalar used in simplification of the covariance matrix

\( \beta_{i,k} \) \( \beta_{i,k} \) real scalar used in simplification of the covariance matrix

\( \varepsilon \) \( \varepsilon \) rms value of the received signal envelope

\( \Gamma_{i}^{[k]} \) \( \Gamma_{i}^{[k]} \) the path metric in the Viterbi algorithm at time \( k \) and state \( i \)

\( \eta(t) \) \( \eta(t) \) additive complex Gaussian noise

\( \lambda \) \( \lambda \) forgetting factor in the RLS algorithm

\( \Lambda_N^m(\{z_k\}) \) \( \Lambda_N^m(\{z_k\}) \) the likelihood index of the \( N \) symbol sequence \{\( z_k \}\), given the hypothesis \( m \)

\( \eta \) \( \eta \) the step-size in the LMS algorithm

\( \phi \) \( \phi \) wavelength of the carrier frequency

\( \Theta \) \( \Theta \) orthogonal transformation matrix

\( \Theta_{i,j} \) \( \Theta_{i,j} \) as defined in section 5.2

\( \omega_c \) \( \omega_c \) carrier frequency

\( \xi(i) \) \( \xi(i) \) cost function for the RLS algorithm
General Functions and Operators

\[ A^T \] transpose of \( A \)

\[ A^H \] conjugate transpose of \( A \) \((\text{Hermitian transpose})\)

\[ a^* \] complex conjugate of \( a \)

\[ \hat{a} \] estimate of the value \( a \)

\[ E(x) \] expected value (mean value) of \( x \)

\[ G_e[.] \] output of channel estimator expressed as a function of its inputs

\[ \log \] natural logarithm

\[ O(n) \] order of magnitude

\[ \delta(\tau) \] the Kronecker delta function

\[ \kappa(P) \] condition number of matrix \( P \)

\[ \sigma^2_x \] variance of the random process \( x \)

\[ \bar{x} \] expected value (mean value) of \( x \)

\[ |x| \] absolute value

\[ * \] convolution sum

\[ \otimes \] taking minimum

\[ \bullet \] special matrix multiplication defined in section 4.2

General Rules Used in Notations

(i) Vectors and matrices are shown with **Boldface** characters except in the computer programs and algorithms that are explaining the programs.

(ii) In the differential Kalman filter algorithm the same parameters are used as in the conventional Kalman filter, marked with an **Underline**.
Digital transmission of information has overwhelming advantages and is increasingly dominating communication systems. It is a certainty that communication will be essentially fully digital in the near future. The extensive progress in the area of microelectronics has been a major reason for the development of advanced digital communication systems, which allows one to implement complex algorithms economically. With the recent advances in very large scale integration (VLSI) system design more floating point operations per second (FLOPS) are feasible nowadays using digital signal processing (DSP) cores and application specific integrated circuits (ASICs). The algorithms that seemed too complex for implementation in the past are now considered feasible for implementation with reasonable cost and are consequently employed in practice.

The demand for faster and more reliable transmission of digital information over wireless mobile communication systems is ever-increasing. The technology for high-capacity portable and mobile communication systems continues to develop with the goal of enhancing telecommunication mobility and quality. A fundamental limiting factor in the performance of mobile wireless systems is the dispersive and time varying nature of the propagation environment. In mobile wireless channels the major reasons for concern are the intersymbol interference (ISI), and the time-variable behavior of the multipath
channel that results in fading. Intersymbol interference causes each transmitted symbol to be extended over a much longer interval than its original duration [1][2]. Fading occurs when the incoming signals from different paths interfere destructively with each other.

In wireless mobile communication systems, data transmission often takes place in severe ISI conditions and over time-varying and frequency selective fading channels [1][3][4][5][6]. Complex equalization techniques are usually required under these conditions to compensate for the channel behavior and to maintain a reliable communication with acceptable Bit Error Rate (BER) [7][8][9]. Equalization in general consists of estimating the response or states of the channel and using the estimate to compensate for the channel effects so as to improve transmission system performance. To achieve a high quality channel equalization, good channel estimates are required and the performance of the estimator can strongly affect the performance of the receiver.

The Kalman filter is the optimum estimation algorithm and can be employed for the tracking of wireless fading channels [10][11][12]. However, the Kalman filter is complex and computationally demanding. In many practical systems usually the simpler and sub-optimal estimation techniques such as Least Mean Squares (LMS) algorithm [13], or Recursive Least Squares (RLS) algorithm [14] are chosen instead of the Kalman filter to reduce the implementation costs. With recent advances of VLSI technology the algorithms that deemed to be infeasible in the past, are now practical and can be implemented with reasonable costs. Nowadays real-time implementation of the Kalman filter is feasible using parallel and pipelined VLSI structures [15]. The Kalman filter has been widely applied to many practical areas such as aerospace, system control and signal processing. It can also be used for tracking the states of the wireless channel in mobile communication systems.

In this dissertation we study the structure of the wireless receiver, where the Kalman filter is employed along with the Viterbi algorithm to estimate the channel jointly with data sequence detection. The Viterbi algorithm can be considered as a form of matrix multiplication, and this can be used to combine the Viterbi structure with the hardware used to implement the estimation algorithm. In the following we start with an overview of
wireless channel equalization methods. Then we express the motivations behind this research followed by the outline of the dissertation.

1.1 Wireless Channel Equalization: An Overview

Modern portable and mobile digital communication systems require reliable signaling methods over multipath fading channels in the presence of ISI and Additive White Gaussian Noise (AWGN). To overcome the signal distortion caused by the time-varying characteristics of the channel, different classes of equalization techniques have been proposed in the literature for different practical situations. In this section we briefly consider the transmitter types, classification of channel types, and equalization techniques for the compensation of wireless fading channel effects.

The Transmitter

Two major classes of transmitted signals are usually considered in the literature. The first is the class of linearly modulated signals [1][2], where the transmitted symbols are taken from an $M$-ary complex constellation and shaped by the impulse response of a shaping filter. Possible constellations include $M$-ary Amplitude Shift Keying (ASK), $M$-ary Phase Shift Keying (PSK), and $M$-ary Quadrature Amplitude Modulation (QAM).

The second is the class of Continuous Phase Modulated (CPM) signals [1][2], that maintains a constant envelope. The transmitted symbol belongs to the $M$-ary real alphabet and is used to modulate the phase of a sine wave with constant amplitude.

The Wireless Multipath Channel

The mobile digital wireless channel requires high performance equalizers, due to different challenging facts such as the mobility of the transmitter and receiver with respect to each other, and the multipath nature of the propagation environment. To design high performance equalizers, adequate models of the channel are required [1][5] to represent the signal distortions under practical situations.

In the multipath propagation each path delay may be conceptually divided into two parts: the cluster delay, which is on the order of a symbol interval and can be preserved in the channel model, and the fine delay, which is on the order of the carrier period and can
be represented together with the path attenuation as a time-varying complex gain. In addition, each path undergoes a Doppler shift, due to the relative motion of the transmitter and receiver. Thus, the received signal is the sum of many Doppler shifted, scaled and delayed versions of the transmitted signal. The complex envelope of this signal usually obeys a Rayleigh distribution, which is widely used in modeling the channel [4][5][6].

The channel may be considered to have a linear model. For low signaling rates, the channel shows more time selectivity and it is more flat in the frequency domain over the signal bandwidth. This channel is called flat fading, where the received signal is scaled by a complex gain. In the flat fading case, the multipath delay spread is small and the equalization consists of estimating the complex gain of the channel due to the fine delay and compensating for its effect.

At higher signaling rates, the channel is typically frequency-selective, but usually the channel characteristics change very slowly compared to the symbol rate. Equalizers have been historically developed for such channels [16]. In general, channels can be classified into four categories based on their time-selectivity and frequency-selectivity behavior. The most general case for equalization is the time-selective and frequency-selective channel, also known as doubly selective channel. Equalization for the doubly selective channel is a challenging problem. Instead of estimating only one random process, as in flat fading channel, there are many parameters to estimate. This can be in the form of estimating the Channel Impulse Response (CIR) vector, or estimating the adaptive weights for the taps in the tapped delay line model [1][2].

**Channel Equalization**

In the frequency-selective channels, equalization consists of estimating the CIR and then using this information to adjust the parameters of some form of linear or nonlinear filter to compensate for the frequency-selective effects. The linear filter can be in form of a transversal equalizer [16], and the nonlinear filter can be a Decision Feedback Equalizer (DFE) [2][16], a Maximum Likelihood Sequence Detector (MLSD) [17][18], or a Maximum A Posteriori (MAP) type of detector [2][19].
**Linear equalizers:** These equalizers are usually implemented by transversal filters and the tap gains are calculated to invert (approximately) the channel's transfer function and reduce the ISI [1]. A training sequence is often transmitted in this method and the detected symbols are compared to the known training sequence. The result is used to estimate the unknown channel coefficients and acquire the tap weights, then the channel is continually tracked in a decision-directed mode [16]. The decisions should be highly reliable, otherwise a string of decision errors might happen during a deep fade and cause the detector to fail. For the adaptation of channel parameters, different estimation algorithms such as the Kalman filter, the RLS algorithm, and the LMS algorithm can be used.

The linear transversal equalizer can track slow variations in the channel characteristics. Also the phase distortion can be eliminated efficiently by the linear equalizer [16]. However, for channels with severe amplitude distortion and in fast fading channels, more effective and more complex equalization methods such as nonlinear equalizers are required [1][2][16].

**Decision feedback equalizer:** DFE consists of two transversal filters, one is like the linear transversal equalizer and the other feeds back the decisions made on the equalized signal to eliminate the ISI contributed by the symbols already detected. The forward and feedback coefficients may be adjusted simultaneously using an estimation algorithm. DFE filters are normally designed to minimize the mean squared error (MSE) and they usually offer a nice balance of complexity and performance [7][9].

**Maximum a posteriori detector:** Kailath provided a clear understanding of MAP detection of digital signals transmitted through a Gaussian random channel in [20][21] based on an estimator-correlator structure. For an $M$-ary set of digital signals transmitted over a time-varying fading channel, the optimal receiver computes $M$ minimum MSE estimates of the fading distortion, one for each hypothesis. The MAP receiver can be interpreted in two ways: (1) a form of equalizer, since the effects of the time-varying channel are compensated by correlating the fading estimates with the received signal; or (2) a form of partially coherent detection, since the channel phase effect is compensated
by correlating the channel estimates and the received signal. The success of the coherent scheme depends on the quality of the channel estimates.

In general, coherent detection is possible if a reference signal is transmitted with the information bearing signal. This can be in form of a pilot-tone [22] or known symbols known as pilot-symbols (or training sequence) [23][24]. In the literature MAP detectors have been considered for time-varying flat fading channels [25][26] and frequency-selective channels [27].

**Maximum likelihood sequence detector:** Coherent detection for data signals received over an AWGN channel can be achieved by MLSD. The Viterbi algorithm presents an efficient way to overcome the exponential complexity of MLSD. The Viterbi algorithm was first applied to the decoding of convolutional codes by Viterbi in 1967 [28]. At that time Omura observed that it was a special case of dynamic programing solution to the problem of estimating the state sequence of a finite-state Markov process observed in memoryless noise [29]. In 1973, Forney published a comprehensive survey [18] of the Viterbi algorithm reviewing all works related to the algorithm.

In the Viterbi algorithm, the branch metrics are computed based on a comparison between the received signal samples and the expected signal values that are computed based on the channel information. The performance of the receiver strongly depends on how well the estimator can track the rapid changes of the CIR in the fast fading conditions. The number of states in the trellis diagram of the Viterbi algorithm increases exponentially with the length of the channel impulse response; and the implementation of the optimal algorithm would be often highly complex. One suboptimal solution is to perform channel truncation, which attempts to reduce the channel impulse response to a shorter duration. This leaves some residual ISI [30].

In the literature, various kinds of MLSD-based receivers are introduced to combat the degradation of error performance due to the severe ISI in fast fading channels (e.g. [31][32][33][34][35]). Generally all versions of MAP and MLSD receivers require estimates of the channel status. The quality of the channel estimation method has a strong impact on the overall BER performance of the receiver. Particularly in fast fading
conditions, only the more advanced channel estimators can provide reasonable receiver performances.

An inherent difficulty associated with applying the estimation algorithms for channel estimation is that the unknown transmitted data is required for the estimator adaptation. In the “decision directed mode” the actual transmitted data, which is not available at the receiver a priori, is replaced by an estimate of the data stream. However, there is usually a decoding delay, namely a “decision delay” inherent in the Viterbi Algorithm, that causes poor tracking performance of conventional adaptive MLSD receivers on time varying channels.

To reduce the effects of this decision delay three main procedures have been developed in the literature [36]. In one method the CIR is estimated using a fixed delay Viterbi algorithm [37]. This method suffers from a serious degradation in tracking due to the existing decision delay. The second method estimates CIR by an adaptive DFE embedded in the MLSD structure, without any delay in decision estimates [30], but the error propagation problem has a serious effect on the BER performance in this method. The third approach has appeared in many studies (e.g. [32][36][38][39]) and was presented by Tzou, Raheli, and Polydoros in [40][41] and named the Per-Survivor Processing (PSP) method.

**PSP based MLSD**: This method is an adaptive MLSD in which the CIR is estimated along the surviving paths associated with each state of the trellis [40][41][42][43][44]. Each surviving path maintains its own estimate of the channel based on the hypothesized transmitted data sequence. Channel estimates are updated based on the information related to the survivor path which is the best available information.

This approach eliminates the decision delay and its performance is superior to that of other MLSD methods in fading channels and can accommodate somewhat higher Doppler spreads. In PSP, channel estimation is usually performed via LMS or RLS algorithms and it can also be carried out using the Kalman filter [44].
1.2 Motivation

With the rapidly growing use of digital communications, there has been an increased interest in the application of the Viterbi algorithm. It has been applied in many areas of signal processing and communications, and it is considered as an efficient method in the implementation of MLSD decoders. So far, there has been a large number of papers in the literature addressing the implementation issues related to the Viterbi algorithm and several VLSI structures are proposed for this purpose (For example, see [45][46][47][48] and the references therein). In [45], the Viterbi algorithm is formulated as a form of matrix-vector multiplication, suitable for implementation with the systolic array techniques of [15].

In the application of channel equalization for wireless channels the Viterbi algorithm is often accompanied with channel estimation methods, such as the Kalman filter, the RLS algorithm or the LMS algorithm. Lodge and Moher [35] presented an MLSD approach for CPM signals over Rayleigh flat-fading channels by combining Kalman filtering techniques and the Viterbi algorithm. Dai and Shwedyk [31] successfully applied the Kalman filtering technique to the frequency-selective Rayleigh fading channels. In this method a vector autoregressive moving average (ARMA) channel model is used, and a bank of Kalman filters, one for each survivor sequence, is employed for tracking the channel. In [49], Rollins and Simmons extended this work and applied a PSP scheme employing a simplified Kalman filter for data detection over fast fading frequency-selective channels.

There is also a large number of publications devoted to different implementation techniques for the Kalman filter (e.g. [50][51][52][53][54]). These works mostly appear in the areas of signal processing and control, and they study the implementation of the Kalman filter from a general point of view, and not usually for a specific problem. However, in the application of the Kalman filter to the estimation of fading channels we can take advantage of the facts specific to this problem and tailor the implementation for our purpose. For instance, the measurement matrix of the state space model of the fading channel is actually a vector, and this can lead to simplified implementations by employing the techniques that do not require matrix inversion.
The underlying motivation for this work is to study the implementation of the Kalman filter and its performance in an MLSD receiver. The performance of the estimation algorithm is always assessed based on the MSE criteria, but in tracking a channel for data detection the final BER of the receiver should be considered for assessment of the estimator performance. Moreover, it is well known that to apply the Kalman filter for the estimation of the fading channel, a state space model for the channel has to be defined [10][31][49]. However, there has been no established method for relating the parameters of the state space model to the known fading channel model parameters. It is possible to derive this relation [55], and the effects of the state space model parameters on the overall performance can be studied.

In fast fading, the receivers exhibit an irreducible error floor [35][56][57], which means that beyond a certain point increasing the signal to noise ratio does not improve the BER of the receiver. It was found that improvement in the receiver error performance in fast fading is obtained if the detector processes more than one sample per channel symbol [57]. Lodge and Moher developed a multisampling receiver for CPM signals in [35]. Vitetta and Taylor used the multisampling technique for PSK signals in [56] and [57]. However, there is a data dependency in the structure of PSP receivers that makes it impossible to have more than one channel estimation per symbol interval [44]. This has motivated us to look for a more general solution, and propose the Per-Branch Processing (PBP) method [44][58].

On the implementation of the Kalman filter, as stated above, there are many algorithms and structures developed in the literature. In 1986 Jover and Kailath [54] proposed an elegant method to implement the measurement update equations of the Kalman filter. With some extension, this work can be applied to the problem of Kalman estimation for wireless channels and also it can be used for the implementation of the RLS algorithm with simplified complexity [59][60]. The implementation of the Kalman filter has to be supplemented with algorithms that compute the time update equations of the Kalman filter, and VLSI structures are needed to realize them.
The Kalman filter seems to be very promising for channel estimation in the base station receivers, where more complexity in the receiver structure can be tolerated in exchange for better performance. The enhanced performance can lead to a better quality of service, and increased range of coverage. In practical implementations we need to study the required digital word-length in the hardware, and its effects on the stability of the estimation algorithm. Also it can be shown that by using a new formulation for the Kalman filter, based on the differential bases, we can achieve some interesting results in reducing the complexity of the algorithm [61].

The major contributions in this study are:

- Simplification of Clarke's model to a third order fading filter and establishing the mathematical relations between the coefficients of the fading filter and the parameters in the Kalman filtering algorithm.
- Studying the effects of different system parameters of the estimation algorithm, on the overall BER performance of the receiver.
- Proposing the PBP method as a general case of PSP and developing structures for joint data detection and channel estimation using PSP and PBP.
- Proposing several parallel and pipelined VLSI structures for the implementation of the Kalman filter and the RLS algorithm, including the systolic structures for the realization of LDC method.
- Development of the novel differential Kalman filtering method, and studying the resulting simplifications in the implementation of the Kalman filter.

1.3 Outline of Dissertation

The introduction of chapter 1 is followed by six chapters. In Chapter 2 we will present the signal model used throughout this work. Then it is vital to define the channel model that we are targeting, since different channel models lead to different equalizer structures. A state space model for the channel is obtained, based on the approximation of Clarke's fading model [62] by a third order fading filter, and the parameters of the state space model are related to the fading model characteristics.

Chapter 3 briefly reviews the formulation of the Kalman filter, the RLS algorithm, and the LMS algorithm. It is emphasized that the RLS algorithm is a special case of the
Kaiman filter, and that the algorithms for the measurement equations of Kalman filter can be used to implement the RLS algorithm with minor modifications.

The structure of the detector is considered in Chapter 4. In this chapter we introduce the PBP method as a generalized form of PSP for application in a multisampling scheme. In this chapter we study the effects of employing different estimators on the BER performance of the MLSD receiver. We study the effects of error in the estimation of channel model parameters on the overall BER, and also the complexity-performance trade off in considering a lower order model for the channel.

Chapter 5 is devoted to the implementation issues related to the Kalman filter and the RLS algorithm. First, square-root filtering is explained. Then the Jover-Kailath algorithm is extended for the implementation of the measurement update equation, and parallel structures are proposed for the hardware realization. The same algorithm is used for the RLS algorithm and parallel and pipelined structures are introduced for this purpose.

The implementation of time update equation is considered with the Weighted Gram-Schmidt (WGS) method, and another algorithm called the LD Correction (LDC) method. A systolic structure is offered for the LDC method and the performances of competing methods are compared. The overall structure of the receiver for implementation of the Viterbi algorithm along with the channel estimator is studied in this chapter.

Chapter 6 proposes a new approach to define the states in the state space model of the Kalman filter. Differential states are considered instead of the conventional basis and this leads to a reduced complexity in implementation with the same performance. Also, the Kalman filtering algorithm becomes more robust against approximations (rounding) made to reduce the digital wordlength of the algorithm.

Chapter 7 contains conclusions, a summary of major contributions, and suggestions for further study. Finally, some of the algorithms that are explained in the context of the thesis are shown in appendices in the form of computer programs. This has been done to clarify the details of the algorithms and facilitate future studies of these algorithms.
Chapter 2

Signal Transmission and the Channel Model

In a typical mobile radio communication scenario one station is fixed in position and the other is moving. In most of the radio channels of interest, the channel impulse response is the result of many concurring phenomena. The mobility leads to changes in the propagation environment and this may result in a dispersive, time-varying, fading channel.

The line-of-sight path for communication is usually obstructed by buildings or other obstacles; and the propagation will be largely by way of scattering. Multiple propagation paths arise due to reflection and diffraction in the environment. Destructive interference among these paths results in fading, defined as changes in the received signal level in time. The received signal may also experience dispersion, which is the spreading of the signal in time or frequency [1]. For complete characterization of a mobile communication system it is necessary to describe the fading dispersive channel by a model. The channel can be modeled based on the mathematics used to describe its physical properties.

Fading dispersive channels are usually best described as random linear time-varying filters. The received signal consists of the sum of multiple time-variant paths. It is reasonable to assume zero mean Gaussian random statistics for the received signal. With this assumption, characterization of the channel reduces to the specification of either the correlation function of the random channel impulse response or the spectral density of the received signal given the transmitted waveform [63].
Clarke's model for fading channel simulation is widely accepted for the multipath fading environment [62][6][8][4]. This model assumes a dense array of randomly oriented scattering objects located around the mobile unit. All the scatter components arrive with the same amplitude (termed isotropic scattering), and with a uniform distribution of phases and angles of arrival. The addition of phasors with uniformly distributed phase angles will result in a Rayleigh distribution for the magnitude of the complex sum of all the paths. This model provides an accurate representation of the mobile radio signals in urban areas [62][6].

For the simulation of a Rayleigh mobile channel, we need to generate a random process with the desired density function for the fading envelope and a specific Doppler spectrum. One method to generate such a process is to shape the spectral density of a random process by passing it through a filter with specific transfer function. An alternative method proposed by Jakes [6], based on Clarke's model [62], is to assemble a series of oscillators with different frequencies and add the outputs to form the specific spectrum. The above techniques for channel simulation can also be employed to describe the channel model used in the estimation of the channel.

In a physical communication system, channel parameters affect the received signal. These parameters are unknown to the receiver. In order to retrieve the transmitted information, the receiver must estimate these unknown parameters from the received waveform. The estimates are then used as if they were true values and employed for data detection. Mobile communication channels are time-variant and the task of the channel estimator consists of estimating a time-variant set of parameters. Mathematically this problem can be described as estimating a random signal (i.e., the channel impulse response) in a noisy environment. To solve this problem, some estimation algorithms require information about the system model - the channel model in this case - to estimate the random signal. The Kalman filter and many of its variants [14] are model dependent and assume a state space model for the system. In the estimation of the channel impulse response these estimation algorithms assume perfect knowledge of the model parameters.
When the channel impulse response is a slowly time-varying process, the channel model is usually considered as time-invariant.

The application of the Kalman filter to the channel estimation of Rayleigh fading channels has been addressed by some authors [31][64]; however, in these applications a relation between the parameters of the actual fading channel model and the state space model is not established. In this chapter we propose a new method for obtaining an Autoregressive (AR) representation for the impulse response of the fading channel based on the fading model parameters. It will be shown that the state space model parameters can be easily obtained at the receiver by estimating the maximum Doppler frequency shift or equivalently finding the AR spectral estimation of CIR. This enables us to use the optimal Kalman filter consisting of both time and measurement updates for channel estimation. In this method, the fading coefficients are obtained by passing a Gaussian process through an IIR filter. It is shown that the complexity of the Kalman filter depends on the order of the IIR filter and the trade-offs between the complexity and performance are studied. Although a two-ray fading channel is considered here, the complexity of this method and hence the complexity of the Kalman filter will not be affected by increasing the number of rays.

In the following section we start with the signal model and will present our assumptions about signaling and the modulation scheme.

### 2.1 The Signal Model

To study a digital communication system over a frequency-selective Rayleigh fading channel we adopt the North American narrowband TDMA standard (IS-136). The $\pi/4$-shifted differentially coded Quadrature Phase-Shift Keying (DQPSK) modulation technique with a symbol rate of 25 ksymbol/s and an excess bandwidth of 25% is used. For simplicity we will consider the DQPSK signaling scheme instead of the $\pi/4$-shifted mode, which should not lead to significant differences in performance [38]. The complex baseband signal model for the communication system is shown in Fig. 2-1. The complex data sequence $\{a_i\}$ with the symbol period $T$ is the input sequence to the fading channel.
Digital data signals are packed into TDMA blocks starting with a preamble training sequence. The training sequence helps the receiver to extract the necessary information about the channel. The data symbols are shaped in a raised-cosine shaping filter with impulse response \( f(t) \) before transmission.

The equivalent low-pass time-variant impulse response of the Rayleigh fading channel, \( c(t,u) \), represents the channel response at time \( t \) due to an impulse applied at time \( t-u \). In practice the impulse response \( c(t,u) \) is usually modeled as a wide-sense stationary uncorrelated scattering (WSSUS) process \([1][6]\). The assumption of wide sense stationarity is somewhat controversial, since any change in the vehicle speed will affect the maximum Doppler frequency, and this changes the statistics of the channel \([65]\). However, in practical situations the WSSUS model can be adopted, assuming constant vehicle speed for the duration of one or a few data frames. As a rule of thumb, the short term stationarity assumption is valid within a mobile moving distance of about 10-50 wavelengths \([7]\).

The additive noise \( \eta(t) \) is a circularly symmetric \([2]\) complex Gaussian process with power spectral density \( N' \). The received signal is passed through an Ideal Low Pass Filter (ILPF). The filter bandwidth \( W \) is wide enough that for practical purposes all of the signal energy is passed, including the signal energy that is spread by the fading process. Therefore, the bandwidth of the signal \( z(t) \) is \( W \) and it is sampled at Nyquist rate \( (T_s = 1/2W) \). The noise samples \( \eta(kT_s) \) are complex uncorrelated Gaussian random variables with variance \( \text{E}[|\eta(kT_s)|^2]=N_o=2WN' \). In our treatment the channel impulse response
(CIR) includes the impulse response of the cascade of the shaping filter, \( f(t) \), and the fading channel, \( c(t,u) \).

The receiver samples the incoming signal at the rate \( 1/T_s \) at the output of the low pass filter where \( T=sT_s \), and \( s \) is the number of samples per symbol interval. By defining the information sequence at the sampling times as

\[
b_k = \begin{cases} 
    a_k/s & \text{k=integer} \\
    0 & \text{otherwise} 
\end{cases}
\]  

(2-1)

the sampled signal, \( z_k \), can be written as

\[
z_k = \sum_{i=0}^{q-1} b_{k-i} h_{k,i} + n_k
\]  

(2-2)

where \( h_{k,i} \), the CIR at time \( k \) due to an impulse that was applied at time \( k-i \), describes both \( f(t) \) and \( c(t,u) \) blocks of Fig. 2-1 in the discrete time domain. In practical situations it is possible to truncate the CIR to a finite length and we assume its total length to be \( q \). The additive white Gaussian noise, \( n_k \), represents \( \eta(kT_s) \).

### 2.2 The Channel Model

Propagation in urban areas is mainly by way of scattering from the surfaces of the buildings and this makes the mobile communication channel a time varying multipath-fading medium. The multipath spread of the received signal can result in a frequency-selective channel, which means different frequency components are affected differently by the channel. The channel is called frequency-nonselective (flat fading) if the transmitted signal bandwidth is smaller than the coherence bandwidth of the channel [1, ch.14]. In a frequency-nonselective channel the transmitted signal is distorted by a multiplicative random signal, which is a zero-mean complex-valued Gaussian process (Fig. 2-2). In a multipath fading channel the received signal consists of several scattered components and each path is characterized by a time delay. In this situation energy arrives via several paths simultaneously, and various incoming radiowaves arrive from different
FIG. 2-2  Autocorrelation and power spectral density functions in the flat fading channel. The transmitted signal is a single tone. The operator ' denotes convolution.

directions with different time delays. The envelope of the fading process has a Rayleigh distribution and its phase is uniformly distributed over the interval (-\pi, \pi).

For narrow-band signals where the signal bandwidth is much smaller than the carrier frequency, it suffices to derive the characteristics of the received complex low-pass signal by considering the transmission of an unmodulated carrier. In Fig. 2-2 a single tone is transmitted over a flat fading channel. Complex white Gaussian noise is filtered by the fading filter to generate the multiplicative random process. It is clear from Fig. 2-2 that the spectrum of the unmodulated carrier is spread due to the multiplicative signal effect. This is called the Doppler spread and is related to the aggregate of Doppler shifts of multipath components. The power spectral density of the channel output is centered on the carrier frequency and is shaped by the power spectrum of the fading filter.

Clarke's channel model [62] is based on a two-dimensional isotropic scattering model. In this model it is assumed that the plane waves arrive at the antenna from all directions in the (x,y) plane with equal probability. In this model the theoretical power spectral density of the complex envelope of the received signal is represented [62][6] as

$$S(f) = \begin{cases} \frac{e^2}{2\pi f_d} \left[1 - \left(\frac{f - f_c}{f_d}\right)^2\right]^{-1/2} & |f - f_c| \leq f_d \\ 0 & \text{elsewhere} \end{cases}$$

(2.3)
where $\varepsilon$ is the rms value of the signal envelope and $f_c$ is the carrier frequency. The maximum Doppler frequency shift, $f_d$, is given by $f_d = V/\lambda$, where $V$ is the vehicle speed and $\lambda$ is the wavelength of the carrier frequency. The power spectral density $S(f)$ is plotted against the normalized frequency difference $(f-f_c)/f_d$ in Fig. 2-3. The spectrum is centered on the carrier frequency and is zero outside the limits $\pm f_d$ on either side of the carrier. In reality the power spectrum will not go to infinity and the reason for this behavior is that the propagation of the plane waves were assumed in a 2-D plane for the simplicity of computations, where the actual propagation is in 3-D space [5].

Simulation of the fading spectrum appropriate to mobile radio is obtained by choosing an appropriate characteristic for the fading filter in Fig. 2-2 and properly shaping the spectrum of the Gaussian noise processes. As shown in Fig. 2-2, a complex Gaussian noise process is passed through a fading filter to create the multiplicative fading signal. It is important to notice that although the spectrum of the Gaussian processes is affected by filtering, the probability density function is not, so the process at the output of the fading filter remains Gaussian. The spectral density of the received signal envelope is determined
by the transfer function of the fading filter, $P(f)$. To simulate the spectral density of (2-3), one must choose $P(f)$ proportional to the square root of $S(f)$. It is impossible to design a finite order filter whose output spectrum truly follows this shape, so an approximation has to be sought.

A third order fading filter with the impulse response $p(k)$ can be designed so that its output spectral density is an approximation to $S(f)$ [5][8][66]. If white Gaussian noise is applied to the input of the fading filter, the output envelope will have a Rayleigh distribution. The problem of designing a low order fading filter for shaping the spectral density of a white noise signal to be used as the complex envelope of the received signal in simulators is addressed in [66] and the performance of the simulator is shown to agree very closely with theory. The proposed frequency response in [66] is a low pass characteristic with 0 dB gain at lower band, 6 dB peak at $f_p = f_d/r_f$, and -60 dB per decade slope after the frequency $f_p$ as shown in Fig. 2-4. The $r_f$ ratio can be chosen so that the fading filter transfer function curve is a close fit to the theoretical curve. By placing the peak point of $|P(f)|^2$ on the $S(f)$ curve we obtain $r_f=1.03$. With the above constraints the design of the above filter can be completed if the maximum Doppler frequency shift, $f_d$, is
given; and as we discussed in section 2.5, \( f_d \), can be estimated from the received signal. The computer program of Appendix A can be used to calculate the fading filter coefficients given the maximum Doppler frequency \( f_d \).

The above characteristic can be realized by a digital filter. Implementation of this filter can be easily achieved by a third order fading filter. The transfer function in the \( z \) domain can be written as

\[
P(z) = \frac{D}{1 - Az^{-1} - Bz^{-2} - Cz^{-3}} \tag{2-4}
\]

where the filter coefficients depend on \( f_d \). Estimating the maximum Doppler frequency shift suffices to design this filter based on the constraints given in Fig. 2-4, and to obtain the filter coefficients \((A, B, C,\) and \(D)\), which describe the characteristics of the fading channel.

Although this method has been used for the simulation of a multipath fading channel, we will use it to describe a model for the channel and establish a relation between the channel parameters and the parameters of the channel state space model. In the following we will show that the transfer function of (2-4) can be used to derive an AR representation for the CIR based on the parameters of the fading filter. This will in turn help us to define the state space model parameters of the fading channel used in channel estimation. The effects of considering a lower order filter at the receiver for simplifying the channel estimation process is studied in chapter 4.

### 2.3 The AR Model for the CIR

Here we will derive an AR representation for the CIR based on the above fading channel model. The impulse response of a multipath fading channel includes several pulses from different paths with different delays. Associated with each path is a time varying propagation delay and an attenuation factor. Here we will consider the simple case of a two-ray fading channel. The baseband impulse response at time \( t \) caused by an impulse applied at time \( u \) can be written as

\[
c(t, t-u) = \alpha_0(u)\delta(t-u) + \alpha_1(u)\delta(t-u-\tau) \tag{2-5}
\]
FIG. 2-5  The two-ray fading channel model

where $\alpha_0$ and $\alpha_1$ are circularly symmetric Gaussian complex random coefficients. Fig. 2-5 illustrates a model for this channel.

In this context, the CIR is the impulse response of a system including both $f(t)$ and $c(t,u)$ (Fig. 2-1). As shown in Fig. 2-6(a) the shaping filter is considered as a part of the fading channel and the CIR consists of the impulse response of the cascade of these two systems. The response of the fading channel at discrete time $k$ to an impulse applied at time $j$ is shown in Fig. 2-6(b) and can be expressed as

$$c(k, k-j) = \alpha_0(j)\delta(k-j) + \alpha_1(j)\delta(k-j-\tau)$$  \hspace{1cm} (2-6)

The response of this time-variant system to an arbitrary input $u(k)$ can be written as

$$\sum_m c(k, k-m)u(m)$$  \hspace{1cm} (2-7)

For the cascade of two systems we want to find the response to $\delta(k-j)$, or $h_{k,k-j}$. This is equivalent to finding the response of the fading channel to the input signal $f(k-j)$ and from (2-7) we obtain

$$h_{k,k-j} = \sum_m c(k, k-m)f(m-j)$$  \hspace{1cm} (2-8)

Using (2-6), $h_{k,k-j}$ can be expressed as

$$h_{k,k-j} = \sum_m [\alpha_0(m)\delta(k-m) + \alpha_1(m)\delta(k-m-\tau)]f(m-j)$$  \hspace{1cm} (2-9)

$$= \alpha_0(k)f(k-j) + \alpha_1(k-\tau)f(k-j-\tau)$$
FIG. 2-6 (a) The combination of raised-cosine filter $f(k)$ and fading channel $c(k,k-j)$, (b) Time variant impulse response of the fading channel.

and if we define $i = k-j$, (2-9) becomes

$$h_{k,i} = \alpha_o(k) f(i) + \alpha_1(k-\tau) f(i-\tau)$$

(2-10)

On the other hand, from Fig. 2-5 we can see that $\alpha_o(k)$ and $\alpha_1(k-\tau)$ are outputs of the fading filter and can be written as

$$\alpha_o(k) = x(k) * p(k)$$

(2-11)

and

$$\alpha_1(k-\tau) = y(k-\tau) * p(k)$$

(2-12)

where $p(k)$ is the impulse response of the fading filter. Hence (2-10) becomes

$$h_{k,i} = [x(k) * p(k)] f(i) + [y(k-\tau) * p(k)] f(i-\tau)$$

(2-13)

or

$$h_{k,i} = [f(i) x(k) + f(i-\tau) y(k-\tau)] * p(k)$$

(2-14)

and if we define

$$w_{k,i} = f(i) x(k) + f(i-\tau) y(k-\tau)$$

(2-15)

then

$$h_{k,i} = w_{k,i} * p(k)$$

(2-16)
Equation (2-16) suggests that the impulse response of the combination of shaping filter, and the fading channel, can be obtained at the output of the fading filter, if the input is the Gaussian noise process $w_{k,i}$, as shown in Fig. 2-7.

Here we have considered a two ray model; however, this result can be generalized to any number of rays. In a multi-ray condition the CIR can be obtained as the output of the fading filter where the input is a sum of weighted Gaussian noise components, similar to the situation of Fig. 2-7. As in ([1] ch. 14), we can consider a resolution of $I/W$ in the multipath delay profile of the baseband signal. Assuming that the total multipath spread is $T_m$, a tapped delay line model can be obtained with $L=[T_mW]+1$ taps. Fig. 2-8(a) shows such a tapped delay line for a multi-ray fading channel. In each ray a multiplicative fading coefficient is obtained by filtering a white Gaussian noise source. It should be noted that several fading filters on different rays can be replaced with one fading filter as in Fig. 2-8(b) with (2-16) remaining valid.

The CIR, $h_{k,i}$, is a wide-sense stationary Gaussian random signal and has an AR representation ([10] ch. 2). Using (2-16) and given the transfer function of the fading filter $P(z)$ as in (2-4), one can obtain the AR representation of the channel impulse response as

$$h_{k,i} = Ah_{k-1,i} + Bh_{k-2,i} + Ch_{k-3,i} + Dw_{k,i}$$  \hspace{1cm} (2-17)

This shows that the AR representation of the CIR directly depends on the fading filter characteristics. Also, as we mentioned before, to define the fading filter coefficients one only requires to know the maximum Doppler frequency shift, $f_d$. This means if the receiver estimates $f_d$ on a regular basis, like at the beginning of each data frame, it will have the AR representation of the CIR.
FIG. 2-8  (a) The tapped delay line model for multi-ray fading channel.  
(b) equivalent to (a) with the fading filter moved to the output.

In this model it is assumed that all rays experience the same fading spectrum. In reality, the ray with the shortest delay has the (long-term average) spectrum of (2-3), because the fading is typically due to a number of scatterers located close to and around the vehicle. However, the delayed rays are typically due to a large and distant scatterer (e.g., a large building, a cliff face, etc.) and are characterized by a much narrower spectrum. Nevertheless, in practical situations $f_d$ is usually a small fraction of the symbol rate and the assumption that all rays have the same fading spectrum will not result in much loss. This assumption allows for the factorization shown in (2-13) and (2-14), leading to the AR representation of (2-17). An alternative approach to obtain the AR model of (2-17) for the CIR, is to employ one of the spectral estimation methods of ([67] ch. 6) to find the AR model parameters. In the following we will show that the AR model can be used to define the state space model of the fading channel.
2.4 The State Space Model

To derive the state space model for the fading channel consider the $q$ dimensional complex Gaussian random vector at sampling time $k$

$$h_k = (h_{k,0}, h_{k,1}, ..., h_{k,q-1})^T$$  \hspace{1cm} (2-18)

where $(.)^T$ denotes matrix transposition. Using (2-17) we obtain

$$h_k = Ah_{k-1} + Bh_{k-2} + Ch_{k-3} + Dw_k$$ \hspace{1cm} (2-19)

where $I$ is a $q \times q$ unit matrix. The vector $w_k$ is a $q \times 1$ zero mean white Gaussian process with the covariance matrix defined as $E\{w_kw_l^T\} = Q\delta(k-l)$. Here, $\delta(k-l)$ is the Kronecker delta function, and $(.)^T$ denotes Hermitian transposition. According to (2-19), $h_k$ only depends on its three past values; and if we define the states of the state-space model as a vector composed of 3 consecutive impulse responses, then

$$x_k = (h_{k}^T, h_{k-1}^T, h_{k-2}^T)^T$$  \hspace{1cm} (2-20)

Note that considering a higher order approximation to the fading model will result in a higher order AR model and increases the state vector dimension. Using (2-19) and (2-20) we can write

$$x_{k+1} = \begin{bmatrix} A & B & C I \\ I & 0 & 0 \\ 0 & I & 0 \end{bmatrix} x_k + \begin{bmatrix} D I \\ 0 \\ 0 \end{bmatrix} w_k$$ \hspace{1cm} (2-21)

where $I$ and $0$ are $q \times q$-identity matrix and $q \times q$-zero matrix, respectively. The above equation can be written as

$$x_{k+1} =Fx_k + Gw_k$$ \hspace{1cm} (2-22)

where $F$ and $G$ are $3q \times 3q$ and $3q \times q$ matrices respectively. $F$ is called the state transition matrix and $G$ is the process noise coupling matrix.

The $3q \times 1$ vector $H_k$ can be defined as

$$H_k = \begin{bmatrix} b_k, b_{k-1}, b_{k-2}, ..., b_{k-q+1}, 2q \text{ zeros} \\ 0, 0, ..., 0 \end{bmatrix}$$ \hspace{1cm} (2-23)

where $2q$ zeros are inserted after $b_{k-q+1}$. The received signal can be expressed by

$$z_k = H_kx_k + n_k$$ \hspace{1cm} (2-24)
FIG. 2-9 Linear time varying model of signal transmission over a Rayleigh fading channel.

Equations (2-22) and (2-24) describe the linear time varying system of Fig. 2-9 where $x_k$ is the state vector of this system, $H_k$ is called the measurement matrix and the received signal $z_k$ can be assumed to be a noisy measurement of the states of the system.

As mentioned before, the covariance matrix of the Gaussian noise process $w_{k,i}$ is $E\{w_{k,i}w_{i}^T\} = Q\delta(k-l)$ . The matrix $Q$ can be obtained using (2-15). The element on the $i$th row and the $j$th column of $Q$ is

$$q_{ij} = E\{w_{k,i}w_{k,j}\}$$

or

$$q_{ij} = E[[f(i)x(k) + f(i-\tau)y(k-\tau)] \times [f(j)x(k) + f(j-\tau)y(k-\tau)]]$$

The $x(k)$ and $y(k)$ signals are white processes with variances $\sigma_x^2$ and $\sigma_y^2$, therefore $q_{ij}$ is zero for $i \neq j$ and for the diagonal elements of $Q$ we obtain

$$q_{ii} = \sigma_{w_{k,i}}^2 = f^2(i)\sigma_x^2 + f^2(i-\tau)\sigma_y^2$$

Having defined the parameters of the state space model, we are ready to employ an estimation method for estimating the states of the system or the impulse response of the channel.
### 2.5 Parameter Estimation Using The Received Signal

Channel estimation becomes more important at higher fading rates that occur at higher vehicle speed. Table 2-1 shows some typical fading rates for the IS-136 and GSM standards.

<table>
<thead>
<tr>
<th>Speed &amp; Doppler Shift</th>
<th>IS-136 T=1/24,300 seconds</th>
<th>GSM T=1/270,000 seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 (km/h) f_d (Hz)</td>
<td>8.3 \times 10^{-4}</td>
<td>8.3 \times 10^{-5}</td>
</tr>
<tr>
<td>f_d T</td>
<td>6.9 \times 10^{-4}</td>
<td>16.7 \times 10^{-5}</td>
</tr>
<tr>
<td>50 (km/h) f_d (Hz)</td>
<td>41.6 \times 10^{-3}</td>
<td>41.6 \times 10^{-4}</td>
</tr>
<tr>
<td>f_d T</td>
<td>3.4 \times 10^{-3}</td>
<td>3.1 \times 10^{-4}</td>
</tr>
<tr>
<td>100 (km/h) f_d (Hz)</td>
<td>83.3 \times 10^{-3}</td>
<td>83.3 \times 10^{-4}</td>
</tr>
<tr>
<td>f_d T</td>
<td>6.9 \times 10^{-3}</td>
<td>6.2 \times 10^{-4}</td>
</tr>
</tbody>
</table>

In practice, for some channel estimation algorithms, it is necessary to extract the required channel state space model parameters from the received signal. The parameters of the fading filter in (2-4) are used to generate \( F \) and \( G \) and an estimation of these parameters is required at the receiver. Also the matrix \( Q \) and the noise variance \( N_0 \) need to be calculated for the implementation of the Kalman filter.

To obtain \( F \) and \( G \), we need to find the AR model parameters of (2-17). The estimated state vector at the receiver, \( \hat{x}_k \), consists of the estimates of CIR taps, \( \hat{h}_{k,i} \), (see (2-18),(2-20)). The process \( \hat{h}_{k,i} \) is characterized by the AR model of (2-17). There are several spectral estimation methods ([67] ch. 6) that can be employed to find these AR model parameters. The AR spectral estimation provides the parameters of \( F \) and \( G \) and the variance of the AR model noise. As mentioned earlier, assuming the filter characteristics of Fig. 2-4, estimating the AR parameters for the CIR is equivalent to finding the maximum Doppler frequency shift. Hereafter, we refer to this AR spectral estimation as the estimation of maximum Doppler frequency shift.
To obtain $Q$ we notice that it is a diagonal matrix as defined in (2-26). The diagonal elements are the variance of the AR model noise, $\sigma_{w_k,i}^2$, and can be obtained in the process of spectral estimation techniques of [67]. The additive noise variance $N_o$ can be estimated based on a comparison of the detected sequence and the received signal.

2.6 Summary

In this chapter we derived an AR representation for the CIR in the frequency selective Rayleigh fading channel. A third order model is adopted to approximate Clarke's fading model [62]. A relation is established between the fading channel model and its state space model parameters, which allows the implementation of the Kalman filter as the optimum channel estimation technique. This relation does not depend on the number of rays in the model; and describes the impulse response of the fading channel as the response of the fading filter to a random Gaussian input process. The matrices $F$ and $G$ are used in the Kalman filter algorithm and can be obtained by the AR spectral estimation of the estimated CIR.

To calculate the likelihood ratio, the MLSD receiver requires some information about the channel. This information is usually provided in form of an estimate of the channel states, as in (2-20). Different estimation algorithms such as the Kalman filter, the RLS algorithm, and the LMS algorithm may be employed to estimate the channel states. In the following chapter we introduce these algorithms to explain their requirements and their relative performance.
Maximum likelihood sequence detection (MLSD) is the optimum detection technique for digital signals transmitted over mobile channels with intersymbol interference (ISI). In order to perform the task of MLSD, the channel impulse response (CIR) is assumed to be known. The CIR of the mobile channel is time-variant and it needs to be identified using an estimator. Channel estimation can be effectively performed provided that the channel input data are known. In practice, the channel input data are not exactly known. Some methods use a known training sequence for identifying the channel parameters [22][23][24]. In other methods the detected data is used as the known channel input [2] or data detection and channel estimation are performed jointly in the same structure [55][38]. In all of the above methods the quality of the channel estimation method has a strong impact on the overall bit error rate of the receiver. Therefore, a key factor in the receiver design is the estimation of the fading channel with high accuracy [1].

A linear filter can be employed to perform the process of channel estimation [10][12]. The estimating filter extracts information about the channel at time $t$ by using the data measured up to and including time $t$ (i.e. the received signal). The channel is modeled as a system with unknown parameters (states) and the received signal is considered as a noisy measurement of these parameters. Certain statistics, such as mean and correlation functions, of the channel random parameters can be available to the estimator. The task of
the linear filter is to process the received signal as a noisy measurement of the channel states and to minimize the effect of noise at the filter output according to some statistical criterion. A useful approach is to minimize the mean square of the estimation error. The estimation error is the difference between the actual parameter value and the output of the estimator (the estimate).

The Kalman filter is a powerful and widely used solution to the above problem and it has been successfully used in many real-world applications [1][10][14][12][11]. The output of the Kalman filter is computed recursively, and each state update is computed from the new input data and the previous estimate. Therefore, only the previous estimate needs to be stored in the memory and this makes the algorithm suitable for digital implementation.

The mathematical formulation of the Kalman filtering problem can be described based on state space concepts. A state space model can be defined and the estimated parameters are the states of this model. The model itself is assumed to be known to the Kalman filter. This information is in the form of state space model parameters and the statistical knowledge of the system variables. The internal states of the model are determined by an input random process. The Kalman filter is an optimum estimator which receives a noisy measurement of the internal states and provides the minimum mean-squared estimation of the state values based on its knowledge of the system model and the received signal.

In the application of channel estimation for mobile fading channels, the state space model of the channel has to be known for the implementation of the Kalman filter. Obtaining this information is not trivial and it has to be extracted from the received signal. When the system model information is not available sub-optimal methods such as the recursive least-squares (RLS) family of adaptive filters can be employed. The Kalman filter provides a common framework for deriving all of these adaptive filters. In [14] Sayed and Kailath showed that several different variants of the RLS algorithm can be directly related to the Kalman filtering problem. The optimum Kalman filter requires the exact parameters of the state space model and the second order statistics of the random model-parameters. The RLS algorithm is a special case of the Kalman filter where the...
required information about the state space model are simply replaced by constant values [14].

Another sub-optimal solution when the channel state space model is not available is the LMS algorithm [13]. The LMS algorithm has been widely used in practice due to its simplicity. It can be implemented in a practical system efficiently while it does not need squaring, averaging, or differentiation. The LMS algorithm uses a special estimate of the error surface gradient to update its state estimate.

The performance of the receiver strongly depends on how well the estimator can track the rapid changes of the CIR in the fast fading conditions. In practice, channel estimation is usually performed via LMS or RLS algorithms. However, the Kalman filter is the optimum estimation method that minimizes the mean square estimation error. The Kalman filter is a complex and a computationally demanding algorithm and this has limited its usage in some real-time applications. Nowadays, with the recent significant developments in VLSI technology the processing power of digital signal processors has grown dramatically and this makes the implementation of the Kalman filter more feasible and practical at the data rates found in commercial personal communication systems.

In this chapter we briefly introduce the Kalman filtering algorithm for the estimation of mobile channel impulse response, and also we present the RLS and LMS algorithms as channel estimation methods. The relation between the Kalman filter and the RLS algorithm will be studied and the tracking performance of the above algorithms in channel estimation will be compared.

### 3.1 The Kalman Filter

The Kalman filter [68] is an optimal linear minimum variance estimator, introduced by R. E. Kalman in 1960. Since then it has been widely applied in academic, industrial, military, and aerospace engineering areas such as system control, navigation and guidance, signal processing, and communications. The Kalman filter is considered to be optimal estimator since it provides linear, unbiased and minimum variance estimates for
unknown state vectors of a linear state space model. It can provide real-time estimates of the state vector of a linear system from noisy observation data.

In this section we employ the Kalman filter to estimate the impulse response of the mobile fading channel. A state space model for this channel was introduced in section (2.4). As shown in Fig. 2-9, the impulse response of the channel, \( x_k \), is the internal state of the state space model. State transition in time is described by (2-22). The Kalman filter receives a noisy measurement, \( z_k \), of the channel states and this measurement is linearly related to the state by an equation of the form (2-24). It is assumed that \( w_k \) and \( n_k \) are individually zero mean stationary white Gaussian processes with known covariance. Hence, the state vector of the system, \( x_k \), is a Gaussian random Markov process [10].

The estimation process in the Kalman filter is composed of two parts: the measurement update and the time update. In the measurement update stage the optimal Kalman filter uses its latest measurement of the channel output and minimizes the estimation mean squared error conditioned on the received signal up to time \( k \). The measurement update estimate of the channel state at time \( k \), is computed given observations \( \{ z_0, z_1, ..., z_k \} \) and will be denoted as \( \hat{x}_{k|k} \) where the "\( ^{\wedge} \)" denotes estimate. The estimation error is defined as the difference between the true value of the channel state \( x_k \) and the estimate \( \hat{x}_{k|k} \). The task of the Kalman filter is to minimize

\[
E[(x_k - \hat{x}_{k|k})^T (x_k - \hat{x}_{k|k}) | \{ z_0, z_1, ..., z_k \}]
\]

The measurement update estimation is called a filtering process since it is performed by using data measured up to and including time \( k \). The next step is a prediction process and is called time update estimation, in which the Kalman filter predicts the channel estimates at time \( k+1 \) based on the measurements up to and including time \( k \). In this part the Kalman filter takes advantage of its information about the state space model and employs the state transition matrix to predict the channel at time \( k+1 \). This estimate can be presented as \( \hat{x}_{k+1|k} \) and we define

\[
\hat{x}_{k+1} = \hat{x}_{k+1|k}
\]

From the above discussion, the basic computation to perform the Kalman filtering algorithm involves an estimation of the states based on the current observation and a
prediction for the next time instant. The prediction is independent of the observation sample and can be computed without waiting for the future observation. Therefore, the computations involved in the estimation and prediction can be done recursively and separated in two different groups called the measurement update equations and the time update equations as follows [10]:

**Measurement Update Equations:**

\[
\hat{x}_{k|k} = \hat{x}_k + K_k(z_k - H_k \hat{x}_k)
\]  
(3-3)

\[
K_k = P_k H_k^T R_k^{-1}
\]  
(3-4)

\[
R_k = H_k P_k H_k^T + N_o
\]  
(3-5)

\[
P_{k|k} = P_k - K_k H_k P_k
\]  
(3-6)

**Time Update Equations:**

\[
\hat{x}_{k+1} = F \hat{x}_{k|k}
\]  
(3-7)

\[
P_{k+1} = F P_{k|k} F^T + G Q G^T
\]  
(3-8)

The covariance matrix for the measurement update estimation error, \( P_{k|k} \) is defined as

\[
P_{k|k} = E[(x_k - \hat{x}_{k|k})(x_k - \hat{x}_{k|k})^T]
\]  
(3-9)

and the covariance matrix for the time update estimation error, \( P_k \) is

\[
P_{k|k} = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]
\]  
(3-10)

The error covariance matrix is positive definite with Hermitian symmetry and provides a statistical description of the error in the estimates. It can be noticed that the error covariance matrix is computed in a recursive form and is independent of the measurements from the channel, \( z_k \). This means that any set of measurements have the same effect in eliminating the uncertainty about \( x_k \). The Kalman gain, \( K_k \), shows the influence of the new measurement, \( z_k \), in modifying the estimate \( \hat{x}_k \). The Kalman gain is also independent of the input measurement. In general case \( P_k \) and \( K_k \) can be precomputed before the filter is actually run. However, in some cases the elements used in the
computation of $P_k$ and $K_k$ depend on $z_k$ and then the Kalman gain and the error covariance matrix are not precomputable.

The initial conditions for the state estimate, $\hat{x}_0$, and for the error covariance matrix, $P_0$, are required to start the recursive loop. In the absence of observed data at time $k=0$, we may choose the initial estimate as [12]

$$\hat{x}_0 = E[x_1]$$  \hspace{1cm} (3-11)

and the initial error covariance matrix as

$$P_0 = E[(x_1 - E(x_1))(x_1 - E(x_1))^T]$$  \hspace{1cm} (3-12)

In the digital implementation of the Kalman filter the size of the available register for each value is finite, and, therefore, the values have to be quantized. Roundoff errors occur as a result of the quantization, and can seriously degrade the performance of the Kalman filter. Calculations involving the error covariance matrix are the major cause for numerical difficulties. The Hermitian symmetry of the covariance matrix can be violated by the quantization errors and this will culminate in numerical instability. To improve the robustness of the filter against numerical errors, measures that symmetrize the result of computation and the employment of square root filtering methods are beneficial.

The process of channel estimation has to be performed in real time, which means the filter has to do its task between measurement inputs. This requirement has to be verified in the design and implementation of the filter; and the amount of computation required for each iteration should be considered and compared with hardware capabilities. The computational complexity of the algorithm is a function of the problem size, which can be represented by the size of the matrices involved. If the state vector size is assumed to be $n$, then the approximate number of multiplications/additions needed for the Kalman filter is $O(n^3)$. Systolic arrays are good candidates for such intensive matrix computation and the realization of the Kalman filter with systolic VLSI structure will be addressed in chapter 5.

Using the measurement update equations, the Kalman filter estimates the next state vector of the linear system or the CIR based on a noisy measurement which is the input signal at the receiver. In this part of the estimation only the received signal and the information about the measurement matrix, $H_k$, and the noise variance, $N_o$, are used. In the
next stage using the time update equations, the Kalman filter yields its estimate of the next
state vector according to its knowledge of the linear system parameters such as \( F, G, \) and
\( Q \). To obtain the matrices \( F \) and \( G \) the receiver has to estimate the maximum Doppler
frequency shift, and calculate the fading filter parameters of (2-4) based on this estimation.
Since the changes in the vehicle speed are not very fast, the estimation of \( f_d \) can be made
off-line and once for every block or every few blocks of data. We have referred to this
issue in section 2.5.

Sometimes it is not practical to obtain the parameters of the channel state space model at
the receiver. Therefore, with the lack of this information we may just perform the
measurement update estimation and eliminate the state prediction stage of the Kalman
filter. This results in the sub-optimal RLS estimator which is described in the next section.

### 3.2 The RLS Algorithm (Kalman Algorithm)

The RLS algorithm \([1][12]\) is a least squares method to estimate the states of a system
based on the noisy observation inputs. When the parameters of the state space model are
unknown to the estimator the RLS algorithm can replace the Kalman filter. As mentioned
in section 3.1 the Kalman filter minimized the estimation mean square error, which is a
statistical average. In the RLS algorithm we deal directly with the received data to
minimize a weighted time average of \( z_k - H_k \hat{x}_k \). We receive the signal \( z_k \) and we wish
to minimize the cost function

\[
\xi(i) = \sum_{k=1}^{i} \lambda^{i-k} |z_k - H_k \hat{x}_k|^2
\]

which is a time average squared error with exponential weighting. The parameter \( \lambda \) is a
forgetting factor and we have \( 0 < \lambda < 1 \). From (2-24) we can see that \( z_k - H_k \hat{x}_k \) is the
noise component at the receiver according to the estimates of the channel impulse
response. Using this cost function the estimator tries to estimate \( \hat{x}_k \) so that \( H_k \hat{x}_k \) is as
close as possible to the received signal plus noise.

The RLS algorithm to minimize the above cost function is given as
Channel Estimation Algorithms

\[
\hat{x}_{k+1} = \hat{x}_k + K_k(z_k - H_k\hat{x}_k)
\]  
(3-14)

\[
K_k = P_kH_k^TR_k^{-1} 
\]  
(3-15)

\[
R_k = H_kP_kH_k^T + \lambda 
\]  
(3-16)

\[
P_{k+1} = \lambda^{-1}(P_k - K_kH_kP_k) 
\]  
(3-17)

By comparing the Kalman filter and the RLS algorithm, we observe that the RLS algorithm is basically the same as the measurement update equations of the Kalman filter. The RLS estimator uses the information of the received signal to update its state estimates and the estimation is performed in one stage similar to the measurement update equations of the Kalman filter. The Kalman filter performs extra computations for predicting the states at the next time step using the time update equations. To do so, the Kalman filter uses its knowledge about the linear system, obtained from the matrices \(F\) and \(G\), and updates the estimated values once more. Hence, when we do not have enough information about the channel system (i.e. matrices \(F\), \(G\) and \(Q\), and \(N_o\)), the RLS algorithm is a good choice and when the channel parameters are known we can implement the Kalman filter which is the optimal estimator.

The RLS algorithm is considered as a special case of the Kalman filter, and hence, it is model dependent. To obtain the RLS equations, the actual model parameters of the system are replaced with constants. This can be optimal only if the actual model parameters match our assumptions. However, since we have made simplifications to the actual model parameters, we would expect a degradation in the performance due to the model mismatch [69].

For the communication system described in section 2.1, and using the channel model described in section 2.4, Fig. 3-1 compares the MSE obtained in the channel state estimation with the RLS algorithm and the Kalman filter. The MSE in the estimation with the Kalman filter and the RLS algorithm, using different values for the forgetting factor \((\lambda)\), are compared here at different values of \(E_b/N_o\) where \(E_b\) is the average bit energy of the transmitted symbol. It is clear from the figure that the performance of the Kalman filter
FIG. 3-1 Comparison of the minimum squared error in channel estimation for the RLS algorithm and the Kalman filter.

is superior to that of the RLS algorithm. For the RLS algorithm the performance depends on the signal to noise ratio or equivalently $E_b/N_o$. For high $E_b/N_o$ values a lower forgetting factor ($\lambda$) results in better estimates, and for low $E_b/N_o$ values higher values for $\lambda$ yield a better performance. From (3-13) it can be seen that with lower values for $\lambda$ the estimator tends to forget its past information faster, while a $\lambda$ value close to one indicates a longer memory and that the estimator keeps more information from the past for the current estimation. For low signal to noise ratio the RLS algorithm needs a longer history of the channel estimates while for high signal to noise ratio it can rely on the current values and a shorter history of past values.
3.3 The LMS Algorithm

The LMS algorithm is a classical tracking method and is important because of its simplicity and ease of computation [13]. However, it is more restricted in its use and performance when compared to the Kalman and RLS algorithms. The state estimation update formula will be as follows

$$\hat{x}_{k+1} = \hat{x}_k + \mu (z_k - H_k \hat{x}_k)H_k^T$$

(3-18)

where $\mu$ is a constant step-size and regulates the speed and stability of the tracking method.

The performance of the RLS algorithm has been compared to that of LMS algorithm in the literature extensively [69][70][71][72]. In [70], the RLS algorithm is studied assuming that the fluctuations of individual channel parameters are Gaussian, mutually uncorrelated, and of the same mean-square value. In this case it is shown that the RLS algorithm has the same tracking capability as the LMS algorithm when the input covariance matrix, $E(H_k^T H_k)$, has equal eigenvalues. In [72], a tracking analysis of the RLS algorithm is provided without the need for the above assumptions on the type of distribution and mutual correlation among the channel parameters. The RLS and LMS algorithms are compared in [72] based on two criteria, which are: minimizing the steady-state excess mean-square of $(z_k - H_k \hat{x}_k)$, and minimizing the steady-state mean-square state estimation error. For both algorithms it is shown that a single value of the adaptation parameter does not minimize both of the above measures. In addition, different conditions are found where the performance of the RLS is better (or worse) than that of the LMS algorithm.

The general conclusion is that typically, the RLS algorithm has a faster rate of convergence than the LMS algorithm, while the LMS algorithm exhibits better tracking behavior than the RLS algorithm [69][72]. The optimum estimator, when tracking fast fading channels, has the additional task of tracking the statistical variations of the channel conditions. However, the LMS algorithm does not consider these statistical variations. The LMS algorithm is not a model dependent algorithm and its tracking behavior, which is a
steady state phenomenon, will not decline when operating in a non-stationary environment.

Fig. 3-2 compares the mean squared error (MSE) obtained in the channel state estimation with the LMS algorithm and the Kalman filter. The MSE in the estimation with the Kalman filter and the RLS algorithm using different values for $\mu$, are compared for different $E_b/N_0$ values. It is clear that the performance of the LMS algorithm depends on $\mu$, and the performance of the Kalman filter is higher than that of LMS. By comparing the results of Fig. 3-1 and Fig. 3-2 it is evident that the RLS and LMS algorithms offer similar performance which highly depends on the value of their parameters (i.e., $\lambda$ and $\mu$).
3.4 Sub-optimal variants of the Kalman filter

Many other estimation algorithms have been introduced in the literature and used in practice [10][12][14]. There are several variants to the Kalman filter algorithm, that essentially differ in the ways they propagate the quantities that are needed in the Kalman recursion, such as the estimation error covariance matrix or the Kalman gain. The Kalman filter is considered as a common framework for deriving all of these algorithms.

The exact parameters of the state space model and the second order statistics of the random model parameters are needed for the implementation of the Kalman filter. However, this information is not always available to the estimator. In this case sub-optimal solutions are used to avoid the requirement of unavailable parameters or to reduce the computational complexity of the estimation process. In these algorithms the unavailable parameters are usually approximated or replaced by known constants [10][11][14][69]. An extensive explanation for different variants of the Kalman algorithm can be found in [14].

In this work, we have mainly considered the Kalman filter and the RLS and LMS algorithms. Studying the advantages of other reduced-complexity and sub-optimal algorithms in different practical situations for application in the MLSD receiver remains open for future research.

3.5 Summary

In this chapter, we introduced three main estimation algorithms that can be employed in the estimation of CIR. The Kalman filter is the optimum estimation method, and the RLS algorithm can be used when the matrices $F$ and $G$ are not available. The LMS algorithm is a simple estimation method that is not model dependent and does not require information about the channel model. The performance of the RLS and LMS algorithms are compared to that of the Kalman filter.

In order to compare the performance of different estimation algorithms, the MSE curve is usually obtained as a function of $E_b/N_0$ values. This is widely used as a performance measure for comparing estimation algorithms. However, in an MLSD receiver that
employs an estimation algorithm for channel estimation, the ultimate performance measure is the BER of the receiver. As stated before, the performance of the estimator has a strong impact on the final BER performance. This motivates us to study the performance of the MLSD receivers employing different estimation algorithms and to evaluate the estimators based on the final BER of the data detector. In the next chapter, we study the problem of joint data detection and channel estimation; and will study the effects of different estimators in that structure.
Chapter 4

Joint Data Detection and Channel Estimation

The mobile communication channel exhibits time-variant and dispersive behavior, which affects nearly every stage of the communication system. Due to the multipath nature of propagation, fading results. If the multipath spread of the channel is large, the higher rate data transmission systems may experience frequency-selective fading. In this case, the signal will be severely distorted by the channel. Typically in systems with finite bandwidth and frequency response distortion, the individual transmitted pulses tend to spread out and overlap, which leads to the condition known as intersymbol interference (ISI). The ISI can be greatly reduced by using signal waveform shaping techniques, and by using an equalizer which helps to minimize ISI due to overlapping of pulses [1][2].

An equalizer, in its broad definition, estimates the state or response of the channel and then attempts to compensate for the effects of the channel to improve transmission system performance [9][73]. There are various kinds of equalizers such as the linear equalizer, and the decision feedback equalizer that have been studied extensively in the literature [1][2]. However, the optimum equalization technique from the viewpoint of minimizing the probability of sequence error in the detection of the transmitted symbol sequence from the received signal samples, is the MLSD.

The Viterbi algorithm is a very efficient algorithm used for the implementation of maximum likelihood sequence detection, which was originally devised for decoding
convolutional codes [17][18]. The Viterbi algorithm can be viewed as a technique which finds the shortest path in the trellis diagram to detect the received sequence by a dynamic programming approach. The Viterbi algorithm assigns a measure to each transition in the trellis that indicates the likelihood of that transition. This measure is the path length between two states and is called the branch metric. The branch metric is usually computed as a distance between an expected value and the received value. For the calculation of the branch metric an estimate of the channel state is required and this can be in the form of an estimate of the CIR.

Different estimation methods such as the Kalman filter, RLS and LMS algorithm, or other variants of the Kalman filter can be used to estimate the channel. However, there is an inherent difficulty associated with applying the estimation methods for tracking the channel. The problem is that the unknown transmitted data is required for the estimator adaptation while in practice this is not available directly to the receiver. The information that is required by the estimator can be provided in one of the following three forms. In the first method this information can be provided as training symbols inserted in a data frame (symbol-aided channel estimation)[29]. For a time invariant channel only a single training sequence is sufficient. For the time variant frequency selective channels the adjacent data symbols overlap the training symbols and obscure the channel information carried by the known symbol, hence more general methods are required for arranging the training symbols [74].

The second method is to use the detected data stream, which is called the decision directed mode [2]. In this case the tentative detected symbols are fed back to the estimator. This method involves some delay and the estimator receives dated information about the channel, causing poor tracking performance. The third approach is to use the joint data and channel estimation techniques [55][38], which can be performed by implementing the channel estimation and the Viterbi algorithm in a Per-Survivor Processing (PSP) fashion [41]. In this chapter we introduce the Per-Branch Processing (PBP) method [58] as a generalized form for PSP. The PBP method can be used when there is more than one channel estimation per symbol interval. Here, we will discuss the general method of
per-branch processing for more than one channel estimation per symbol and we will show that in the special case of one sample per symbol it can be reduced to the PSP method.

In the following sections we will first study maximum likelihood sequence detection and the Viterbi algorithm as an efficient implementation scheme for realizing MLSD. Then we introduce PBP and PSP and we conclude this chapter with simulation results.

4.1 Maximum Likelihood Sequence Detection

In a digital communication system that transmits information over a channel that is subject to ISI, the optimum detector is a maximum-likelihood symbol sequence detector. This detector produces the most probable symbol sequence at its output for the given received sampled sequence. In the following we show that MLSD can be performed using the Viterbi algorithm and a channel estimator such as Kalman filter.

For a sequence of \( m \)-ary transmitted information \( \{ a_i \} \) with \( N \) symbols, there are \( M = m^N \) possible data sequences. If we denote the corresponding received signal vector by \( \{ z_i \} \), then the maximum likelihood sequence detection rule is to chose the sequence \( \{ \hat{a}_i \} \) that maximizes the conditional probability density function (pdf) \( p(z_i|\{a_i\}) \), called the likelihood function. Let's \( Z_k(m) \) represent \( \{ z_1, z_2, ..., z_k, \{a_m\} \} \), the observation up to and including time \( k \) and the hypothesized transmitted sequence \( \{a_m\} \). Then by repeated application of the relation

\[
p(A|B)p(B) = p(A \cap B)
\]

we obtain

\[
p((z_1, z_2, ..., z_N)|\{a_m\}) = \prod_{k=1}^{N} p(z_k|Z_{k-1}(m))
\]

and to simplify the computations we use the natural logarithm

\[
\log p((z_1, z_2, ..., z_N)|\{a_m\}) = \sum_{k=1}^{N} \log p(z_k|Z_{k-1}(m))
\]
If each of the received samples has a Gaussian distribution, then the conditional pdf is also a complex Gaussian function with conditional mean $\overline{z}_k(m)$ and variance $\sigma_k^2(m)$ given by [10]

$$\overline{z}_k(m) = H_k \overline{x}_k$$

$$\sigma_k^2(m) = H_k P_k H_k^T + N_o$$

where $H_k$ is the hypothesized transmitted sequence assuming $\{a_m\}$ was transmitted. The conditional mean and variance are determined by the Kalman filter recursively. In this case the following likelihood index can be considered

$$\Lambda_N^m(\{z_k\}) = \sum_{k=1}^{N} \frac{(z_k - \overline{z}_k(m))^2}{\sigma_k^2(m)} + \log(\sigma_k^2(m))$$

The $m$th hypothesis is the maximum likelihood one if the index $\Lambda_N^m$ is less than that for any of the other hypothesis. The Viterbi algorithm implements MLSD by using the above metric to find the shortest path in the trellis diagram of the transmitted symbols and by finding the most likely sequence corresponding to this path.

In the implementation of MLSD for channels with ISI, a major drawback is the exponential behavior in computational complexity as a function of the span of ISI. The size of the trellis grows exponentially with the channel memory. MLSD using the Viterbi algorithm is more practical where the ISI spans only a few symbols, and the ISI is so severe that the linear equalizer or DFE cannot render an acceptable performance.

### 4.2 The Viterbi Algorithm

The Viterbi algorithm [17][18] is widely applied to many decoding and detection applications. It is an effective method to implement MLSD for convolutional codes or for transmitted sequences over a channel with ISI. The Viterbi algorithm can be used to detect signals generated by a finite-state machine, when the noise component in the samples are independent.

The Viterbi algorithm is defined in terms of a trellis diagram. The trellis diagram is used to represent a state diagram as a function of time. Each discrete time step is a single baud
Joint Data Detection and Channel Estimation

interval $T$, and corresponds to one stage of the trellis. The number of stages in the trellis diagram corresponds to the length of the input data sequence. The total number of states at each time step is $m^l$, where $m$ is the alphabet size and $l$ is the channel memory length. Edges in the trellis diagram show the possible state transitions.

The Viterbi algorithm assigns a value, called the path metric, $\Gamma_i^{[k]}$, to each node in the trellis where $k$ is the time index and $i$ is the state number. For each incoming branch (to the state $j$), the path metric of the node from which the branch originated (state $i$) is added to a branch metric, $b_{ij}$, which gives the likelihood that the corresponding state transition occurred. The branch metric is calculated using the information received from the channel estimator based on the value received over the noisy channel. The sums calculated for each of the incoming branches to a state are compared and the smallest one (corresponding to the most likely transition) is selected. The selected sum is used as the new path metric for the node. The calculation of the path metric is often implemented by an Add-Compare-Select (ACS) unit.

There is one hypothesized transmitted symbol associated with each branch of the trellis at each stage $k$ of the trellis. The branch metric is a numerical value assigned to each branch and indicates the likelihood of the corresponding hypothesized symbol. The Euclidean distance is a suitable branch metric (BM) when noise is Gaussian with constant variance

$$BM = |z_k - H_k \hat{x}_k|^2$$

and when the variance is not constant the log-likelihood branch metrics are used

$$BM = \frac{|(z_k - H_k \hat{x}_k)|^2}{\sigma_k^2} + \log(\sigma_k^2)$$

where $\sigma_k^2 = \sigma_k^2(m)$ is the time varying conditional variance and we have dropped the index $m$ for simplicity. As can be seen from (4-5) and (3-5) this variance can be obtained as a by-product of the Kalman filter ($R_k$) in the process of channel estimation.

It is interesting to note that the Viterbi algorithm can be formulated as a form of matrix multiplication [45]. Consider the trellis diagram of Fig. 4-1. If the path metric at state $i$ and
time \( k \) is shown by \( \Gamma_i^{[k]} \) and the branch metric from node \( i \) to node \( j \) is \( b_{ij} \), then to obtain the path metrics at time \( k+1 \), the Viterbi algorithm computes

\[
\begin{bmatrix}
\Gamma_1 & \Gamma_2 & \Gamma_3 & \Gamma_4
\end{bmatrix}^{[k+1]} = \begin{bmatrix}
\Gamma_1 & \Gamma_2 & \Gamma_3 & \Gamma_4
\end{bmatrix}^{[k]} \circ \begin{bmatrix}
b_{11} & b_{12} & b_{14} \\
b_{21} & b_{22} & b_{24} \\
b_{41} & b_{42} & b_{44}
\end{bmatrix}
\]  
\( (4-9) \)

The operation '\( \circ \)' is not an ordinary multiplication and the elements of the left side vector can be written as

\[
\Gamma_j^{[k+1]} = (\Gamma_1^{[k]} + b_{1j}) \circ (\Gamma_2^{[k]} + b_{2j}) \circ ... \circ (\Gamma_4^{[k]} + b_{4j})
\]  
\( (4-10) \)

where the '+' operator is conventional addition, and the operation '\( \circ \)' denotes taking minimum. Hence (4-10) becomes

\[
\Gamma_j^{[k+1]} = \text{Min}_{1 \leq i \leq 4}(\Gamma_i^{[k]} + b_{ij})
\]  
\( (4-11) \)

In chapter 5 we will see that this formulation will help us to design the VLSI structures for the Viterbi algorithm.

As we mentioned above, to calculate the branch metrics of the Viterbi algorithm for detecting the transmitted data we need an estimate of the channel states. Also, to estimate the channel the transmitted data is required. We addressed three methods to handle this
problem and in the following section we consider the issue of channel estimation and data detection, jointly in a common structure.

4.3 Sequence Detection with PSP and PBP

MLSD of the data sequence by means of the Viterbi algorithm represents an optimum detection technique for the transmitted data over a noisy channel with ISI, provided that the receiver has a perfect knowledge of the parameters characterizing the channel [17]. However, for time varying mobile fading channels the characteristics of the channel has to be estimated dynamically using an estimation algorithm; and in practice the receiver does not have a perfect knowledge of the channel parameters. This results in a suboptimal performance at the receiver. The information obtained from the channel estimator can be in the form of estimates of the channel impulse response that is required for the computation of the branch metrics in (4-7) or (4-8).

Fig. 4-2 illustrates the model of the adaptive receiver in which a channel estimator is employed to provide an estimate of the channel parameters for the detector. The estimator assesses the validity of its current estimate of CIR, \(\hat{x}_k\), by constructing \(\hat{z}_k = H_k \hat{x}_k\), and comparing it against \(z_k\). This is the convolution sum of the estimated CIR and the
transmitted signal. The error signal defined as, \( e_k = z_k - H_k \hat{x}_k \), is used for adaptation of the channel estimator and to update the current estimate of CIR, used by the detector in branch metric generation.

The estimation techniques described in chapter 3 can be used here for tracking the channel. All of these estimation algorithms require the vector \( H_k \), which is defined in (2-23) and depends on the transmitted data sequence. In the state space model of Fig. 2-9, \( H_k \) is the measurement matrix and is considered as part of the channel model parameters. However, the transmitted data is not available at the receiver in practical situations. This problem is sometimes called "state estimation with model uncertainty", where the channel estimator has to estimate the states of the linear system of Fig. 2-9 and the vector \( H_k \), is unknown.

One solution in this situation is to use the 'tentative decisions' instead of the actual transmitted data to construct \( H_k \). This can be viewed as incorporating the decision feedback mechanism within the Viterbi decoder. It is clear that the quality of the channel estimation will depend on the quality of the tentative decisions fed to the estimator. In low signal to noise ratios (SNR) there will be a degenerative loop of poor estimates and poor decisions in the receiver. However, if the detector's decisions are highly reliable, then the decision-directed adaptation of the channel estimates can be a successful method. There is normally a decision delay involved in this process which leaves the estimator with outdated information about the channel for adaptation of its estimates. This can be tolerable in slowly time varying channels and for fast fading channels it can result in a poor performance.

The other solution proposed in the literature [41][42][43] is to implement the channel estimation and the Viterbi algorithm in a PSP fashion. In this case, to overcome the problem of uncertainty in \( H_k \), on each branch of the trellis a hypothesized data vector, \( H_k \), will be chosen according to the state transition corresponding to that branch. Then a separate estimator is required for any of the hypothesized \( H_k \) vectors on each branch. In this chapter we introduce a more general approach called PBP [58] which can be reduced to PSP in the special case of one channel estimation per symbol interval. In the following
we first introduce data detection using PBP technique and then demonstrate that PSP is a simplified, special case of this method.

**Per-Branch Processing (PBP)**

In this technique we use the hypothesized data associated with each branch of the trellis to create the hypothesized $H_k$ for channel estimation along that branch. On each branch of the trellis diagram of the Viterbi algorithm there will be one or more channel estimators, depending on the number of estimations performed per symbol interval. Intuitively, it is clear that the problem of decision delay will be eliminated at the expense of employing more estimators. In the decision directed mode the tentative decisions were used to obtain the metrics and to update the channel estimates.

In the per-branch processing method all of the possible data sequences are considered in the trellis diagram of the Viterbi algorithm. In this way, the estimators on each branch use their own hypothesized data vector for $H_k$ and based on that, they update the estimates of the CIR. Here, on each branch instead of a delayed data sequence, a hypothesized data vector corresponding to the state transition on that branch is used. This guarantees that we are using the data sequence of the shortest path for the channel estimation along the same path, which is obviously the best available information at the receiver. This method also eliminates the problem of decision delay, since the detected data associated with each survivor path is used for channel estimation on the same path immediately.

It is important to note that the vector $H_k$ is required for two reasons at the receiver. First, it is required for the calculation of the branch metrics. If $s$ samples are received during one symbol interval, a distance measure can be calculated for each received sample and summed up to obtain the cumulative branch metric as

$$BM = \sum_{l=1}^{s} \left| z_{k+l} - H_{k+l} \hat{x}_{k+l} \right|^2$$  \hspace{1cm} (4-12)$$

or

$$BM = \sum_{l=1}^{s} \left( \left| \frac{z_{k+l} - H_{k+l} \hat{x}_{k+l}}{\sigma^2_{k+l}} \right|^2 + \log(\sigma^2_{k+l}) \right)$$  \hspace{1cm} (4-13)$$
Second, the vector $H_k$ is required in the channel estimation process as all of the estimation algorithms of chapter 3 require this vector. The output of the channel estimator at time $k+1$ can be expressed as a function of its inputs

$$\hat{x}_{k+1} = G_\epsilon[\hat{x}_k, z_k, H_k] \quad (4-14)$$

As is clear from (4-13) and (4-14), there is a data dependency involved in the calculation of the branch metric and the channel estimates. The channel estimation for the CIR at each epoch has to be ready before the calculation of the distance metric can start. Fig. 4-3 shows the computational procedure of the PBP method for the case of three samples per symbol interval. The computation is shown for a branch between states $S_i$ and $S_j$. There are three received samples ($z_k, z_{k+1}, z_{k+2}$) and three hypothesized data vectors ($H_k, H_{k+1}, H_{k+2}$) on this branch. After receiving the first received signal sample, a Branch Metric Generator (BMG) unit obtains a measure of the likelihood of the hypothesized data vector, given the received sample, as in (4-12) or (4-13). At the same time an estimator unit updates the channel estimates based on the received signal and the hypothesized $H_k$. The new estimates will be passed to an accumulative BMG and another estimator to be processed with the second received sample and hypothesized $H_k$. 

FIG. 4-3 Data flow for the computations required on each branch of the trellis in PBP for three channel estimations per symbol interval.
The hypothesized $H_k$ is determined by the branch and the state transition associated with that. From (2-1) we can see that with $s$ samples per symbol interval, there are $s-1$ zeros inserted between data symbols to obtain $H_k$ in (2-23). Then on the same branch $H_k$ has to be shifted to obtain the consecutive $(H_{k+1}, H_{k+2}, \ldots)$ values.

After processing all three samples in three stages, as shown in Fig. 4-3, the branch metric is ready and the procedure of Add-Compare-Select can be started at node $S_j$ to find the survivor branch. Similar to the regular Viterbi algorithm the information of the survivor path to each node has to be stored. Moreover, the estimated CIR on the survivor path will be used as the initial value for the estimators on the outgoing branches from that node. This is illustrated in Fig. 4-4. Thus, we can see that the calculations on the incoming branches to one state are not only different in terms of considering different hypothesized $H_k$, but also different channel estimates are used initially on each branch for channel adaptation. The same routine has to be performed on all of the branches of the trellis diagram hence, it is called PBP.
Joint Data Detection and Channel Estimation

Per-Survivor Processing (PSP)

In this section we will show that for the special case of one channel estimation per symbol interval, PBP reduces to PSP. This will be performed by changing the order of branch metric calculation and channel estimation, which was not possible due to the data dependency in PBP. This can be explained based on the data flow of the algorithm shown in Fig. 4-5.

In the simple case of one sample per symbol interval, Fig. 4-3 reduces to Fig. 4-5(a) in which only one estimator and one BMG unit are used. As in PBP the same structure is

---

**FIG. 4-5** (a) Data flow for PBP with one sample per symbol interval, which can be reduced to PSP. (b) Data flow for PSP equivalent to (a).
used on all of the trellis branches. However, in this case it is possible to reduce the complexity of the algorithm and avoid unnecessary estimations. Since only the estimates along the survivor branch will be retained and other estimates will be discarded one could reduce the required computations by finding the survivor branch first, and then estimate the channel just for that branch. As shown in Fig. 4-5(b) the channel estimations on all of the branches can be postponed until the branch metrics are computed and the Add-Compare-Select (ACS) procedures are performed to find the survivor branch to each node. Then only the estimators on the surviving paths will be used and the result employed to update the channel estimates for the next symbol interval. Since only the estimators on the survivor paths are used, this is equivalent to having one estimator for each state in the trellis instead of each branch, as in Fig. 4-5(b). This method is called PSP, where the number of estimations is reduced to the number of surviving paths. Therefore, PSP can be considered as a special case of PBP.

The procedures required to perform the PBP algorithm are summarized in Fig. 4-6(a). The boxes represent the channel estimation on each branch and the bold lines represent the channel estimates on the survivor paths that are passed to the outgoing branches. The PBP algorithm can be performed in four steps as shown in Fig. 4-6(a). First the channel estimations have to be performed and then the branch metrics can be computed. After finding the survivor path only the estimates along this path will be retained and passed to the next stage as the initial values.

Fig. 4-6(b) shows the procedures required to perform the PSP algorithm. Estimators are moved to the nodes from branches and hence a smaller number of estimators is required. By comparing the four steps that are required for PSP to that of PBP, we can realize that channel estimation is postponed to the third step and is restricted to the surviving paths. It should be noted that in the previous case (Fig. 4-3) with more than one sample per symbol, because of the data dependency it was not possible to postpone the channel estimations until the branch metrics are ready and the survivor path is known. By studying the data dependency on this diagram we can realize that only the last estimation could be deferred in this case.
Joint Data Detection and Channel Estimation

FIG. 4-6 The PBP and PSP algorithms

On fast fading channels, the error floor in the BER curve can be appreciably lowered if more than one sample of the received signal is processed at the receiver [35][56]. In a fast fading channel one sample per symbol interval is no longer a sufficient statistic for the decision process. Therefore, the number of samples per symbol can be increased to attain the sufficient statistics and to allow for more channel estimations per symbol interval; which results in a better tracking performance and a more accurate channel estimation at the receiver. However, more samples per symbol interval demand more signal processing and faster hardware for implementation.

A bank of estimators is required to implement PBP or PSP, one or more for each branch in PBP and one for each survivor sequence in PSP. By choosing the Kalman filter for channel tracking, the complexity of the receiver can be prohibitive, particularly for
channels with a long impulse response duration. The Kalman filter is computationally demanding and very sensitive to round-off errors. The efforts to overcome the design complexity of the Kalman based PBP or PSP receiver can be focused in two major directions. The first approach is to employ simpler channel estimators. This can be in the form of seeking suboptimal alternatives to the Kalman filter [69][14]. A second direction is to take advantage of recent advances in VLSI technology in parallel information processing and proposing parallel algorithms and structures for the implementation of Kalman filter [59][60][49]. The latter topic will be discussed in chapter 5.

It should be noted that the proposed receiver scheme can be used for the multi-ray channel as long as we assume the same length for the channel impulse response and fixed channel memory. Although we have assumed a two ray fading channel model in chapter 2, the structure of the receiver and its complexity will not change if there is a larger number of rays in the channel. The only difference will be in the computation of the covariance matrix elements of (2-26), and other parameters will be unaffected.

The above joint channel estimation and data detection method can be performed with any estimation algorithm. The difference in the results will be due to the tracking performance and precision of the estimators. By using the channel model of chapter 2, it is possible to employ the Kalman estimator which is the optimal estimation method and, as is shown in the next section, simulation results confirm that its performance is superior to that of other estimators.

### 4.4 Simulation Results

In the computer simulations, the modulation scheme employed is differentially coherent Quadrature Phase-Shift Keying (DQPSK), with a symbol rate of 25 ksymbols/s, which is a little higher than 24.3 ksymbols/s in the IS-136 standard. A detailed block diagram of the simulated system is shown in Fig. 4-7. As in the IS-136 standard, the differentially encoded data sequence is arranged into 162 symbol frames. The first 14 symbols of each frame is a training preamble sequence to help the adaptation of the channel estimator. For the shaping filter \( f(t) \) at the transmitter, we implement an FIR filter which approximates a
Joint Data Detection and Channel Estimation

FIG. 4-7 Block diagram of the simulation system.

raised cosine frequency response with the excess bandwidth of 25%. The fading channel is simulated as a symbol-spaced two-path model with time varying complex coefficients. The two fading paths are independent with equal strength, and are implemented as shown in the model of Fig. 2-5. The receiver takes 3 samples per symbol interval and the complex samples are processed by the digital processor to detect the transmitted data. Differential encoding and differential detection enables the receiver to avoid errors due to phase ambiguity.

There is no intersymbol interference for the transmitted symbols at the transmitter and the ISI at the receiver is due to the multipath nature of the channel. The total length of the channel impulse response is 2 symbol intervals and there are four possible states in the trellis diagram of the Viterbi algorithm at the receiver. For each state in the trellis there are four possible transitions to the four states in the next stage.

1. The discrete impulse response of the shaping filter has a finite length, equal to the symbol interval and hence does not produce ISI; the samples for the given excess bandwidth of 25% were obtained using the SPW™ software package.
Fig. 4-8 shows the simulation results for a vehicle speed of 100 km/h. The BER performance of different estimators are compared here at different values of $E_b/N_0$. In each simulation different channel estimators are used to estimate the channel impulse response on every received sample. From the results, it can be observed that the performance of the RLS algorithm is superior to that of LMS algorithm by about 3 dB at a BER=$10^{-3}$. This is due to the faster tracking behavior of the RLS algorithm in the fast fading conditions. Choosing the Kalman estimator provides 7 dB improvement in performance over the LMS algorithm at the same BER, and it shows the superiority of this estimation method over the LMS and RLS algorithms.

The last curve is for the case of data detection over a known channel. In this case we have assumed that the exact channel impulse response is always known to the receiver and there is no need to estimate it. Of course this situation is not possible in a practical implementation, and this can be viewed as an error-free estimation method giving a lower
FIG. 4-9 Simulation results with different estimation methods, when the speed of the vehicle is 50 km/h.

BER bound for comparison. The performance of the Kalman filter is about 2 dB poorer than the best possible results obtained with a known channel at a BER=$10^{-3}$.

Fig. 4-9 shows the same situation for a slower fading rate. In this case the vehicle speed is 50 km/h and this results in slower variations in the envelope of the received signal. This means the channel impulse response is changing slowly over time and hence it would be easier for the channel estimator to track its variations. It is clear from Fig. 4-9 that the performances of the estimators are better than those of Fig. 4-8. The $f_dT$ factor indicates how fast the fading process is changing by time. The $f_dT$ values given in Fig. 4-8 and Fig. 4-9 are calculated for the IS-136 900 MHz standard at the given speed. For the GSM-900 MHz we obtain $f_dT=0.00015$ at 50 km/h and $f_dT=0.0003$ at 100 km/h.

When the channel impulse response does not change very rapidly, the channel estimation can be performed at a lower rate. This leads to less computation at the receiver. Fig. 4-10 compares the results for the Kalman estimator for the situation of one channel.
FIG. 4-10 The result of changing the estimation rate.
estimation every symbol interval using the PSP method and the situation of three channel estimations per symbol interval using PBP. As we can see there is a difference of about 2 dB at a BER=10^{-4} between the two methods, which in some cases, might be tolerated to reduce the complexity of the receiver.

As mentioned before, to employ the Kalman filter estimator the receiver computes the matrices $F$ and $G$ based on its assumption about the maximum Doppler frequency shift. In Fig. 4-11 we can observe the effect of error in the estimation of the maximum Doppler frequency shift. The curve labeled 100S-100E is for the normal case where the actual speed of the vehicle is 100 km/h and it is correctly assumed as 100 km/h in the receiver. The curves 150S-100E and 50S-100E show the situation where the actual speed is 150 and 50 km/h, respectively, but in both cases the speed is assumed to be 100 km/h at the

**FIG. 4-11** The effect of error in estimation of Doppler frequency shift, using the Kalman Filter.
receiver. And finally for the 50S-50E curve the receiver assumes the correct speed for a
vehicle with a speed of 50 km/h.

In Fig. 4-11 the dashed lines show the situation when the estimate of the vehicle speed is
in error by 50 km/h and in both cases the performance is about 2 to 4 dB poorer than the
case where we assume the correct speed for the vehicle. It can be easily observed that we
can always attain better results than what we are expecting by overestimating the speed.
By accepting a reasonable margin in BER performance, one may assign a limited number
of speeds and switch from one preselected speed threshold to another when the vehicle
speed changes.

To reduce the complexity of the receiver we may consider a lower order fading filter at
the receiver. In this case the dimensions of the matrix $F$ will decrease and it mitigates the
computational burden of time update equations of the Kalman filter. Fig. 4-12 shows the

**FIG. 4-12** The effect of considering lower order fading filters at the receiver.
results for this case, where the channel is simulated with a third order fading filter as before, and the receiving filter is assumed to have a lower order. In Fig. 4-12 the results for the case of second order and first order fading filters are compared to that of a third order filter. The second order filter has been designed to have 6 dB overshoot at the maximum Doppler frequency shift. In this case the parameter C in (2-4) will be eliminated and this results in smaller dimensions for the matrices and vectors in the receiving algorithm.

For the first order filter, it was realized that the performance of the receiver is very sensitive to the choice of cut-off frequency. By keeping the cut off frequency close to that of the second order filter attempts were made to find the optimum cut off frequency that leads to the best BER performance. It was observed that without a proper cut off frequency for the first order fading filter the time update equations of the Kalman filter seem to have negligible effect and we obtain a BER performance very close to the case of using the RLS algorithm, where the time update equations are absent.

In the RLS algorithm the overall BER performance depends on the chosen value for the forgetting factor, $\lambda$. Fig. 4-13 shows the BER curves for different values of $\lambda$. It can be seen that for small $E_b/N_0$, larger values of $\lambda$ yield better results, while for high $E_b/N_0$ values, $\lambda$ should be smaller.

The RLS algorithm minimizes its cost function over an interval which is determined by $\lambda$. The above consequence means that in poor $E_b/N_0$ conditions the estimator should consider a larger interval to minimize the cost function, while for high $E_b/N_0$ values minimization over smaller intervals yields better performance. This is consistent with the results obtained from Fig. 3-1, for the MSE in channel estimation.

It was observed that the results with the RLS algorithm for the case of one estimation per symbol are very close to what is given in [38] for $k=1$, and it is possible to improve the performance by applying the Kalman Filter. From the above results, the superiority of the Kalman filter is clearly evident. The Kalman filter shows the best tracking performance for rapidly changing time-variant channels, followed by the RLS algorithm as the next best choice as a good estimator with fast tracking. However, in spite of their superior tracking performance, the Kalman filter and the RLS algorithm have two disadvantages. One is the
sensitivity of these recursive algorithms to round-off noise. This may cause numerical
instabilities such that the algorithm may diverge due to round-off noise if the word-length
is not long enough in the DSP implementation. The second problem is the complexity of
these algorithms that originates from the iterative processing of the matrix operations.
Square-root filtering, implemented in the form of VLSI systolic architectures can be used
to combat the problem of numerical instability and complexity of the Kalman filter and the
RLS algorithm [50][51][52]. This issue will be addressed in the following chapter.
4.5 Summary

In this chapter Per-branch Processing (PBP) was introduced as a generalization for PSP to be used for the case of more than one channel estimation per symbol interval. The differences between PBP and PSP were described.

The BER results with the Kalman estimator were shown to be superior to those of other estimation methods. For accurate results, it was important to have a correct estimate of the vehicle speed. By considering the effects of error in speed estimation, we concluded that it is always better to overestimate the vehicle speed. Lower order fading filters were used to simplify the receiver structure, however, the first order filter was realized to be very sensitive to the choice of cut off frequency.

In the RLS algorithm choosing a proper value for the forgetting factor, $\lambda$, was considered. For low signal-to-noise ratios it was better to choose values close to one for $\lambda$, and for high signal-to-noise ratios the value of $\lambda$ should be kept small.

To make a decision about choosing an estimator for the receiver, it is vital to have a good understanding of the performance and the implementation complexity of the algorithm. So far we have studied the performance of the estimation algorithms and in the following chapter we study the implementation issues related to an MLSD receiver. This will include VLSI structures for the Kalman filter and the RLS algorithm, and also structures performing the Viterbi algorithm combined with channel estimation.
Chapter 5
Implementing the Estimation Algorithm

The Kalman filter has been successfully applied to many signal processing, control, and communication algorithms. As discussed in the previous chapters, Kalman filtering is an optimum channel estimation technique that can lead to significant improvement in the receiver bit error rate performance. The filter estimates the states of a system from noisy measurements using a mean square criterion. The Kalman filter is a computationally demanding algorithm, where the matrix nature of the equations requires $O(n^3)$ computations in each estimation update [10][11][75]. Parallel processing and systolic structures are considered a practical means to realize a digital implementation of the Kalman filter for real-time applications.

The Kalman filter can be realized either with special-purpose digital hardware or by programming a general-purpose DSP core. In either case, coefficients and intermediate results such as state values are stored in registers with a finite number of bits, therefore, the values must be quantized. The results of all computations have to be quantized to fit the results in finite-sized registers. The quantization process causes round off error in numerical calculations.

The conventional Kalman filter algorithm is very sensitive to round-off errors. In order to obtain a numerically accurate and stable algorithm, square-root solutions have been proposed for implementation of the Kalman filter. Square-root implementation methods
implement the robustness of the algorithm against numerical errors. Parallel and systolic structures are often used for the implementation of square-root Kalman filtering [53]. Systolic arrays are regular lattices of simple cells with nearest-neighbor connections. They were originally proposed by Kung and Leiserson [76] to exploit the high speed processing potential offered by VLSI technology.

With recent advances in VLSI technology parallel information processing has become more and more feasible, allowing for the implementation of dedicated systolic structures for square-root Kalman filtering. Real-time computation of the Kalman estimator in a mobile communication receiver calls for parallel and pipelined structures to take advantage of the inherent parallelism in the algorithm. An overview of some algorithms for the implementation of Kalman filter is given in [53].

The RLS algorithm (also referred to as the Kalman algorithm in literature [1]) also shares many of the above issues with the Kalman filter. The amount of computation required is $O(n^2)$ in the RLS algorithm and it is sensitive to computational errors caused by round off noise. Square-root methods can help to provide a robust algorithm for RLS as well as the Kalman filter. Also systolic and parallel structures can overcome the burden of high computational demand for real-time applications.

The implementation of the Kalman filter can be divided into two parts: implementation of the measurement update equations and implementation of time update equations. The first part is basically the same as the RLS algorithm. Jover and Kailath have proposed an algorithm and a parallel structure for the measurement update equations [54]. This algorithm has been adopted in [60] with some modifications for the application to wireless mobile communications, and it is shown that the VLSI structure can be drastically simplified if it is used for the realization of a RLS estimator. To implement the time update equations of the Kalman filter, a weighted Gram-Schmidt (WGS) orthogonalization method is widely used. In a study of target-tracking methods, Raghavan et. al [77], proposed the application of a LD Correction (LDC) method for time update measurements of the Kalman filter. This algorithm requires less computation compared to the weighted Gram-Schmidt orthogonalization method.
In the literature, the implementation of estimation algorithms are usually considered only generally and not for a specific application. In this chapter we study the implementation of fading channel estimators along with the Viterbi detector. The LDC algorithm of [77] is adopted here, for the first time, for the application to mobile communication receivers, and systolic structures are developed and studied for VLSI implementation of the LDC algorithm. Utilization of the estimators is considered in the structure of the Viterbi based receiver, implementing joint data detection and channel estimation. Parallel structures are proposed for the implementation of the Viterbi detector in PSP [41] and PBP [58] fashion that offers an improved and robust detection technique. Finally, the robustness of the receiver structure is studied with regard to the wordlength required in a digital implementation. Studies show that the weighted Gram-Schmidt orthogonalization method and the correction algorithm need the same number of bits in implementation while the latter requires less computation.

In the following sections we start with an introduction to the square-root filtering method and then we continue with the implementation issues for this method.

5.1 Square-root Filtering

Studies show that some implementations are more robust against roundoff errors and ill-conditioned problems. The so-called "square-root" filter implementations have generally better error propagation bounds than the conventional Kalman filter equations [11]. In the square-root forms of the Kalman filter matrices are factorized, and triangular square-roots are propagated in the recursive algorithm to preserve the symmetry of the covariance (information) matrices in the presence of roundoff errors.

There are different factorization methods and in these methods different techniques are used for changing the dependent variable of the recursive estimation algorithm to factors of the covariance matrix. A Cholesky factor of a symmetric nonnegative definite matrix \( M \) is a matrix \( C \) such that \( CC^T = M \). Cholesky decomposition algorithms solve for \( C \) that is either upper triangular or lower triangular. The modified Cholesky decomposition algorithms solve for a diagonal factor and either a lower triangular factor \( L \) or an upper
triangular factor \( U \) such that \( M = UD_U U^T = LD_L L^T \), where \( D_L \) and \( D_U \) are diagonal factors with nonnegative diagonal elements.

The square-root methods propagate the \( L-D \) or \( U-D \) factors of the covariance matrix rather than the covariance matrix. The propagation of square-root matrices implicitly preserves the Hermitian symmetry and non-negative definiteness of the computed covariance matrix. The condition number \( K(P) = [\text{eigenvalue}_{\text{max}}(P)/\text{eigenvalue}_{\text{min}}(P)] \) of the covariance matrix \( P \) can be written as \( K(P) = K(LDL^T) = K(BB^T) = [K(B)]^2 \), where \( B = LD^{1/2} \). Therefore, the condition number of \( B \) used in the square-root method is much smaller than the condition number of the \( P \) and this leads to improved numerical robustness of the algorithm. Moreover, in the square-root method the dynamic range of the numbers entering into computations will be reduced. Loosely speaking, we can say that the computations which involve numbers ranging between \( 2^{-N} \) to \( 2^{+N} \) will be reduced to ranges between \( 2^{-N/2} \) to \( 2^{+N/2} \). All of these will directly affect the accuracy of computer computations.

There are also other factorization methods employed for increasing the numerical stability, like Triangularization (QR decomposition) and Gram-Schmidt orthonormalization used for factoring matrices as products of triangular and orthonormal matrices. The block matrix factorization of a matrix expression is a general approach that uses two different factorizations to represent the two sides of an equation such as

\[
CC^T = AA^T + BB^T = \begin{bmatrix} A \\ B \end{bmatrix}^T \begin{bmatrix} A \\ B \end{bmatrix} \tag{5-1}
\]

The alternative Cholesky factor \( C \) and \( [A \ B] \) can be related by orthogonal transformation [11].

In the following sections we break the problem of implementation for the Kalman filter in two parts. The first part is the implementation of the measurement update algorithms which can also be applied to the implementation of RLS algorithm. This issue is addressed in the next section, and the subsequent section is on the implementation of time update equations.
5.2 Implementation Algorithms for Measurement Update Equations

To compute the measurement update equations of the Kalman filter for real numbers, a square-root method is proposed in [54] by Jover and Kailath. Although this algorithm is not complete for implementing the Kalman filter, in the following we will extend it to complex numbers and then we will add some procedures for computing the time update equations.

The measurement update equation for the covariance matrix can be written from (3-4) and (3-6) as

\[ P_{k|k} = P_k - P_k H_k^T R_k^{-1} H_k P_k \]  \hspace{1cm} (5-2)

Our implementation algorithm is based on working with \( LDU \) (unit lower triangular, diagonal, unit upper triangular) factorizations of \( P_k \) and \( P_{k|k} \), and since \( P_k \) and \( P_{k|k} \) are Hermitian symmetric, \( U=L^T \).

It can be shown that by choosing a suitable orthogonal transformation matrix \( \Theta \), where \( \Theta \Theta^T = I \), we can have

\[
\begin{bmatrix}
N_0^{1/2} & H_k P_k^{1/2} \\
0 & P_k^{1/2}
\end{bmatrix} \Theta =
\begin{bmatrix}
R_k^{1/2} & 0 \\
0 & P_k H_k^T R_k^{-1/2} P_k^{1/2}
\end{bmatrix}
\]  \hspace{1cm} (5-3)

and this can be immediately verified by “squaring” both sides of (5-3). Generally, computing the triangular factor \( P_{k|k}^{1/2} \) requires taking arithmetic square-roots, which are computationally more expensive than multiplication or division. This can be avoided by using \( LDU \) factorizations

\[ P_k = LDL^T \quad \text{and} \quad P_{k|k} = L_p D_p L_p^T \]  \hspace{1cm} (5-4)

Using (5-4) in (5-3) and dropping all time-index subscripts yields

\[
\begin{bmatrix}
1 & HL \\
0 & L
\end{bmatrix}
\begin{bmatrix}
N_0 & 0 \\
0 & D
\end{bmatrix}^{1/2} \Theta =
\begin{bmatrix}
1 & 0 \\
K & L_p
\end{bmatrix}
\begin{bmatrix}
R & 0 \\
0 & D_p
\end{bmatrix}^{1/2}
\]  \hspace{1cm} (5-5)
Implementing the Estimation Algorithm

Therefore to compute the measurement update equation for the covariance matrix we start with the left hand side of (5-5) and by applying an orthogonal transformation, the left side of (5-5) can be converted to the LD form on the right side. This is possible by application of the Fast Givens Transformation as we have modified from [54]. We can express (5-5) in terms of the components of the matrices as shown below where the size of $L$ and $D$ is considered to be $n \times n$:

$$
\begin{bmatrix}
1 & p_1 & p_2 & \ldots & p_n \\
0 & 1 & 0 & \ldots & 0 \\
0 & L_2 & 1 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & L_n & L_{n-1} & \ldots & 1
\end{bmatrix}
\begin{bmatrix}
N_o & 0 & 0 & \ldots & 0 \\
0 & D_1 & 0 & \ldots & 0 \\
0 & 0 & D_2 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & D_n
\end{bmatrix}^{1/2}
= 
\begin{bmatrix}
1 & 0 & 0 & \ldots & 0 \\
K_1 & 1 & 0 & \ldots & 0 \\
K_2 & L_{2,1} & 1 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
K_n & L_{n,1} & L_{n-1,2} & \ldots & 1
\end{bmatrix}
\begin{bmatrix}
R & 0 & 0 & \ldots & 0 \\
0 & \overline{D}_1 & 0 & \ldots & 0 \\
0 & 0 & \overline{D}_2 & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \ldots & \overline{D}_n
\end{bmatrix}^{1/2}
$$

(5-6)

The goal is to obtain a lower triangular and a diagonal matrix on the right hand side of (5-6) by applying an orthogonal transformation $\Theta$ and setting the $p_j$ components to zero. We can perform this by considering a series of orthogonal transformations as

$$
\Theta = \Theta_{1,n}\Theta_{1,n-1}\ldots\Theta_{1,2}\Theta_{1,1}
$$

(5-7)

where the matrix $\Theta_{i,j}$ is a $(n+1) \times (n+1)$ identity matrix with only four elements changed on $(1,1), (1,j+1), (j+1,1)$, and $(j+1,j+1)$ positions. Therefore, the $\Theta_{i,j}$ transformation only affects the first and $(j+1)$th columns of the right hand side matrices in (5-6). If we just consider these two columns the pair $[1 \ p_j]$ will be on top and, $\Theta_{i,j}$ can be found so that the vector $[1 \ p_j]$ be transformed to $[1 \ 0]$, in order to triangularize the matrix.

If we only consider the top pair of the two columns, for instance $[1 \ p_2]$ for first and third columns, with a proper orthogonal transformation matrix we will have

$$
\begin{bmatrix}
1 & p_2
\end{bmatrix}
\begin{bmatrix}
d_{p_1} & 0 \\
0 & d_{p_2}
\end{bmatrix}^{1/2}
\Theta_{1,2}
= 
\begin{bmatrix}
1 & 0
\end{bmatrix}
\begin{bmatrix}
d_{q_1} & 0 \\
0 & d_{q_2}
\end{bmatrix}^{1/2}
$$

(5-8)

where $d_{p_1}$ and $d_{p_2}$ are the components in (5-6) that affect $[1 \ p_2]$, and $d_{q_1}$ and $d_{q_2}$ have to be found based on the applied orthogonal transformation.
It is shown in [60][54] that by using the parameters in the left hand side of (5-8), we can choose

\[ d_{q1} = d_{p1} + |p_2|^2 d_{p2} \]  

(5-9)

and

\[ d_{q2} = \frac{d_{p1} d_{p2}}{d_{q1}} \]  

(5-10)

to obtain the proper orthogonal transformation. The \( \Theta_{I,2} \) transformation is found to rotate the vector \([1 \ p_2]\) to lie along the vector \([1 \ 0]\) keeping the equality of weighted norms [54]. It is also necessary to apply this transformation to other pairs of the first and third columns and find the new transforms of these vectors. By applying the transformation to an arbitrary vector \([ \bar{p}_1, \bar{p}_2 ]\) to lie along \([ \bar{q}_1, \bar{q}_2 ]\), we obtain

\[
\begin{bmatrix}
\bar{p}_1 & \bar{p}_2 \\
\end{bmatrix}
\begin{bmatrix}
d_{p1} & 0 \\
0 & d_{p2} \\
\end{bmatrix}^{1/2}
\Theta_{I,2} =
\begin{bmatrix}
\bar{q}_1 & \bar{q}_2 \\
0 & d_{q2} \\
\end{bmatrix}
\begin{bmatrix}
d_{q1} & 0 \\
0 & d_{q2} \\
\end{bmatrix}^{1/2}
\]

(5-11)

where

\[ \bar{q}_2 = -p_2 \bar{p}_1 + \bar{p}_2 \]  

(5-12)

and

\[ \bar{q}_1 = \bar{p}_1 + \left(p_2^{-d_{p2}} \right) \bar{q}_2 \]  

(5-13)

The complex conjugate of \( p_2 \) is denoted by \( p_2^* \).

The algorithm to implement the above triangularization is to consider the first and the other columns of the matrix in the right side of (5-6) one by one and apply all \( \Theta_{I,j} \) transforms to obtain a lower triangular and a diagonal matrix. The algorithm presented in Appendix B is based on the above method to compute the measurement update equations of the Kalman filter.

The covariance update algorithm that we explained in this section computes (5-2) or equivalently (3-4)-(3-6), however, we need to compute (3-3) to update the state estimates as well. It can be shown [54] that with an appropriate arrangement for the input data structure, the covariance update algorithm can also be used for updating the state.
Implementing the Estimation Algorithm

The Jover-Kailath Algorithm

FIG. 5-1 The data structure for the inputs and outputs of the Jover-Kailath algorithm [54].

estimates. Fig. 5-1 shows the arrangement for the inputs and outputs of the algorithm given in Appendix B. The components of the state estimate vector are fed to the algorithm along with the components of $L$, and the algorithm yields the update as in Fig. 5-1. A parallel architecture is proposed in [54] to implement this algorithm for real numbers. However, this architecture can also be used for complex numbers with some modifications [60].

The above method for the implementation of the measurement update equations of the Kalman filter, can also be used for the RLS algorithm. In the following we will propose a pipelined structure for the implementation of the RLS estimator based on the above algorithm.

Parallel Structures for the RLS algorithm

The above square-root filtering method can be used both for the Kalman filter and the RLS algorithm; however, the implementation can be simplified drastically for the RLS algorithm leading to an affordable hardware realization.
Note that the matrix $F$ is not used in the equations of the RLS algorithm (i.e. (3-14)-(3-17)). It is also possible to assume that the matrix $F$ is equal to the identity matrix for the RLS algorithm. In this case, from (2-20)-(2-22) we can deduce that the CIR of the system at time $k+1$ is only obtained from the CIR of the system at time $k$ without any prediction. Assuming the simple form of the identity matrix for $F$ converts the optimal Kalman filter to a suboptimal algorithm. This means that we are not considering the AR representation of (2-19) for RLS and hence, the state vector of the system, as defined in (2-20), only consists of the CIR at the present time, $h_k$. This will reduce the dimensionality of the state vector and covariance matrices in the RLS algorithm by a factor of 3, which is the order of the AR representation in this case, compared to the Kalman filter. Obviously, a smaller matrix size produces less complexity and more robustness against round-off errors.

To develop the parallel computing structures for the RLS algorithm, we consider a channel impulse response with the total length of 2 symbol intervals as an example. In this case $H_k$ is a $1 \times 6$ vector with two nonzero components (see (2-23)). In the RLS algorithm equations (i.e., (3-14)-(3-17)) $\hat{x}_k$ is only used in the first equation and is multiplied by $H_k$ in which only the first two elements are nonzero; therefore, a maximum size of 2 is required for the matrixes and vectors in the computation of the first equation in this example.

Another interesting consequence is that the unit lower triangular matrix $L$ in the $LDU$ decomposition will reduce to a $2 \times 2$ matrix with only one sub-diagonal element to be updated in each iteration, and thus, it can simply be considered as a scaler. The algorithm of Appendix C implements the RLS method based on the above observations.

A parallel structure for this algorithm is shown in Fig. 5-2; this is a simplified version for the structure proposed in [54]. At the left side, by arranging the input data as shown in the figure, $H_kL$ and $H_k\hat{x}_k-z_k$ are computed and passed to the processor $\mu_p$. Since $H(1)$ and $H(2)$ choose one of the four values ($\pm 1 \pm j$), the multipliers will be trivial. The input data is also buffered in a delay unit until the coefficients for the processors on the right hand side are calculated in the processor $\mu_p$. Once $a1$ and $a2$ are computed and loaded into
Implementing the Estimation Algorithm

FIG. 5-2 A parallel structure for implementation of the RLS Algorithm.

The right side processing units, and $bI$ and $A0$ are ready as inputs, the new values for the state estimates and the covariance factor $Lp$ will be computed.

The internal structure of the processor $\mu_p$ is shown in Fig. 5-3. It computes the required coefficients and the covariance factors of the vector $Dp$. Before any computation, the input values are first arranged in the buffers and then applied to the processing elements. It is helpful to note that different precisions are required at different processing units; for example for representing $\lambda$, which is the forgetting factor of the RLS algorithm, 4 bits are sufficient, and this will reduce the size of required hardware.
Implementing the Estimation Algorithm

In the final receiver structure, there is at least one channel estimation for each hypothesized transmitted signal during each symbol interval. Using only one estimator for this purpose slows down the process and building several identical units of the above structure might not be cost efficient. A proper solution to this problem is to seek pipeline methods where the same hardware can be utilized to carry out the required channel estimations at a higher speed and with more cost-effectiveness. The above structure is not suitable for pipelining, and the parallelism among the instructions of the algorithm in Appendix C has to be exploited to obtain an appropriate structure for this purpose.

It can be verified that the pipelined structure of Fig. 5-4 implements the above RLS algorithm. There are eight distinct pipeline stages, and the computational result of each one is passed to the subsequent stage. The new channel estimates and new covariance factors are computed in different stages; moreover, the proposed architecture computes the required branch metrics (BM) for the Viterbi algorithm. The overall speed-up compared to
Implementing the Estimation Algorithm

FIG. 5-4 The pipeline structure for implementing the RLS channel estimator and the branch metric generator.
the non-pipelined structure depends on the latency of the pipeline stage with maximum delay.

5.3 Implementation Algorithms for Time Update Equations

The Jover-Kailath method [54] can be used to implement the measurement update equations of the Kalman filter. Since the RLS algorithm is basically the same as the measurement update equations of the Kalman filter the Jover-Kailath method could also be used for implementing the RLS algorithm. However, to implement the Kalman filter we need to calculate the time update equations (3-7) and (3-8) and another algorithm is required to perform this part. In the Jover-Kailath algorithm instead of $P_{kk}$, its factors, $L_p$ and $D_p$ are computed as in (5-4), and therefore, we need to employ an algorithm that uses these factors for the computations. The propagation of LD factors implicitly preserves symmetry and non-negative definiteness of the computed covariance matrix. In the following we will present and compare three different methods for implementing the time update equations.

A. Direct Computation of the Covariance Matrix

One simple approach to carry out the computation in (3-8), is the direct computation of

$$P_{k+1} = FL_p L_p^T F^T + GQG^T$$

(5-14)

When the noise process is time invariant, $GQG^T$ needs to be computed only once and (5-14) requires four matrix multiplications and one addition. Then, since we need to propagate the LD factors of $P_{k+1}$ it can be factorized in the $LDL^T$ form using the factorization algorithms of [78]. The direct method is not very robust against round off errors, and we will use it for comparison to show the advantages of using square-root techniques in the implementation of time update equations. The following methods are based on the direct computation of LD factors for the covariance matrix and result in better numerical stability.
B. The Weighted Gram-Schmidt Orthogonalization (WGS) Algorithm

In this method the covariance update equation implementation is based on a Block Matrix Factorization. The equation (3-8) can be rewritten in the following matrix form

\[ P_{k+1} = \begin{bmatrix} FP_{k|k}^{1/2} & GQ_{k+1/2}^{1/2} \end{bmatrix} \begin{bmatrix} P_{k|k}^{T/2} & F^T \\ Q^{T/2} & G^T \end{bmatrix} \]  

(5-15)

Again if we use the \(LDU\) factorization for the covariance matrix as in (5-4) and indicate the diagonal matrix of \(Q\) with \(D_Q\), after dropping all time index subscripts equation (5-15) becomes

\[ P = LDL^T = \begin{bmatrix} FP & G \\ 0 & D_Q \end{bmatrix} \begin{bmatrix} L_p^T & F^T \\ 0 & G^T \end{bmatrix} \]  

(5-16)

\(L_p\) and \(D_p\) in the right side of the equation are known from the measurement update procedure, and \(L\) and \(D\) have to be computed.

The Weighted Gram-Schmidt Orthogonalization (WGS) [11] can be employed here. It is an algorithm for finding a set of \(n\) mutually orthogonal vectors \(b_1, b_2, b_3, ..., b_n\) that are a linear combination of a set of \(n\) linearly independent vectors \(a_1, a_2, a_3, ..., a_n\). For a given matrix \(A\) the Gram-Schmidt algorithm defines a unit upper triangular matrix \(U\) such that \(A=BU\), or

\[ A = \begin{bmatrix} a_1 & a_2 & a_3 & ... & a_n \end{bmatrix} = BU \]  

(5-17)

\[ = \begin{bmatrix} b_1 & b_2 & b_3 & ... & b_n \end{bmatrix} \begin{bmatrix} 1 & u_{12} & u_{13} & ... & u_{1n} \\ 0 & 1 & u_{23} & ... & u_{2n} \\ 0 & 0 & 1 & ... & u_{3n} \\ : & : & : & ... & : \\ 0 & 0 & 0 & ... & 1 \end{bmatrix} \]  

(5-18)

The Gram-Schmidt Orthogonalization is called weighted if the vectors \(b_i\) are orthogonal with respect to the weights \(w_1, w_2, w_3, ..., w_n\). The vectors \(x\) and \(y\) are said to be orthogonal with respect to the weights \(w_i\) if
Implementing the Estimation Algorithm

\[ \sum_{i=1}^{n} x_i w_i y_i = x^T D_w y = 0 \quad (5-19) \]

where

\[ D_w = \text{diag}_{1 \leq i \leq n} \{w_i\} \quad (5-20) \]

Hence the weighted norms of the mutually orthogonal vectors \( b_1, b_2, b_3, \ldots, b_n \), appear as the diagonal elements of the diagonal matrix

\[ D = B^T D_w B \quad (5-21) \]

To apply the Gram-Schmidt method let

\[ A = \begin{bmatrix} L_p^T F^T \\ G^T \end{bmatrix} \quad (5-22) \]

and

\[ D_w = \begin{bmatrix} D_p & 0 \\ 0 & D_q \end{bmatrix} \quad (5-23) \]

then from (5-16) we have

\[ P_{k+1} = A^T D_w A \quad (5-24) \]

The WGS algorithm will produce a unit upper triangular matrix \( U \) (5-17) and a diagonal matrix \( D \) (5-21) such that

\[ P_{k+1} = A^T D_w A \quad (5-25) \]

\[ = (BU)^T D_w (BU) \quad (5-26) \]

\[ = U^T B^T D_w B U \quad (5-27) \]

and from (5-21)

\[ P_{k+1} = U^T D U = LDL^T \quad (5-28) \]

Therefore, the inputs of the WGS algorithm are the information on the right side of (5-16) in the form of \( A \) and \( D_w \), and the output of the algorithm is the lower triangular matrix \( U^T = L \) and the diagonal matrix \( D \).
Implementing the Estimation Algorithm

The algorithm in Appendix D performs the temporal update for the covariance matrix based on the WGS algorithm. Before applying this algorithm a matrix multiplication is required to compute $FL_p$ (see (5-16)). Also for computing the temporal update of state estimations in (3-7) the $\hat{x}_{k|k}$ vector, obtained at the output of the measurement update procedure, must be pre-multiplied by $F$. Both of these multiplications can be carried out together by appending the vector $\hat{x}_{k|k}$ to the matrix $L$, and multiplying the combination by $F$. If the multiplication is carried out using array processors, in this way, the same structure can perform both of these multiplications, without any change in the hardware.

C. The LD Correction (LDC) Algorithm

In the WGS algorithm, as we can see from (3-8) and (5-15), the computation of $GQG^T$ will be repeated in every iteration. When the process noise is time invariant and the matrix $Q$ is not changing over time, there is no need to re-compute this term in every iteration. This idea leads to the introduction of a less complex algorithm.

The LDC algorithm is used in linear algebra [78][79] to update the LD factorization of matrix $A$ to the LD factorization of $A+vv^T$, where $A$ is symmetric and positive definite, and $v$ is an arbitrary vector with appropriate size. When the process noise covariance is time invariant, this algorithm can be used to implement the time update equations of the Kalman filter and it is shown to have substantial computational saving when compared to the Gram-Schmidt algorithm [77]. To implement this method let

$$P_{k|k} = \sum_{i=1}^{n} d_i L_i L_i^T \quad \text{(5-29)}$$

The covariance prediction in (3-8) can be written using the $L$ and $D$ factors of $P$ and $GQG^T$ as

$$P_{k+1} = \sum_{i=1}^{n} d_i (FL_i)(FL_i)^T + \sum_{i=1}^{p} d_q L_q L_q^T \quad \text{(5-30)}$$

The LDC algorithm can be employed to compute the $LDL^T$ factorization of the sum

$$\sum_{i} d_q L_q L_q^T + svv^T \quad \text{(5-31)}$$
In (5-30) LDC algorithm can be applied \( n \) times, and each time one of the components of the first sum is considered to be the \( svr \) vector. The result will be a \( LDL^T \) factorization for \( P_{k+1} \).

The complete algorithm to implement the LDC method is given in Appendix E. This algorithm requires \( O(6n^3+16n^2-2n) \) multiply-add operations, while the WGS method requires \( O(10.7n^3+11.8n^2-2.5n) \) operations. So the LDC algorithm requires less computation compared to the WGS method, since the process noise is time invariant and the term \( GQG^T \) needs to be factorized only once during the initialization stage and these factors will be used repeatedly during each filter iteration.

The performance of the above three methods are compared in section 5.6. The direct method is not a square root method and is very sensitive to numerical errors, the performance of the WGS and LDC algorithms are very close in terms of numerical accuracy, while the LDC algorithm requires fewer computations. In the next section we will introduce a systolic structure for the implementation of the LDC algorithm.

### 5.4 A Systolic VLSI Structure for the LDC Algorithm

To employ the Kalman filter as a channel estimator in a mobile communication receiver, it is important to carry out all of the required computations in real-time. The Kalman estimator is computationally intensive and to speed up the estimation process parallel VLSI structures have to be sought for implementation.

The LDC algorithm is more appropriate than other methods for implementation of the time update measurement equations of the Kalman channel estimator. The fading channel model and, hence, the process noise can be reasonably assumed to be time invariant in a short period of time (e.g. one frame interval). This allows us to employ the LDC algorithm which results in a considerable saving in computations compared to the WGS method.

A systolic VLSI structure is proposed in Fig. 5-5 for implementation of the LDC algorithm. This structure is used for implementing (5-31) and is based on the \( idltup \) function of Appendix E. Two types of processors are employed, and the function of each is described in the figure. The size of the state vector is assumed to be \( n=6 \) in this example.
Implementing the Estimation Algorithm

FIG. 5-5  A systolic VLSI structure for implementation of the LD correction algorithm.
Implementing the Estimation Algorithm

and the size of the $L$ matrix is $6 \times 6$. The LD factors get updated in place and there is no need to transfer these values during the computation. The inputs to this structure are different columns of the matrix $FL$ in the form of the $v$ vectors scaled by the elements of $D$. The function $ldltup$ is called $n$ times for each column of $FL$, and this can be carried out by applying $n$ input vectors to the systolic structure. Once the computation in one row is finished, the next column of $FL$ can be applied as the new input vector. This allows for pipelining and results in higher processor utilization and, hence, higher speed. If the computation in each row takes one time step, for the time update of the covariance matrix, $2n-1$ time steps will be required.

It is also possible to map the above algorithm to a smaller number of processors. Mapping can be performed along different directions as shown in Fig. 5-6. By using different mapping vectors we obtain structures with different performance and capabilities. The function of the processing units and the details of communication between units is not shown in the figure. Links only represent the direction of data transfer between processors. Table 5-1 summarizes and compares the features for different mapped structures, when $n$ is the size of the state vector. It is possible to employ the mapping along $d_3 = [-1, 0]$ in a pipeline mode, but the other mappings cannot be pipelined.

There are two kinds of processors used in each structure. If we assume that the maximum time required for the computations in each processor is $T$, then the total latency for one application of the correction algorithm will be $11T$. This should be obvious from the data dependency in the two dimensional structure of Fig. 5-6(a). By using the structure of Fig. 5-6(a) in pipeline mode, the throughput, or the time interval between any two applications of the algorithm, will reduce to $T$. In this case processor utilization will be 100%. It can be shown that the structure of Fig. 5-6(d) can also be pipelined. There are 11 processors used here and a throughput of $5T$ is attainable in this case. The other two
FIG. 5-6  Mapping the systolic structure of the correction algorithm to a smaller number of processors.
mappings use 6 processors and cannot be pipelined, resulting in a latency and throughput of \(11T\).

**Table 5-1** Comparison between different mappings of the systolic structure (\(n=\) size of the state vector).

<table>
<thead>
<tr>
<th></th>
<th>(a) The two dimensional array</th>
<th>(b) Mapping along (d_1)</th>
<th>(c) Mapping along (d_2)</th>
<th>(d) Mapping along (d_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Processors</td>
<td>(n(n+1)/2)</td>
<td>(n)</td>
<td>(n)</td>
<td>(2n-1)</td>
</tr>
<tr>
<td>Pipelineable</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Latency</td>
<td>((2n-1)T)</td>
<td>((2n-1)T)</td>
<td>((2n-1)T)</td>
<td>((2n-1)T)</td>
</tr>
<tr>
<td>Throughput</td>
<td>(T)</td>
<td>((2n-1)T)</td>
<td>((2n-1)T)</td>
<td>((n-1)T)</td>
</tr>
<tr>
<td>Processor Utilization</td>
<td>1.0</td>
<td>((n+1)/(4n-2))</td>
<td>((n+1)/(4n-2))</td>
<td>((n+1)/(4n-2))</td>
</tr>
</tbody>
</table>

### 5.5 Structures for Viterbi Detection and Channel Estimation

In the Viterbi equalization of a mobile fading channel, a channel estimator has to be used in combination with the Viterbi detector. The impulse response of the channel has to be estimated by the estimation algorithm, and this impulse response is used to generate the branch metrics in the Viterbi algorithm. In the PSP equalization method there is one channel estimation associated with each state of the Viterbi trellis. All of the channel estimations associated with different states, depend on the received signal and the hypothesized data sequence in the Viterbi algorithm and can be performed in parallel.

Fig. 5-7 shows how to employ the algorithms introduced in chapter 3 to estimate the CIR and calculate the branch metrics for one node of the trellis using the PSP method. The estimator consists of the Jover-Kailath and LDC algorithms. The inputs and outputs are shown based on the parameters introduced in the above algorithms. There is one Branch Metric Generator (BMG) for each of the \(m\) branches leading to the considered node. Each
Implementing the Estimation Algorithm

FIG. 5-7 The required PSP computation for one node (state) of the trellis diagram. First, the branch metrics are generated and the survivor path is found, then the estimator will receive the survivor path information to estimate the channel.
branch has its own hypothesized data sequence, $H_k^i$, corresponding to the state transition on that branch. The Node Processor unit receives all the branch metrics and determines the survivor path using the Add-Compare-Select (ACS) operations. After the survivor path is known, the appropriate values for $H_k$, $L$, $D$, and $\hat{x}_k$ have to be sent to the estimator to update the channel estimates. For $H_k$, the hypothesized data sequence of the survivor path has to be chosen, and for $L$, $D$, and $\hat{x}_k$, the output of the channel estimator on the node from which the survivor path is originating has to be considered.

The Per-Branch Processing (PBP) method of (3-17) is a generalized form for PSP and can be used when there is more than one sample and more than one estimation per symbol interval. In PBP there is one channel estimation on each branch of the Viterbi trellis to generate the branch metrics. In a parallel implementation several estimators can be employed to generate the branch metrics. The number of estimators required is equal to the number of states in PSP and equal to the number of branches in PBP. In the following we propose parallel structures for implementation of PSP and PBP.

In a parallel structure for the Viterbi algorithm different estimators must communicate with the Viterbi processing units so that independent computations can be handled in parallel and also pipelined if possible. To obtain a systolic structure for the Viterbi algorithm, we realize that, as explained in section 4.2, it can be formulated as a form of matrix multiplication. Therefore it is possible to consider the systolic architectures for vector-matrix multiplications proposed in the literature to implement the Viterbi algorithm. The following structures are based on two different systolic designs proposed for matrix-vector multiplication in [15].

The parallel structure of Fig. 5-8 is proposed for the joint implementation of the estimator and the Viterbi algorithm. This structure is performing parallel state-computations since the computations for all of the states are performed in parallel by considering the incoming branches sequentially. There is one channel estimation on every branch of the trellis and hence, the equalization is performed in a PBP fashion. The buffer on top contains four sets of data for different trellis nodes. $X$ is the estimated state, $L$ and $D$ are covariance factors, $\Gamma$ is the likelihood ratio in the Viterbi algorithm, and $P$ contains the
Implementing the Estimation Algorithm

PATH INFORMATION (SURVIVOR SEQUENCE) OBTAINED IN THE VITERBI ALGORITHM. ALL OF THE DATA SETS ARE PROPAGATED TO THE FOUR ESTIMATORS SEQUENTIALLY. EACH ESTIMATOR ALSO RECEIVES THE HYPOTHESIZED TRANSMITTED DATA, $H$, RELATED TO THE BRANCHES THAT END IN THE SAME NODE, AND THE RECEIVED SIGNAL $z_k$. IN FOUR SUBSEQUENT PIPELINE StAGES THE CHANNEL ESTIMATES AND THE BRANCH METRICS ARE COMPUTED. EACH ESTIMATOR COMPUTES THE VALUES RELATED TO THE INCOMING BRANCHES TO ONE NODE. THE OUTPUT OF THE BMG IS PASSED TO THE ACS UNIT. THE ACS UNIT ADDS THE ORIGINAL NODE VALUE AND THE COMPUTED BRANCH METRIC, AND SELCTS THE SMALLEST

FIG. 5-8 A parallel architecture for joint data detection and channel estimation (parallel state-computation structure).
obtained value. This determines the survivor path at the end of four pipeline stages. The information of each surviving path, including the states of the channel estimator are passed to the top buffer to be used at the next symbol interval. The new values will be written into memory after all of the states are updated and all of the computations for the current symbol interval are finished. After processing a number of received symbols there will be an agreement on the shortest path up to a certain point which depends on the depth of the algorithm and signal-to-noise ratio. This part of the path information can be written into the survivor path memory and it determines the detected sequence.

In the PSP method the number of channel estimations is reduced to the number of states and in the above example (see Fig. 4-6) only four estimations are required in each symbol interval. In this case, for all of the incoming branches to a node, first the survivor branch will be determined and then only the channel estimation associated with the survivor branch will be carried out. The parallel structure of Fig. 5-8 can be modified for the implementation of PSP. In any of the four parallel branches, the estimator has to be moved after the BMG and ACS units. First, the ACS unit determines the survivor branch and then the estimator computes the CIR based on the information corresponding to that branch.

Fig. 5-9 is another approach to the above problem. It is also derived from one of the systolic designs for matrix-vector multiplication in [15]. This structure is performing \textit{sequential state-computations} since the computations for all of the states are performed sequentially by considering the incoming branches in parallel. The computations associated with all incoming branches to a node will be done simultaneously in four different estimators. After all four branch metrics are added to the corresponding node values, the compare-select operations will be performed in a tree structure to find the minimum value. The updated states and covariance factors corresponding to the survivor branch will be directed to the buffer on top. Since in this structure the updated states and new values for the nodes are computed in a sequential way, they need to be buffered until all of the new state values are computed and old values can be over-written by the new values. This might be done by doubling the size of the top buffer. When different states
Implementing the Estimation Algorithm

agree on a common ancestral path, the information related to that path can be dumped into the survivor path memory.

The sequential state-computation structure can be simplified significantly for the PSP method. As described before, for PSP there is only one channel estimation for the survivor path and, hence, the estimator has to be moved after the add-compare-select operations. In
the structure of Fig. 5-9, four estimators will be removed and one estimator will be used after the final compare-select unit. The channel estimator usually requires a considerable amount of computation power and area in a VLSI design. Therefore, reducing the number of estimators to only one is an effective step in the simplification of the VLSI structure. If the structure of the channel estimator is pipelineable, it will speed up the necessary computations for all the states.

5.6 Hardware Implementation of the Estimator

The estimation algorithms of chapter 3 have to be realized with digital hardware, where state values and coefficients are stored in registers with a finite number of bits. An important issue in the implementation of a filtering algorithm is to consider the problems that arise in dealing with floating-point computation and finite wordlength. The time consumed on the computations and the area used in the VLSI implementation of an algorithm are proportional to the wordlength used in the computations for addition and proportional to its square for multiplication. Therefore, it is always important to use as few a number of bits in the wordlength of the computations as possible.

Some estimation algorithms, like the Kalman filter, are more sensitive to roundoff errors and require a larger number of bits per computation word compared to other estimators. The finite-wordlength effects on the design of the Kalman filter have been addressed somewhat in the literature [80][81], however in these analysis the estimation accuracy has been the main concern of the authors. Here, we study the performance of the estimator in a Rayleigh fading channel based communication system and specifically in the context of a Viterbi based receiver. Hence, we will have to note the overall effect of roundoff noise and estimation accuracy on the BER performance in this receiver.

In an adaptive MLSD receiver the CIR is estimated along the surviving paths associated with each state of the trellis. The quality of the channel estimation method has a strong impact on the overall BER performance of the receiver. Particularly in fast fading conditions, only more advanced and more accurate channel estimators can provide reasonable receiver performances. The estimator precision depends on the employed
Implementing the Estimation Algorithm

FIG. 5-10 The effects of changing the wordlength on estimation methods (Eb/N_0 = 15 dB). Kalman-1 represents the Gram-Schmidt method, Kalman-2 is for the correction method, and Kalman-3 is the direct method. PSP method is employed for detection.

algorithm, the wordlength, and roundoff noise which is inevitable in the hardware implementation. We will consider both the estimation accuracy and BER performance by studying the simulation results.

In Fig. 5-10 the expectation of the mean square error in estimation of the impulse response of a Rayleigh fading channel is plotted versus the mantissa wordlength in the floating point operations. The simulated system is as described in section 4.4. Three different implementation methods of section 5.3 for the Kalman estimator are considered along with the RLS and LMS algorithms, when Eb/N_0 is 15 dB (E_b is energy per
Implementing the Estimation Algorithm

FIG. 5-11 The effects of changing the wordlength on estimation methods (E_b/N_0=15 dB). See Fig. 5-10 for details.

transmitted bit). For the measurement update of the Kalman filter and also for the RLS estimator, the square-root method of section 5.2 has been used. As is clear from Fig. 5-10, with the Kalman filter the minimum achievable MSE is much lower than that of LMS and RLS algorithms. However, a larger number of bits is required for the Kalman filter. Direct implementation of the Kalman filter requires at least 26 bits per mantissa while two other methods require 22 bits. The minimum number of bits per mantissa required for channel estimation with the RLS and LMS algorithms are 12 and 8 respectively.
Implementing the Estimation Algorithm

The effect of reducing the number of bits on the overall BER performance is shown in Fig. 5-11. Joint data detection and channel estimation is performed using the PSP method. Using the Kalman channel estimator leads to a very good BER performance. The BER performance of the Kalman filter is about 10 dB better compared to the RLS and LMS estimators, while it requires a longer wordlength. The required mantissa length, obtained from Fig. 5-11, is less than what we would expect by observing the mean square error of the estimator. Also we can see that using the WGS method or the LDC algorithm for the time update equations of the Kalman filter result in the same BER performance while the latter is cheaper and has less computation involved. Both of these methods are much more efficient compared to the direct method which uses a non square-root algorithm for the time update equations.

5.7 Summary

After a brief introduction to the square-root filtering method, the Jover-Kailath algorithm was used in this chapter for the implementation of the measurement update equations of the Kalman filter. Parallel and pipeline structures were proposed for this algorithm to be used in the implementation of the Kalman filter and the RLS estimator.

For the implementation of time update equations in the Kalman filter three different methods were studied. The direct method is the most complex, and the WGS method is relatively less complex. However the complexity of the LDC method is less than other methods and a systolic structure is proposed for its implementation.

The overall structure of an MLSD receiver, implementing the Viterbi detection and Kalman estimation, is studied. The proposed structures can be used for both PSP and PBP implementation. Finally, the required word length for the digital implementation of the estimation algorithm is studied. The LMS and RLS require a small number of bits per mantissa, and the Kalman filter requires a longer digital word length. The word length required for LDC and WGS are identical, while LDC offers reduced complexity.

The next chapter provides a different approach for defining the state space model of the Kalman filter and this will be used to simplify the digital implementation of the algorithm.
Chapter 6

Differential Kalman Filtering

Generally all adaptive versions of MLSD receivers require estimation algorithms for identifying the parameters of time varying channels. In mobile communication systems, an estimator is used for tracking the time varying states of the fading channel. The quality of the channel estimation method has a strong impact on the overall Bit Error Rate (BER) performance of the receiver. Therefore, a key factor in the receiver design is the estimation of the fading channel with high accuracy [1].

The conventional Kalman filter is an optimum estimator; however, it is computationally demanding and very sensitive to round-off errors. It requires the exact parameters of the state space model and the second order statistics of the random model-parameters. A bank of estimators are required to implement PBP or PSP, one for each branch or each survivor sequence, for channel tracking and branch metric computation. By choosing the Kalman filter for channel tracking, the complexity of the receiver can be prohibitive, particularly for channels with a long impulse response duration.

Two major design approaches have been used to overcome the design complexity of the Kalman based MLSD receiver. The first approach is to take advantage of recent advances in VLSI technology in parallel information processing by proposing parallel algorithms and structures for the implementation of Kalman filter [59][60][49]; this issue was the main focus of chapter 5. The second direction is to look for suboptimal and simpler
variants of the Kalman filter [69][14], which involves a trade-off between performance and complexity.

The purpose of this chapter is to introduce a new method to define the Kalman filter algorithm, in order to attain a simpler implementation while preserving the robustness of the algorithm. A new approach is proposed for the implementation of the Kalman filter based on differential channel states. This leads to a robust differential Kalman filtering algorithm that can be simplified further to ease the implementation without any major loss in performance [61].

Here we propose a different method to define the state space model of the channel from what has been reported in the literature so far. To derive the ARMA model of the channel impulse response, usually the consecutive instances of the impulse response are used as the basis [49][31]. We will show that by choosing the impulse response and its time derivatives as an equivalent set of basis, the Kalman filter algorithm remains unchanged and only a new set of parameters are used.

With the new definition of states, the Kalman filter becomes more robust against simplifications made to reduce the implementation problems. The complex covariance matrix can be simplified to a reduced size real matrix to mitigate the computational complexity. Also the state transition matrix can be rounded to have only one and zero elements. The simplifications are aimed towards obtaining an LMS-type algorithm from the optimal Kalman filter, identifying it as a special case of the Kalman filter.

In the following sections we start with the formulation of conventional Kalman filtering. Then by using a special transformation a new set of basis will be used to obtain the differential Kalman filter equations. We will show that in the new algorithm the transition matrix can be rounded or forced to be a very simple upper triangular matrix. Also the covariance matrix can be approximated by a simpler real symmetric matrix that can reduce the complexity of the computations. Simulation results show that the above simplifications do not cause a major degradation in the performance of the Kalman filter.
6.1 The Differential Bases for the Kalman filter

The state space model of (2-22) and (2-24) represents a linear time-variant system described by a Markov model. This model has been used in chapter 2 to describe a Rayleigh fading mobile communication channel. The states of this system are defined as the time varying impulse response of the fading channel. The Kalman filter can be employed as a channel estimation algorithm to estimate the state vector of this system, or in other words the CIR of the fading channel, based on the noisy received signal samples.

In the following, first we repeat the standard Kalman filtering solution for estimating the CIR. Then we introduce the Differential Kalman filtering method and will study the practical effects of this method on performance and implementation.

The Conventional Kalman filter

The Kalman filter can recursively estimate the states of the linear system defined in (2-22) and (2-24). The recursion equations of the Kalman filter can be written from (3-3)-(3-8) as:

\[ R_k = H_k P_k H_k^T + N_o \]  \hspace{1cm} (6-1)

\[ e_k = z_k - H_k \hat{x}_k \]  \hspace{1cm} (6-2)

\[ \hat{x}_{k+1} = F \hat{x}_k + F P_k H_k^T R_k^{-1} e_k \]  \hspace{1cm} (6-3)

\[ P_{k+1} = F (P_k - P_k H_k^T R_k^{-1} H_k P_k) F^T + GQG^T \]  \hspace{1cm} (6-4)

where \( \hat{x}_k \) is the state estimate and \( P_k \) is the state estimation error covariance matrix. The Kalman filter is an optimum estimator that minimizes the mean squared estimation error.

In order to obtain a numerically accurate and stable algorithm, the algorithm has to be implemented with a rather long digital word-length and square-root solutions should be used for more robustness [53]. In general the implementation of the Kalman filter requires complex hardware, and the complexity grows very quickly for larger state vector size. It is usually common practice to make a trade-off between cost and performance of the algorithm. One approach along these limits is to look for suboptimal versions of the
Kalman filter trying to keep the performance acceptable while reducing the implementation cost.

Another issue with the implementation of a Kalman filter for data detection over fading channels is that it requires some information about the channel model parameters, such as $F, G, Q,$ and $N_0$. This information is not easily available at the receiver and it has to be extracted from the received signal which leads to state estimation with model uncertainty. In practice, it is reasonable to assume that the parameters of the channel model are almost constant compared to the variations of states over a certain time interval. In this way constant channel parameters can be used in the Kalman filter as long as they are valid, say, over one or a few data blocks. Some receivers avoid this problem totally or partially by employing estimation algorithms that require a minimum amount of a priori information about the channel, e.g. the LMS algorithm only considers the received signal.

In the following we propose a solution to mitigate the above problems. This is in the form of a change of basis in the state space model, which leads to a simpler implementation for the Kalman filtering algorithm.

**The Differential Kalman Filter**

We will show that the implementation of the channel estimation and tracking process will be enhanced if we use another set of states in the state space model. This is equivalent to choosing a new set of basis in the same space. The new set of basis will be obtained by applying a transformation matrix to the equations that describe the state space model and also the equations of the Kalman filtering algorithm. Obviously, the new approach does not provide any more information for the algorithm, and hence, the theoretical performance of the channel estimator remains unchanged. However, under the new transformation the implementation can be accomplished with reduced complexity.

In (2-20) we defined the state vector at time $k$ consisting of the impulse response at time $k, h_k,$ and its past values. We replace the state vector of (2-20) with a new state vector. The elements of the new set of basis at time $k$ are $h_k$ and its first and second order time derivatives. The new $3q$-dimensional channel state vector can be written as
It is easy to verify that the new state vector can be obtained from the old state vector by using a transformation as

$$x_k = T x_k$$  \hspace{1cm} (6-7)

where $T$ is the $3q \times 3q$ \textit{Differencing Matrix} as

$$T = \begin{bmatrix} I & 0 & 0 \\ I & -I & 0 \\ I & -2I & I \end{bmatrix}$$  \hspace{1cm} (6-8)

where $I$ and $0$ are $q \times q$-identity matrix and $q \times q$-zero matrix, respectively. It is very interesting to note that we have $T = T^{-1}$, and this property will result in some simplifications in deriving the new form for the Kalman filter from the old set of equations. This result can be extended to larger matrix sizes and a general formulation can be given for the transformation. In general for an $nq$ dimensional state vector the components of the \textit{Differencing Matrix} can be expressed as

$$T_{i,j} = \begin{cases} (-1)^{j-1} \binom{i-1}{j-1} I & j \leq i \\ 0 & j > i \end{cases}$$ \hspace{1cm} (6-9)

This is a lower triangular matrix and the elements of each row are the coefficients of a binomial series.

To define the state space model based on the new basis, the above transformation can be applied to both sides of (2-22) and (2-24). The state space model will be obtained as

$$x_{k+1} = E x_k + G w_{k+1}$$  \hspace{1cm} (6-10)

$$z_k = H_k x_k + n_k$$  \hspace{1cm} (6-11)

and the Kalman filtering algorithm of (6-1)-(6-4) will be transformed to
\begin{align*}
R_k &= H_k p_k H_k^T + N_o \\
e_k &= z_k - H_k \hat{\hat{x}}_k \\
\hat{\hat{x}}_{k+1} &= F \hat{\hat{x}}_k + F P_k H_k^T R_k^{-1} e_k \\
P_{k+1} &= F (P_k - P_k H_k^T R_k^{-1} H_k P_k) F^T + G Q G^T
\end{align*}

where \( F = T F T \), \( G = T G \), and \( P_k = T P_k T^T \). Also note that \( H_k T = H_k \). Hence, we realize that by using the above transformation the algorithms for conventional Kalman filter and differential Kalman filter will be the same. This result was expected, since by applying the above transformation we obtain a state space model similar to that of Fig. 2-9 with different parameters. Therefore the equations of the Kalman filtering solution remain unchanged and only the old parameters are replaced by the new ones.

In the following section we will show that applying the above transformation matrix enables us to simplify the implementation of the Kalman filter and reduce its computational complexity without a major loss in performance.

### 6.2 Simplifying the Transition Matrix

The elements of the transition matrix \( F \) are derived from the coefficients of the fading filter (see (2-4), (2-21), and (2-22)). The coefficients of the fading filter are not a priori known to the receiver and they need to be estimated. Moreover, the transition matrix \( F \) is present in both terms of the state update equation and also in the covariance update equation of the Kalman filter. Therefore, a considerable amount of computational power is consumed for the multiplications related to this matrix.

Now we can show that for the differential Kalman filter the transition matrix can be rounded to a very simple form with small approximations. The new matrix form consists of only zeros and ones and this makes it ideal for digital implementation.

**Rounding \( F \) to make an upper triangular matrix**

The transition matrix \( F \) can be computed as

\[
F = T F T = \begin{bmatrix}
(A + B + C)I & (-B - 2C)I & CI \\
(A + B + C - 1)I & (-B - 2C)I & CI \\
(A + B + C - 1)I & (-B - 2C - 1)I & CI
\end{bmatrix}
\]
In the following we will show that in practical situations the components of this matrix can be rounded to obtain an upper triangular matrix with all nonzero elements equal to one as

\[
E \approx \begin{bmatrix}
  1 & 1 & 1 \\
  0 & 1 & 1 \\
  0 & 0 & 1
\end{bmatrix}
\]  

(6-17)

This is very appealing for the digital implementation of the algorithm and the complexity is reduced dramatically. Matrix multiplication will only consist of additions and no multiplication is required.

To study the possibility of rounding the elements of matrix $E$ consider Fig. 6-1. The coefficients of the fading filter, or the elements of matrix $F$ are plotted in a two dimensional plane against vehicle speed and normalized sampling rate. The fading filter is designed based on the specifications given in section 2.2. To obtain the coefficients a Rayleigh fading channel, and similar to the IS-54 standard a symbol rate of 25 ksymbols/s in the 900 MHz band is assumed. It can be observed from the Fig. 6-1 that for lower vehicle speed and higher sampling rates, the coefficients are more stable and can be rounded to the nearest integer value with small approximation.

The coefficients of matrix $E$ are mainly made of three elements: $C$, $A+B+C$, and $-B-2C$. The parameter $C$ can be approximated by one as can be observed from Fig. 6-1. The variations of $A+B+C$ and $-B-2C$ versus vehicle speed and sampling rate are illustrated in Fig. 6-2. It is clear that these two elements are very stable with the changes in vehicle speed and sampling rate and can be approximated by one with a very high accuracy. This approximation converts the matrix $E$ to an upper triangular matrix as in (6-17); and the state transition equation can be written from (6-10) as

\[
\begin{bmatrix}
  \dot{h}_{k+1} \\
  \dot{\dot{h}}_{k+1} \\
  \ddot{h}_{k+1}
\end{bmatrix} =
\begin{bmatrix}
  I & I & I \\
  0 & I & I \\
  0 & 0 & I
\end{bmatrix}
\begin{bmatrix}
  h_k \\
  \dot{h}_k \\
  \ddot{h}_k
\end{bmatrix} +
\begin{bmatrix}
  I \\
  I \\
  I
\end{bmatrix} Dw_{k+1}
\]

(6-18)

Similar results can be obtained based on the definition of differential states. We have

\[
h_{k+1} = 
\dot{h}_k + \ddot{h}_{k+1} = h_k + \dot{h}_k + \ddot{h}_{k+1} = h_k + \dot{h}_k + \ddot{h}_k + \dddot{h}_{k+1}
\]

(6-19)
FIG. 6-1 Variation of the coefficients of matrix $F$ with the vehicle speed and sampling rate. (Rayleigh fading channel, Symbol rate = 25 ksymbols/s, 900 MHz band)
FIG. 6-2  Variation of the coefficients of matrix $F$ with the vehicle speed and sampling rate. (Rayleigh fading channel, Symbol rate = 25 ksymbols/s, 900 MHz band)
Hence, it is easy to verify
\[
\begin{bmatrix}
    \bar{h}_{k+1} \\
    \bar{\bar{h}}_{k+1} \\
    \bar{x}_{k+1}
\end{bmatrix} = \begin{bmatrix}
    I & I & 0 \\
    I & I & 0 \\
    0 & 0 & I
\end{bmatrix}\begin{bmatrix}
    h_k \\
    \bar{h}_k \\
    \bar{x}_k
\end{bmatrix} + \begin{bmatrix}
    I \\
    0 \\
    0
\end{bmatrix}\bar{h}_{k+1}
\] (6-20)

By comparing (6-18) and (6-20) we can see that the approximation in rounding the elements of the \( E \) matrix is equivalent to assuming that \( \bar{h}_{k+1} \) is a zero mean white Gaussian process. This makes the \( E \) matrix to be upper triangular with only ones and zeros. In (6-20), \( \bar{h}_{k+1} \) is a random process and to implement the Kalman filtering algorithm its diagonal covariance matrix is required, which can be defined as

\[
E(\bar{h}_k\bar{h}_j^T) = Q(k-I)
\] (6-21)

It is interesting to note that with the previous form of the channel state definition the matrices \( F \), and \( G \) (or \( F \) and \( Q \)) were required for the implementation of the Kalman filter. In this case, from (6-20) one can realize that the above matrices become trivial. This simplifies both the computations and also the task of estimating the channel model parameters.

In the following section we will try to go one step further and take advantage of the properties of the differential Kalman filter to come up with a more simplified algorithm, while maintaining acceptable performance.

### 6.3 Simplifying the covariance matrix

We can reduce the computational burden of the differential Kalman filtering algorithm, (6-12)-(6-15), even further by simplifying the covariance matrix \( P_k \). After simplifying the \( E \) matrix, most of the computational power is spent on computations related to \( P_k \). This is a \( 3q \times 3q \) complex matrix with Hermitian symmetry. It is possible to consider a reduced form for \( P_k \) to simplify the calculations with a small effect on performance.

We propose a Hermitian symmetric form for the covariance matrix as

\[
P_k = \begin{bmatrix}
    \mu_{1,1} I & \alpha_{1,1} I & \beta_{1,1} I \\
    \alpha_{1,1} I & \alpha_{2,2} I & \beta_{2,2} I \\
    \beta_{1,1} I & \beta_{2,2} I & \beta_{3,3} I
\end{bmatrix}
\] (6-22)
where $\mu_{i,k}$, $\alpha_{i,k}$, and $\beta_{i,k}$ are real scalers. This will simplify the computation of the Kalman filtering equations drastically as $q^2$ complex values are replaced by one real scalar. To interpret the results of the above simplification one can consider the state update equation of (6-14) as two stages. First is the measurement update equation as

$$\hat{x}_{k|k} = \hat{x}_k + P_k H_k^T R_k^{-1} e_k$$

(6-23)

where $\hat{x}_{k|k}$ is the estimate of the CIR updated based on the information of the received signal. Second is the time update equation for the channel state based on our knowledge of the system model as

$$\hat{x}_{k+1} = E \hat{x}_{k|k}$$

(6-24)

The measurement matrix $H_k$ of (2-23) can be written as

$$H_k = \begin{pmatrix} 2q & \text{zeros} \\ D_k & 0, 0, \ldots, 0 \end{pmatrix}$$

(6-25)

where $D_k$ represents the $1 \times q$ nonzero part of the measurement matrix. Using (6-25) and (6-22) the measurement update equation of (6-23) can be viewed as three LMS-type update equations for the states and the differential states of different orders

$$\hat{h}_{k|k} = h_k + R_k^{-1} \mu_{1,k} D_k^T e_k$$

(6-26)

$$\hat{h}_{k|k} = \hat{h}_k + R_k^{-1} \alpha_{1,k} D_k^T e_k$$

(6-27)

$$\hat{h}_{k|k} = \tilde{h}_k + R_k^{-1} \beta_{1,k} D_k^T e_k$$

(6-28)

where $R_k^{-1} \mu_{1,k}$, $R_k^{-1} \alpha_{1,k}$, and $R_k^{-1} \beta_{1,k}$ serve as the step-size parameter of the LMS algorithm. However, these parameters are not fixed as in a regular LMS method, and will be updated adaptively by the covariance equation of (6-15). The computation of (6-15) can be simplified one step further by reducing the matrix $I$ to one in $E$, $P_k$ and $Q$ and also approximating $D_k^T D_k$ with the constant value of $\|d_k\|^2$. In this case the size of the matrices in (6-15) reduces from $3q \times 3q$ to $3 \times 3$ leading to a simpler computation.

Similar to the approach of Sayed and Kailath in [14] we can see that the LMS algorithm, like RLS, is a simplified variant of the Kalman filter. This can be verified if we let $E$ be the
identity matrix and in (6-22) have all of the parameters but $\mu = \mu_{1,k}$ equal to zero, and $\mu$ can be constant as in LMS.

The conventional Kalman filter is very sensitive to numerical errors. The simplification of the covariance matrix as in (6-22) leads to very poor BER performance and occasional divergence of the adaptive algorithm in our simulations of a channel estimator with conventional Kalman filter. However the differential Kalman filter allows for the above simplifications with a small degradation in performance as will be shown in the performance results.

6.4 Square-root Implementation

As another interesting feature, we note that the differential Kalman filtering method conforms very well with the square-root implementation techniques. As seen in chapter 5, in the square-root methods the $LDU$ factorizations of the covariance matrix are propagated in the recursive algorithm to preserve its Hermitian symmetry. To obtain the $LDU$ factors of $P_k$ and $P_{k|k}$ we can simply use the definition $P_k = TP_kT^T$. Then it is easy to note that by applying the differencing matrix we have

$$P_k = TP_kT^T = TLDL^T T^T = LDL^T$$

and since the $T$ transformation is lower triangular then $L$ remains lower triangular, too.

The $LDU$ factors of $P_{k|k}$ can also be defined in the same way.

Therefore, the algorithms and the structures that are employed for the implementation of the conventional Kalman filter, can be used for the differential Kalman filter only with different values for the parameters.

6.5 Performance Results

We consider a simulation system as described in section 4.4. The fading channel is simulated as a symbol-spaced two-path model with time varying complex coefficients. The two fading paths are independent with equal strength. The modulation scheme is differentially coherent QPSK, with a symbol rate of 25 ksymbols/s.
The differential Kalman filter was introduced using a new set of basis in the state space model. The algorithm for this filter is similar to that of the conventional Kalman filter with different parameters. However, the simplifications made to mitigate the implementation problems can provide a more practical and a less complex algorithm. The complex error
covariance matrix was approximated by a real matrix of reduced size. The transition matrix was also modified to contain ones and zeros which simplifies matrix multiplications. Finally, we showed that the LMS algorithm can be obtained along with these simplifications. This is similar to the approach of [14] and proves that LMS also belongs to the family of the Kalman filter variants.
Chapter 7

Conclusions and Future Research

7.1 Conclusions

This work started with the aim of studying implementation issues related to the Kalman based MLSD receivers. The receiver type had to be defined before proposing algorithms for implementation. This in turn, required a definition of the conditions and the propagation environment that this application is targeting. Following the work by Dai and Shwedyk [31], attempts were made to perform the joint channel estimation and data detection in a combined structure including the Viterbi algorithm and the Kalman filter. This naturally culminated in a PSP MLSD solution for detection. The matrices in the Kalman filter algorithm had to be found based on the channel parameters, and this led to our third-order-modeling of the fading channel and the related studies. This is a simple model that can be completely defined by one parameter, which is the maximum Doppler frequency shift. It was shown that a trade-off between performance and complexity can be made by considering a lower order fading filter at the receiver.

The role of different estimators was studied in tracking the fast fading channels. The estimators have usually been compared by considering the MSE performance measure. For the application of channel estimation the BER performance measure is more
appropriate. We compared the estimators in an MLSD receiver based on this measure; and the Kalman filter was shown to have superior performance compared to other estimators.

In data detection over fading channels the PSP-MLSD method appeared to be insufficient for multisampling (when there were more than one channel estimation per symbol interval), our solution was the PBP method and we offered structures that can implement both PSP and PBP.

The implementation of the Kalman filter was divided in two parts. For the measurement update equations the Jover-Kailath [54] approach was adopted and VLSI structures were proposed for implementation. We also showed how the Jover-Kailath algorithm could be used in parallel and pipelined structures for the implementation of the RLS algorithm.

The LDC algorithm was adopted to implement the time update equations of the Kalman filter. We compared this algorithm to the WGS algorithm and a direct method of implementation. The LDC algorithm was computationally less complex; and we developed systolic structures for its implementation.

Finally, a new method of differential Kalman filtering is proposed, which seems promising to simplify the implementation of the Kalman filter. In this approach we used differential bases for the states of the Kalman filter. This led to very interesting simplifications where the matrix $F$ reduced to a matrix with elements of one and zero. This result is appealing for digital implementation, since the matrix multiplication can be performed solely by addition. We also showed that a LMS-type algorithm can be obtained with further simplifications to the differential Kalman filter.

### 7.2 Major Contributions

The results of this research are published in [44][55][58][59][60][61] and the major contributions found in this work can be summarized as follows:

- **(i)** Simplification of Clarke’s model to a third order model resulting in a third order fading filter. Deriving an autoregressive model for the impulse response from the fading filter. Establishing the mathematical relations between the coefficients of the fading filter and the parameters in the Kalman filtering algorithm.
Conclusions and Future Research

(ii) Studying the effects of different system parameters of the estimation algorithm, on the overall BER performance of the receiver.

(iii) Proposing the PBP method as a general case of PSP and developing structures for joint data detection and channel estimation using PSP and PBP.

(iv) Proposing several parallel and pipelined VLSI structures for the implementation of the Kalman filter and the RLS algorithm, including the systolic structures for the realization of LDC method.

(v) Development of the novel differential Kalman filtering method, and studying the resulting simplifications in the implementation of the Kalman filter.

7.3 Future Research

Some aspects of this work can be followed in further studies. Estimation algorithms can be used at the receiver for estimating a priori unknown channel parameters, such as symbol timing, as well as the channel impulse response. A detection algorithm for fast fading channels is proposed in [33], where estimators are used to estimate the CIR and the symbol timing in a Viterbi algorithm structure. There are several estimators, each conditioned on one of the survivor sequences in the trellis. The structures studied in this work can be extended to accommodate the joint estimation of the channel parameters and the symbol timing.

Adaptive equalization in conjunction with diversity transmission can combat the adverse effects of frequency selective and frequency non-selective fading. The estimation algorithms such as the Kalman filter can be used in the receiver for tracking the weights of a diversity combiner. In [82], Adachi studied an adaptive diversity reception for M-ary differentially encoded PSK signals. In this method diversity combining weights are adaptively estimated along the surviving paths in the trellis diagram of the Viterbi algorithm. The PSP-MLSD structures of this research conform very well with these solutions for adaptive computation of weights of a diversity combiner.
In this work we have studied the VLSI implementation of the algorithms at the structural level. Further research can be done to study the implementation issues at the circuit level. This will involve a complete design of the digital hardware that can be implemented using ASIC or available DSP cores.

The differential Kalman filtering method proposed in chapter 6 needs further studies on convergence speed and stability and the required number of bits in digital implementation. To achieve this, more simulations need to be performed.
Appendix A

Design of The Fading Filter

The following Matlab program is used to design the fading filter and generate the coefficients of (2 - 4). The filter is a cascade of a second order filter with 9 dB peak at \( W_{\text{max}} \) and a first order filter with -3 dB gain at this frequency. The result will be a third order filter with 6 dB peak at \( W_{\text{max}} \). The inputs of the program are the speed of the vehicle in m/sec, Sampling frequency in Hz, and the radio propagation frequency \( f_p \) in Hz.

```matlab
function [A,B,C,D] = fadefilt(speed,Samp_freq,f_p)
  fm=v/(3e8/fp); % Maximum Doppler Frequency
  rf=1.03;
  Wmax=2*pi*(fm/rf);
  Q=2.7823834; % Q is chosen to yield a 9 dB peak
  Wo=Wmax/(sqrt(1-(2*Q^2)));
  % poles of the second order in s domain
  s1=Wo/(2*Q)+Wo*(sqrt(1-(4*Q^2)))*i;
  s2=Wo/(2*Q)-Wo*(sqrt(1-(4*Q^2)))*i;
  % Second order filter coefficients
  A2=2*exp(-real(s1)/Samp_freq)*cos(abs(imag(s1)/Samp_freq));
  B2=-exp(-real(s1)/Samp_freq)^2;
  D2=1-A2-B2;
  % First order filter coefficients
  A1=exp(-Wmax/Samp_freq);
  D1=1-A1;
  % Third order filter coefficients
  A=A2+A1;
  B=B2-A2*A1;
  C=-B2*A1;
  D=D1*D2;
  % Some typical values for Samp_freq=75000 Hz and fp=9e8 Hz:
  % speed=50 km/h: A=2.996 B=-2.992 C=0.996 D=2.6261e-08
  % speed=100 km/h: A=2.992 B=-2.9841 C=0.9921 D=2.0967e-07
```

114
Appendix B

The Jover-Kailath algorithm for measurement update equations

The following square-root algorithm, described with MATLAB, is for the Measurement Update Equations based on the method of [54]. This recursive algorithm is described in 5.2 and the inputs and outputs are defined in Fig. 5-1. The parameter "size" is equal to 3q.

```matlab
function [Xk, Lp, Dp, Sigm] = Measurement_Update(Zk, Hk, Xk, L, D, No, size)

N=size+1;
Beta(1)=0; Delta(N)=No;
b=Hk*L;
for i=N-1:-1:1,
    Delta(i)=Delta(i+1)+b(i)*b(i)'*D(i);
    Beta(i+1)=b(i)'*(D(i)/Delta(i));
    Dp(i)=Delta(i+1)*(D(i)/Delta(i));
    a(i)=Beta(i+1);
end
% adding one element to the vector [b],
bn=[Hk*Xk-Zk, b];

for i=1:N-1;
    Lin(i,1)=Xk(i);
    Lout(i,1)=0;
    for j=1:N-1;
        Lin(i,j+1)=L(i,j);
        Lout(i,j+1)=0;
    end
    Lout(i,i+1)=1;
end

for i=1:N-1,
    for j=0:N-1-i,
        Lout(i+j,j+1)=Lin(i+j,j+1)-bn(j+1)*a(j+i);
        a(j+i)=a(j+i)+Beta(j+1)*Lout(i+j,j+1);
    end
end

for i=1:N-1,
    Xk(i)=Lout(i,1);
    for j=1:N-1,
        Lp(i,j)=Lout(i,j+1);
    end
end
```

115
Appendix C

The simplified algorithm for the RLS estimator

This algorithm implements the RLS estimator as explained in chapter 5. The matrices are reduced in size as compared to the Kalman filter and L is essentially a scalar. The state vector $X$, the covariance factors $L$ and $D$, and the Lambda factor will be set to initial values before calling the function for the first time. Then the output values can be used as the input for the next iteration along with the new received signal $Z$ and the new hypothesis vector $H$.

```matlab
function [Xp,Lp,Dp]=RLS(Z,H,X,L,D,Lambda)
    A0=H*X-Z;
    b(1)=H(1)+H(2)*L;
    b(2)=H(2);
    Delta(3)=Lambda;
    for i=2:-1:1,
        Delta(i)=Delta(i+1)+(b(i)*b(i)'*D(i));
        M=D(i)/Delta(i);
        a(i)=b(i)'+M;
        Dp(i)=Delta(i+1)*M/sqrt(Lambda);
    end
    Xp(1)=X(1)-A0*a(1);
    Xp(2)=X(2)-A0*a(2);
    Lp=L-b(1)*a(2);
```
Appendix D

The algorithm for time update equations based on WGS method

The following MATLAB algorithm is for the temporal update using the WGS method

[11]. Inputs of the program are the $n \times n$ matrix product $FL = F * L_p, D_p$, the $n \times p$ matrix $G$, and the $p \times p$ matrix $D_q$. In the output $FL$ and $G$ are overwritten by intermediate results, and $D$ and $L$ are factors of the updated covariance.

```matlab
function [L,D] = Grm_Schmt(FL,Dp,G,Dq,n,p)
    for i=1:n,
        sigm=0;
        for j=1:n,
            sigm=sigm+FL(i,j)*FL(i,j)'*Dp(j);
        end
        for j=1:p,
            sigm=sigm+G(i,j)*G(i,j)'*Dq(j);
        end
        D(i)=sigm;
        L(i,i)=1;
        for j=i+1:n,
            sigm=0;
            for k=1:n,
                sigm=sigm+FL(i,k)'*FL(j,k)*Dp(k);
            end
            for k=1:p,
                sigm=sigm+G(i,k)'*G(j,k)*Dq(k);
            end
            L(j,i)=sigm/D(i);
            for k=1:n,
                FL(j,k)=FL(j,k)-L(j,i)*FL(i,k);
            end
            for k=1:p,
                G(j,k)=G(j,k)-L(j,i)*G(i,k);
            end
        end
    end
end
```
Appendix E

The algorithm for time update equations based on the LDC algorithm

The following MATLAB algorithm is for the temporal update using the LDC algorithm. This algorithm consists of two functions. The main function receives the \( n \times n \) matrix product \( F_{L} = \overline{F}L_{p} D_{p} \), and the LD factors of \( GQG^T \) as \( GQGL \) and \( GQGD \). Then the correction algorithm will be applied \( n \) times as in (5-30), by calling the \( ldltp \) function. This function updates the Cholesky factorization of \( A \) to the Cholesky factorization of \( A + v^*v' \), i.e. If \( A = L*D*L^T \) then \( A + v^*v' = newL*newD*newL^T \). It is assumed that \( A \) is symmetric and positive definite. The details of this algorithm are given in [79].

function \( [L,D] = \text{Correction}(FL, Dp, GQGL, GQGD) \)

\[
\% GQGL and GQGD are the initial L-D factors of GQG^T
\]

\[
L = GQGL; \quad D = GQGD;
\]

\[
n = \text{size}(L, 1);
\]

\[
\text{for } i = 1:n,
\]

\[
v = FL(1:n, i);
\]

\[
s = \text{sqrt}(Dp(i, i));
\]

\[
[nL, nD] = ldltp(L, D, s*v);
\]

\[
L = nL; \quad D = nD;
\]

\[
\end
\]

function \( [\text{newL}, \text{newD}] = ldltp(L, D, v) \)

\[
n = \text{size}(L, 1);
\]

\[
\text{newL} = L; \quad \text{newD} = D;
\]

\[
\text{oldt} = 1;
\]

\[
\text{for } j = 1:n,
\]

\[
p = v(j);
\]

\[
t = \text{oldt} + p^2 / D(j, j);
\]

\[
\text{newD}(j, j) = D(j, j) * t / \text{oldt};
\]

\[
\beta = p / (D(j, j) * t);
\]

\[
\text{if } (j < n),
\]

\[
v(j+1:n) = v(j+1:n) - p*L(j+1:n, j);
\]

\[
\text{newL}(j+1:n, j) = L(j+1:n, j) + \beta * v(j+1:n);
\]

\[
\end;
\]

\[
\text{oldt} = t;
\]

\[
\end; \]

118
References


[59] M. J. Omidi, P. G. Gulak, and S. Pasupathy, “Parallel structures for joint channel estimation and data detection over fading channels,” Accepted for Publication in the *Journal of Selected Areas in Communication*.


