

DESIGN OF SCALABLE AND LOW COMPLEXITY DECODERS FOR WIRELESS SENSOR NETWORKS

by

Ruchira Yasaratna

A thesis submitted to The Faculty of Graduate Studies
in partial fulfillment of the requirements for the degree of
Master of Science

Department of Electrical and Computer Engineering
The University of Manitoba
Winnipeg, Canada

January 2008

Copyright © 2008 by Ruchira Yasaratna

THE UNIVERSITY OF MANITOBA
FACULTY OF GRADUATE STUDIES

COPYRIGHT PERMISSION

**DESIGN OF SCALABLE AND LOW COMPLEXITY DECODERS FOR WIRELESS SENSOR
NETWORKS**

BY

Ruchira Yasaratna

**A Thesis/Practicum submitted to the Faculty of Graduate Studies of The University of
Manitoba in partial fulfillment of the requirement of the degree**

Master of Science

Ruchira Yasaratna © 2007

**Permission has been granted to the University of Manitoba Libraries to lend a copy of this
thesis/practicum, to Library and Archives Canada (LAC) to lend a copy of this thesis/practicum,
and to LAC's agent (UMI/ProQuest) to microfilm, sell copies and to publish an abstract of this
thesis/practicum.**

**This reproduction or copy of this thesis has been made available by authority of the copyright
owner solely for the purpose of private study and research, and may only be reproduced and copied
as permitted by copyright laws or with express written authorization from the copyright owner.**

To my parents.

Abstract

Recent technological advances in sensors, embedded processors, and wireless devices, often all integrated on a single chip, are leading to the use of wireless sensor networks (WSN) for interacting with the physical world in wide range of applications such as security and surveillance, monitoring of natural habitats and eco-systems, medical monitoring etc. This thesis investigates the practical design of a joint decoder for a large scale wireless sensor network having a limited transmission bandwidth.

Joint decoders are an effective means of decoding correlated signals gathered by a sensor network. However, the optimal joint decoder designed for a large sensor network suffers from its high computational complexity. We consider *minimum mean square error* (MMSE) decoding in a dense sensor network where distributed quantization is used to improve the performance. As a solution to the problem of exponential complexity of the optimal decoder, we present a framework based on *Bayesian networks* for designing a scalable, but near-optimal decoder. In our approach, a complexity-constrained factor graph is obtained by an algorithm which constructs an equivalent Bayesian network based on a training set of sensor observations, using the *maximum likelihood* (ML) criterion. Our simulation results show that, the scalable decoders constructed using the proposed approach perform close to optimal with both Gaussian and non-Gaussian sensor data. Moreover, the complexity of the decoder grows only linearly with the size of the network.

Acknowledgements

This thesis is the result of two years of work, in which I have been accompanied and supported by many people. It is a pleasant aspect that I now have the opportunity to express my gratitude to all of them.

I would like to express my deep gratitude to my advisor, Professor P. Yahampath for his invaluable support and guidance in every respect during the course of this work. It has been a great privilege working with him during the past years.

I am also grateful to the Department of Electrical and Computer Engineering at the University of Manitoba for providing me with an excellent work environment. I would like to thank all of my friends who kept me sane, and made living in Winnipeg enjoyable.

I cannot finish without saying how grateful I am to my family. I am forever indebted to my parents for their love and encouragement when it was most required. I am grateful to my brother and sister, for rendering me the sense and the value of brotherhood. My biggest “thanks” goes to my wife Niludi, for her never ending love, great understanding, and patience throughout all these years.

Ruchira Yasaratna

January 2008

Table of Contents

List of Figures	vii
List of Tables	x
List of Acronyms	xi
1 Introduction	1
1.1 Distributed Source Coding	3
1.2 System Model for a Sensor Network	7
1.3 Minimum Mean Square Error (MMSE) Decoder	10
1.4 Complexity of MMSE Decoder	12
1.5 Contributions of this Thesis	13
1.6 Thesis Organization	14
2 Low Complexity Decoding for Sensor Networks	15
2.1 Decoding without “a priori” Information	15
2.2 Scalar MMSE Decoder	16
2.3 Representing a Factorized PDF using Factor Graphs	17
2.4 Scalable Decoding using Factor Graphs	19
2.4.1 Source Clustering	21
2.4.2 Source Factorization	22
2.5 Complexity of the Various Decoders	23
2.5.1 Complexity of using the KLD	25
2.6 Simulation Results	25
3 Maximum Likelihood based Factorization	32
3.1 PDF Approximation	33
3.2 Probabilistic Graphical Models	35
3.3 Bayesian Networks	35
3.4 The likelihood of a Bayesian Network Structure	39
3.5 An algorithm to search for ML Bayesian Network	42
3.6 Simulation Results	43

TABLE OF CONTENTS

vi

4	Factorization of a PMF based on Sensor Clustering	54
4.1	An approach to Sensor Clustering based on Statistical Dependence .	56
4.1.1	Divisive Clustering	58
4.1.2	Agglomeration Clustering	59
4.2	Factor Graph Construction using Bayes Networks	62
4.3	Optimization of Binning Functions	66
4.4	Simulation Results	68
5	Conclusions and Suggestions for Future work	76
5.1	Summary and Conclusions	76
5.2	Future Research Directions	77

Appendices

A	Conditional Mean Estimator	79
B	Derivation of (1.6)	81
C	The Sum-product Algorithm	83
D	Proof - Likelihood of a Bayesian Network	88
	Bibliography	92

List of Figures

1.1	A direct sensor network model.	3
1.2	Separate encoding of correlated sources - Slepian-Wolf problem. . . .	4
1.3	The Slepian-Wolf region.	5
1.4	Block diagram of the sensor network model.	8
2.1	Factor graph representation of (2.7).	18
2.2	Factor graph for the function $P(\mathbf{i}) \cdot \prod_{n=1}^5 P(\hat{i}_n i_n)$ which corresponds to a sensor network with 5 sensors. $P(\mathbf{i}) \approx \prod_{n=1}^5 P(I_n)$ results degree-1 function nodes and degree-2 variable nodes.	19
2.3	Decomposition of the decoding in to three stages.	24
2.4	The performance of the decoder designed without a priori information. Rate of the quantizers $Q=1, 2, 3, 4$ bits/sample and the sensor network consists of 3 sensors.	26
2.5	The performance of the scalar MMSE decoder. Rate of the quantizers $Q=1, 2, 3, 4$ bits/sample and the sensor network consists of 3 sensors.	27
2.6	The performance of the MMSE optimal decoder. Rate of the quantizers $Q=1, 2, 3, 4$ bits/sample and the sensor network consists of 3 sensors.	28
2.7	Performance comparison of the three decoders at rate $Q=4$ bits/sample. The sensor network consists of 3 sensors.	30
2.8	Factor graph corresponds to the expansion (2.22). The design parameters are chosen as $G = 2$ and $H = 1$	30
2.9	Performance of several decoders for a Gaussian source, as a function of CSNR. The network consists of 9 sensor nodes and each node uses quantizers of rate 1-bit/sample.	31
3.1	Bayesian network representation of (a) Eq. (3.3) and (b) Eq. (3.4). . . .	36
3.2	Bayesian network representation of (a) Eq. (3.5) and (b) Eq. (3.6). . . .	37
3.3	(a) Bayesian network representation of global pdf, $p(X_1, X_2, X_3, X_4) = p(X_2)p(X_1 X_2)p(X_4 X_1, X_2)p(X_3 X_4)$. (b) The Bayesian network shown as a factor graph.	38
3.4	(a) BN Λ corresponds to factorization $P(I_2)P(I_1 I_2)P(I_3 I_1)$. (b) BN Λ' corresponds to factorization $P(I_1)P(I_2 I_1)P(I_3 I_2)$	41

3.5	Factor graph belongs to non-optimal BN with maximum number of parents, $S = 2$. Corresponding factorization is as (3.14) and the likelihood of the structure is -1.5052×10^5	48
3.6	Factor graph corresponds to optimal BN found using Algorithm 1. Maximum number of parents, $S = 2$. The resulting factorization is (3.13) and the likelihood of the structure is -1.3682×10^5	49
3.7	Factor tree corresponds to the optimal BN found using the mutual information as the measurement of closeness. Maximum number of parents, $S = 1$. The resulting factorization is (3.17) and the overall closeness of the structure is 2.9349 bits.	50
3.8	Performance of the decoders as a function of CSNR. The network has 9 sensors and each of them uses 1 bit/sample Lloyd-Max quantization.	51
3.9	Performance of the decoders as a function of CSNR. The network has 9 sensors and each of them uses 2 bits/sample Lloyd-Max quantization.	52
3.10	Plot of RSNR against number of clock cycles during the execution of the sum-product algorithm, for CSNR= 5 dB.	53
4.1	Clustering of 15 sensors, $\mathcal{Z}_{max} = 4$ and $\mathcal{Z}_{min} = 2$. Clusters, $\Gamma_{c_1} = \{I_2, I_4, I_9, I_{12}\}$, $\Gamma_{c_2} = \{I_5, I_{10}\}$, $\Gamma_{c_3} = \{I_3, I_7, I_8\}$, $\Gamma_{c_4} = \{I_{11}, I_{13}\}$, $\Gamma_{c_5} = \{I_{14}\}$, $\Gamma_{c_6} = \{I_1\}$, $\Gamma_{c_7} = \{I_{15}\}$, $\Gamma_{c_8} = \{I_6\}$	60
4.2	Final result after step 1 and step 2. $\mathcal{Z}_{max} = 4$ and $\mathcal{Z}_{min} = 2$. Clusters, $\Gamma_1 = \{I_2, I_4, I_9, I_{12}\}$, $\Gamma_2 = \{I_5, I_{10}\}$, $\Gamma_3 = \{I_3, I_7, I_8, I_{15}\}$, $\Gamma_4 = \{I_6, I_{11}, I_{13}\}$ and $\Gamma_5 = \{I_1, I_{14}\}$	61
4.3	(a) Factor graph corresponds to the clusters (b) Separate decoding based on the clusters.	61
4.4	(a) Factor graph corresponds to distribution $\tilde{P}(\mathbf{I}) = P(I_2, I_4, I_7)P(I_5, I_6, I_9)P(I_1, I_3, I_8)$ (b) Factor graph after linking clusters, $\tilde{P}(\mathbf{I}) = P(I_2, I_4, I_7)P(I_5, I_9 I_4, I_7)P(I_6 I_5, I_9)P(I_1, I_3 I_2, I_4)P(I_8 I_1, I_3)$	64
4.5	System model with binning	66
4.6	Simulation results - Source is distributed as multivariate Gaussian. Scalar quantization with a subsequent binning function at the encoder reduces channel rate to 1-bit/sample.	70
4.7	Performance of several decoders as a function of CSNR. The network has $N = 9$ sensing nodes and each node uses rate 1-bit/sample quantization, without subsequent binning.	72
4.8	Performance of various decoders when binning-based distributed quantizers are used in sensors. Each sensor uses a quantizer rate of Q bits/sample and a transmission rate of 1 bit/sample. The curves (a) and (b) are the same as those in Fig. 4.7. For the system with $Q = 2$, the performance of the optimal decoder is shown in curve (e).	73

4.9	Performance comparison of the decoders designed based on PMF approximations in (4.1) and (4.8) with $N = 100$ sensors. Curves (b) and (c) for channel rate 1-bit/sample, without subsequent binning.	75
B.1	Transmission of quantized data over a noisy channel.	81
C.1	Generated messages at each step of the sum-product algorithm.	84

List of Tables

2.1	Computational complexity of the decoders (a) using equations (2.16) and (2.17) (b) Using factor graphs/sum-product algorithm, N variable nodes with $d = 2$ and N function nodes with $d = 1$ (c) Using factor graphs/sum-product algorithm, N variable nodes with $d = 2$ and 1 function node with $d = N$	24
3.1	Training set-the outputs of the quantizers	41
3.2	Numerical values of the parameters	42
3.3	Likelihoods of different factorizations	46
4.1	Binning functions - Gaussian source, $Q = 3$ bits/sample, $U_n = h_n(I_n)$	71
4.2	Binning functions - Non-Gaussian source, $Q = 3$, $U_n = h_n(I_n)$	74
C.1	Complexity of the sum-product algorithm	86

List of Acronyms

AWGN	Additive White Gaussian Noise
BN	Bayesian Network
CCRE	Constrained Chain Rule Expansion
CME	Conditional Mean Estimator
CRE	Chain Rule Expansion
CSNR	Channel Signal to Noise Ratio
DAG	Directed Acyclic Graph
DSC	Distributed Source Coding
FG	Factor Graph
FT	Factor Tree
KLD	Kullback-Leibler Distance
ML	Maximum Likelihood
MMSE	Minimum Mean Square Error
MRF	Markov Random Fields
MSE	Mean Square Error
PDF	Probability Density Function
PGM	Probabilistic Graphical Model
PMF	Probability Mass Function
RSNR	Reconstruction Signal to Noise Ratio
SA	Simulated Annealing

Chapter 1

Introduction

In a wireless sensor network (WSN), sensor nodes are distributed over an area of interest to observe some physical phenomenon. The information gathered by sensors are sent to a central processing station wirelessly, for further processing. WSNs find applications in environmental monitoring [1], such as monitoring of air, soil and water, habitat monitoring [2] (e.g. determining the plant and animal species population and behavior), monitoring the safety of buildings [3], military surveillance, health care, and inventory tracking etc. For example, sensors which measure the climate variations could be used in different elevations of a tree to compute the average temperature at each elevation. On a very large scale, the system for the vigilance of the Amazon (SIVAM) [4] provides environmental monitoring, drug trafficking monitoring, and air traffic control for the Amazon basin.

Recent advancements in technology have made the deployment of small, inexpensive, low-power, distributed devices, which are capable of local processing and wireless communication, a reality. In contrast to classical wireless networks, such as mobile, data and voice communication systems, WSNs are designed for specific applications. They have much more severe constraints on power consumption and complexity.

Based on information dissemination, sensor networks can be modeled either as *multi-hop* or *direct*. In the multi-hop model, data gathered by one sensor transits through an intermediate sensor node before reaching the central processing station. In this thesis we are concerned the direct model, in which the nodes directly communicate with a base station. This model is depicted in Fig. 1.1.

The sensors perform four basic operations, namely, *source coding*, *channel coding*, *modulation*, and *transmission*. The source encoder (quantizer) generates discrete representations of continuous valued sensor observations, usually in binary format. The main objective of source coding is to minimize the number of symbols in the discrete representation of observations, by removing the redundancy. In general there are two approaches to source coding - *lossy coding* [5] and *lossless coding* [6]. In a lossy coding scheme, as the name implies, some of the original information is discarded when it is compressed. Therefore, it is impossible to reproduce an exact replica of the original signal at the receiver. On the other hand, lossless coding produce an exact replica of the original source. For example, the zip file format produced by the *Winzip*® program on a PC produces exact copies of the original material that was encoded within the zip file. *Huffman coding*, *arithmetic coding* and *Lempel-Ziv coding* are examples for the practical lossless source coding schemes. *JPEG*, *MP3* and *MPEG* are some of the widely used lossy coding standards.

The channel coding adds redundancy to the transmitted bit stream in a controlled manner to achieve robustness against the channel errors, which helps channel decoder to recover transmitted symbols reliably. The function of the modulator is to represent the discrete data using analog waveforms for transmission over the wireless network. At the receiver, each of these signals coming from the sensor nodes are demodulated, channel decoded, and directed to a joint decoder for reconstruction.

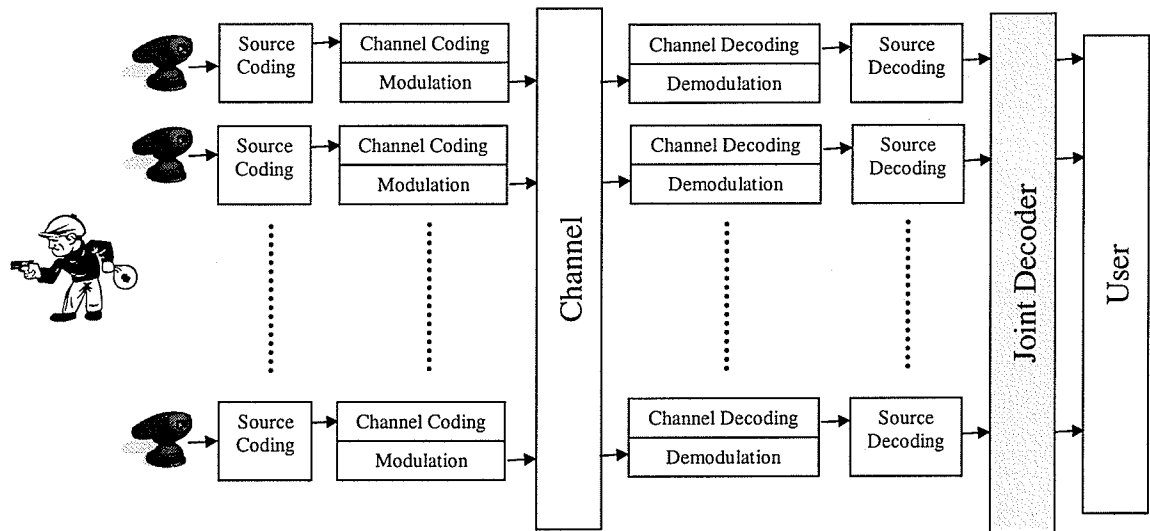


Figure 1.1: A direct sensor network model.

In this thesis, we are considering the problem of communicating the information gathered by sensor nodes to a remote receiver over a noisy channel. Due to low spatial resolution of the sensors, the measurements are highly correlated, resulting in a set of new and challenging communication problems. For example, how do we reduce the transmission rate and how do we minimize the transmission errors within this kind of a network? Since the data acquisition is distributed, conventional source coding methods such as vector quantization are not efficient. Several methods have been proposed to solve this specific problem, either with multi-hop networks based on inter-sensor communication or with distributed source coding [7], where no communication between the sensors is necessary.

1.1 Distributed Source Coding

The problem of reducing the transmission rate of distributed sources, also known as *distributed source coding* (DSC) problem was first addressed by Slepian and Wolf

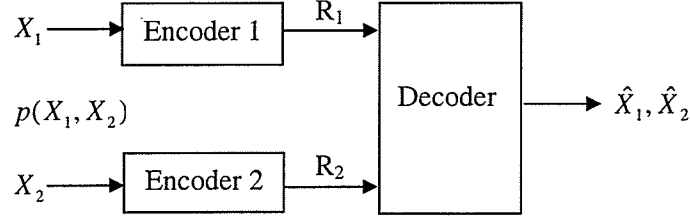


Figure 1.2: Separate encoding of correlated sources - Slepian-Wolf problem.

in 1973 [8]. The system studied in [8] is shown in Fig. 1.2. Consider two correlated signals, X_1 and X_2 with joint distribution $p(X_1, X_2)$, generated by two separate sources which can not communicate with each other. That is, the signals are encoded independently. The receiver, on the other hand, can see both encoded signals and performs the joint decoding of both signals. A sensor network, which consists of spatially separated sensor nodes sending correlated information to a central processing station, is an example of such system. In this case, it is not immediately clear which encoding rates guarantee a perfect reconstruction at the receiver. Not surprisingly, if we have a common encoder and a common decoder, the problem reduces to the classical point-to-point communication problem, for which Shannon's source coding theorem [9] applies. i.e., the messages can be perfectly reconstructed at the receiver if and only if $R_1 + R_2 > H(X_1, X_2)$, where R_1 and R_2 are the encoding rates and $H(X_1, X_2)$ is the *joint entropy* [10] of the sources.

If we encode X_1 at the rate $R_1 > H(X_1)$ and X_2 at rate $R_2 > H(X_2)$, then the source coding theorem guarantees lossless reconstruction. In fact the sum of the rates $R_1 + R_2 > H(X_1) + H(X_2)$, which is generally greater than the joint entropy $H(X_1, X_2)$. Slepian and Wolf in their landmark paper [8] make a surprising conclusion: for perfect reconstruction, the separated encoders should have the sum rate, $R_1 + R_2 \geq H(X_1, X_2)$. In other words, a combined rate of $H(X_1, X_2)$ is sufficient even if the correlated signals are encoded separately. But, it should be noted that the

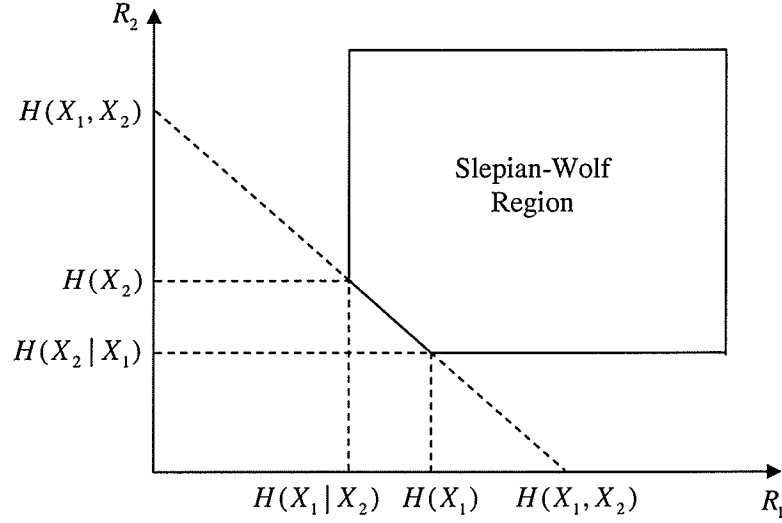


Figure 1.3: The Slepian-Wolf region.

joint decoder requires a minimum amount of rate from each encoder, specifically the average remaining uncertainty about the messages of one source given the messages of the other source. According to the Slepian-Wolf coding theorem, the achievable rate pair (R_1, R_2) for distributed sources X_1 and X_2 is given by,

$$R_1 \geq H(X_1|X_2), \quad (1.1)$$

$$R_2 \geq H(X_2|X_1), \quad (1.2)$$

$$R_1 + R_2 \geq H(X_1, X_2). \quad (1.3)$$

The three inequalities above define the so called Slepian-Wolf region, containing all the achievable rate pairs (R_1, R_2) , as shown in Fig. 1.3. Pradhan and Ramchandran in [11] demonstrate the intuition behind Slepian-Wolf coding by providing a solution based on channel codes. They developed a method called DISCUS (distributed source coding using syndromes) [11]. The *syndromes* associated with a channel code is defined as $s = Hx$, where H is the parity-check matrix of the code, and x is any valid

codeword. Possibility of using syndromes for a similar situation was first introduced by Wyner in [12], but no implementation algorithm was given. In [11], instead of sending the codeword corresponds to X_1 , index of the *coset* (the set of all patterns which produce the same syndrome s [11]) which includes the corresponding codeword is transmitted. The receiver decodes it by choosing the codeword in the given coset that is closest to X_2 . In [11] DISCUS was used with lattice and trellis encoding, to encode correlated Gaussian sources. But, it was not shown that how the techniques approach the information-theoretic bounds in Fig. 1.3. In [13] DISCUS has been extended to the case of symmetric encoding, where all sensors send partial information to the decoder. An alternative approach for the asymmetric scenario with side information at the decoder was provided by Servetto in [14], where a constructive approach for Gaussian sources based on linear codes and lattices was presented. Apart from these contributions, iterative channel coding techniques such as turbo codes [15] and low density parity check codes (LDPC) [7] have also been used for DSC. These methods have been shown to achieve the theoretical limits of [8].

So far, we have looked at conventional DSC problem, in which there is no constraint on the transmission bandwidth. i.e., near lossless DSC. In addition to this scenario, a problem of interest for sensor networks involves *sensor fusion* (combining the measurements from more than one sensor into a single quantity) under a given rate constraint, where the overall system is lossy. Motivated by the formative study of Flynn and Gray [16] for the case of two memoryless correlated sources, several authors have proposed encoding and decoding solutions for this problem. In [16], each encoder was implemented as a two stage encoder, in which the first stage consists of a quantizer optimized to the source probability density function. And the second stage uses the correlation of discrete outputs of the first stage to reduce the bit rate.

The process is referred to as *binning*. Other alternative approaches were proposed by Rebello-Monedero and Girod [17], where the main focus was on the optimization of the quantization stage, and Wang and Orchard [18], where the trellis codes were used to achieve a coding gain at the encoder as well as at the decoder.

Despite these contributions, a very little is known about DSC in dense sensor networks until recently. The main reason is that, the above mentioned methods become infeasible in a large-scale sensor network. One of the key problems is that the complexity of the joint decoder increases exponentially with the number of sensors. In order to address this problem, a *scalable* solution for distributed estimation was first proposed by Gubner in [19]. Later, Barros and Tüchler in [20] proposed a scalable solution for joint decoding in a large sensor network and further extended in [21]. The performance of their decoder for Gaussian sources is very close to the optimal decoder and the decoding complexity only grows linearly with the network size.

1.2 System Model for a Sensor Network

In a sensor network, the sensors observe some physical process, and subsequently send these observations over a wireless channel to a fixed or moving receiver. The challenge faced by the receiver is to jointly decode this data optimally to achieve reconstructions as close as possible to the original observations, while keeping the computational complexity manageable.

If the channel between sensors and the receiver introduces no errors, the joint decoder acts on the compressed versions of the sensor observations and the decoding is possible if perfect knowledge of the correlation between sensors is available. The previous work reported in the literature, such as the work in [16, 22] comes under this category. But, in this thesis we use a sensor model which includes the channel noise.

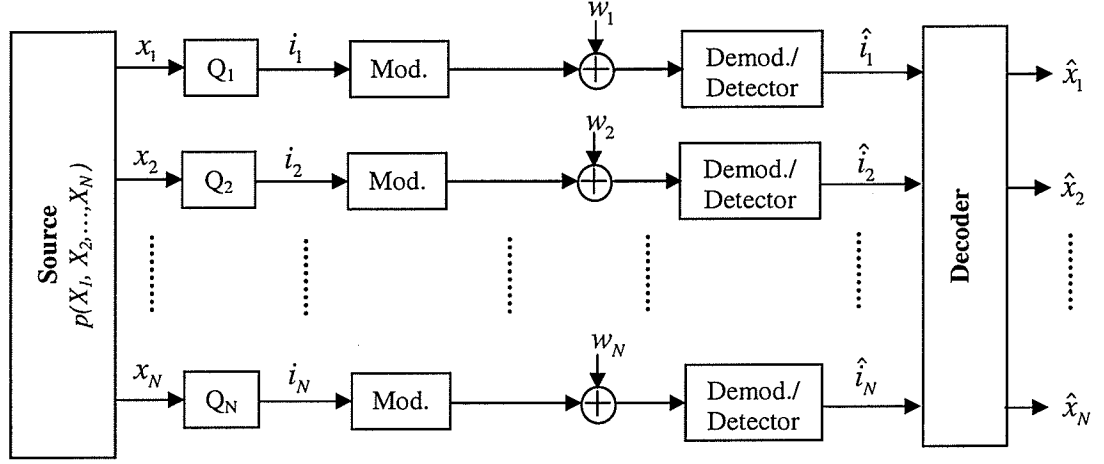


Figure 1.4: Block diagram of the sensor network model.

The effect of channel noise on the performance of the joint decoder will be discussed in detail. Later, we will design joint source channel codes based on the correlation of the compressed sensor outputs, where the DSC principles will be incorporated into our system.

A block diagram of the sensor network model considered in this thesis is shown in Fig. 1.4. The system consists of the following blocks.

- A **source model**, which models the data gathered by the sensor nodes. As the sensors are distributed in dense manner, we can assume the sensor measurements to be highly dependent. The outputs of the N sensor nodes can be represented by a joint probability distribution $p(X_1, X_2, \dots, X_N)$, where each random variable X_n corresponds to a data source which delivers real-valued samples $x_n(t)$, $n = 1, 2, \dots, N$. In this thesis we focus on the spatial dependance of the sensor measurements rather than their temporal dependance. Hence, the time index t is dropped. But, our results can be extended to include temporal correlation as well. For simplicity, we denote the vector of source sample from N sensors by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_N]^T$.

- **N identical sensor nodes** are expected to measure physical data from their individual territories. To cover the entire field of interest, a large number of sensors are distributed over an area. The sensors are assumed to be designed only to send data and not to receive. In other words, the inter-sensor communication is not possible. It is assumed that the i -th sensor quantizes its input using a Q -bit scalar quantizer [5] (denoted by Q_i in Fig. 1.4). The quantizer output for a given input X_n is represented by the index I_n , where $I_n \in \Omega = \{1, 2, \dots, 2^Q\}$. Each index $I_n = i_n$ corresponds to a unique reconstruction value $\tilde{x}_n(i_n)$. We denote the index vector by $\mathbf{I} = [I_1, I_2, \dots, I_N]^T$ and the corresponding realizations by $\mathbf{i} = (i_1, i_2, \dots, i_N)^T$. The scalar quantizers are designed to minimize the mean square error (MSE) of the reconstruction, $E\{(X_n - \tilde{X}(I_n))^2\}$. The quantizer output is transmitted over the channel using an appropriate modulation method such as *binary phase shift keying* (BPSK) or *quadrature phase shift keying* (QPSK) [23].
- The **wireless channel** is composed of N parallel and independent channels, with a sensor node at each input and the receiver at the output. We assume the channel is discrete memoryless channel which converts I_n to $\hat{I}_n$. The effect of multi-path fading is not considered. After demodulation and detection [24], the decoder receives the index $\hat{I}_n$, which may be different to the transmitted index I_n . We denote the received index vector by $\hat{\mathbf{I}} = [\hat{I}_1 \ \hat{I}_2 \ \dots \ \hat{I}_N]^T$.
- The **decoder** processes the index vector $\hat{\mathbf{I}}$ and produces the estimate $\hat{\mathbf{X}} = [\hat{X}_1 \ \hat{X}_2 \ \dots \ \hat{X}_N]^T$ of the sensor measurements $\mathbf{X}$. In the rest of this thesis, we will consider the design of the decoder which minimizes the MSE criterion $E\{\|\mathbf{X} - \hat{\mathbf{X}}\|^2\}$.

1.3 Minimum Mean Square Error (MMSE) Decoder

The decoder which minimizes the MSE fidelity criterion is the conditional mean estimator (CME) (see proof in Appendix A) and is given by,

$$\hat{\mathbf{X}}^*(\hat{\mathbf{I}}) = E\{\mathbf{X}|\hat{\mathbf{I}}\}. \quad (1.4)$$

Then, for a received index vector $\hat{\mathbf{i}} = (\hat{i}_1, \hat{i}_2, \dots, \hat{i}_N)^T$, the optimal MMSE decoder's output for the n -th sensor observation is,

$$\begin{aligned} \hat{x}_n^*(\hat{\mathbf{I}} = \hat{\mathbf{i}}) &= E\{x_n|\hat{\mathbf{I}} = \hat{\mathbf{i}}\} \\ &= \int_{-\infty}^{\infty} x_n \cdot p(x_n|\hat{I}_1 = \hat{i}_1, \hat{I}_2 = \hat{i}_2, \dots, \hat{I}_N = \hat{i}_N) dx_n \\ &= \int_{-\infty}^{\infty} x_n \cdot \frac{p(x_n, \hat{I}_1 = \hat{i}_1, \hat{I}_2 = \hat{i}_2, \dots, \hat{I}_N = \hat{i}_N)}{P(\hat{I}_1 = \hat{i}_1, \hat{I}_2 = \hat{i}_2, \dots, \hat{I}_N = \hat{i}_N)} dx_n. \end{aligned} \quad (1.5)$$

Note that, to obtain the optimal reconstruction $\hat{x}_n^*$, the mixed probability distribution $p(\cdot)$ should be known beforehand. In practice, the joint distributions $p(\cdot)$ s are not available, and as can be seen, estimating them can also be difficult. Even if they are known, the right hand side of (1.5) will have to be pre-computed and stored for all the different values of $\hat{\mathbf{i}}$. For a N -sensor system with Q -bit quantizers, there are 2^{QN} different values to be computed and stored for each sensor n .

If the quantizers are PDF-optimized, we can prove that (see Appendix B),

$$E\{(x_n - \hat{x}_n)^2\} = E\{(x_n - \tilde{x}_n(i_n))^2\} + E\{(\hat{x}_n - \tilde{x}_n(i_n))^2\}. \quad (1.6)$$

Then, the estimator which minimizes $E\{(\tilde{x}_n(i_n) - \hat{x}_n)^2\}$ also minimizes $E\{(x_n - \hat{x}_n)^2\}$, where $\tilde{x}_n(i_n)$ is the source representation of the n -th sensor corresponding to the

transmitted index i_n . In this case, instead of using (1.4) we can use,

$$\begin{aligned}\hat{\mathbf{X}}^*(\hat{\mathbf{I}}) &= E\{\tilde{\mathbf{X}}(\mathbf{I})|\hat{\mathbf{I}}\} \\ &= \sum_{\mathbf{i} \in \Omega^N} \tilde{\mathbf{x}}(\mathbf{I} = \mathbf{i}) P(\mathbf{I} = \mathbf{i}|\hat{\mathbf{I}})\end{aligned}\quad (1.7)$$

The only unknown quantities in (1.7) are the *a posteriori probabilities* $P(\mathbf{I} = \mathbf{i}|\hat{\mathbf{I}})$. By applying the Bayes rule

$$\begin{aligned}P(\mathbf{I} = \mathbf{i}|\hat{\mathbf{I}}) &= \frac{P(\hat{\mathbf{I}}|\mathbf{I} = \mathbf{i})P(\mathbf{I} = \mathbf{i})}{P(\hat{\mathbf{I}})} \\ &= \gamma \cdot \underbrace{P(\hat{\mathbf{I}}|\mathbf{I} = \mathbf{i})}_{\text{from channel}} \cdot \underbrace{P(\mathbf{I} = \mathbf{i})}_{\text{a priori information}},\end{aligned}\quad (1.8)$$

where $\gamma = \frac{1}{P(\hat{\mathbf{I}})}$ is a constant normalizing the sum over the product of probabilities to one. Generally, it is not necessary to calculate γ explicitly. $P(\mathbf{I} = \mathbf{i})$ is the probability distribution of the index vector $\mathbf{I}$, which is a priori information for the decoder. Then using the results of (1.8), we can write (1.7) as

$$\hat{\mathbf{x}}^*(\hat{\mathbf{i}}) = \gamma \cdot \sum_{\mathbf{i} \in \Omega^N} \tilde{\mathbf{x}}(\mathbf{I} = \mathbf{i}) P(\hat{\mathbf{I}}|\mathbf{I} = \mathbf{i}) P(\mathbf{I} = \mathbf{i}). \quad (1.9)$$

To simplify, we will consider a scalar version of (1.9), which we call the optimal scalar estimate $\hat{x}_n^*(\hat{\mathbf{i}})$. Now, $\hat{x}_n^*(\hat{\mathbf{i}})$ can be calculated as

$$\begin{aligned}\hat{x}_n^*(\hat{\mathbf{i}}) &= E\{\tilde{x}(I_n)|\hat{\mathbf{I}} = \hat{\mathbf{i}}\} \\ &= \gamma \cdot \sum_{i \in \Omega} \tilde{x}_n(I_n = i) \cdot \sum_{\mathbf{i} \in \Omega^N; i_n = i} P(\hat{\mathbf{I}}|\mathbf{I} = \mathbf{i}) P(\mathbf{I} = \mathbf{i}).\end{aligned}\quad (1.10)$$

We assumed that N parallel AWGN channels are independent. Therefore, the channel

probability distribution $P(\hat{\mathbf{I}}|\mathbf{I} = \mathbf{i})$ can be factorized into

$$P(\hat{\mathbf{I}}|\mathbf{I} = \mathbf{i}) = \prod_{l=1}^N P(\hat{i}_l|i_l). \quad (1.11)$$

Substituting this result in (1.10)

$$\hat{x}_n^*(\hat{\mathbf{i}}) = \gamma \cdot \sum_{\mathbf{i} \in \Omega} \tilde{x}(I_n = i) \cdot \sum_{\mathbf{i} \in \Omega^N; i_n = i} P(\mathbf{I} = \mathbf{i}) \prod_{l=1}^N P(\hat{i}_l|i_l). \quad (1.12)$$

The probability mass function (PMF) $P(\mathbf{i})$ in (1.12) can be obtained either numerically by integrating the source PDF $p(\mathbf{x})$ over the Lloyd-Max quantization regions [5], or by estimating it using Monte-Carlo simulations. We use the latter approach in the simulations.

1.4 Complexity of MMSE Decoder

We measure the computational complexity of the decoding process by the number of additions and multiplications necessary to compute the estimates $\hat{x}_n^*$, $n = 1, 2, \dots, N$. It is especially important to know how the complexity depends on the number of sensors N , since we want to use the decoder for a sensor network which contains many sensors, may be in the order of hundreds or thousands.

Let's factorize the joint probability $P(\mathbf{i}, \hat{\mathbf{i}})$ as

$$P(\mathbf{i}, \hat{\mathbf{i}}) = P(\mathbf{i})P(\hat{\mathbf{i}}|\mathbf{i}) = P(\mathbf{i}) \prod_{l=1}^N P(\hat{i}_l|i_l). \quad (1.13)$$

Comparing (1.12) and (1.13), it is clear that $\sum_{\mathbf{i} \in \Omega^N; i_n = i} P(\mathbf{I} = \mathbf{i}) \prod_{n=1}^N P(\hat{i}_n|i_n)$ in (1.12) corresponds to the marginalization of indices I_n of $P(\mathbf{i}, \hat{\mathbf{i}})$. The most compu-

tationally intensive operation in the decoding process is this marginalization. It can be proven that, for each sensor, one requires $2^{Q(N-1)} - 1$ additions and $N2^{QN}$ multiplications for the whole decoding process. Therefore, the computational complexity of the decoding process grows *exponentially* with N and the MMSE decoding is infeasible for networks with a large number of sensors. When the source distribution is Gaussian, a scalable and near optimal solution to this problem was given in [20] based on *factor graphs* and the *sum-product algorithm* [25].

1.5 Contributions of this Thesis

In contrast to previous work, this thesis presents a more general approach to factor graph construction based on representing a factorized multi-variate PMF using a graphical model referred to as a *Bayesian network* (BN). The optimal factorization is then obtained by maximum likelihood (ML) estimation of the underlying BN structure using a training set of sensor observations. Thus, the proposed method is applicable to a sensor network with any source distribution, not necessarily Gaussian. Furthermore, in order to preserve the universality of our approach, the sensor clustering for designing binning-based encoders is carried out using an algorithm which uses the *Pearsons Chi-square* statistic. We have used the MSE as the distortion measure due to its mathematical tractability and its effectiveness in many relevant applications. The algorithms proposed in this thesis are not necessarily optimal. However, comparison with many known results are provided to demonstrate their suitability for a large scale sensor network.

1.6 Thesis Organization

This thesis organized as follows. Chapter 2 reviews previously reported low complexity decoding methods for a large sensor network. The main contributions of this thesis are presented in Chapters 3 and 4. In Chapter 3, we will introduce Bayesian network-based approach to constructing a low complexity decoder. Chapter 4 describes sensor clustering and an algorithm for approximating a high dimensional PMF based on the sensor clusters. Finally, Chapter 5 presents conclusions of this thesis, and gives some directions for future research.

Chapter 2

Low Complexity Decoding for Sensor Networks

In this chapter, we will review some low complexity decoding schemes and compare their performance with the MMSE optimal decoder. In particular, we will consider the decoder which ignores *a priori* information (i.e., the source distribution) to produce output and the *scalar MMSE decoder*, in which the decoder ignores the correlation between the sensors. Sec. 2.4 discusses a low complexity sub-optimal decoder based on factor graphs and the sum-product algorithm proposed in [25], followed by an experimental evaluation of these schemes in Sec. 2.6.

2.1 Decoding without “a priori” Information

We may not always have full knowledge of the source distribution $p(\mathbf{X})$. In such scenarios, we wish to use a decoder which does make no assumptions about the source PDF. Recall that in (1.8) we have two factors, the channel transition probability $P(\hat{i}_n|i_n)$, and the *a priori* information, $P(\mathbf{i})$. The former solely depends on the channel

and can be estimated by transmitting a training sequence over the channel. On the other hand, the latter may either be known or can be estimated from a training set. But, if we assume that we do not know anything about the PMF $P(\mathbf{I})$, then what is the most fair substitution for this term? We appeal to the *maximum entropy principle*, first introduced by Jaynes [26] to find an answer. It states that,

“if there is not complete information about a distribution, the optimum estimate is as unbiased as possible, and so choose the most random possible distribution.”

Then, for our problem it is intuitive to use the *uniform distribution* as the priori distribution, which gives the maximum entropy for $P(\mathbf{i})$ among all possible PMFs. With this, we can write

$$P(\mathbf{I} = \mathbf{i}) = \frac{1}{|\Omega|^N}, \quad (2.1)$$

where $|\Omega| = 2^Q$ is the number of quantization levels. Substituting (2.1) in to (1.12)

$$\begin{aligned} \hat{x}_n(\hat{\mathbf{I}}) &= \gamma \sum_{\forall i \in \Omega} \tilde{x}_n(i) \cdot \frac{1}{|\Omega|} \prod_{k=1}^N P(\hat{i}_k | i_k) \\ &= \gamma \sum_{\forall i \in \Omega} \tilde{x}_n(i) \cdot \frac{1}{2^Q} \prod_{k=1}^N P(\hat{i}_k | i_k), \end{aligned} \quad (2.2)$$

Therefore, under the assumption of uniform prior distribution, we can perform the decoding.

2.2 Scalar MMSE Decoder

By ignoring the correlation between the sensors, we can derive the scalar MMSE decoder. In this case, the decoder uses only the redundancy within the output I_n of the each scalar quantizer to reduce the distortion due to the channel noise, w .

In contrast to (1.7), the problem now involves the scalar CME, which is given by

$$\begin{aligned}\hat{x}_n(\hat{i}_n) &= E\{\tilde{x}(I_n)|\hat{I}_n = \hat{i}_n\} \\ &= \sum_{i \in \Omega} \tilde{x}(I_n = i)P(I_n = i|\hat{i}_n).\end{aligned}\tag{2.3}$$

Observe that, the probabilities are conditioned on the channel output $\hat{i}_n$ of one sensor instead of $\hat{\mathbf{i}}$, as in the MMSE optimal decoder. This is where we drop the information about the correlation between the sensors and assume that they are independent. Now, by applying the Bayes rule for posterior probabilities

$$P(I_n = i|\hat{i}_n) = \frac{1}{P(\hat{i}_n)}P(\hat{i}_n|I_n = i)P(I_n = i).\tag{2.4}$$

The scalar MMSE decoder can be considered as an optimal decoder with mismatched a priori PMF $P(\mathbf{I})$

$$P(\mathbf{I} = \mathbf{i}) \approx \prod_{n=1}^N P(I_n = i_n),\tag{2.5}$$

where the random variables I_n are assumed statistically independent. The PMF of the output of each quantizer can be found using training data or by integrating the marginal PDF $p(X_n)$ over each Lloyd-Max partition interval. With this approximation, $\hat{x}(\hat{\mathbf{i}})$ can be found by following (2.3) and (2.4).

2.3 Representing a Factorized PDF using Factor Graphs

Suppose that, function $f(X_1, X_2, \dots, X_n)$ can be expressed as a product of several local functions, referred to as *factors*, as follows:

$$f(X_1, X_2, \dots, X_n) = \prod_{k=1}^K f_k(\mathbf{X}_k),\tag{2.6}$$

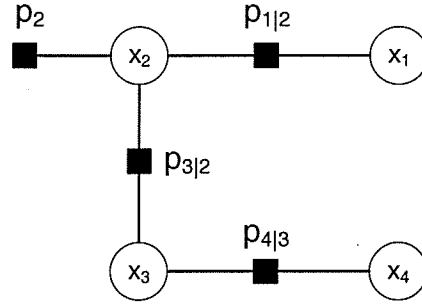


Figure 2.1: Factor graph representation of (2.7).

where $\mathbf{X}_k \subseteq \{X_1, X_2, \dots, X_n\}$. A factor graph [25] can be used to express the structure of the factorization (2.6). A factor graph is a bipartite graph which contains *variable nodes* for each variable X_l , and *function nodes* to represent local functions $f_k(\cdot)$. An edge in the factor graph connects the variable node X_l to the function node f_k if and only if X_l is an argument of the function f_k . Therefore, in general, a factor graph represents “*is an argument of*” relationship between variables and local functions. To illustrate this, consider the following PDF factorization:

$$p(X_1, X_2, X_3, X_4) = p_2(X_2)p_{1|2}(X_1|X_2)p_{3|2}(X_3|X_2)p_{4|3}(X_4|X_3). \quad (2.7)$$

This factorization is represented by the factor graph in Fig. 2.1. The function node $p_{1|2}$ is connected to variable nodes X_1 and X_2 . Therefore, the factor $p_{1|2}$ has the variables X_1 and X_2 as arguments. We define the *degree- d* of a function node as the number of variable nodes connected to it. Then, $p_{1|2}$ is a degree-2 function node. Similarly, we can define the degree of a variable node as well.

The importance of the factor graphs representation is that we can calculate the marginal functions very efficiently, if the factorized PDF is represented as a factor graph. An algorithm called *the sum-product algorithm* [25] allows us to compute the marginal functions of all variables in a PDF, which is represented in the factor graph

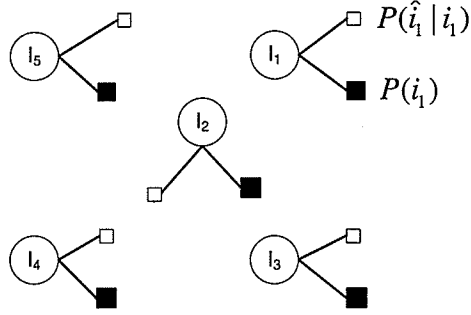


Figure 2.2: Factor graph for the function $P(\mathbf{i}) \cdot \prod_{n=1}^5 P(\hat{i}_n | i_n)$ which corresponds to a sensor network with 5 sensors. $P(\mathbf{i}) \approx \prod_{n=1}^5 P(I_n)$ results degree-1 function nodes and degree-2 variable nodes.

form. Appendix C describes the implementation of the sum-product algorithm based on factor graphs.

The factorization (2.5) has factors of one variable. Therefore, the factor graph which corresponds to the scalar MMSE decoder should have function nodes of degree-1. Fig. 2.2 shows the factor graph corresponding to a scalar MMSE decoder for a sensor network with 5 sensors.

2.4 Scalable Decoding using Factor Graphs

In Chapter 1, we derived the MMSE optimal decoder. It was noted that it is not feasible in a large sensor network because of its computational complexity. Sec. 2.1 and Sec. 2.2 discussed the decoder without priori information and the scalar MMSE decoder. As per simulation results in Sec. 2.6 their performance is considerably lower than that of the optimal decoder. One approach to scalable and near optimal decoding was proposed in [20] whose complexity grows only linearly with the number of sensors. In this section, we will summarize these results. As discussed previously, this approach is applicable only to Gaussian distributed sources.

The source PDF $p(\mathbf{X})$ can be approximated by assuming that it can be factorized as $\tilde{p}(\mathbf{X}) = \prod_{k=1}^K g_k(\mathbf{X}_{\mathcal{S}_k})$, where $\mathcal{S}_k \subseteq \mathcal{M} = \{1, 2, \dots, N\}$, $k = 1, 2, \dots, K$, are subsets of source indices, such that $\bigcup_{k=1}^K \mathcal{S}_k = \mathcal{M}$. The resulting $\tilde{p}(\mathbf{X})$ is an approximation to $p(\mathbf{X})$, since, in general $\tilde{p}(\mathbf{X}) \neq p(\mathbf{X})$. A well known measure of closeness between $\tilde{p}(\mathbf{X})$ and $p(\mathbf{X})$ is the *Kullback-Leibler distance* (KLD) defined by

$$D(p(\mathbf{X}) \parallel \tilde{p}(\mathbf{X})) = \int \dots \int p(\mathbf{X}) \log \frac{p(\mathbf{X})}{\tilde{p}(\mathbf{X})} d\mathbf{X}. \quad (2.8)$$

The distance between two distributions is measured in *bits* if the logarithm is base-2. This can be used as the optimization criteria for the construction of source factors $g_k(\mathbf{X}_{\mathcal{S}_k})$. By removing some of the conditioning variables from the *chain rule expansion* (CRE) we can obtain *constrained chain rule expansion* (CCRE) [20]. Then, each factor $g_k(\mathbf{X}_{\mathcal{S}_k})$ can be described more formally as

$$g_k(\mathbf{X}_{\mathcal{S}_k}) = p(\mathbf{X}_{\mathcal{A}_k} | \mathbf{X}_{\mathcal{B}_k}), \quad (2.9)$$

where $\mathcal{A}_k, \mathcal{B}_k \subseteq \mathcal{M}$ and $\mathcal{S}_k = \mathcal{A}_k \cup \mathcal{B}_k$. Further, if $\mathcal{A}_k$ and $\mathcal{B}_k$ meet the conditions stated in (2.10), then the expansion in (2.11) is referred as a CCRE of the PDF $p(\mathbf{X})$.

$$\mathcal{A}_k \cap \mathcal{B}_k = \emptyset, \mathcal{B}_k \subseteq \bigcup_{l=1}^{k-1} \mathcal{A}_l, \bigcup_{k=1}^K \mathcal{A}_k = \mathcal{M} \quad (2.10)$$

$$\tilde{p}(\mathbf{X}) = \prod_{k=1}^K p(\mathbf{X}_{\mathcal{A}_k} | \mathbf{X}_{\mathcal{B}_k}), \quad (2.11)$$

In [20] it was shown, that the KLD can be calculated explicitly for a CCRE of a

Gaussian PDF $\mathcal{N}(\mathbf{0}_N, \mathbf{R})$ using

$$D(p(\mathbf{X}) \parallel \tilde{p}(\mathbf{X})) = -\frac{1}{2} \log_2 |\mathbf{R}| + \frac{1}{2} \sum_{k=1}^K \log_2 \frac{|\mathbf{R}_{\mathcal{S}_k}|}{|\mathbf{R}_{\mathcal{B}_k}|}, \quad (2.12)$$

where $\mathbf{R}_{\mathcal{S}_k}$ and $\mathbf{R}_{\mathcal{B}_k}$ are the corresponding covariance matrices of $p(\mathbf{X}_{\mathcal{S}_k})$ and $p(\mathbf{X}_{\mathcal{B}_k})$. For non-Gaussian distributions a closed-form expression for KLD does not exist. Note that, by introducing factor $g_k(\mathbf{X}_{\mathcal{S}_k}) = p(\mathbf{X}_{\mathcal{A}_k} | \mathbf{X}_{\mathcal{B}_k})$ has a KLD advantage of $\frac{1}{2} \sum_{k=1}^K \log_2 \frac{|\mathbf{R}_{\mathcal{S}_k}|}{|\mathbf{R}_{\mathcal{B}_k}|}$. So each factor $p(\mathbf{x}_{\mathcal{A}_k} | \mathbf{X}_{\mathcal{B}_k})$ should be selected such that it gives the maximum advantage, i.e., which reduces the KLD between $p(\mathbf{X})$ and $\tilde{p}(\mathbf{X})$ by the maximum amount. Once the optimal factorization is found, a factor graph is used to represent the factorization and the marginals can be calculated by running the sum-product algorithm. Rather than finding the factors by searching through all possible combinations, a two step procedure was introduced in [21]. As the first step, sensors were clustered based on the KLD. Then, the sensor clusters were linked to obtained a complete factorization.

2.4.1 Source Clustering

The goal is to group the set of sensors $\mathcal{M}$ into clusters, $\Lambda_c \subseteq \mathcal{M}$, such that $\bigcup_{c=1}^C \Lambda_c = \mathcal{M}$ and $\Lambda_p \cap \Lambda_q = \emptyset$, $\forall p \neq q$. Then, an initial approximation to $p(\mathbf{X})$ would be

$$\tilde{p}(\mathbf{X}) = \prod_{c=1}^C p(\mathbf{X}_{\Lambda_c}). \quad (2.13)$$

By assigning $\mathbf{X}_{\mathcal{B}_k} = \emptyset$, the objective function in (2.12) then reduces to

$$D(p(\mathbf{X}) \parallel \tilde{p}(\mathbf{X})) = -\frac{1}{2} \log_2 |\mathbf{R}| + \frac{1}{2} \sum_{c=1}^C \log_2 |\mathbf{R}_{\Lambda_c}|. \quad (2.14)$$

$\mathbf{R}_{\Lambda_c}$ represents the covariance matrix of random variables within the cluster Λ_c , which can easily be obtained by choosing corresponding entries from $\mathbf{R}$.

The clusters Λ_c , are constructed by successive merging process, starting with single-element clusters. They are merged together step-by-step, taking into account the objective function (2.14). Each step selects two clusters, when merged, reduces the number of clusters by one. This process determines the best possible pair of clusters leading to the smallest value for (2.14). The procedure is repeated until only one cluster remains. The history of mergings during the process is recorded. By analyzing the order of the merging, the clusters can be extracted with a maximum cluster size of S (i.e., the process ensures clusters with $|\Lambda_c| \leq S$). As we have seen, this process is based on a local decision at each step. So the solution can be locally optimal.

2.4.2 Source Factorization

Once the clusters are found, the second step would be to obtain a factorization as expressed in (2.11). For this, the derived clusters Λ_c have to be incorporated into CCRE, such that they obey the conditions stated in (2.10). By linking unconnected clusters systematically this can be achieved.

To establish a link between two clusters Λ_i and Λ_j , we have to choose a set of variables $\mathcal{G}_i \subseteq \Lambda_i$ and $\mathcal{H}_j \subseteq \Lambda_j$ from each cluster. Two design parameters G and H are introduced, such that $|\mathcal{G}_i| \leq G$ and $|\mathcal{H}_j| \leq H$. The linking of two clusters adds two factors $p(\mathbf{x}_{\mathcal{H}_j}|\mathbf{x}_{\mathcal{G}_i})$ and $p(\mathbf{x}_{\mathcal{G}_j}|\mathbf{x}_{\mathcal{H}_i})$ to the factorization and the KLD improvement associated with this link is

$$KLD_{i \rightarrow j} = \frac{1}{2} \log_2 \frac{|\mathbf{R}_{\mathcal{G}_i \cup \mathcal{H}_j}|}{|\mathbf{R}_{\mathcal{G}_i}|}. \quad (2.15)$$

After selecting a cluster to start with, one of the unconnected clusters is linked to the already connected clusters, such that it results in the maximum KLD improvement.

As the clustering algorithm, the linking process is also not optimal. But the simulation results in Sec. 2.6 reveal that the decoder designed using these algorithms are considerably superior to the scalar MMSE decoder. Also, its complexity is much lower compared to the MMSE optimal decoder.

2.5 Complexity of the Various Decoders

We wish to compare the complexity of various decoders in terms of the number additions and multiplications the processor has to perform (i.e., *the computational complexity*). First, we analyze the complexity of the decoders designed without using the sum-product algorithm.

Let us revisit the equations, which we have derived in Chapter 1.

$$\hat{x}_n(\hat{\mathbf{i}}) = \sum_{\forall i \in \Omega} \tilde{x}_n(i) P(i_n = i | \hat{\mathbf{i}}) \quad (2.16)$$

$$P(i_n = i | \hat{\mathbf{i}}) = \gamma \cdot \sum_{\forall \mathbf{i} \in \Omega^N; i_n = i} P(\mathbf{i}) \prod_{l=1}^N P(\hat{i}_l | i_l) \quad (2.17)$$

By analyzing these two equations, we can decompose the whole decoding process into three distinct sub-processes as depicted in Fig. 2.3.

Note that, calculation of channel transition probabilities $p(\hat{i}_l | i_l)$ are common to all decoders. With the use of received index vector $\hat{\mathbf{i}}$, $N \cdot 2^Q$ multiplications are necessary to find channel transition probabilities.

Computational complexity of calculating the posterior probabilities $P(i_n = i | \hat{\mathbf{i}})$ depends on the decoder, since each decoder is based on different approximations for

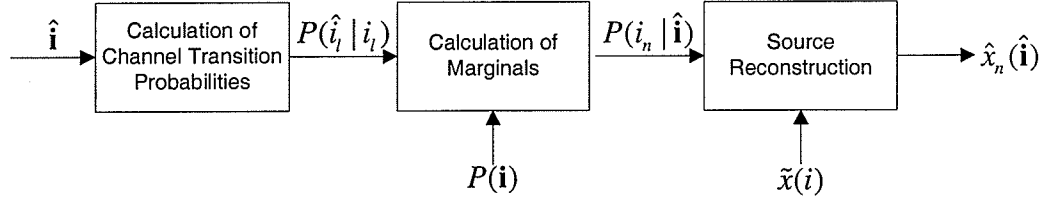


Figure 2.3: Decomposition of the decoding into three stages.

the PMF $P(\mathbf{i})$. For scalar MMSE decoder we only need to compute $N \cdot 2^Q$, probabilities as per (2.4).

The complexity of the reconstruction stage is also common to all decoders and it needs $N \cdot 2^Q$ multiplications and $N \cdot (2^Q - 1)$ additions as per (2.16).

We have discussed the complexity of the sum-product algorithm in Appendix C (refer table C.1). If the sum-product algorithm is used instead of the direct calculation, we can not gain any advantage for the scalar MMSE decoder (refer Table 2.1). But implementing MMSE optimal decoder and scalable decoder (Sec. 2.4) using the sum-product algorithm is always beneficial. The complexity of the scalable decoder depends on the degrees of factors used in the approximated PMF. For a degree- d node, computational complexity is shown in Table C.1.

Table 2.1: Computational complexity of the decoders (a) using equations (2.16) and (2.17) (b) Using factor graphs/sum-product algorithm, N variable nodes with $d = 2$ and N function nodes with $d = 1$ (c) Using factor graphs/sum-product algorithm, N variable nodes with $d = 2$ and 1 function node with $d = N$.

<i>Scalar Decoder</i>	<i>Additions</i>	<i>Multiplications</i>
(a)	0	$N \cdot 2^Q$
(b)	0	$N \cdot 2^Q$
<i>Optimal Decoder</i>	<i>Additions</i>	<i>Multiplications</i>
(a)	$N(2^{Q(N-1)} - 1)$	$N^2 \cdot 2^{QN}$
(c)	$N(2^{Q(N-1)} - 1)$	$N(N-1)2^{QN} + N \cdot 2^Q$

2.5.1 Complexity of using the KLD

Now consider the KLD between PMF $P(\mathbf{I})$ and its approximation $\tilde{P}(\mathbf{I})$:

$$\begin{aligned} D(P(\mathbf{I})\|\tilde{P}(\mathbf{I})) &= \sum_{\mathbf{i}} P(\mathbf{i}) \log \frac{P(\mathbf{i})}{\tilde{P}(\mathbf{i})} \\ &= \sum_{\mathbf{i}} P(i_1, i_2, \dots, i_N) \log \frac{P(i_1, i_2, \dots, i_N)}{\prod_{k=1}^K P(\mathbf{i}_{\mathcal{A}_k} | \mathbf{i}_{\mathcal{B}_k})}. \end{aligned} \quad (2.18)$$

As can be seen, the calculation of (2.18) needs the knowledge of the joint distribution $P(\mathbf{I})$, which is hard to find when N is large. If the number of quantization levels are large (i.e., large Q), the situation becomes even worse. Notice that, further simplification of (2.18) is not possible either. Hence, the use of KLD for the PMF $P(\mathbf{I})$ is highly complex and difficult to use in practice.

2.6 Simulation Results

To perform the simulations, the sensor network model described in Chapter 1 was implemented. The joint source distribution is assumed to be multivariate Gaussian. To investigate the end-to-end performance of the system, we measure the reconstructed-signal-to-noise ratio (RSNR) given by

$$RSNR = 10 \cdot \log \left(\frac{E\|\mathbf{x}\|^2}{E\{\|\mathbf{x} - \hat{\mathbf{x}}\|^2\}} \right) dB \quad (2.19)$$

at different channel-signal-to-noise ratios (CSNR). The expectations are estimated by taking 100000 training samples.

We first investigate the performance of various decoders at transmission bit rates $Q = 1, 2, 3, 4$ bits/sample. The quantizers are Lloyd-Max optimal [5]. Fig. 2.4 shows the simulation results of the decoder, which does not exploit *a priori* information. At

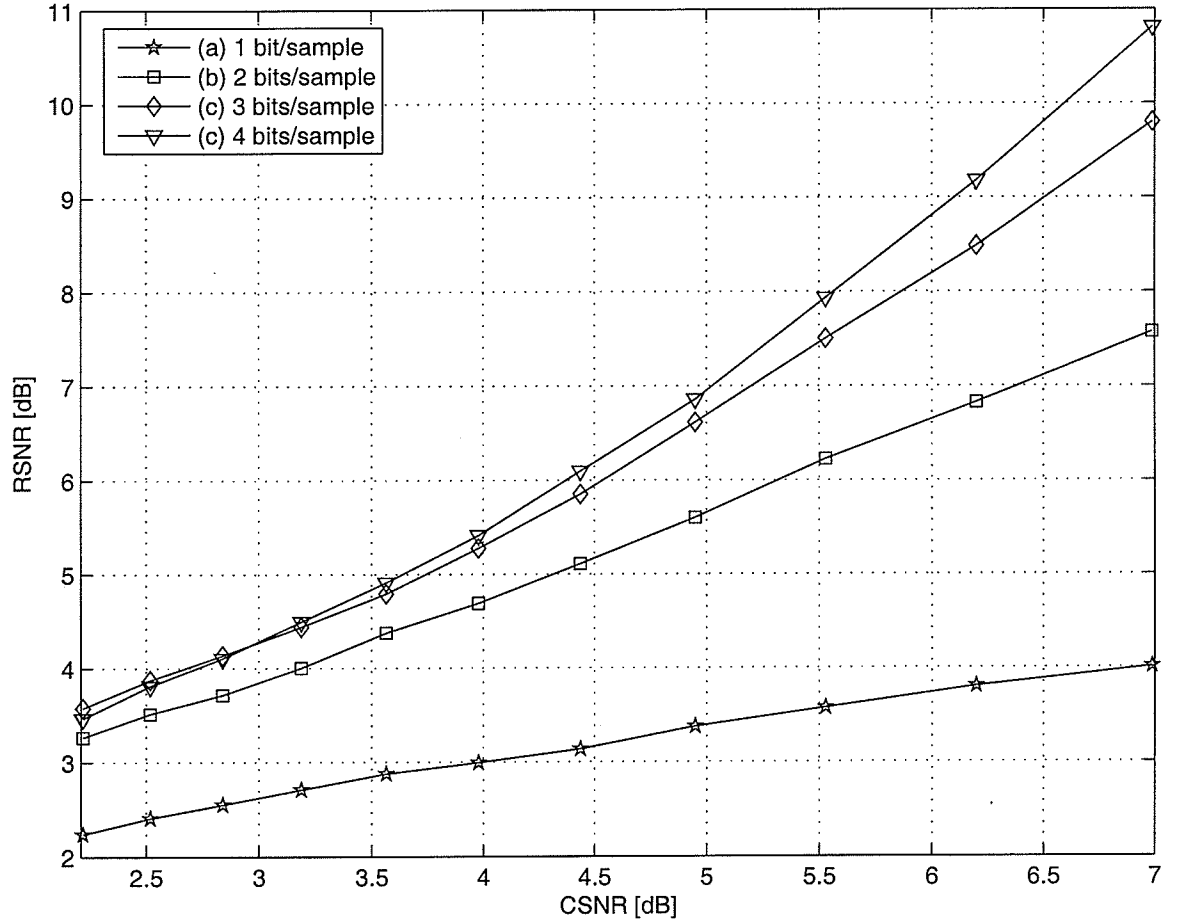


Figure 2.4: The performance of the decoder designed without a priori information. Rate of the quantizers $Q=1, 2, 3, 4$ bits/sample and the sensor network consists of 3 sensors.

high CSNR, the distortion caused by channel noise is very small and the decoder can perform the decoding almost without error. Thus, at high CSNR the performance is governed by the quantization error. This is a common scenario for all the decoders discussed in this thesis. At low CSNR the decoder performance for high values of rate Q is not as good as for low rates.

Fig. 2.5 depicts the performance of scalar MMSE decoder for different bit rates Q and the Fig. 2.6 shows the simulation results for MMSE optimum decoder. It was

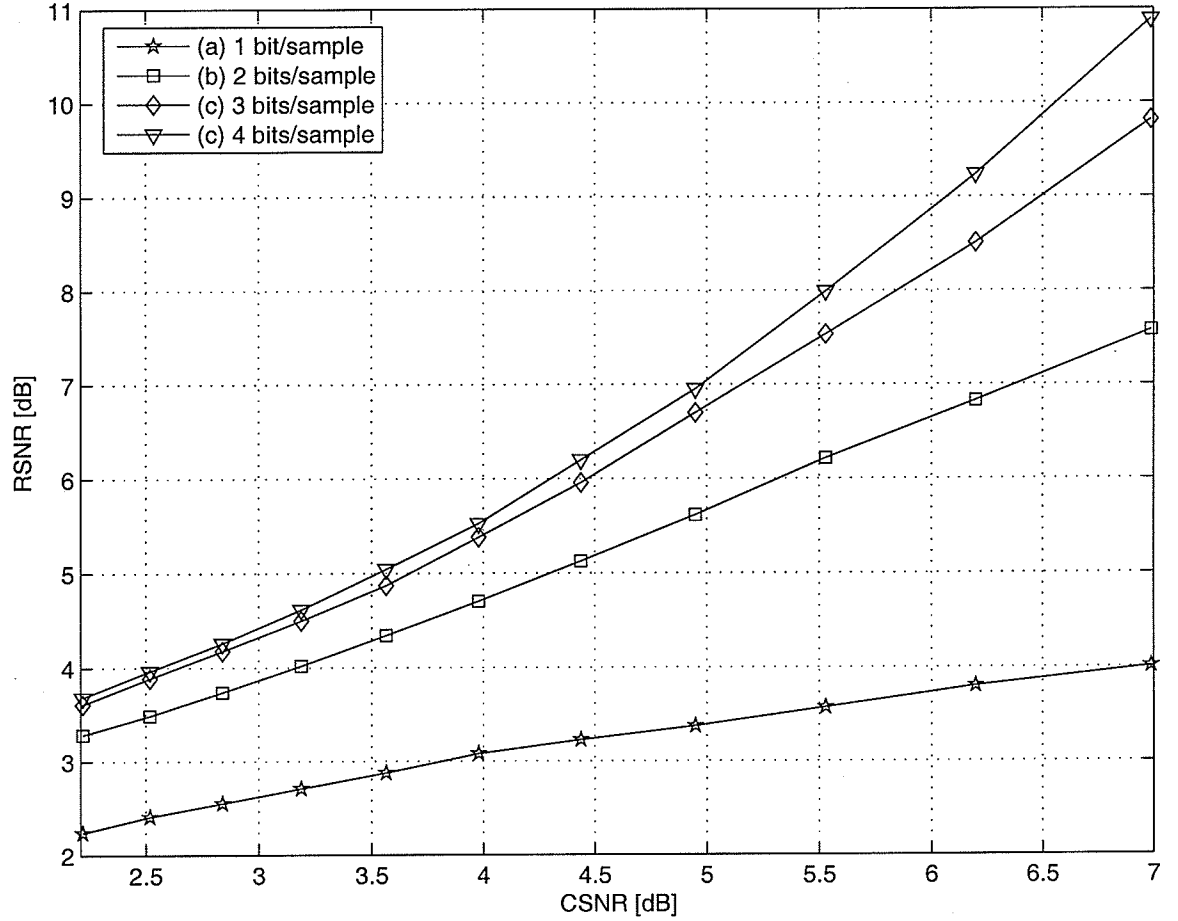


Figure 2.5: The performance of the scalar MMSE decoder. Rate of the quantizers $Q=1, 2, 3, 4$ bits/sample and the sensor network consists of 3 sensors.

observed that at low CSNR, the MMSE optimal decoder performances significantly better than the other two decoders.

Now we will compare the decoders at the same bit rate. Fig. 2.7 depicts the RSNR against CSNR for the 3 decoders the MMSE optimal, the scalar MMSE and the decoder designed without priori information at $Q = 4$ bits/sample. It is clear that using the priori information in the scalar MMSE decoder gives us a slight gain compared to the decoder designed without a priori information. Exploiting the inter-sensor correlation in the MMSE optimal decoder gives 0.6 dB performance gain over

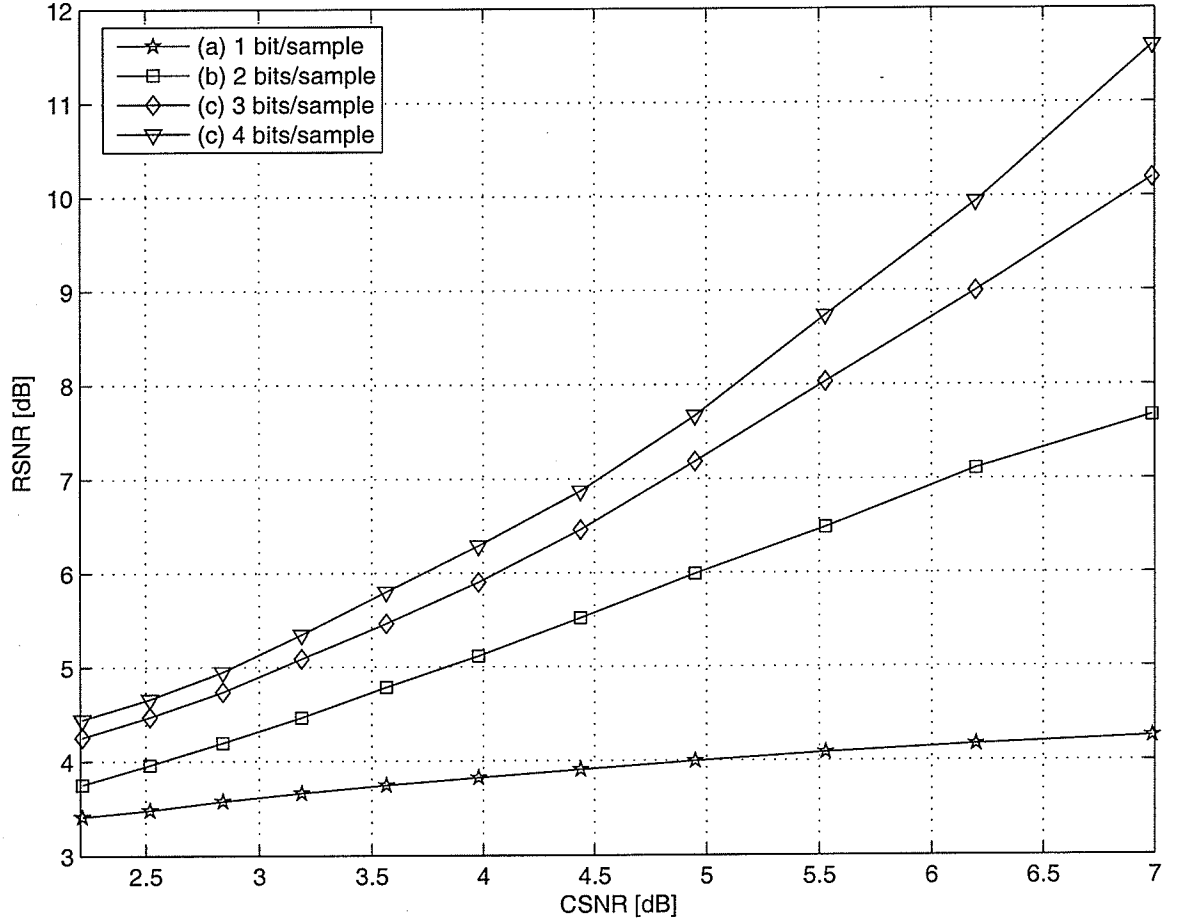


Figure 2.6: The performance of the MMSE optimal decoder. Rate of the quantizers $Q=1, 2, 3, 4$ bits/sample and the sensor network consists of 3 sensors.

the scalar decoder at $\text{CSNR} = 3$ dB.

Next, let us investigate the performance of the scalable decoder discussed in Sec. 2.4. For this scenario we consider a sensor network with 9 sensors and a multivariate Gaussian source distribution. The maximum cluster size is set to $S \leq 3$. By following the clustering algorithm in Sec. 2.4.1, three clusters $\Lambda_1 = \{X_1, X_3, X_7\}$, $\Lambda_2 = \{X_6, X_8, X_9\}$ and $\Lambda_3 = \{X_2, X_4, X_5\}$ were obtained. Then, we can initially approxi-

mate the PMF $P(\mathbf{X})$ by

$$\tilde{P}(\mathbf{X}) = P(X_1, X_3, X_7)P(X_6, X_8, X_9)P(X_2, X_4, X_5). \quad (2.20)$$

The KLD of this approximation is $D(P(\mathbf{X})\|\tilde{p}(\mathbf{X})) = 0.1753$ bits/sensor. By linking these clusters as described in Sec. 2.4.2, the following factorization was obtained (we assumed $G = 2$ and $H = 1$):

$$\tilde{P}(\mathbf{X}) = P(X_6, X_8, X_9)P(X_1, X_7|X_8)P(X_3|X_1, X_7)P(X_2, X_4|X_8)P(X_5|X_2, X_4). \quad (2.21)$$

For this factorization, the KLD $D(P(\mathbf{X})\|\tilde{p}(\mathbf{X})) = 0.067$ bits/sensor. Therefore, the KLD is reduced by linking the clusters together.

Each index i_n depends on a unique sensor symbol x_n . Therefore, factorization of $\tilde{P}(\mathbf{X})$ corresponds to a unique factorization of $\tilde{P}(\mathbf{I})$ and vice-versa. Then, the factorization:

$$\tilde{P}(\mathbf{I}) = P(I_6, I_8, I_9)P(I_1, I_7|I_8)P(I_3|I_1, I_7)P(I_2, I_4|I_8)P(I_5|I_2, I_4) \quad (2.22)$$

can be used to construct the decoder. The factor graph, which corresponds to the factorization (2.22) is shown in Fig. 2.8, and Fig. 2.9 depicts the performance of the corresponding decoder. It is clear that this decoder performs quite close to the MMSE optimal decoder for Gaussian distributed sources and is suitable for a large sensor network, due to its low computational complexity.

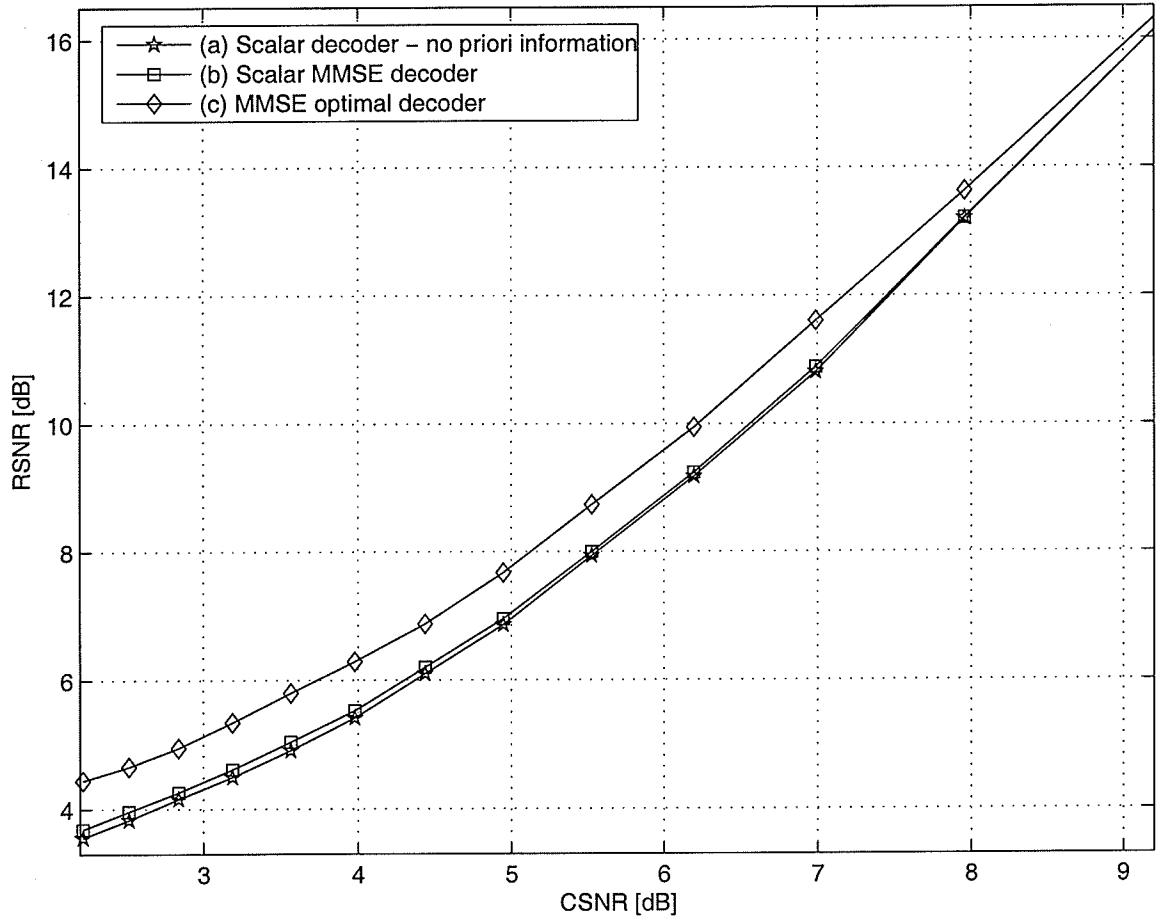


Figure 2.7: Performance comparison of the three decoders at rate $Q=4$ bits/sample. The sensor network consists of 3 sensors.

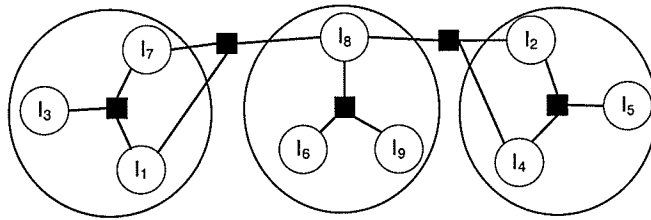


Figure 2.8: Factor graph corresponds to the expansion (2.22). The design parameters are chosen as $G = 2$ and $H = 1$.

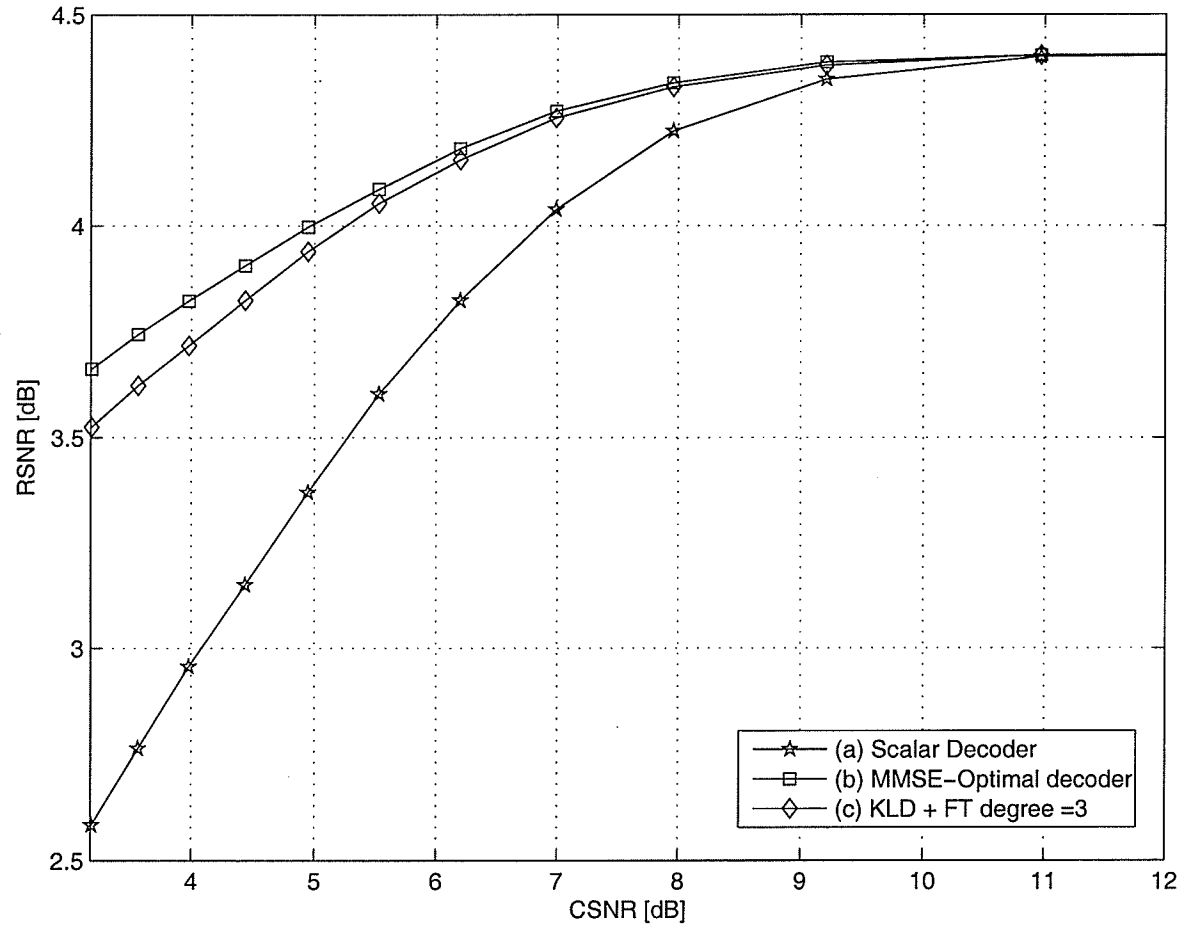


Figure 2.9: Performance of several decoders for a Gaussian source, as a function of CSNR. The network consists of 9 sensor nodes and each node uses quantizers of rate 1-bit/sample.

Chapter 3

Maximum Likelihood based Factorization

In the previous chapters we have presented the scalar MMSE decoder and the optimal MMSE decoder. We have seen that the complexity of the optimal MMSE decoder increases exponentially with the number of sensors and is impractical to handle in a large sensor network. On the other hand, the scalar MMSE decoder is quite simple, due to the correlation of the sensors is ignored in the design. As we have seen in the Chapter 2, it has a significant end-to-end performance loss compared to the MMSE optimal decoder. Therefore, we need to find a decoder which can approach the performance of the optimal decoder, while having a computational complexity, as low as possible. The use of scalable decoders for Gaussian sources, whose complexity is linear in the number of sensors was proposed in [20]. These decoders use factor graphs constructed based on KLD, and the sum-product algorithm (see Appendix C) for the decoding stage. Except for Gaussian distributed sources, a closed-form expression for KLD is difficult to obtain. Hence, the approach [20] can not be used when the joint distribution of sensors is not Gaussian. In this chapter, we will develop

a more general approach to factor graph construction based on *Bayesian networks* (BN) [27]. To factorize $P(\mathbf{I})$, we use the maximum likelihood (ML) estimation of a Bayesian network, using a training set of sensor observations. Unlike in [20], in this case, the joint distribution of the sensor observations need not be necessarily Gaussian. Rather than first finding optimal factors for source PDF, $p(\mathbf{X})$ and then assign that factorization for PMF $P(\mathbf{I})$ [20], our algorithm directly factorizes, $P(\mathbf{I})$, using a training set for index vector $\mathbf{I}$.

3.1 PDF Approximation

Finding an approximation for PMF $P(\mathbf{I})$ in terms of *local functions* or *factors*, is the initial step in the process of searching for a scalable decoder. As we have seen before, a general PDF, $p(\mathbf{X}) = p(X_1, X_2, \dots, X_N)$ can be factorized into an approximate form

$$\tilde{p}(\mathbf{X}) = \prod_{m=1}^M p_m(\Gamma_m), \quad (3.1)$$

where Γ_m is a vector with components made up of a subset of $\{X_1, X_2, \dots, X_N\}$, and each $p_m(\cdot)$ is a factor of the function $\tilde{p}(\mathbf{X})$. If we can factorize $p(\mathbf{X})$ exactly, then $\tilde{p}(\mathbf{X}) = p(\mathbf{X})$. In general, all PDFs can not be expressed exactly in this form. However, we know that by using the *chain rule expansion* (CRE), any PDF can be expressed as a product of conditional probability density functions (CDF) as

$$\begin{aligned} p(\mathbf{X}) &= p(X_1)p(X_2|X_1)\dots p(X_N|X_{N-1}, \dots, X_1) \\ &= \prod_{r=1}^N p(X_r|X_{r-1}, \dots, X_1). \end{aligned} \quad (3.2)$$

From the above expansion, we see a possibility of approximating a PDF using factors. But, some of the factors in this expansion are functions of a large number of

variables. As an example N -th factor is having N variables in it's distribution. i.e., the factorization using the CRE does not scale with the network size N . Consider the r -th factor of this expansion, $p(X_r|X_{r-1}, \dots, X_1)$. We can find an approximation for this factor by retaining only a fixed number of variables, which are most dependant on X_r . But, this has to be done in such a way that the resulting factor is the best possible approximation for the factor $p(X_r|X_{r-1}, \dots, X_1)$.

Let us define some notations which are helpful to express this approximation. We call conditioning variables as *parent variables*. For example, X_l, X_m, X_n are referred to as parent variables for the factor $p(X_i|X_l, X_m, X_n)$. We will denote the set of parents of X_r as Ψ_r .

Example : Let's consider a 5-dimensional PDF $p(\mathbf{X})$. Then, $p(\mathbf{X}) = p(X_1, X_2, X_3, X_4, X_5)$ can be factorized using the CRE as:

$$p(\mathbf{X}) = p(X_1)p(X_2|X_1)p(X_3|X_2, X_1)p(X_4|X_3, X_2, X_1)p(X_5|X_4, X_3, X_2, X_1). \quad (3.3)$$

Now, we wish to approximate these factors by some other factors involving fewer variables. Suppose we know following conditions beforehand:

- X_3 is mostly depends on X_1 .
- X_4 is mostly depends on X_2 .
- X_5 is mostly depends on X_4 .

Then, the approximation for the CRE will be:

$$p(\mathbf{X}) \approx p(X_1)p(X_2|X_1)p(X_3|X_1)p(X_4|X_2)p(X_5|X_4). \quad (3.4)$$

3.2 Probabilistic Graphical Models

One of the most effective ways to represent a distribution in a factorized form is using a *probabilistic graphical model* (PGM) [28, 29]. A PGM can be seen as merger of two disciplines, graph theory and probability theory. A PGM consists of two components, namely *the structure* and *the parameters*. The structure of a PGM depicts the interaction between random variables in the form of a graph. Each node represents a random variable and each edge connecting two nodes represents the existence of an interaction between two variables. The absence of an edge between two nodes represent absence of interaction between the corresponding node-variables. The parameters of a PGM are the functions associated with a node or set of nodes. According to the type of structure used, a PGM can be categorized into two groups.

1. Directed models (e.g. Bayesian networks).
2. Undirected models (e.g. Markov random fields).

Finding the dependency structure and learning parameters of a MRF is computationally more intensive than in a Bayesian network [30]. So, in this thesis we concentrate on Bayesian networks. Factor graphs, which we have discussed in Sec.2.3 are also a special case of PGMs.

3.3 Bayesian Networks

A Bayesian network (BN) is a PGM which can be used to compactly represent the joint PMF of a large number of random variables which exhibit conditional dependencies. It can be regarded as a pair (S, Λ) , where S is the structure of the model and Λ is a set of parameters of the model. The structure S is a *directed acyclic graph* (DAG) [28], where each node corresponds to a variable and each edge corresponds to

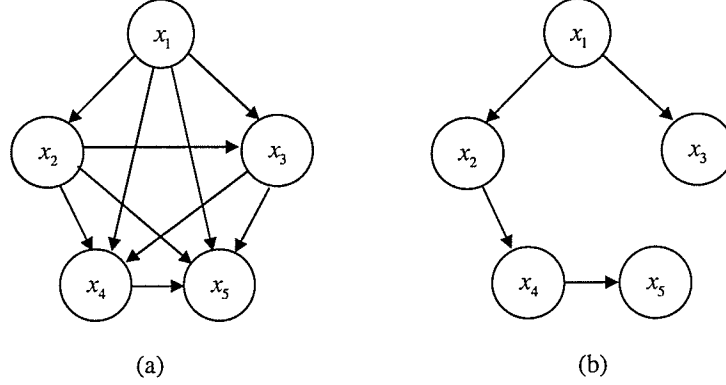


Figure 3.1: Bayesian network representation of (a) Eq. (3.3) and (b) Eq. (3.4).

a conditional dependency. A set of nodes Ψ_r are said to be the parent of X_r if there are edges from each variable in Ψ_r pointing to X_r . In this case, the parameters are $\Lambda = \{p(X_1|\Psi_1), p(X_2|\Psi_2), \dots, p(X_N|\Psi_N)\}$.

Fig. 3.1 shows the BN for PMF given by (3.3) and its approximation given by (3.4). For example, in Fig. 3.1(a), the vertex X_3 has edges from X_1 and X_2 directed towards it. Therefore the parent variables of X_3 in the expansion (3.3) are X_1 and X_2 .

Notice that Fig. 3.1(a) shows an almost fully-connected BN. In general, if a BN is fully connected, then it represents the original PDF. But this need not be always true. The BN shown in Fig. 3.1(b), on the other hand, is not fully connected. Such a BN will allow us to find an approximate expression for a joint PDF.

A BN is a representation for some *specific* expansion of the joint PDF. However, there are more than one CRE for a given PDF, and the expansion depends on the actual order of the variables. For example, we may also expand the function $p(\mathbf{X})$ in (3.3) as

$$p(\mathbf{X}) = p(X_2)p(X_1|X_2)p(X_3|X_2, X_1)p(X_4|X_3, X_2, X_1)p(X_5|X_4, X_3, X_2, X_1). \quad (3.5)$$

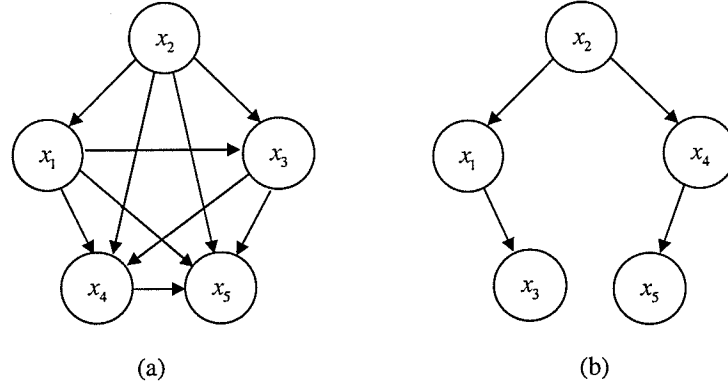


Figure 3.2: Bayesian network representation of (a) Eq. (3.5) and (b) Eq. (3.6).

Using the same assumptions considered for the expansion (3.4), we can approximate $p(\mathbf{X})$ by

$$p(\mathbf{X}) \approx p(x_2)p(x_1|x_2)p(x_3|x_1)p(x_4|x_2)p(x_5|x_4). \quad (3.6)$$

The BNs representing (3.5) and (3.6) are shown in Fig. 3.2. Now, it is clear that there are several ways to factorize a joint PDF using the CRE and consequently, there are many BN representations for the same distribution. But, not all of them correspond to the best approximation for the joint PDF.

Our problem of approximating the PMF $P(\mathbf{I})$ can now be expressed in terms of a BN. The goal here is to obtain a BN that is not fully connected, which contains as many dependencies as possible from the original, fully-connected BN. The number of dependent variables (i.e., the number of variables in the parent set), $\max |\Psi_r|$ (where $|\cdot|$ denotes the size of a set) is a design parameter which controls the connectivity of the BN.

If we know the BN corresponding to a factorization for $P(\mathbf{I})$, we can convert it into a factor graph. That is, a given BN can also be represented by a factor graph. The edges of the factor graph can be directed to illustrate the conditional dependencies

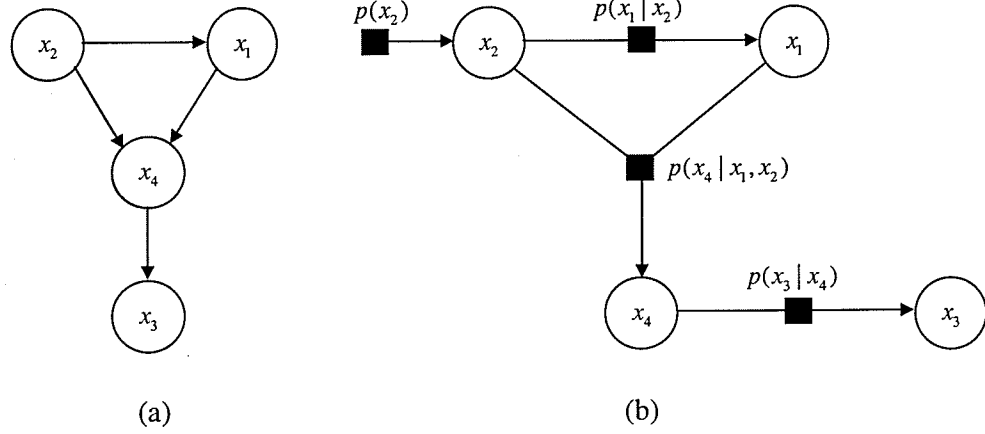


Figure 3.3: (a) Bayesian network representation of global pdf, $p(X_1, X_2, X_3, X_4) = p(X_2)p(X_1|X_2)p(X_4|X_1, X_2)p(X_3|X_4)$. (b) The Bayesian network shown as a factor graph.

as shown in Fig. 3.3. Once the appropriate factor graph is constructed, we can run the sum-product algorithm (see Appendix C) on the factor graph to find the reconstruction values as described in Chapter 2. But, the problem of determining the best BN remains.

In [31], an algorithm based on *mutual information* was proposed to approximate a discrete probability distribution by a product of second-order distributions. Mutual information between two random variables is considered as a measure of dependance. If we confine ourselves to a maximum of one parent for every node (in other words, factor graph of degree-2), we can follow the algorithm in [31] to find an approximate BN for the PMF, $P(\mathbf{I})$. As we will show in Sec. 3.6, for a typical sensor network problem, the decoder performance based on degree-2 factor graph may not acceptable. Thus, our goal is to search for a more flexible method to find factor graphs of different degrees.

3.4 The likelihood of a Bayesian Network Structure

In Sec. 3.3, we have seen that there are several possible ways to approximate a PDF by a BN. But, not all of them would be optimal in the sense of maximizing the decoder performance. The optimal approximation will lead to the best decoder performance. Thus, our goal is to find the optimal BN for the PMF $P(\mathbf{I})$, given the maximum number of parent variables in a factor (i.e., $\max |\Psi_r|$). We propose to use the *maximum-likelihood* (ML) criterion to choose the best BN. Let Λ_k , $k = 1, 2, \dots, K$, be a candidate set of BNs to be used as an approximation to the PMF $P(\mathbf{I})$, and let $\mathcal{S}$ be a sample set from $P(\mathbf{I})$. We calculate the likelihood $P(\mathcal{S}|\Lambda_k)$ over all possible K BNs and choose the best BN as the one with the largest $P(\mathcal{S}|\Lambda_k)$.

Let's denote two BNs which correspond to the PMF $P(\mathbf{I})$ by $\Lambda(\mathbf{I})$ and $\Lambda'(\mathbf{I})$. Then, for given BNs $\Lambda(\mathbf{I})$ and $\Lambda'(\mathbf{I})$, the log-likelihood ratio of the training set $\mathcal{S}$ can be written as,

$$\begin{aligned} \mathcal{L}_{\Lambda\Lambda'} &= \log \frac{P(\mathcal{S}|\Lambda(\mathbf{I}))}{P(\mathcal{S}|\Lambda'(\mathbf{I}))} \\ &= \log P(\mathcal{S}|\Lambda(\mathbf{I})) - \log P(\mathcal{S}|\Lambda'(\mathbf{I})). \end{aligned} \quad (3.7)$$

Now, if $\mathcal{L}_{\Lambda\Lambda'} \geq 0$, the BN $\Lambda(\mathbf{I})$ is preferred over its competitor $\Lambda'(\mathbf{I})$. One approach to comparing the likelihood of a BN with that of another BN was proposed in [32] for a discrete probability distribution. We adopt this approach for our problem.

Let's denote a BN which corresponds to PMF $P(\mathbf{I})$ by $\Lambda(\mathbf{I})$ and parent nodes of I_n by $\Psi_n \in \{\psi_{n1}, \psi_{n2}, \dots, \psi_{nq_n}\}$, $n \in \{1, \dots, N\}$, where N is the number of sensor nodes in the network. Assume that there are q_n unique instantiations of Ψ_n and the set of quantization levels of the n -th node be $\Omega_n \in \{v_{n1}, \dots, v_{nr_n}\}$. Further, let L_{njl} denotes the number of times $I_n = v_{nl}$ and $\Psi_n = \psi_{nj}$ jointly occurred in the training

set $\mathcal{S}$. Then, it can be shown that (see Appendix D for proof),

$$P(\mathcal{S}|\Lambda(\mathbf{I})) = \prod_{n=1}^N \prod_{j=1}^{q_n} \frac{(r_n - 1)!}{(L_{nj} + r_n - 1)!} \prod_{l=1}^{r_n} L_{njl}! \quad (3.8)$$

where,

$$L_{nj} = \sum_{l=1}^{r_n} L_{njl}. \quad (3.9)$$

If we assume that each sensor node has a quantizer of the same rate Q , (3.8) can be reduced to,

$$P(\mathcal{S}|\Lambda(\mathbf{I})) = \prod_{n=1}^N \prod_{j=1}^{q_n} \frac{(Q - 1)!}{(L_{nj} + Q - 1)!} \prod_{l=1}^Q L_{njl}!. \quad (3.10)$$

We assume our training set is fairly large such that we can calculate any PMF very close to its actual value. But, this results in large values for L_{njl} and L_{nj} , and the calculation of the factorials in (3.10) becomes intractable with finite precision computations. On the other hand, comparing the logarithms as in (3.7) will make such calculations more tractable. We can now further reduce $\mathcal{L}_{\Lambda\Lambda'}$ into the following form using equation (3.10):

$$\mathcal{L}_{\Lambda\Lambda'} = \sum_{l=1}^Q \log \prod_{n=1}^N \frac{\prod_{j=1}^{q_n} L_{njl}!}{\prod_{j=1}^{q'_n} L'_{njl}!} - \log \prod_{n=1}^N \frac{\prod_{j=1}^{q_n} (L_{nj} + Q - 1)!}{\prod_{j=1}^{q'_n} (L'_{nj} + Q - 1)!}. \quad (3.11)$$

Example 3.1 *Let's consider a small sensor network with 3 sensors. In this system source samples X_1, X_2 , and X_3 , are quantized with 1-bit scalar quantizers. Then, the outputs of the quantizers assume values in the set $\{0,1\}$. Table 3.1 depicts the output indices for source training set of 10 samples. We will compare the two BNs shown in Fig. 3.4 to check which one of them explains the given PMF $P(I_1, I_2, I_3)$ better. For this particular case, $N = 3$ and $Q = 2$. For the structure Λ , the parents $\Psi_1 = I_2$ and*

Table 3.1: Training set-the outputs of the quantizers

Sample	i_1	i_2	i_3
1	1	0	1
2	0	0	0
3	1	1	0
4	0	0	0
5	1	1	1
6	0	1	0
7	0	0	0
8	1	1	1
9	0	0	0
10	1	1	1

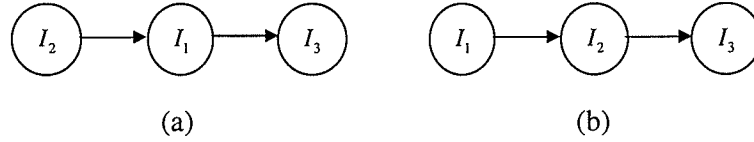


Figure 3.4: (a) BN Λ corresponds to factorization $P(I_2)P(I_1|I_2)P(I_3|I_1)$. (b) BN Λ' corresponds to factorization $P(I_1)P(I_2|I_1)P(I_3|I_2)$.

$\Psi_3 = I_1$ and for structure Λ' , $\Psi_2 = I_1$ and $\Psi_3 = I_2$. Then for a given I_n , Ψ_n can take only two possible values (in this example 0 or 1). Therefore, $q_n = 2$. Table 3.2 provides the numerical values of L_{njl} and L'_{njl} , which can be easily computed using the training set. Assigning these values to equation (3.11) results in $\mathcal{L}_{\Lambda\Lambda'} = 1 \geq 0$. Then, given the assumptions, the training set implies that the BN Λ is 10 times more likely than the BN Λ' . Therefore, the factorization $P(I_2)P(I_1|I_2)P(I_3|I_1)$ is a better approximation to $P(I_1, I_2, I_3)$ than $P(I_1)P(I_2|I_1)P(I_3|I_2)$.

Table 3.2: Numerical values of the parameters

Structure Λ	Structure Λ'
$L_{111} = 4$	$L'_{211} = 4$
$L_{112} = 1$	$L'_{212} = 1$
$L_{121} = 1$	$L'_{221} = 1$
$L_{122} = 4$	$L'_{222} = 4$
$L_{311} = 5$	$L'_{311} = 4$
$L_{312} = 1$	$L'_{312} = 2$
$L_{321} = 0$	$L'_{321} = 1$
$L_{322} = 4$	$L'_{322} = 3$

3.5 An algorithm to search for ML Bayesian Network

Once we have found a metric to measure the goodness of the network structure, a search through the space of all possible candidate structures should be performed in order to find the BN which corresponds to the ML approximation.

We can write any BN Λ in terms of factors as follows.

$$\tilde{P}(\mathbf{I}) = \prod_{n=1}^N P(I_n | \Psi_n), \quad (3.12)$$

where

$$\Psi_n \subset \Phi_n.$$

The set Φ_n contains the set of nodes which precede I_n in the expansion. Then, Λ can be considered as a combination of BNs Λ_n , $n = 1, 2, \dots, N$, where Λ_n corresponds to the factor $P(I_n | \Psi_n)$. Now the optimization is a matter of selecting Λ_n such that $\log P(\mathcal{S} | \Lambda)$ is the maximum. A fully connected tree with vertices corresponds to factors $P(I_n | \Psi_n)$ can be constructed to represent this problem. The edge between two vertices represents the weight $\log P(\mathcal{S} | \Lambda_n)$. A full search through this tree will provide the ML BN. However, the complexity of this task is exponential in

the number of vertices. We can adopt Prim's *weighted spanning tree algorithm* [33] to find the optimal BN with comparatively low complexity. But, we propose a greedy optimization algorithm due to its simplicity. Even though the proposed algorithm in general generates a sub-optimal solution, the experimental results show that the resulting solution is quite effective in sensor network problems.

This algorithm starts with an initial structure with no edges. Then, at each step an edge is added to the structure such that it results in the maximum increase in likelihood. This process is continued until all the nodes are added to the network. The final BN will have all the nodes connected. But, we have to guarantee that it does not have cycles i.e., satisfies the DAG principle.

Further, let there be an upper limit of S for the number of parent variables, i.e., $|\Psi_n| \leq S$. A pseudocode describing the algorithm is given in Algorithm 1 on page 44. This algorithm is greedy, since we accept only the factor $P(I_j|\Psi_j)$ with the maximum likelihood in each step (i.e., the solution may not be globally optimal). Alternatively, an algorithm such as the M-L algorithm [34] can be used to improve the result.

3.6 Simulation Results

Having seen how we can find the best approximation for the CRE by using the knowledge of the likelihood of a Bayesian structure, we shall now analyze the performance of the decoders designed based on this approach. We convert each of the factorizations (or corresponding BNs) into a factor graph. Some factorizations yield factor graphs with cycles and some are cycle free i.e., factor trees (FT) [25]. For factor graphs with cycles, the algorithm becomes iterative because the messages circulate in end-less cycles and the marginal probabilities can not be computed exactly. A flooding schedule [25] was used in such situations with a maximum of 6 time steps.

Algorithm 1 Greedy search for the best BN

1. set max to small negative number
 2. **FOR** $n = 1$ **TO** N
 3. $\Pi = \{I_1, \dots, I_N\}$
 4. $size = 1$;
 5. $\Phi = I_n, \mathcal{L}_n = 0$;
 6. **WHILE** $\{\Pi - \Phi\} = \emptyset$
 7. Find (I_j, Ψ_j) pair which maximizes $\mathcal{L}$ for all combinations
where, $I_j \in \{\Pi - \Phi\}$, $\Psi_j \subset \Phi$ and $|\Psi_j| = size$;
 8. add (I_j, Ψ_j) to the variable $list_n$
 9. $\mathcal{L}_n = \mathcal{L}_n + \mathcal{L}$;
 10. $\Phi = \Phi \cup I_j$;
 11. **IF** $size < S$ **THEN**
 12. $size = size + 1$;
 13. **END{IF}**
 14. **END{WHILE}**
 15. **IF** $max < \mathcal{L}_n$
 16. $max = \mathcal{L}_n$;
 17. accept $list_n$;
 18. **END{IF}**
 19. **END{FOR}**
- { $list_n$ contains the BN corresponding to the best approximation
for CRE and max holds its goodness of fit.}
-

It was seen that the results are converging after 5-6 time steps and no considerable gain could be achieved thereafter.

The sensor network model we consider here is as described in Chapter 1. For the simulations, a sensor network having 9 sensors was considered and two plots were generated. In the first, we use one-bit Lloyd-Max quantizers and in the second we used two-bit Lloyd-Max quantizers. For these cases, the simulation of the MMSE optimal decoder is computationally feasible and we can compare it's performance with that of the scalable (sub-optimal) decoders. As the decoding complexity grows exponentially with the number of bits used for quantization, we did not use more than two bits. We illustrate the performance of the decoders for a general source distribution described below.

First, we generate two Gaussian distributed data sets D_{n1} and D_{n2} according to $\mathcal{N}(\mu_{n1}, \sigma_{n1})$ and $\mathcal{N}(\mu_{n2}, \sigma_{n2})$, where $n = \{1, \dots, N\}$. Then, we assign a probability γ_n which governs the selection of a sample set S_n from the above two data sets. S_n of size 100000 is generated, where $\gamma_n \times 100000$ samples are selected from the data set D_{n1} and $(1 - \gamma_n) \times 100000$ are selected from D_{n2} . To impose correlation, we chose an arbitrary matrix $\mathbf{A}_{N \times N}$ with $\mathbf{A}(i, j) = 1$ for $i = j$ and derive the required sensor output vector $\mathbf{X} = (X_1, X_2, \dots, X_N)^T$ from $\mathbf{X} = \mathbf{A}\mathbf{Y}$, where $\mathbf{Y} = [S_1, S_2, \dots, S_N]^T$. To investigate the end-to-end performance of the system, we measure the RSNR at different CSNR.

Now we will investigate the performance mismatch of the scalable decoders with respect to likelihoods of the BNs, which are constructed as approximations to the joint PMF $P(\mathbf{I})$. Approximations (3.13), (3.14), and (3.15) were used in the simulations.

$$\tilde{P}(\mathbf{I}) = P(I_3)P(I_7|I_3)P(I_4|I_3, I_7)P(I_2|I_4, I_7)P(I_9|I_2, I_7)P(I_5|I_4, I_7)P(I_8|I_5, I_7)P(I_1|I_2, I_9)P(I_6|I_1, I_8) \quad (3.13)$$

$$\tilde{P}(\mathbf{I}) = P(I_3)P(I_7|I_3)P(I_4|I_3, I_7)P(I_5|I_4, I_7)P(I_8|I_4, I_5)P(I_2|I_4, I_7)P(I_9|I_2, I_4)P(I_6|I_2, I_9)P(I_1|I_2, I_9) \quad (3.14)$$

$$\tilde{P}(\mathbf{I}) = P(I_9)P(I_2|I_9)P(I_4|I_2, I_9)P(I_8|I_2, I_9)P(I_5|I_2, I_9)P(I_1|I_2, I_9)P(I_7|I_2, I_9)P(I_6|I_2, I_9)P(I_3|I_2, I_9) \quad (3.15)$$

The maximum number of parent variables in a factor was limited to 2. The log-likelihoods of these approximations are listed in Table 3.3.

Table 3.3: Likelihoods of different factorizations

Factorization	Likelihood
(3.13)	-1.3682×10^5
(3.14)	-1.5052×10^5
(3.15)	-1.7246×10^5

Factorization (3.13) is the best approximation we could get by following the Algorithm 1. As we explained in Sec. 3.5 this may be a sub-optimal solution. But, as can be seen from the Figs. 3.8 and 3.9 this factorization provides performance very close to the MMSE decoder and is worth accepting due to its simplicity compared to the MMSE decoder.

For the comparison purposes, we implement the decoder based on PMF approximation method proposed by Chow and Liu in [31]. They used mutual information, $I(i_j, i_k)$ (refer (3.16)) between two random variables as the measure of closeness of PMFs, given by

$$I(i_j, i_k) = \sum_{i_j, i_k} P(i_j, i_k) \log \left(\frac{P(i_j, i_k)}{P(i_j)P(i_k)} \right). \quad (3.16)$$

If $I(i_j, i_k) > I(i_l, i_m)$, then the factor $P(i_j|i_k)$ is added to the factorization. Approximated PMF we could get from this method is given by (3.17) and the corresponding factor tree is illustrated in Fig. 3.7. It has a overall closeness of 2.9349 bits.

$$\tilde{P}(\mathbf{I}) = P(I_2)P(I_9|I_2)P(I_4|I_2)P(I_5|I_4)P(I_8|I_5)P(I_1|I_2)P(I_6|I_1)P(I_7|I_4)P(I_3|I_7) \quad (3.17)$$

But, their method gives factors with only 1 parent variable (i.e., $S = 1$), which construct a degree-2 factor tree. Obviously, their performance is worse than the degree-3 factor graph with optimal factorization (refer to Fig. 3.8), but may be better than other non-optimal factorizations.

Fig. 3.10 illustrates the convergence of the decoding algorithm when the factor graph contains cycles. Note that, the convergence is quite fast, and needs only about 3 time steps.

We conclude with that, there is a strong relationship between the likelihoods and the decoder performance.

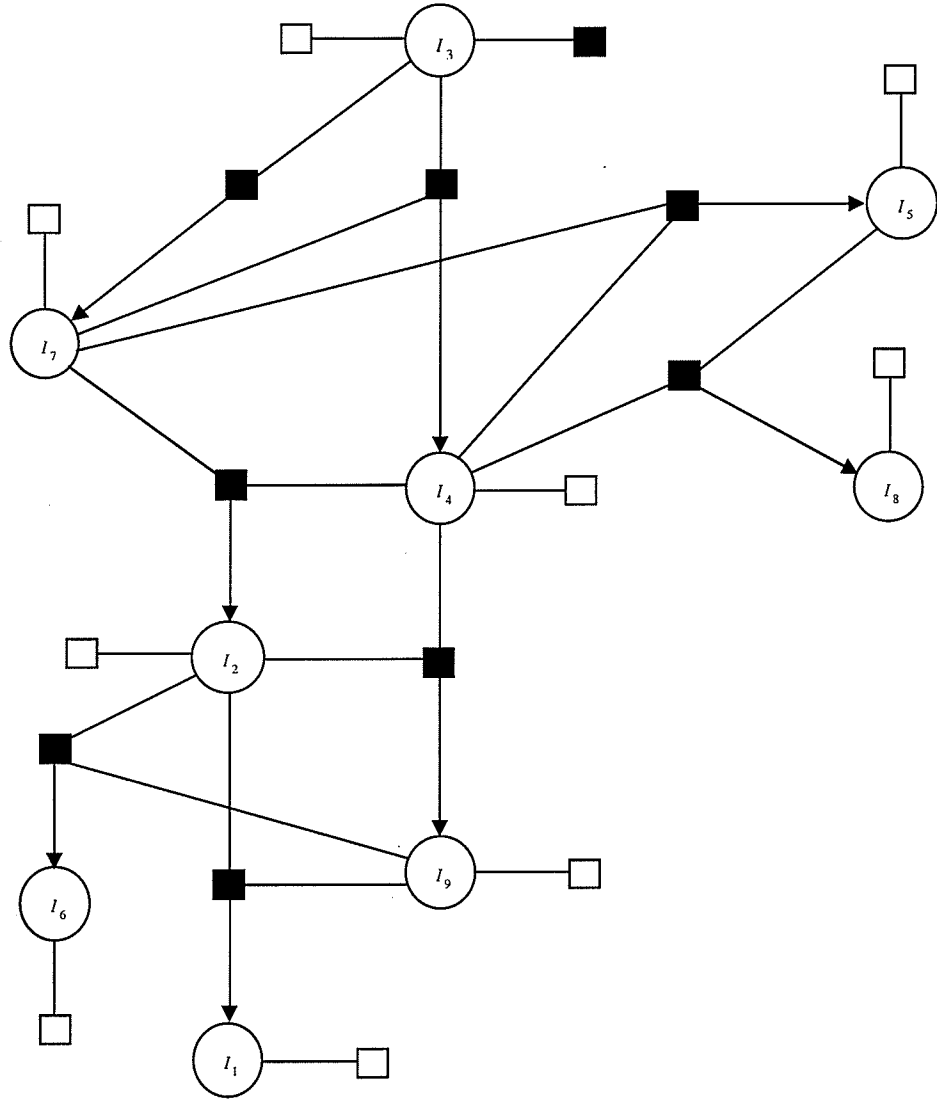


Figure 3.5: Factor graph belongs to non-optimal BN with maximum number of parents, $S = 2$. Corresponding factorization is as (3.14) and the likelihood of the structure is -1.5052×10^5 .

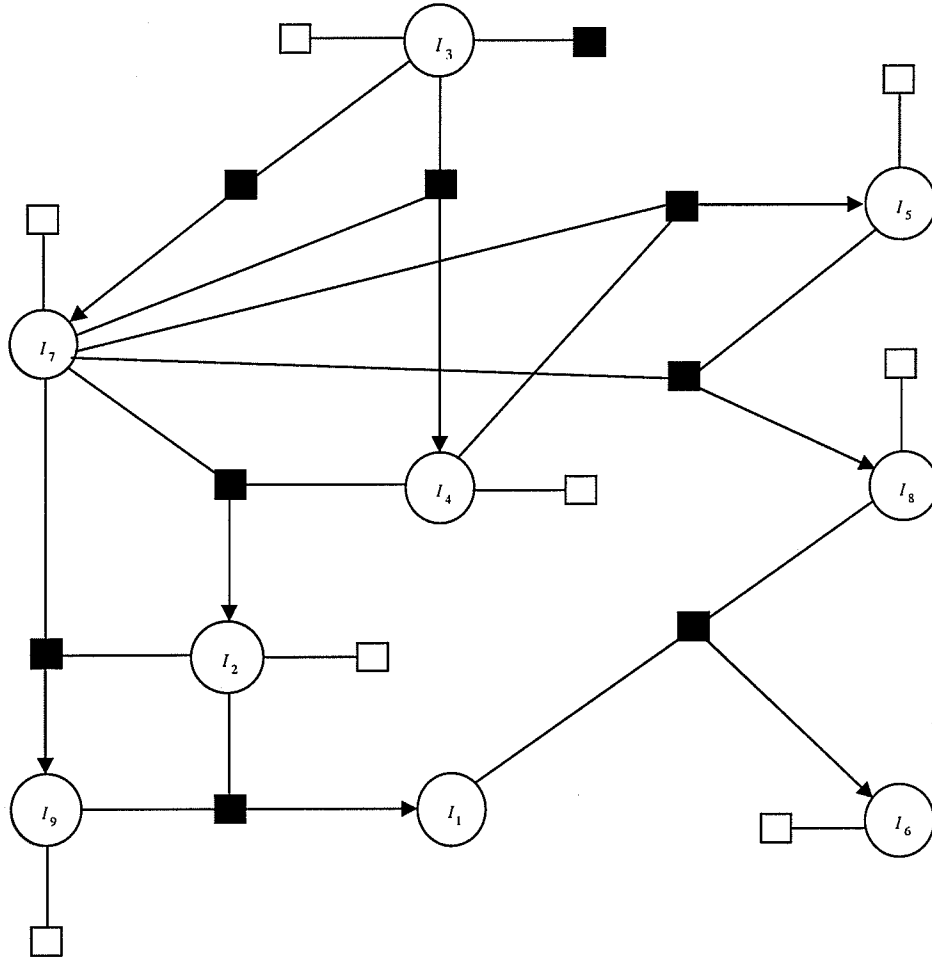


Figure 3.6: Factor graph corresponds to optimal BN found using Algorithm 1. Maximum number of parents, $S = 2$. The resulting factorization is (3.13) and the likelihood of the structure is -1.3682×10^5 .

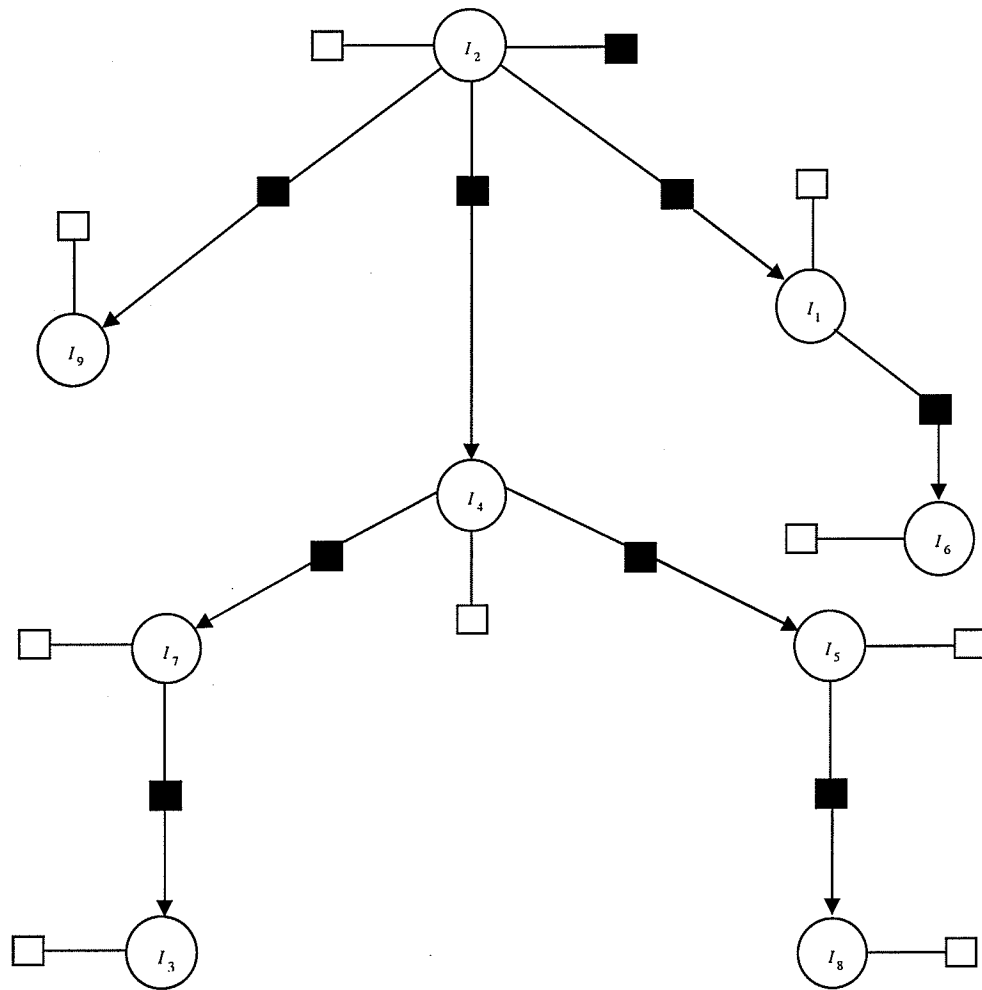


Figure 3.7: Factor tree corresponds to the optimal BN found using the mutual information as the measurement of closeness. Maximum number of parents, $S = 1$. The resulting factorization is (3.17) and the overall closeness of the structure is 2.9349 bits.

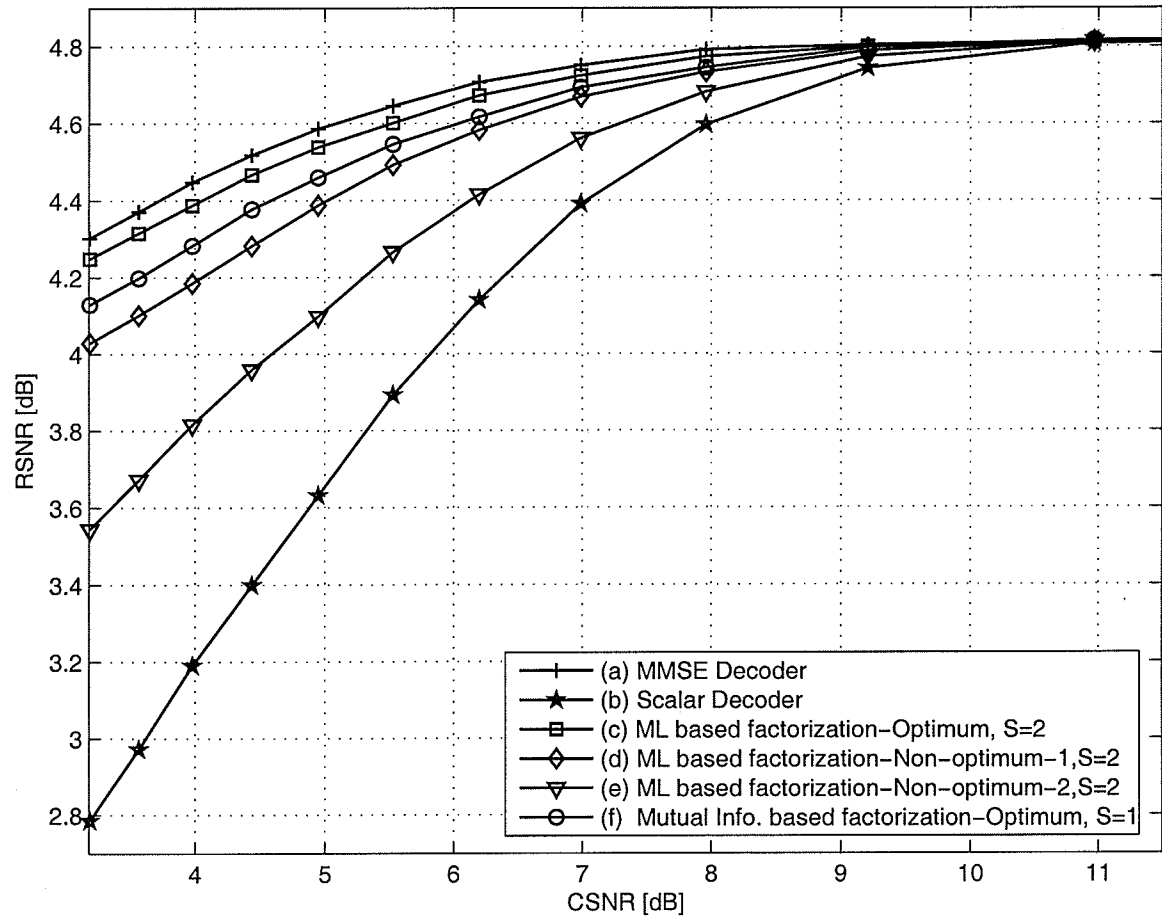


Figure 3.8: Performance of the decoders as a function of CSNR. The network has 9 sensors and each of them uses 1 bit/sample Lloyd-Max quantization.

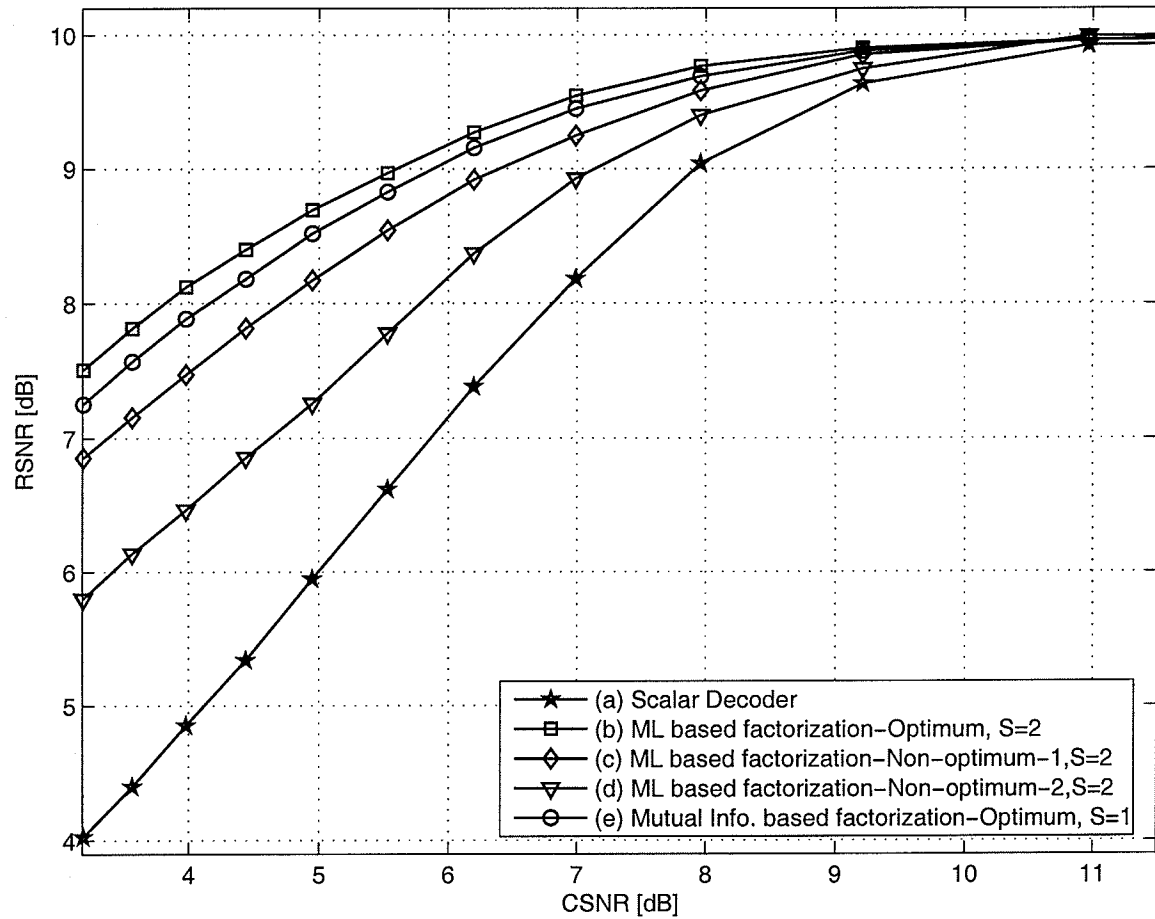


Figure 3.9: Performance of the decoders as a function of CSNR. The network has 9 sensors and each of them uses 2 bits/sample Lloyd-Max quantization.

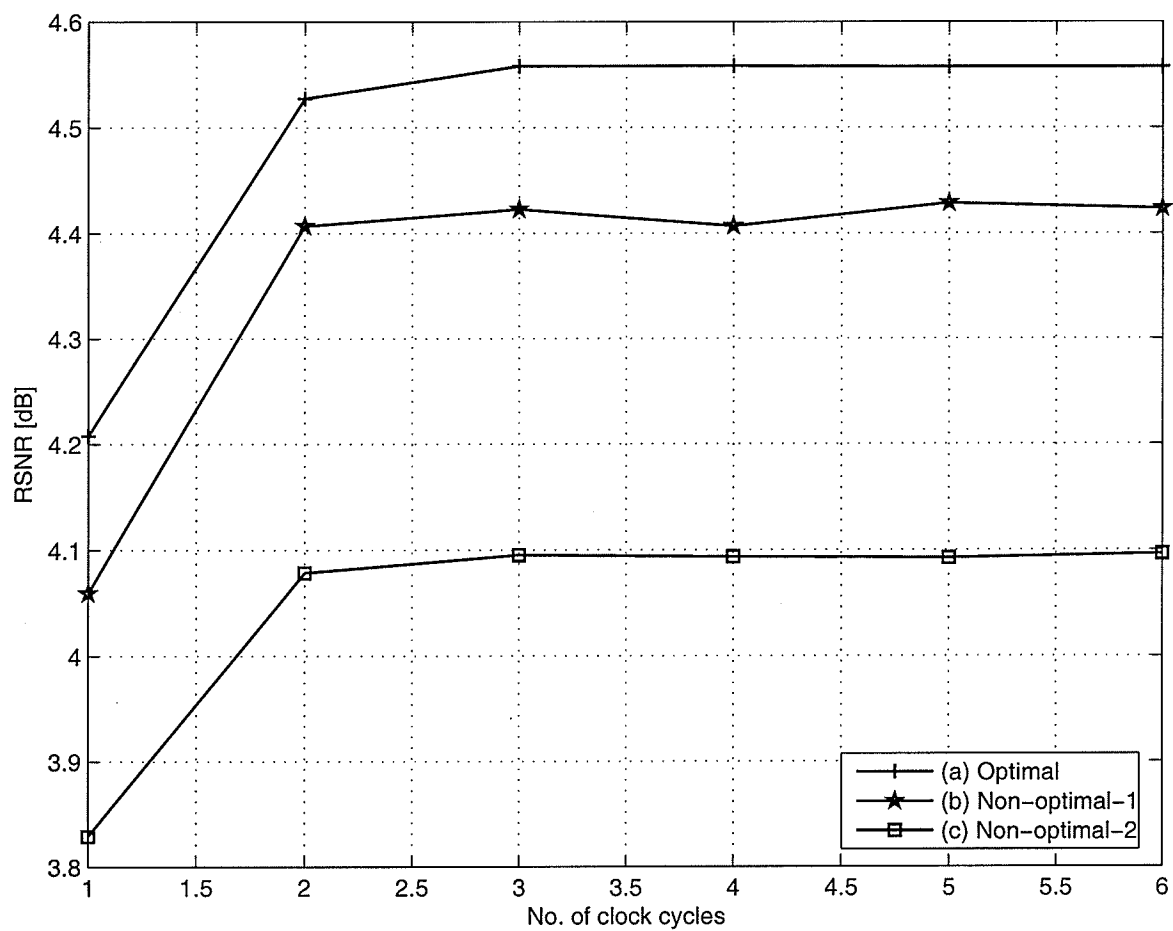


Figure 3.10: Plot of RSNR against number of clock cycles during the execution of the sum-product algorithm, for CSNR= 5 dB.

Chapter 4

Factorization of a PMF based on Sensor Clustering

Even though we could get decoders with promising performance by using the CRE to factorize the joint PMF of quantized sensor outputs as discussed in Chapter 3, the number of factors in the expansion is equal to the number of sensor nodes, N . Therefore, such decoders do not scale with the network size. Also, the more factors we have in the expansion, higher will be the complexity of calculating the marginals using the sum-product algorithm. In this chapter a solution to this problem is proposed.

As we noted in the previous chapters, complexity of the decoder can be decreased if we can factorize PMF, $P(\mathbf{I})$ into factors, which are functions of a smaller number of indexes. As an example, we can simply factorize $P(\mathbf{I})$ into marginal functions of I_n and use the sum-product algorithm on a degree-1 factor graph. This yields a scalar MMSE decoder which doesn't take the inter-source correlation into account. On the other hand, CRE provides an exact approximation to $P(\mathbf{I})$, but implementation of the corresponding decoder is difficult due to cycles in the factor graph and the

high computational complexity. Therefore, our aim is to find functions $f_m(\cdot)$, $m = 1, 2, \dots, M$ such that the PMF, $\tilde{P}(\mathbf{I}) = \prod_{m=1}^M f_m(\cdot)$ is a close approximation to $P(\mathbf{I})$, while eliminating the disadvantages discussed above. In this thesis it is assumed that a sufficiently large set $\mathcal{S}$ of observations of $\mathbf{I}$, representing the PMF $P(\mathbf{I})$ is available to the designer. Then, the factorization $\tilde{P}(\mathbf{I})$ is constructed to match the empirical PMF represented by $\mathcal{S}$.

Rather than starting with the BN corresponding to the case of all independent variables, as in the previous chapter, a better method is to first determine a set of factors according to the degree of their statistical dependence and then construct the final factorization $\tilde{P}(\mathbf{I})$ based on these factors. Based on this idea, in this chapter we propose a two step procedure for constructing an approximate PMF for $P(\mathbf{I})$.

- **Step 1:** Clustering of sensor nodes to obtain an initial factorization for $P(\mathbf{I})$.

The grouping of highly dependent sensors in this manner also simplifies the design of *binning* functions to be used in the encoders, while retaining the ability to exploit correlation among highly correlated sensors in distributed encoding. i.e., binning performed only among the sensor nodes within a cluster.

We denote l -th clusters by Γ_l . Initially, $P(\mathbf{I})$ is approximated by using factors formed by these clusters as,

$$\tilde{P}(\mathbf{I}) = \prod_{l=1}^L P(\Gamma_l), \quad (4.1)$$

where

$$\Gamma_l \subset \mathbf{I}$$

$$\Gamma_j \cap \Gamma_k = \emptyset \text{ for } j \neq k \text{ and}$$

$$\bigcup_{l=1}^L \Gamma_l = \mathbf{I}.$$

- **Step 2:** Once the clusters are obtained, the second step will be to link the unconnected clusters together, so that the resulting PMF leads to a factor graph of a given degree. To accomplish this task, we propose a ML based factor graph construction method. Our goal is to obtain a factor graph, which allows us to compute a near-optimal approximation to (1.12) at a complexity linear in the network size.

We will investigate the factorization based on clusters and how we can optimize the encoders further by using binning in Sec. 4.2 and Sec. 4.3, respectively. As the first step, we will propose below an algorithm to group the sensors into clusters.

4.1 An approach to Sensor Clustering based on Statistical Dependence

We cluster sensors based on the degree of dependence between the sensors. This method starts with a complete undirected graph, i.e., a graph in which each node is connected to every other node. Then, edges corresponding to mutually independent nodes are eliminated systematically, leaving the most dependant edges intact.

There are several statistical tests available to measure the statistical dependence between variables. The *G-test* [35], *Fisher's exact test* [36] and the *Pearson's Chi-square test* [37, 38] are the most common. In case of a small training set, the Fisher's exact test is preferred over the Chi-square test, while for a large training set the G-test gives approximately the same results as the Chi-square test. We restrict our attention to the Chi-square test in this thesis, assuming that the given training set is fairly large.

Pearson's Chi-square test is used to assess the dependence of paired observations of two variables, expressed in a contingency table. For example one could test, whether the people from different regions differ in the frequency with which they report that they support a political candidate.

Let O_l be the *observed* frequency and E_l be the *expected* frequency of an event l , then the Pearson's test statistic is defined as [38]

$$\chi^2 = \sum \frac{(O_l - E_l)^2}{E_l}. \quad (4.2)$$

In terms of probabilities this can be written as,

$$\chi^2 = \sum_{l=1}^k \frac{(O_l - MP_l)^2}{MP_l} \quad (4.3)$$

where, P_l is the *hypothesized probability* [38] and M is the size of the training set. Then, if we divide χ^2 statistic by M , for large M , the value of χ^2/M approximates to

$$\Upsilon^2 = \sum_{l=1}^k \frac{(P_l^* - P_l)^2}{P_l}, \quad (4.4)$$

where, P_l^* is the *true probability* [38] of realization l and $\Upsilon^2 = \chi^2/M$. To conduct the above test for a sample data set, we need to extend it for all realizations of variables I_j and I_k as shown below.

$$\Upsilon_{j,k}^2 = \sum_{i_j, i_k} \frac{(\hat{P}(i_j, i_k) - \hat{P}(i_j)\hat{P}(i_k))^2}{\hat{P}(i_j)\hat{P}(i_k)}, \quad (4.5)$$

where $\hat{P}(\cdot)$ are the PMFs estimated from the training set.

In general Chi-square procedure, $\Upsilon_{j,k}^2$ is compared with value $\chi_{1-\alpha}^2(m-1)$, where α is the degrees of freedom and m is the number of possible outcomes of each event

to check the dependence of I_j and I_k . But, it should be noted that we use $\Upsilon_{j,k}^2$ only as a measure of dependence and do not compare $\Upsilon_{j,k}^2$ with the Chi-square values i.e., we conclude I_j is more dependent on I_k than I_l , iff $\Upsilon_{j,k}^2 > \Upsilon_{j,l}^2$.

As two design parameters, a maximum number $\mathcal{Z}_{max}$ and a minimum number $\mathcal{Z}_{min}$ of nodes to be included in each cluster are pre-specified. On the one hand, it is necessary to limit the cluster size, in order to control the complexity associated with the degree- d of the final factor-graph. On the other hand, the lower limit on the cluster size ensures that the correlation among sufficiently large number of sensors is exploited by each encoder through binning. Note that, limiting the degree- d also simplifies the design of binning functions to be used in the encoders.

The clustering algorithm described here is based on *divisive-hierarchical clustering algorithm* [39] and *agglomerative-hierarchical clustering algorithm* [40]. In the formulation proposed, we modify the general form of these algorithms to accommodate the constraints on the cluster size. The goal is to cluster N sensor nodes into L clusters which obey (4.1) and satisfy the two size constraints.

4.1.1 Divisive Clustering

We start the algorithm with a single cluster Γ , which includes all the nodes connected to each other. The PMF associated with this setup is $P(\mathbf{I}) = P(I_1, I_2, \dots, I_N)$. A successive edge removal process is then executed, based on Υ^2 given by (4.5). In each run of this process, the edge between nodes I_j and I_k which corresponds to the lowest $\Upsilon_{j,k}^2$ is removed.

- Suppose the removal of the edge between nodes I_j and I_k splits the cluster Γ_c

into two clusters Γ_{c_1} and Γ_{c_2} i.e., $\Gamma_c = \Gamma_{c_1} \cup \Gamma_{c_2}$. If

$$|\Gamma_{c_1}| < \mathcal{Z}_{min} \text{ and } |\Gamma_{c_2}| < \mathcal{Z}_{min}, \quad (4.6)$$

(both conditions are true), then we mark that edge elimination as invalid and leave the cluster Γ_c intact. However, since Γ_c violates the upper limit on the size, we may need to break this cluster in the future. In general, another edge removal step in the future will break this cluster into two clusters with acceptable sizes.

- If only one of the conditions in (4.6) is not satisfied, the removal is marked as valid and the clusters Γ_{c_1} and Γ_{c_2} are separated. If an edge removal resulted in a cluster Γ_{c_i} , where $\mathcal{Z}_{min} \leq |\Gamma_{c_i}| \leq \mathcal{Z}_{max}$, then Γ_{c_i} is add to the set Δ (the set of acceptable clusters), and nodes of that cluster are kept away from further node separation.
- The above procedure is repeated such that $\bigcup_{i=1}^C \Gamma_{c_i} = \Gamma$ and $\Gamma_{c_j} \cap \Gamma_{c_k} = \emptyset$, for all $j \neq k$. It should be noted that, during this phase we accept clusters with sizes less than $\mathcal{Z}_{min}$, such as singletons (one-element clusters). In the next step, we will fix this problem.

In Fig. 4.1, the final results of divisive clustering process is illustrated for an exemplary case of 15 sensors. Note that clusters $\Gamma_{c_5}, \Gamma_{c_6}, \Gamma_{c_7}$ and Γ_{c_8} do not satisfy the minimum size limit of 2.

4.1.2 Agglomeration Clustering

In this phase, the lower limit of cluster size is achieved by successive merging of clusters obtained by divisive clustering. Let the set $\bar{\Delta}$ contain the clusters with

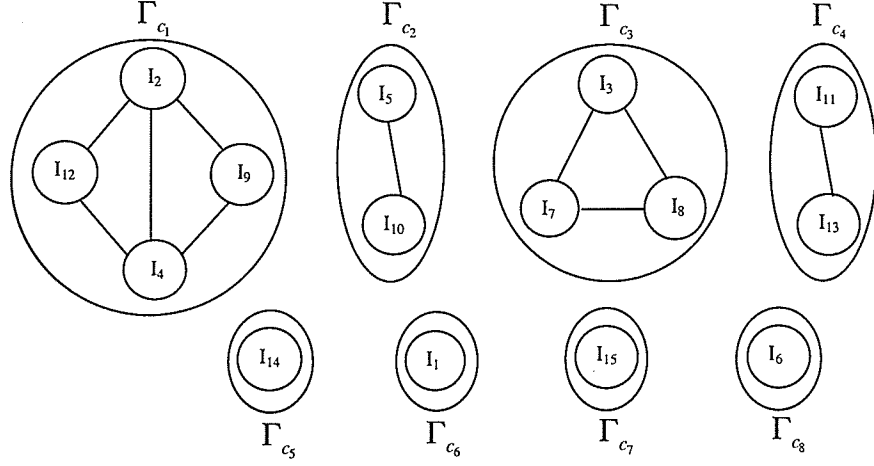


Figure 4.1: Clustering of 15 sensors, $\mathcal{Z}_{max} = 4$ and $\mathcal{Z}_{min} = 2$. Clusters, $\Gamma_{c_1} = \{I_2, I_4, I_9, I_{12}\}$, $\Gamma_{c_2} = \{I_5, I_{10}\}$, $\Gamma_{c_3} = \{I_3, I_7, I_8\}$, $\Gamma_{c_4} = \{I_{11}, I_{13}\}$, $\Gamma_{c_5} = \{I_{14}\}$, $\Gamma_{c_6} = \{I_1\}$, $\Gamma_{c_7} = \{I_{15}\}$, $\Gamma_{c_8} = \{I_6\}$.

$|\Gamma_{c_i}| < \mathcal{Z}_{min}$. We need to merge clusters in $\bar{\Delta}$ with each other or merge one cluster from $\bar{\Delta}$ and other from Δ to form a cluster Γ_l , such that $\mathcal{Z}_{min} \leq |\Gamma_l| \leq \mathcal{Z}_{max}$

Let, the set $\mathcal{O}$ contain the nodes of the clusters Γ_{c_i} , $i = 1, 2, \dots, C$. The node I_j of the cluster Γ_{c_r} and the node I_k of the cluster Γ_{c_s} is connected to form cluster Γ_l if,

$$\Upsilon_{j,k}^2 > \Upsilon_{u,v}^2, \forall u, v \in \mathcal{O},$$

and

$$|\Gamma_l| \leq \mathcal{Z}_{max}.$$

In Fig. 4.2, the final result after executing the step 1 and the step 2 of the algorithm is illustrated for the previous exemplary scenario. Note that $\Gamma_{c_1}, \Gamma_{c_{14}} \in \bar{\Delta}$ are merged to form the cluster Γ_5 and $\Gamma_{c_{15}} \in \bar{\Delta}$, $\Gamma_{c_3} \in \Delta$ have formed the cluster Γ_3 .

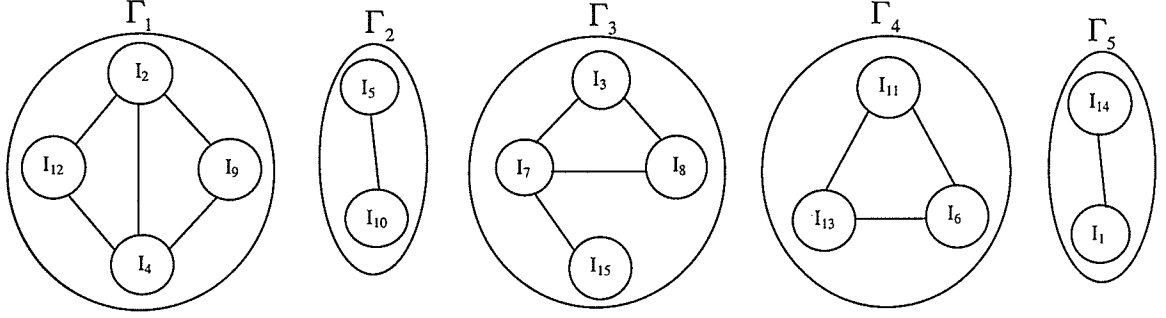


Figure 4.2: Final result after step 1 and step 2. $\mathcal{Z}_{max} = 4$ and $\mathcal{Z}_{min} = 2$. Clusters, $\Gamma_1 = \{I_2, I_4, I_9, I_{12}\}$, $\Gamma_2 = \{I_5, I_{10}\}$, $\Gamma_3 = \{I_3, I_7, I_8, I_{15}\}$, $\Gamma_4 = \{I_6, I_{11}, I_{13}\}$ and $\Gamma_5 = \{I_1, I_{14}\}$.

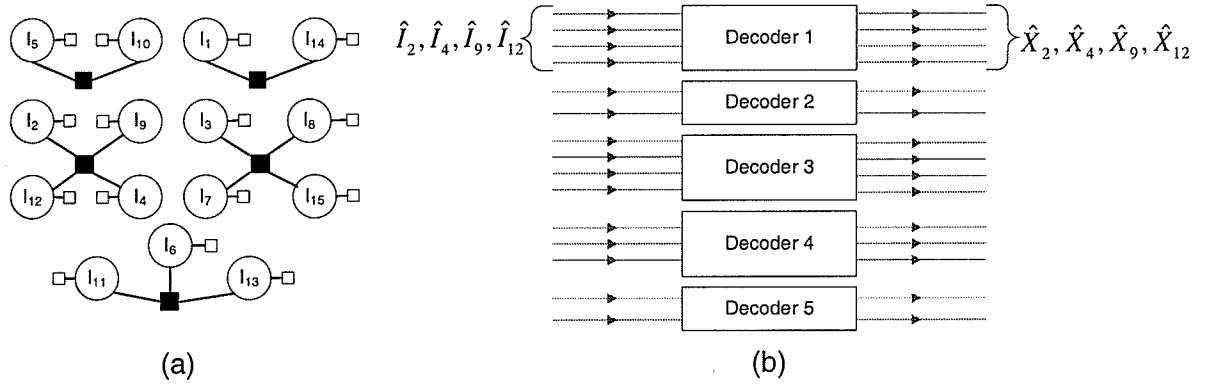


Figure 4.3: (a) Factor graph corresponds to the clusters (b) Separate decoding based on the clusters.

The relationship in Fig. 4.2 yields the following approximation to the PMF $P(\mathbf{I})$.

$$\begin{aligned} \tilde{P}(\mathbf{I}) &= \prod_{l=1}^5 P(\Gamma_l) \\ &= P(I_2, I_4, I_9, I_{12})P(I_5, I_{10})P(I_3, I_7, I_8, I_{15})P(I_6, I_{11}, I_{13})P(I_1, I_{14}). \end{aligned} \quad (4.7)$$

This approximation can be used to construct a decoder using the factor graph shown in Fig. 4.3(a). The end-to-end performance of this decoder will be superior to that of the scalar decoder since it uses the correlation of the sensors within a cluster

for decoding. The resulting system is similar to a system in which the large sensor network is divided into small groups of sensors and a separate decoder is used to decode each group (see Fig. 4.3(b)). If we can find a way to include the correlation between the clusters in the decoding process, then the performance of the system can be further improved.

4.2 Factor Graph Construction using Bayes Networks

In the Sec. 4.1 we have explained how to construct the clusters to fit our purpose, using the statistical dependence of the training set. The next step would be to link the unconnected clusters Γ_l , $l \in \{1, \dots, L\}$ together so as to obtain a fully connected factor graph (or a factor tree). We propose a step-by-step procedure, which starts with the factor graph obtained by the clustering process (see Fig. 4.3(a)). At each step the existing factor graph is modified as follows. Lets assume we have the factor graph Λ_t at step t . Then, Λ_1 for the above example is depicted in Fig. 4.3(a). At each iteration t , the current factor graph Λ_t is modified by linking an isolated cluster Γ_l to the already connected clusters, such that Λ_{t+1} corresponds to a more closer approximation of PMF $P(\mathbf{I})$ than Λ_t . At the end of the linking process we expect to achieve an approximation to PMF $P(\mathbf{I})$ of the form,

$$\tilde{P}(\mathbf{I}) = \prod_{m=1}^M P(\alpha_m | \beta_m), \quad (4.8)$$

where α_m and β_m consist subsets of nodes in different clusters or in a same cluster. We call β_m as the parent of α_m and they obey the following conditions:

$$\alpha_m \cap \beta_m = \emptyset, \quad (4.9)$$

$$\bigcup_{m=1}^M \alpha_m = \mathbf{I}, \quad (4.10)$$

$$\beta_m \subseteq \bigcup_{h=1}^{m-1} \alpha_h. \quad (4.11)$$

Thus, the parent set β_1 is always empty and $\tilde{P}(\mathbf{I})$ consists $M - 1$ conditional probability functions as factors. Note that, the initial factor is not a conditional density function.

The procedure for linking the clusters starts with an arbitrary cluster α_1 , and selects one of the unconnected clusters to be linked with α_1 . Before going into the details of how this is done, we have to define two design parameters λ and μ which bound the sizes of α_m and β_m as,

$$|\alpha_m| \leq \lambda, \quad |\beta_m| \leq \mu.$$

Otherwise, there are quite a number of possible ways to select the variables from the clusters, which can be used for the factorization. Establishing a link between nodes $\alpha_r \subseteq \Gamma_r$ and $\beta_s \subseteq \Gamma_s$ adds two factors, namely $P(\alpha_r|\beta_s)$ and $P(\{\Gamma_r - \alpha_r\}|\alpha_r)$, to the factorization. Here $\{\Gamma_r - \alpha_r\}$ denotes the set of nodes in Γ_r which excludes the set α_r . This is equivalent to replacing a factor $P(\Gamma_r)$ in (4.1) by the new factor

$$\begin{aligned} P(\Gamma_r|\beta_s) &= P(\alpha_r|\beta_s)P(\{\Gamma_r - \alpha_r\}|\alpha_r, \beta_s) \\ &\approx P(\alpha_r|\beta_s)P(\{\Gamma_r - \alpha_r\}|\alpha_r). \end{aligned} \quad (4.12)$$

The approximation in (4.12) is reasonable since the sensors within a given cluster are considered highly correlated. For example, consider the factor graph shown in Fig. 4.4, where $\lambda = \mu = 2$. In this case, the cluster $\Gamma_2 = \{I_5, I_6, I_9\}$ is linked to the

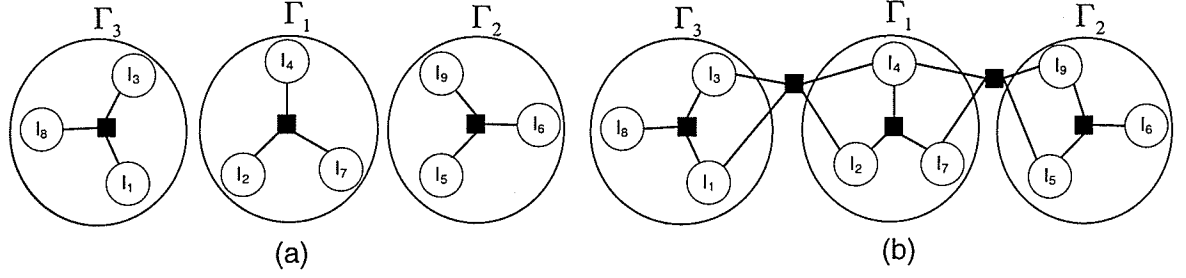


Figure 4.4: (a) Factor graph corresponds to distribution $\tilde{P}(\mathbf{I}) = P(I_2, I_4, I_7)P(I_5, I_6, I_9)P(I_1, I_3, I_8)$ (b) Factor graph after linking clusters, $\tilde{P}(\mathbf{I}) = P(I_2, I_4, I_7)P(I_5, I_9|I_4, I_7)P(I_6|I_5, I_9)P(I_1, I_3|I_2, I_4)P(I_8|I_1, I_3)$.

cluster $\Gamma_1 = \{I_2, I_4, I_7\}$ to obtain the factor $P(\Gamma_2|\beta) = P(I_5, I_9|I_4, I_7)P(I_6|I_5, I_9)$, where $\alpha = \{I_5, I_9\}$, $\beta = \{I_4, I_7\}$, and $\{\Gamma_2 - \alpha\} = \{I_6\}$.

With this background, we are now in a position to explain how to select α_r from the unconnected clusters and β_s from the connected clusters, such that the approximated PMF is optimal. Let there be two pairs (α_r, β_s) and $(\alpha_{r'}, \beta_{s'})$ from which we have to select the best. When linked, they produce two factors $P(\alpha_r|\beta_s)P(\{\Gamma_r - \alpha_r\}|\alpha_r)$ and $P(\alpha_{r'}|\beta_{s'})P(\{\Gamma_{r'} - \alpha_{r'}\}|\alpha_{r'})$, respectively. Now, by using Bayes theorem, we can further expand each factor $P(\alpha_r|\beta_s)$ [or $P(\{\Gamma_r - \alpha_r\}|\alpha_r)$] into the following form,

$$P(\alpha_r|\beta_s) = \prod_{h=1}^{|\mathcal{H}_r|} P(\Phi_h|\Psi_h, \beta_s), \quad (4.13)$$

where $\mathcal{H}_r$ is the set contains elements in α_r . Each Φ_h is a single element set such that,

$$\bigcup \Phi_h = \mathcal{H}_r, \Phi_{h_1} \cap \Phi_{h_2} = \emptyset \text{ for } h_1 \neq h_2,$$

and

$$\Psi_h = \bigcup_{l=1}^{h-1} \Phi_l.$$

Note that there are $|\mathcal{H}_r|!$ possible ways of writing $P(\alpha_r|\beta_s)$ in the form given by

(4.13). For example, $P(i_a, i_b | i_c) = P(i_a | i_c)P(i_b | i_a, i_c) = P(i_b | i_c)P(i_a | i_b, i_c)$. But, we observed in the simulations that all of these forms yield similar decoder performance.

An expansion similar to (4.13) can also be found for the factor $P(\{\Gamma_r - \alpha_r\} | \alpha_r)$. Once we convert these factors in to Bayesian form we can use BNs to represent them. Let BNs $\Lambda_{rs}^{(1)}$ and $\Lambda_{rs}^{(2)}$ correspond to factors $P(\alpha_r | \beta_s)$ and $P(\{\Gamma_r - \alpha_r\} | \alpha_r)$ respectively. Then,

$$P(\mathcal{S} | \Lambda_{rs}) = P(\mathcal{S} | \Lambda_{rs}^{(1)}) \times P(\mathcal{S} | \Lambda_{rs}^{(2)})$$

$$\log P(\mathcal{S} | \Lambda_{rs}) = \log P(\mathcal{S} | \Lambda_{rs}^{(1)}) + \log P(\mathcal{S} | \Lambda_{rs}^{(2)}) \quad (4.14)$$

where Λ_{rs} is the BN found combining $\Lambda_{rs}^{(1)}$ and $\Lambda_{rs}^{(2)}$. A similar result can be obtained for $P(\alpha_{r'} | \beta_{s'})P(\{\Gamma_{r'} - \alpha_{r'}\} | \alpha_{r'})$. Then, we can find the overall log-likelihood ratio as,

$$\mathcal{L} = \log \frac{P(\mathcal{S} | \Lambda_{rs})}{P(\mathcal{S} | \Lambda_{r's'})} = \log P(\mathcal{S} | \Lambda_{rs}) - \log P(\mathcal{S} | \Lambda_{r's'}). \quad (4.15)$$

If $\mathcal{L} \geq 0$, $P(\alpha_r | \beta_s)P(\{\Gamma_r - \alpha_r\} | \alpha_r)$ is preferred over $P(\alpha_{r'} | \beta_{s'})P(\{\Gamma_{r'} - \alpha_{r'}\} | \alpha_{r'})$ and it is added to the factorization and the corresponding link is made in the factor graph (i.e., $\Lambda_t \rightarrow \Lambda_{t+1}$). We can use (3.10) derived in Chapter 3 to calculate the likelihood of a training set, given the BN Λ . Therefore, $\mathcal{L}$ can be calculated easily using (4.14) and (4.15). We can extend this method to find the (α, β) pair with the maximum likelihood, from all possible combinations.

The algorithm we have presented above to construct the best factor graph is a greedy search algorithm yielding a locally optimal solution. But simulation results revealed that the performance of the decoder constructed using this algorithm is quite close to the MMSE optimal decoder. In order to find the global optimal, a full tree search must be carried out, which is extremely complex and is impractical for a large scale network. A reduced complexity search method such as the M-L algorithm [34]

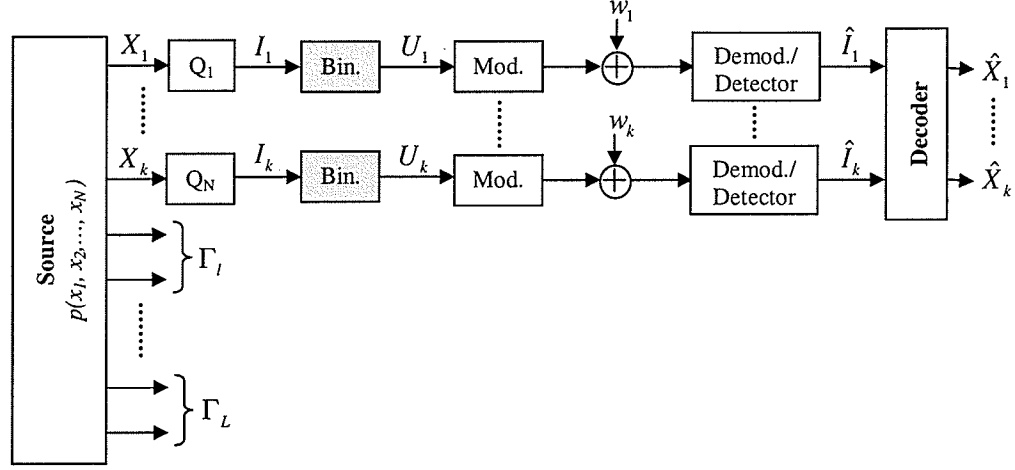


Figure 4.5: System model with binning

may be used to find an improved solution. We have not considered this approach in this thesis.

4.3 Optimization of Binning Functions

The binning can be used in the encoder to reduce the transmission bit rate by using the correlation between the sensor outputs of different sensors $(I_1, I_2, \dots, I_n)$. In binning, the quantizer index $I_n \in \{1, 2, \dots, R_{I_n}\}$ is mapped to $U_n \in \{1, 2, \dots, R_{U_n}\}$ prior to the transmission, where $R_{I_n} \geq R_{U_n}$. The binning stage can be represented by a mapping function h_n such that $u_n = h_n(i_n)$, $n \in \{1, 2, \dots, N\}$ (see Fig. 4.5). To optimize the binning functions h_n , we use a *simulated annealing* (SA) based combinatorial optimization algorithm introduced in [41].

Consider the cluster 1 depicted in Fig. 4.5. Let $\mathbf{X}^{(1)} = (X_1, \dots, X_k)^T$ and $\hat{\mathbf{X}}^{(1)} =$

$(\hat{X}_1, \dots, \hat{X}_k)^T$. Then, the end-to-end distortion $d(\Gamma_1)$ can be determined by

$$\begin{aligned}
 d(\Gamma_1) &= E\{\|\mathbf{X}^{(1)} - \hat{\mathbf{X}}^{(1)}\|^2\} \\
 &= E\{(\mathbf{X}^{(1)} - \hat{\mathbf{X}}^{(1)})^T \cdot (\mathbf{X}^{(1)} - \hat{\mathbf{X}}^{(1)})\} \\
 &= \sum_{x_m \in \Gamma_1} E\{(x_m - \hat{x}_m)^2\}.
 \end{aligned} \tag{4.16}$$

Each distortion term, $E\{(x_m - \hat{x}_m)^2\}$ is the sum of distortions due to joint decoding (based on indices $I_1, \dots, I_k$) and the distortion due to binning. Our concern here is to choose optimal mapping functions such that they minimizes (4.16). For a cluster of size k , MSE due to binning is (assuming Q -level quantizers)

$$D_{bin.} = \sum_{i_1=1}^Q \dots \sum_{i_k=1}^Q \left(\sum_{n=1}^k \|E(X_n|i_1, \dots, i_k) - E(X_n|u_1, \dots, u_k)\|^2 \right) P(i_1, \dots, i_k). \tag{4.17}$$

Then, we have a combinatorial optimization problem of minimizing (4.17), with respect to the index mapping functions h_n . The SA-based optimization algorithm given in [41, 42] is used to solve this problem and the interested reader is referred to these two publications.

Then, we can replace $P(\hat{i}_n|i_n)$ in (1.12) by

$$P(\hat{i}_n|i_n) = \sum_{\forall u \in U_n} P(\hat{i}_n|u_n = u) \cdot P(u_n = u|i_n), \tag{4.18}$$

which allows us to use binning in decoding. Note that,

$$P(u_n|i_n) = \begin{cases} 1 & \text{if } i_n \text{ is mapped to } u_n \\ 0 & \text{otherwise.} \end{cases} \tag{4.19}$$

4.4 Simulation Results

In order to investigate the performance of scalable decoders constructed with the proposed algorithm, we conducted a number of simulation experiments involving both Gaussian and non-Gaussian sensor observations. In all of our experiments with jointly Gaussian sensor observations, the factor-graphs constructed with the proposed approach and the KLD-based approach [20, 21] were found to be identical, and hence the performances of the resulting decoders were also identical. However, when the sensor observations were non-Gaussian, the proposed approach leads to designs which were significantly better. Throughout the simulations, we choose parameters for clustering and factorization as $Z_{min} = 2$, $Z_{max} = 3$ and $\lambda = 2$, $\mu = 1$. A training set of 100,000 samples was used.

To generate non-Gaussian training sets, we follow the same procedure described in Chapter 3. We have considered two scenarios: a sensor network with 9 sensors and a relatively large sensor network with 100 sensors. In the Gaussian case, the sensor output vector $\mathbf{x} = (x_1, x_2, \dots, x_N)^T$ is assumed to be a N -dimensional Gaussian random vector $\mathbf{X}$, with the PDF $\mathcal{N}(\mathbf{0}_N, \mathbf{R})$, where the covariance matrix $\mathbf{R}$ is given by

$$\mathbf{R} = \begin{pmatrix} 1 & \rho_{1,2} & \dots & \rho_{1,N} \\ \rho_{1,2} & 1 & \dots & \rho_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{N,1} & \rho_{N,2} & \dots & 1 \end{pmatrix}.$$

Each individual sensor sample X_n is distributed as $\mathcal{N}(\mathbf{0}_N, 1)$ and the correlation between two sensors, $\rho_{j,k} = cov(X_j, X_k) = e^{-a \cdot d(j,k)}$, where constant a controls the correlation and $d(j, k)$ is the Euclidean distance between the sensors j and k . We arbitrary choose $a = 1.2$ for the simulations.

First, we consider the performance of the joint decoder for the case of multivariate Gaussian distributed sensor observations. The clustering algorithm described in Sec. 4.1 was used to group 9 sensors into 3 clusters $\Gamma_1 = \{I_1, I_3, I_7\}$, $\Gamma_2 = \{I_2, I_4, I_5\}$, $\Gamma_3 = \{I_6, I_8, I_9\}$. The factorization obtained was,

$$\tilde{P}(\mathbf{I}) = P(I_6, I_8, I_9)P(I_1, I_7|I_8)P(I_3|I_1, I_7)P(I_2, I_4|I_8)P(I_5|I_2, I_4). \quad (4.20)$$

A similar result was also by obtained following the KLD method [21]. The expansion (4.20) yields a cycle free factor graph (i.e., FT), so exact marginals could be calculated. Fig. 4.6 shows the RSNR of several decoding schemes as a function of the CSNR. When sensors do not use the knowledge of inter-sensor correlation for distributed quantization, i.e., no binning used, each sensor uses the same rate (=1 bit/sample) for quantization and transmission. Otherwise, the quantization was done at a higher rate ($Q = 2$ or 3 bits/sample), and the transmission rate reduced to 1 bit/sample via binning. Recall that, for binning, only those sensors within each cluster were considered. The optimal binning function to be used at the output of each quantizer has been designed by the SA based procedure described in Sec. 4.3. Table 4.1 presents the optimized binning functions $h_n(\cdot)$ for the multivariate Gaussian source.

Next, we investigate the performance of joint decoding when the source PDF is not Gaussian. The performance of several systems without binning is compared in Fig. 4.7. As in the previous case, each sensor uses a transmission rate of 1 bit/sample. With the same design parameters as in the Gaussian case, the factorization obtained was,

$$\tilde{P}(\mathbf{I}) = P(I_2, I_4, I_9)P(I_1, I_6|I_2)P(I_5, I_8|I_4)P(I_3, I_7|I_4). \quad (4.21)$$

Note that, compared to expansions analyzed in Chapter 3, factorization (4.21) consists

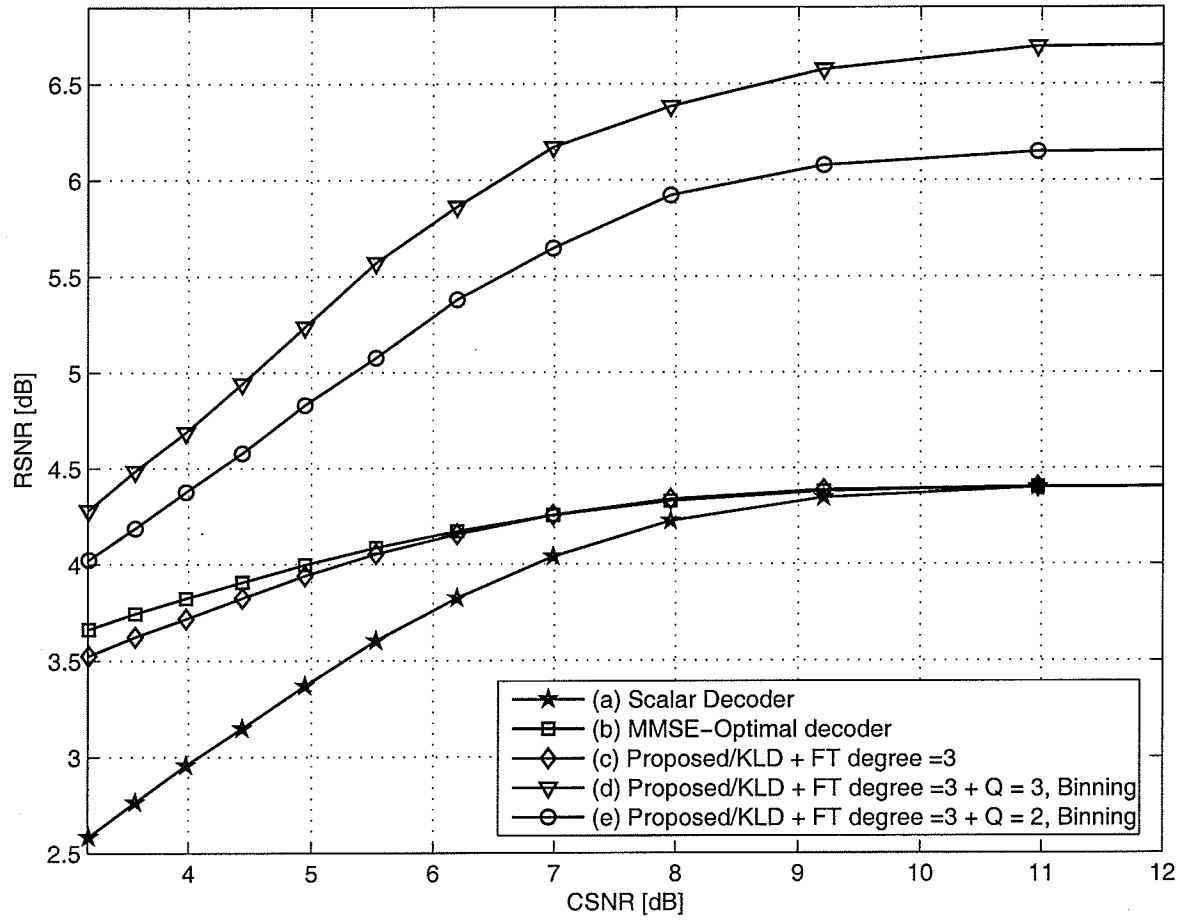


Figure 4.6: Simulation results - Source is distributed as multivariate Gaussian. Scalar quantization with a subsequent binning function at the encoder reduces channel rate to 1-bit/sample.

Table 4.1: Binning functions - Gaussian source, $Q = 3$ bits/sample, $U_n = h_n(I_n)$.

$I_n =$	0	1	2	3	4	5	6	7
U_1	0	0	1	1	1	1	1	0
U_2	0	0	0	1	1	1	1	0
U_3	0	0	0	0	1	1	1	1
U_4	0	0	0	0	1	1	1	1
U_5	0	0	0	0	1	1	1	1
U_6	1	0	0	0	0	0	1	1
U_7	0	0	0	0	1	1	1	1
U_8	0	0	0	0	1	1	1	1
U_9	1	1	1	0	0	0	0	1

of a lesser number of factors. This reduces the computational complexity of the sum-product algorithm. However the end-to-end performance remains nearly optimal.

In order to benefit from binning, each quantizer can use a rate of $Q > 1$. In Fig. 4.8, we consider systems with $Q = 2$ and $Q = 3$ bits/sample. The performance improvement due to the use of distributed quantization is of course clear. But more importantly, the decoders designed with the proposed method perform significantly better than those designed with the KLD-based method. In the case of $Q = 2$ bits/sample, we also include the performance of the optimal decoder for comparison (with $Q = 3$ bits/sample, estimating the optimal decoder performance was computationally infeasible), which shows that the performance of the proposed design is within less than 0.25 dB in the entire CSNR range. In contrast, the KLD-based design performs more than 1.0 dB below at low CSNR.

To apply the proposed algorithm to a fairly large network, a network of 100 sensors was also considered. The sensors are assumed to have non-Gaussian distribution described in Sec. 3.6. The simulation results for this system is depicted in Fig. 4.9. Fig. 4.9 highlights the performance gain which can be achieved by merging the clusters (i.e., using inter-cluster correlation) over separate decoding, as illustrated in

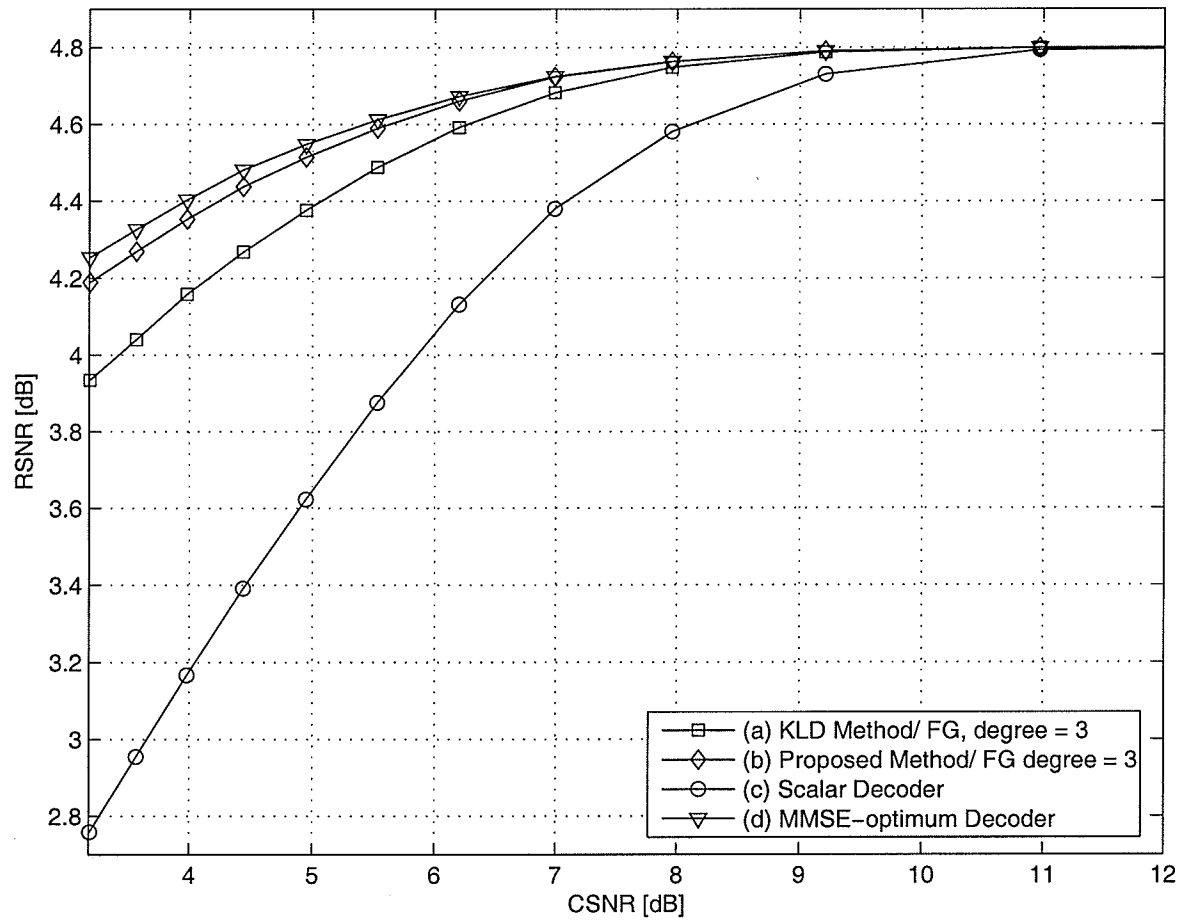


Figure 4.7: Performance of several decoders as a function of CSNR. The network has $N = 9$ sensing nodes and each node uses rate 1-bit/sample quantization, without subsequent binning.

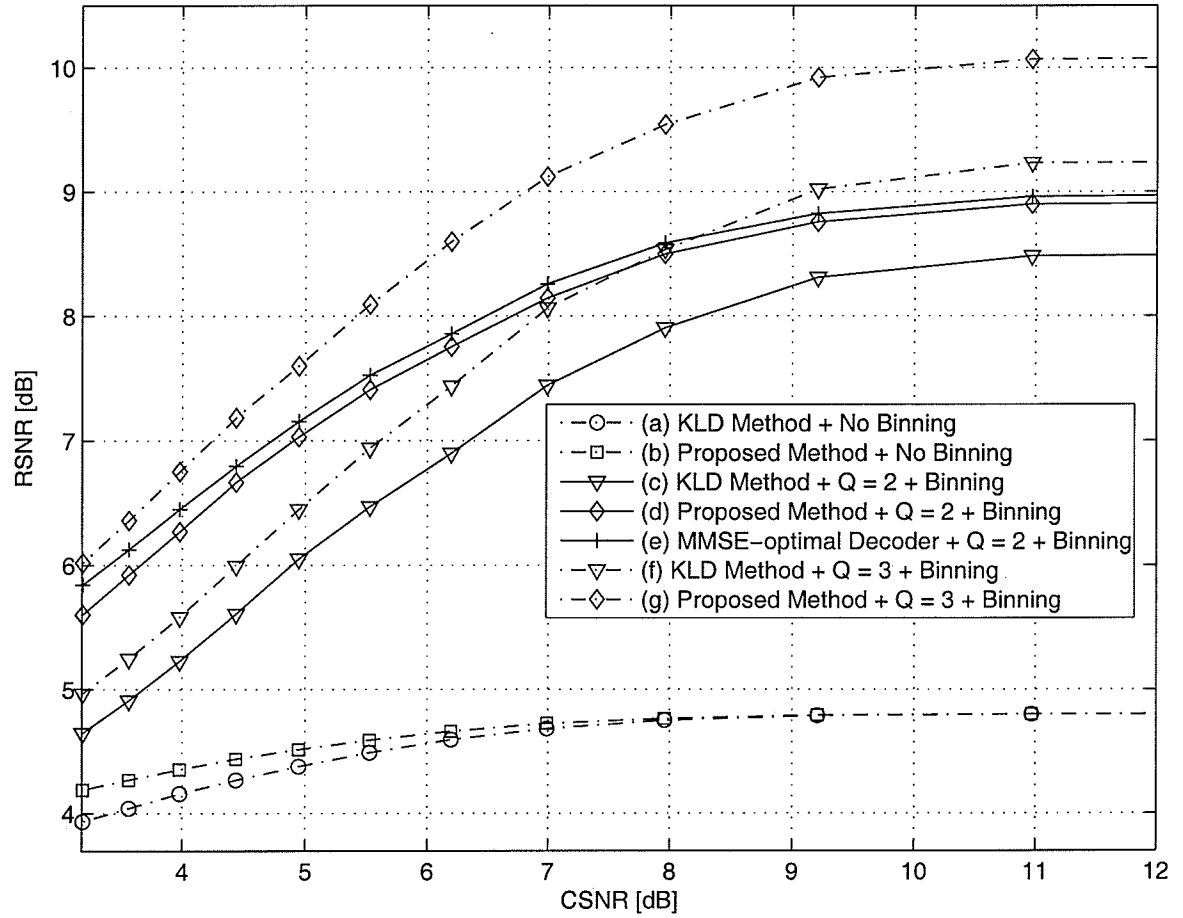


Figure 4.8: Performance of various decoders when binning-based distributed quantizers are used in sensors. Each sensor uses a quantizer rate of Q bits/sample and a transmission rate of 1 bit/sample. The curves (a) and (b) are the same as those in Fig. 4.7. For the system with $Q = 2$, the performance of the optimal decoder is shown in curve (e).

Table 4.2: Binning functions - Non-Gaussian source, $Q = 3$, $U_n = h_n(I_n)$.

$I_n =$	0	1	2	3	4	5	6	7
U_1	0	0	1	1	1	1	0	0
U_2	1	1	0	0	0	0	1	1
U_3	0	0	0	0	0	1	1	1
U_4	0	1	1	1	1	0	0	1
U_5	1	1	0	0	0	0	0	1
U_6	0	0	0	0	1	1	1	1
U_7	1	1	0	0	0	0	0	1
U_8	0	0	0	0	0	1	1	1
U_9	0	0	0	0	1	1	1	1

Fig. 4.3. Design with merged clusters performs more than 0.5 dB higher at 3 dB CSNR, in contrast to the separate decoding, for quantizers with rate 1 bit/sample (see Fig. 4.9 curves (b) and (c)). If we increase the quantizer rate Q and keep the transmission rate at 1 bit/sample, this performance gap seems to widen up further and a 3.5 dB gain could be seen at 3 dB CSNR at $Q = 3$ bits/sample. Observe that, with increasing noise power, the advantage of binning disappears. At low CSNR, the system designed with rates of $Q > 1$ and subsequent binning performs poorly compared to the system designed with rate $Q = 1$ bits/sample and no binning.

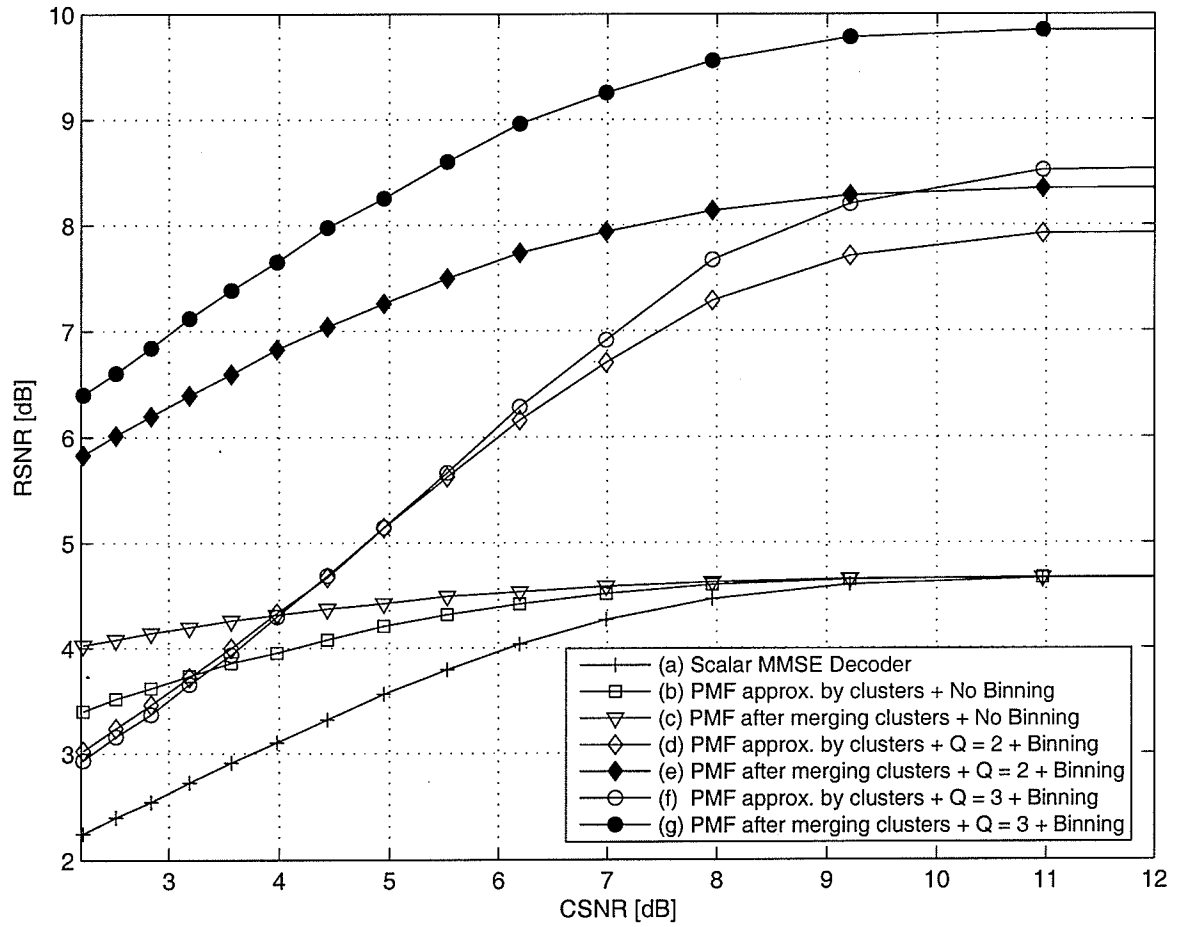


Figure 4.9: Performance comparison of the decoders designed based on PMF approximations in (4.1) and (4.8) with $N = 100$ sensors. Curves (b) and (c) for channel rate 1-bit/sample, without subsequent binning.

Chapter 5

Conclusions and Suggestions for Future work

5.1 Summary and Conclusions

In this thesis, we have investigated the problem of designing a low complex joint decoder suitable for use in a large scale sensor network. Main contributions of this work can be summarized as follows.

- An algorithm for sensor clustering was introduced based on statistical dependence of sensors. It was observed that, use of the Pearsons Chi-square statistics for clustering is very effective and simple, when a training set of sensor observations is given. It was demonstrated that the decoding using the clusters is extremely effective than using the scalar decoding. Distributed quantization (binning) for the complete sensor network is not scalable. Hence we used binning based on the clusters, which is fairly simple to be optimized with the SA approach. Our simulation results reveal that, despite of the simplicity of the encoders (consisting only a scalar quantizer and a binning function), a significant

performance gain can be achieved with proposed decoding approach.

- The algorithm which we introduced in Chapter 4 to construct a factor graph based on a BN is a more general than the KLD approach proposed in [20], where the source distribution has to be necessarily Gaussian. In experiments with Gaussian sources, it was observed that the proposed method yield a system which performs identically to the system obtained by the KLD-based method. More interestingly, we have seen that for non-Gaussian sources, the proposed method can yield systems with significantly better performance compared to the KLD-method. We have observed that merging of the clusters increased the closeness of the approximate PMF to its original, which makes the decoder perform better.

5.2 Future Research Directions

We mainly confined our attention to evaluating the performance of the proposed decoding algorithm for synthetic data sets generated by the computer. A natural extension of this work could be the application of the algorithms developed here to design decoders for real world sensor observations. For example, decoding of correlated data gathered by audio field sensors (microphones) is an interesting experiment.

We have used the directed graphical model - the Bayesian networks, to approximate a PMF. Another important avenue of exploration is to investigate the possibility of using the undirected graphical models such as Markov random fields (MRF) [29, 43].

Factor graphs and the sum-product algorithm play a significant role in the work of this thesis. It is a well known fact that, when applied to factor graphs with cycles, the sum-product algorithm does not always exhibit good convergence properties.

However, in simulations we have seen our systems converge fast and surprisingly well. Therefore, further investigation on the convergence properties of the sum-product algorithm is worthwhile.

In this thesis, we have considered only the spatial correlation of sensor measurements. It would be beneficial to investigate how to exploit the temporal correlation to enhance the decoder performance. Possible further extensions also include evaluation of other methods of binning such as DISCUS [44].

Appendix A

Conditional Mean Estimator

In this appendix, the solution to the conditional mean estimation problem is derived. Let, $\mathbf{X} \in \mathbb{R}^N$ is a random vector and $\hat{\mathbf{I}} \in \Omega^N$ ($\Omega = \{1, 2, \dots, 2^Q\}$) with joint distribution $p(\mathbf{x}, \hat{\mathbf{i}})$. For given $\hat{\mathbf{I}}$, we wish to find an estimate $\hat{\mathbf{X}} = \mathcal{D}(\hat{\mathbf{I}})$ for the vector $\mathbf{X}$, such that the MSE $E\{\|\mathbf{X} - \mathcal{D}(\hat{\mathbf{I}})\|^2\}$ is minimized.

$$\begin{aligned} E\{\|\mathbf{X} - \mathcal{D}(\hat{\mathbf{I}})\|^2\} &= \int_{\mathbb{R}^N} \sum_{\forall \hat{\mathbf{i}} \in \Omega^N} \|\mathbf{x} - \mathcal{D}(\hat{\mathbf{i}})\|^2 p(\mathbf{x}, \hat{\mathbf{i}}) d\mathbf{x} \\ &= \int_{\mathbb{R}^N} \sum_{\forall \hat{\mathbf{i}} \in \Omega^N} \|\mathbf{x} - \mathcal{D}(\hat{\mathbf{i}})\|^2 p(\mathbf{x}|\hat{\mathbf{i}}) P(\hat{\mathbf{i}}) d\mathbf{x}. \end{aligned} \quad (\text{A.1})$$

Note that,

$$E\{\|\mathbf{x} - \mathcal{D}(\hat{\mathbf{i}})\|^2 | \hat{\mathbf{i}}\} = \int_{\mathbb{R}^N} \|\mathbf{x} - \mathcal{D}(\hat{\mathbf{i}})\|^2 p(\mathbf{x}|\hat{\mathbf{i}}) d\mathbf{x}. \quad (\text{A.2})$$

Then, the MSE in (A.1) can be minimized by minimizing the conditional expectation (A.2) for every $\hat{\mathbf{i}}$. i.e.,

$$\mathcal{D}^*(\hat{\mathbf{i}}) = \arg \min_{\mathcal{D}} E\{\|\mathbf{x} - \mathcal{D}(\hat{\mathbf{i}})\|^2 | \hat{\mathbf{i}}\}. \quad (\text{A.3})$$

By taking partial derivation w.r.t. $\mathcal{D}$

$$\begin{aligned}
 \frac{\partial}{\partial \mathcal{D}} E\{\|\mathbf{x} - \mathcal{D}(\hat{\mathbf{i}})\|^2 | \hat{\mathbf{i}}\} &= \mathbf{0} \\
 \Rightarrow -2 \int_{\mathbb{R}^N} (\mathbf{x} - \mathcal{D}^*(\hat{\mathbf{i}})) p(\mathbf{x} | \hat{\mathbf{i}}) d\mathbf{x} &= \mathbf{0} \\
 \Rightarrow \mathcal{D}^*(\hat{\mathbf{i}}) &= E\{\mathbf{X} | \hat{\mathbf{I}} = \hat{\mathbf{i}}\}
 \end{aligned} \tag{A.4}$$

Appendix B

Derivation of (1.6)

Consider the system shown in Fig.B.1. In the presence of transmission errors, reconstructed signal $v(k)$ includes both the *quantization error* $q(k)$ and the *channel error* $c(k)$. We can write,

$$q(k) = x(k) - u(k), \quad (\text{B.1})$$

$$c(k) = u(k) - v(k). \quad (\text{B.2})$$

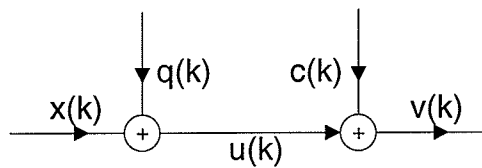


Figure B.1: Transmission of quantized data over a noisy channel.

Then, consider the reconstruction error,

$$\begin{aligned}
E\{(x(k) - v(k))^2\} &= E\{(x(k) - u(k) + c(k))^2\} \\
&= E\{(q(k) + c(k))^2\} \\
&= \underbrace{E\{q(k)^2\}}_{\text{quantization error}} + \underbrace{E\{c(k)^2\}}_{\text{channel error}} + \underbrace{2E\{q(k)c(k)\}}_{\text{mutual error}} \quad (\text{B.3})
\end{aligned}$$

If the transition probability $P_{ij} = P(v(k) = v_j | u(k) = v_i)$, then the mutual error becomes

$$E\{q(k)c(k)\} = \sum_{\forall i} \sum_{\forall j} P_{ij}(v_i - v_j) \int_{R_i} (x - v_i) p(x) dx. \quad (\text{B.4})$$

Now, if v_i s are centroids of the intervals, i.e., quantizers are PDF optimized Lloyd-max quantizers, the intergral part of (B.4) vanishes and the mutual error becomes zero. Then,

$$E\{(x(k) - v(k))^2\} = E\{q(k)^2\} + E\{c(k)^2\}. \quad (\text{B.5})$$

Appendix C

The Sum-product Algorithm

We have used factor graphs and the sum-product algorithm to implement the joint decoder. To see how this works, we will use the expansion given in (2.7). The marginal function of $p(\mathbf{x})$ for x_1 is,

$$p_1(x_1) = \left(\sum_{\sim\{x_1\}} p_2(x_2) p_{1|2}(x_1|x_2) \left(\sum_{\sim\{x_2\}} p_{3|2}(x_3|x_2) \left(\sum_{\sim\{x_3\}} p_{4|3}(x_4|x_3) \right) \right) \right), \quad (\text{C.1})$$

where $\sum_{\sim\{\cdot\}}$ is the summary operation, in which, instead of listing the variables being summed over, we list the variables that are not summed over. When the factorization is presented as a factor graph, the sum-product algorithm can be described as a procedure to calculate the marginals. The algorithm operates by passing messages between the nodes of the factor graph. There are two forms of messages, messages from variable nodes to function ($\mu_{x \rightarrow f}$) nodes and vice-versa ($\mu_{f \rightarrow x}$).

Product rule : At variable node x , take the product of expressions formed at the descendants of x .

$$\mu_{x \rightarrow f}(x) = \prod_{h \in n(x) \setminus f} \mu_{h \rightarrow x}(x), \quad (\text{C.2})$$

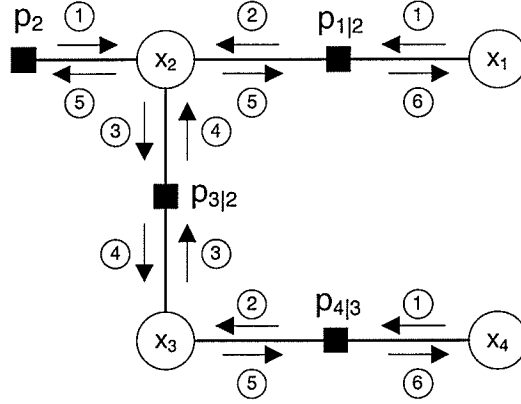


Figure C.1: Generated messages at each step of the sum-product algorithm.

where $n(x)$ are the set of neighbors of x .

Sum-product rule : At a function node f , take the product of with expressions formed at the descendants of f and then perform the not-sum over the parent of f .

$$\mu_{f \rightarrow x}(x) = \sum_{\sim\{x\}} \left(f(X) \prod_{y \in n(f) \setminus x} \mu_{y \rightarrow f}(y) \right), \quad (\text{C.3})$$

where $n(f)$ are the set of neighbors of f .

An illustrative example can be used to clarify this algorithm.

Example : *Fig. C.1 shows the steps of running the sum-product algorithm on the factor graph shown in Fig. 2.1. The messages generated at each steps are as follows.*

Step 1:

$$\begin{aligned}\mu_{x_1 \rightarrow p_{1|2}}(x_1) &= 1 \\ \mu_{x_4 \rightarrow p_{4|3}}(x_4) &= 1 \\ \mu_{p_2 \rightarrow x_2}(x_2) &= \sum_{\sim\{x_2\}} p_2(x_2) = p_2(x_2)\end{aligned}$$

Step 2:

$$\begin{aligned}\mu_{p_{1|2} \rightarrow x_2}(x_2) &= \sum_{\sim\{x_2\}} p_{1|2}(x_1|x_2) \mu_{x_1 \rightarrow p_{1|2}}(x_1) \\ \mu_{p_{4|3} \rightarrow x_3}(x_3) &= \sum_{\sim\{x_3\}} p_{4|3}(x_4|x_3) \mu_{x_4 \rightarrow p_{4|3}}(x_4)\end{aligned}$$

Step 3:

$$\begin{aligned}\mu_{x_3 \rightarrow p_{3|2}}(x_3) &= \mu_{p_{4|3} \rightarrow x_3}(x_3) \\ \mu_{x_2 \rightarrow p_{3|2}}(x_2) &= \mu_{p_2 \rightarrow x_2}(x_2) \cdot \mu_{p_{1|2} \rightarrow x_2}(x_2)\end{aligned}$$

Step 4:

$$\begin{aligned}\mu_{p_{3|2} \rightarrow x_2}(x_2) &= \sum_{\sim\{x_2\}} p_{3|2}(x_3|x_2) \mu_{x_3 \rightarrow p_{3|2}}(x_3) \\ \mu_{p_{3|2} \rightarrow x_3}(x_2) &= \sum_{\sim\{x_3\}} p_{3|2}(x_3|x_2) \mu_{x_2 \rightarrow p_{3|2}}(x_2)\end{aligned}$$

Step 5:

$$\begin{aligned}\mu_{x_2 \rightarrow p_{1|2}}(x_2) &= \mu_{p_{1|2} \rightarrow x_2}(x_2) \cdot \mu_{p_{3|2} \rightarrow x_2}(x_2) \\ \mu_{x_2 \rightarrow p_{1|2}}(x_2) &= \mu_{p_2 \rightarrow x_2}(x_2) \cdot \mu_{p_{3|2} \rightarrow x_2}(x_2) \\ \mu_{x_3 \rightarrow p_{4|3}}(x_3) &= \mu_{p_{3|2} \rightarrow x_3}(x_3)\end{aligned}$$

Step 6:

$$\begin{aligned}\mu_{p_{1|2} \rightarrow x_1}(x_1) &= \sum_{\sim\{x_1\}} p_{1|2}(x_1|x_2) \mu_{x_2 \rightarrow p_{1|2}}(x_2) \\ \mu_{p_{4|3} \rightarrow x_4}(x_4) &= \sum_{\sim\{x_4\}} p_{4|3}(x_4|x_3) \mu_{x_3 \rightarrow p_{4|3}}(x_3)\end{aligned}$$

At the termination of the algorithm marginal functions can be computed using the messages received at corresponding variable node. For example, marginal function $p(x_2)$ would be calculated as

$$p(x_2) = p_2(x_2) \cdot \mu_{p_{1|2} \rightarrow x_2}(x_2) \cdot \mu_{p_{3|2} \rightarrow x_2}(x_2).$$

Table C.1 summarizes number of additions and multiplications required at a node with degree- d with Q level scalar quantization.

Table C.1: Complexity of the sum-product algorithm

Node	Additions	Multiplications
Variable	—	$d(d-2)2^Q$
Function	$d(2^{Qd} - 2^Q)$	$d(d-1)2^{Qd}$

It should be noted that there are no cycles present in the factor graph, i.e., *cycle*

free. We refer such a factor graph as a *factor tree*. In such situations, the sum-product algorithm calculates the marginal functions exactly. However, if the factor graph contains cycles, the sum-product algorithm may not calculate the marginal functions exactly. In such situations, the sum-product algorithm has to be modified to handle factor graphs with cycles.

When there are cycles in factor graph the sum-product algorithm follows the same message passing rules which we have described in (C.2) and (C.3). The major difference here is that, due to the presence of cycles, the algorithm will not naturally terminate - instead it becomes iterative. In this situation, it is not clear where we should initiate the message passing. We can get rid of this difficulty by assuming, that a unit message has been received at each node at every incident edge. With this assumption, it is possible to initiate message passing at any node in the factor graph. An edge e of node f has a pending message, if node f has received a message on any edge other than e . There are variety of message passing schedules available, such as *flooding schedule* and *serial schedule*. The so-called flooding schedule follows a simple rule - in each clock tick every node that has pending messages will send those messages. In serial message passing, at most one message is passed anywhere in the graph, in each clock tick. We have used the flooding schedule in the simulations.

Appendix D

Proof - Likelihood of a Bayesian Network

The following proof is based on the Appendix of [32].

Let the sensor network has N sensors and each sensor consists of a r_n , $n = \{1, 2, \dots, N\}$ bit scalar quantizer. Sensor observations $\mathbf{X} = (X_1, X_2, \dots, X_N)^T$ are quantized by the corresponding quantizers to produce the vector of indices $\mathbf{I} = (I_1, I_2, \dots, I_N)^T$. Then, each I_n is a discrete random variable, where $I_n \in \{1, 2, \dots, 2^{r_n}\}$. As the training set, let there be M source samples. Therefore, the training index set $\mathcal{S}$ is a $M \times N$ matrix.

Definitions

- We denote the parent set of I_n as Ψ_n . For example, in $P(I_1|I_2, I_3)$, the parent set of I_1 is $\Psi_1 = (I_2, I_3)$.
- Let the j^{th} unique instantiation of Ψ_n , in $\mathcal{S}$, be u_{nj} . Suppose there are q_n unique instantiations of Ψ_n . As an example, for 1 bit/sample quantization, instantiations of Ψ_1 are $u_{11} = (1, 1)$, $u_{12} = (1, 2)$, $u_{13} = (2, 1)$ and $u_{14} = (2, 2)$.

Then, $q_1 = 4$.

- L_{njl} denotes the number of samples, in which I_n has value v_{nl} (where $v_{nl} \in \{1, 2, \dots, 2^{r_n}\}$) and the parent, Ψ_n has value u_{nj} .

Let us assume that the sample set $\mathcal{S}$ which is believed to be generated by a Bayesian network $\Lambda_{\mathcal{S}}$ and we are free (without considering $\mathcal{S}$) to assign conditional probabilities $P(I_n|\Psi_n)$. Let the set containing all such $P(I_n|\Psi_n)$ be $\Lambda_{\mathcal{P}}$. Since I_n 's are discrete, we can write

$$\begin{aligned} P(\mathcal{S}, \Lambda_{\mathcal{S}}) &= \int_{\Lambda_{\mathcal{P}}} f(\mathcal{S}|\Lambda_{\mathcal{S}}, \Lambda_{\mathcal{P}}) f(\Lambda_{\mathcal{S}}, \Lambda_{\mathcal{P}}) d\Lambda_{\mathcal{P}} \\ &= \int_{\Lambda_{\mathcal{P}}} f(\mathcal{S}|\Lambda_{\mathcal{S}}, \Lambda_{\mathcal{P}}) f(\Lambda_{\mathcal{P}}|\Lambda_{\mathcal{S}}) f(\Lambda_{\mathcal{S}}) d\Lambda_{\mathcal{P}} \\ &= P(\Lambda_{\mathcal{S}}) \int_{\Lambda_{\mathcal{P}}} P(\mathcal{S}|\Lambda_{\mathcal{S}}, \Lambda_{\mathcal{P}}) f(\Lambda_{\mathcal{P}}|\Lambda_{\mathcal{S}}) d\Lambda_{\mathcal{P}}. \end{aligned} \quad (\text{D.1})$$

We consider a sensor network, in which the samples occur independently. Therefore,

$$P(\mathcal{S}|\Lambda_{\mathcal{S}}, \Lambda_{\mathcal{P}}) = \prod_{m=1}^M P(s_m|\Lambda_{\mathcal{S}}, \Lambda_{\mathcal{P}}), \quad (\text{D.2})$$

where s_m is the m^{th} sample in $\mathcal{S}$. But, we know

$$P(I_1, I_2, \dots, I_N) = \prod_{n=1}^N P(I_n|\Psi_n). \quad (\text{D.3})$$

Then

$$P(s_m|\Lambda_{\mathcal{S}}, \Lambda_{\mathcal{P}}) = \prod_{n=1}^N P(I_n = d_{nm}|\Psi_n = u_n^m, \Lambda_{\mathcal{P}}), \quad (\text{D.4})$$

where d_{nm} denotes the value assignment of index I_n and u_n^m is the value of the parent

Ψ_n of the sample m . Substituting these results in (D.1)

$$P(\mathcal{S}|\Lambda_{\mathcal{S}}) = \int_{\Lambda_{\mathcal{P}}} \left(\prod_{m=1}^M \prod_{n=1}^N P(I_n = d_{nm} | \Psi_n = u_n^m, \Lambda_{\mathcal{P}}) \right) f(\Lambda_{\mathcal{P}} | \Lambda_{\mathcal{S}}) d\Lambda_{\mathcal{P}}. \quad (\text{D.5})$$

Now, instead of considering a sample at a time, we can group the samples having the same value assignments. Then,

$$P(\mathcal{S}|\Lambda_{\mathcal{S}}) = \int_{\Lambda_{\mathcal{P}}} \left(\prod_{n=1}^N \prod_{j=1}^{q_n} \prod_{l=1}^{r_n} P(I_n = v_{nl} | \Psi_n = u_{nj}, \Lambda_{\mathcal{P}})^{L_{njl}} \right) f(\Lambda_{\mathcal{P}} | \Lambda_{\mathcal{S}}) d\Lambda_{\mathcal{P}}. \quad (\text{D.6})$$

Before observing $\mathcal{S}$, we are indifferent regarding which numerical probabilities we have to assign for $P(I_n | \Psi_n)$. i.e., selecting $\Lambda_{\mathcal{P}}$ is arbitrary. Therefore, we choose $f(\Lambda_{\mathcal{P}} | \Lambda_{\mathcal{S}})$ as the uniform distribution.

Let $\Theta_{njl} = P(I_n = v_{nl} | \Psi_n = u_{nj}, \Lambda_{\mathcal{P}})$. Then, Θ_{njl} , $l = 1, 2, \dots, r_n$ is itself a probability distribution, while $f(\Theta_{nj1}, \Theta_{nj2}, \dots, \Theta_{njr_n})$ is uniform. Therefore,

$$\int \dots \int_{\Theta_{njl}} C_{nj} d\Theta_{nj1} \dots d\Theta_{njr_n} = 1, \quad (\text{D.7})$$

where, $f(\Theta_{nj1}, \Theta_{nj2}, \dots, \Theta_{njr_n}) = C_{nj}$. Now, we can write $f(\Lambda_{\mathcal{P}} | \Lambda_{\mathcal{S}})$ as [32]

$$f(\Lambda_{\mathcal{P}} | \Lambda_{\mathcal{S}}) = \prod_{n=1}^N \prod_{j=1}^{q_n} f(\Theta_{nj1}, \Theta_{nj2}, \dots, \Theta_{njr_n}). \quad (\text{D.8})$$

We have used the property of independence between $f(\Theta_{nj1}, \Theta_{nj2}, \dots, \Theta_{njr_n})$ and $f(\Theta_{n'j'1}, \Theta_{n'j'2}, \dots, \Theta_{n'j'r'_n})$ in (D.8). By substituting (D.8) in (D.6)

$$P(\mathcal{S}|\Lambda_{\mathcal{S}}) = \prod_{n=1}^N \prod_{j=1}^{q_n} \int \dots \int_{\Theta_{njl}} \left(\prod_{l=1}^{r_n} \Theta_{njl}^{L_{njl}} \right) f(\Theta_{nj1}, \Theta_{nj2}, \dots, \Theta_{njr_n}) d\Theta_{111} \dots d\Theta_{nq_n r_n} \quad (\text{D.9})$$

By using the knowledge of Dirichlet's distribution, we can deduce

$$\begin{aligned}
\int \dots \int_{\Theta_{njl}} \left(\prod_{l=1}^{r_n} \Theta_{njl}^{L_{njl}} \right) d\Theta_{111} \dots d\Theta_{nq_n r_n} &= \frac{\Gamma(L_{nj1} + 1) \dots \Gamma(L_{njr_n} + 1)}{\Gamma[(L_{nj1} + 1) + \dots + (L_{njr_n} + 1)]} \\
&= \frac{L_{nj1}! \dots L_{njr_n}!}{(L_{nj} + r_n - 1)!} \\
&= \frac{\prod_{l=1}^{r_n} L_{njl}}{(L_{nj} + r_n - 1)!} \tag{D.10}
\end{aligned}$$

where $\Gamma(\cdot)$ is the Gamma function and we use the knowledge, $\Gamma(z) = (z - 1)!$. Note that, $L_{nj} = \sum_{l=1}^{r_n} L_{njl}$. Also, by substituting $L_{njl} = 0$ to (D.10)

$$\int \dots \int_{\Theta_{njl}} d\Theta_{nj1} \dots d\Theta_{njr_n} = \frac{1}{(r_n - 1)!}. \tag{D.11}$$

Then, from (D.7) and (D.11), $C_{nj} = (r_n - 1)!$. By substituting these results to (D.9)

$$P(\mathcal{S}|\Lambda_{\mathcal{S}}) = \prod_{n=1}^N \prod_{j=1}^{q_n} \frac{(r_n - 1)!}{(L_{nj} + r_n - 1)!} \prod_{l=1}^{r_n} L_{njl}!. \tag{D.12}$$

We can use (D.12) to find the likelihood of the training set $\mathcal{S}$, given a Bayesian network $\Lambda_{\mathcal{S}}$. To choose Ψ_n (in D.3), if there is a constraint of the maximum number of elements in Ψ_n , we can use (D.12).

Bibliography

- [1] D. Culler, D. Estrin, and M. Srivasta, "Overview of sensor networks," *IEEE Computer*, vol. 37, pp. 41–49, Aug. 2003.
- [2] J. Porter et.al., "Wireless sensor networks for ecology," *BioScience*, vol. 55, no. 7, pp. 561–572, Jul. 2005.
- [3] K. Chintalapudi et.al., "Monitoring civil structures with a wireless sensor network," *IEEE Internet Computer*, vol. 10, no. 2, pp. 26–34, Mar/Apr. 2006.
- [4] D. Jensen, "SIVAM: Communication, navigation and surveillance for the Amazon," June 2002, [Online] Available: <http://www.aviationtoday.com/av/categories/military/12730.html>.
- [5] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.
- [6] David Salomon, *Data Compression: The Complete Reference*, Springer, 2007.
- [7] Z. Xiong, A. D. Liveris, and S. Cheng, "Distributed source coding for sensor networks," *IEEE Signal Processing Magazine*, pp. 80–94, Sept. 2004.
- [8] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, no. 4, pp. 471–480, Jul. 1973.

- [9] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, Jul./Oct. 1948.
- [10] T. M. Cover and J. A. Thomas, *Elements of information theory*, New York : Willy, 1991.
- [11] S. S. Pradhan and K. Ramachandran, "Distributed source coding using syndromes (DISCUS): Design and construction," *IEEE Transactions on Information Theory*, vol. 49, no. 3, pp. 626–643, Mar. 2003.
- [12] A. D. Wyner, "Recent results in the shannon theory," *IEEE Transactions on Information Theory*, vol. 20, no. 1, pp. 2–10, Jan. 1974.
- [13] S. S. Pradhan and K. Ramachandran, "Distributed source coding: Symmetric rates and applications to sensor network," in *Data Comp. Conf.*, March 2000, pp. 363–372.
- [14] S. D. Servetto, "Lattice quantization with side information," in *Data Comp. Conf.*, March 2000, pp. 510–519.
- [15] A. Aaron and B. Girod, "Compression with side information using turbo codes," in *Data Comp. Conf.*, Apr. 2002, pp. 252–261.
- [16] T. J. Flynn and R. M. Gray, "Encoding of correlated observations," *IEEE Transactions on Information Theory*, vol. 33, no. 6, pp. 773–786, Nov. 1987.
- [17] R. Zhang B. Girod, D. Rebollo-Monedero, "Design of optimal quantizers for distributed source coding," in *Data Comp. Conf.*, March 2003, pp. 13–22.
- [18] X. Wang and M. T. Orchard, "Design of trellis codes for source coding with side information at the decoder," in *Data Comp. Conf.*, March 2001, pp. 361–370.

- [19] J. A. Grubner, "Distributed estimation and quantization," *IEEE Transactions on Information Theory*, vol. 39, no. 4, pp. 1456–1459, Jul. 1993.
- [20] J. Barrós M. Tüchler, "Scalable decoding on factor trees: A practical solution for wireless sensor networks," *IEEE Transactions on Communications*, vol. 54, no. 2, pp. 284–294, Feb. 2006.
- [21] M. Maierbacher and J. Barros, "Source-optimized clustering for distributed source coding," in *Proc. IEEE GLOBECOM*, 2006, pp. 1–5.
- [22] K. Ramachandran S. S. Pradham and R. Koetter, "A constructive approach to districuted source coding with symmetric rates," in *IEEE International Symposium of Information Theory (ISIT)*, 2000, p. 178.
- [23] J. G. Proakis and M. Salehi, *Fundamentals of Communicaiton Systems*, Pearson-Prentice Hall, 2005.
- [24] H. V. Poor, *An Introduction to Signal Detection and Estimation*, Springer Verlag, 1994.
- [25] F. R. Kschischang, B. Frey, and H. A. Leoliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.
- [26] E. T. Jaynes, *Maximum Entropy Formalism*, MIT Press, 1979.
- [27] F. V. Jensen, *An Introduction to Bayesian Networks*, Springer, 1996.
- [28] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufmann Publishers, 1988.

- [29] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*, MIT Press, 1998.
- [30] M. Welling and C. Sutton, "Learning in Markov random fields with contrastive free energies," in *Proc. AISTATS*, 2005.
- [31] C. Chow and C. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, 1968.
- [32] F. F. Cooper and E. Herskovits, "A Bayesian method for the induction of probabilistic networks from data," *Machine Learning*, vol. 9, no. 4, pp. 309–347, Oct. 1992.
- [33] K. Thulasiraman and M. N. S. Swamy, *Graphs: Theory and Algorithms*, New York: Wiley, 1992.
- [34] S. A. Mahmoud W. P. LeBlanc, B. Bhattacharya and V. Chapman, "Efficient search and design procedures for multi-stage VQ of LPC parameters for 4 kb/s speech coding," *IEEE Transactions on Speech and Audio Processing*, vol. 1, no. 4, pp. 373–385, Oct. 1993.
- [35] F. J. Rohlf and R. R. Sokal, *Biometry : the principles and practice of statistics in biological research*, New York : W.H. Freeman, 1995.
- [36] R. A. Fisher, "On the interpretation of χ^2 from contingency tables, and the calculation of P," *Journal of the Royal Statistical Society*, vol. 85, no. 1, pp. 87–94, Jan. 1922.
- [37] L. A. Marascuilo and M. McSweeney, *Nonparametric and distribution-free methods for the social sciences*, Wadsworth Publishing, Belmont, CA, 1977.

- [38] Y. M. M. Bishop, S. E. Feinberg, and P.W. Holland, *Discrete Multivariate Analysis: Theory and Practice*, MIT Press, 1975.
- [39] B. Jaumard A. Guénoche, P. Hansen, "Efficient algorithms for divisive hierarchical clustering with the diameter criterion," *Journal of Classification*, vol. 8, no. 1, pp. 5–30, Jan. 1991.
- [40] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.
- [41] P. Yahampath, "Predictive vector quantizer design for distributed source coding," in *Proc. IEEE ICASSP*, Apr. 2007, pp. 629–632.
- [42] N. Farvardin, "A study of vector quantization for noisy channels," *IEEE Transactions on Information Theory*, vol. 36, no. 4, pp. 799–809, July 1990.
- [43] A. Dogandžić and B. Zhang, "Distributed estimation and detection for sensor networks using hidden Markov random field models," *IEEE Transactions on Signal Processing*, vol. 54, no. 8, pp. 3200–3215, Aug. 2006.
- [44] S. S. Pradhan, J. Kusuma, and K. Ramachandran, "Distributed compression in a dense microsensor network," *IEEE Signal Processing Magazine*, pp. 51–60, March 2002.