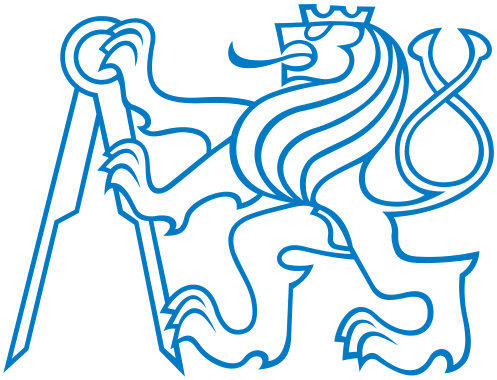
Distributed Iterative Decoding and Processing in Multi-Source Cooperative Networks.

PhD Thesis

Pavel Procházka



Czech Technical University in Prague
Faculty of Electrical Engeneering
Department of Radioelectronics

PhD Programme: Electrical Engineering and Information Technology
Branch of study: Radioelectronics (2601V010)

January, 2014 (A1)

Supervisor: Prof. Ing. Jan Sýkora CSc.

Annotation

The wireless network coding [26, 30, 39, 54] (wireless extension of the network coding [1, 67]) was shown to have a potential to significantly improve the performance in the wireless networks against the conventional networks based on independent (or interfering) point to point channels directed by higher layers. The whole network must be however designed jointly to take advantages from the wireless network coding. The receiver processing then should take into account this global design. This thesis aims to (1) the global design of the channel encoders and corresponding decoders among all nodes within the wireless network and (2) the receiver processing of a particular node in the wireless communication network.

A joint solution of the channel coders design across the network is called the *block structure layered design* [46]. This solution allows to incorporate arbitrary particular channel code rates. The design is based on a *linear* description of the wireless network containing hierarchical maps (network coding) and channel coding. The design combines already known approach suited just for symmetric channel code rates (equal channel encoders) [30] properly utilizing advances of the network coding paradigm [67] in wireless extension. This principle is mixed with traditional joint decoding such that the desired code rates and relay information flows are achieved in average over whole codeword. This mixture results in the *block diagonal* structure using the proposed linear system description. We numerically evaluate the performance for a particular wireless network that is designed by means of the wireless network coding to prove the design properties.

The fundamental topic underlaying the whole thesis is the sum product algorithm on factor graphs [27, 33]. This generic tool enables a construction of the receiver processing for a general class of scenarios including the wireless networks with properly respected global design. The thesis aims to a direct implementation of the sum product algorithm that is applied in the receiver within the wireless networks. Our endeavour results in a generic implementation framework of the sum-product algorithm [44, 47] that is applicable in a wide class of scenarios including the receiver processing in the wireless networks. The framework is based on the proposed KLT message representation. The KLT message representation is capable to methodically describe an arbitrary continuous message in FG-SPA, whenever the stochastic description (even numerical) is available and the KLT of the random process formed by random message realizations can be evaluated. The description is moreover the best one among all linear message representations for a given message dimension as a consequence of the KLT. We stress that the uniqueness of the proposed message representation consists in its *methodical nature* and applicability on the continuous messages (the FG-SPA definition straightforwardly enables only representation of discrete messages). We further derive the update rules upon linear message representations with orthogonal kernels. The KLT message representation satisfies both the linearity and the orthogonality of the kernels and therefore the proposed update rules design is applicable also for the KLT message representation. The KLT message representation jointly with the update rules design therefore form the generic *FG-SPA implementation framework*. The proposed implementation framework is verified in a particular scenario (a receiver aiming with joint phase estimation data detection) and compared with a conventional solution.

Acknowledgement

I would like to thank everybody supporting me and my research in both working and personal live during my studies. Particularly, to my supervisor for his suggestions and ideas that are reflected in this work, mainly in the topic related to Chapter 5, especially the KLT message representation. I would like to thank also the members of DiRaC lab working on similar topic for mutual discussions leading to improve my (hopefully our) knowledge related not only to this thesis. The next acknowledgement is dedicated to the anonymous reviewers of both journal and conference papers, editors and the interacting audience in different venues (conferences, meetings, workshops, etc.) for their often apt comments and suggestions that helped to improve this work.

An inseparable part enabling me to concentrate to the research beyond this thesis is the environment of my personal live, mainly my family and friends. I would like to thank all of them for assisting me to relax and to overcome the desperate whiles resulting from numerous deadlocks that sometimes crossed my research. Namely, I would like to thank my wife Marketa among others for her support and tolerance of my asocial behaviour during long nights before numerous of deadlines.

Last but not the least, the work was supported by the following grants:

- EU COST-IC1004: *Cooperative Radio Communications for Green Smart Environments*, 2011-2015,
- EU COST 2100: *Pervasive Mobile and Ambient Wireless Communications*, 2006-2010,
- FP7-ICT/STREP (INFSO-ICT-248001): *SAPHYRE - Sharing Physical Resources Mechanisms and Implementations for Wireless Networks*, 2010-2012,
- FP7-ICT-2011-8/ICT-2009.1.1: *DIWINE - Dense Cooperative Wireless Cloud Network*, 2013-2015,
- Grant Agency of the Czech Republic (GACR 102/09/1624): *Mobile Radio Communication Systems with Distributed, Cooperative and MIMO processing*, 2009-2012,
- Ministry of Education, Youth and Sports (OC 188): *Signal Processing and Air-Interface Technique for MIMO radio communication systems*, 2007-2010,
- Ministry of Education, Youth and Sports (LD12062): *Wireless Network Coding and processing in Cooperative and Distributed Multi-Terminal and Multi-Node Communication Systems*, 2012-2015,
- Grant Agency of the CTU in Prague: *SGS10/287/OHK3/3T/13*, 2010-2012,
- Grant Agency of the CTU in Prague: *SGS13/083/OHK3/1T/13*, 2013.

The main contribution of this work can be also found in the final (concurrent) reports of the aforementioned grants.

Contents

List of Figures

List of Tables

Nomenclature

$\hat{\mathbb{M}}$	domain of approximated messages in FG-SPA
$\mathbb{R}$	set of real numbers
$\mathbb{C}$	set of complex numbers
$\mathbb{M}$	domain of messages in FG-SPA
$\mathbb{N}$	set of natural numbers
$\mathbb{R}^+$	set of non-negative real numbers
AF	Amplify and Forward
AWGN	Additive White Gaussian Channel
BER	Bit Error Rate
BICM	Bit Interleaved Coded Modulation
CF	Compress and Forward
CPE	Continuous Phase Encoder
DF	Decode and Forward
EXIT	Extrinsic Information Transfer
FBA	Forward backward Algorithm
FG	Factor Graphs
FG-SPA	Sum Product Algorithm on Factor Graphs
FN	Factor Node
FSA	Flood Schedule Algorithm
FSM	Finite State Machine
GF(q)	Galois Field Modulo q
H-SI	Hierarchical-Site Information
HDF	Hierarchical Decode and Forward
iid	independent and identically distributed (random vector)
JDF	Joint Decode and Forward
KLT	Karhunen Loeve Transform (principal component analysis)
LCMR	Linear Canonical Message Representation
LD	Loop Disabled
LDPC	Low Density Parity Check

LE Loop Enabled

LF Likelihood Function

LLF Log-Likelihood Function

LLR Log Likelihood Ratio

LPD Limited Phase Discriminator

MAC Multiple Access Channel

MAP Maximum A Posteriori (probability)

MIMO Multiple Input Multiple Output (channel)

ML Maximum Likelihood

MMSE Minimum Mean Square Error

MPSK M-ary Phase Shift Keying

MSE Mean Square Error

MSK Minimum Shit Keying

NC Network Coding

NMM Non-linear Memoryless Modulator

OSI Open Systems Interconnection (model)

P2P Point to Point (channel)

pdf probability density function

pf probability function

pmf probability mass function

Pr Probability

PS Phase Shift

RV Random Variable (Value)

RW Random Walk (phase shift model)

SD-IC Successive Decoding-Interference Cancellation

SINR Signal to Interference plus Noise Ratio

SISO Soft In Soft Out

SNR Signal to Noise Ratio

SoDeM Soft Output Demodulator

SPA Sum Product Algorithm

TWRC Two-Way Relay Channel

VA Viterbi Algorithm

VN Variable Node

WN Wireless Network

WNC Wireless Network Coding

XDF Whatever Decode and Forward

1 Introduction

1.1 Motivation

The wireless network coding attained a large popularity recently, because it stands for one of the possible flows improving the current state of art of the digital communications. The gaps between the fundamental limits and the current state of art implementations shrank to negligible values for almost all utility metrics in conventional point to point channels. Nevertheless, if we consider a wireless network as a collection of independent point to point channels that are directed by higher layers (conventional OSI model) the situation is no longer optimal from the global point of view. The situation in wireless networks is therefore fundamentally different.

Considering the whole wireless network as one unit to be optimized opens a whitespace for new fundamental limits related to the wireless network communication as well as for novel procedures improving the network performance. The main part of the research community in communication aims to fill this whitespace, which can be proven by majority of such topics in the communication journals and conferences nowadays. A lot of work has been done, especially in simple network models such as two-way relay channel containing two nodes that want to exchange the data using an auxiliary relay. A performance very close (half bit below) to the first order cut-set bound was proven in [38] for the two way relay channel upon AWGN channels. It means that the relay receives a superimposed signal from two sources *simultaneously* and the performance of the multiple access channel is limited almost only by the Gaussian noise, the *interference* limitation is therefore almost avoided.

The proof of the achievability is, however, based on the lattice code approach that does not allow a straightforward implementation. There is therefore need for a directly implementable channel coding strategy allowing to approach the performance close to this limit. Many papers suggest to adopt the conventional channel coding strategies to properly respect the wireless network coding, but only limited results are available so far. Particularly a natural benchmark, i.e. the joint decoding followed by the pure network coding, can be obviously implemented for arbitrary channel code rates in sources, nevertheless the advantage of the wireless network coding is not fully exploited. On the other hand, for a special case, one can compose a compound decoder processing jointly the channel decoding and the network operation. This approach is however limited for a particular class of scenarios, where the equal source rates are assumed. A solution utilizing the advantage of the wireless network coding and allowing arbitrary channel code rates is an obvious target.

Suppose now that we have already described the receiver in a mathematical way. The point is how to properly implement the processing in the receiver. This processing should be efficient to avoid waisting the resources and also it should be as exact as possible. One can feel that this is another crucial step for the system implementation. As an example from the history, we can mention the conventional capacity achieving channel codes that were known (LDPC codes developed by Gallager) for a long time, but their implementation was trackable since the iterative processing was employed. Nevertheless, this took over twenty years.

The sum product algorithm on factor graph [27, 33] was shown as a tool that successfully solves a wide family of problems. The generic nature of the sum product algorithm allows its use also to describe the receiver processing in a wireless network properly taking into account the global design. The sum product algorithm is, however, exactly described in systems leading to a cycle-free factor graph. The capacity achieving channel codes that are supposed to be utilized in the receiver typically lead to the factor graphs containing loops. One should take attention about the system convergence in such a case, which is rather analytical than the implementation problem. Another problem is that the sum product algorithm is described as an iterative algorithm that is directed by some update rules. These update rules provide basically marginalization of a soft information (message) describing a particular local variable in the system. The problem arising from the implementation of the sum product algorithm is therefore given

by a description of a general continuously valued message for the processing purposes and the numerical intensity of the continuous message marginalization (integration) required by the update rules.

1.2 Goals and Summary of Main Results

The goal of this thesis is twofold. The first goal aims to the overall channel codes construction within the network that utilizes the wireless network coding. The second goal is dedicated to the implementation issue of generic sum product algorithm that can be used for the receiver processing design. The generic nature of the sum product algorithm enables to incorporate a various number of flavours resulting from the overall network design.

The answer to the first goal is contained in the block structured layered design describing the network design for an arbitrary channel code rates and simultaneously profiting from the wireless network coding advantage. This contribution is situated in a self-contained form in Chapter 7 as well as in [46]. The block structured layered design is build upon the system linearity assumption, that is linear channel codes and hierarchical functions. The channel codes can be then described by matrices (LDPC codes) that are properly composed within a global matrix in each node. Such a (block) construction indeed allows to incorporate arbitrary source rates as well as an arbitrary information flow division between the hierarchical streams (paths of the mixed information) in the network. To best of our knowledge, no such a design is available in the current state of art literature.

The second goal is an implementation framework for the generic sum product algorithm applicable among others also in the wireless networks. This framework should efficiently and *methodically* solve the message representation as well as an efficient implementation of the update rules upon the given message parameterization. We therefore formally describe the problem of the message representation in Chapter 4. The solution is then situated into the sequel Chapter 5, where we first serve a generic KLT message representation that is based on the stochastic description of the message. The message representation (approximation) obtaining is both *methodical* and *efficient*, where the optimized criterion is the MSE of the approximated message (Sec. 5.2.1 or [44]). The update rules upon the message representation for a special type of linear canonical message representation with orthogonal kernels are then proposed in Section 5.2.3 or [47]. An example demonstrating the power of the proposed implementation framework can be found in Section 5.3.

1.3 Structure of the Thesis

The thesis is divided into three main parts, where the core of the work is situated into the second and third part. The main contribution is contained in Chapters 5 (also in [44,47]) and 7 (also in [46] and partly in [58]). These Chapters are written such that they are supposed to be self-contained. It means that the rest of the thesis is some kind of supplementary material for these two main Chapters.

The first part situated immediately after introduction contains a collection of prerequisites that will be further used in the sequel parts. A common characteristic of the first part is that the prerequisites are supposed to be presented just for the completeness. The first part is therefore not written in the tutorial form at all. Instead, some basic ideas and definitions are discussed with reference to the current state of art. We often point out the relation of a particular topic with the contribution that is situated into the sequel parts.

The second part contains topic related to the implementation of the sum product algorithm. We formally state the problem of the implementation framework and then we introduce the proposed implementation framework containing the KLT message representation and the generic update rules design.

The third part is dedicated to the channel code design across the wireless network properly respecting the global design requests (wireless aware network coding). This is presented after a short introduction to

the wireless network coding.

The contents of the individual chapters can be summarized as:

- **Part I** contains the prerequisites for the sequel part.
  - **Chapter** 2 provides a collection of some basic terms that form some fundamental limits (channel capacity) and starting points for the receiver algorithm derivation (maximum a posteriory estimation criterion).
  - **Chapter** 3 briefly lists conventional channel coding and decoding techniques that approach the channel capacity. Namely the turbo and LDPC codes. The decoding algorithms are also discussed including the tools dedicated for the investigation of the decoder convergence properties (EXIT charts).
- **Part II** presents a solution to the first goal, that is the generic implementation framework.
  - **Chapter** 4 is supposed to formally define the sum product algorithm on factor graph as well as what does it exactly mean the sum product algorithm implementation framework. Although the implementation framework can be explained by two sentences, we found a formalism underlaying the message representation (approximation) and update rules upon the message representation forming the implementation framework together. Some examples of implementation frameworks are considered for an illustration. For the sake of completeness, we then also discuss the issues of the sum product algorithm resulting from its use on the looped factor graphs.
  - **Chapter** 5 presents one of the main contribution of this work, that is the implementation framework based on the KLT message representation. This message representation is introduced as a message representation solution methodically minimizing the MSE of the message approximation for a given message dimensionality. The generic KLT message representation is supplemented by the generic update rules design that is applicable for a wide class of messages (among others the KLT message representation). The proposed framework is numerically verified and compared with a conventional solution.
- **Part III** covers the topic of the error protection in the wireless network operating with the wireless network coding.
  - **Chapter** 6 offers a basic overview of the wireless network coding and its basic classification. Some basic assumptions and constraints are discussed. The fundamental principles characterizing the wireless network coding are demonstrated when the basic network topologies are presented.
  - **Chapter** 7 serves the second main contribution of this work. This chapter basically corresponds to [46]. We describe both hierarchical operation and channel codes by a linear operation. The block structured layered design is then introduced upon this description. The block structured layered design enables to design jointly the hierarchical functions and the channel codes such that arbitrary desired rates can be incorporated. The principle is finally verified in a concrete wireless network example.

1.4 Thesis Organization

The aim of this Section is to guide through some common rules that are used within this thesis (e.g. some global notation). We try to avoid overloading of all symbols (at least locally).

Vector Notation and Operations

A vector is denoted by a bold-face lower-case letter and it is implicitly assumed to be in a column form. It means that a vector $[b_1, \dots, b_k]^T$ is denoted $\mathbf{b}$. A zero vector with a single non-zero unit value on the ith position is denoted by $\mathbf{e}_i$, for example $\mathbf{e}_3$ is a vector given by $[0, 0, 1, 0, \dots]^T$. If the order of a particular symbol in the vector is not important or clear from the context, the subscript is dropped, i.e. b instead of b_i.

The vector can be also written in the form $\boldsymbol{b}_A$, where A is an ordered set or vector of indexes, e.g. $A = \{1, 2, 5\}$ gives $\boldsymbol{b}_A = \boldsymbol{b}_{1,2,5} = [b_1, b_2, b_5]^T$. We further allow to remove a particular dimension by the tilde operation[1] $\sim$. As an example, we consider a vector $\boldsymbol{b} = [b_1, b_2, b_3]^T$. The vector $\boldsymbol{b} \sim b_2$ is therefore given by $\boldsymbol{b} \sim b_2 = [b_1, b_3]^T$.

Matrix Notation and Operations

A matrix is conventionally denoted by a bold upper-case letter. For example $\mathbf{X}_{k\times n} = [\mathbf{x}_1 \dots \mathbf{x}_n]$ is $k \times n$ matrix composed from n vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$, where each of them is k symbols long. If the matrix is square, its dimension is denoted by a single index, that is $\mathbf{X}_{k\times k} = \mathbf{X}_k$. The matrix dimension in the subscript is omitted whenever the matrix dimension is clear from the context. A zero matrix $\mathbf{0}$ is a matrix containing all zero elements and a matrix $\mathbf{E}$ denotes the unitary matrix that can be written as $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \dots]$. A matrix $\mathbf{X}_{k\times n}$ is said *full-rank* if $\mathrm{rank}(\mathbf{X}) = \min(k, n)$.

We denote $\mathbb{R}$ to be a set of real numbers. An n-dimensional set A^n is denoted by A^*, whenever the number of dimension is not important for that time. We further implicitly assume a symbol-wise operations upon given sets excepting conventional vector and matrix multiplication.

Random Variables and Density Functions

The random variables and random processes [42] are widely used within this thesis. We assume a random variable X described by its probability function (either pdf or pmf) $p_X(x) = \partial \Pr(X < x)/\partial x$. This is denoted just by $p(x)$ for simplicity if it is clear which probability function is meant. The conditional probability $p_{x|y}(x, y) = \partial \Pr(X < x \cap Y < y)/\partial x \Pr(Y < y)$ will be simply denoted by $p(x|y)$. We denote a random realization of the random process in subscript, i.e. $\mu_x(t)$ as a random process of a variable t. We will consider basic moment operations upon the random variables and process. Particularly mean value given by $\mathrm{E_x}[\cdot] = \int \cdot \mathrm{p(x)dx}$ for a real domain of x. Note that the mean value is sometimes denoted by an over-bar symbol or by m letter, that is $\mathrm{E_x}[\cdot] = \mathrm{m.} = \bar{\cdot}$. Similarly the variance is given by the second moment, i.e. $\mathrm{var}[x] = \mathrm{E_x}[(\mathrm{x} - \mathrm{E_x[x]})^2]$. The variance is sometimes denoted by σ letter, i.e. $\mathrm{var}[x] = \sigma_x$. The most notable distribution used within this thesis is the Gaussian distribution given by

$$\rho_\sigma(y - x) = \frac{1}{\sigma^2 \pi} \exp\left(\frac{-|y - x|^2}{\sigma^2}\right), \tag{1.1}$$

for complex random variable and

$$\rho_\sigma(y - x) = \frac{1}{\sqrt{2\sigma^2 \pi}} \exp\left(\frac{-|y - x|^2}{2\sigma^2}\right), \tag{1.2}$$

for real random variable.

[1] The tilde operation is also used for the proportionality relation. The concrete meaning of a given tilde operation can be recognized from the context within this thesis.

Part I

Background

2 Theoretical Background

This chapter aims with the theories essentially affecting the decoding procedure in wireless communication. We will consider the information and estimation theory. Although some results from other fields can be also applied, we focus just to these. This Chapter is dedicated as a reference and therefore it is not written in the tutorial form. We point out some important definitions and theorems that will be used in later part of the work. We refer excellent tutorials [11] for the information theory and [24] for the estimation theory.

2.1 Information Theory

The information theory answers some principal questions from the communication design, mainly regarding the fundamental limits that can be used as a benchmark for a real application, i.e. the gap between the fundamental limit and the real application refers to the space for an improvement for a given realization.

The fundamental limits are ordinarily connected with some utility metric. These limits state the best possible performance of the system from a given metric point of view for given constraints. The main utility metric that is considered within this work is the *channel capacity* in a power constraint wireless channel, because it answers an essential natural question, how many information can be reliably transmitted through a noisy channel for a given transmit power. We also touch the latency issue, because the capacity optimization leads to processing of long sequences unavoidably causing the latency.

2.1.1 Capacity of Scalar Point to Point AWGN Channel

The channel capacity is given by the stochastic description of the channel. The most important channel for our consideration is the Additive White Gaussian Channel (AWGN) described by the Gaussian conditional pdf, that is

$$p(y|x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-|y-x|^2}{2\sigma^2}\right) = \rho_\sigma(y-x),\ x, y \in \mathbb{R}. \tag{2.1}$$

The power constraint channel capacity for such a channel is given by

$$\mathrm{C}_{AWGN} = 1/2 \log(1+\gamma), \tag{2.2}$$

where $\gamma = P/N$ is the signal to noise ratio and the unit is bit per transmission in case of binary base of the logarithm, which is implicitly assumed. The capacity of complex channel, i.e. $x, y \in \mathbb{R}$ becomes to $\mathrm{C}_{AWGN} = \log(1+\gamma)$.

The channel capacity for AWGN is shown to be achievable in [11, Theorem 9.1.1]. The proof of the capacity theorem also reveals how the codeword should look like to attain the capacity. Particularly, the distribution $p(x)$ maximizing the mutual information $\mathrm{I}(X,Y)$ is the *Gaussian* one. The channel code approaching the channel capacity moreover relies on the asymptotic arguments and therefore the channel codeword is expected to be *infinitely long* to achieve the channel capacity.

Practically, the codeword cannot be infinitely long, so one must choose some trade-off between the codeword sequence length and the gap from the channel capacity. The processing of a long sequence brings two issues: (1) a time required for the long sequence processing unavoidably brings *latency* and (2) one need to design a channel code which is implementable with a reasonable complexity (e.g. linear with respect to the codeword length) and that approaches the channel capacity. The random coding used in [11, Theorem 9.1.1] cannot be directly implemented due to its exponential complexity with respect to the codeword length. We dedicate Chapter 3 as an overview of the conventional coding strategies including their implementation in P2P channels.

2.1.2 Capacity of Multiple Access AWGN Channel

If more nodes transmit simultaneously to one receiver, the situation is referred as a Multiple Access Channel (MAC). The receiver hears a superposition of the signals from more nodes corrupted by a noise. This can be described by a conditional pdf $p(y|x_1, \dots, x_n)$. We assume that no other[1] information about the particular streams is available in the receiver (the information is passed only through the channel) for this moment. We also assume that the receiver wishes to fully decode all of the received signals.

The channel capacity (2.2) does not refer only to a single rate, but now it is given by the rates of all transmitting nodes. We therefore speak about the capacity region, which is given by a union of all achievable rates for a given MAC channel [11].

In special case of two sources (operating at rates R_1 and R_2), the capacity region (complex channels assumed) is given by $R_1 \leq \log(1 + P_1/N)$ and $R_2 \leq \log(1 + P_2/N)$, where P_1 and P_2 stand for the power of individual signals. The *achievable* single rate cannot be larger than in case, where the second signal is not present . This is referred as the *first order cut-set bound* of the classical MAC channel. The *second order cut-set bound* is given by $R_1 + R_2 \leq \log(1 + (P_1 + P_2)/N)$ and if refers to the joint decoding, i.e. full decoding of data from both sources.

Note 2.1 (Moving Beyond the Second Order Cut Set Bound in More Complex Networks)**.** We will discuss the wireless network coding in sequel part of this work. This strategy is able to operate beyond the second order cut-set bound thanks to avoiding of the full decoding in auxiliary relays. Instead a function of the pair is decoded and one relies that a complement of this function is delivered by an alternative path into the final destination. The final destination then solves a system of equations.

This capacity region can be achieved e.g. by a Successive Decoding-Interference Cancellation (SD-IC) method, where the receiver decodes one of the signal threating the latter as noise and after decoding, this signal is re-encoded and subtracted from the superimposed signal. In case of error-less decoding of the first stream, the subtracted signal is equal to the ordinary P2P channel as one can find e.g. in [11].

2.2 Estimation Theory

Estimation and detection theory stands for one of the fundamental cornerstones for the data communication. This theory provides some criteria of a communication receiver design optimality. A nice tutorial for the estimation theory can be found e.g. in [24].

In this section, we briefly point out the most important criteria (ML, MAP, MMSE estimation) for the receiver design within the wireless communication.

2.2.1 Problem Statement

Assume a random parameter $\boldsymbol{\xi} = [\dots, \xi_i, \dots]$ to be estimated (e.g. data vector, phase shift, etc.) that is observed through a (noisy) observation $\boldsymbol{\theta} = [\dots, \theta_i, \dots]$. A system description defines a joint pdf $p(\xi_i, \boldsymbol{\theta})$, $\forall i$.

We can now consider several approaches of the desired vector $\boldsymbol{\xi}$ estimation. We will distinguish *symbol* estimation $\hat{\xi}_i$ and the whole *vector* estimation $\hat{\boldsymbol{\xi}}$.

[1] Breaking this assumption opens a way to the wireless network coding significantly favouring this work. The wireless network coding will be discussed in Chapters 6 and 7

2.2.2 Maximum Likelihood Estimator

The likelihood function (LF) is defined[2] by

$$\Lambda(\boldsymbol{\xi}) = p(\boldsymbol{\theta}|\boldsymbol{\xi}). \tag{2.3}$$

The maximization of (2.3), i.e.

$$\hat{\boldsymbol{\xi}} = \arg\max_{\check{\xi}} p(\boldsymbol{\theta}|\check{\boldsymbol{\xi}}) \tag{2.4}$$

is called maximum likelihood (ML) estimator. From the monotonicity of the logarithm operation, the maximization of log-likelihood function (LLF) $\Lambda^{'}(\boldsymbol{\xi}) = \ln \Lambda(\boldsymbol{\xi})$ in (2.4) instead of the LF leads to the same result.

Note 2.2 (Optimality of ML Estimator)**.** One can show that [24, Theorem 7.3] the estimation is asymptotically minimal variance unbiased (MVU), that is the estimated vector $\hat{\boldsymbol{\xi}}$ asymptotically obeys a Gaussian distribution $\rho(\boldsymbol{\xi}, I^{-1}(\boldsymbol{\xi})))$, where the variance matrix is given by the inverse of the Fisher $n \times n$ information matrix given by

$$-\mathrm{E}\left[\frac{\partial^2 \ln \Lambda(\boldsymbol{\xi})}{\partial \xi_\mathrm{i} \partial \xi_\mathrm{j}}\right], \forall \mathrm{i}, \mathrm{j}. \tag{2.5}$$

The ML estimator therefore asymptotically achieves the Crammer-Rao bound [24].

2.2.3 Bayesian Estimators

An another branch of estimators is based on a Bayesian approach. According to nature of the estimated parameter $\boldsymbol{\xi}$, a proper type of the estimator is chosen. The general Bayesian estimator is defined by a cost function $C(\epsilon)$, where $\epsilon = \xi_i - \hat{\xi}_i$ denotes an error in the estimation. For our purposes, we consider

- hit or miss cost function defined by

$$C(\epsilon) = \begin{cases} 0 & |\epsilon| \leq \delta \\ 1 & |\epsilon| > \delta \end{cases}, \tag{2.6}$$

  where δ defines a threshold,

- vector hit or miss function defined by

$$C(\boldsymbol{\epsilon}) = \begin{cases} 0 & \|\boldsymbol{\epsilon}\| \leq \delta \\ 1 & \|\boldsymbol{\epsilon}\| > \delta \end{cases}, \tag{2.7}$$

  where $\boldsymbol{\epsilon} = \boldsymbol{\xi} - \hat{\boldsymbol{\xi}}$ and $\|\boldsymbol{\epsilon}\|^2 = \sum_i \epsilon_i^2$ and

- mean square error cost function defined by

$$C(\epsilon) = \epsilon^2. \tag{2.8}$$

A minimization of the Bayes risk given by

$$\mathcal{R} = \mathrm{E}[\mathrm{C}(\epsilon)] = \int\int \mathrm{C}(\xi_\mathrm{i} - \hat{\xi}_\mathrm{i})\mathrm{p}(\xi_\mathrm{i}, \boldsymbol{\theta})\mathrm{d}\xi_\mathrm{i}\mathrm{d}\boldsymbol{\theta}$$

for a given observation $\boldsymbol{\theta}$ provides [24]:

[2] One can consider a symbol-LF $\Lambda(\xi_i) = p(\boldsymbol{\theta}|\xi_i)$.

- Minimal mean square error (MMSE) estimation for the cost function (2.8) is given by

$$\hat{\xi}_i = \int \xi_i p(\xi_i|\boldsymbol{\theta}) d\xi_i = \mathrm{E}[\xi_\mathrm{i}|\boldsymbol{\theta}], \tag{2.9}$$

  where the vector $\boldsymbol{\theta}$ may contain additionally the whole vector $\boldsymbol{\xi}$ excepting ξ_i. Note that the vector form of the MMSE estimator, that is

$$\hat{\boldsymbol{\xi}} = \mathrm{E}[\boldsymbol{\xi}|\boldsymbol{\theta}]$$

  minimizes the MSE of each component $\hat{\xi}_i$. The MMSE criterion is commonly used e.g. in the phase estimation.

- Maximal a posterior probability (MAP) estimation for the cost function (2.6) is given by

$$\hat{\xi}_i = \arg\max_{\xi_i} p(\xi_i|\boldsymbol{\theta}). \tag{2.10}$$

  The MAP criterion is suitable to use for the data detection, because it is equivalent to the minimal average bit error rate (BER) criterion standing for a natural goal of the reliable communication. A considerable part of this work aims with the evaluation of $p(\xi_i, \boldsymbol{\theta})$ serving for the MAP estimation (2.10).

- Vector maximal a posterior probability estimation for the cost function (2.7) is given by

$$\hat{\boldsymbol{\xi}} = \arg\max_{\boldsymbol{\xi}} p(\boldsymbol{\xi}|\boldsymbol{\theta}). \tag{2.11}$$

  Note that the vector MAP estimation does not generally give the same result as the classical MAP estimator (2.10).

If a priory distribution of ξ_i is uniform (as data in the digital communication are conventionally assumed), the MAP estimator is equal to the ML estimator. As a consequence, the likelihood metric is a *sufficient statistics* for the minimal average BER criterion for a given observation.

3 Error Protection Techniques

This Chapter shortly reviews the channel coding techniques with accent to the decoding processing and the investigation of the decoding behaviour. Since the core of this work is basically the decoding processing, the basic coding theory forms one of the prerequisites for the sequel part serving the main contribution of this work.

The main purpose of the channel code is a protection of the data for a reliable transmission through an unreliable channel (wireless medium). The fundamental limits are given by means of the information theory, mainly the channel capacity. This channel code design (Sec. 3.1) aims to approach the channel capacity. The receiver then needs to properly incorporate the received signal taking into account the considered channel encoder in the source. A direct application of the optimal solution leads to the *intrackable* implementation. We therefore focus on some advanced techniques (iterative detection) incorporating a slightly suboptimal, but *trackable*, approach to the code design and the corresponding receiver processing in Sec. 3.2. The suboptimality (convergence) investigation techniques are the topic of Sec. 3.3.

Throughout this Chapter, we will consider a *memoryless only* channel, that is the channel described by a conditional pdf $p(\boldsymbol{x}|\boldsymbol{c}) = \prod_i p(x_i|c_i)$. All memory is therefore situated in the channel code.

3.1 Encoding Techniques

We briefly list several encoding techniques attempting to approach the channel capacity (e.g. [49]). We will consider exclusively *linear* codes. As it was shown in the lattice code case [17], the linearity of the code still enables to achieve the channel capacity.

3.1.1 FSM/Block Codes

The block and FSM based codes are the oldest ways of the channel coding techniques. The block code is described by a matrix $\mathbf{G}$ such that the codeword is given by $\boldsymbol{c} = \boldsymbol{G}\boldsymbol{d}$, where $\boldsymbol{d}$ is a data vector and $\boldsymbol{c}$ is a codeword vector. The rate of this code is given by ratio of dimension of the matrix $\boldsymbol{G}$ in case of full-rank generator matrix.

The trellis (FSM) based codes [51] are described by a FSM. We will use the conventional octal notation to identify the code. The analysis of the FSM-based codes from the performance point of view can be found e.g. in [51].

Since the instant (ith) codeword symbol is given by the data vector up to the ith symbol, the trellis codes are *causal* in contrast to the block codes, where one needs a whole data vector for obtaining the instant codeword symbol in general[1].

The block code can be also described by a FSM, but both output and state functions are generally *time variant* in the corresponding FSM.

3.1.2 Turbo Codes

In contrast with the FSM/block codes described above having a significant gap from the channel capacity, the turbo codes [6,61] attain the performance very close to the capacity. The main idea is in concatenation of FSM encoders separated by an interleaver to reduce the probability of terminating sequences for the recursive FSM code and therefore to almost meet the assumption of the infinite codeword length. The concatenation can be considered either parallel or serial.

We will again not investigate the performance of the turbo coding here, since it is not directly related with the core of this work. Instead we refer some literature containing the details on the turbo codes including the performance analysis, e.g. [51, 61]. We note here, that the "turbo" principle is quite more

[1] Using a systematic form of the code, the whole data vector is needed only for the parity part of the codeword.

general. The "turbo principle" basically means two correlated entities (e.g. FSM), that are able to somehow exchange a randomized (interleaved) soft information such that the information input improve the information output for each entity. One can find some other systems, where the turbo principle is applied (e.g. MIMO equalization [60] or decoding of correlative sources [3]).

The decoding of the turbo code is based on an iterative algorithm that will be described later. Note that the use of the interleaver corrupts the *causality* of the overall encoding process.

3.1.3 LDPC Codes

The LDPC codes [14] are also popular easy to implement encoding technique that attains a close-capacity performance. The LDPC code is basically a block code described by a large dense parity check matrix $\mathbf{H}$ such that $\mathbf{HG} = \mathbf{0}$. Due to the matrix description, the LDPC code is linear and it will form the basic building block in Chapter 7.

The LDPC decoding is a nice example of the FG-SPA implementation that will be discussed in Example 4.13. Although the FG-SPA can be directly applied for the LDPC decoding, it is much more efficient to employ an implementation framework, where the message is represented by soft bit and the corresponding update rules are shown e.g. in [23].

Note 3.1 (Enconding Procedure)**.** The encoding of the LDPC codes is not straightforward, because they are defined using the parity check matrix. One can obtain the corresponding generator matrix directly using the Gaussian elimination from the parity check matrix. The problem is in the numerical intensity for large matrices. We refer [49, Appendix A] for an efficient way of the LDPC encoding.

Note 3.2 (Parity Check Matrix Obtaining)**.** Design of good parity check matrix is a nutshell, because the matrix should be very large to respect the very long codeword assumption for the capacity achieving code. Particularly the properties of the LDPC code are given by degree distribution as well as by the girth, that is the length of the shortest loop in the corresponding graph [49]. The size of the matrix makes numerically intensive to provide some additional loops removing algorithms from a given code as suggested e.g. in [34].

3.1.4 Other Encoding Techniques

One can find much more channel codes with various utility targets (channel capacity, latency). This is however not directly related with this thesis. We therefore point out mainly the lattice codes [17] from the other encoding techniques. This family of codes is very nice tool used mainly in the information theory. The problem of the lattice codes is that their direct implementation is not straightforward and therefore they serve rather as a tool proving some performance than a manual for a direct implementation.

3.2 Receiver Processing

The receiver is assumed to obtain a noisy observation of the encoded vector. It aims to recover the original data [25,43]. First a (symbol-wise) metric is evaluated, since we assume a memoryless channel. This metric is served to the decoder. We will shortly discuss some fundamental algorithms capable to provide the MAP estimation (or some approximation) for the aforementioned encoding procedures [9].

3.2.1 FSM Decoding - Viterbi Algorithm

The Viterbi Algorithm (VA) [9] is a well known algorithm providing the *vector*-MAP estimation of the FSM-coded data vector in with a linear complexity.

The algorithm optimizes a metric across an underlaying FSM trellis in the forward part keeping surviving paths active, that is a hard decision about a path in the trellis according to the metric increments

is performed. In the backward part, one resulting path is selected according to the overall accumulated optimized metric.

We start with the *vector* MAP description to derive a metric for the VA. The causality of the FSM coding and ability to be described by the trellis [18] gives

$$\begin{aligned} p(\boldsymbol{d}|\boldsymbol{x}) &\sim p(\boldsymbol{x}|\boldsymbol{d})p(\boldsymbol{d}) && (3.1)\\ &= \prod_i p(x_i|\boldsymbol{x}_{0,\dots,i-1},\boldsymbol{d})p(d_i) && (3.2)\\ &= \prod_i p(x_i|\boldsymbol{x}_{0,\dots,i-1},\boldsymbol{d}_{0,\dots,i})p(d_i) && (3.3)\\ &= \prod_i p(x_i|\boldsymbol{x}_{0,\dots,i-1},\varsigma_{i-1},d_i)p(d_i), && (3.4) \end{aligned}$$

where the causality was used in (3.3) and the FSM description in (3.4). Using the memoryless property of the channel, we finally obtain

$$p(\boldsymbol{d}|\boldsymbol{x}) \sim \prod_i p(x_i|\varsigma_{i-1},d_i)p(d_i) \tag{3.5}$$

The trellis transition is unambiguously given by a pair $T_i = (\varsigma_{i-1}, d_i)$ and the probability of the transition with respect to a given received vector $\boldsymbol{x}$ refers to (3.5), namely

$$\gamma(T_i) = p(x_i|\varsigma_{i-1},d_i)p(d_i) \tag{3.6}$$

for memoryless channel. The term $\gamma(T_i)$ can be viewed as an increment of the metric for a corresponding transition T_i since $p(\boldsymbol{d}|\boldsymbol{x}) \sim \prod_i \gamma(T_i)$.

Thanks to the monotonicity of the logarithm operation, the equation (3.4) can be also written as

$$\log(p(\boldsymbol{d}|\boldsymbol{x})) \sim \sum_i \log(\gamma(T_i)) \tag{3.7}$$

and the corresponding vector MAP estimation can be written in logarithm form as $\hat{\boldsymbol{d}} = \arg\max_{\boldsymbol{d}} \log(p(\boldsymbol{d}|\boldsymbol{x}))$. The VA now presents an efficient way of this maximization. The VA relies on a fact that each possible data vector corresponds unambiguously with a path in the whole trellis. The main simplification is that one is interested only is the path with the *maximal* probability leading to a given state in the ith step. All of the other pats can be trashed, because they surely does not refer to the part of data vector maximizing the vector MAP estimation.

We define a metric $\rho_i \in \mathbb{R}$ as an accumulated metric in the ith state and $\Delta\rho$ as the metric increment referred to a transition T_i, e.g. $\Delta\rho(T_i) \to \gamma(T_i)$ and $\rho_i = \prod_{j=1}^{i} \gamma(T_j)$. The accumulated metric ρ_i corresponds to a certain path [2] in the trellis, i.e. $\{T_j\}_{j=0}^{i}$. The Viterbi Algorithm is then given by:

1. initialization: set the worst possible metric to all states excepting the initial one ς_0
2. the forward part of the VA also know as add-compare select unit: In each possible state ς_i provide the following:
   (a) assuming that the metric is already evaluated in the $i-1$th step, evaluate the metric in the next state by $\rho_i = \Delta\rho(T_i)(\text{add})\rho_{i-1}$ for all possible transitions leading to a given state T_i : $S(d_i,\varsigma_{i-1}) = \varsigma_i$, where the *add* operation stands for plus in the log-domain and product in the probability domain.

[2] The certain means is the path optimizing ρ_i among all possible paths in VA.

(b) *compare* all evaluated metrics evaluated in the previous step,

(c) *select* the transition with the best ρ_i as the surviving path, save that and discard all other transitions.

3. backward part: select the final state with optimal metric and by the surviving paths select the overall path in the trellis. The final selected path refers to the vector MAP estimation of the data vector.

3.2.2 FSM decoding - FBA Algorithm

The Forward Backward Algorithm (FBA) is suited for the turbo codes decoding. The FBA aims with the *symbol*-MAP estimation for a given FSM. The algorithm consists from a recursive forward and backward part.

Using a similar argumentation as in the VA derivation, we can derive [18] the symbol MAP estimation with the FSM description:

$$
\begin{aligned}
p(d_i|\boldsymbol{x}) &= \sum_{\varsigma_{i-1}} p(\varsigma_{i-1}, d_i|\boldsymbol{x}) && (3.8)\\
&\sim \sum_{\varsigma_{i-1}} p(\boldsymbol{x}|\varsigma_{i-1}, d_i)p(\varsigma_{i-1}, d_i) && (3.9)\\
&= \sum_{\varsigma_{i-1}} p(\boldsymbol{x}_{0,\dots,i-1}|\varsigma_{i-1}, d_i)p(x_i|\varsigma_{i-1}, d_i, \boldsymbol{x}_{0,\dots,i-1})\\
&\quad p(\boldsymbol{x}_{i+1,\dots,N-1}|\varsigma_{i-1}, d_i, \boldsymbol{x}_{0,\dots,i})p(\varsigma_{i-1}, d_i) && (3.10)\\
&= \sum_{\varsigma_{i-1}} \underbrace{p(\boldsymbol{x}_{0,\dots,i-1}|\varsigma_{i-1})p(\varsigma_{i-1})}_{\alpha(\varsigma_{i-1})}\\
&\quad \underbrace{p(x_i|\varsigma_{i-1}, d_i, \boldsymbol{x}_{0,\dots,i})p(d_i)}_{\gamma(T_i)} \underbrace{p(\boldsymbol{x}_{i+1,\dots,N-1}|\varsigma_i)}_{\beta(\varsigma_i)}, && (3.11)
\end{aligned}
$$

where (3.10) follows from the chain rule and (3.11) is given by causality of the FSM. Since $\varsigma_i = S(\varsigma_{i-1}, d)$, we can rewrite (3.11) to

$$
p(d_i|\boldsymbol{x}) \sim \sum_{\varsigma_{i-1}} \alpha(\varsigma_{i-1})\gamma(T(d_i, \varsigma_{i-1}))\beta(S(d_i, \varsigma_{i-1})), \tag{3.12}
$$

where $T(d_i, \varsigma_{i-1}) = T_i$.

One can derive (e.g. [18]) a recursive formula

$$
\begin{aligned}
\alpha(\varsigma_{i-1}) &= \sum_{T_{i-1}:\varsigma_{i-1}} \alpha(\varsigma_{i-2})\gamma(T_{i-1}) && (3.13)\\
\beta(\varsigma_{i-1}) &= \sum_{T_i:\varsigma_i} \beta(\varsigma_i)\gamma(T_i), && (3.14)
\end{aligned}
$$

for α and β metrics evaluation serving jointly with (3.12) for an efficient evaluation of the symbol-wise MAP estimation which is called the *Forward Backward Algorithm*.

Note 3.3 (Turbo Decoding)**.** The main purpose of the FBA lies in the turbo decoding. The FBA is applied to the SISO blocks referring to the concatenated FSM based encoders. The algorithm is iterative and the convergence can be described by EXIT-charts. The decoder consists of two either parallel or serial concatenation of Soft In Soft Out (SISO) blocks connected by (de)interleaver.

The FBA is iteratively performed in the SISO blocks, where the output is evaluated according to an input from the complementary SISO block and the output serves as the input for the second SISO block and conversely.

3.2.3 Graph Based Solution

A common point of the aforementioned decoding algorithms is that the system can be described using a graph, that is

- FSM trellis in the VA case,
- compound FSM trellis with interleaver in the FBA-based turbo decoding.

One can consider a generalization of these algorithms. The factor graph describing a structure of an arbitrary system can be also used for the aforementioned decoding techniques. Some algorithms can be then defined upon the factor graph. Since the algorithms operate upon the decomposed system, they are much more efficient than the brute-force solution optimizing the global goal at the same time.

One of such algorithm (sum product algorithm) will be described later in greater detail. The LDPC codes can be decoded using this algorithm as it will be shown later.

3.3 Convergence Criteria

The turbo decoding as well as the graph based decoding have an iterative nature. The essential question is a convergence of such algorithms. The solution is achieved in a suboptimal way, that is the prize for the system trackability. The main tool that we will discuss are the EXIT charts [19,56] enabling to describe the convergence properties by means of the mutual information transfer investigation between some entities. Some other convergence criteria are discussed in Sec. 4.3.2.

3.3.1 Log-likelihood Arithmetics

The log-likelihood algebra [21] is used with advantage for a description of a random variable. This description is best suited in the binary case, that is random variable with two possible states. This refers to binary data in the digital communications. This algebra simplifies the description of the random variable.

Assume a random variable S with pmf given by $P(S=-1)=p_{-1}$ and $P(S=1)=p_1$. The domain of S is a binary GF with zero element -1. The Log-Likelihood Ratio (LLR) is defined as $\lambda = \log(p_1/p_{-1})$.

One can see that the binary random variable is stochastically described just by one number. Using the natural identity $p_{-1}+p_1=1$, we can express the pmf as

$$\begin{aligned} p_{-1} &= \frac{1}{\exp(\lambda)+1} = \frac{\exp(-\lambda/2)}{\exp(\lambda/2)+\exp(-\lambda/2)} \\ p_1 &= \frac{\exp(\lambda)}{\exp(\lambda)+1} = \frac{\exp(\lambda/2)}{\exp(\lambda/2)+\exp(-\lambda/2)}. \end{aligned} \tag{3.15}$$

The a posteriori probability after receiving an observation x is $p(s|x) \sim p(x|s)p(s)$, which can be written using LLR as

$$\lambda_{s|x} = \log\left(\frac{p(s=1|x)}{p(s=-1|x)}\right) = \underbrace{\log\left(\frac{p(x|s=1)}{p(x|s=-1)}\right)}_{x\lambda_C} + \underbrace{\log\left(\frac{p(s=1)}{p(s=-1}\right)}_{\lambda_s}. \tag{3.16}$$

If we consider Gaussian channel $p(x|s)$, one can show that $\lambda_C = 4\gamma$. We further note that

$$\lambda_{s|x_1,x_2} = \lambda_{C_1}x_1 + \lambda_{C_2}x_2 + \lambda_s. \tag{3.17}$$

for conditionally independent x_1 and x_2.

The GF(2) addition denoted by $\oplus$ is a cornerstone operation for the coding. We now assume a random variable given by $x = x_1 \oplus x_2$, where x_1 and x_2 are described by their LLR λ_{x_1} and λ_{x_2}. It can be shown that

$$\lambda_x = 2\tanh^{-1}(\tanh(\lambda_{x_1}/2)\tanh(\lambda_{x_2}/2)). \tag{3.18}$$

3.3.2 Binary EXIT Charts

The EXIT charts developed in [56] bring a semi-analytical tool well describing the convergence of the parallel turbo codes. The principle consists in measuring of the extrinsic information exchange of concatenated soft in soft out blocks (SISO). Nevertheless this principle is more general and it can be extended also for LDPC codes, where the SISO blocks are given by check nodes and variable nodes.

We assume a SISO block C. A mutual information between systematic data d and a prior input of the SISO block A is called an intrinsic information $I(d, A) = I_A$. The mutual information between the data and the SISO output $I(d, E) = I_E$ is called an extrinsic information. A function T given by $I_E = T(I_A)$ is called extrinsic transfer function. If the prior information is in the Gaussian form, the mutual information is given only by variance of A denoted σ_A.

In case of a concatenation of more SISO blocks, where the extrinsic information of the first one becomes to the intrinsic information of an another block and conversely, the convergence of the system as whole can be described using the extrinsic transfer functions of individual blocks.

We demonstrate the principle on the parallel turbo code example [56]. We assume that the received signal is given by $x = s + w$, where w is an AWGN and $s \in \{-1, 1\}$ is a systematic bit. The likelihood is then given by

$$\Lambda(s) = p(x|s) = \frac{1}{\sqrt{2\pi\sigma_w^2}} \exp\left(-\frac{(x-s)^2}{2\sigma_w^2}\right). \tag{3.19}$$

We use the log-likelihood form

$$\begin{aligned} X &= \log\frac{p(x|s=1)}{p(x|s=-1)} = \frac{(x+1)^2 - (x-1)^2}{2\sigma_w^2} \\ &= \frac{2x}{\sigma_w^2} = \frac{2}{\sigma_w^2}(s+w), \end{aligned} \tag{3.20}$$

which can be rewritten in the form $X = \mu_X s + n_X$ with mean

$$\mu_X = \frac{2}{\sigma_w^2} \tag{3.21}$$

and a zero-mean Gaussian random variable n_X with variance

$$\sigma_X = \frac{4}{\sigma_w^2}. \tag{3.22}$$

The variance and mean of X are connected through σ_w^2 as

$$\mu_X = \frac{\sigma_X^2}{2}. \tag{3.23}$$

The log-likelihood form of A in case of SISO block in AWGN channel was shown to have the Gaussian distribution. Therefore it can be written in the form $A = \mu_A s + n_A$, where s is the systematic bit. The mean is given by

$$\mu_A = \frac{\sigma_A^2}{2}. \tag{3.24}$$

Using this mean, we can write the likelihood (3.19) as

$$p_A(a|s) = \frac{1}{2\pi\sigma_A^2}\exp\left(-\frac{(a-\mu_A s)^2}{2\sigma_A^2}\right) \tag{3.25}$$

$$= \frac{1}{2\pi\sigma_A^2}\exp\left(-\frac{(a-\frac{\sigma_A^2}{2}s)^2}{2\sigma_A^2}\right). \tag{3.26}$$

A joint pdf $p(a,s)$ used to evaluate the intrinsic mutual information is given by

$$p_A(a,s) = p_A(a|s=-1) + p_A(a|s=1) \tag{3.27}$$

and the intrinsic information is therefore given by

$$I_A = \sum_s \int p_A(a,s)\frac{p_A(a)p(s)}{p_A(a,s)}da \tag{3.28}$$

$$= \frac{1}{2}\sum_{s=\pm 1}\int_{-\infty}^{\infty} p_A(a|S=s) \log\left[\frac{2p_A(a|S=s)}{p_A(a|s=1)+p_A(a|s=-1)}\right]da. \tag{3.29}$$

The extrinsic information can be evaluated in the same way, where the extrinsic distribution is used, that is

$$I_E = \sum_s \int p_E(a,s)\frac{p_E(a)p(s)}{p_E(a,s)}da \tag{3.30}$$

$$= \frac{1}{2}\sum_{s=\pm 1}\int_{-\infty}^{\infty} p_E(a|S=s) \log\left[\frac{2p_E(a|S=s)}{p_E(a|s=1)+p_E(a|s=-1)}\right]da. \tag{3.31}$$

Note 3.4 (Investigation of turbo decoding)**.** The EXIT function is evaluated for the SISO blocks given by the FSMs. During the evaluation of the EXIT function, one must pay attention that the probability serving for the evaluation (3.29) and (3.31) is in a proper form.

Note 3.5 (Investigation of LDPC decoding)**.** The first SISO block is formed by variable nodes with connection to observation and the second SISO block is given by the check node. One can find the EXIT chart application on LDPC codes e.g. [57].

Part II

FG-SPA Processing and its Implementation

4 Factor Graphs and Related Algorithms

The aim of this Chapter is to introduce the sum product algorithm on factor graphs and to formally describe the issues that should be taken into account during the implementation of this algorithm. We formally provide motivation for the sequel Chapter 5, where a concrete implementation framework *methodically* solving the issues is introduced as one of the main contribution of this work. Finally, we briefly touch the extension to looped FG-SPA and corresponding problems within this Chapter.

The factor graph (FG) is a tool graphically describing an arbitrary function (system) called the global function using its inner structure. The sum-product algorithm on factor graph (FG-SPA) is a highly-efficient algorithm capable to solve a wide family of problems. The most important one for this work is the MAP-detection.

The FG-SPA evolved as a generalization of trellis algorithms, where a general graph is considered instead of a trellis [62]. The framework of the FG-SPA is introduced in [27, 33], where the authors also show a number of applications such as the FSM/decoding, Kalman filtering, etc.

The problem of the inference (marginalization of the global function serving for e.g. the MAP-detection) is complex when it is applied on the global function. The point of the FG-SPA is that it marginalizes local functions in the decomposed system instead of the global function with typically significantly less complexity. A crucial assumption is a system decomposability. If some variables represented in the FG are random and *independent*, their joint probability (global function) can be *composed* as a product of some parts referring to the independent variables. The independence implies a strict limitation of the FG-SPA, that is the FG representing the global function must be *cycle-free*[1].

It was shown that the FG-SPA often works well also in the systems leading to a looped-graph, where the FG-SPA was not originally designed for. We stress that the inference is only *approximation* in the FG-SPA operating on the looped graphs (called also the loopy belief propagation [36, 66]). The algorithms very similar to the looped FG-SPA were re-discovered independently in more fields.

Many practical problems across many fields lead to the looped-graphs. Most notably for this work, we again mention the MAP-detection of the capacity achieving codes such as turbo or LDPC codes. We stress that the FG-SPA (or some of its variant) is capable to cope among others *all decoding algorithms* discussed in Chapter 3.

Furthermore, the global function need not to be restricted only to the decoder itself. We can consider e.g. a joint decoder and phase estimator as one global function. Therefore all decoding and signal processing can be covered by one global function. The advantage is that the individual tasks are processed jointly and therefore with a potentially better performance than solving the tasks independently.

The major problems that are presented in the looped FG-SPA are:

- convergence of the FG-SPA issue,
- FG-SPA message representation.

The convergence of the FG-SPA issue stands for a principle-nature issue. The cycled graph implies unknown fixed points existence and their uniqueness. The convergence of the FG-SPA to a proper fixed point is not also guaranteed in general. We refer [36] and references from there (e.g. [66]) for the reader interested in the investigation of the looped FG-SPA convergence topic.

While the issue of the FG-SPA convergence is a principle-nature problem, the message representation stands for the *implementation nature* problem. The problem is, roughly said, how to represent a general continuously valued function by means of several parameters (the less the better), because the messages are passed through the corresponding factor graph and therefore their description is unavoidable for the implementation. A *methodical* solution for this problem is one of the main contribution of this work. We refer Section 5.1.2 or e.g. [10, 13, 28] and references from there as an overview of the current state of art for the FG-SPA message representation problem.

[1] According to the original definition presented in [27, 33].

4.1 Cycle-Free Factor Graph and Sum Product Algorithm

We will first consider a cycle-free FG-SPA and present a fundamental theorem 4.1 stating the possibility of an exact inference evaluation in the cycle-free FG-SPA. A discussion of the implementation issues will follow.

4.1.1 Formal Definition of the Factor Graph

We start with some formal description of the system. A motivation of the following notation is to describe a collection of functions $\{f_I\}_I$ forming some local parts of the global function. The factor graph for such a system is then defined.

Let $\boldsymbol{x} \in \mathcal{A}_{\boldsymbol{x}}$ be a multi-dimensional variable with dimensions $[x_1, \dots, x_N] \in \mathcal{A}_{x_1} \times \cdots \times \mathcal{A}_{x_N} = \mathcal{A}_{\boldsymbol{x}}$, where a set[2] given by the individual dimensions is denoted by $\mathcal{X}$, i.e. $\mathcal{X} = \{x_1, \dots x_N\}$. Further let $\mathcal{F} = \{f_1, \dots f_M\}$, $f_I : \mathcal{A}_{\psi(f_I)} \to C_I, \forall I \in \{1, \dots, M\}$ be a set of functions, where $A_{\psi(f_I)}$ is a domain of f_I and C_I is its codomain. We defined a mapping $\psi(f_I) : \mathcal{F} \to \mathcal{X}^{n_I}$ assigning a vector of arguments to a given function f_I and n_I denoting the number of arguments of f_I. Similarly, we also define a mapping $\upsilon(i)$ giving a vector of functions, where x_i as the argument.

Example 4.1 (Simple Demonstration of Notation)**.**

We consider two functions $f_1(x_1, x_3, x_4)$ and $f_2(x_2, x_4)$. The domain of $f_2(x_2, x_4) = f_2(\boldsymbol{x}_{2,4}) = f_2(\psi(f_2))$ is $\mathcal{A}_{\psi(f_2)} = \mathcal{A}_{x_2,x_4} = \mathcal{A}_{x_2} \times \mathcal{A}_{x_4}$, the codomain is C_2, the mapping $\psi(f_2) = \boldsymbol{x}_{2,4} = [x_2, x_4]$ and $n_2 = 2$. Similarly domain of $f_1(x_1, x_3, x_4) = f_1(\boldsymbol{x}_{1,3,4}) = f_1(\psi(f_1))$ is $\mathcal{A}_{\psi(f_1)} = \mathcal{A}_{x_1,x_3,x_4} = \mathcal{A}_{x_1} \times \mathcal{A}_{x_3} \times \mathcal{A}_{x_4}$, the codomain is C_1, the mapping $\psi(f_1) = [x_1, x_3, x_4]$ and $n_1 = 3$. Finally $\upsilon(x_1) = f_1$, $\upsilon(x_2) = f_2$, $\upsilon(x_3) = f_1$ and $\upsilon(x_4) = [f_1, f_2]$.

Using the introduced notation, we can formally define the factor graph as a visualization of a global function, where the global function (whole factor graph) is composed from local functions as in Example 4.1. The global function $g(\mathbf{x})$ can be composed from two local functions $f_1(x_1, x_3, x_4)$ and $f_2(x_2, x_4)$ as shown in Fig. 4.1. One can see that the factor graph respects inner structure of the global function (see Example 4.2), where the functions are visualized by factor (check) nodes and the variables by variable nodes. The arguments of the function are represented by edge between the corresponding factor and variable node.

Definition 4.1 (Factor Graph)**.**

If $C = \bigcup_I C_I$ forms a commutative semi-ring, the *factor graph* is defined as a bipartite graph $\mathcal{G} = (\mathcal{N}, \mathcal{V}, \mathcal{E})$, where $\mathcal{N} = \{F_1, \dots F_M\}$ is a set of FN referring one to one to the set of functions $\mathcal{F}$, $\mathcal{V} = \{\nu_1, \dots, \nu_N\}$ is a set of VN referring one to one to the set $\mathcal{X}$ and $\mathcal{E} = \{e_1, \dots, e_L\}$ is a set of edges connecting FN and VN if and only if the variable represented by the VN is argument of the local function referring to the FN. It means that an edge connects a FN $F_I \in \mathcal{N}$ referring to f_I and a VN $\nu_i \in \mathcal{V}$ if the variable x_i represented by the VN ν_i is an argument of f_I. This can be written using the mapping ψ as ν_i is connected with F_I if and only if x_i is in the vector given by $\psi(f_I)$.

[2] We mean an ordered set by default.

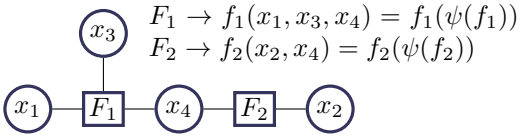


Figure 4.1: Simple demonstration of a factor graph corresponding to Example 4.1.

We will further not recognize between a VN or FN and the corresponding variable or function itself. It means that both a VN (ν_i) and the corresponding variable are denoted by x_i and both a FN and the corresponding function (f_I) are denoted by F_I. We will also use the mappings υ and ψ also for the factor nodes and variable nodes, since the sets $\mathcal{V}$ and $\mathcal{X}$ are one to one as well as the sets $\mathcal{F}$ and $\mathcal{N}$. It means that we will not recognize between $\psi(f_I)$ and $\psi(F_I)$. Both of them give a vector of either variables or variable nodes according to the context. We form a convention to visualize the FN by a rectangle and the VN by a circle. We will also use upper case letters for FN and lower case letters for VN.

A global function $g : \mathcal{A}_{\boldsymbol{x}} \to C$ represents a joint pf of $\boldsymbol{x}$. The global function can be composed[3] from the local functions as

$$g(\boldsymbol{x}) = 1/Z \prod_{I=1}^{M} f_I(\psi(f_I)),$$

where

$$Z = \sum_{\mathcal{X}} \prod_{I=1}^{M} f_I(\psi(f_I))$$

is basically a norming of the pf. Note that this norming constant is not a function of any $x \in \mathcal{X}$. The constant is conventionally normed such as $Z = 1$, because $g(\boldsymbol{x})$ is supposed to be the pf. We will further slightly abuse the notation such that we denote $\prod_{I=1}^{M}$ simply by $\prod_I$.

A pf of a particular set of variables A, where $A \subseteq \mathcal{X}$ can be evaluated by a marginalization of the global function, i.e.

$$p(\boldsymbol{x}_A) = \sum_{\mathcal{X}\setminus\mathcal{A}} g(\boldsymbol{x}) = \sum_{\mathcal{X}\setminus\mathcal{A}} \prod_{I} f_I(\psi(f_I)). \tag{4.1}$$

The following example shows a basic factor graph and also it demonstrates the idea that is beyond the implementation efficiency. We can also find there a basic motivation for the message passing that will be discussed later.

[3]Note that the operation $\prod$ stands for a composition operation and $\sum$ for the marginalization operation in general. Since this work aims with the sum product algorithm that will be formally defined in Sec. 4.1.3, we consider the summation and product as the marginalization and composition operations.

Example 4.2 (Simple Factor Graph).

A factor graph corresponding to Example 4.1 is shown in Fig. 4.1. The global function can be decomposed to two local functions such that $g(\boldsymbol{x}) = f_1(x_1, x_3, x_4) f_2(x_2, x_4) = f_1(\psi(f_1)) f_2(\psi(f_2))$. If we are interested in the marginalization of e.g. x_2, we can see that

$$\sum_{x_1,x_3,x_4} g(\boldsymbol{x}) = \sum_{x_1,x_3,x_4} f_1(x_1, x_3, x_4) f_2(x_2, x_4). \tag{4.2}$$

Nevertheless, this can be written as

$$\begin{aligned} \sum_{x_1,x_3,x_4} g(\boldsymbol{x}) &= \sum_{x_4} \underbrace{\sum_{x_1,x_3} f_1(x_1, x_3, x_4)}_{\mu(x_4)} f_2(x_2, x_4) \\ &= \sum_{x_4} \mu(x_4) f_2(x_2, x_4). \end{aligned} \tag{4.3}$$

If we carefully check 4.3, we can see that we decomposed the marginalization into two steps. In the other words, it is not necessary to provide marginalizations over three parameters at one time, i.e. 3 encapsulated loops, but it is sufficient to first evaluate $\mu(x_4)$ using marginalization over two variables and the last marginalization $\sum_{x_4} \mu(x_4) f_2(x_2, x_4)$ can be provided extra. It means that we save the third encapsulated loop in the evaluation, which is no longer encapsulated when (4.3) is used. One can feel that this simplification becomes to be more crucial in complex systems.
In addition, if we express similarly the marginalization of x_1 or x_3, we obtain

$$\sum_{x_1(\text{or } x_3),x_2,x_4} g(\boldsymbol{x}) = \sum_{x_1(\text{or } x_3)} f_1(x_1, x_3, x_4) \underbrace{\sum_{x_2,x_4} f_2(x_2, x_4)}_{\mu(x_1,x_3)} \tag{4.4}$$

$$= \sum_{x_1(\text{or } x_3)} f_1(x_1, x_3, x_4) \mu(x_1, x_3). \tag{4.5}$$

It is therefore sufficient to evaluate the marginalization $\mu(x_1, x_3) = \sum_{x_2,x_4} f_2(x_2, x_4)$ just once and to use it for the both x_1 and x_3 marginalization evaluations.

4.1.2 Message Passing

We shown how to visualise a global function by means of the FG using its inner structure and also how to express a marginalized probability function (4.1). The motivation is now to efficiently evaluate the marginalized function. We partly reveal the principle in Example 4.2. We defined a marginalized sub-result $\mu(x_4)$ that is then used to evaluate the marginal $p(x_2) = \sum_{\boldsymbol{x}_{1,3,4}} g(\boldsymbol{x})$. The sub-result $\mu(x_4)$ contains all information about x_1 and x_3 required for evaluation of $p(x_2)$ contained in a message. We now define this message more formally and form the fundamental Theorem 4.1 stating the way of the marginal evaluation using a message passing.

Definition 4.2 (Message in FG-SPA)**.**

For each edge given by a FN X_I and a VN x_i, we assign two *messages*. The first one is from the VN to the FN ($\mu^{x_i \to X_I}(x_i)$) and the second one with a converse direction ($\mu^{X_I \to x_i}(x_i)$). The message is a function of the variable x_i. A domain of the message is a set $\mu \in \mathcal{A}_\mu = \mathbb{M}$.

Note 4.3 (Notation comments to messages)**.** The domain inherits the identification of the message, i.e. $\mu^{x_i \to X_I}(x_i) \in \mathbb{M}^{x_i \to X_I}$. Each message can be parameterized by some further parameters which are situated into the subscript. We drop (simplify) the message parameters or its identification, whenever the this information is clear or not important.

Motivated by the marginal evaluation using the messages, we define the update rules defining a relation of the messages. Note that the definition corresponds to a commonly used update rules [27, 33].

Definition 4.3 (Update Rules)**.**

The FN update rule of a given message[a] $\mu^{F_I \to x_i}$ is a mapping $\mathcal{U}_F : \mathbb{M}^{\psi(F_I) \sim x_i \to F_I} \to \mathbb{M}^{F_I \to x_i}$, where $\mu^{\psi(F_I) \sim x_i \to F_I} \in \mathbb{M}^{\psi(F_I) \sim x_i \to F_I}$ are the messages inputing the node F_I excepting $\mu^{x_i \to F_I}$ and similarly for the variable node, i.e. $\mathcal{U}_V : \mathbb{M}^{\upsilon(x_i) \sim F_I \to x_i} \to \mathbb{M}^{x_i \to F_I}$. The particular form of the update rules is given by

$$\mu_{k+1}^{F_I \to x_i}(x_i) = \sum_{\psi(f_I) \sim x_i} f_I(\psi(f_I)) \prod_{x_j : \psi(f_I) \sim x_i} \mu_k^{x_j \to F_I}(x_j) \tag{4.6}$$

$$\mu_{k+1}^{x_i \to F_I}(x_i) = \prod_{F_J : \upsilon(x_i) \sim f_I} \mu_{k+1}^{F_J \to x_i}(x_i), \tag{4.7}$$

where (4.6) defines the update rule in FN and (4.7) defines the VN-update. Note that the sum operator $\sum$ becomes to integral $\int$ whenever the domain $\mathcal{A}_{x_i}$ is a continuous interval. The subscript k denotes the iteration.

[a] The unary tilde operation means the whole vector expecting the argument. See Section 1.4 for details.

The *marginal* of a variable x_i given by

$$\prod_{I \in \upsilon(i)} \mu^{X_I \to x_i}(x_i) \tag{4.8}$$

and it represents a marginalized global function serving for further purposes.

We state a fundamental theorem of the FG-SPA serving for the marginal evaluation using the message passing directed by the update rules.

Theorem 4.1 (Fundamental FG-SPA Theorem)**.**

Let $\boldsymbol{x}$ be a multi-dimensional random variable described by a joint pf $g(\boldsymbol{x})$. If the FG $\mathcal{G}$ referring to g is *cycle-free*, the message passing directed by the update rules defined by (4.6) and (4.7) provides messages enabling an exact inference evaluation (4.8) with an arbitrary scheduling algorithm containing all messages.

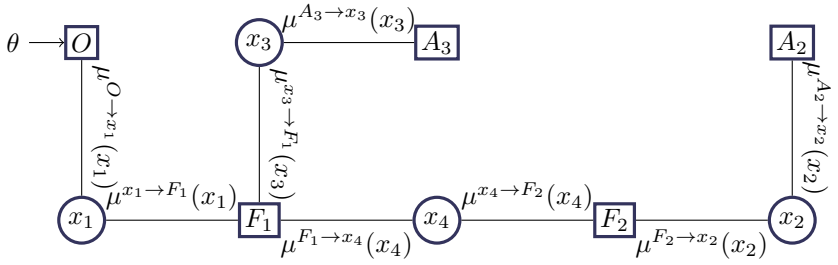


Figure 4.2: Simple demonstration of FG-SPA corresponding to Example 4.5. The messages required for the marginalization of $p(x_2, \theta)$ are denoted.

The theorem is so fundamental that the proof can be found in many tutorials to FG-SPA e.g. in [27].

4.1.3 Sum Product Algorithm

Although the FG can be established for any commutative semi-ring codomain, we consider $\mathcal{A}_{\boldsymbol{x}}$ to be a support of a multi-dimensional random variable $\boldsymbol{x}$. Providing further specification, we can formally define the sum product algorithm.

Definition 4.4 (Sum Product Algorithm)**.**

We consider a factor graph $\mathcal{G}$. The sum-product algorithm on the factor graph is the message passing algorithm directed by the update rules (Def. 4.3), where

- the local functions f_I are conditional pf.
- the marginal (4.8) is a marginalized pf $p(x_i)$, i.e. $p(x_i) = \prod_{I\in v(x_i)} \mu^{X_I\to x_i}(x_i)$.

We recognize forward messages having a prior form and backward messages in likelihood form.

The local functions f_I are some probability functions (pf - pdf, pmf or their mixture) up to a non-negative scale factor describing a multi-dimensional random variable $\psi(f_I)$. Consequently we state the codomain $C_I \in \langle 0, \infty), \forall f_I$.

Note 4.4 (Other Algorithms on FG)**.** The composition and marginalization operations are more general. One can consider e.g. max-sum algorithm, where the marginalization is given by maximization and the composition by sum operation. The Viterbi Algorithm in the log-domain can be mentioned as an instance of the max-sum algorithm.

Example 4.5 (Message passing in a simple example)**.**

We return to Example 4.2. We specify that x_1 is connected to an observation θ. Further, we assume an a prior information x_2 and x_3. This is expressed by adding further factor node to the graph as one can see in Fig. 4.2 The application of the update rules gives

$$\begin{aligned}\mu^{F_1\to x_4}(x_4) &= \sum_{x_1,x_3} \underbrace{p(x_1|x_3,x_4)}_{f_1(x_1,x_3,x_4)} \underbrace{p(x_3)}_{\mu^{x_3\to F_1}} \underbrace{p(\theta|x_1)}_{\mu^{x_1\to F_1}} \\ &= \sum_{x_1,x_3} p(\theta,x_1,x_3|x_4) = p(\theta|x_4). \end{aligned} \tag{4.9}$$

Using the VN update, we can see that $\mu^{F_1\to x_4}(x_4) = \mu^{x_4\to F_2}(x_4)$ and the second FN update is given by

$$\mu^{F_2\to x_2}(x_2) = \sum_{x_4} \underbrace{p(x_4|x_2)}_{f_2(x_2,x_4)} \underbrace{p(\theta|x_4)}_{\mu^{x_4\to F_2}(x_4)} = p(\theta|x_2). \tag{4.10}$$

Using a prior information about x_2 (message $\mu^{A\to x_2}(x_2) = p(x_2)$), we evaluate the marginal as $\mu^{F_2\to x_2}(x_2)\mu^{A\to x_2}(x_2) = p(x_2)p(\theta|x_2) = p(\theta,x_2)$. Note that we can recognize forward form of the messages (a prior form) and backward messages (likelihood).

One can recognize the message $\mu(x_4)$ evaluated in Example 4.2 in (4.9).

4.1.4 MAP Decoding and MMSE Estimation Using FG-SPA

The MAP decoding is a fundamental capability of the FG-SPA for our purposes. The maximum a posterior probability of a parameter ξ_i given by an observation $\boldsymbol{\theta}$ is given by (2.10) $\hat{\xi}_i = \arg\max_{\xi_i} p(\xi_i|\boldsymbol{\theta})$. This can be rewritten as

$$\hat{\xi}_i = \arg\max_{\xi_i} p(\xi_i|\boldsymbol{\theta}) \tag{4.11}$$

$$= \arg\max_{\xi_i} p(\xi_i,\boldsymbol{\theta}) \tag{4.12}$$

$$= \arg\max_{\xi_i} \sum_{\boldsymbol{x}\sim\xi_i} g(\boldsymbol{x}) \tag{4.13}$$

$$= \arg\max_{\xi_i} \sum_{\boldsymbol{x}\sim\xi_i} \prod_I f_I(\psi(f_I)). \tag{4.14}$$

Assuming a cycle-free FG $\mathcal{G}$ referring to the global function g, we can use fundamental SPA Theorem 4.1 to evaluate *exactly* the marginal pf $p(\xi_i,\boldsymbol{\theta})$. An evaluation of the MAP estimation is then straightforward with a reasonable complexity. The MMSE estimation can be done in a similar manner, i.e. using (2.9), we have

$$\hat{\xi}_i = \mathrm{E}[p(\xi_i|\boldsymbol{\theta})] \tag{4.15}$$

$$= \mathrm{E}\left[\sum_{\boldsymbol{x}\sim\xi_i} \prod_I f_I(\psi(f_I))\right] \tag{4.16}$$

for a given observation $\boldsymbol{\theta}$.

4.2 FG-SPA Implementation

One of the main advantage of the FG-SPA is that it defines powerful algorithms. These algorithms are described precisely using the FG-SPA. Nevertheless if one would like to implement these algorithms, some difficulties arise. We browse these difficulties including their formal description. According to this description, we form a formal requirement for the implementation framework serving for a direct implementation of the FG-SPA.

4.2.1 Message Types in FG-SPA and their Representation

We start with description of the message representation. We have to somehow cope the representation of the message for the processing purposes. We are able to store and process a finite number of discrete values. This gives as a basic constraint, which is not important in case of discrete messages. The FG-SPA is, however, defined also for messages describing continuously valued variables (pdf) requiring uncountable number of parameters for their direct exact representation in general. This brinks a question about a parameterization of these messages.

Since we are able to store a finite number of parameters, the message is therefore parameterized by an n-dimensional vector $\boldsymbol{q} \in \mathcal{A}_{\boldsymbol{q}} \subseteq \mathbb{R}^n$. The message $\mu \in \mathbb{M}$ is then given by $\mu = \Phi(\boldsymbol{q})$, where $\Phi : \mathcal{A}_{\boldsymbol{q}} \to \mathbb{M}$ is a function defining the *parameterization*. Obviously, the less number n the more efficient parameterization. On the other hand some messages may need an infinite number of parameters to be exactly represented. We will also use an inverse mapping $\Phi^{-1} : \mathbb{M} \to \mathcal{A}_{\boldsymbol{q}}$ to determine the parameter vector for a given message, i.e. $\boldsymbol{q} = \Phi^{-1}(\mu)$.

The *parameterization of the message* is therefore given by the mappings $\Phi : \mathcal{A}_{\boldsymbol{q}} \to \mathbb{M}$ and $\Phi^{-1} : \mathbb{M} \to \mathcal{A}_{\boldsymbol{q}}$ determining the relation between the message $\mu \in \mathbb{M}$ and its parameterization $\boldsymbol{q} \in \mathcal{A}_{\boldsymbol{q}} \subseteq \mathbb{R}^n$. We further define a *dimensionality of the message* as number of parameters $\mathcal{D}$ required for the message representation, that is the length of the vector $\boldsymbol{q}$.

Within this work, we assume all variables x_i to be scalar variables in complex domain or some of its subset, i.e. $x_i \in \mathcal{A}_{x_i} \subseteq \mathbb{C}, \forall x_i \in \mathcal{X}$. According to the domain $\mathcal{A}_{x_i}$ of a variable x_i, we recognize the following types of the messages connected to the VN x_i:

Discrete Messages

The discretely valued message is a message connected to x_i, if $\mathcal{A}_{x_i}$ has a finite number of elements. A straightforward representation of such a message is $\boldsymbol{q} = [\dots, q_j, \dots]$, where the components q_j refers to the elements of $\mathcal{A}_{x_i}$. This refers to the pmf of the corresponding variable x_i. The minimal dimensionality over all parameterizations of this message is obviously upper-bounded by the number of parameters, i.e. $\min(\mathcal{D}) \leq |\mathcal{A}_q| = M_q$.

Example 4.6 (Log-likelihood message representation)**.**

A typical example is a binary message, where $x_i \in \mathcal{A}_{x_i} = \{-1, 1\}$ and $\mu(x_i) = [q_0, q_1] = [p_{-1}, p_1]$. An another form of the binary message representation is log-likelihood ratio (LLR). Since x_i is a binary random value, it can be described by a log likelihood ratio given by $\lambda = \log(p_1/p_{-1})$ standing for mapping Φ^{-1}. Since $p_{-1} + p_1 = 1$, we can express the message by (3.15), i.e.

$$\mu(x_i) = [p_{-1}, p_1] = \Phi(\lambda) = \left[\frac{1}{1+\exp(\lambda)}, \frac{\exp(\lambda)}{1+\exp(\lambda)}\right]. \tag{4.17}$$

Note that the dimension of the message is $\mathcal{D} = 1$, because the message can be fully described by a single real number (λ). We therefore found a parameterization of a pmf with lower dimension then the description by the pmf itself, where $\mathcal{D} = 2$ for the binary variable.

Parameterizable Continuous Messages

The *continuous parameterizable message* is a message connected to x_i, if $\mathcal{A}_{x_i}$ is an infinite set, e.g. a continuous interval $I \subseteq \mathbb{R}$. This message is continuous, but directly parameterizable. The direct parameterization is a concrete form of the function Φ. The number of parameters again forms the upper-bound on the message dimensionality.

Example 4.7 (Gaussian message representation)**.**

As an example, we mention the real Gaussian message fully described by its mean and variance $\boldsymbol{q} = [\bar{x}_i, \sigma_x]$ and the message is given by

$$\mu(x_i) = \Phi(\bar{x}_i, \sigma_x) = \frac{1}{\sqrt{2\pi\sigma_x^2}} \exp\left(\frac{|x_i - \bar{x}_i|^2}{2\sigma_x^2}\right). \tag{4.18}$$

The inverse mapping Φ^{-1} is given by $\boldsymbol{q} = \Phi^{-1}(\mu(x)) = [\mathrm{E_x}[\mu(\mathrm{x})], \mathrm{var}[\mu(\mathrm{x})]]$. Note that it can be shown that the mapping Φ is bijection in this case.

General Continuous Message

A *general continuous message* is a message connected to x_i, if $\mathcal{A}_{x_i}$ is an infinite set and there is not any straightforward parameterization of the message. This means that the dimension of the message is either infinite, or we are not able to find a parameterization with a reasonable dimension.

Example 4.8 (Mixed-Type Message)**.**

The message appears whenever both discretely and continuously valued messages are presented in one FG-SPA. We mention the phase model of the joint decoder phase estimator as an instance. The problem is increasing number of parameters required for the message parametrization with increasing iteration. See a demonstration in Fig. 4.3.

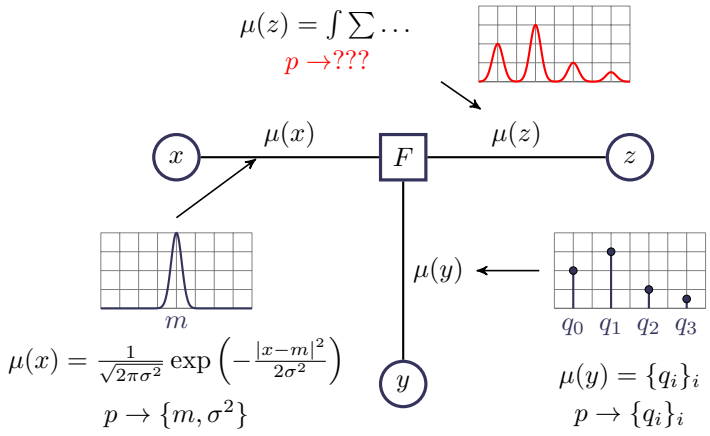


Figure 4.3: Illustration of the mixed-density message parameterization problem.

4.2.2 Message Approximation

Motivated by the representation of the general continuous message, we consider some parameterization that does not exactly determine the message, but instead some *approximation* of the true message is provided. Introducing of the approximated message, we bring some kind of inaccuracy to the FG-SPA, which must be somehow handled. On the other hand, we are able to (at least) approximately describe all messages with a given number of parameters. More formally, we define the message approximation using a parameterization, which does not fully describe the exact message. Suppose that the exact message $\mu \in \mathbb{M}$ is parametrized by $\boldsymbol{q} = \Phi^{-1}(\mu)$ as we considered so far. But up to now, we silently assumed that the message can be then obtained by $\mu = \Phi(\boldsymbol{q})$, as in Example 4.7. We now suppose that no such a mapping Φ that $\mu = \Phi(\Phi^{-1}(\mu)), \forall \mu \in \mathbb{M}$ exists. Instead, the mapping Φ is given by $\Phi : \mathcal{A}_{\boldsymbol{q}} \rightarrow \hat{\mathbb{M}}$, where the approximated message must not be in $\hat{\mathbb{M}}$ as one can see in Fig. 4.4. The messages $\tilde{\mu} \in \mathbb{M} \setminus \hat{\mathbb{M}}$ are therefore not exactly parameterizable by Φ. The approximated message $\hat{\mu} \in \hat{\mathbb{M}}$ is thus given by a parameterization $\hat{\mu} = \Phi(\boldsymbol{q})$.

The principle of the parameterization causing the message approximation is shown in Fig. 4.4, where we also considered a hypothetical vector $\boldsymbol{q}_E \in \mathcal{A}_{\boldsymbol{q}_E}$ exactly representing the true message. Particularly, we assume:

- The parameterization assigns a parameter vector $\boldsymbol{q} = \Phi^{-1}(\mu)$ to all messages, i.e. $\Phi^{-1}(\mu) \in \mathcal{A}_{\boldsymbol{q}}, \forall \mu \in \mathbb{M} \cup \hat{\mathbb{M}}$. This mapping is not injective in general and therefore more messages are mapped to the same parameterization as shown in Fig. 4.4. We should also note that the mapping Φ^{-1} is formally given by $\Phi^{-1} : \mathbb{M} \cup \hat{\mathbb{M}} \rightarrow \mathcal{A}_{\boldsymbol{q}}$.
- The approximated message can be unambiguously assigned for all parameter vectors, i.e. $\Phi(\boldsymbol{q}) \in \hat{\mathbb{M}}, \forall \boldsymbol{q} \in \mathcal{A}_{\boldsymbol{q}}$. The mapping Φ is supposed to be a bijection[4], that is the parameter vector and the approximated message are one to one mapping (see Fig. 4.4).
- The last assumption is that the mapping Φ^{-1} is also bijection, if it is provided upon the approximated message[5]. As a consequence, the approximated message referring to the parameter vector of the approximated message is the same, i.e. $\hat{\mu} = \Phi(\Phi^{-1}(\hat{\mu})), \forall \hat{\mu} \in \hat{\mathbb{M}}$.

[4] One can consider a parameterization such that more parameters can represent the same (approximated) message. Nevertheless it is not necessary for our purposes, so we will assume Φ to be clearly a bijection within this work.

[5] The mapping Φ^{-1} is in fact composed from two mappings. The first one is the inverse of Φ, which is clearly given in $\hat{\mathbb{M}}$, since Φ is bijection. The second part is a mapping from $\mathbb{M} \setminus \hat{\mathbb{M}}$ that is not invertible (see Fig. 4.4).

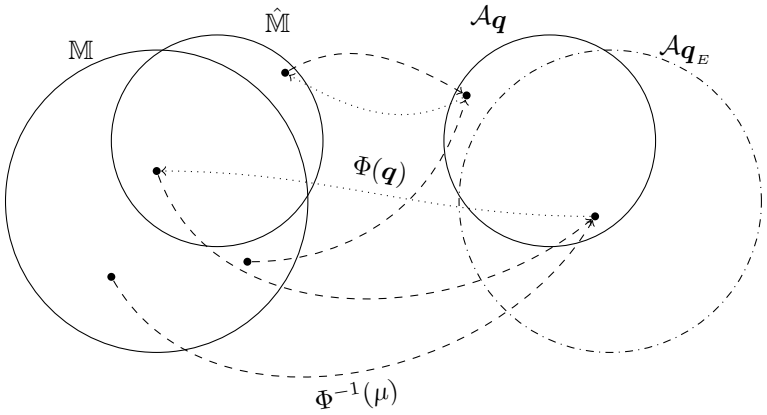


Figure 4.4: Principle of the message approximation given by the parameterization $\hat{\mu} = \Phi(\boldsymbol{q})$. The true message $\mu \in \mathbb{M}$ is mapped through a mapping Φ^{-1} (dashed arrows) to a parameter vector $\boldsymbol{q} \in \mathcal{A}_{\boldsymbol{q}}$. The vector $\boldsymbol{q} \in \mathcal{A}_{\boldsymbol{q}}$ is unambiguously related with an approximated message $\hat{\mu} \in \hat{\mathbb{M}}$ by a mapping Φ (dotted arrows). We denoted a hypothetical vector $\boldsymbol{q}_E \in \mathcal{A}_{\boldsymbol{q}_E}$ exactly representing the true message μ (dash-dotted set).

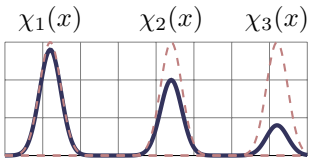
Example 4.9 (Representation of a Continuous Message by Samples)**.**

If a domain of the variable x_i that is represented by a message $\mu(x_i)$ is some real interval, i.e. $x_i \in \mathcal{A}_{x_i} = I \subseteq \mathbb{R}$, the message can be surely parameterized by samples. It means that an original true continuously valued message $\mu(x_i) \in I = \langle a, b)$ in the pdf form is represented a vector $\boldsymbol{q} = [q_1, \dots, q_{\mathcal{D}}]^T = [\mu(a), \dots, \mu(a + (\mathcal{D}-1)\Delta)]]^T$, where $\Delta = (b-a)/\mathcal{D}$. This formally defines the mapping $\Phi^{-1}(\mu)$. And the approximated continuous message is then composed (mapping Φ) as a piecewise function from the samples

$$\hat{\mu}(t) = \sum_{i=0}^{\mathcal{D}-1} q_{i+1}\nu(t - i\Delta), \tag{4.19}$$

where $\Delta = (b-a)/\mathcal{D}$, $\nu(t) = 1$ for $t \in \langle 0; \Delta)$ and $\nu(t) = 0$ otherwise. Using a finite number of samples (see Fig.4.8a), we are able to represent an arbitrary message in a finite interval. Intuitively, the more samples the more exact representation can be expected but at expanse of higher computational complexity.

The number of samples in Example 4.9 suggests a crucial natural question about quality of the approximation. One can feel a trade-off between the fidelity of the message representation and the dimension of the parameterization. The fidelity can be measured by means of the overall FG-SPA performance. Nevertheless, it would be useful to define a metric $\mathcal{M}(\mu, \hat{\mu})$ that assigns some quality to the message representation computable just from the message approximation and the true message. As an example of such a metric, we mention the mean square error of the approximated message. This metric will be optimized by the proposed KLT message representation in Sec. 5.2.1.



$$\mu(x) \approx c_1\chi_1(x) + c_2\chi_2(x) + c_3\chi_3(x)$$

Figure 4.5: Demonstration of the canonical message representation.

Canonical Message Representation

A general concept of the message representation is proposed in [63], where the authors propose the canonical message representation consisting in a set of kernel function representing the message as a superposition of some kernel functions. The message is then defined by a parameter vector $\boldsymbol{q}_j$.

The message is written in the form

$$\hat{\mu}(x_i) = \sum_j \chi_j(\boldsymbol{q}_j, x_i), \tag{4.20}$$

where χ_j are the kernel canonical functions and $\boldsymbol{q}_j$ is a set of parameters (see Fig 4.5 for an illustration). This is, however, rather a generic concept how to describe a certain class of message representation, because no way how to obtain methodically the kernel functions is not given. Furthermore, the canonical representation solves only the parameterization $\hat{\mu} = \Phi(\boldsymbol{q})$. No way of the evaluation of the parameter $\boldsymbol{q}$ is given in general (function Φ^{-1}).

Note 4.10 (Linear Canonical Message Representation)**.** If the mapping Φ can be written as a superposition a some kernel set, we refer such a representation given by Φ as a Linear Canonical Message Representation (LCMR). It means that a message can be written in form

$$\mu(x) \approx \hat{\mu}(x) = \sum_{i=1}^{\mathcal{D}} q_i\chi_i(x), \tag{4.21}$$

where $\boldsymbol{q}$ is the vector parameterizing the message and $\{\chi_i(x)\}_{i=1}^{\mathcal{D}}$ is a collection of (canonical) kernel functions. Note that in this case, we still have available only the mapping Φ. In the other words, we no way of the parameterization obtaining from the true message is given in general. An example with three canonical kernels is shown in Fig. 4.5.

Note 4.11 (Linear Canonical Message Representation with Orthogonal Kernel)**.** If we assume a LCMR, where the kernel set $\{\chi_i(x)\}_{i=1}^{\mathcal{D}}$ is orthogonal, we are able to construct also the mapping Φ^{-1} by means of the inner product, since $\langle\chi_i, \chi_j\rangle = 0$, $\forall i \neq j$ and therefore the i-th component of the parameter vector is given by $q_i = \langle\mu, \chi_i\rangle$.

4.2.3 Update Rules Upon the Message Representation

We introduced the update rules upon the messages in (4.6) and (4.7) as mapping[6] $\mathcal{U} : \mathbb{M}^n \to \mathbb{M}$, for some n. That is the update rules defined upon the true messages. Nevertheless during the processing, the message

[6] We assume an equal domain for all messages corresponding to the update for the notation simplicity. Nevertheless there is no reason for this assumption in general.

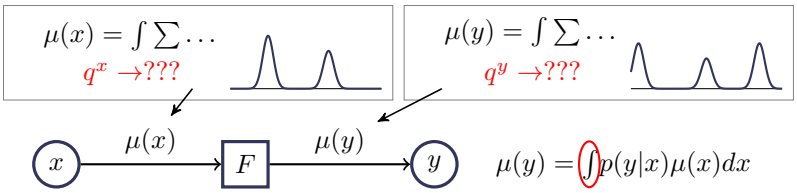


Figure 4.6: Demonstration of the update rules (Def. 4.3) provided upon the exact form of the message, which is however difficult to be directly handled in the processing. Moreover the integration of a continuous function must be evaluated within the update rule in case of continuous messages.

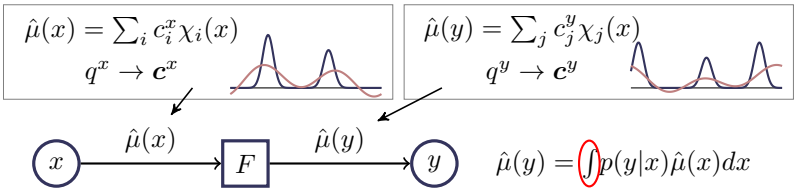


Figure 4.7: Demonstration of the update rules (Def. 4.3) provided upon the (approximated) message representation. This case solves the problem with message representation, nevertheless, the integration of a continuous function must be evaluated within the update rule in case of continuous messages.

are supposed to be represented rather by their parameterization. It would be therefore useful to find the corresponding update rules that will be applied directly to the message representation.

More formally, the implementation consisting from the message representation and the update rules can be given either directly by the definition of the FG-SPA or some more sophisticated (efficient) message representation and the corresponding update rules, particularly

1. Messages are given $\mu \in \mathbb{M}$ and the update rules $\mathcal{U} : \mathbb{M}^n \rightarrow \mathbb{M}$ in conventional FG-SPA.
2. Messages are given by their parameterization $\boldsymbol{q} \in \mathcal{A}_{\boldsymbol{q}}$ and the UR are defined upon the parameterization, i.e. $\mathcal{U}' : \mathcal{A}_{\boldsymbol{q}}^n \rightarrow \mathcal{A}_{\boldsymbol{q}}$, where the mapping $\mathcal{U}'$ properly respects both the message parameterization Φ, Φ^{-1} and the update rules $\mathcal{U}$ defined in Def. 4.3.

We particularly recognize the following update rules according to description of the messages.

Update Rules Upon the Exact Messages

The update rules provided directly according to the definition of the update rules (Def. 4.3) face two problems in case of general continuous messages. The fist one is the *representation* of the general continuously valued message and the second on is the integration required for the update rules, which is often *numerically intensive* when it is provided numerically (see Fig. 4.6 for a demonstration).

Note that no mapping Φ is used for the message parameterization, the update rules are given directly by the corresponding pf. This case is rather a starting point for some further analytical derivations then a direct manual of the FG-SPA implementation excepting the update rules upon the discrete messages.

Update Rules Upon the Parameterized (Approximated) Messages

In this case, we solved the problem with the message parameterization, because we have available mappings Φ and Φ^{-1}. Nevertheless we still use the update rules according to Def. 4.3 containing the integration that is not suitable for an efficient direct implementation in general. It means that the update rules are

provided by a chain of mappings given by $\Phi \circ \mathcal{U} \circ \Phi^{-1} : \mathcal{A}_{\boldsymbol{q}}^{n} \to \hat{\mathbb{M}}^{n} \to \hat{\mathbb{M}} \to \mathcal{A}_{\boldsymbol{q}}$ for some n, where we assume the *parameterization* of the message as both input and output to the update rules. See Fig. 4.7 for demonstration.

This is some middle-step between the direct definition of the update rules and the desired implementation framework.

Update Rules Directly Upon the Parameterization

This case refers to the target of the derivation of the update rules for a general message representation. It provides a direct relation between the message parameterizations. It means that the chain of mappings from the previous case is reduced to $\mathcal{A}_{\boldsymbol{q}^n} \to \mathcal{A}_{\boldsymbol{q}}$ and the integration required in Def. 4.3 is avoided for finite length of vector $\boldsymbol{q}$ (dimension $\mathcal{D}$). We will closely focus on this topic in Section 5.2.1, where we introduce the update rules for a generic LCMR with orthogonal kernels introduced in Note. 4.11.

Example 4.12 (Update Rules Upon Sampling)**.**

As an example, we mention the update rules upon sampling of the continuously valued message showed in Example 4.9. One can show [13] that the update rules (Def. 4.3) applied on the samples refer to the numerical integration with rectangular rule and therefore a natural solution for the update rules. The samples (parameterization) of pdf (true message) are taken as pmf inputting into the update rules.

Example 4.13 (Update LLR for LDPC)**.**

Another example is the update rules design for the LLR (or rather soft bit) message (4.17) suited for the decoding of LDPC codes. The factor nodes refer to the check nodes proving GF-sum, i.e. $\oplus$ operation. The solution can be found e.g. in [23].

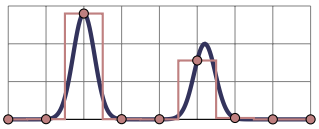
4.2.4 FG-SPA Implementation Framework

The FG-SPA can be easily (and exactly) implemented for the discrete (especially binary) messages. The messages are exactly represented by their pmf and the update rules are performed directly according to their definition (4.6) and (4.7). This is a straightforward way, which is not necessarily the most efficient. As an example, we can mention the soft bit implementation of LDPC. This implementation attains less complexity, because both the message representation and the update rules process a single-valued messages for the binary case as we discussed in Example 4.13.

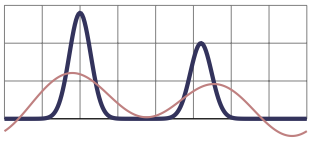
To unify the ways of the FG-SPA, we define the *implementation framework* as a message representation and the corresponding update rules upon the given message representation.

Definition 4.5 (Implementation FG-SPA Framework)**.**

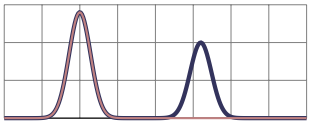
The FG-SPA *implementation framework* is a tuple $(\Phi, \mathcal{U}')$, where Φ defines a message parameterization (even approximately) and the corresponding *update rules* $\mathcal{U}'$ are defined directly upon the given message representation.



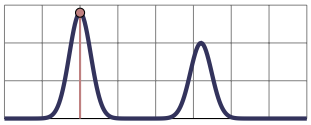
(a) Discretization of the continuously valued message.



(b) Representation using the coefficients of the Fourier decomposition.



(c) Gaussian representation given by mean and variance of the original message.



(d) Representation by Dirac-delta function posed into the argument of maximum of the original message.

Figure 4.8: Different message representations of general continuously valued messages.

Note 4.14 (Scheduling Algorithm)**.** We said that the implementation framework is given by the message representation and the corresponding update rules upon the message representation. We should remind that in case of looped FG-SPA, we have to also consider some scheduling algorithm. The scheduling algorithms will be discussed in Sec. 4.3.1.

Criteria of the FG-SPA Implementation Framework

We have already defined the implementation framework, but we did not consider any way, how to compare the frameworks from some quality point of view. A natural requests of the FG-SPA implementation are:

- *efficiency*,
- *exactness*.

Moreover, we are often by a limited *memory* available to processing. These requests are naturally in contradiction and therefore some trade-off is often needed. The efficiency request basically means a simple message representation (represented by a few parameters) that leads to efficient update rules design (relation between the few parameters).

4.2.5 Instances of the FG-SPA Implementation Frameworks

We show some possible FG-SPA message representation frameworks that can be found in literature in greater detail (see Fig 4.8). Some of these representations serve later as reference implementation frameworks for the numerical verification of the proposed design.

Sample Representation with Numerical Integration

The discretization of the continuous message is the most straightforward method of the practically feasible representation as it is discussed in [12, 13]. An arbitrary precision might be achieved using this representation by increasing the number of samples (complexity of the approximated message). This framework is given by merging Examples 4.9 and 4.12.

A straightforward evaluation of the update rules makes this representation case very important, because it is principally usable for every case, where we do not care about the numerical intensity of the FG-SPA. A direct application of the marginalization upon the message representation (the discrete vector $\boldsymbol{p}$ taking

as pmf) leads to an approximation of the marginalization upon pdf, where the integration is performed by the numerical integration with rectangular rule (see e.g. [13]).

Fourier Representation

The well known Fourier decomposition enables to parameterize the message by the Fourier's series as

$$\hat{\mu}(t) = \Phi(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{\alpha_0}{2\sqrt{\pi}} + \frac{1}{\sqrt{\pi}} \sum_{i=1}^{\mathcal{D}} \alpha_i \cos(it) + \beta_i \sin(it), \tag{4.22}$$

where $\alpha_i = 1/\sqrt{\pi} \int \mu(t) \cos(it) dt$ and $\beta_i = 1/\sqrt{\pi} \int \mu(t) \sin(it) dt$ gives the inverse mapping Φ^{-1}. Dimensionality is given by $\mathcal{D} = 2N + 1$. In [10], the authors have derived update rules for the Fourier coefficients in a special case of the random-walk phase model.

This message representation is the LCMR with orthogonal kernels as we stated in Note 4.11. The Fourier representation also exemplifies the case when $\hat{\mathbb{M}}$ is not a subset of $\mathbb{M}$ as one may wonder, when we discussed the message approximation (see Fig. 4.4). Consider that the set $\hat{\mathbb{M}}$ is given by all pdfs, that is all *non-negative* functions that satisfies $\int \mu(t) dt = 1$. The Fourier approximation, however, does not guarantee a non-negative message as one can see in Fig. 4.8b.

Dirac-Delta Representation

The message is represented by a single number situated in the maximum of the message (see Fig. 4.8d). The approximated message is given by

$$\hat{\mu}(t) \quad = \quad \delta(t - \arg\max_{\hat{t}} \mu(\hat{t})). \tag{4.23}$$

The update rules upon the Dirac-Delta message approximation can be found e.g. in [12] for the phase estimation problem. The update rules are based on the gradient methods

Gaussian Representation

Gaussian representation and its modifications are widely used message representations (e.g. [52]). This is given by simplicity of the message representation given just by mean and variance (m, σ) with the interpretation

$$\hat{\mu}(t) \quad = \quad \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{|t-m|^2}{2\sigma^2}\right), \tag{4.24}$$

where $m = \mathcal{E}[\mu(t)]$ and $\sigma^2 = \mathrm{var}[\mu(t)]$ (see Fig. 4.8c) for a real message. We have already discussed this case in Example 4.7.

The update rules for linear only factor nodes can be found in [32]. The Gaussian representation is moreover often used to approximate such a message representation that is difficult to be parametrized directly (e.g. [52, 53, 55]).

4.3 SPA on Looped Factor Graphs

This section is written as an overview what happens if the factor graph contains loop. Since the FG-SPA implementation framework as we discussed so far does not explicitly need a cycle-free corresponding FG, we can freely extend the terms defined in Sec. 4.2 to the looped FG-SPA. Nevertheless, we must pay attention to the conventional problems of the looped FG-SPA.

The FG-SPA as defined for the cycle-free FG can be used also in the looped case. The main differences are summarized bellow:

- The FG-SPA becomes to be an *iterative algorithm* and a convergence uncertainty occurs.
- Even, if the looped FG-SPA converges, the inference is evaluated only *approximately* and a further result investigation is required. Moreover, the stationary point must not be unique.
- The performance of the FG-SPA depends on a *message scheduling*.
- The *message representation* issue often becomes to be hardly trackable for mixed-type messages (Example 4.8).

Although the mentioned issues, the looped FG-SPA presents a tool to solve much more problems then the cycle-free FG-SPA. We have already mentioned some examples, but the most important application is capability of the algorithms described in Chapter 3.

4.3.1 Message Scheduling in FG-SPA

We have several choices of the scheduling. There are three fundamentally different approaches, concretely:

1. *Random scheduling* selects randomly the messages to be updated.
2. *Predefined fixed scheduling* strictly defines the scheduling prior the SPA starts. It can be parallel, i.e. all messages are updated once per iteration, e.g. flood schedule algorithm (FSA), or sequential, i.e. there is a given order and the updated messages are used directly after evaluation. A notable sequential scheduling algorithm is the FBA used in the turbo decoding.
3. *Adaptive scheduling* selects message to be updated according to a comparison between a potentially updated message with the non-updated message. It means that a metric $\mathcal{M}(\mu_k, \mu_{k+1})$ is selected and then the message with the metric satisfying some properties e.g. [16] is updated.

We can mix the mentioned principles. For example, one part of messages are passed randomly and the second one by a predefined scheduling. This approach is reasonable, when the system is composed from a smaller sub-systems and each subsystem has its own well defined scheduling order.

4.3.2 Convergence of the Looped FG-SPA

The convergence of the looped FG-SPA becomes a huge complication. We cannot use the fundamental theorem 4.1 and thus we have no guarantee of an approximated inference evaluation. We can mention several approaches of the looped FG-SPA convergence that can be found in literature.

- Analytical approach [36,37], where the authors describe the update function by an analytical function of the all messages in one iteration and then they investigate the properties of the function by the Banach theorem. Their analysis gives a very practical method for binary messages and relatively small graphs. The FG-SPA with more complex (even non-binary) messages suffers from very difficult analytical formulas and the convergence test is dropped absolutely for FG-SPA containing continuous messages in [37].
- Statistical physics approach presented in [66] is completely different to [36, 37]. The authors have used methods of statistical physics. They found a correspondence between the fixed points of the FG-SPA and the stationary points of Bethe approximation of the free energy.

- EXIT chart convergence description 3.3.2 [56] was designed as a description of the turbo decoding. However, the principle is more general, it describes the convergence according to the mutual information transfer of some entities. As it was stated in [2], the FG automatically handle correctly the extrinsic and intrinsic information. Nevertheless the difficulty occurs with the numerical evaluation of the mutual information, since we cannot generally assume that it can be fully determined by a single parameter as in case of binary EXIT charts.

We can finally mention that it is known that the correlation decays exponentially with the loop length in the FG.. Therefore (if possible) it is convenient to maximize the FG-loops length (girth) for the FG-SPA performance optimization, because the long loops with reduced correlation are closer to the crucial FG-SPA assumption, that is the decomposability of the system (i.e. independence of the parameters).

5 FG-SPA Implementation Framework

This Chapter aims with the implementation issue of the sum product algorithm on factor graphs (FG-SPA) standing for one of the main results of this work. We first propose a methodical way of a canonical message representation obtaining for a general continuously valued messages in FG-SPA [44]. This method is based on Karhunen Loève Transform (KLT) exploiting the stochastic nature of the messages in the FG-SPA[1]. The orthogonal canonical kernels serving for the message representation are evaluated. As a consequence of the KLT, the mean square error of the message approximation caused by dropping a given set of kernels can be easily controlled. This can be used for such a message representation that the complexity/fidelity trade-off can be easily found for a given fidelity goal. We further propose an update rule design for a general family of linear canonical message representations with orthogonal kernels [47]. Since the eigenfunctions resulting from the KLT are orthogonal, a generic FG-SPA implementation framework can be composed from the KLT-message representation and the generic update rules design.

The wide introduction including the problem statement and fundamental definitions are formally stated in Chapter 4, nevertheless this Chapter is written to be self-contained so that the introduction and definitions is briefly repeated.

5.1 Introduction

5.1.1 Background

The sum product algorithm on factor graphs (FG-SPA) provides a popular generic framework usable in a wide spectrum of problems across many fields such as communications, signal processing, etc. [27,33]. The aim of the FG-SPA consists mainly in an *efficient evaluation* of the decision and estimation problems in Bayesian networks.

The FG-SPA serves for an exact efficient inference evaluation algorithm, whenever the FG underlaying the problem is cycle-free. If the FG-SPA contains loop, it can still serve as a high-efficient evaluation algorithm (e.g. LDPC decoding), but the loops cause that the inference evaluation is no longer guaranteed to be the exact one. The FG-SPA becomes to be an iterative algorithm and its convergence issue arises. This convergence issue stands for a *principal nature problem* and its investigation is out of scope of this work (see e.g. [36] for an exhaustive overview of the looped FG-SPA convergence investigation possibilities).

Even if a message passing directed by the update rules [27] leads to a correct (not necessarily exact) marginal evaluation (inference), an efficient FG-SPA implementation is often not straightforward even if the update rules are clearly defined. The difficulty consists mainly in an infinite memory requested to store a general continuously valued messages and also in an evaluation often intractable marginalization integrals required in the update rules evaluation.

A commonly used way of the FG-SPA implementation is to represent (even approximately) the messages by several discrete parameters which are easy to be stored within the iteration process and to develop the update rules upon these parameters instead of the update rules upon the original messages. A natural goal of the FG-SPA implementation is to have

- the simplest possible message representation,
- a low-complex evaluation of the update rules upon the message representation,
- a minimal inaccuracy caused by the use of the message representation and the corresponding update rules.

[1] The stochastic process is formed by messages in different realizations and iterations.

Obtaining a proper message representation and the corresponding update rules is a crucial point in the FG-SPA implementation, because a wrong choice of the message representation easily leads to a need for a very complex messages and the update rules required for a given fidelity goal.

Since the message is represented by a pdf or pmf, it is conventional that messages are properly scaled such that the marginalization of the message is normed to 1. This normalization is can be easily done for pdf or pmf. The message representation has to somehow cope this normalization.

5.1.2 Related Work

We can find several message representation - update rules design frameworks in literature. Some of them are discussed in greater detail in Sec. 4.2.5. An example of the FG-SPA implementation improvement in case of LDPC decoding over GF(2) consists in the use of single-valued LLR values instead of the binary pmf as the message representation and the corresponding update rules can be found in [27]. In this case, the LLR-based FG-SPA implementation gives equal[2] results as the direct FG-SPA implementation with better efficiency.

In the previous case, we had still choice to directly implement the FG-SPA, since all considered messages were discretely valued. If the messages are continuously valued (represented by a pdf), a message parametrization is unavoidable. Nevertheless, there exists a straightforward parameterization for a family of continuously-valued messages (e.g. Gaussian message fully represented by its mean and variance). The update rule upon the Gaussian messages remains the Gaussian one, whenever the update is linear (only superposition and addition factor nodes are allowed). In such a case, it is sufficient to find a relation between means and variances. An exhaustive overview of the Gaussian messages use in the Linear models can be found in [32].

The easily parameterizable case is, however, a very specific one. Whenever the FG contains both the discretely and continuously valued messages, an appearance of a *mixed-type* message becomes unavoidable (see Fig. 4.3 for an illustration). The mixed-type message is a continuously valued message arising from the mixed marginalization, i.e. marginalization over both discrete and continuously valued messages. The problem is with the arising number of parameters in the iterative process requested for the mixed-type message parameterization. An example of this problem is joint data-detection phase-estimation, where a discrete-nature data are mixed with a continuous-nature phase shift.

The implementation of the FG-SPA containing the mixed type message appears several time in the literature. The most straightforward method is sampling of the continuously valued message. The update rules provided upon the discretely valued sampled messages stand for a numerical integration with rectangular rule (e.g. [12,13]). An another approach consists in the representation of the message by a Kronecker-delta function posed into maximum of the message. The update rules are (e.g. gradient methods) are proposed in [12,13]. These update rules are built on the non-linear optimization basis. We can also mention particle method, where the message is represented by (weighted) particles (see e.g. [13] for details).

Several works (e.g. [28, 52, 55]) solve the increasing number of parameters required for the mixed-type message representation by an approximation of the mixed-type message by a properly selected well parameterizable function. The largest attention attracts the Gaussian-form pdf because of its implementation simplicity.

A unified framework to the message representation issue can be found in [63]. The authors propose that a general message may be represented using a set of kernel functions and some coefficients related to the kernel (canonical) functions. This framework does, however, not solve a way of the kernel functions obtaining and also the update rules design is not solved there.

We can found several works deriving the canonical kernels and related update rules for a particular system model. As an example, the authors in [5] propose Fourier and Tikhonov message parameterization

[2] Excepting numerical inaccuracies.

and corresponding update rules for the joint data decoding phase estimation problem. This way of the canonical kernels obtaining is however derived for the concrete model, no *methodical way* is proposed.

5.1.3 Summary of Results - Contribution

The additional value of this work can be summarize as follows:

1. We propose a *methodical* way of the canonical kernels obtaining for an arbitrary FG-SPA implementation, where the KLT is defined upon the process formed by different message realizations.
2. The KLT-based procedure serves as a connection between the message description complexity and the fidelity criterion in terms of the mean square error of the message approximation.
3. We design a generic efficient implementation of the update rules design for linear canonical message representations with orthogonal kernels.
4. The orthogonality of the canonical kernel functions guaranteed by the KLT allows to state a generic FG-SPA implementation framework, where the messages are represented by the KLT-message representation[3] and the update rules are given[4] by the proposed generic efficient implementation.
5. We numerically verify the proposed design.

5.1.4 Summary of Notation

We briefly remind the basic notation. See Chapter 4 for full description.

We denote a message from a node F to x by $\mu^{F\to x}(x) \in \mathbb{M}^{F\to x}$. The notation of this message and its domain can be simplified to $\mu^{F}(x) \in \mathbb{M}^{F}$, $\mu^{x}(x) \in \mathbb{M}^{x}$ or just to $\mu(x) \in \mathbb{M}$ if the notation remains unambiguous. The superscripts are simplified whenever it is possible without lost of unambiguity or they are not important for that moment. Other parameters of a message are shown in subscript e.g. $\mu_k(t)$ as the k^{th} iteration of the message $\mu(t)$. The upper-case letters denote the factor nodes and the lower-case letters denote the variable nodes.

We assume a set of square integrable real-valued functions defined on an interval $I \subseteq \mathbb{R}$ to be a domain of the *continuously-valued* messages. The domain of the *discretely-valued* messages is supposed to be a $\mathcal{D}$-tuple, where $\mathcal{D}$ denotes dimensionality of the message.

5.2 FG-SPA Implementation Framework

We now propose the FG-SPA implementation framework, where we employ a LCMR with orthonormal kernels resulting from the KLT. We then derive the update rules for a general LCMR with otrhogonal kernels that forms a complete implementation framework with the proposed KLT message representation.

5.2.1 KLT Message Representation

We assume a FG-SPA without any implementation issues with a given scheduling algorithm. The particular shape of each message depends on (1) the iteration number k and (2) a random observation input x. The true message[5] describing a variable T in a given iteration and with given observation is denoted by $\mu_{x,k}(t)$,

[3] Other representations (Gaussian, Discrete, ...) can be also considered within the FG-SPA.

[4] As in case of the message representation, the update rules can be also performed by an arbitrary other way.

[5] Note that we assume a continuously valued message within this derivation, concretely the message from L_2 space. When the message is discretely valued, the proposed results can be achieved by equal steps respecting properly the discrete domain.

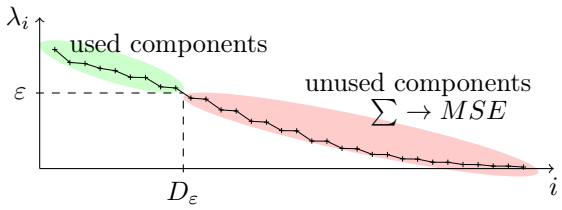


Figure 5.1: Demonstration of the KLT message representation fidelity-complexity trade-off handling. The residual MSE is indeed easily controlled by the sum of the eigenvalues corresponding to the unused (truncated) components.

where we assume the message to be a real valued function of argument $t \in I \subseteq \mathbb{R}$. This message can be approximated using a canonical kernel set $\{\chi_i(t)\}_i$ by

$$\mu_{x,k}(t) \approx \hat{\mu}_{x,k}(t) = \Phi(\boldsymbol{q}(x,k)) = \sum_i q_i(x,k)\chi_i(t). \tag{5.1}$$

The expansion coefficients $[\dots, q_i(x,k), \dots]$ fully describing the approximated message $\hat{\mu}_{k,x}(t) \in \hat{\mathbb{M}}$ are random.

The KLT of the process formed by the messages $\mu_{x,k}(t)$ for random iteration and observation input uses the second order statistics - the autocorrelation function $r_{xx}(s,t)$ to create an orthogonal kernel set with uncorrelated expansion coefficients and the second order moments are directly related with the residual mean square error.

The autocorrelation is function given by

$$r_{xx}(s,t) = \mathrm{E}_{x,k}\left[(\mu_{x,k}(s) - \mathrm{E}_{x,k}\left[\mu_{x,k}(s)\right])(\mu_{x,k}(t) - \mathrm{E}_{x,k}\left[\mu_{x,k}(t)\right])\right], \tag{5.2}$$

where $\mathrm{E}_x[\cdot]$ denotes expectation over x and $(s,t) \in I^2$. The solution of the eigenequation given by

$$\int_I r_{xx}(s,t)\chi(s)ds = \lambda\chi(t) \tag{5.3}$$

produces the eigenfunctions $\{\chi_i(t)\}_i$ serving for the *orthogonal canonical kernel set* and the eigenvalues $\{\lambda_i\}_i$ serving for the importance weight of the corresponding message representation component in terms of the mean square error.

The expansion coefficients can be evaluated using the orthogonal property of the canonical kernel set by

$$q_i(x,k) = \Phi^{-1}(\mu_{x,k}(t)) = \int_I \mu_{x,k}(t)\chi_i(t)dt. \tag{5.4}$$

These coefficients serve for the approximated message description by (5.1).

The complexity (dimensionality) of the message representation can be further reduced by omitting several components $\{q_i\chi_i(t)\}$. The components with index $i > \mathcal{D}_\varepsilon$ are dropped to reduce the representation complexity. We denoted $\mathcal{D}_\varepsilon$ the number of used components (dimensionality of the message) that refers to mean square error ε. The resulting mean square error of the approximated message

$$\hat{\mu}_{x,k} = \sum_{i=0}^{\mathcal{D}_\varepsilon} q_i(x,k)\chi_i(t) \tag{5.5}$$

is indeed given by the term $\varepsilon = \sum_{i>\mathcal{D}_\varepsilon} \lambda_i$ as one can check in Fig. 5.1.

Note 5.1 (Linear Canonical Message Representation)**.** Thanks to the KLT, the kernels are orthogonal and the representation is linear. It means that the KLT message representation is a *linear canonical message representation with orthogonal kernels* as we discussed in Note 4.11. Therefore both mappings determining the parameterization Φ, Φ^{-1} are clearly given.

5.2.2 Evaluation of the KLT Approximated Message in Real Applications

We have clearly described the procedure of the KLT message representation obtaining. Nevertheless the evaluation of the autocorrelation function (5.2) can be very complex. This can be simplified considering the approximation of the continuously valued message given by a discrete vector resulting from a sufficient sampling of the message[6]. The estimated autocorrelation function is given by

$$R_{xx} = \mathrm{E}_{x,k}[\boldsymbol{\mu}_{x,k}\boldsymbol{\mu}_{x,k}^T] \approx \frac{1}{KM}\sum_{k=1}^{K}\sum_{x=1}^{M}\boldsymbol{\mu}_{x,k}\boldsymbol{\mu}_{x,k}^T, \tag{5.6}$$

where K is the number of iteration, M is the number of realizations and $\boldsymbol{\mu} = [\dots, \mu_{x,k}[i], \dots]^T$ is a discrete vector referring to the sampled message $\mu_{x,k}(t)$.

The eigenequation (5.3) of the discrete message is given by

$$R_{xx}\boldsymbol{\chi} = \lambda\boldsymbol{\chi}. \tag{5.7}$$

The vector $\boldsymbol{\chi}$ refers to the sampled kernel functions $\chi(t)$. The expansion coefficients are given by

$$q_i(x,k) = \sum_j \boldsymbol{\mu}_{x,k}[j]\boldsymbol{\chi}_i[j]. \tag{5.8}$$

Finally the sampled version of the approximated message is given by

$$\boldsymbol{\mu}_{x,k} \approx \hat{\boldsymbol{\mu}}_{x,k} = \sum_i q_i(x,k)\boldsymbol{\chi}_i. \tag{5.9}$$

The truncation of the coefficients can be performed equally as in the continuous case.

5.2.3 Generic Update Rules of the Canonical Message Representation with Orthogonal Kernels

We call a Linear Canonical Message Representation (LCMR) such a message (see Note 4.10) that the message can be written in the form

$$\mu^{F\to x}(x) \approx \hat{\mu}^{F\to x}(x) = \sum_{i=1}^{\mathcal{D}(F\to x)} q_i^{F\to x}\chi_i^{F\to x}(x), \tag{5.10}$$

We remind that the proposed KLT message representation (Sec. 5.2.1) is LCMR as we already mentioned in Note. 5.1.

We now derive the FG-SPA update rules for a general LCMR (5.10) with orthogonal canonical kernels *upon the parameterization*, that is a direct relation between the parameters of the message as we discussed in Sec. 4.2.3. We derive the FG-SPA update rules for the LCMR assumed in a whole or in a part of the system. The coexistence of more message representations in the FG-SPA requires interfaces with another message representations. Particularly, we investigate

[6]The complexity issue of the autocorrelation function evaluation is not complication at all, because it is sufficient to estimate the autocorrelation function just once prior the processing. The required number of samples enabling a full message recovery from samples is given by the sampling theorem.

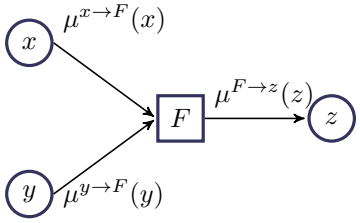


Figure 5.2: Visualization of the FN updated message $\mu^{F\to z}$.

- variable node (VN) update rules with the LCMR,
- factor node (FN) update rules
  - with the LCMR only,
  - with the LCMR and the representation by samples,
  - with the LCMR and an another representation.

We also address our attention to a proper scaling of the approximated messages and we point out the exact-form updated-message rectification problem, when the approximated message is negative (non-positive) at some points.

For simplicity, we will assume only three neighboring nodes. It means that for a FN F update, we consider just three neighboring variable nodes x, y and z (see Fig. 5.2) and for a VN x update, we consider just three neighboring factor nodes F, G and H (see Fig. 5.3). An extension to a general case with an arbitrary number of neighbors is straightforward.

Factor Node Update Rules

We first investigate the update rules related to factor nodes. We start with a conventional FN update rule formula (4.6), [27]

$$\begin{aligned}\mu^{F\to z}(z) &= \int p(z|y,x)\mu^{x\to F}(x)\mu^{y\to F}(y)dxdy \\ &\approx \int p(z|y,x)\sum_i q_i^x\chi_i^x(x)\sum_j q_j^y\chi_j^y(y)dxdy \\ &= \sum_i\sum_j q_i^x q_j^y\int p(z|y,x)\chi_i^x(x)\chi_j^y(y)dxdy,\end{aligned}$$

where we used (5.10) to express the approximated messages.

Our target is to obtain the expansion coefficients q_k^z of the approximated message $\hat{\mu}^{F\to z}(z)=\sum_k q_k^z\chi_k^z(z)$. We exploit the orthogonality of the kernels to evaluate the expansion coefficient by means of the inner product as

$$q_k^z = \langle\hat{\mu}^{F\to z}(z),\chi_k^z(z)\rangle = \sum_i\sum_j q_i^x q_j^y w_{i,j,k}^F, \tag{5.11}$$

where[7] $w_{i,j,k}^F=\int p(z|y,x)\chi_i^x(x)\chi_j^y(y)\chi_k^z(z)dxdzdy$ is a table determined by given kernels and a conditional probability density function (pdf) $p(z|y,x)$. The table is parameterized by i,j,k and it can be precomputed prior the processing.

[7] Since we assume real-valued kernel functions and real expansion coefficients, we drop the complex conjugation related to the inner product.

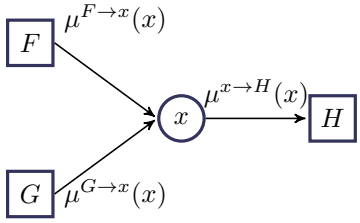


Figure 5.3: Visualization of the VN updated message $\mu^{x\to H}$.

We stress that (5.11) indeed relates the parameterization of the input and output messages as we requested in Sec. 4.2.4 to the implementation framework.

Variable Node Update Rules

The investigation of the VN update rules can be conducted similarly. We start again with the definition of the VN update rule according to [27]

$$\begin{aligned}\mu^{x\to H}(x) &= \mu^{F\to x}(x)\mu^{G\to x}(x)\\ &\approx \sum_i q_i^F \chi_i^F(x) \sum_j q_j^G \chi_j^G(x).\end{aligned} \tag{5.12}$$

The expansion coefficients of the approximated message are evaluated again by means of the inner product by

$$q_k^H = \langle \hat{\mu}^{x\to H}(x), \chi_k^H(x)\rangle = \sum_i \sum_j q_i^F q_j^G w_{i,j,k}^V, \tag{5.13}$$

where the update table $w_{i,j,k}^V = \int \chi_i^F(x)\chi_j^G(x)\chi_k^H(x)dx$ is given purely by the sets of the kernel functions and it can be precomputed prior the processing.

Factor Node Update with both Discretely and Linearly Represented Continuous Messages

We now consider a case of a mixed updating, when a FN F is neighboring both discretely and continuously valued messages. Particularly, the variable y is a discretely valued and the message $\mu^{y\to F}(y)$ is represented by pmf of y. The variables x and z are continuously valued.

Our goal is to evaluate the LCMR of the message $\mu^{F\to z}(z)$ from the LCMR $\mu^{x\to F}(x)$ and discretely valued message $\mu^{y\to F}(y) = [\dots, q_j^y, \dots]$ and conversely the discrete message $\mu^{F\to y}(y)$ from the LCMR $\mu^{z\to F}(z)$ and $\mu^{x\to F}(x)$.

The derivation is very similar as in Sec. 5.2.3. We shortly write the FN-update as (see Fig. 5.2)

$$\mu^{F\to z}(z) = \int \sum_j p(z|y_j, x)\mu^{x\to F}(x)\mu^{y\to F}(y)dx \tag{5.14}$$

$$\approx \int \sum_j p(z|y_j, x) \sum_i q_i^x \chi_i^x(x) q_j^y dx \tag{5.15}$$

and the expansion coefficients q_k^z of the approximated message $\hat{\mu}^{F\to z}(z) = \sum_k q_k^z \chi_k^z(z)$ are evaluated as

$$q_k^z = \langle \hat{\mu}^{F\to z}(z), \chi_k^z(z)\rangle = \sum_i \sum_j q_i^x q_j^y w_{i,j,k}^F, \tag{5.16}$$

where $w^F_{i,j,k} = \int p(z|y_j, x)\chi^x_i(x)\chi^z_k(z)dxdz$ is a precomputed table for given kernel sets and the conditional pdf $p(z|y_j, x)$.

After a very similar derivation, we obtain the update of the discrete message $\mu^{F\to y}(y)$ by

$$q^y_j = \sum_i \sum_k q^x_i q^z_k w^F_{i,j,k}. \tag{5.17}$$

This update is, however, not guaranteed to be a non-negative number for all j as one requests to pmf. A rectification of the updated message is briefly discussed in Sec. 5.2.3.

Interface with Other Message Representations

We have shown a closed update-relation for linearly represented continuously valued messages and discrete messages represented by their pmf. Nevertheless, it may be still useful to have two different representations of a continuously valued message. In such a case, we split this VN to two variable nodes and insert an interface factor node between them.

Assume a VN $\bar{x}$ connected to two parts of a FG-SPA. The first part of the FG-SPA operates with an arbitrary representation of the continuously valued message (e.g. Gaussian) and the second one is assumed to be represented by the LCMR with orthogonal kernels. The VN $\bar{x}$ is split to variable nodes x and y, where a FN F is inserted between them. It is assumed that the VN x is represented by the LCMR with orthogonal kernels and the VN y by an arbitrary another message representation capable to describe the continuously valued message.

The node F is described by the conditional (pdf) given by $p(y|x) = \delta(y - x)$ and the expansion coefficients q^x_i of the LCMR message $\hat{\mu}^{F\to x}(x) = \sum_i q^x_i \chi^x_i(x)$ are evaluated as

$$q^x_i = \langle \hat{\mu}^{F\to x}(x), \chi^x_i(x)\rangle = \int \mu^{y\to F}(x)\chi^x_i(x)dx. \tag{5.18}$$

The message $\hat{\mu}^{y\to F}(x)$ is used if $\mu^{y\to F}(x)$ is not available. If the message $\hat{\mu}^{y\to F}(x)$ is parameterized by a vector $\boldsymbol{p}$, one can avoid the integration during the processing by precomputing a set of functions $\left\{\Psi_i(\boldsymbol{q}) = \int \hat{\mu}^{y\to F}(x;\boldsymbol{q})\chi^x_i(x)dx\right\}_i$.

Conversely, the corresponding message representation is evaluated from the LCMR approximated message $\hat{\mu}^{x\to F}(x) = \sum_i q^x_i \chi^x_i(x)$ by means of the particular message representation. We can mention e.g. Gaussian representation given by

$$[m^y, \sigma^y] = \left[\mathrm{E}_x[\hat{\mu}^x(x)], \mathrm{E}_x[(\hat{\mu}^x(x))^2] - \mathrm{E}_x[\hat{\mu}^x(x)]^2\right], \tag{5.19}$$

where $\mathrm{E}_x[\mu(x)] = \int x\mu(x)dx$ denotes expectation over x. One can find out that

$$\mathrm{E}_x[(\hat{\mu}^x(x))^n] = \sum_i (q_i)^n \mathrm{E}_x[(\chi_i(x))^n]$$

and $\mathrm{E}_x[(\chi_i(x))^n]$, $\forall i, n$ can be again precomputed.

Norming and Rectifying the Approximated Message

Since the original message $\mu(t)$ is supposed to represent a pdf, we may not care about the message scaling during the update rules. When the message is approximated, we need to be able to achieve $\int \hat{\mu}(t)dt = 1$. Suppose that $\hat{\underline{\mu}}(t) = \sum_k \underline{q}_k \chi_k(t)$ is a non-normed LCMR. We consider a norming constant $\alpha \in \mathbb{R}$ such that $\int \alpha\hat{\underline{\mu}}(t)dt = \int \hat{\mu}(t)dt = 1$.

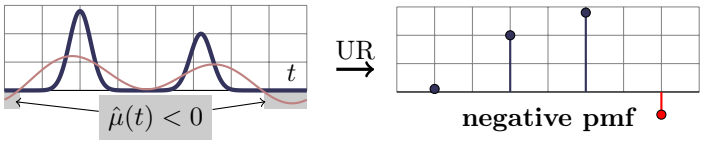


Figure 5.4: Demonstration of the FG-SPA updated exact-form negative message problem.

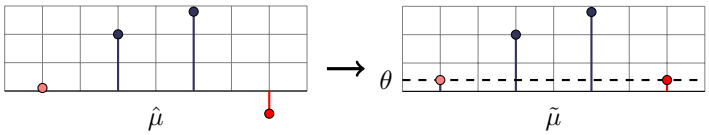


Figure 5.5: Rectification of the negative message appearance.

We can express $\alpha = 1/\int \underline{\hat{\mu}}(t)dt = 1/\sum_i P\underline{q}_i\Omega_i$, where we defined $\Omega_i = \int \chi_i(t)dt$. Since $\alpha\underline{\hat{\mu}}(t) = \hat{\mu}(t)$ surely guarantees a proper norming, we can express the norming expression as

$$q_i = \alpha\underline{q}_i = \frac{\underline{q}_i}{\sum_j \underline{q}_j\Omega_j}, \forall i. \tag{5.20}$$

As a consequence of the message approximation, the approximated message $\hat{\mu}(t) = \sum_k q_k\chi_k(t)$ can be negative at some points. The messages are conventionally assumed to be non-negative in the FG-SPA. The negativeness of the approximated message can be ignored, when the message is used to evaluate an another *approximated* message, because the non-negative request is casted to the true (exact-form) messages. The approximated (or the updated) message must, however, be somehow rectified in case of the exact-form updated message (see Fig. 5.4). There is a number of ways of the negative message rectification (e.g. subtracting $\min_t(\hat{\mu}(t))$ from the approximated message and norming it properly as ad-hoc suggested in [44]).

Motivated by our simulation results in Sec. 5.3.5, it is beneficial to avoid a hard decision provided by the FN-update rule upon the approximated message. It means that the updated exact-form message should not contain any zero element (e.g. discrete message represented by pmf should satisfy $p_j > 0, \forall j$, if it is a result of the FN-update with an approximated message input). Following this simple idea, we add-hoc suggest to define a softness threshold θ and the rectified updated exact-form message $\tilde{\mu}(t) = \hat{\mu}(t)$ if $\hat{\mu}(t) \geq \theta$ and $\tilde{\mu}(t) = \theta$ otherwise (see Fig. 5.5).

We numerically find out a very high sensitivity of the performance with respect to the updated-message rectification in a specific scenario (Sec. 5.3.5). With the increasing precision of the message representation the impact of this phenomenon will decrease.

5.3 Applications and Numerical Verification

We exemplify the proposed FG-SPA implementation framework on the well-know case of the joint phase estimation data detection. Various FG-SPA implementation frameworks suited for this case can be found in literature (e.g. [10, 12, 52, 53, 55]). We stress that while all mentioned frameworks are designed for this particular scenario, the proposed implementation framework is more *generic*.

The verification consists from two parts. First, we obtain the canonical kernel functions of the KLT message representation. This evaluation is based on the stochastic analysis of the messages in the FG-SPA (Sec. 5.2.2). The impact of this approximation is investigated and compared with other message representations mentioned in Sec. 4.2.5. The second part investigates behavior of a whole implementation framework.

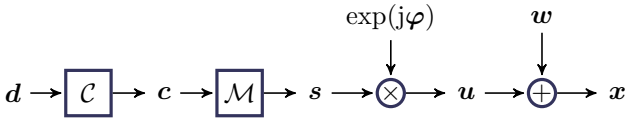


Figure 5.6: The system models in signal space.

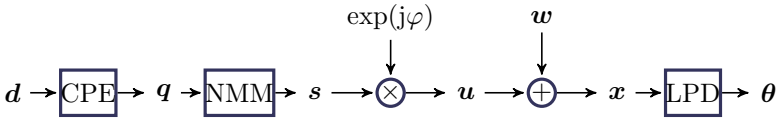


Figure 5.7: The MSK modulation in phase space.

5.3.1 System Model

We consider several system model variants to achieve a more complex view into the system. This is desired mainly for the stochastic analysis of the messages in the phase model. We consider several linear modulations models in signal space space and a non-linear minimal shift keying (MSK) modulation in the phase space model. We also consider two phase models, the constant phase and the random walk (RW).

Signal Space Models

We assume a binary i.i.d. data vector $\boldsymbol{d} = [d_1, \ldots, d_n]^T$ of length N as an input. The coded symbols are given by $\boldsymbol{c} = \mathcal{C}(\boldsymbol{d})$. The modulated signal vector is given by $\boldsymbol{s} = \mathcal{M}(\boldsymbol{c})$, where $\mathcal{M}$ is a signal space mapper. The channel is selected to be the AWGN channel with phase shift modeled by the random walk (RW) phase model. The phase as the function of time index is described by $\varphi_{j+1} = mod(\varphi_j + w_\varphi)_{2\pi}$, where w_φ is a zero mean real Gaussian random value (RV) with variance σ_φ^2. Thus the received signal is $\boldsymbol{x} = \boldsymbol{s}\exp(\mathrm{j}\boldsymbol{\varphi}) + \boldsymbol{w}$, where $\boldsymbol{w}$ stands for the vector of complex zero mean Gaussian RV with variance $\sigma_w^2 = 2N_0$. The model is depicted in Fig. 5.6.

MSK in Phase Space

We again assume the vector $\boldsymbol{d}$ as an input into the minimum-shift keying (MSK) modulator. The modulator is supposed to be in the canonical form, i.e. it is modelled by the continuous phase encoder (CPE) and non-linear memoryless modulator (NMM) as shown in [50]. The modulator is implemented in the discrete time with two samples per symbol. The phase of the MSK signal is given by $\phi_j = \pi/2\,(\varsigma_i + d_i(j-2i)/2) \bmod_4$, where ϕ_j is the j^{th} sample of the phase function, ς_i is the i^{th} state of the CPE and the sampled modulated signal is given by $s_j = \exp(\mathrm{j}\phi_j)$. The communication channel is selected to be the AWGN channel with constant phase shift, i.e. $\mathbf{x} = \mathbf{s}\exp(\mathrm{j}\varphi) + \mathbf{w}$, where $\mathbf{x}$ stands for the received vector, φ is the constant phase shift of the channel and $\boldsymbol{w}$ is the AWGN vector. The nonlinear Limiter Phase Discriminator (LPD) captures the phase of the vector $\mathbf{x}$, i.e. $\theta_j = \angle(x_j)$. The whole system is shown in Fig. 5.7.

5.3.2 Factor Graph of the Joint Phase Estimator Channel Decoder

The FG-SPA is used as a joint synchronizer-decoder (see Fig. 5.8) for all mentioned models. Note that the FG for the considered models might be found in the literature (phase space model in [55] and signal space models in [10, 12]).

Prior the description itself, we found a notation to enable a common description of the models. We define

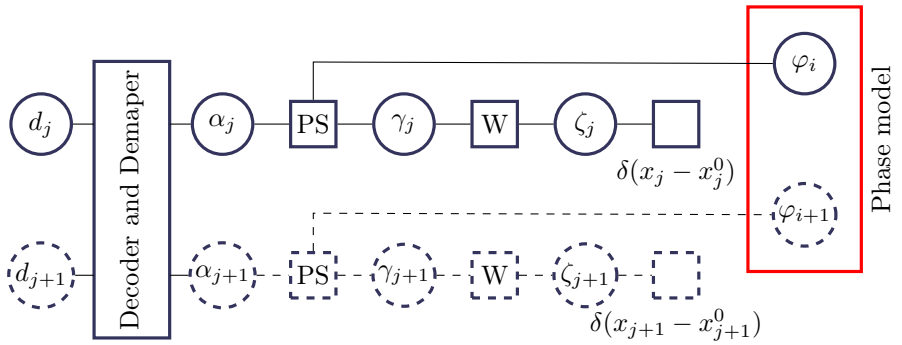


Figure 5.8: The factor graph of the receiver covering both phase models.

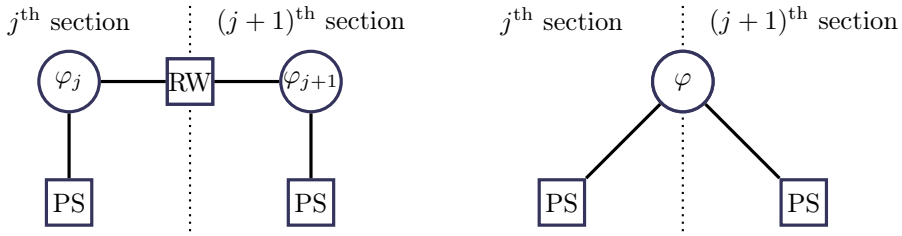


Figure 5.9: Phase shifts models: Random walk model (left) and the constant phase shift model (right).

- $\alpha_j = s_j$, $\gamma_j = \alpha_j \exp(\mathrm{j}\varphi_j)$ and $\zeta_j = x_j$, where α_j, γ_j, $\zeta_j \in \mathbb{C}$ for the signal space models with the RW phase model,
- $\alpha_j = \angle(s_j)$, $\gamma_j = (\alpha_j + \varphi_j) \mathrm{mod}_{2\pi}$ and $\zeta_j = \theta_j$, where α_j, γ_j, $\zeta_j \in \mathbb{R}$ for the phase space model with the constant phase model.

One can see, that φ_j, $\varphi \in \mathbb{R}$ for both models. The FG is depicted in Fig. 5.8. We shortly describe the presented factor nodes and message types presented in the FG.

Factor Nodes

We denote $\rho_{\sigma^2}(x-y) = \exp(-|x-y|^2/(2\sigma^2))/\sqrt{2\pi\sigma^2}$ and then we use $p_\Xi(\sigma^2, \xi - \varkappa)$ as the phase distribution of the RV given by $\Xi = \exp(j(\xi - \varkappa)) + w$, where w stands for the zero mean complex Gaussian RV with variance σ_w^2; $x, y \in \mathbb{C}$ and $\xi, \varkappa \in \mathbb{R}$. The phase models are shown in Fig. 5.9.

- Factor nodes in the signal space models

Phase Shift (PS):

$$p(\gamma_j|\varphi_j, \alpha_j) = \delta(\gamma_j - (\alpha_j \exp(j\varphi_j)))$$

AWGN (W):

$$p(\zeta_j|\gamma_j) = \rho_{\sigma_w^2}(\zeta_j - \gamma_j)$$

- Factor nodes in the phase space model

Phase Shift (PS):

$$p(\gamma_j|\varphi, \alpha_j) = \delta(\gamma_j - (\alpha_j + \varphi) \mathrm{mod}_{2\pi})$$

AWGN (W):

$$p(\zeta_j|\gamma_j) = p_\Xi(\sigma_W^2, \zeta_j - \gamma_j)$$

- Factor nodes common for both signal and phase space *random walk (RW)*:

$$p(\varphi_{j+1}|\varphi_j) = \sum_{l=-\infty}^{\infty} \rho_{\sigma_\varphi^2}(\varphi_{j+1} - \varphi_j + 2\pi l)$$

Other factor nodes such as the coder, CPE, and signal space mappers FN are situated in the discrete part of the model and their description is obvious from the definition of the related components (see e.g. [27] for an example of such a description).

Message Types Presented in the FG-SPA

The FG-SPA contains both discretely and continuously valued messages. The channel encoder and the signal space mapper contains only the discretely valued messages. The messages are represented by their pmf the update rules can be directly implemented according their definitions.

The AWGN part contains well parameterizable continuously valued messages and the phase model contains the mixed-type messages. These mixed type messages are represented by means of the implementation framework (KLT message representation in our case).

5.3.3 Concrete Scenarios

We specify several concrete scenarios for the analysis purposes. All of the scenarios may be seen as a special case of the system model described in the section 5.3.1. The first four configurations are used for the stochastic analysis of the mixed type message (Sec. 5.3.4) and the fifth one is used for the implementation of the whole implementation framework including the KLT message representation and the proposed update rules implementation (Sec.5.3.5).

Uncoded QPSK modulation

The Uncoded QPSK modulation scheme assumes the QPSK modulation, which is situated in the AWGN channel with RW phase model. The length of the frame is $N = 24$ data symbols, the length of the preamble is 4 symbols and the preamble is situated at the beginning of the frame. The variance of the phase noise equals $\sigma_\varphi^2 = 0.001$. This scenario leads to a *cycle-free* FG-SPA and thus only inaccuracies caused by the imperfect implementation are presented. The information needed to resolve the phase ambiguity is contained in the preamble and, by a proper selection of the analyzed message, we can maximize the approximation impact to the key metrics such as BER or MSE of the phase estimation. We thus select the message $\mu^{RW_- \to \varphi_3}$ to be analyzed.

Coded 8PSK modulation

The coded 8PSK modulation contains in addition to the previous scenario the $(7, 5, 7)_8$ convolutional encoder[8]. The length of the frame is $N = 12$ data symbols, the length of the preamble is 2 symbols. The same message is selected to be analyzed ($\mu^{RW_- \to \varphi_3}$).

MSK modulation

The MSK modulation with constant-phase model of the phase shift assumes the length of the frame is $N = 20$ data symbols. The analyzed message is $\mu^{\varphi \to PS}$. These messages are nearly equal for all possible PS factor nodes (e.g. [13]).

[8] We mean by $(7, 5, 7)_8$ a conventional octal definition of the convolutional encoder.

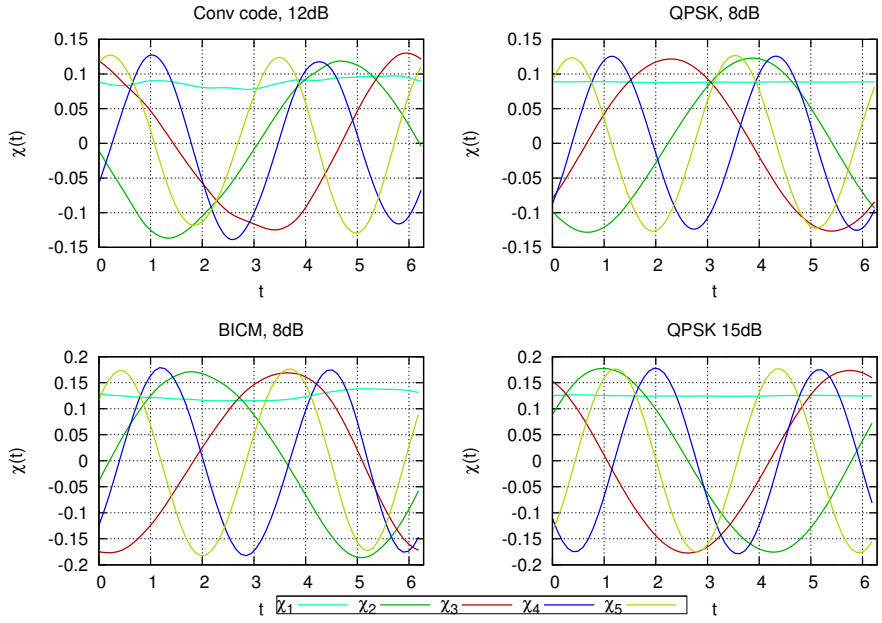


Figure 5.10: Shape of canonical kernel functions for different system configuration in phase space.

BICM 1

The bit-interleaved coded modulation (BICM) model employs the BICM consisting of $(7,5)_8$ convolutional encoder and QPSK signal-space mapper. The phase is modeled by the RW model with $\sigma_\varphi^2 = 0.005$. The length of the frame is $N = 1500$ data symbols and 150 of those are pilot symbols. This model slightly changes our concept. Instead of the investigation of the single message, we analyze all $\mu^{PS\to\varphi}$ and $\mu^{\varphi\to PS}$ messages jointly. It means that all of the analyzed messages are approximated in the simulations and the stochastic analysis is performed over all investigated messages.

BICM 2

We assume again the BICM. The model contains $(7,5)_8$ convolutional encoder, QPSK signal-space mapper, RW phase model with $\sigma_\varphi^2 = 0.004$, frame length $N = 120$ and preamble of length 15 pilot symbols.

5.3.4 Stochastic Analysis of the Mixed-Density Messages in FG-SPA Run

Once the FG-SPA is described, the stochastic analysis described in Sec.5.2.2 can be applied. We describe the reference model that aims with the collection of the mixed-type messages for the stochastic analysis.

The FG-SPA Reference Model

The empirical stochastic analysis requires a sample of the message realizations. Thus we ideally need a perfect implementation of the FG-SPA for each model. We call this perfect or better said almost perfect FG-SPA implementation [9] as the reference model. Note that the reference model might suffer (and our

[9] Note that even if the implementation of the FG-SPA is perfect, the convergence of the FG-SPA is still not secured in the looped cases. We call the perfect implementation such a model that does not suffer from the implementation related issues such as an update rules design and a messages representation. The flood schedule message passing algorithm is assumed.

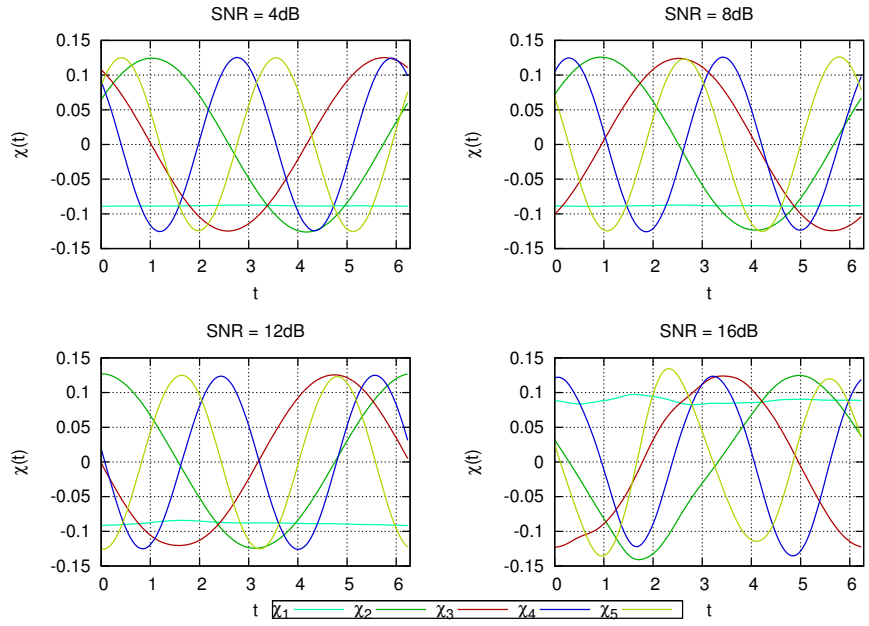


Figure 5.11: Shape of canonical kernel functions for the MSK modulation.

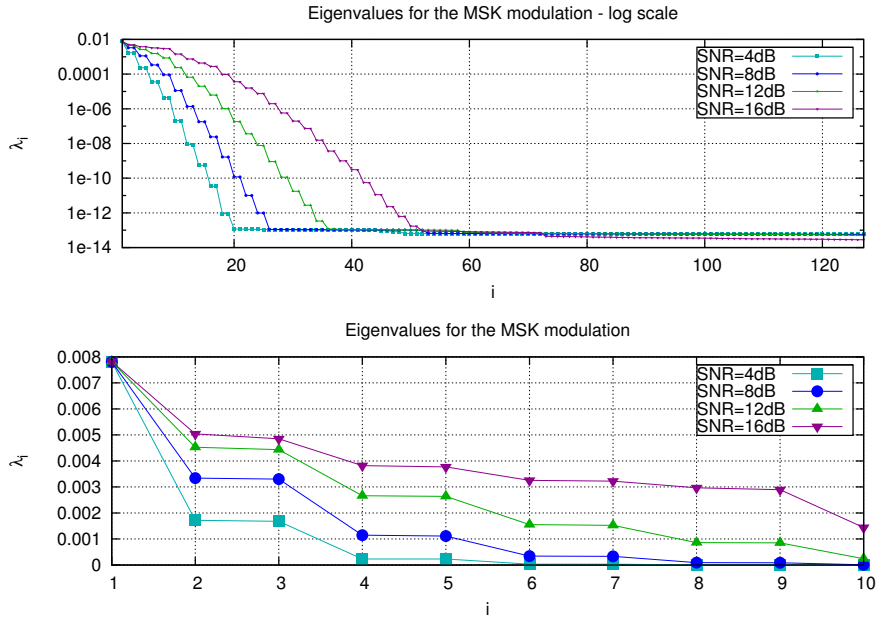


Figure 5.12: Eigenvalues descending for the MSK modulation.

models do) from the numerical complexity and it is therefore unsuitable for a direct implementation.

We classify the messages appearing in the reference FG-SPA model (Fig. 5.8) and their update rules.

- *Discrete messages* are all messages on left from the PS FN (see Fig 5.8). It means all messages in the encoder and mapper part. These discrete messages are fully represented by their pmf and the evaluation of the update rules is performed directly according to [27].
- *Unimportant messages* ($\mu^{PS\to\gamma_j}$, $\mu^{\gamma_j\to W}$, $\mu^{W\to\zeta_j}$ and $\mu^{\zeta_j\to\delta_j}$) lead to the open branch and neither an update nor a representation of them is required, because these messages cannot affect the estimation of the target parameters (data, phase).
- *Parameterizable continuous messages* ($\mu^{\delta_j\to\zeta_j}$ and $\mu^{\zeta_j\to W}$) can be easily represented by a number x_0 meaning $\mu(x)=\delta(x-x_0)$, $\mu^{W\to\gamma_j}$ and $\mu^{\gamma_j\to PS}$ are representable by the pair $\{m,\sigma_w^2\}$ meaning $\mu(x)=\rho_{\sigma_w^2}(x-m)$, $\mu(x)=p_\Xi(\sigma_W^2,x-m)$ respectively. One can easily find the slightly modified update rules derived from the standard update rules. Examples of those may be seen in [13].
- The *mixed-type messages* situated in the phase model ($\mu^{PS\to\varphi_j}$, $\mu^{\varphi_j\to PS}$, $\mu^{RW_+\to\varphi_j}$, $\mu^{\varphi_j\to RW_+}$, $\mu^{RW_-\to\varphi_j}$ and $\mu^{\varphi_j\to RW_+}$) are[10] discretized and the marginalization is performed using the numerical integration with the rectangle rule [13, 32] in the update rules. The number of samples is chosen sufficiently large so that the impact of the approximation can be neglected. The mentioned mixture messages are real valued one-dimensional functions for all considered models.

Eigensystem Analysis

We first provide the analysis of the eigensystem. For the investigated messages, we collect the sample realizations to obtain the second order statistics (the empirical autocorrelation function) according to (5.6). The eigenfunctions solving the eigenequation (5.7) result in harmonic functions independently of the parameters of the simulation as one can see in Fig. 5.10 and 5.11.

The related eigenvalues are shown[11] in Fig. 5.12. We can see several artefacts. The eigenvalues provide an important information about the importance of a particular message approximation component. The MSE resulting by dropping the component is given directly by the corresponding eigenvalue as discussed in Sec.5.2.1. The floor in Fig. 5.12 is caused by the finite floating precision. As one can see, the higher SNR, the slower is the descent of the eigenvalues with the dimension index. The curves in the plots point out to the fact that the eigenvalues are descending in pairs e.g. $\lambda_1-\lambda_2\gg\lambda_2-\lambda_3$.

Relation of the MSE of the Approximated Message with the Target Criteria Metrics

The KLT message representation provides the best approximation in the MSE sense for a given dimensionality. This is, unfortunately, not the target metric to be optimized in almost all cases. This relation is moreover not easy to analytically obtain. We therefore provide a numerical simulation to inspect the relation between the approximation error and the target metrics (MSE and BER) at least in several distinct system model configurations.

Few notes are addressed before going over the results. The MSE of the phase estimation is computed as an average over all MSE of the phase estimates in the model. The measurement of the MSE is limited by the granularity of the reference model. The simulations of the stochastic analysis are numerically intensive. We are limited by the computing resources. The simulations of the BER might suffer from this, especially for small error rates. The threshold of the detectable error rate is about $P_{be}^{min}\approx 10^{-6}$ for the Uncoded QPSK model and $P_{be}^{min}\approx 2\cdot 10^{-5}$ for the BICM model.

[10] We have denoted RW_+ the factor node RW, which lies between j^{th} and $(j+1)^{\text{th}}$ section according to the Fig. 5.9. Analogously, RW_- stands for the FN between $(j-1)^{\text{th}}$ and $(j)^{\text{th}}$ section.

[11] The results in other model seems very similarly and they are therefore not explicitly shown.

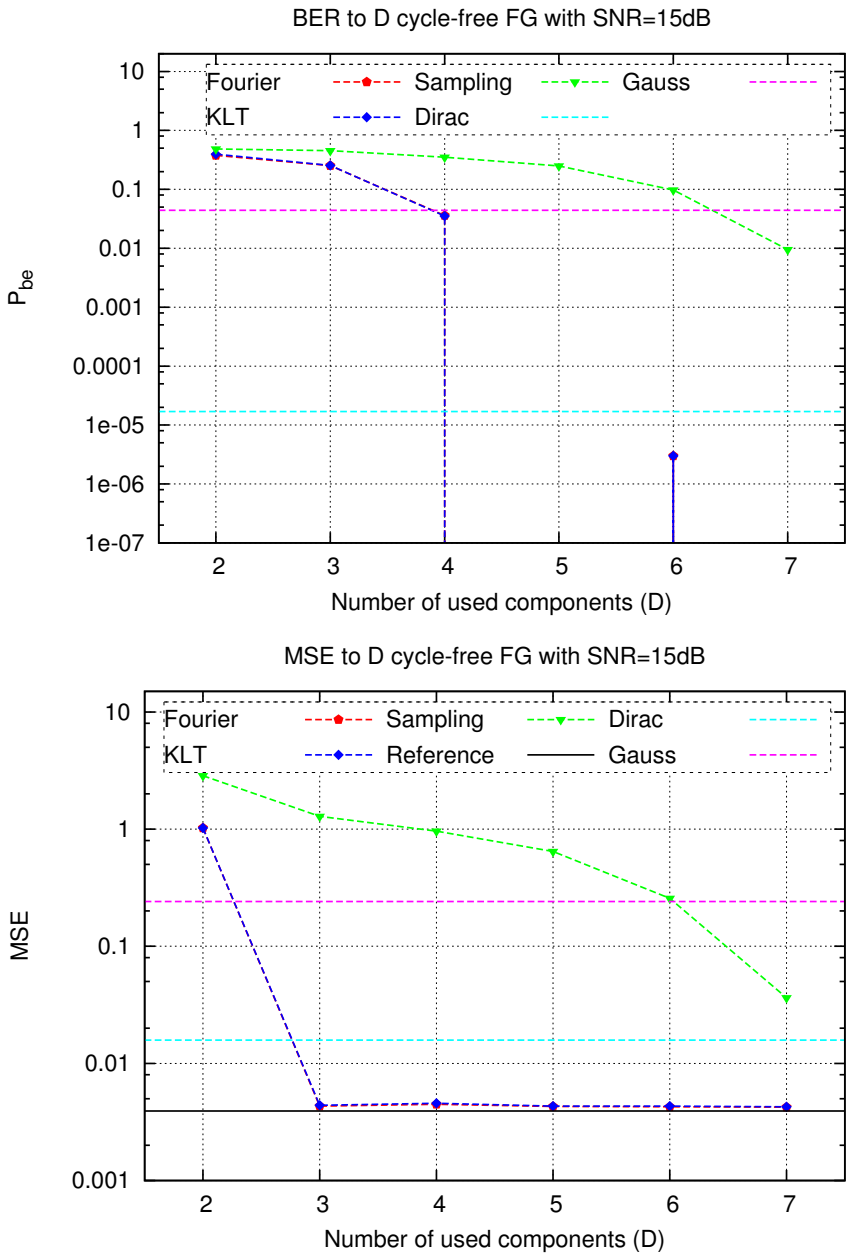


Figure 5.13: MSE and BER for the cycle-free FG-SPA (uncoded QPSK modulation with RW phase model).

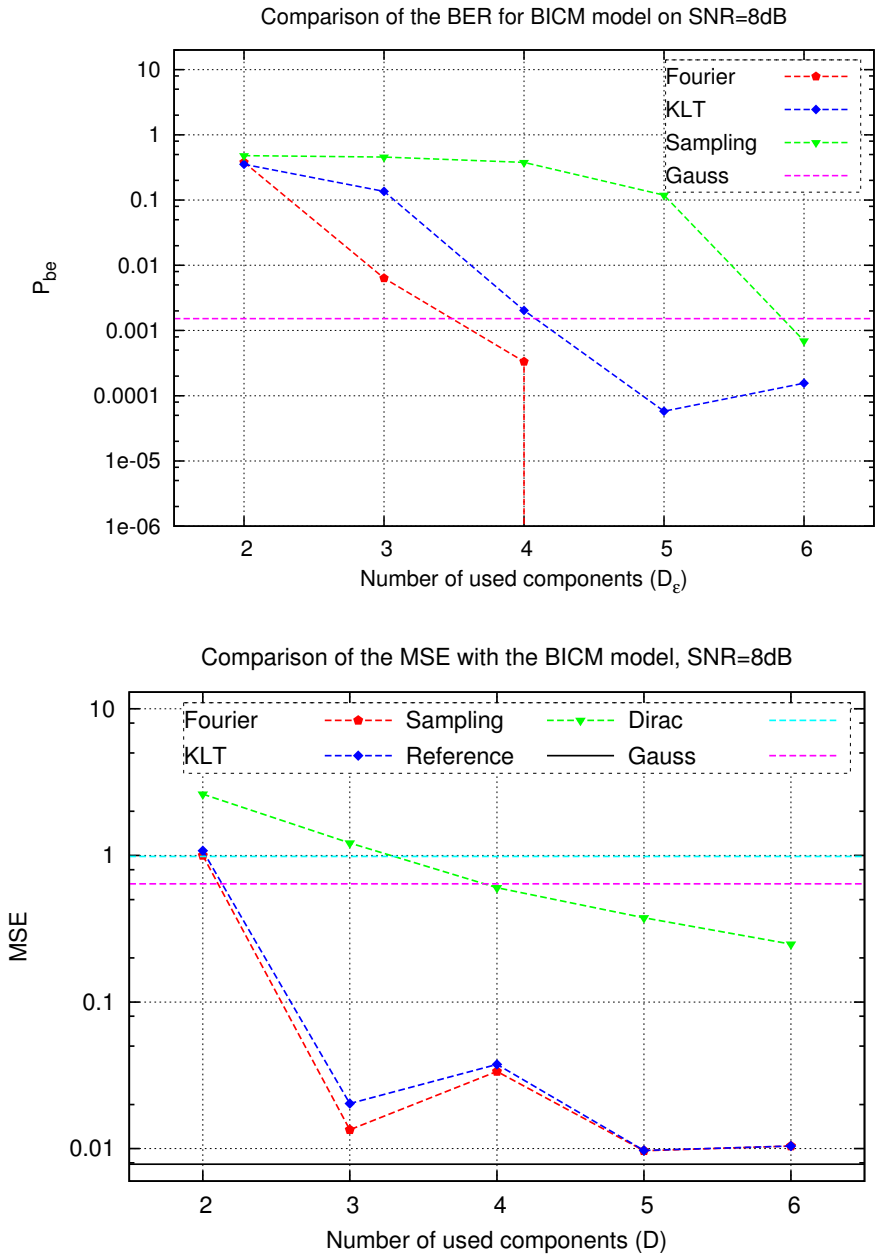


Figure 5.14: MSE and BER for the BICM with 8PSK modulation and RW phase model.

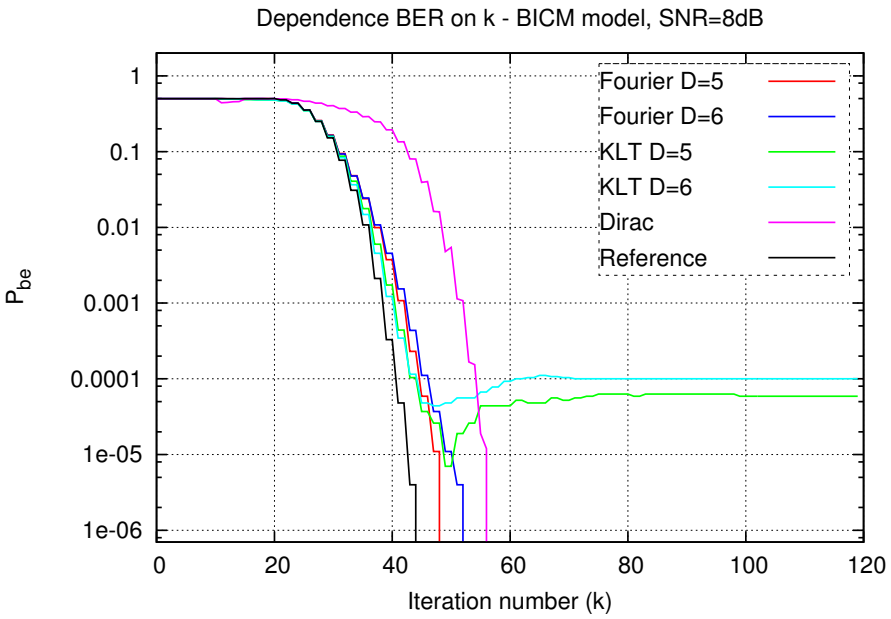


Figure 5.15: BER for the BICM with 8PSK modulation and RW phase model with dependence on the iterations considering the flood-schedule message passing algorithm.

- The results of the *uncoded QPSK modulation* are shown in Fig. 5.13. The capabilities of the proposed KLT message representation are equal with the capabilities of the Fourier representation. This is not any surprise taking into account the results of the eigenanalysis (Sec.5.3.4), but it means that the Fourier representation provides the best linear message representation in the MSE sense. The expected proportional relationship between the MSE of the approximated message and the target criteria does not strictly hold (compare BER of the five-component and six-component approximated message in Fig. 5.13). The discretization does not work well for small number of samples, because the higher SNR (15dB) brinks narrower shapes of the messages. Consequently the messages suffers by a wider spectral bandwidth and a higher amount of samples is required to satisfy the sampling theorem.

- The results of the *BICM* model at SNR=8dB are shown in Fig. 5.14 and 5.15. We can see that the KLT message representation lost a little bit against the Fourier representation. The KLT approximated message converges a little bit faster than the Fourier representation up to approx. 45th iteration, where the KLT approximated message achieves the error floor. We suggest two possible explanations of the error floor appearance (see Fig. 5.15). First, the eigenvectors constituting the basis system are not evaluated with a sufficient precision. Second, the evaluated KLT basis is the best linear approximation in average through all iterations and this basis is not capable to describe the messages appearing in the higher iterations sufficiently. The relationship between the MSE of the approximated message and the target criteria is not again strictly proportional (compare the 3-component and 4-component messages in Fig. 5.14). One can suggest that the eigenfunctions need to be in pair to increase a precision to the results.

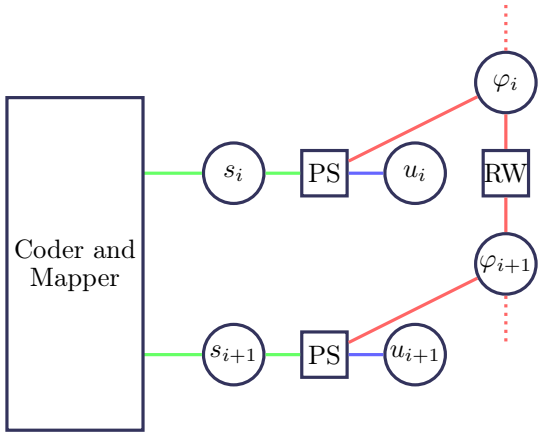


Figure 5.16: The factor graph of the decoder for the random walk phase model implemented in the signal space. This can be seen as a special case of Fig. 5.8, where we recognize the discrete type of the messages (green edges), parameterizable (Gaussian) continuous messages (blue edges) and mixed type messages (red edges).

5.3.5 Proposed Generic Update Rules Implementation

Within the implementation of the update rules, we consider the system model described in Sec.5.3.1 with the BICM modulation with RW phase model implemented in signal space. The factor graph of the receiver for this particular model can be found in Fig 5.16 as a special case of 5.8.

Canonical Kernel Functions Assignment

The results of eigenanalysis (Sec. 5.3.4 or Fig. 5.11 and Fig. 5.10) suggest to use the harmonic functions as the canonical kernel basis system for the mixed-type messages. Using the Fourier representation offers the best message approximation in the MSE sense for a given dimensionality. We therefore employ the Fourier message representation, where the approximated message is given by (4.22).

A proper scaling of the Fourier representation can be done using Sec. 5.2.3. One can check that $\Omega_k = \int_0^{2\pi} \chi_k(t)dt = 0, \forall k \neq 0$ and $\Omega_0 = \int_0^{2\pi} \chi_0(t)dt = 1/\sqrt{2\pi}$. The scaling factor α is thus given by $\alpha = \sqrt{2\pi}/\underline{q}_0$. The scaling of the approximated message to achieve $\int \sum_k q_k \chi_k(t)dt = 1$ can be easily done using (5.20), concretely by setting $q_k = \underline{q}_k \sqrt{2\pi}/\underline{q}_0$. One can easily check that $\int \alpha \underline{\hat{\mu}}(t)dt = \alpha \Omega_0 \underline{q}_0 = 1$.

Update Rules Derivation for the Assigned Canonical Kernel Functions

According to the set of kernel functions, we can prepare the table of weights and the functions needed for the efficient evaluation of the update rules[12]. We start with the RW update.

RW-FN Update

Recalling results from Sec. 5.2.3, we can write

$$w_{i,j} = \int \int p(\varphi_{i+1}|\varphi_i)\chi_i(\varphi_i)\chi_j(\varphi_{i+1})d\varphi_i d\varphi_{i+1}. \tag{5.21}$$

[12]We can note that the update rules design for the Fourier representation can be found also in [10] for this model. In contrast with this work, where the authors propose the implementation framework suited for this *concrete* scenario, we use this model just to verify the proposed generic design.

According to the numerical evaluation, we observed that the weights $w_{i,j}$ are non-zero valued only on the main diagonal in this case. This makes the evaluation of the expansion coefficients very efficient, that is the complexity grows linearly with the dimensionality of the messages. The update is given by

$$q_j^{\mathrm{RW}\to\varphi_i} = w_{jj} q_j^{\varphi_{i+1}\to\mathrm{RW}} \tag{5.22}$$

$$q_j^{\mathrm{RW}\to\varphi_{i+1}} = w_{jj} q_j^{\varphi_i\to\mathrm{RW}}. \tag{5.23}$$

PS-FN Update

The PS-FN is a hybrid node, because all adjacent variable nodes differ in their message representation types. We need two updates, concretely $\mu^{PS\to\varphi}$ and $\mu^{PS\to s}$ for the decoding purposes. The message $\mu^{PS\to u}$ leads to the open branch and its update is not required within the receiver.

We start with the discrete message $\mu^{PS\to s}$, which is represented by a discrete message

$$q_i^s = \int\int p(u|s,\varphi)\mu^{\varphi}(\varphi)\mu^{u}(u)dud\varphi \tag{5.24}$$

$$\approx \frac{1}{\sqrt{2\pi\sigma^2}} \sum_k q_k \Psi_{i,k}(u_0), \tag{5.25}$$

where we used the sampling property of δ function, the conditional pdf describing the PS FN and a set of functions

$$\Psi_{i,k}(u_0) = \int \exp\left(-\frac{|u_0 - s_i \exp(\mathrm{j}\varphi)|^2}{2\sigma^2}\right) \chi_k(\varphi) d\varphi. \tag{5.26}$$

The set $\{\Psi_{i,k}(u_0)\}_{i,k}$ is formed by some real-valued functions of a complex argument that are parameterized by the harmonic kernel functions $\chi_k(\varphi)$, constellation points s_i and AWGN variance σ^2. The set can be precomputed prior the processing to make the main FG-SPA run more efficient.

Similarly, one can derive

$$q_k = \frac{1}{\sqrt{2\pi\sigma^2}} \sum_i q_i^s \Psi_{i,k}(u_0). \tag{5.27}$$

Note that the update (5.25) is subjected to the rectification mentioned in Sec. (5.2.3), if $\hat{\mu}^{\varphi\to PS}(t) < 0$ for any t.

VN Update

The VN $\{\varphi_i\}_i$ updates are given by (5.13).

FG-SPA Implementation Framework Numerical Verification

We adopt the model 5.3.3 and implemented the KLT-message representation with the proposed update rules. The sample representation with numerical integration as the update rules is selected as a reference implementation framework. The results are shown in Fig. 5.17.

As a reference case, we have chosen the sample representation of the mixed-type density messages with numerical integration as the update rules. The results are shown in Figs. 5.17 and 5.18 for different dimensions of the messages (D).

Observing Fig. 5.17, we can identify the "waterfall" behavior of the proposed model and thus the curves confirm the framework functionality. One can also see that the proposed implementation framework

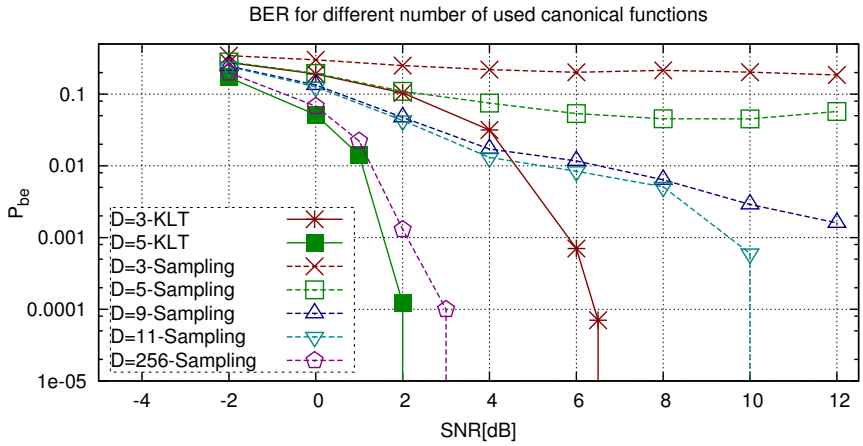


Figure 5.17: The bit error rate comparison of the proposed implementation framework base on the KLT approximated message with the representation by samples updated using the numerical integration with rectangular rule.

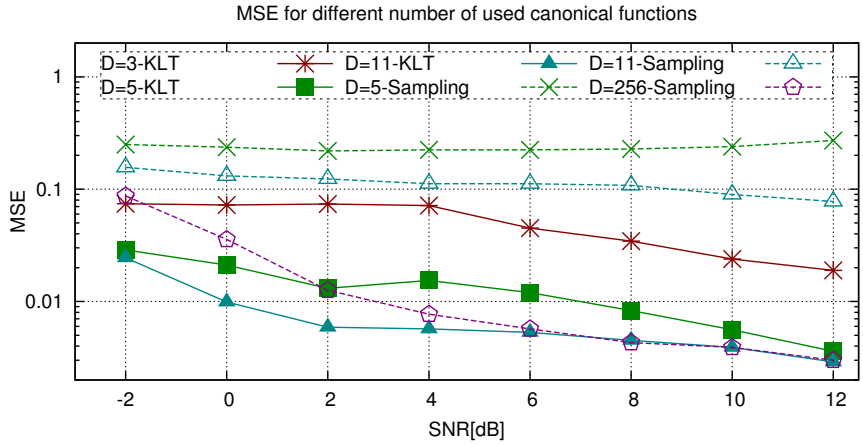


Figure 5.18: The mean square error of the phase estimation of the proposed implementation framework base on the KLT approximated message with the representation by samples updated using the numerical integration with rectangular rule.

	Sampling			KLT	
	$D=3$	$D=11$	$D=256$	$D=3$	$D=5$
Table	0	0	0	0.8	3
$+$	1.4	14.1	6746	0.7	2.8
$\times$	2.2	18.1	7640	1.7	6
$\exp(\cdot)$	0.6	5.5	2543	0	0
$\text{abs}(\cdot)$	0.2	0.8	19.7	0	0
$\Psi_{i,k}(u)$	0	0	0	0.01	0.02

Table 5.1: Number of symbol by symbol operations of some selected representations during the evaluation of the phase model - red color part of the receiver FG shown in Fig. 5.8. All numbers are divided by 10^6. The function $\Psi_{i,k}(u)$ is given in (5.26).

considerably outperforms the reference sample-based implementation FG-SPA framework in both MSE and BER cases for a given dimensionality.

We have (ad-hoc) chosen a threshold (softness coefficient) $\theta = 10^{-4}$ to avoid a hard decision about a symbol in codeword for the message $\mu^{PS\to s_i}$. The reason is that the discrete soft in soft out decoder much better incorporates a potentially wrong soft decision then the hard one with $\theta = 0$.

The reference sample-based implementation of the phase-model FG-SPA suffers by error floors for small number of samples. A particular reason can be seen in Fig. 5.18, where one can see a considerably worse phase estimation capabilities comparing to the proposed framework. The error-floor in the reference case can be explained by the presented narrow-shaped high-SNR Gaussian-like messages that are often below the resolution of a few (equidistant) samples.

The performance is,however, not only one measure of the FG-SPA implementation framework quality. We should also consider the complexity of the implementation, which is shown in Tab. 5.1. One can see that the complexity of the proposed framework is *lower* then the complexity of the reference solution. We should note that the complexity results in Tab. 5.1 are quite rough estimate, because they are just snapped from the simulation run without any further complexity-optimization in both methods.

In summary, the presented FG-SPA implementation framework is very generic as well as the sample-based framework. The KLT based framework requires a lot of evaluations (numerically intensive) prior the processing, while the reference model does not require any pre-computed calculation. The pre-computation effort is, however, rewarded by a significant gain in both BER and MSE performance metrics as well as in a lower implementation complexity for a given message dimensionality as discussed above.

5.4 KLT Based Implementation Framework Summary

We have proposed a methodical way for the canonical message representation based on the KLT. The method itself is not restricted for a particular scenario. It is sufficient to have a stochastic description of the message or at least a satisfactory number of message realizations and the KLT upon this process must be defined. The method, as it is presented, is restricted to real-valued one-dimensional messages in the FG-SPA.

We further derived a generic update rule design suited for an arbitrary linear canonical message representation with orthogonal canonical kernels. This generic update rule design is combined with the proposed KLT-message representation providing inherently the linear canonical message representation with orthogonal canonical kernels to form a generic framework. This linear canonical message representation is moreover *the best one* among all linear approximations in the MSE sense for a given number of representing parameters. Its fidelity can be well controlled by the sum of eigenvalues referring to the unused components.

We numerically verified the proposed framework in a concrete application - joint data detection - phase estimation in a conventional point-to-point AWGN channel with. We shown by simulations that the methodical way of the canonical kernel functions obtaining leads to the harmonic functions independently to the tested phase models, the channel encoders and the signal space mappers.

We investigated the capabilities of the KLT approximated message and compared them with the conventional message representations appearing in the literature. Furthermore, we compared the results of the proposed generic FG-SPA implementation framework including the proposed update rules design with the conventional sampling method. The reason of the sampled message selection as the reference case is that it stands also for a generic framework as well as the proposed design.

Part III

Decoding in Multi-Source Wireless Networks

6 Physical Wireless Network Communication

This Chapter provides a gentle introduction to wireless communication networks. The goal of this Chapter is to provide some basic overview and classification of the wireless networks to prepare a ground for the next Chapter, where we propose a decoder suited for the networks with wireless network coding. The reader familiar with the wireless network coding (physical layer network coding) [26, 30, 39, 54] can skip this Chapter directly to Chapter 7.

The physical layer wireless communication is often viewed as an extension of the conventional communication, where the higher layers direct the basic communication elements (physical layer) containing only P2P channels and MAC channels with SD-IC. The higher layers (e.g. routing) are optimized independently to the physical layer. The optimization of several layers (OSI-model) independently is intuitively not optimal from the global point of view. The global optimization is addressed as the *cross-layer design* solving the optimization across more (all) layers jointly.

The main problem of the cross-layer design is complexity of the cross-layer optimization. In this work, we rather focus on some basic phenomena (network coding paradigm [1, 67]) that can be applied in wireless networks and that promises a significant gain against the conventional independent optimization.

The network coding is however designed in the discrete world and it still uses the physical layer as a system of *orthogonal* pipes. The adoption of the physical layer taking into account the network coding paradigm (wireless network coding) then brings another potential improvement. Particularly the performance beyond the higher (e.g. second)-order cut-set bound can be achieved avoiding the conventional method such as either *joint decoding* or orthogonal P2P approach. Instead, a *hierarchical decoding* is introduced. The hierarchical decoding roughly means that the node decodes *directly a function* of data streams instead of decoding of all streams followed by a network operation upon the decoded streams.

The decoding of a function of data can be also viewed as reduction of *interference* that unavoidably corrupts the performance in the P2P based communication (MAC region limitation [11]). The largest potential gain of WNC is therefore in dense network, where the interference presents the main limitation of the system. Since still arising number of wireless devices communicates in a given (constant) area with given (constant) resources (wireless medium) with increasing demand on the transmission speed nowadays, the method of *non-orthogonal* interference reduction obviously attains a lot of attention of the research community in the past few years [68].

The WNC can be viewed as a cooperative diverse scheme [29]. The MIMO system is in fact a multi-source multi-destination(relay) wireless network with a *perfect* cooperation. It can be used as a benchmark for many applications. The full awareness of the system allows to adjust the precoding matrix with respect to the received signal. If the links between the individual antennas are not perfect, that is some constraints are assumed on these links (e.g. wireless channel), we can consider the wireless network as a generalized MIMO.

Prior we dive into the topic of the wireless network coding, we provide some basic classification of a general wireless network from the topology, constraints and processing point of view within this Chapter to get a better categorization of our work. Since the topic of detection in the multi-source wireless is closely and formally described in the following Chapter 7, this Chapter is written in intro-based style without any formal notation and definitions.

6.1 Wireless Network

6.1.1 Basic Elements

The wireless network is formed by a set of nodes that are located in an area and that are equipped by devices enabling a wireless communication. The nodes are supposed to be connected only through the wireless medium (wireless channels). The nodes can be classified either as *terminal nodes* or *relay nodes*.

The relay nodes are neither data (information) source nor data sink. The terminal nodes are divided to *sources* (data source) and *destinations* (data sinks). Some examples of the models for wireless networks that are significant for our work are discussed in Sec. 6.1.6.

We can describe the network by a graph, where the edges are given by wireless channels between the nodes and the nodes are given by the physical nodes. The *information flow* is given by some requirements from destinations. That is each destination is interested in information from a certain set of sources. A crucial difference between the conventional wireless networks based on OSI layer is that the information content in the pipes is *correlated* and therefore some description is dedicated also for some cuts of the network. According to the connections in the network and potentially further criteria (constraints), something (e.g. node itself or genie entity) decides about behaviour of each node.

6.1.2 Assumptions

Within this part of the work, we will explicitly assume the following constrains and assumptions.

Half Duplex Constraint

The half duplex constraint is a consequence of real hardware limitation, particularly the interference of the transmitted and received signal arises in the antenna if the signals are passed simultaneously (full-duplex). Due to this, it is assumed that the each node can either transmit or receive the signal at one time instance.

One can found many papers aiming with consequences of the half-duplex constraint investigation. A straightforward natural time division simplifying the network design is the same time portion dedicated for both the transmission and the reception of the signal in each node. On the other hand, a desired power allocation can be achieved adjusting the time division properly.

Uniform Unitary Transmit Power Distribution

A unit transmit power of all nodes is assumed from the simplicity reasons. An unequal power distribution of nodes within the network can be still adjusted by means of the flat fading coefficient in necessary.

Perfect Time Synchronization

The imperfect time synchronization basically leads to the inter-symbol interference. Since the network contains more nodes, various levels of synchronization imperfectness can be assumed. We consider a perfect time synchronization and channel estimation among all nodes in the network within this part. We refer e.g. [30] as a tutorial considering also the synchronization issue.

Acyclic Wireless Networks

From the complexity reasons, we will consider exclusively cycle-free networks within this work. It means that the directed graph representing the wireless network and the flow of the information is cycle free, that is if a node transmits some information, this information cannot be delivered back to the node in some later stage.

6.1.3 Network Processing Classification

Within the network processing, we recognize basically two types of the processing, particularly (see e.g. [41])

- *centralized* processing,
- *distributed* processing.

One can also combine these types of processing to form a hybrid processing that is directed partly in both distributed and centralized manner.

Centralized Processing

The centralized processing is called whatever direction of the processing that uses a central entity (genie) for the management of the processing in each node. It means that the processing in all nodes is directed by some genie. This solution is often taken as a reference, due to the possibility of the optimal solution upon the global description of the system. The problem is the communication overhead required to manage the network, since all communication between the central entity and a given node is supposed to be passed through the wireless medium (at expanse of data).

Distributed Processing

The distributed processing does not contain any central entity managing the network behaviour. It means that the nodes decide about their processing according to their own. The distributed processing saves the resource overhead required for the communication with the genie. On the other hand, only a local information is available for a decision about the behaviour of the node and therefore the optimality cannot be achieved from the global point of view in general.

Mixture Processing

The near-optimal fully-distributed processing of nodes is obviously the holy grail of the wireless network designers. This endeavor is, however, often difficult considering the fully-distributed system. We therefore classify also a mixture of the distributed and centralized processing that defines an genie entity for a given set of nodes. The decision about the behaviour is still provided in a distributed level from the global point of view, but the individual nodes are managed by a sub-genie entity.

6.1.4 Type and Nature of Information Appearing in the Wireless Network

A transmitted signal can be recognized only as a useful signal or nuisance interference in the conventional higher layer directed P2P approach. A fundamental difference between the WN and the higher layer directed P2P channels is that the WNC recognizes a Hierarchical-Side Information (H-SI). This information does not contain directly the desired data for the receiver, but instead, it somehow helps to decode the desired data.

To be more specific, we provide the following classification:

Direct Link

The receiver requires the data directly contained in the link from the given source. This is the same as in the conventional P2P communication.

Interference Link

The receiver does not need the data from the given source directly and also the hierarchical function is not a function of the given data. The interference link is again something that is well know from the conventional solution.

Figure 6.1: Relay-node processes the signal space observation vector $\boldsymbol{x}_0$. The output of the relay is again a signal space vector $\boldsymbol{s}_R$ depending on the relay processing strategy.

Hierarchical Side Information Link

The H-SI link presents some complementary information to desired data. This is in sense of the network coding paradigm, where the information can be temporary mixed in the network to increase the overall throughput. The H-SI is then used to reveal the desired data from the temporary mixed information that does not fully contain the desired data in general.

6.1.5 Relaying Strategies

The relaying strategy is called all the processing in the relay between the observation x_0 and the transmission of $\boldsymbol{s}_R$ (see Fig. 6.1). We omit a dummy relaying strategy, where the relay is silent or its behaviour does not depend on its observation x_0.

Amplify and Forward

The Amplify and Forward (AF) strategy is the simplest possible relaying strategy. It works just as a multiplier, that is $\boldsymbol{s}_R = \beta \boldsymbol{x}_0$, where β is a constant securing that $\boldsymbol{s}_R$ has unit energy per symbol. If we consider a MIMO channel received in the relay, β becomes to be a matrix.

This strategy preserves all information contained in $\boldsymbol{x}_0$, but the power of relay is wasted on the amplifying of the noise and unwanted interference collected in the wireless channel(s). The AF strategy is latency free. A description of the AF strategy in 2-WRC with half duplex constraint can be found e.g. in [48].

Hierarchical Decode and Forward

The Hierarchical Decode and Forward (HDF) is a strategy that provides a *decision* about a function of the *received data* (hierarchical data). The entropy of the hierarchical data *is less or equal* to the entropy of all received data vectors. The decoded hierarchical data are then again encoded and mapped to the signal space vector $\boldsymbol{s}_R$.

This strategy will be described later more precisely. We can mention that the HDF strategy includes (among others) as e.g. Compute and Forward strategy [40] or Denoise and Forward [26]. The HDF strategy suffers from the latency, because the full decoding requires to operate with whole sequence.

Note 6.1 (X-Decode and Forward)**.** The hierarchical function is not closer specified here. It means that the HDF strategy contains also conventional decode and forward (DF), where the function is identity for a given data stream or joint decode and forward (JDF), where the function is joint identity for all streams.

Compress and Forward

The Compress and Forward (CF) strategy is called such a strategy that somehow process $\boldsymbol{x}_0$ to provide a metric γ. This metric should be a sufficient statistics about the (hierarchical) data. A special case of this metric is the hierarchical data vector considered in HDF.

The CF strategy covers more approaches, typically quantize and forward, where the metric γ is given by somehow sampled version of $\boldsymbol{x}_0$.

6.1.6 Physical Network Topologies

We browse some basic topologies that are important for this work. We also exploit the particular topologies to demonstrate some principles of a given strategy (mainly HDF) that can be clearly seen in the given topology that are related with the sequel Chapter.

Two-Way Relay Channel

The two-way relay channel (TWRC - Fig. 6.2) is the simplest multi-source relay wireless network containing three nodes. The terminals want to exchange the data using the auxiliary relay. This model is also often used to demonstrate the power of the wireless network coding, since it is clearly sufficient to consider two time slots to exchange the information[1]. It nicely demonstrates the difference between the conventional P2P orthogonal channels, network coding provided upon orthogonal P2P models and the wireless network coding.

- Orthogonal P2P based communication without interference, where one needs *four* time-slots to exchange the information between the terminal nodes, particularly

  1. source S_A transmits to R data $\boldsymbol{b}_A$ that are decoded in relay, i.e. $S_A(\boldsymbol{b}_A) \to R(\boldsymbol{b}_A)$,
  2. $S_B(\boldsymbol{b}_B) \to R(\boldsymbol{b}_B)$,
  3. $R(\boldsymbol{b}_A) \to D_A(\boldsymbol{b}_A)$,
  4. $R(\boldsymbol{b}_B) \to D_B(\boldsymbol{b}_B)$.

  Each time-slot is occupied with a conventional P2P transmission without any interference. The transmission can thus operate on the rates corresponding to the capacity of the channels.

- P2P communication with *network coding* requiring *three* time slots for the full exchange, i.e.

  1. $S_A(\boldsymbol{b}_A) \to R(\boldsymbol{b}_A)$,
  2. $S_B(\boldsymbol{b}_B) \to R(\boldsymbol{b}_B)$,
  3. $R(\boldsymbol{b}_A \oplus \boldsymbol{b}_B) \to D_A(\boldsymbol{b}_A \oplus \boldsymbol{b}_B), D_B(\boldsymbol{b}_A \oplus \boldsymbol{b}_B)$.

  The destinations are able to decode the desired data utilizing the identity $\boldsymbol{b}_A = \boldsymbol{b}_A \oplus \boldsymbol{b}_B \oplus \boldsymbol{b}_B$ and analogously for $\boldsymbol{b}_B$. The first two time slots are equal to the previous case and the third one can also operate on the channel capacity considering symmetric channels.

- *Wireless network coding* demanding only *two* time-slots for full data exchange, particularly

  1. $S_A(\boldsymbol{b}_A), S_B(\boldsymbol{b}_B) \to R(\boldsymbol{b}_A \oplus \boldsymbol{b}_B)$
  2. $R(\boldsymbol{b}_A \oplus \boldsymbol{b}_B) \to D_A(\boldsymbol{b}_A \oplus \boldsymbol{b}_B), D_B(\boldsymbol{b}_A \oplus \boldsymbol{b}_B)$.

  The first time slot is occupied by a multiple-sources transmission (MAC) channel. The second order cut-set bound is avoided by decoding of $\boldsymbol{b}_A \oplus \boldsymbol{b}_B$ instead of the joint pair $(\boldsymbol{b}_A, \boldsymbol{b}_B)$ decoding. The performance within half bit below the first order cut-set bound was proven to be achievable in [38] for AWGN channels.

One can see that the main advantage of the wireless network coding in TWRC is in the bidirectional communication, where the *symmetry* of the information amount exchange is assumed.

[1] With some further assumptions.

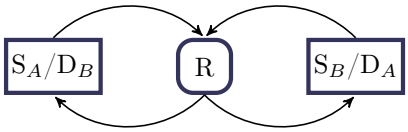


Figure 6.2: Two way relay channel.

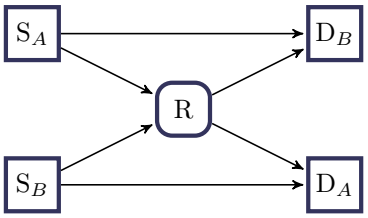


Figure 6.3: Butterfly network.

Note 6.2 (Hierarchical Function)**.** We used operation $\oplus$, which is viewed as addition upon a given finite field, typically bitwise XOR upon GF(2), relying that the destination is able to decode the mixed stream $\boldsymbol{b}_A \oplus \boldsymbol{b}_B$ using its data. This can be generalized to an arbitrary function enabling the decoding exploiting own data. Such a function is called *hierarchical function* or also *hierarchical map*.

Note 6.3 (Hierarchical Decoding)**.** The possibilities of the direct decoding of the hierarchical function (hierarchical stream) $\boldsymbol{b}_{AB} = \mathcal{X}_b(\boldsymbol{b}_A, \boldsymbol{b}_B) = \boldsymbol{b}_A \oplus \boldsymbol{b}_B$ in relay are the main topic of the this part (Chapter 7).

Note 6.4 (Relation of Hierarchical Function and Pure Network Coding)**.** Since the hierarchical function is a function of the data streams, one can be confused with the conventional network coding basically providing a function upon data as well. The difference is that the conventional network code is provided upon an *estimation* of whole data packets that are *jointly decoded*, i.e. the *joint decision* about all data streams was first performed and consequently the higher order cut-set bound limits the performance. The hierarchical function gives the function of data that are to be decoded (no joint decision is required). It is assumed only a decision about the *hierarchical data* that are some function of the source data streams and the higher order cut-set bound can be therefore overcome.

Butterfly Network

The butterfly network is a generalization of the TWRC, where we assume that the destinations do not have available the complementary information directly, but the information is received via a wireless channel as shown in Fig. 6.3. The complementary information flowing from the sources directly to the destinations called Hierarchical Side Information (H-SI) can be *imperfect*. The hierarchical function provided in relay must properly respect the H-SI such that the desired data should be computable from the hierarchical stream $\boldsymbol{b}_{AB} = \mathcal{X}_b(\boldsymbol{b}_A, \boldsymbol{b}_B)$ and H-SI (some function of $\boldsymbol{b}_A$ in D_B and $\boldsymbol{b}_B$ in D_A) in the final destinations.

Two-Relay Single-Destination Network

The two-relay single-destination network (Fig. 6.4) is a further generalization of the butterfly network. It is often assumed that the direct links are not available for simplicity. The extension against the butterfly network consists in a *generalization of the H-SI* from one relay point of view, which is formed by the hierarchical operation in the complementary relay. Indeed we have two hierarchical streams $\boldsymbol{b}_{AB} = \mathcal{X}_b(\boldsymbol{b}_A, \boldsymbol{b}_B)$

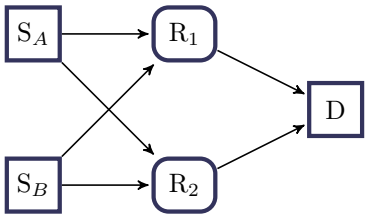


Figure 6.4: Two source, two relay, single destination network without direct source-destination H-SI links.

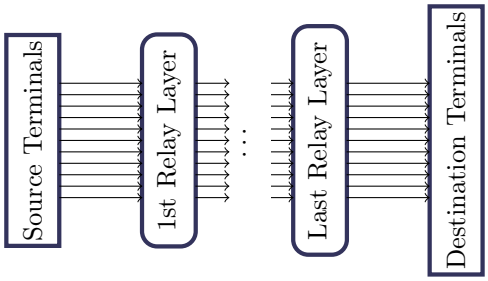


Figure 6.5: Acyclic layered network.

in relay R_1 and $\bar{\boldsymbol{b}}_{AB} = \bar{\mathcal{X}}_b(\boldsymbol{b}_A, \boldsymbol{b}_B)$ in relay R_2. These hierarchical streams must jointly contain all information about the tuple $(\boldsymbol{b}_A, \boldsymbol{b}_B)$ to keep the decoding possible in the final destination, i.e. some decoding function $[\boldsymbol{b}_A, \boldsymbol{b}_B] = \boldsymbol{\mathcal{X}}_b^{-1}(\boldsymbol{b}_{AB}, \bar{\boldsymbol{b}}_{AB})$ must exists.

The *overall hierarchical function* (formed by $\mathcal{X}_b$ and $\bar{\mathcal{X}}_b$) can provide either the minimal possible information, i.e. the vectors $\boldsymbol{b}_{AB}, \bar{\boldsymbol{b}}_{AB}$ are independent and the system has no diversity. If the information is larger than the minimal one, the hierarchical data streams are $\boldsymbol{b}_{AB}, \bar{\boldsymbol{b}}_{AB}$ correlated and the system might utilize the correlation of the received symbols in the final destinations. A typical example of the minimal possible information is TWRC or the butterfly network with perfect H-SI. On the other hand, the correlation occurs if both relays jointly decode both data streams $\boldsymbol{b}_A, \boldsymbol{b}_B$.

General Acyclic Network

A wireless network coding generalization brinks couple of complications. The hierarchical functions must be defined in each relay such that each destination receives a proper information for its purposes. Due to the complexity of the design, we proposed a unified framework [58], where the problem of the overall network design is decomposed to several sub-task. The core sub-task is the *wireless aware network coding* determining the hierarchical functions according to both the desired end-to-end information flows and the constrains given by the wireless network. A simplified model of the general acyclic network is a layered model (Fig. 6.5), where we assume a radio visibility just among the neighboring layers of nodes.

Multi-Hop Relay Channel

The multi-hop relay channel model (see Fig. 6.6 for two hops example) is a single-source scenario. It is typically associated with the latency problem. The point is that the relay must wait for a whole sequence followed by the decoding process in case of DF strategy, while the AF strategy works symbol-wisely and therefore no latency is accumulated in relay. The performance difference between the DF and AF presents a playground for some strategies (CF-based) providing better performance than AF strategy, but with lower latency than DF strategy.

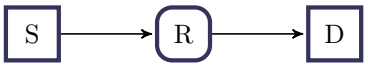


Figure 6.6: Multi-hop relay channel with two hops.

6.1.7 Challenging Problems in Wireless Networks

Within this Chapter, we briefly summarized the principle of the wireless network coding up to now taking accent mainly on the *hierarchical decoding*. Some further challenges in the wireless networks will be pointed out. These challenges are given by the tasks required for the cross-layer design in general. Nevertheless, we point out the challenges that are related directly with the goal of the thesis.

Wireless Aware joint Network Coding

The utilization of the network coding paradigm in the wireless network extension is available only in simple networks. The challenge is to design a method solving an ideal hierarchical maps distribution within a general wireless network such that some criteria are optimized (e.g. maximal utilization of the network coding paradigm still enabling desired decoding in final destinations). This solution should be moreover achieved in a distributive way. Note that the proposed block structured layered design (Chapter 7) can be applied to provide the hierarchical maps and the network-consistent channel coding design for some given coding rates, which can be derived from particular channel capacities. The next challenge is to derive a staging algorithm assigning properly stages *when* and *how long* each of the node should receive/transmit such that the power resources are utilized as efficiently as possible.

Channel Parameterization

The pure network coding directly shows how to utilize the network coding paradigm. Nevertheless, the utilization is quite complicated in the signal space, because the network coding is no longer provided upon the finite alphabets, but upon the complex-valued signal space. The channel is moreover parameterized by further constants (e.g. complex valued fading). Since the parameterization is related to more sources, it is not possible to avoid some crucial combination of the channel parameters purely by the processing in relay. It means that the constellation seen in a node receiving more signals (typically relay) is parameterized by a channel parameter, which can cause a constellation distortion decreasing the (hierarchical) channel capacity (available rate).

The way leading to solve this problem is twofold. One can consider some feedback to sources (e.g. channel pre-rotation [26]) or to design some alphabet that is resistant to the vulnerability of the performance with respect to the channel parameterization [59].

Synchronization and Other Real Implementation Problems

Of cause, there is plenty of problems that are beyond the scope of this thesis. As an instance, we assume a signal space description of the signal in network, which means that both the time and frequency synchronization are available in each node. It is however quite challenging problem to achieve this without a central unit.

7 Network Channel Code Design

The relay strategies based on a decoding (hard decision) of a *function* of the source data streams (e.g. bitwise XOR) were shown to move ahead the system performance in wireless networks. A proper implementation of these relay strategies needs to consistently incorporate channel coding across the network. Such an implementation is available only for special cases (mainly symmetric rates and equal alphabet cardinalities) in the current state of art. This Chapter presents a composition of the channel encoders in sources as well as the corresponding relay decoder for an arbitrary (asymmetric) individual source rates and for the corresponding (hierarchical) function of the source data streams. The proposed design consists of a proper mixture of the already known symmetric-rate based decoder design and the conventional joint decoding. We develop a unifying (linear) description of the processing in wireless networks. Utilizing this description, the proposed design results in a block-diagonal matrix composition preserving the (good) properties of the block elements (e.g. LDPC codes). The proposed design is demonstrated in a simple example and verified by the numerical simulation.

7.1 Introduction to Wireless Network Coding

The coding and modulation design in a general wireless physical network arose as a huge challenge recently. A pioneering work [1] showed that the information content can be temporarily mixed (network coded) within the network to increase either the overall network throughput (capacity) by means of reduced wireless channel use or the system diversity. An extension of the network coding (NC) to include also the wireless environment properties (wireless network coding (WNC) a.k.a. physical layer network coding [26, 30, 39]) then brings additionally further potential performance improvement. The idea of WNC consists in decoding (providing hard decision) of a (hierarchical) function of source data streams directly in constellation space [40] and forwarding it further, thus it is named Hierarchical Decode (Compute) and Forward strategy (HDF). The hierarchical function of source data streams (hierarchical stream) contains less information than the joint source data streams. Providing a hard-decision on the hierarchical stream, the classical Multiple Access Channel (MAC) region [11, 15] restrictions limiting the conventional joint-decoding based strategies does not longer limit the performance.

Since the hierarchical stream contains less information than the joint source data streams, the requested complement of this information must be delivered into the final destination by alternative paths in order to secure a reliable end to end transmission. This request is reflected e.g. by a condition that the final destination must collect enough number of equations from all paths in [40]. The information amount flowing by the alternative paths is called Hierarchical Side Information (H-SI).

A proper utilization of the WNC with HDF across the network potentially takes advantage of both reduced wireless channel use and operation rates beyond the classical MAC regions. This utilization is, however, not trivial in a general wireless network. The network design contains a joint optimization over immense degrees of freedom that has to be solved. Assuming a given known network state information and a perfect time synchronization among all nodes in the network, the overall optimization can be partitioned to some interacting subtasks [58], particularly (1) *network information flow* determining the hierarchical functions in each relay, i.e. the information routing that is aware of the wireless environment, (2) *network-consistent channel encoders* design respecting properly the network information flow as well as the maximal possible achievable coding rates resulting from the network state information and (3) *signal space mappers* connecting the codewords with the wireless environment. One can also consider various metrics to be optimized, e.g. the overall throughput maximization. The closed-form solution for such an optimization is not available in the current state-of-art.

The available literature offers the solution either of the cases with some very specific circumstances, typically small wireless networks with fixed network information flows, or a deeper insight in some particular sub-task without taking into account the others. One can recognize two major approaches. The first one

serves mainly for proving the achievability of a performance by means of the information theory tools, typically the lattice codes [17]. Some works based on this approach show very nice results, e.g. [38], where the authors proof the achievability of the performance within a half bit under the first order cut-set bound in Gaussian Two-Way Relay Channel (TWRC). The lattice code based models are, however, not suitable for a direct implementation. The second approach is therefore more constructive and directly implementable. It is based mainly on conventional capacity achieving channel codes and signal space mappers for a given network information flow design. Design of the directly implementable consistent channel encoders across the wireless network for arbitrary channel code rates, signal space mappers and network information flow stands for the main focus of this work. To best of our knowledge, no such design enabling arbitrary asymmetric rates is available in the current state of art (excepting our initial approach [45] and traditional joint decoding followed by a network code).

7.1.1 Related Work

Throughout the literature, it is very common to assume a *special* case of a *uniform* hierarchical ratio[1]. The uniform hierarchical ratio allows (among others) a popular bit-wise XOR of the source data streams as the hierarchical function [7, 8, 30, 64, 65], i.e. the traditional understanding of the (wireless) network coding. The hierarchical ratio is classified by means of the relay-alphabet cardinality in [54]. We stress that a commonly used TWRC as well as its generalization the butterfly network without a direct link from sources to destinations *does not allow arbitrary hierarchical ratio* in relay, because the H-SI cannot contain an arbitrary information amount (maximally one data stream). The case of maximal information flow through the H-SI (implicitly satisfied in TWRC with symmetric rates) allows the uniform hierarchical ratio case referred as the minimal hierarchical cardinality (meaning minimal information flow through the relay) in [54].

The consistent channel encoders design in the network is basically joint channel-wireless network coding that is able to benefit from both reduced channel use and the operational rates beyond the classical MAC region. Several works focus on the joint network-channel encoding (see e.g. [4, 20, 22]). These works are typically provided upon some graphical models (e.g. Tanner graph in [22] or factor graph in [20]). The low density parity check (LDPC) channel encoding technique is commonly used as the channel encoding part (e.g. in [20]). Linearity of both network and channel coding is standardly assumed. None of these works allows to work with arbitrary channel rates and hierarchical ratios.

The joint channel-wireless network coding design appears in literature only for very simple networks with further restrictions. These restrictions are typically *symmetric* channel rates and the *uniform* hierarchical ratio. For example, the authors of [30] propose both optimal and suboptimal soft metric based decoder relay design for TWRC, where the source encoders are assumed to be equal. The optimal design relies on the repeat-and accumulate codes in [30]. A similar design is proposed in [64], where the optimal design is built upon higher-order LDPC codes and the turbo codes based optimal relay decoder design is proposed in [8]. All of these optimal decoder designs rely on *equal channel encoders* in sources and *uniform* hierarchical ratio.

It is common to assume *memoryless* wireless channels connecting the nodes as well as a *symbol by symbol mapping* of the coded symbols to signal space symbols across a whole network[2]. In such a case, a symbol-wise processing can be applied in soft output demodulator (SoDeM) in each node to produce a soft metric for the channel decoder providing hard decision about a required hierarchical stream for a given node. The symbol-by-symbol metric evaluation enables a separate investigation of the channel symbol

[1] Notice that the hierarchical ratio determines the information amount flowing through the relay and it will be formally defined later in Definition 7.1.

[2] Notice that some works assume more general assumptions, e.g. more generic channels with inter-symbol interference caused by imperfect time synchronization, or the signal space mapping in sources that is over more than one symbol. Nevertheless none of these works allows the joint network-channel encoders design for an arbitrary source rates.

processing, where channel coding cannot be straightforwardly added (e.g. [26]). The symbol-by-symbol evaluated soft metric can be either optimal, i.e. a sufficient statistic for the hierarchical data (e.g. joint likelihood), or suboptimal (typically some marginalization of the optimal metric). The use of a suboptimal soft decoding metric is motivated by simplicity of the relay decoder implementation (see e.g. [30, 54]).

A construction of the relay decoder for an optimized (even non-linear) hierarchical function and LDPC encoders in sources in TWRC model is proposed in [31]. The assumption strictly limiting this work is that the non-zero elements of the LDPC matrix must be on the same position in all sources and relay LDPC matrices. This design thus does not allow arbitrary individual rates of sources.

7.1.2 Goals & Contribution

The main goal of this part of the work is to propose a *network-consistent channel encoders* design for a generic class of network structures respecting arbitrary network information flows, arbitrary (including asymmetric) rates of the channel encoders and arbitrary signal space mappers.

We develop a unifying robust network description based on linearity of the information flow as well the linearity of the channel encoders across a whole network. The available hierarchical data stream decoding options are revised by means of the proposed description. The advantages of the available strategies are combined in order to fulfill the overall goal, that is the network-consistent channel encoders design capable to incorporate arbitrary rates, information flows given by the hierarchical ratio and signal space mappers. We investigate some basic properties of the proposed network-consistent channel encoders design and numerically verify its performance.

7.1.3 Structure of Chapter

The rest of this Chapter is structured as follows. Notation and basic definitions are found in Sec. 7.2. The system description is addressed into Sec. 7.3. The following Section 7.4 aims with classification of the relay decoding options from the optimality, robustness and complexity point of view. The core of this chapter is presented in Sec. 7.5, where we describe a consistent directly implementable channel encoders construction concept across the whole network allowing to process *arbitrary* channel code rates as well as arbitrary hierarchical ratio. This construction is built upon the description in Sec. 7.3, where the building blocks are described in Sec. 7.4. We discuss also some properties of the proposed design. The proposed design is exemplified in Sec. 7.6 and some numerical verifications inspecting the performance of the proposed design can be found in Sec. 7.7. Some generalizations are addressed into Sec. 7.8 and finally the work is concluded in Sec. 7.9.

7.2 Notation & Definitions

We remind the notation that is used within this Chapter. A vector is denoted by a bold-face lower-case letter and it is implicitly assumed to be in a column form. It means that a vector $[b_1, \dots, b_k]^T$ is denoted $\mathbf{b}$. A zero vector with a single non-zero unit value on the ith position is denoted by $\mathbf{e}_i$, for example $\mathbf{e}_3$ is a vector given by $[0, 0, 1, 0, \dots]^T$. If the order of a particular symbol in the vector is not important or clear from the context, the subscript is dropped, i.e. b instead of b_i.

Similarly a matrix is denoted by a bold upper-case letter. For example $\mathbf{X}_{k\times n} = [\mathbf{x}_1 \dots \mathbf{x}_n]$ is $k \times n$ matrix composed from n vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$, where each of them is k symbols long. If the matrix is square, its dimension is denoted by a single index, that is $\mathbf{X}_{k\times k} = \mathbf{X}_k$. The matrix dimension in the subscript is omitted whenever the matrix dimension is clear from the context. A zero matrix $\mathbf{0}$ is a matrix containing all zero elements and a matrix $\mathbf{E}$ denotes the unitary matrix that can be written as $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \dots]$. A matrix $\mathbf{X}_{k\times n}$ is said *full-rank* if $\mathrm{rank}(\mathbf{X}) = \min(k, n)$.

Source data are conventionally associated with letter b and their length with k, similarly the codewords are denoted by c with associated length denoted n. The domain of both source data and codewords is an extended Galois Field GF[q^m], where the base q may differ for data and codewords as well as m. Throughout this work, we assume binary data, but an arbitrary extended GF can be also employed. The signal space points are associated with s and the soft metric outputting the SoDeM is denoted by μ.

The operator $\mathcal{S}_{\mathbf{b}}$ stands for a set of all possible vectors $\mathbf{b}$ and its size is denoted $M_{\mathbf{b}} = |\mathcal{S}_{\mathbf{b}}|$. In case of binary independent and identically distributed vector $\mathbf{b} \in \mathrm{GF}[2^{\mathrm{k}}]$, the size is $M_{\mathbf{b}} = 2^k$

7.3 System Description

We partly follow the partitioning suggested in [58] to describe the system. It means that we start with the information flow description, then we add the channel coding and signal space connection.

A considerable part of the investigated design is demonstrable in a simple 2-source 2-path network scenario. It means two sources, one stream flowing through a relay and one complementary stream flowing by an another way to a final destination. The complementary link, i.e. H-SI, is not closer specified, it may either flow though a complex network or be directly presented in the final destination. We will consider the simple 2-source 2-path network scenario in the main part of this work. Some generalizations are addressed into Sec. 7.8.

7.3.1 Information Flow Description

Assuming two informational sources A and B. The source data are independent and identically distributed (iid.) vectors $\mathbf{b}_A$ and $\mathbf{b}_B$ of lengths k_A and k_B that is $\mathbf{b}_A \in \mathrm{GF}[2^{\mathrm{k_A}}]$ and $\mathbf{b}_B \in \mathrm{GF}[2^{\mathrm{k_B}}]$.

We assume *linear* only hierarchical function. It means that the hierarchical data stream $\mathbf{b}_{AB}$ of length k_{AB} flowing through the relay can be expressed as

$$\mathbf{b}_{AB} = \mathbf{X}_{b,A}\mathbf{b}_A + \mathbf{X}_{b,B}\mathbf{b}_B, \tag{7.1}$$

where the hierarchical function is described by $k_{AB} \times (k_A + k_B)$ matrix $\mathbf{X}_b = \left[\begin{array}{cc} \mathbf{X}_{b,A} & \mathbf{X}_{b,B} \end{array} \right]$. The complementary data stream (H-SI) is formally denoted by an over-bar symbol and the H-SI data stream is given similarly by

$$\bar{\mathbf{b}}_{AB} = \bar{\mathbf{X}}_{b,A}\mathbf{b}_A + \bar{\mathbf{X}}_{b,B}\mathbf{b}_B.$$

The overall information flowing through the whole network (both relay and its complement) can be written as

$$\left[\begin{array}{c} \mathbf{b}_{AB} \\ \bar{\mathbf{b}}_{AB} \end{array} \right] = \left[\begin{array}{c} \mathbf{X}_b \\ \bar{\mathbf{X}}_b \end{array} \right] \left[\begin{array}{c} \mathbf{b}_A \\ \mathbf{b}_B \end{array} \right] = \left[\begin{array}{cc} \mathbf{X}_{b,A} & \mathbf{X}_{b,B} \\ \bar{\mathbf{X}}_{b,A} & \bar{\mathbf{X}}_{b,B} \end{array} \right] \left[\begin{array}{c} \mathbf{b}_A \\ \mathbf{b}_B \end{array} \right], \tag{7.2}$$

where the overall matrix must be invertible to keep the source data fully decodable in the final destination.

7.3.2 Channel Coding upon the Information Flow

Due to the error protection reason, we consider a channel encoding in the network. The source data vectors are encoded by *linear* encoders $\mathbf{c}_A = \mathbf{G}_A\mathbf{b}_A$ with coding rate $r_A = k_A/n_A$ and $\mathbf{c}_B = \mathbf{G}_B\mathbf{b}_B$ with coding rate $r_B = k_B/n_B$. The hierarchical codewords $\mathbf{c}_{AB}$ are related to the hierarchical data by a virtual (generally non-linear) encoder such that $\mathbf{c}_{AB} = \mathcal{C}_{AB}(\mathbf{b}_{AB})$ with rate $r_{AB} = k_{AB}/n_{AB}$. We stress that domains of $\mathbf{b}_A, \mathbf{b}_B$ and $\mathbf{b}_{AB}$ are not equal in general and therefore it is not straightforward how to find $\mathcal{C}_{AB}$ given $\mathbf{G}_A$ and $\mathbf{G}_B$.

The relation between the source and hierarchical codewords is also given by a vector codeword hierarchical function $\vec{\mathcal{X}}_c$ as $\mathbf{c}_{AB} = \vec{\mathcal{X}}_c(\mathbf{c}_A, \mathbf{c}_B)$. In order to enable a simplified soft-metric evaluation, we also

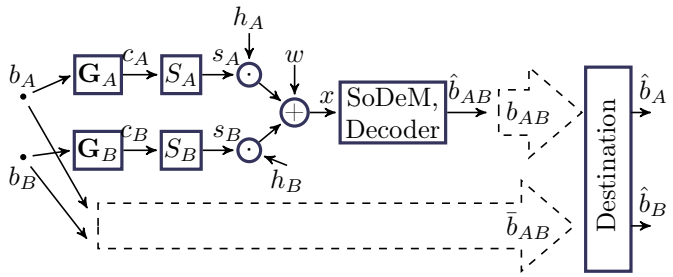


Figure 7.1: 2-Source 2-path scenario. The goal of this work consists in a joint design of both source encoders and relay decoder for arbitrary source rates as well as for an arbitrary hierarchical ratio.

define a symbol-wise hierarchical codeword function $\mathcal{X}_c(c_A, c_B)$ as the ith element of the vector hierarchical codeword function $\vec{\mathcal{X}}_c(\mathbf{e}_i c_{Ai}, \mathbf{e}_i c_{Bi})$. The hierarchical symbol is then given by[3] $c_{AB} = \mathcal{X}_c(c_A, c_B)$.

The hierarchical codeword satisfies

$$\begin{aligned} \mathbf{c}_{AB} &= \vec{\mathcal{X}}_c(\mathbf{c}_A, \mathbf{c}_B) = \vec{\mathcal{X}}_c(\mathbf{G}_A\mathbf{b}_A, \mathbf{G}_B\mathbf{b}_B) && (7.3)\\ &= \mathcal{C}_{AB}(\mathbf{b}_{AB}) = \mathcal{C}_{AB}(\mathbf{X}_{b,A}\mathbf{b}_A + \mathbf{X}_{b,B}\mathbf{b}_B), && (7.4) \end{aligned}$$

where the codeword hierarchical function is used in (7.3) and the virtual hierarchical encoder in (7.4).

If both the codeword hierarchical mapping $\vec{\mathcal{X}}_c$ and the virtual encoder $\mathcal{C}_{AB}$ are linear, the equations (7.3) and (7.4) can be written in the matrix form

$$\begin{aligned} \mathbf{c}_{AB} &= \mathbf{X}_{c,A}\mathbf{c}_A + \mathbf{X}_{c,B}\mathbf{c}_B = \mathbf{X}_{c,A}\mathbf{G}_A\mathbf{b}_A + \mathbf{X}_{c,B}\mathbf{G}_B\mathbf{b}_B && (7.5)\\ &= \mathbf{G}_{AB}\mathbf{b}_{AB} = \mathbf{G}_{AB}(\mathbf{X}_{b,A}\mathbf{b}_A + \mathbf{X}_{b,B}\mathbf{b}_B). && (7.6) \end{aligned}$$

Our goal to have the consistent channel encoders respecting properly the information flow can be now stated as follows. For arbitrary, but given source rates and hierarchical ratio, find consistent generator matrices[4] with good[5] corresponding parity check matrices and the inner structure of the hierarchical functions (hierarchical matrices). Dimensions and ranks of the matrices must be in accordance with the requested rates.

7.3.3 Connection with Signal Space

The hierarchical data stream is stochastically connected though the hierarchical codewords with the source codewords. The connection can be expressed by conditional probability $p(\mathbf{x}|\mathbf{c}_{AB})$, where $\mathbf{x}$ is an observation available in the relay.

The signal space space symbols $s_A(c_A), s_B(c_B)$ are given by signal space mappers in sources. We assume a memoryless channel, i.e. the relay receives a noisy superposition of these $x = h_A s_A + h_B s_B + w$, where h_A, h_B is flat fading channel parametrization and w is complex zero mean additive white Gaussian noise (AWGN) with unit variance. The variance of the AWGN can be adjusted by a proper scaling of the channel gains $\gamma_A = |h_A|^2$ and $\gamma_B = |h_B|^2$. The overall signal to noise ratio (SNR) available in relay is thus given by $SNR = \gamma_A + \gamma_B$ assuming $E[|s_A|^2] = E[|s_B|^2] = 1$. See Fig. 7.1.

[3] This relation is valid only for a memoryless channel, where the symbol c_{ABi} depends only on c_{Ai} and c_{Bi}. Otherwise, this symbol-wise hierarchical mapping has no sense.

[4] The consistent generator matrices are channel encoders in sources and decoder in relay designed in such a way that the decoder in relay properly decodes the desired hierarchical data stream in relay. See Def. 7.2 for a formal definition.

[5] E.g. LDPC matrix with proper degrees distribution and maximized girth.

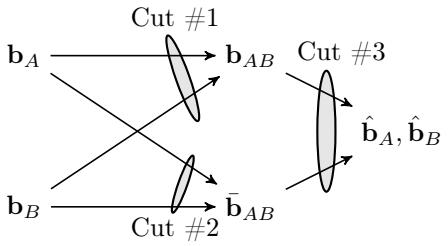


Figure 7.2: Visualization of the information flow within the network. The information amount flowing through the relay ($\mathbf{b}_{AB}$) is visualized by the Cut #1, the information flowing through H-SI is denoted Cut #2 and the overall information that needs to be delivered into a final destination(s) is denoted by Cut #3. One can see that the Cuts #1 and #2 can be seen as a multi-source broadcast channel. Setting the hierarchical ratios λ and $\bar{\lambda}$, one can adjust the information amount flowing though particular Cuts.

7.3.4 Classification of Information Flow in Relay

The information flow through the relay can be expressed by means of entropy of the hierarchical stream, that is $\mathcal{H}(\mathbf{b}_{AB}) = \text{rank}(\mathbf{X}_b)$ for binary uniform iid. $\mathbf{b}_A$ and $\mathbf{b}_B$ (see Cut #1 in Fig. 7.2). Note that for full-rank $\mathbf{X}_B$, $\text{rank}(\mathbf{X}_b) = k_{AB}$. We can also consider the overall information amount related to the number of channel use measured by length of the hierarchical codeword $\mathbf{c}_{AB}$, i.e. $\text{rank}(\mathbf{X}_b)/n_{AB}$.

The wireless network is desired to deliver $\mathcal{H}([\mathbf{b}_A^T \mathbf{b}_B^T]) = k_A + k_B$ bits into the final destination (see Cut #3 in Fig. 7.2). We therefore define the *hierarchical ratio* determining a portion of the overall information to be processed by the relay.

Definition 7.1 (Hierarchical Ratio)**.**

The ratio

$$\lambda = \frac{\text{rank}(\mathbf{X}_b)}{k_A + k_B} \tag{7.7}$$

is called the *hierarchical ratio* for a given relay.

If the matrix $\text{rank}(\mathbf{X}_b)$ is full-rank, we can express the hierarchical ratio as

$$\lambda = \frac{k_{AB}}{k_A + k_B} \tag{7.8}$$

In oder to reliably deliver $k_A + k_B$ bits into the final destination, the H-SI must process at least the complement of the information amount flowing through the relay. This amount is given by $\text{rank}(\bar{\mathbf{X}}_b) \geq k_A + k_B - \text{rank}(\mathbf{X}_b)$ bits (Cut #2 in Fig. 7.2). We can also define *complementary hierarchical ratio* as

$$\bar{\lambda} = \frac{\text{rank}(\bar{\mathbf{X}}_b)}{k_A + k_B} \geq 1 - \frac{\text{rank}(\mathbf{X}_b)}{k_A + k_B}. \tag{7.9}$$

Now the diversity of the system is given by $\lambda + \bar{\lambda} - 1$ and it takes values from the interval $\langle 0, 1 \rangle$, where 0 denotes no diversity and 1 refers to full diversity. Notice that in case of non-zero diversity, the correlation of the streams $\mathbf{b}_{AB}$ and $\bar{\mathbf{b}}_{AB}$ is implied.

A special case of full-rank hierarchical matrices and $k_A = k_B = k_{AB}$ refers to the *uniform information flow*, because the information flowing through both the relay and the H-SI is equal assuming no diversity. This case enables an elegant solution of the joint network channel encoder, where $G_A = G_B = G_{AB}$ and this will be discussed later.

The paper [54] classifies the information flow through the relay according to the relay alphabet cardinality, which is equal to our entropy-based classification for iid. data. They assume a symmetric butterfly network, where the H-SI is given maximally by one data stream. It means that the minimal information flowing though the relay must be the second data stream. The classification is concretely given by:

- *minimal cardinality*: $M_{\mathbf{b}_{AB}} = \max(M_{\mathbf{b}_A}, M_{\mathbf{b}_B})$,
- *extended cardinality*: $\max(M_{\mathbf{b}_A}, M_{\mathbf{b}_B}) < M_{\mathbf{b}_{AB}} < M_{\mathbf{b}_A} M_{\mathbf{b}_B}$
- *full cardinality*: $M_{\mathbf{b}_{AB}} = M_{\mathbf{b}_A} M_{\mathbf{b}_B}$.

7.4 Classification of the Relay Decoding in Wireless Networks with HDF

The HDF technique requires a decision about the hierarchical stream $\mathbf{b}_{AB}$. The main area of interest of this work lies in all processing (decoding) between the vector $\mathbf{x}$ received in relay (the superimposed signal from several sources parameterized by the sources-relay channels) and the estimation of the hierarchical stream $\mathbf{b}_{AB}$ in relay.

The goal of the hierarchical decoder in HDF strategy is to obtain the hierarchical stream $\mathbf{b}_{AB}$ from the observation $\mathbf{x}$. The decoder can be classified as the joint-network channel decoder. The objectives of this decoder can be summarized as:

- Performance properties.
- Implementation complexity.
- Robustness of design, that is an ability to properly incorporate arbitrary rates and hierarchical ratio..

There are basically three major approaches differing in the metric used for the decoder design.

7.4.1 Symbol-Wise Metric Evaluation for they Decoder in Relay

As it was stated, the goal is to obtain the hierarchical data stream $\mathbf{b}_{AB}$. The hierarchical data are related with the hierarchical codewords by a virtual hierarchical channel encoder[6] $\mathcal{C}_{AB}$. Assuming a memoryless channel, it is reasonable to evaluate a symbol-wise soft metric in demodulator due to the complexity reasons.

Optimal Symbol-Wise Metric

The MAP rule gives

$$P(\mathbf{b}_{AB}|\mathbf{x}) = P(\mathbf{c}_{AB}|\mathbf{x}) \sim p(\mathbf{x}|\mathbf{c}_{AB})p(\mathbf{c}_{AB}). \tag{7.10}$$

[6] We assume just that such an encoder exists. A particular form of the encoder will be discussed later.

Using the vector codeword hierarchical function, we can evaluate the conditional likelihood by

$$p(\mathbf{x}|\mathbf{c}_{AB}) = \sum_{\mathbf{c}_A,\mathbf{c}_B:\vec{\mathcal{X}}_c(\mathbf{c}_A,\mathbf{c}_B)=\mathbf{c}_{AB}} p(\mathbf{x}|\mathbf{c}_A,\mathbf{c}_B) \tag{7.11}$$

$$= \sum_{\mathbf{c}_A,\mathbf{c}_B:\vec{\mathcal{X}}_c(\mathbf{c}_A,\mathbf{c}_B)=\mathbf{c}_{AB}} \prod_i p(x_i|c_{Ai},c_{Bi}). \tag{7.12}$$

Therefore the symbol-wise metric $p(x_i|c_{Ai},c_{Bi})$ is *optimal* with respect to the ML as well as the MAP detection[7].

Hierarchical Symbol-Wise Metric

The hierarchical metric $p(x|c_{AB})$ is given by a marginalization

$$p(x|c_{AB}) = \sum_{c_A,c_B:\mathcal{X}_c(c_A,c_B)=c_{AB}} p(x|c_A,c_B). \tag{7.13}$$

This metric is, however, not the exact one for the data vector decoding purposes. The probability of the observation conditioned by the hierarchical codeword *is not generally independent* for different time instances. The use of the symbol-wise hierarchical metric is therefore unavoidably connected with the approximation

$$p(\mathbf{x}|\mathbf{c}_{AB}) \approx \prod_i p(x_i|c_{ABi}). \tag{7.14}$$

Note that in case of uncoded system and uniform hierarchical ratio, the approximation in (7.14) becomes to equality.

Network Coding over Joint Decoding

An another approach is to evaluate first hard decisions $\hat{\mathbf{b}}_A$, $\hat{\mathbf{b}}_B$ and using these decisions to apply the classical network code. The metrics used in the demodulator are $p(x|c_A)$ and $p(x|c_B)$ obtained by a proper marginalization from $p(x|c_A,c_B)$. Of cause, one can consider a conventional Successive Decoding - Interference Cancellation (SD-IC) method to improve the performance (to achieve the conventional MAC region).

7.4.2 Decoding Strategies in Relay

The different soft metric evaluations in relay enables different decoding strategies.

Joint Decoding with Classical Network Coding

This is a natural sub-optimal basic solution, where no advantage of the wireless network coding is used. The decoder uses a conventional joint decoding in MAC. Upon this hard decision, the network operation described by the matrices $\mathbf{X}_{b,A}$ and $\mathbf{X}_{b,B}$ is performed separately.

This approach can be straightforwardly extended to both more sources and more paths. The performance is limited by the conventional MAC region.

Given matrices $\mathbf{X}_{b,A}$ and $\mathbf{X}_{b,B}$, this strategy works regardless both the hierarchical ratio and the rates of the encoders in sources. Formally, the codeword hierarchical matrices are unitary matrices $\mathbf{X}_{c,A} = \mathbf{E}_{n_A}$,

[7] Assuming a uniform distribution of c_{AB}.

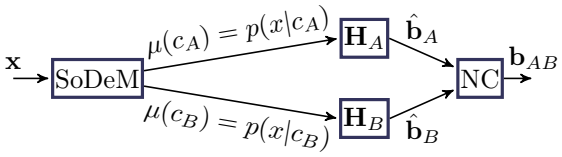


Figure 7.3: The relay SoDeM-decoder scheme for the joint decoding over network code. Soft output demodulator evaluates symbol-wisely $p(x_i|c_{Ai})$ and $p(x_i|c_{Bi})$ for all i. This metric is served to the separated decoders $\mathbf{H}_A$ and $\mathbf{H}_B$. These decoders provide a *hard decision* about $\hat{\mathbf{b}}_A$ and $\hat{\mathbf{b}}_B$. This is then used to compute the hierarchical stream in NC block by $\hat{\mathbf{b}}_{AB} = \mathbf{X}_{b,A}\hat{\mathbf{b}}_A + \mathbf{X}_{b,B}\hat{\mathbf{b}}_B$.

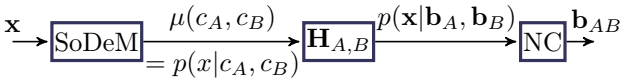


Figure 7.4: The optimal relay SoDeM-decoder scheme for the joint decoding and soft network code. Soft output demodulator evaluates symbol-wisely a sufficient statistic metric $p(x_i|c_{Ai}, c_{Bi})$ for all i. This metric is served to joint decoder $\mathbf{H}_{A,B}$ providing a soft metric $p(\mathbf{x}|\mathbf{b}_A, \mathbf{b}_B)$. This metric is then used to compute the hierarchical stream in *soft* NC block by (7.17).

$\mathbf{X}_{c,B} = \mathbf{E}_{n_B}$ and the decoding taking into account the codeword hierarchical relation (7.5) is solved separately according to

$$\mathbf{c}'_A = \mathbf{X}_{c,A}\mathbf{c}_A = \mathbf{E}_{n_A}\mathbf{c}_A = \mathbf{c}_A = \mathbf{G}_A\mathbf{b}_A, \tag{7.15}$$

$$\mathbf{c}'_B = \mathbf{X}_{c,B}\mathbf{c}_B = \mathbf{E}_{n_B}\mathbf{c}_B = \mathbf{c}_B = \mathbf{G}_B\mathbf{b}_B, \tag{7.16}$$

where the marginalized metrics $p(x_i|c_{Ai})$ and $p(x_i|c_{Bi})$ are evaluated by SoDeM. This separated decoding leads to decision upon the both data streams, that is $\hat{\mathbf{b}}_A$ and $\hat{\mathbf{b}}_B$. Finally, the hierarchical data stream is given by $\hat{\mathbf{b}}_{AB} = \mathbf{X}_{b,A}\hat{\mathbf{b}}_A + \mathbf{X}_{b,B}\hat{\mathbf{b}}_B$. This setup is shown in Fig. 7.3.

Optimal Solution - Joint Decoding with Network Coding

This method stands for the optimal one (Fig. 7.4), where the meaning of the optimality is explained in Note 7.2. The demodulator in relay evaluates symbol-wisely the metric $p(x|c_A, c_B)$ (7.12) as a sufficient statistics for the hierarchical data. Using the evaluated optimal metric $p(x|c_A, c_B)$,the joint decoder evaluates the metric $p(\mathbf{x}|\mathbf{b}_A, \mathbf{b}_B)$ for the soft network decoder. The soft network decoder provides a marginalization

$$p(\mathbf{x}|\mathbf{b}_{AB}) = \frac{1}{p(\mathbf{b}_{AB})} \sum_{\mathbf{b}_A, \mathbf{b}_B : \mathbf{b}_{AB} = \mathbf{X}_{b,A}\mathbf{b}_A + \mathbf{X}_{b,B}\mathbf{b}_B} p(\mathbf{x}|\mathbf{b}_A, \mathbf{b}_B) \tag{7.17}$$

which is then passed to the hard decision as

$$\hat{\mathbf{b}}_{AB} = \arg\max_{\mathbf{b}_{AB}} p(\mathbf{x}|\mathbf{b}_{AB}). \tag{7.18}$$

The problem of this method is in evaluation $p(\mathbf{x}|\mathbf{b}_A, \mathbf{b}_B)$, which is often numerically intensive. Moreover, this brute force solution needs to design all encoders and decoders *jointly*, because the relay-decoder properties will be determined by *different* source encoders. The complexity of the optimal decoder will rapidly grow with more considered sources.

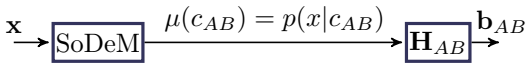


Figure 7.5: The relay SoDeM-decoder scheme for the hierarchical decoding. Soft output demodulator evaluates symbol-wisely $p(x_i|c_{AB\,i})$ for all i. This metric is served to one hierarchical decoder $\mathbf{H}_{AB}$. This decoder directly provides a *hard decision* $\mathbf{b}_{AB}$.

To formally describe this solution, we use a similar formalism as in [64] that is both codewords $\mathbf{c}_{AB}$ and joint data pair $(\mathbf{b}_A, \mathbf{b}_B)$ take values from the extended GF with four elements[8] in $\{0, 1, D, D+1\}$. The codeword hierarchical matrices are assumed to be the matrices upon the extended GF given by $\mathbf{X}_{c,A} = \mathbf{E}_{n_A}$, $\mathbf{X}_{c,B} = D\mathbf{E}_{n_B}$ and the decoding with (7.5) can be expressed as

$$\mathbf{c}_{AB} = \mathbf{E}_{n_A}\mathbf{c}_A + D\mathbf{E}_{n_B}\mathbf{c}_B = \mathbf{G}_A\mathbf{b}_A + D\mathbf{G}_B\mathbf{b}_B. \tag{7.19}$$

The decoder aims with a joint soft metric about[9] $\mathbf{b}_A + D\mathbf{b}_B$ that serves as the *soft output* of the decoder, that is $p(\mathbf{x}|\mathbf{b}_A, \mathbf{b}_B)$ in (7.17).

Example 7.1 (Optimal Solution - Uniform Hierarchical Ratio)**.**

In a special symmetric case of uniform hierarchical ratio and equal encoders in sources, that is $\mathbf{G}_A = \mathbf{G}_B = \mathbf{G}$, we obtain for (7.19)

$$\mathbf{c}_{AB} = \mathbf{G}(\mathbf{b}_A + D\mathbf{b}_B) \tag{7.20}$$

and a straightforward decoder construction evaluating $p(\mathbf{x}|\mathbf{b}_A, \mathbf{b}_B)$ can be applied. Such a decoder construction on LDPC codes is proposed in [64] and similar construction based on turbo encoder is proposed in [8].

Note 7.2 (Optimality of Joint Decoding with Network Coding)**.** The optimality of the processing is with respect to obtain as much information as possible. It means that the soft metric $p(\mathbf{x}|\mathbf{b}_A, \mathbf{b}_B)$ contains all available information for the hierarchical data stream decision. Nevertheless the optimality is paid by the complexity of the decoder.

Direct (Hierarchical) Decoding

The direct (hierarchical) decoding uses the hierarchical metric $p(x_i|c_{AB\,i})$ in the demodulator and the hierarchical stream $\mathbf{b}_{AB}$ is evaluated directly from the hierarchical metric. See Fig. 7.5. The capacity regions are computed in [54] for the hierarchical MAC phase in case of uniform hierarchical ratio. The metric $p(x|c_{AB})$ is suboptimal, but a direct evaluation of the hierarchical stream brings an *efficient solution*, because it is required only one run of the decoder. The problem of the this type decoding is to find a consistent encoders and hierarchical maps such that both (7.5) and (7.6) are satisfied. A straightforward construction of consistent hierarchical functions and encoders is not obvious in general.

[8] We assume that both $\mathbf{c}_A$ and $\mathbf{c}_B$ take values from binary extended finite fields.

[9] If the vector lengths are not equal, that is $k_A \neq k_B$, the shorter vector is filled by zeros such that the dimensions will pass.

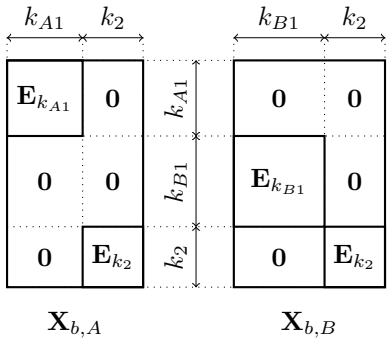


Figure 7.6: The dimension of hierarchical matrices in the proposed block structured layered design. Notice that the structure of the codeword hierarchical matrices is equal with dimensions $n_\star$ instead of $k_\star$.

Example 7.3 (Direct Decoding - Uniform Hierarchical Ratio)**.**

A simple, however very special solution, is available for *uniform hierarchical ratio*. Then the design in a particular form containing equal encoders $\mathbf{G}_A = \mathbf{G}_B = \mathbf{G}$ and unitary the hierarchical functions $\mathbf{X}_{b,A} = \mathbf{X}_{b,B} = \mathbf{E}$, $\mathbf{X}_{c,A} = \mathbf{X}_{c,B} = \mathbf{E}$ implies

$$\mathbf{c}_{AB} = \mathbf{G}(\mathbf{b}_A + \mathbf{b}_B). \tag{7.21}$$

One can check that both (7.5) and (7.6) are fulfilled and therefore this network design stands for a valid network configuration (see e.g. [54]).

It is obvious that this approach will not work for arbitrary hierarchical ratio and asymmetric codes.

7.5 Block-Structured Layered Design

Our approach aims with a maximal utilization of the methods suited for *uniform hierarchical rate*, because we are able to design either optimal (Section 7.4.2, particularly Example 7.1) or hierarchical (direct) (Section 7.4.2, particularly Example 7.3) channel encoder implementation for such a case. To do this, we restrict slightly the structure of the encoding matrices into block-wise form. The inner matrices of the block-structured design can be arbitrary valid capacity achieving single-user linear block codes and thus the layered design paradigm is preserved. Those component codes can be e.g. LDPC codes and the overall design preserves the sparsity of the overall matrices. An arbitrary hierarchical ratio as well as arbitrary source rates can be achieved by a proper mixture of the block-matrices dimensions.

7.5.1 Formal Definition

We formally define our goal to have a valid network configuration. It means that $\mathbf{G}_{AB}$ must properly define the virtual hierarchical encoder, i.e. (7.5) must be satisfied jointly with (7.6) for all $\mathbf{b}_A, \mathbf{b}_B$.

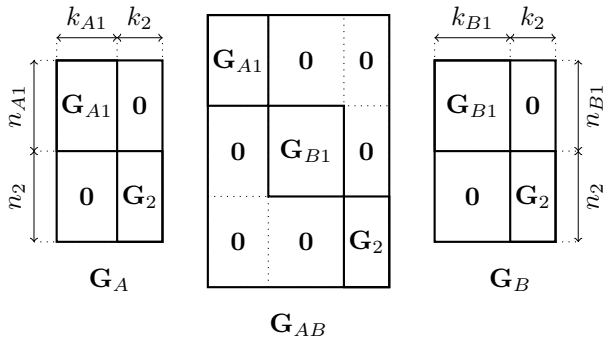


Figure 7.7: Diagonal block encoders composition.

Definition 7.2 (Consistent Generator Set)**.**

We call a set of full-rank generator matrices $\{\mathbf{G}_{AB}, \mathbf{G}_A, \mathbf{G}_B\}$ jointly with full-rank hierarchical matrices $\{\mathbf{X}_{b,A}, \mathbf{X}_{b,B}, \mathbf{X}_{c,A}, \mathbf{X}_{c,B}\}$ satisfying

$$\mathbf{G}_{AB}\mathbf{X}_b = \begin{bmatrix} \mathbf{G}_{AB}\mathbf{X}_{b,A} & \mathbf{G}_{AB}\mathbf{X}_{b,B} \end{bmatrix} = \begin{bmatrix} \mathbf{X}_{c,A}\mathbf{G}_A & \mathbf{X}_{c,B}\mathbf{G}_B \end{bmatrix} \tag{7.22}$$

a *consistent generator set*.

We extend the *consistent generator set* by valid parity check matrices for decoding purposes.

Definition 7.3 (Consistent Parity Check Set)**.**

The full-rank parity check matrices $\{\mathbf{H}_{AB}, \mathbf{H}_A, \mathbf{H}_B\}$ jointly with the consistent generator set are called the *consistent parity check set* if $\mathbf{H}_A$, $\mathbf{H}_B$ and $\mathbf{H}_{AB}$ are valid parity check matrices of $\mathbf{G}_A$, $\mathbf{G}_B$ and $\mathbf{G}_{AB}$.

The consistent parity check set formally states our overall goal. Among a number of approaches, we choose a joint network-channel encoding construction that produces a valid network implementation. We introduce orthogonal matrix-blocks $\mathbf{G}_{A1}$, $\mathbf{G}_{B1}$ and $\mathbf{G}_2$. One part of both individual data streams is encoded by a joint encoder $\mathbf{G_2}$ and the second part of them by separated encoders $\mathbf{G}_{A1}$ and $\mathbf{G}_{B1}$.

Definition 7.4 (Block Structured Layered Code)**.**

We assume full-rank matrices $\mathbf{G}_{A1} : n_{A1} \times k_{A1}$, $\mathbf{G}_{B1} : n_{B1} \times k_{B1}$ and $\mathbf{G}_2 : n_2 \times k_2$. If the generator matrices can be written as $\mathbf{G}_A = \mathrm{diag}(\mathbf{G}_{A1}, \mathbf{G}_2)$, $\mathbf{G}_B = \mathrm{diag}(\mathbf{G}_{B1}, \mathbf{G}_2)$, $\mathbf{G}_{AB} = \mathrm{diag}(\mathbf{G}_{A1}, \mathbf{G}_{B1}, \mathbf{G}_2)$ (see Fig. 7.7) and the HNC matrices are in the form (see Fig. 7.6)

$$\mathbf{X}_{b,A} = \begin{bmatrix} \mathbf{E}_{k_{A1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{k_2} \end{bmatrix} \quad \text{and} \quad \mathbf{X}_{b,B} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{E}_{k_{B1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{k_2} \end{bmatrix}, \tag{7.23}$$

$$\mathbf{X}_{c,A} = \begin{bmatrix} \mathbf{E}_{n_{A1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{n_2} \end{bmatrix} \quad \text{and} \quad \mathbf{X}_{c,B} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{E}_{n_{B1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{n_2} \end{bmatrix}. \tag{7.24}$$

The matrices are called *block structured layered code.*

Note 7.4 (Mixturing Joint Decoding and Uniform Hierarchical Ratio)**.** Notice that $\mathbf{E}_{(.)_{(.)1}}$ matrices define the *joint decoding over NC* mapping part and $\mathbf{E}_{(.)2}$ matrices define the *uniform hierarchical ratio* mapping part. The overall map is thus a mixture of the strategies. A proper choice of matrix sizes allows to achieve an arbitrary hierarchical ratio as well as arbitrary source rates

We state some basic properties of the *block structured layered code.*

Lemma 7.1. *Assuming a* block structured layered code*, all matrices* $\mathbf{G}_{AB}$*,* $\mathbf{G}_A$*,* $\mathbf{G}_B$*,*$\mathbf{X}_{b,A}$*,*$\mathbf{X}_{b,B}$*,* $\mathbf{X}_{c,A}$ *and* $\mathbf{X}_{c,B}$ *are full-rank.*

Proof. The proof of this Lemma is straightforward, because the block-matrices are full-rank from definition and the diagonal composition of the full-rank block matrices as shown in Fig. 7.7 surely preserves the full-rank property. □

Lemma 7.2. *An arbitrary* block structured layered code *forms a consistent generator set.*

Proof. The full-rank property is guaranteed by Lemma 7.1. One can verify that (7.22) is also satisfied by substituting Definition 7.4 into (7.5) and (7.6). □

We show a construction of the parity check matrices based on the *block structured layered* code satisfying the consistency parity check conditions.

Theorem 7.3 (Block-Structured Layered Code Construction)**.**

We assume an arbitrary *block structured layered* code and denote $\mathbf{H}_{A1} : (n_{A1} - k_{A1}) \times n_{A1}$, $\mathbf{H}_{B1} : (n_{B1} - k_{B1}) \times n_{B1}$ and $\mathbf{H}_2 : (n_2 - k_2) \times n_2$ parity check matrices of the generator blocks $\mathbf{G}_{A1}$, $\mathbf{G}_{B1}$ and $\mathbf{G}_2$ in a full-rank form. Then the parity check matrices $\mathbf{H}_A = \mathrm{diag}(\mathbf{H}_{A1}, \mathbf{H}_2)$, $\mathbf{H}_B = \mathrm{diag}(\mathbf{H}_{B1}, \mathbf{H}_2)$ and $\mathbf{H}_{AB} = \mathrm{diag}(\mathbf{H}_{A1}, \mathbf{H}_{B1}, \mathbf{H}_2)$ form a consistent parity check set.

Proof. The full-rank property of the parity check matrices can be shown similarly as in the generator matrices case in proof of Lemma 7.1. The consistent generator set is formed according to Lemma 7.2.

To verify that the matrices are valid parity check matrices, we write e.g. for $\mathbf{H}_A$

$$\begin{aligned}\mathbf{H}_A\mathbf{G}_A &= \operatorname{diag}(\mathbf{H}_{A1},\mathbf{H}_2)\operatorname{diag}(\mathbf{G}_{A1},\mathbf{G}_2) &(7.25)\\ &= \operatorname{diag}(\mathbf{H}_{A1}\mathbf{G}_{A1},\mathbf{H}_2\mathbf{G}_2) &(7.26)\\ &= \operatorname{diag}(\mathbf{0}_{n_{A1}-k_{A1},k_{A1}},\mathbf{0}_{n_2-k_2,k_2}) &(7.27)\\ &= \mathbf{0}_{n_A-k_A,k_A}. &(7.28)\end{aligned}$$

The same procedure can be applied to both $\mathbf{H}_B$ and $\mathbf{H}_{AB}$. □

7.5.2 Block-Structured Layered Design Properties

We now show that the proposed block-structured layered design is indeed able to work with arbitrary source rates as well as with an arbitrary hierarchical ratio.

Theorem 7.4 (Block-Component Size Adjustment)**.**

We assume some desired given channel code rates r_A, r_B and r_{AB}. The channel encoders satisfying the desired coding rates can be composed according to Theorem 7.3 with rates of individual block-parts given by

$$r_{A1} = \frac{k_{AB}-k_B}{n_{AB}-n_B}, \quad r_{B1} = \frac{k_{AB}-k_A}{n_{AB}-n_A}, \quad r_2 = \frac{k_A+k_B-k_{AB}}{n_A+n_B-n_{AB}}.$$

Proof. Since $\mathbf{G}_A = \operatorname{diag}(\mathbf{G}_{A1},\mathbf{G}_2)$, $k_A = k_{A1}+k_2$ as well as $n_A = n_{A1}+n_2$. Similarly $\mathbf{G}_B = \operatorname{diag}(\mathbf{G}_{B1},\mathbf{G}_2)$ implies that $k_B = k_{B1}+k_2$ jointly with $n_B = n_{B1}+n_2$ and $\mathbf{G}_{AB} = \operatorname{diag}(\mathbf{G}_{A1},\mathbf{G}_{B1},\mathbf{G}_2)$ determines $k_{AB} = k_{A1}+k_{B1}+k_2$ and $n_{AB} = n_{A1}+n_{B1}+n_2$.

The particular rates can be obtained expressing $k_{A1} = k_{AB}-k_B$, $k_{B1} = k_{AB}-k_A$, $k_2 = k_A+k_B-k_{AB}$ and similarly for codeword lengths $n_\star$. See Figures 7.6 and 7.7. □

According to Theorem 7.4, we are able to compose the block decoders for arbitrary source rates $r_A = k_A/n_A$, $r_B = k_B/n_B$ and for the hierarchical rate $r_{AB} = k_{AB}/n_{AB}$. In case of hierarchical rate, we have not absolute freedom of choice, because $n_{AB} \leq n_A + n_B$. The rates are connected (for fixed codeword length) through the hierarchical ratio by the relation

$$k_{AB} = \lambda(k_A + k_B) \tag{7.29}$$

for full-rank $\mathbf{X}_b$. This relation is a straightforward consequence of (7.8).

We now focus how to achieve arbitrary hierarchical ratio. It seems to be useful to classify the hierarchical ratio λ according to some intervals, which will be investigated separately. Particularly

- $\lambda \geq (\max(k_A,k_B))/(k_A+k_B)$: The encoders can be composed according to Theorem 7.3 and the requested source rates are adjusted by using Theorem 7.4 for given r_A, r_B and λ. Notice that this condition can be rewritten for fixed codeword length using (7.29) to $k_{AB} \geq \max(k_A,k_B)$.
- $\lambda \leq (\min(k_A,k_B))/(k_A+k_B)$: The hierarchical ratio in this interval can be achieved adjusting $k_2 = \min(k_A,k_B) = k_{AB}$ and set the hierarchical matrices to

$$\mathbf{X}_{b,A} = \begin{bmatrix}\mathbf{0}\\ \mathbf{E}_{k_2}\end{bmatrix}, \quad \mathbf{X}_{b,B} = \begin{bmatrix}\mathbf{0}\\ \mathbf{E}_{k_2}\end{bmatrix}, \tag{7.30}$$

$$\mathbf{X}_{c,A} = \begin{bmatrix}\mathbf{0}\\ \mathbf{E}_{n_2}\end{bmatrix}, \quad \mathbf{X}_{c,B} = \begin{bmatrix}\mathbf{0}\\ \mathbf{E}_{n_2}\end{bmatrix}. \tag{7.31}$$

The only block-component part of the decoder in relay is a common encoder $\mathbf{G}_2$ and thus all methods of the decoders design across the networks relying on the *uniform hierarchical ratio* can be applied. The rest of the encoder design in sources can be provided without any restrictions, just according to needs of H-SI. Notice that if this case is considered as H-SI to the previous case $\lambda \geq (\max(k_A, k_B))/(k_A + k_B)$, either $\mathbf{X}_{b,A} = \mathbf{0}, \mathbf{X}_{c,A} = \mathbf{0}$ or $\mathbf{X}_{b,B} = \mathbf{0}, \mathbf{X}_{c,B} = \mathbf{0}$. In such a case the relay provides a single-stream decision to have a correct complementary information to H-SI. We can again simplify the condition defining this case to $k_{AB} \leq \min(k_A, k_B)$ for fixed codeword length.

- The remaining case $(\min(k_A, k_B))/(k_A + k_B) \leq \lambda \leq (\max(k_A, k_B))/(k_A + k_B)$ can occur only for $k_A \neq k_B$. Assuming without loss of generality that $k_A < k_B$, then $k_2 = k_A$ and $n_2 = n_A$. The hierarchical matrices securing the ratio from this range are given by

$$\mathbf{X}_{b,A} = \begin{bmatrix} \mathbf{0} \\ \mathbf{E}_{k_2} \end{bmatrix}, \quad \mathbf{X}_{b,B} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{E}_{k_{B12}} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{k_2} \end{bmatrix}, \tag{7.32}$$

$$\mathbf{X}_{c,A} = \begin{bmatrix} \mathbf{0} \\ \mathbf{E}_{n_2} \end{bmatrix}, \quad \mathbf{X}_{c,B} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{E}_{n_{B12}} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{n_2} \end{bmatrix}, \tag{7.33}$$

where the stream $\mathbf{b}_{B1}$ is further divided to $[\mathbf{b}_{B11}^T \mathbf{b}_{B12}^T]$ according to the requested hierarchical ratio λ. In this case, the relay decoder provides partly joint decoding of $\mathbf{b}_{B12}^T$ and partly decoding enabling the uniform hierarchical ratio based methods.

7.5.3 Signal Space Mappers and Soft Metric for Decoder

We leave the optimization of concrete signal space mappers in sources out of its focus. Instead, we address our attention to 1) a concrete way of the signal space symbol composition from the codeword sub-streams and 2) the (symbol-wise) soft metric evaluation that was investigated in Sec. 7.4.1. The proposed block-structured layered code requires just the soft metric for the block-decoders. Due to the memoryless channel, the metric is evaluated symbol-wisely in the SoDeM.

The proposed design offers a freedom, how the particular codeword sub-streams are mapped into the signal space. We can consider orthogonal (e.g. time) sharing methods as well as a common signal space symbol that contains information from all streams. Of cause a mixture of these options is also available. In the case of a common signal space symbol, one can see such signal space mapping as superposition coding for the multiple access broadcast channel shown in Fig. 7.2.

7.5.4 Block-Structured Layered Design Convergence Properties

In this part, we investigate the overall decoder performance properties. We will be interested mainly in case of hierarchical ratio $\lambda \geq (\max(k_A, k_B))/(k_A + k_B)$, because the other cases can be seen as special cases of this one from the decoder convergence point of view.

Due to the block-structured layered design, the block decoders $\mathbf{H}_{A1}$, $\mathbf{H}_{B1}$ and $\mathbf{H}_2$ work *independently* and the overall performance of the hierarchical decoder $\mathbf{H}_{AB}$ is limited by the weakest block decoder. The hierarchical decoder $\mathbf{H}_{AB}$ therefore operates on the rate of its weakest block sub-decoder.

This limitation can be naturally reduced, when bits from individual block encoders are combined into one higher order channel symbol and a SoDeM is applied to evaluate the soft metric for the decoder. A soft information exchange between the SoDeM and the decoder (see Fig. 7.8) is a conventional way of a performance improvement. In our case, the improvement is emphasized by the fact that the soft

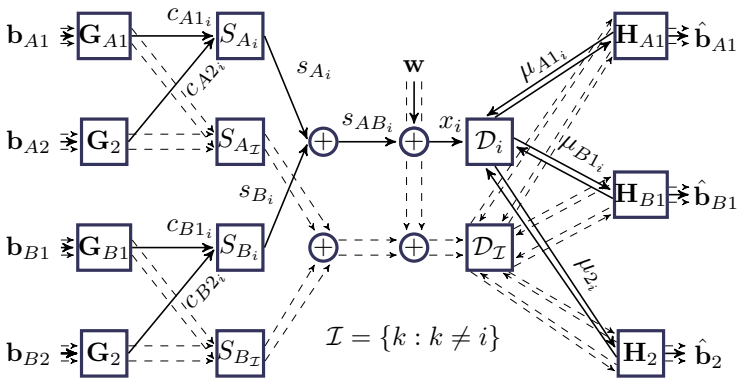


Figure 7.8: 2-Source 1-Relay system example from bit-wise processing point of view. It is assumed that $n_{A1} = n_{B1} = n_2$ for simplicity. The hierarchical decoder is shown in the decomposed form. A soft message μ_X is capable to carry the extrinsic information about c_X evaluated in both the decoder and SoDeM. Consequently, the block decoders are able to utilize the information from other block decoders carried via SoDeMs.

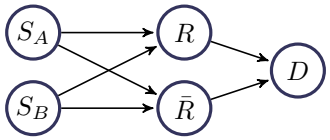


Figure 7.9: System model for the motivational example.

information exchange between SoDeM and decoder also *enables an information exchange between the independent individual block decoders.*

Even if one assumes the soft information exchange through the SoDeM, the overall decoder is still limited by the worst block-decoder although with aid from the SoDeM. To reduce this limitation, we can consider interleaving of the codewords in sources prior they are mapped into the signal space to average the encoders convergence properties. The metric evaluated in the SoDeM must be then properly deinterleaved and combined according to needs of a particular method.

The information exchange can be described by means of EXIT charts [19, 56]. We assume that the conditions required for the EXIT chart [56] analysis hold, that is the independent input to soft in soft out (SISO) blocks and Gaussian distribution of the systematic log-likelihood ratio (LLR).

7.6 Block Structured Layered Design in Simple Example

In order to better understand the proposed concept, we provide its demonstration in a simple wireless network design.

We consider a simple two-source two-relay one destination wireless network without a direct link from the sources to destination (see Fig. 7.9). The destination aims with decoding both data streams.

We distinguish 3 sub-streams referring to $\mathbf{b}_{AB}$ in the relay R, which are formed by the individual streams inputing the encoders $\mathbf{G}_{A1}$, $\mathbf{G}_{B1}$ ($\mathbf{b}_{A1}$, $\mathbf{b}_{B1}$) and a virtual stream $\mathbf{b}_2 = \mathbf{b}_{A2} \oplus \mathbf{b}_{B2}$ (upon GF(2)) encoded by $\mathbf{G}_2$. The second (complementary) hierarchical stream $\bar{\mathbf{b}}_{AB}$ in the relay $\bar{R}$ must therefore contain at least either $\mathbf{b}_{A2}$ or $\mathbf{b}_{B2}$ to secure decodability in the final destination. It means that the hierarchical ratio in R is $\lambda \geq (\max(k_A, k_B))/(k_A + k_B)$ (it means extended cardinality in terms of [54]) in our notation. To

λ	r_{A1}	r_{B1}	r_2	r_{AB}
9/14	1/2	1/6	5/6	9/18
10/14	2/3	1/3	2/3	10/18
11/14	5/6	1/2	1/2	11/18

Table 7.1: Particular block-codes rates for different λ and fixed $r_A = 1/2$, $r_B = 2/3$ evaluated according to (7.34-7.37).

make the example more straightforward, we assume given fixed coding rates in sources, i.e. r_A and r_B.

7.6.1 Adjusting Block Rates

Fixing the length of codewords to $n_{A1} = n_{B1} = n_2 = n$ and considering the case $\lambda \geq (\max(k_A, k_B))/(k_A + k_B)$, we can express the particular block component rates as

$$r_{A1} = 2r_A\lambda - 2r_B(1-\lambda) \tag{7.34}$$
$$r_{B1} = 2r_B\lambda - 2r_A(1-\lambda) \tag{7.35}$$
$$r_2 = 2(r_A + r_B)(1-\lambda). \tag{7.36}$$

The virtual hierarchical rate r_{AB} can be shown to be

$$r_{AB} = \frac{2}{3}\lambda(r_A + r_B). \tag{7.37}$$

Some examples of the block rates evaluation for different λ can be seen in Tab. 7.1.

As an example, we fix $r_A = 1/2$ and $r_B = 2/3$ and choose the middle line in Tab. 7.1. We further construct the particular block components as conventional simple block coders on the corresponding rates. We choose the block codeword length n as short as possible, that is $n = 3$ for given rates and the hierarchical ratio. The block decoders properly respecting the requested rates can be then considered e.g.

$$\mathbf{H}_{A1} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}, \mathbf{H}_{\mathrm{B1}} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \tag{7.38}$$

and

$$\mathbf{H}_2 = \begin{bmatrix} \iota & \iota & \iota \end{bmatrix}, \tag{7.39}$$

where $\iota = 1$ in case of GF(2) and $\iota = 1 + D$ in GF(4) implementation case.

The block composition of the hierarchical decoder according to Theorem 7.3 is then given by

$$\mathbf{H}_{AB} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \iota & \iota & \iota \end{bmatrix}. \tag{7.40}$$

7.6.2 Signal Space Mapping

The encoders and the signal space mappers in sources including the bit mapping jointly with the adequate receiver construction in relay are shown in Fig. 7.10, where the channel parameters are for simplicity considered as $h_A = h_B = 1$. The codeword bits are mapped in the order $[c_{A1}, c_{A2}, c_{B1}, c_{B2}]$ to the superimposed modulation seen in relay, that is [blue, violet, cyan, magenta] order of colors in Fig. 7.10.

The hierarchical (symbol-wise) codeword mapping is therefore given either by[10] $c_{AB} = [c_{A1}, c_{B1}, c_{A2} \oplus c_{B2}]$ (upon GF(2)) or by $c_{AB} = [c_{A1}, c_{B1}, c_{A2} + Dc_{B2}]$ upon GF(4). This can be written using matrices $\tilde{\mathbf{X}}_{c,A}$ and $\tilde{\mathbf{X}}_{c,B}$ in a block diagonal form. The desired block form of $\mathbf{X}_{c,A}$ and $\mathbf{X}_{c,B}$ can be obtained by a column swapping properly reflected in SoDeM.

7.6.3 Soft Metric Evaluation

The decoder in relay (see bottom part of Fig. 7.10) incorporates the soft metric symbol-wisely evaluated in the SoDeM. One can see that the particular streams are mapped into a certain position in the superimposed modulation (see different colors in Fig. 7.10) and thus the bits associated to different streams are unequally resistant against the noise. The connection of the decoder and the demodulator can be seen also in Fig. 7.8, where the soft information exchange is shown.

The unequal resistance of the streams against the noise can be equalized[11] by averaging using interleaver over a *whole codeword* as shown in Fig. 7.10. This can be simply implemented such that both the codewords $\mathbf{c}_A$ and $\mathbf{c}_B$ are interleaved by an equal interleaver of length $n_A = n_B = 2n$. The joint demodulator-decoder shown in Fig. 7.10 illustrates the mapping connection for both interleaved and non-interleaved case.

In case, where the interleaver is not used, the first bit of the superimposed symbol is always associated with the decoder $\mathbf{H}_{A1}$, similarly the third bit with $\mathbf{H}_{B1}$ and finally both second and fourth bit with $\mathbf{H}_2$ as one can see in Fig. 7.10. On the other hand, the use of interleaver causes that all bits are associated with the particular decoders randomly as show in Fig. 7.10 (thick dashed lines). Note that particular metrics as shown in Fig. 7.8 are evaluated by marginalization of the joint likelihood $p(x|c_A, c_B)$ in SoDeM exploiting a priory information from other streams.

7.6.4 Overall Network Design

Up to this time, we investigated processing only in one of the relays. We set the particular block rates for the block-component decoder in the relay. It is obvious that we will cannot independently set arbitrary rates in the complementary relay. It means that the complementary relay seems to be forced to decode at least either $\mathbf{b}_{A2}$ or $\mathbf{b}_{B2}$ on rate r_2 in order to have an error-less minimal complementary information required in destination. Nevertheless, we have available at least two approaches how to solve the inconvenience, when the stand-alone rate r_2 is not directly achievable in $\bar{R}$. The first one is that we have choice to increase the power in source at expanse of the second transmitted data stream (e.g. $\mathbf{b}_{B1}$). The second choice might be to communicate on the rate of a whole data codeword or some of its part[12] (e.g. whole $\mathbf{b}_B$). As we will observe (Fig. 7.12), the enabled loops between the SoDeM and decoder in case of higher order modulations enables to operate on some averaged rate (e.g. $r_B < r_2$ in our case).

If we use this averaged rate, we have more information that is directly needed for the final destination in the relays. Considering a decoding of the whole data vector in $\bar{R}$, that is $\bar{\mathbf{b}}_{AB} = \mathbf{b}_B$ in our case, the relays have available the following streams:

- Relay R: $\mathbf{b}_{A1}$, $\mathbf{b}_{B1}$, $\mathbf{b}_2 = \mathbf{b}_{A2} \oplus \mathbf{b}_{B2}$
- Relay $\bar{R}$: $\mathbf{b}_{B1}$, $\mathbf{b}_{B2}$.

It can be clearly seen that the stream $\mathbf{b}_{B1}$ is available in both relays. The relays can therefore transmit all streams, where some power must be allocated to this redundant information. On the other hand, it is

[10] Notice that this concrete setup may be seen similar to the approach proposed in [22]. The fundamental difference is that their approach is based on clear network coding, that is the aforementioned relation is *only* upon the hierarchical data.

[11] Notice that this averaging is not necessary. One can consider the case when the unequal error protection is exploited e.g. to enable a particular block decoder operation on higher rate. Such a signal space mapping design is a challenging question that is addressed for future work.

[12] Further block structuring can be considered.

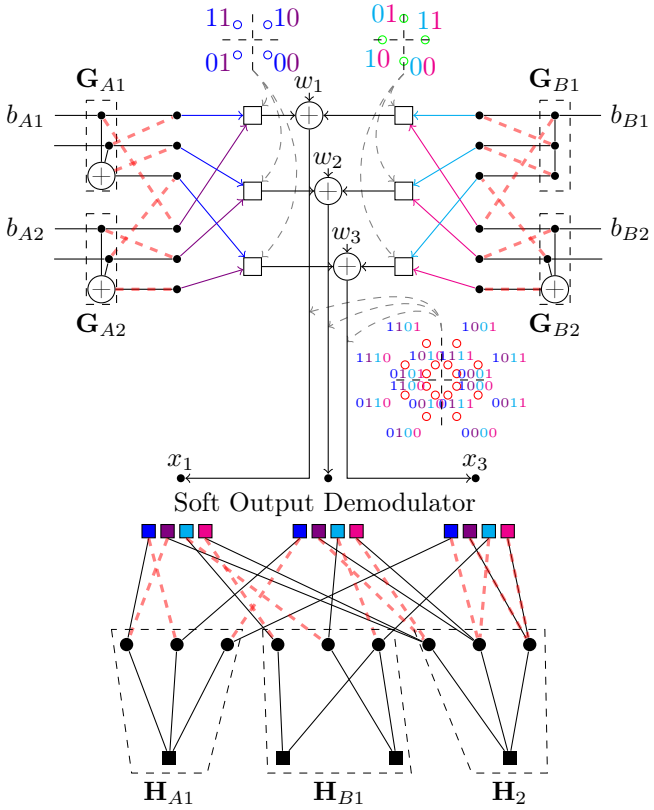


Figure 7.10: The bit mapping visualization including the source encoders and relay decoder. The connection of SoDeM and decoders from the soft information transfer point of view can be seen in Fig. 7.8 for the case without interleaver.

possible to save this power and to transmit only the required information (no diversity). A good question is, if the gain of the proper correlated sources utilization, i.e. the distributed source coding in the relays R, $\bar{R}$ and the corresponding decoding in the final destination, is more attractive than the saved power required for the system diversity.

7.7 Numerical Verification

Since our work is modular, there is not an obvious reference, what should be numerically verified to demonstrate any gain of the proposed work[13]. We rather focus on the waterfall-behavior of BER of the proposed decoders proving proper functionality.

7.7.1 Models to be Numerically Simulated

We consider several ways of the hierarchical stream $\mathbf{b}_{AB}$ evaluation from the received superimposed vector $\mathbf{x}$. We assume that we have given requested *asymmetric* code rates for the source channel encoders like in the example in Sec. 7.6.

As a reference case, we choose the conventional joint decoding strategy with successive interference cancellation followed by the network coding (Sec. 7.4.2).

We focus on the following features to be verified by the numerical simulations.

1. Performance of the proposed framework.

2. Performance (BER) difference of the relay receiver, where the hierarchical decoder and SoDeM work separately (i.e. loops between SoDeM and decoder are disabled) against decoding with enabled loops. (Figs. 7.11, 7.12)

3. Performance difference of different decoder construction, particularly

   - hierarchical decoder operating whole on GF(2) without interleaver,
   - hierarchical decoder operating whole on GF(2) with interleaver,
   - hierarchical decoder operating partly on GF(4) with interleaver, that is the decoder $\mathbf{H}_2$ works on GF(4).

   The results for a given hierarchical ratio are shown in Fig. 7.13.

4. Trade-off between the diversity of the processing and stronger broadcast phase as discussed in Sec. 7.6.4 at least in a particular scenario (Fig. 7.15).

The first three features can be shown on 2-source 1-relay scenario and the last one is suited into a whole network as shown in Fig. 7.9. Nevertheless to keep the things simple and better analyzable, we consider a subset of the whole model containing only the relays and the final destination for investigation of the fourth feature. The relays are assumed to be able to correctly decode the requested data streams, i.e. they can be seen as (correlated) sources.

[13] We remind that the main contribution of this work consists in the relay decoder construction for arbitrary source rates, where the decoder takes advantage the wireless network coding.

7.7.2 Simulation Setup and Outer Parameters

We compose a system to provide a numerical verification of the proposed design. We consider the equal system model as in Sec. 7.6, where only one difference is in the codeword lengths $n_{A1} = n_{B1} = n_2 = n = 2400$ and the block-coders are assumed to be regular LDPC codes with maximized girth according to [35]. We choose (ad-hoc) the signal space mappers at bit level to be QPSK for both s_A and s_B. The bit mapping is ad-hoc selected to maximize the minimal hierarchical distance. The symbol received in relay is given by $x = s_A + hs_B + w$, where $h = \exp(\mathrm{j}\,\pi/4)$ and w is assumed to be a zero-mean AWGN. The relay evaluates a corresponding soft metric in SoDeM. The model is in fact shown in Fig. 7.10, where some longer encoders and interleaver are considered.

To demonstrate the system diversity, we consider the hierarchical matrices as

$$\mathbf{X} = \begin{bmatrix} \mathbf{E}_{k_{A1}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{E}_{k_{B1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_{k_2} & \mathbf{0} & \mathbf{E}_{k_2} \end{bmatrix} \tag{7.41}$$

and two complementary matrices,

$$\bar{\mathbf{X}} \in \left\{ \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E}_{k_2} \end{bmatrix}, \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{E}_{k_{B1}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{E}_{k_2} \end{bmatrix} \right\}. \tag{7.42}$$

It means that the first relay R has available the streams $[\mathbf{b}_{A1}, \mathbf{b}_{B1}, \mathbf{b}_2 = \mathbf{b}_{A2} \oplus \mathbf{b}_{B2}]$ and the complementary relay $\bar{R}$ has available either stand-alone $[\mathbf{b}_{B2}]$ or both $[\mathbf{b}_{B1}\ \mathbf{b}_{B2}]$ as in Sec. 7.6.4. For simplicity, we assume that $k_{A1} = k_{B1} = k_2 = k = 1600$ in this case (code rate is $2/3$ for all streams). All streams are encoded in the relays by equal encoder and mapped to BPSK modulation, where the mean energy of the signal $\bar{\mathbf{s}}_{B2,\mathrm{min}}(\mathbf{b}_{B2})$ referring to the scheme without diversity is split between $\bar{\mathbf{s}}_{B1,\mathrm{mid}}(\mathbf{b}_{B1})$ and $\bar{\mathbf{s}}_{B2,\mathrm{mid}}(\mathbf{b}_{B2})$ in the diverse scheme, where the "min" subscript denotes no diversity (minimal complementary information) and "mid" subscript denotes the diversity given by a redundant stream $\mathbf{b}_{B1}$. The streams are then transmitted *orthogonally* into the final destination through AWGN channel, where the decoder based on the factor graph is performed to decode the streams.

7.7.3 Simulation Results

First of all, we can recognize the waterfall behavior of all BER curves indicating a proper functionality of the proposed design. The performance of the individual streams is given by signal space properties and by the rate of the referring block-decoder. We can see (Figs. 7.11 and 7.12) that the overall hierarchical decoder is indeed limited by its weakest sub-decoder, when the iterative loop via SoDeM is not applied.

The soft information exchange between SoDeM and decoder significantly improves the performance (see Figs. 7.11 and 7.12). The properties of the block decoders are averaged and the hierarchical encoder definitely operates on a larger rate then its stand-alone weakest block decoder. The EXIT curves of the hierarchical decoder are shown in Fig. 7.14.

Using the block decoder $\mathbf{H}_2$ working upon higher field, i.e. GF(4), we obtain some further performance gain (Fig. 7.13), which is however paid by higher complexity of the relay decoder. Using the interleaver to average the quality of the metric serving to the block decoders does not have any important impact to the performance as one can check in Fig. 7.13 for given rates and the signal space mapping.

According to the provided results, we can conclude that the hierarchical (direct) decoding (H_2 upon GF(2)) achieves similar performance to the reference JDF over NC with less complexity, while the decoding with (H_2 upon GF(4)) benefits approximately half dB gain against the reference solution for given scenario (see Fig. 7.13).

The performance comparison between diverse and no-diverse scheme can be found in Fig. 7.15. The diversity improves the performance of the stream $\mathbf{b}_{B1}$, while the performance of the streams $\mathbf{b}_{A2}, \mathbf{b}_{B2}$ is

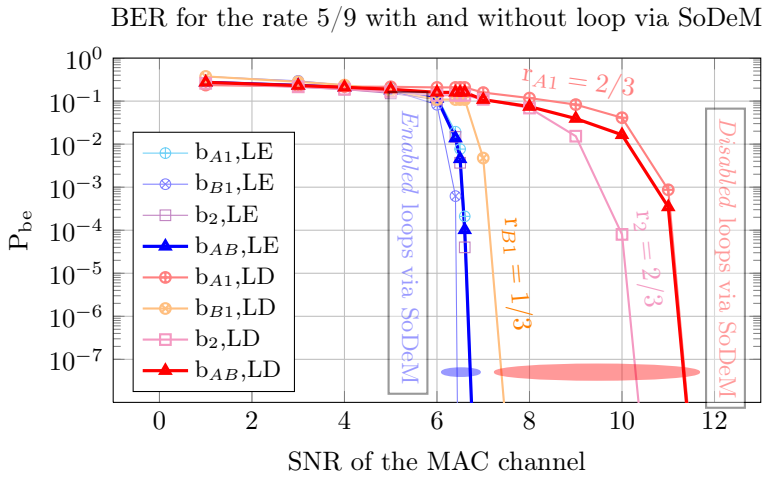


Figure 7.11: BER of the hierarchical stream $\mathbf{b}_{AB}$ and its individual sub-streams $\mathbf{b}_{A1}$,$\mathbf{b}_{B1}$ and $\mathbf{b}_2$ in case of pure decoder (loop disabled-LD) and decoder with enabled information exchange with SoDeM (LE) for different overall rates.

worse due to the weaker relay to destination link (transmission power must be shared with the stream $\mathbf{b}_{B1}$).

7.8 Generalization and Future Work

The area that is covered in this work presents a part of the overall wireless network design. Particularly, the proposed design is able to incorporate:

- Arbitrary network information flow expressed by means of hierarchical ratio.
- Arbitrary source rates.
- Arbitrary symbol-wise signal space mapping, where a white space in both the modulation design and a concrete bit mapping optimization arises.

The generalization can be considered in several ways, particularly:

- general channels across the network,
- complex network (more sources and streams),
- allowing non-linear coding and hierarchical functions.

The first way of generalization is relatively straightforward, because the core of this work needs just a metric for decoders.

In case of linear system description, we can easily generalize the relation of the virtual decoder in relay and the codeword hierarchical function (7.5) and (7.6). We denote $\mathcal{R} = \{A, B, \dots\}$ as a set of all sources, then

$$\mathbf{c}_{\mathcal{R}} = \sum_{i\in\mathcal{R}} \mathbf{X}_{c,i}\mathbf{c}_i = \sum_{i\in\mathcal{R}} \mathbf{X}_{c,i}\mathbf{G}_i\mathbf{b}_i \tag{7.43}$$

$$= \mathbf{G}_{\mathcal{R}}\mathbf{b}_{\mathcal{R}} = \mathbf{G}_{\mathcal{R}} \sum_{i\in\mathcal{R}} \mathbf{X}_{b,i}\mathbf{b}_i. \tag{7.44}$$

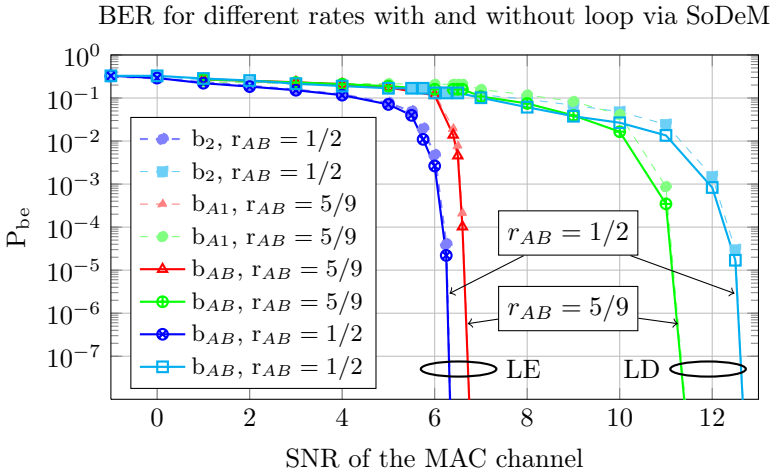


Figure 7.12: Bit error rate verification of the proposed design for the overall hierarchical stream $\mathbf{b}_{AB}$ and its weakest individual sub-streams from $\mathbf{b}_{A1}$,$\mathbf{b}_{B1}$ and $\mathbf{b}_2$ for different hierarchical rates (as well as hierarchical ratios). The BER simulation is performed in two cases: 1) pure decoder, where the hierarchical decoder works as three independent decoders and therefore it operates on the rate of its weakest block-part and 2) decoder with enabled information exchange with SoDeM. As one can expect, the SoDeM usage results in a performance gain. Moreover, the enabled information exchange between the SoDeM and decoder causes an averaging of the sub-decoders properties such that the hierarchical decoder is not further limited by the *rate* of its weakest block sub-decoder and it operates on the averaged (hierarchical) rate r_{AB}.

Generalization towards more relays is relatively straightforward, because we just considered a complementary H-SI, but our approach does not rely how many such H-SI links are presented. The information streams can be further separated in order to comply the requested source coding rates.

Nevertheless, in case of more sources, the information flow can request either a decision upon more streams (e.g. $\mathbf{b}_A + \mathbf{b}_B + \mathbf{b}_C$) and/or decision upon doublets (triples, etc.) of data streams (e.g. $\{\mathbf{b}_A + \mathbf{b}_B\}, \{\mathbf{b}_B + \mathbf{b}_C\}$). In such a case a revision of the block structured layered should be performed, but the basic building blocks based on *uniform hierarchical ratio* for N sources with equal rates such that $\lambda = 1/n$ are preserved. The encoder design based on equal source and relay decoder can be applied accordingly, where the order of finite field is arising in case of the optimal decoding.

7.9 Discussion and Conclusions

We proposed a design of the joint network-channel encoding for linear encoders and hierarchical functions enabling to incorporate an arbitrary individual rates. This design is based on a predefined block-structure of both the channel encoders and the hierarchical functions. We proved the network consistency of the proposed design. We shown that the overall hierarchical decoder inherits the properties of its sub-decoders. We numerically verified the performance of such a design.

We further verified the implementation of the proposed design, where we consider a decoding with both enabled and disable iterative loops through the soft output demodulator. The loops connecting the block-diagonal parts enable the information exchange among the block-component parts causing averaging of the particular block-rates.

Finally, we shown that the proposed linear description enables an extension of the principle to more complex network topologies, where further block structuring can be considered. Nevertheless a detailed

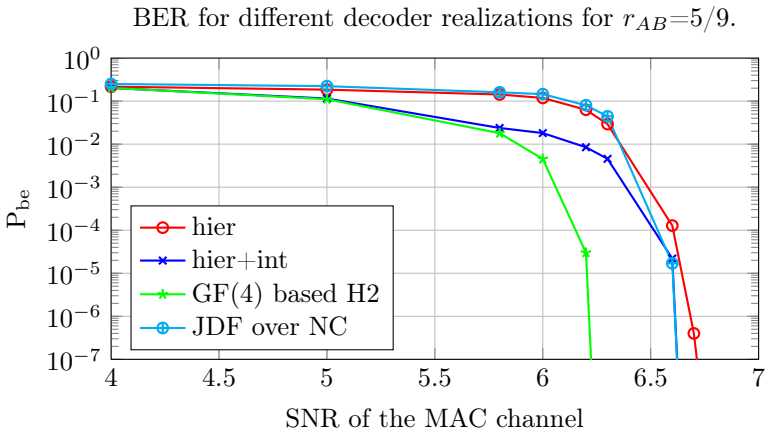


Figure 7.13: Comparison of the relay-decoder performance decoder for different realizations. We consider conventional decoders as G_{A1} and G_{B1} and different variants of G_2 that are (1) hierarchical form, i.e. GF(2) decoder with $G_2 = G_{A2} = G_{B2}$ with and without interleaver, (2) optimal form, i.e. GF(4) as in [64].

study of these more complex network topologies is addressed for the future work.

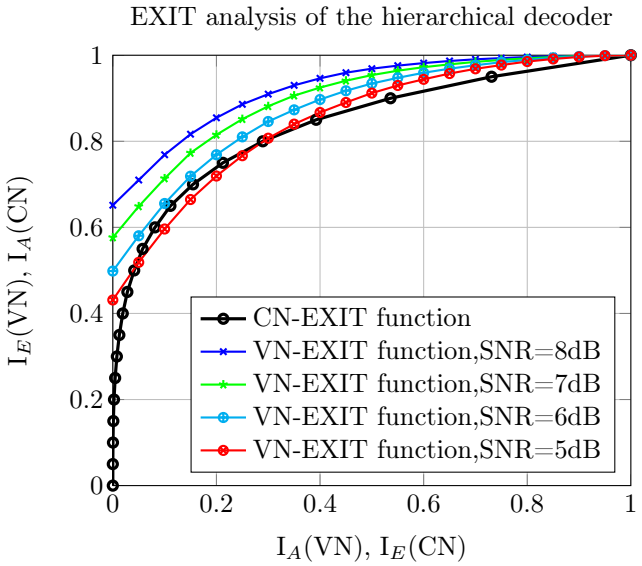


Figure 7.14: EXIT behaviour of the hierarchical decoder operation on rate 5/9 for different SNR of the MAC channel. Comparing to Fig. 7.13, we can see that the 5 dB level of the SNR in the MAC channel indeed does not work well. In case of 6 dB, one can expect an error-less behaviour according to the EXIT chart. Nevertheless the cliff appears approximately in the level 6.7 dB. This gentle inaccuracy is probably caused by a slight dependency of the SISO block inputs violating the assumptions for the EXIT chart measurement.

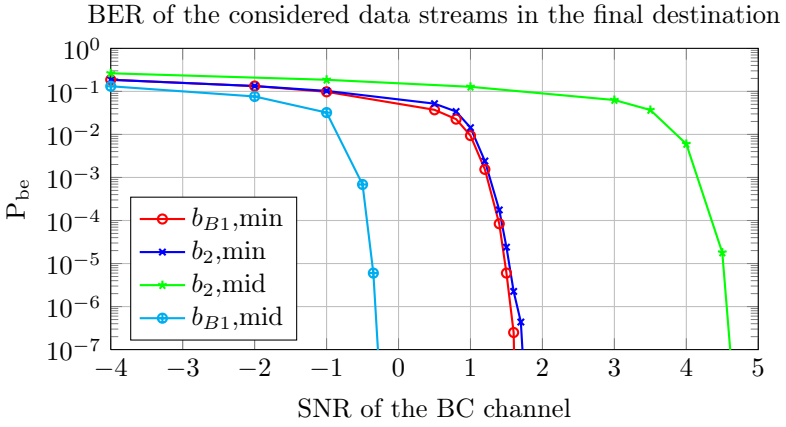


Figure 7.15: The performance of the particular streams in the final destination. We dropped the performance of the stream $\mathbf{b}_{A1}$, because it is equal to the performance of the stream $\mathbf{b}_{B1}$ in case of no diversity. This case is denoted by the min subscript, because only minimal information is passed towards the destination. The diversity case is denoted by the mid subscript, because the information passed towards the destination is somewhere in middle between the minimal information and the full information. Concretely, we assume that $\mathbf{b}_{B1}$ is transmitted from both relays in the "middle case".

8 Overall Conclusions

The main topic of this thesis is the receiver design in a multi-source multi-node wireless network. We cope this topic from two layers. The first one is the overall network design on the channel code level, that is to design a joint consistent cross-network channel code that satisfies some requests. The latter consists in a direct implementation of the generic sum product algorithm that is capable to describe among others the receiver in the wireless network. One can see a clear relation between the mentioned topics, that is the overall network design properly and efficiently implemented by means of the sum product algorithm.

The sum product algorithm direct implementation was solved by the implementation framework containing the proposed generic KLT-message representation and the update rules design upon the message representation that are applicable among others to the proposed generic KLT message representation. We numerically figured out that the messages in a wide class of phase models lead to the harmonic eigenfunctions applying the KLT on their stochastic description. This proves the Fourier representation to be the best possible linear message representation in the MSE sense. We can note that the implementation framework relying on the Fourier message representation was independently proposed in [5]. We verified the capability of the KLT (Fourier) representation by a numerical simulation for a given model, where the performance results are shown in Figures 5.13, 5.14 and 5.15.

We further proposed an update rules design for an arbitrary linear message representation with orthogonal canonical kernels. The assumption of the linearity and orthogonality is inherently satisfied for the KLT-message representation. The proposed update rules design jointly with the KLT-message representation therefore form a generic implementation framework applicable whenever the stochastic description of the messages is available, i.e. KLT of the random process given by the different message realizations can be evaluated. The framework is moreover built upon the best linear message approximation in the MSE sense for a given dimensionality as a consequence of KLT.

The Fourier representation forms an orthogonal kernel basis system. Being aware of the eigen-analysis results (see Figures 5.10, 5.11 and 5.12), we used the Fourier representation for the proposed implementation framework verification in the given joint data detection-phase estimation system model. The proposed implementation framework that results in a highly efficient implementation (check Table 5.1) for this particular scenario.

The demand for a flexible wireless aware joint network-channel code design becomes clear after proving the wireless network coding potential (see e.g. [38]). We started with defining a robust system description. This description is based on the system linearity, that is the linear hierarchical function (in fact a wireless aware network code - see Note 6.4) and the channel code. The linearity assumption is satisfied by conventional capacity achieving codes (e.g. LDPC or turbo codes). We therefore form a Theorem 7.3 defining a block structure of the channel codes across the network. The relation of the inner block sizes (diagonal structure is assumed) determines the particular channel code rates. Arbitrary channel code rates can be set adjusting properly the sizes of inner blocks (Theorem 7.4).

The proposed block structured layered design basically combines two known principles, where both the requested hierarchical ratio (amount of information flowing through the relay) and the requested rates are achieved in average over a whole codeword. We also discussed some possible structures of the decoder properly respecting the block structured layered design. This overall decoder is basically composed from the conventional LDPC block-decoders, where the soft information exchange between the (independent) block parts can be conducted e.g. via soft output demodulator for higher order alphabets. The performance for different metrics evaluated in the soft output demodulator and corresponding decoder structure is evaluated in a particular system model (check Figures 7.12 and 7.11). These results proves the waterfall behaviour indicating a proper decoder implementation.

Index

Bibliography

[1] Rudolf Ahlswede, Ning Cai, Shuo-Yen Robert Li, and Raymond W. Yeung. Network information flow. *IEEE Trans. Inf. Theory*, 46(4):1204–1216, July 2000.

[2] Hans andrea Loeliger. Some remarks on factor graphs. In *Proc. 3rd Int. Symp. on Turbo Codes and Related Topics*, pages 111–115, 2003.

[3] Khoirul Anwar and Tad Matsumoto. Accumulator-assisted distributed turbo codes for relay systems exploiting source-relay correlation. *Communications Letters, IEEE*, 16(7):1114–1117, 2012.

[4] Xingkai Bao and JT Li. Generalized adaptive network coded cooperation (gancc): A unified framework for network coding and channel coding. *IEEE Trans. Commun.*, 59(11):2934–2938, 2011.

[5] Alan Barbieri, Giulio Colavolpe, and Giuseppe Caire. Joint iterative detection and decoding in the presence of phase noise and frequency offset. In *Proc. IEEE Internat. Conf. on Commun. (ICC)*, 2005.

[6] Claude Berrou and Alain Glavieux. Near optimum error correcting coding and decoding: Turbo-codes. *IEEE Trans. Commun.*, 44(10):1261–1271, October 1996.

[7] U. Bhat and T. Duman. Decoding strategies at the relay with physical-layer network coding. *IEEE Trans. Wireless Commun.*, pages 4503–4513, December 2012.

[8] M.C. Castro, B.F. Uchoa-Filho, T.T.V. Vinhoza, M. Noronha-Neto, and J. Barros. Improved joint turbo decoding and physical-layer network coding. In *Proc. IEEE Inf. Theory Workshop (ITW)*, pages 532–536. IEEE, 2012.

[9] Keith Chugg, Achilleas Anastasopoulos, and Xiaopeng Chen. *Iterative Detection: Adaptivity, Complexity Reduction, and Applications*. Kluwer Academic Publishers, 2001.

[10] Giulio Colavolpe, Alan Barbieri, and Giuseppe Caire. Algorithms for iterative decoding in the presence of strong phase noise. *IEEE J. Sel. Areas Commun.*, 23(9):1748–1757, September 2005.

[11] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

[12] Justin Dauwels and Hans andrea Loeliger. Phase estimation by message passing. In *Proc. IEEE Int. Conf. on Communications*, pages 523–527, 2004.

[13] Justin H. G. Dauwels. *On Graphical Models for Communications and Machine Learning:Algorithms, Bounds, and Analog Implementation*. PhD thesis, Swiss Federal Institute of Technology, Zürich, May 2006.

[14] M.C. Davey and D.J.C. MacKay. Low density parity check codes over gf(q). In *Information Theory Workshop, 1998*, pages 70 –71, jun 1998.

[15] Abbas El Gamal and Young-Han Kim. *Network Information Theory*. Cambridge University Press, 2011.

[16] G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the Twenty-second Conference on Uncertainty in AI (UAI)*, Boston, Massachussetts, July 2006.

[17] Uri Erez and Ram Zamir. Achieving $1/2 \log(1 + \mathrm{SNR})$ on the AWGN channel with lattice encoding and decoding. *IEEE Trans. Inf. Theory*, 50(10):2293–2314, October 2004.

[18] Gianluigi Ferrari, Giulio Colavolpe, and Riccardo Raheli. A unified framework for finite-memory detection. *Selected Areas in Communications, IEEE Journal on*, 23(9):1697–1706, 2005.

[19] Alex Grant. Convergence of non-binary iterative decoding. In *Proc. IEEE Global Telecommun. Conf. (GlobeCom)*, 2001.

[20] Z. Guo, J. Huang, B. Wang, S. Zhou, J.H. Cui, and P. Willett. A practical joint network-channel coding scheme for reliable communication in wireless networks. *IEEE Trans. Wireless Commun.*, 11(6):2084–2094, 2012.

[21] Joachim Hagenauer, Elke Offer, and Lutz Papke. Iterative decoding of binary block and convolutional codes. *IEEE Trans. Inform. Theory*, 42:429–445, 1996.

[22] C. Hausl, F. Schreckenbach, I. Oikonomidis, and G. Bauch. Iterative network and channel decoding on a tanner graph. In *Proc. Allerton Conf. on Commun., Control and Computing*. Citeseer, September 2005.

[23] Sheryl L Howard, Vincent C Gaudet, and Christian Schlegel. Soft-bit decoding of regular low-density parity-check codes. *Circuits and Systems II: Express Briefs, IEEE Transactions on*, 52(10):646–650, 2005.

[24] Steven M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, 1993.

[25] Steven M. Kay. *Fundamentals of Statistical Signal Processing: Detection Theory*. Prentice-Hall, 1998.

[26] Toshiaki Koike-Akino, Petar Popovski, and Vahid Tarokh. Optimized constellations for two-way wireless relaying with physical network coding. *IEEE J. Sel. Areas Commun.*, 27(5):773–787, June 2009.

[27] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory*, 47(2):498–519, February 2001.

[28] Brian M. Kurkoski and Justin Dauwels. Belief-Propagation Decoding of Lattices Using Gaussian Mixtures. *CoRR*, abs/0904.4741, 2009.

[29] J.N. Laneman, D.N.C. Tse, and G.W. Wornell. Cooperative diversity in wireless networks: Efficient protocols and outage behavior. *Information Theory, IEEE Transactions on*, 50(12):3062 – 3080, dec. 2004.

[30] Soung Chang Liew, Shengli Zhang, and Lu Lu. Physical-layer network coding: Tutorial, survey, and beyond. *in Phys. Commun.*, 2011 [Online]. Available: http://arxiv.org/abs/1105.4261, 2011.

[31] J. Liu, M. Tao, and Y. Xu. Pairwise Check Decoding for LDPC Coded Two-Way Relay Block Fading Channels. *IEEE Trans. Commun.*, PP(99):1–12, May 2012.

[32] H.-A. Loeliger, J. Dauwels, Junli Hu, S. Korl, Li Ping, and F.R. Kschischang. The Factor Graph Approach to Model-Based Signal Processing. *Proceedings of the IEEE*, 95(6):1295–1322, June 2007.

[33] Hans-Andrea Loeliger. An introduction to factor graphs. *IEEE Signal Process. Mag.*, 21(1):28–41, January 2004.

[34] J.A. McGowan and R.C. Williamson. Loop Removal from LDPC Codes. In *Information Theory Workshop*, pages 230 – 233, april 2003.

[35] James A McGowan and Robert C Williamson. Loop removal from ldpc codes. In *Information Theory Workshop, 2003. Proceedings. 2003 IEEE*, pages 230–233. IEEE, 2003.

[36] J. M. Mooij. *Understanding and Improving Belief Propagation*. PhD thesis, Radboud University Nijmegen, May 2008.

[37] Joris M. Mooij and Hilbert J. Kappen. Sufficient conditions for convergence of the sum-product algorithm. *IEEE Trans. Inf. Theory*, pages 4422–4437, May 2007.

[38] Wooseok Nam, Sae-Young Chung, and Yong H. Lee. Capacity bounds for two-way relay channels. In *Proc. Int. Zurich Seminar on Communications*, 2008.

[39] B. Nazer and M. Gastpar. Reliable physical layer network coding. *Proceedings of the IEEE*, 99(3):438–460, 2011.

[40] Bobak Nazer and Michael Gastpar. Compute-and-forward: Harnessing interference through structured codes. *IEEE Trans. Inf. Theory*, 57(10):6463–6486, October 2011.

[41] O. Oyman, N.J. Laneman, and S. Sandhu. Multihop relaying for broadband wireless mesh networks: From theory to practice. *Communications Magazine, IEEE*, 45(11):116 –122, November 2007.

[42] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, second edition, 1984.

[43] John G. Proakis. *Digital Communications*. McGraw-Hill, 4th edition, 2001.

[44] Pavel Prochazka and Jan Sykora. Karhunen-Loève based reduced-complexity representation of the mixed-density messages in SPA on Factor Graph and its impact on BER. *EURASIP J. on Wireless Comm. and Netw.*, 2010:11, December 2010.

[45] Pavel Prochazka and Jan Sykora. Block-structure based extended layered design of network coded modulation for arbitrary individual using hierarchical decode & forward strategy. In *In COST IC1004 MCM, Bristol, UK*, pages 1–7, September 2012.

[46] Pavel Prochazka and Jan Sykora. Block-Structured Layered Design of Wireless Aware Joint Network-Channel Coding for Asymmetric Rates using HDF Relaying Strategy. *IEEE Trans. Wireless Commun.*, July 2013. Submitted for publication.

[47] Pavel Prochazka and Jan Sykora. Generic efficient fg-spa implementation framework based on the best linear canonical message representation in mse sense. *European Wireless 2013*, 2013.

[48] Boris Rankov and Armin Wittneben. Spectral efficient protocols for half-duplex fading relay channels. *Selected Areas in Communications, IEEE Journal on*, 25(2):379–389, 2007.

[49] Tom Richardson and Ruediger Urbanke. *Modern Coding Theory*. Cambridge University Press, New York, NY, USA, 2008.

[50] Bixio E. Rimoldi. A decomposition approach to CPM. *IEEE Trans. Inf. Theory*, 34(2):260–270, March 1988.

[51] Christian B. Schlegel and Lance C. Perez. *Trellis and Turbo Coding*. John Wiley & Sons, 2004.

[52] Frederik Simoens and Marc Moeneclaey. Code-aided estimation and detection on time-varying correlated MIMO channels: A factor graph approach. *EURASIP J. on Advances in Signal Processing*, 2006:1–11, 2006.

[53] Jan Sykora. Factor graph framework for serially concatenated coded CPM with limiter phase discriminator receiver. In *Proc. IEEE Vehicular Technology Conf. (VTC)*, pages 1–6, Baltimore, USA, October 2007.

[54] Jan Sykora and Alister Burr. Layered design of hierarchical exclusive codebook and its capacity regions for HDF strategy in parametric wireless 2-WRC. *IEEE Trans. Veh. Technol.*, 60(7):3241–3252, September 2011.

[55] Jan Sykora and Pavel Prochazka. Performance evaluation of the factor graph CPM phase discriminator decoder with canonical messages and modulo mean updates. In *Proc. IEEE Asia-Pacific Conf. on Communications (APCC)*, pages 1–5, Tokyo, Japan, October 2008.

[56] Stephan ten Brink. Convergence begavior of iteratively decoded parallel concatenated codes. *IEEE Trans. Commun.*, 49(10):1727–1737, October 2001.

[57] Stephan ten Brink, Gerhard Kramer, and Alexei Ashikhmin. Design of low-density parity-check codes for modulation and detection. *Communications, IEEE Transactions on*, 52(4):670–678, 2004.

[58] Tomas Uricar, Tomas Hynek, Pavel Prochazka, and Jan Sykora. Wireless-aware network coding:solving a puzzle in acyclic multi-stage cloud networks. In *Accepted to ISWCS*, pages 1–5, September 2013.

[59] Tomas Uricar and Jan Sykora. Non-uniform 2-slot constellations for relaying in butterfly network with imperfect side information. *IEEE Commun. Lett.*, 2012.

[60] Nenad Veselinovic, Tad Matsumoto, and Markku Juntti. Iterative receivers for sttrc-coded mimo turbo equalization. In *Vehicular Technology Conference, 2004. VTC 2004-Spring. 2004 IEEE 59th*, volume 1, pages 460–463. IEEE, 2004.

[61] Branka Vucetic and Jinhong Yuan. *Turbo Codes: Principles and Applications*. Kluwer Academic Publishers, 2000.

[62] Niclas Wiberg. *Codes and Decoding on General Graphs*. Phd thesis, Linköping University, 1996.

[63] Andrew P. Worthen and Wayne E. Stark. Unified design of iterative receivers using factor graphs. *IEEE Trans. Inf. Theory*, 47(2):843–849, February 2001.

[64] D. Wübben and Y. Lang. Generalized sum-product algorithm for joint channel decoding and physical-layer network coding in two-way relay systems. In *IEEE 2010 Global Telecommunications Conference (Globecom 2010)*, Miami, FL, USA, Dec 2010.

[65] Sichao Yang and Ralf Koetter. Network coding over a noisy relay: a belief propagation approach. In *Proc. IEEE Internat. Symp. on Inf. Theory (ISIT)*, 2007.

[66] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51:2282–2312, 2005.

[67] Raymond W. Yeung, Shuo-Yen Robert Li, Ning Cai, and Zhen Zhang. *Network Coding Theory*. now Publishers, 2006.

[68] Yan Zhang, Hsiao-Hwa Chen, and Mohsen Guizani. *Cooperative Wireless Communications*. Auerbach Publications, Boston, MA, USA, 2009.