On the Design of Iterative Wireless Receivers: The Divergence Minimization Approach to Approximate Bayesian Inference

Von der Fakultät für Elektrotechnik und Informationstechnik
der Rheinisch–Westfälischen Technischen Hochschule Aachen
zur Erlangung des akademischen Grades
eines Doktors der Ingenieurwissenschaften
genehmigte Dissertation

vorgelegt von
Diplom–Ingenieur Martin Senst
aus Duisburg

Berichter: Universitätsprofessor Dr.-Ing. Gerd Ascheid
Universitätsprofessor Dr.-Ing. Peter Vary

Tag der mündlichen Prüfung: 16.12.2016

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Integrierte Systeme der Signalverarbeitung an der Rheinisch-Westfälischen Technischen Hochschule Aachen.

Mein besonderer Dank gilt Herrn Prof. Dr.-Ing. Gerd Ascheid für die Betreuung dieser Dissertation, sowie für die wertvollen Diskussionen und die Unterstützung meiner Arbeit während der gesamten Zeit meiner Institutszugehörigkeit.

Herrn Prof. Dr.-Ing. Peter Vary danke ich sehr herzlich für die Übernahme des Koreferats und das meiner Arbeit entgegengebrachte Interesse.

Ich hatte das große Glück einer fruchtbaren Zusammenarbeit mit Herrn Prof. Leszek Szczecinski vom Institut National de la Recherche Scientifique in Montreal, Kanada, im Rahmen des Network of Excellence in Wireless Communications. Vielen herzlichen Dank, lieber Leszek, für all die interessanten Gespräche und Deine Gastfreundschaft! Ohne Dich wäre diese Dissertation in der vorliegenden Form nicht entstanden.

Bei allen wissenschaftlichen und nichtwissenschaftlichen Mitarbeitern des Lehrstuhls möchte ich mich für die schöne Zeit und das hervorragende Arbeitsklima am Institut bedanken. Mein besonderer Dank gilt Elisabeth Böttcher, Susanne Hirschmann, Niels Hadaschik, Meik Dörpinghaus, Markus Jordan und Andreas Minwegen für die langjährige gute Zusammenarbeit.

Schließlich danke ich meinen Eltern Irene und Peter, die mir das Studium der Elektrotechnik ermöglicht und mich immer nach Kräften unterstützt haben, sowie meinem Bruder Andreas, der einen maßgeblichen Anteil an meiner Studienwahl hatte.

Dir, liebe Xiansu, danke ich von ganzem Herzen für Deine Geduld und Deine Nähe!

Martin Senst, im April 2017

Zusammenfassung

Die Entdeckung von Turbo-Codes in den 1990er Jahren hat den Entwurf von Kommunikationssystemen revolutioniert. Im Gegensatz zu herkömmlichen Kanalcodes zeichnen sie sich durch eine komplexe, pseudozufällige Struktur aus, die Datenraten nah an der Kanalkapazität ermöglicht. Die entscheidende Innovation von Turbo-Codes ist der neuartige Aufbau ihres Decoders: Er besteht aus zwei Teildecodern, die mehrfach abwechselnd aufgerufen werden und im Laufe dieses Prozesses Informationen über die gesendeten Daten austauschen. Es wurde empirisch beobachtet, dass dieses iterative Verfahren trotz fehlender theoretischer Garantien mit hoher Wahrscheinlichkeit gegen die korrekte Lösung konvergiert. Nach dem bemerkenswerten Erfolg von Turbo-Codes wurde das Konzept der iterativen Informationsverarbeitung auch auf andere Aspekte von drahtlosen Empfängern angewendet, wobei Techniken wie codegestützte Synchronisation sowie iterative Detektion und Decodierung entstanden sind.

Zu Beginn waren die vorgeschlagenen Algorithmen noch eher heuristisch, aber im Laufe der Zeit wurde viel Forschungsarbeit in die Entwicklung von theoretischen Grundlagen investiert. Ein wichtiger Schritt war die Herleitung des Turbo-Decoders als Instanz von *Belief Propagation* (BP), einer Methode zur Lösung von Bayes'schen Inferenzproblemen. Es stellte sich jedoch heraus, dass BP für andere Aufgaben jenseits des Kanaldecoders weniger gut geeignet ist. Beispielsweise führt die Verwendung von BP für die Herleitung von codegestützten Synchronisationsverfahren, die die Schätzung von kontinuierlichen Variablen erfordern, zu Integralen, welche typischerweise keine geschlossene Lösung besitzen. Für diese Klasse von Problemen hat sich der *Expectation-Maximization* Algorithmus (EM) als bessere Alternative herausgestellt, die jedoch ihre eigenen Nachteile besitzt. Des Weiteren führen hochdimensionale Detektionsprobleme, wie sie beispielsweise im Zusammenhang mit Mehrantennensystemen (MIMO) auftreten, zu Summen mit exponentiell vielen Termen, deren Berechnung praktisch unmöglich ist.

In dieser Arbeit untersuchen wir den Entwurf von iterativen drahtlosen Empfängern auf der Grundlage eines generischen Verfahrens zur näherungsweisen Lösung Bayes'scher Inferenzprobleme, das vor Kurzem im Bereich *Machine Learning* entwickelt wurde. Seine zentrale Idee ist die Umwandlung des ursprünglichen Summations- oder Integrationsproblems in ein äquivalentes Optimierungsproblem, welches anschließend

durch einen iterativen Algorithmus näherungsweise gelöst wird. Da es sich um die Minimierung eines Divergenzmaßes zwischen dem probabilistischen Systemmodell und einer einfacheren Hilfsverteilung handelt, bezeichnen wir diesen Ansatz als *Divergence Minimization* (DM). Aufgrund seiner Allgemeinheit bietet DM dem Designer viel Flexibilität, und es beinhaltet spezifische Algorithmen wie BP und EM als Sonderfälle.

Diese Arbeit beginnt mit einer systematischen Herleitung eines kombinierten EM- und BP-basierten Empfängers, der in der Literatur zuvor eher ad hoc beschrieben wurde. Anschließend nutzen wir die Flexibilität von DM zur Entwicklung von mehreren neuartigen Schätz- und Detektionsalgorithmen, die aufgrund ihrer guten Leistung und ihrer geringen Rechenkomplexität interessante Optionen für praktische Implementierungen sind. Darüber hinaus trägt diese Arbeit auch zum theoretischen Verständnis von iterativen Methoden bei. In früheren Arbeiten wurden die Unterkomponenten oft separat entworfen und dann heuristisch miteinander verbunden. Im Gegensatz dazu besteht der Vorteil des vorgeschlagenen Ansatzes zum Empfängerdesign darin, dass er nicht nur die einzelnen Funktionseinheiten, sondern auch ihre Verbindungen spezifiziert. Insbesondere herrscht in der Literatur Unklarheit darüber, ob die Komponenten extrinsische Information wie im Turbo-Decoder austauschen sollten, oder ob der Austausch der vollständigen *a posteriori* Information vorzuziehen ist. Die Herleitungen, die in dieser Arbeit vorgestellt werden, beleuchten diese wichtige Frage.

Abstract

The discovery of turbo codes in the 1990s has revolutionized the design of communication systems. Unlike traditional channel codes, they are characterized by a complex, pseudo-random structure which enables datarates close to the channel capacity. The key innovation of turbo codes has been their novel approach to decoding: it consists of two constituent decoders which run alternately several times and exchange information about the transmitted data during this process. It has been observed empirically that this iterative scheme converges with a high probability to the correct solution, despite the lack of any theoretical guarantees. Following the remarkable success of turbo codes, the concept of iterative information processing has also been applied to other tasks of wireless receivers, yielding techniques like code-aided synchronization and iterative detection and decoding.

Initially, the proposed algorithms were rather heuristic, but a significant research effort has been invested into the development of a theoretical foundation. An important step has been the derivation of the turbo decoder as an instance of belief propagation (BP), a framework for solving Bayesian inference problems. Unfortunately, it turns out that BP is less suitable for other tasks beyond the channel decoder. For example, using BP for the design of code-aided synchronization schemes, which require the estimation of continuous variables, gives rise to integrals that typically do not admit a closed-form solution. For this class of problems, the expectation-maximization (EM) algorithm has emerged as a better alternative, which however has its own drawbacks. Additionally, high-dimensional detection problems, which arise for example in the context of multi-antenna (MIMO) systems, involve sums with exponentially many terms whose evaluation is practically infeasible.

In this thesis, we investigate the design of iterative wireless receivers based on a generic framework for approximate Bayesian inference that has recently been developed in the machine learning community. Its key idea is the conversion of the original summation or integration problem into an equivalent optimization problem, which is then solved approximately by an iterative algorithm. As it consists of the minimization of a divergence measure between the probabilistic system model and a simpler auxiliary distribution, we refer to this approach as divergence minimization (DM). Due to its

generality, DM provides the designer with a great deal of flexibility, and it contains specific algorithms like BP and EM as special cases.

This thesis begins with a systematic derivation of a combined EM- and BP-based receiver, which has been proposed earlier in the literature in a rather ad-hoc way. We then utilize the flexibility of DM for the development of several novel parameter estimation and MIMO detection algorithms, which due to their good performance and low computational complexity are interesting options for practical implementations. Further, this thesis also contributes to the theoretical understanding of iterative methods. In earlier work, the subcomponents were often designed separately and then connected heuristically. In contrast, the virtue of the proposed holistic approach to receiver design is that it does not only specify the individual functional units but also their interactions. In particular, there has been some confusion in the literature on whether the components should exchange extrinsic information as in the turbo decoder, or whether exchanging the full posterior information is preferable. The derivations that are presented in this thesis shed light on this important question.

Contents

Abbreviations

3GPP	3rd Generation Partnership Project
ADF	Assumed Density Filtering
AR	Autoregressive
AWGN	Additive White Gaussian Noise
BCJR	Bahl-Cocke-Jelinek-Raviv
BEM	Bayesian EM
BER	Bit Error Rate
BICM	Bit-Interleaved Coded Modulation
BICM-ID	BICM with Iterative Detection and Decoding
BP	Belief Propagation
bpcu	Bit Per Channel Use
BPSK	Binary Phase Shift Keying
CA	Code-Aided
CF	Characteristic Function
CDF	Cumulative Distribution Function
CDMA	Code Division Multiple Access
CM	Coded Modulation
CRLB	Cramér-Rao Lower Bound
CSI	Channel State Information
DA	Data-Aided
DD	Decision-Directed
DM	Divergence Minimization
DVB	Digital Video Broadcasting

EM	Expectation Maximization
EP	Expectation Propagation
EXIT	Extrinsic Information Transfer
FFG	Forney-style Factor Graph
HPF	High-Pass Filter
IC	Interference Cancellation
IEEE	Institute of Electrical and Electronics Engineers
iid	independent and identically distributed
ISCD	Iterative Source-Channel Decoding
ISI	Inter-Symbol Interference
JSCD	Joint Source-Channel Decoding
KL	Kullback-Leibler
LDPC	Low-Density Parity-Check
LLR	Log-Likelihood Ratio
LMMSE	Linear MMSE
LOS	Line Of Sight
LPF	Low-Pass Filter
LTE	Long Term Evolution
LTV	Linear Time-Variant
MAP	Maximum A Posteriori
MCS	Modulation and Coding Scheme
MF	Mean Field
MIMO	Multiple-Input Multiple-Output
ML	Maximum Likelihood
MMSE	Minimum Mean-Squared Error
MRC	Maximum Ratio Combining
MSE	Mean-Squared Error
NCA	Non-Code-Aided
NDA	Non-Data-Aided
OFDM	Orthogonal Frequency Division Multiplexing
PDF	Probability Density Function
PHY	Physical Layer

PIC	Parallel Interference Cancellation
PLL	Phase-Locked Loop
PMF	Probability Mass Function
PSD	Power Spectral Density
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
QPSK	Quaternary Phase Shift Keying
RF	Radio Frequency
RMSE	Root-Mean-Squared Error
SAGE	Space Alternating Generalized EM
SDD	Soft-Decision-Directed
SIC	Successive Interference Cancellation
SINR	Signal-to-Interference-plus-Noise Ratio
SIR	Signal-to-Interference Ratio
SISO	Single-Input Single-Output
SM	Spatial Multiplexing
SNR	Signal-to-Noise Ratio
SVD	Singular Value Decomposition
TCM	Trellis-Coded Modulation
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
VCO	Voltage-Controlled Oscillator
VLSI	Very Large-Scale Integration
VMP	Variational Message Passing
WER	Word Error Rate
WLAN	Wireless Local Area Network
WLMMSE	Widely Linear MMSE
XTL	Reference Crystal in PLL
ZF	Zero Forcing

Notation

The mathematical notation in this thesis adheres to the conventions that are established in the communications literature. Lower- and uppercase italics denote scalar variables and constants, respectively, while lower- and uppercase boldface letters denote vectors and matrices. Augmented quantities (see page 205) are underlined, and continuous-time signals are written in a sans-serif font.

The common probability distributions are written in a calligraphic font. We distinguish between a distribution and its density function in the following way: $\mathcal{N}(\mu, \sigma^2)$, for example, denotes the Gaussian distribution with mean μ and variance σ^2, whereas $\mathcal{N}(x \,|\, \mu, \sigma^2) \triangleq \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)$ is shorthand for its PDF. Thus, we would write $x \sim \mathcal{N}(\mu, \sigma^2)$, but $p(x) = \mathcal{N}(x \,|\, \mu, \sigma^2)$. Finally, $\mathcal{N}$ without arguments represents the set of all Gaussian distributions.

All indices start at zero rather than one. All norms, written $\|\cdot\|$, are 2-norms, and the inner products of complex quantities are conjugate linear in the second argument.

$\boldsymbol{x}^{\mathsf{H}}$	Conjugate transpose of a complex vector or matrix
$\boldsymbol{x}^{\mathsf{T}}$	Transpose of a vector or matrix
$\boldsymbol{x}_{\setminus n}$	Vector without the nth element
$\boldsymbol{X}_{\setminus n}$	Matrix without the nth column
$\langle \boldsymbol{x}, \boldsymbol{y} \rangle$	Inner vector product: $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \triangleq \boldsymbol{y}^{\mathsf{H}} \boldsymbol{x}$
$\underline{\boldsymbol{x}}$	Augmented vector: $\underline{\boldsymbol{x}} \triangleq (\boldsymbol{x}^{\mathsf{T}} \;\; \boldsymbol{x}^{\mathsf{H}})^{\mathsf{T}}$
$\boldsymbol{0}_N$	All-zero vector of length N
$\mathbb{1}\,[\mathcal{A}]$	Indicator function; returns 1 if $\mathcal{A}$ is true, and 0 otherwise
$A(\boldsymbol{\theta})$	Log-partition function of an exponential family
$\boldsymbol{A}$	State-transition matrix in state space models
$\alpha(t)$	Stochastic time shift of noisy oscillator
$\mathcal{B}(p)$	Bernoulli distribution on $\mathbb{F}$, with $\mathbb{P}\,[b = 1] = p$
$\boldsymbol{b}$	Transmitted message, a sequence of N_{b} bits

$\mathbb{C}$	Set of complex numbers
$\mathcal{C}$	Codebook, the set of legal codewords
$\mathcal{CN}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Proper Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\boldsymbol{c}$	Codeword, a sequence of N_c bits
$\boldsymbol{c}^\mathsf{T}$	Output vector in state space models
$D_\mathrm{KL}(p\|q)$	Kullback-Leibler divergence
$\mathcal{D}$	Time instants used for transmission of payload data symbols
$\delta(x)$	Dirac distribution, assigns all probability mass to $x = 0$
E_s	Received energy per symbol duration and Rx antenna
$\mathbb{E}_{p(x)}[f(x)]$	Expectation with respect to the distribution $p(x)$: $\mathbb{E}_{p(x)}[f(x)] = \int p(x) f(x) \,\mathrm{d}x^\dagger$
$\mathcal{EN}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \tilde{\boldsymbol{\Sigma}})$	Improper Gaussian dist. with mean $\boldsymbol{\mu}$, cov. $\boldsymbol{\Sigma}$ and complementary cov. $\tilde{\boldsymbol{\Sigma}}$
ε	Normalized timing offset
$\mathbb{F}$	Galois field of order 2
$\mathcal{F}$	Set of fully factorized distributions
f	Frequency
f_c	Carrier frequency
f	Anti-aliasing filter
$\mathcal{G}(\alpha, \beta)$	Gamma distribution
g	Real and positive channel gain
g	Transmit pulse
$\boldsymbol{\Gamma}$	Inverse covariance matrix of Gaussian distribution: $\boldsymbol{\Gamma} = \boldsymbol{\Sigma}^{-1}$
γ	Precision (inverse variance) of the AWGN
h	Complex channel gain: $h = g\,\mathrm{e}^{\mathrm{j}\vartheta}$
$\boldsymbol{H}_n$	MIMO channel matrix at time n
$\boldsymbol{I}_N$	Identity matrix of size $N \times N$
j	Imaginary unit: $\mathrm{j}^2 = -1$
K	Number of code bits per complex symbol: $K = \log_2 M$
K	In Appendix B: Number of factors in factorization of p
K_x	Autocovariance function of stochastic process x
$\mathcal{L}_1$	Integrable functions from $\mathbb{R}$ to $\mathbb{C}$
$\mathcal{L}_2$	Square-integrable (energy-limited) functions from $\mathbb{R}$ to $\mathbb{C}$
$\mathsf{L}(\Delta f)$	Single-sided PSD of phase noise process, where $\Delta f = f - f_\mathrm{c}$
$\ell_y(x)$	Ratio of posterior to prior probability of x: $\ell_y(x) = \frac{p_{x\|y}(x\,\|\,y)}{p_x(x)}$
$\vec{\boldsymbol{\lambda}}$	Intrinsic L-values, passed from demapper to decoder

$\overleftarrow{\boldsymbol{\lambda}}$	Extrinsic L-values, passed from decoder to demapper
$\boldsymbol{\lambda}$	Approximation of the posterior L-values: $\boldsymbol{\lambda} = \overrightarrow{\boldsymbol{\lambda}} + \overleftarrow{\boldsymbol{\lambda}}$
M	Size of the modulation alphabet: $M = \lvert\mathcal{X}\rvert$
$\mathcal{M}$	Mapper; converts blocks of code bits into data symbols
$\mathcal{M}(\zeta)$	Von Mises distribution
$\boldsymbol{\mu}$	Mean vector of Gaussian distribution
N	Total number of channel uses per data packet: $N = N_\mathrm{d} + N_\mathrm{p}$
N	In Appendix B: Number of unobserved variables in p
N_0	Power spectral density of the AWGN process at each Rx antenna
N_b	Number of information bits per codeword
N_c	Number of code bits per codeword
N_d	Number of payload data symbols per data packet
N_p	Number of pilot symbols per data packet
N_r	Number of receive antennas
N_s	Number of spatially multiplexed data streams
N_t	Number of transmit antennas
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Real Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
n	Discrete time, normalized to symbol duration: $n = t/T$
$\boldsymbol{\nu}$	Scaled mean vector of Gaussian distribution: $\boldsymbol{\nu} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$
ν	Doppler shift
$\mathbb{P}[A]$	Probability of the event A
$\mathcal{P}$	Time instants used for transmission of pilot symbols
$\mathcal{P}_x$	Set of all probability distributions for a random variable x
$p_x(x)$	Probability mass or density function of x[†]
$p_{x\lvert y}(x \mid y)$	Probability mass or density function of x, conditioned on y[†]
$\boldsymbol{\phi}(x)$	Sufficient statistics of an exponential family
$\varphi_x(\omega)$	Characteristic function of random variable x
$\mathcal{Q}_x$	Set of allowed, sufficiently simple distributions for x; $q_x \in \mathcal{Q}_x$
$q_x(x)$	Belief of x; approximation of the posterior $p(x \mid y)$ for some observation y
R_s	Sampling rate
R_x	Self-similarity function of the deterministic signal x
$\mathbb{R}$	Set of real numbers
r	Code rate: $r = N_\mathrm{b}/N_\mathrm{c}$
$\tilde{\mathsf{r}}$	Received signal without noise
r	Received signal: $\mathsf{r} = \tilde{\mathsf{r}} + \mathsf{v}$

r_f	Received signal after anti-aliasing filter
$\boldsymbol{r}$	Received signal after A/D conversion
ρ_n	Estimate of transmitted energy at time n: $\rho_n = \mathbb{E}\left[\lvert x_n\rvert^2\right]$
S_x	Power spectral density of the stochastic process x
$\boldsymbol{s}_n$	State vector in state space models
s	Transmitted signal
$\boldsymbol{\Sigma}$	Covariance matrix of Gaussian distribution
T	Symbol period
T_s	Sample period
$\mathcal{TN}(m,s)$	Truncated Gaussian distribution, which assigns zero probability to $x < 0$
tr	Trace of a square matrix
t	Time
τ	Delay
$\boldsymbol{\theta}$	Collection of all synchronization parameters
$\boldsymbol{\theta}$	Natural parameters of an exponential family
ϑ	Phase offset
$\mathcal{U}(a,b)$	Uniform distribution over the interval (a,b)
$\boldsymbol{u}_n$	Process noise in state space models
v	Additive white Gaussian noise process
$\boldsymbol{v}_i$	Interference-plus-noise experienced by the ith data stream
$\boldsymbol{w}_n$	White Gaussian noise samples at time n
$\mathcal{X}$	Modulation alphabet, e. g. QAM or PSK
$\boldsymbol{x}_n$	Complex data symbols transmitted at time n
$\boldsymbol{y}_n$	Received signal after A/D conversion and matched filtering: $\boldsymbol{y}_n = \boldsymbol{H}_n\boldsymbol{x}_n + \boldsymbol{w}_n$
Z	A generic normalization constant
$\mathbb{Z}$	Set of integers
$\vec{\zeta}$	Extrinsic phase information, passed from phase estimator to demapper
$\overleftarrow{\zeta}$	Intrinsic phase information, passed from soft mapper to phase estimator
ζ	Soft phase estimates: $\zeta = \vec{\zeta} + \overleftarrow{\zeta}$

† As is common practice, we drop the subscript if the notation remains unambiguous.

Chapter 1

Introduction

The ultimate task of transmitters in wireless digital communication systems is to convert a message, which could be an analog waveform or some data that is already available in digital form, into an RF signal which is suitable for transmission over a radio channel. This process can typically be broken down into functions like source coding, channel coding, mapping to modulation symbols, and D/A conversion, which are executed one after the other. Conventional receiver designs used to mirror this sequential transmission structure: they processed the received signal step by step, essentially inverting the functional units of the transmitter in reverse order.

In the 1990s, information theory was revolutionized by the invention of turbo codes and the subsequent rediscovery of LDPC codes. For the first time, these so-called "modern channel codes" allowed for digital communication over noisy channels with data rates close to the channel capacity, which constitutes a fundamental upper limit. The essential characteristic of these codes is a complex, pseudo-random internal structure, which makes them similar to the capacity-achieving (but impractical) random codes that have been used by Shannon in the proof of his celebrated noisy-channel coding theorem.

The downside of this structural complexity, however, is that it renders optimum decoding (in the sense of minimal error probability) virtually impossible. The receiver must therefore resort to heuristic decoding algorithms. In practice, these usually consist of several interconnected subdecoders which are invoked alternately, exchanging information about the transmitted message during this process. Despite the lack of theoretical performance guarantees, experience has shown that such iterative schemes can yield surprisingly good results. As long as the channel quality exceeds a certain information-theoretic threshold, even if just by a small margin, there is a high probability that the algorithm will converge to the correct solution.

The concept of iterative information processing caused nothing short of a paradigm shift in receiver design. Soon after the development of modern channel codes, the idea

was also applied to other functional units of the receiver. Two important families of iterative receiver algorithms, which we will study in greater detail within this work, are

- *code-aided synchronization:* In coherent communication systems, the receiver must be synchronized to the symbol timing of the transmitted signal, and it must compensate for any carrier phase and frequency offsets. This is conventionally achieved with the aid of some training data that is known *a priori* to the receiver, for example in the form of a preamble, or as pilot symbols that are multiplexed into the data stream. However, training data has the obvious drawback that it decreases the net data rate, since it occupies resources which are not available anymore for the transmission of payload data. Code-aided synchronization attempts to reduce the required amount of training data to a bare minimum, which is necessary to bootstrap the algorithm. The intuitive idea is that after the receiver has processed a block of data once, the detected message (which may still contain some bit errors) can be re-modulated to data symbols, which are then passed back to the synchronization unit where they are treated as "virtual" pilot symbols. Now, the receiver processes the same block of data a second time. Due to the increased amount of available training data, the synchronization parameters can be re-estimated with higher accuracy, which will ultimately lead to a lower bit error rate after decoding. If required, this procedure can be iterated until convergence.

- *iterative detection and decoding:* This technique can be used in systems which combine a binary channel code with a non-binary modulation scheme. If K code bits are transmitted per channel use, there are $M = 2^K$ possible data symbols which the demapper must choose from. However, if the decoder is able to provide the demapper with prior information about K' bits after it has processed the data once, only the $2^{K-K'}$ hypotheses that are compatible with the reliably decoded bits remain. This may assist the demapper in making a better choice about the remaining $K - K'$ bits, which will in turn improve the decoding result. Again, this process can be iterated several times.

As these examples demonstrate, the idea behind iterative receivers is fairly intuitive, and it is easy to devise heuristic algorithms in accordance with the general concept. But inventing algorithms in an *ad hoc* way is unsatisfying, even if they appear to work well. In order to realize the potential benefits of iterative techniques to their full extent, developing a solid theoretical understanding is indispensable.

An important step towards this goal has been the insight that the turbo decoder is an instance of *belief propagation* (BP), also known as the *sum-product algorithm*, which was developed in the machine learning community in the 1980s. It is a generic method for solving Bayesian inference problems (i. e., updating *a priori* to *a posteriori* probability distributions in the light of new data) which typically involve the computation of sums with many terms. BP reduces the required number of operations by exploiting the

distributive law, which is applicable whenever two random variables are conditionally independent (i. e., when there is no immediate interaction between them). A central concept of BP is the so-called *belief network*, a graphical representation of the statistical dependencies between all variables, which is usually rather sparse in practice. If the underlying belief network is free of cycles, BP terminates in finite time and gives exact answers. Unfortunately, modern pseudo-random channel codes do not admit a simple tree-structured representation. Strictly speaking, BP is thus inapplicable to the problem of decoding random-like codes, but it has been recognized that the turbo decoder can be derived by simply ignoring the loops and applying BP nonetheless. This is possible because the algorithm only consists of local computations that are unaware of the global graph structure. However, the presence of loops has the consequence that the exactness guarantees of plain BP get lost. Instead, the local computations are iterated several times, in the hope that the preliminary results will converge towards the correct solution.

While BP has emerged as the standard algorithmic framework for the derivation of modern channel decoders, it is insufficient for other receiver tasks like detection and synchronization, which, as motivated above, can also benefit from iterative techniques. There are no theoretical obstacles which prevent the application of BP to these tasks, but some practical ones. In the context of detection, the number of hypotheses is often so large that the resulting sums cannot be evaluated in real-time anymore. This problem arises for example in multi-antenna systems, where the number of signals which need to be distinguished at the receiver grows exponentially with the number of transmitted data streams. And when BP is applied to the estimation of continuous channel parameters, those sums turn into integrals, which usually do not admit a closed-form solution.

Several authors have proposed the *expectation maximization* (EM) algorithm as an alternative theoretical framework for code-aided synchronization. EM is an iterative method for solving maximum likelihood estimation problems in the presence of unobserved "latent" variables, which in the case of code-aided synchronization are the unknown data symbols. The EM algorithm has been successfully used to generalize conventional parameter estimators, which solely rely on training data, to code-aided algorithms. But while EM indeed avoids some issues of BP, it is not a panacea either. One of its problems is its origin in orthodox statistics, which leads to two undesirable consequences. On the one hand, it treats the unknown parameters as deterministic. The algorithm is therefore unable to exploit prior knowledge that might be available from earlier observations or because of physical constraints. And on the other hand, the output of EM is a "hard" estimate, a single and almost surely incorrect value. In contrast, modern Bayesian methods use the observed data to convert prior into posterior distributions; i. e., they consume and produce "soft" probabilistic information. Another issue is that EM is only used for the derivation of synchronization and estimation algorithms, while the detector and decoder remain instances of belief propagation. Connecting the two parts is possible, but only heuristically, and the resulting estimators are, strictly speaking, not EM algorithms anymore.

The preceding discussion motivates our interest in a unifying algorithmic framework for statistical inference, which is sufficiently expressive to allow for a joint derivation of all functional units of a receiver. The method needs to be practically applicable to mixed belief networks, which contain both discrete and continuous variables, and it should contain BP and EM as special cases. It should be fully Bayesian, in that it treats all unobserved quantities as random variables, whose distributions encode the available incomplete probabilistic knowledge about their state. And in order to keep the computational complexity manageable, it should be possible to introduce approximations in a controlled way. This requirement, however, gives rise to an interesting problem: statistical inference fundamentally reduces to solving sums and integrals, and it is far from obvious how these kinds of computations can be systematically approximated.

An interesting approach to approximate statistical inference, which fulfills these conditions, has been developed in the machine learning community in 2005. Its key idea is to replace the complex posterior with an auxiliary distribution that is simple enough to allow for exact inference. Now, the problem consists of finding an auxiliary distribution which, informally, resembles the true posterior as closely as possible. In order to formalize this task, a *divergence* measure that quantifies the dissimilarity between two probability distributions is introduced. The original summation or integration problem then turns into an optimization problem: finding the particular element of a given set of simple distributions which minimizes the divergence with respect to the true posterior. We will therefore refer to this approach as *divergence minimization* (DM) in the following.

It can be shown that, for a particular choice of divergence measure and feasible set, the new optimization problem is equivalent to the original inference task, in the sense that its solution allows us to recover the marginal posterior distributions that we are interested in, and whose naïve computation gave rise to the infeasible sums and integrals in the first place. Unfortunately, finding the exact solution of the optimization problem is also infeasible in general, so this equivalence might seem like a moot point. However, the optimization perspective turns out to be valuable nonetheless, because it enables us to introduce approximations in a more systematic way than the original formulation would have allowed. First of all, the global optimization problem can be broken down into interconnected local subproblems, whose preliminary solutions are alternately updated by means of a fixed-point iteration scheme. And specific subproblems can furthermore be simplified by modifying their objective function and/or the feasible set.

In this thesis, we will study the applicability of the divergence minimization approach to the systematic design of iterative receiver structures. The aim of this work is twofold. On the one hand, it contributes to the theoretical understanding of iterative methods, particularly the interaction between the individual algorithmic components. And on the other hand, it develops several novel estimation and detection algorithms, which, due to their good algorithmic performance and low computational complexity, are interesting options for practical receiver designs.

Outline

Chapter 2 introduces the *bit-interleaved coded modulation* (BICM) transmission technique, which is the *de facto* standard for modern wireless communication systems. In particular, we derive the BICM receiver with iterative detection and decoding, which will serve as the foundation for the following discussions.

The next two chapters present the main contributions of this thesis. They are organized according to the classical distinction of statistical signal processing between *estimation* and *detection* problems. We begin in Chapter 3 with a discussion of code-aided synchronization and parameter estimation. After a brief overview of state-of-the-art techniques, we introduce the concept of a likelihood function for continuous-time observations, which is required for the following development. Subsequently, we use the divergence minimization approach to derive a receiver with hard parameter estimation, which has earlier been proposed as an instance of the EM algorithm. As mentioned above, the DM framework is inherently Bayesian, but hard estimators can be obtained as special cases by constraining the feasible sets of the auxiliary parameter distributions to Dirac deltas. The resulting algorithms themselves are not novel, but the proposed derivation has the advantage that it admits a natural generalization to soft estimators, simply by relaxing this constraint. This aspect will be illustrated in the remainder of this chapter with case studies on soft estimation of both static and time-varying parameters.

Chapter 4 discusses the application of divergence minimization to detection problems. As a running example, we study the detection of spatially superimposed signals in multi-antenna (MIMO) systems. The same fundamental problem, however, arises in many areas of wireless communication, for example in turbo equalization for frequency-selective channels and multi-user detection in CDMA systems. We will see that the DM approach naturally leads to iterative detection algorithms, where soft information is exchanged between an affine front-end (consisting of interference cancellation followed by a linear filter), and a bank of symbolwise detectors. Their novelty with respect to conventional linear detectors consists in the direct feedback path from the symbolwise detectors to the affine front-end, which improves the algorithmic performance because it enables the detector to exploit its knowledge about the discrete modulation alphabet.

Chapter 5 concludes this thesis with a summary and suggestions for future work.

Appendix A provides some mathematical background. Section A.1 introduces basic concepts from complex and multivariate differential calculus, which are required in the subsequent Section A.2 on exponential families. Besides explaining the general notion of exponential families and their importance for modern inference algorithms, we also introduce the particular families that are used within this thesis.

Finally, Appendix B gives an overview of approximate statistical inference methods, and provides a detailed derivation of the DM framework which is used throughout this work. The reader who is not familiar with the general concepts of divergence minimization is encouraged to scan this appendix before proceeding to the main text.

Chapter 2

Bit-Interleaved Coded Modulation with Iterative Detection and Decoding

The major contemporary wireless communication systems, like 3GPP LTE and current revisions of the IEEE 802.11 WLAN standard, are based on *bit-interleaved coded modulation* (BICM). Due to the overwhelming practical relevance of this transmission scheme, the present thesis focuses exclusively on receiver structures for BICM systems.

This chapter starts with a general overview of modern communication systems in Section 2.1, and a brief introduction to BICM transmission in Section 2.2. The main part of this chapter is contained in Section 2.3, where we discuss the receiver side of BICM systems. In particular, we derive the standard BICM-ID algorithm, which exchanges soft information about the transmitted bits iteratively between the two constituent algorithmic components: the demapper and the soft-input soft-output channel decoder.

The common derivation of the BICM-ID algorithm, reproduced in Section 2.3.1, is based on an inference technique called *belief propagation* (BP). Subsequently, Section 2.3.2 presents an alternative derivation based on *expectation propagation* (EP), which is a generalization of BP. In the particular example of BICM-ID, the additional degrees of freedom provided by EP are not utilized, and both techniques yield the same algorithm. However, EP will play a central role for some later developments in this thesis, and it is instructive to introduce it in the context of a simple and well-known example.

All derivations in this chapter are based on the assumption that the receiver is perfectly synchronized with the transmitter. This results in a simple discrete-time channel model, where all parameters like phase shift, channel gain, and noise variance are known to the receiver. In the following chapter, we will lift these simplifying assumptions, and study receiver structures where the estimation of the required synchronization parameters is entangled with the iterative detection and decoding process.

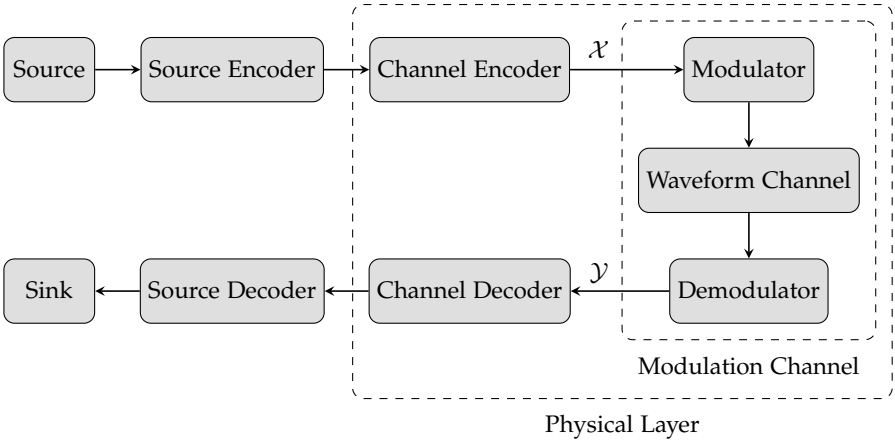


Figure 2.1: General block diagram of a digital transmission system

2.1 A Brief History of Coding and Modulation

Figure 2.1 shows an overview of the general structure of digital communication systems, which has been developed by Shannon in 1948 [123], but is still valid for todays wireless standards. The *source* emits some information, for example an analog signal as in audio or video applications, or any kind of digital data. The information is processed by a *source encoder*, which (after A/D conversion in the case of analog sources) removes the redundancy contained in the source signal, and outputs a sequence of binary digits.

This bit sequence is then passed through several layers of protocols, which are not shown in the figure as they are not relevant for our discussion. In this thesis, we will only be concerned with the lowest level of the protocol stack, the *physical* (PHY) layer. It takes a block of bits (called *message* in the following) from the higher layers, and converts it into an analog waveform that is suitable for transmission over the physical medium. This process can be broken down into two steps: first, the *channel encoder* maps the message to a discrete sequence of (binary or non-binary) symbols, and the *modulator* subsequently generates the analog signal.

In the early days of digital communications, coding[1] and modulation were treated as separate entities that were designed independently. The modulator and a corresponding hard-quantizing demodulator were used to convert the underlying continuous waveform channel into a discrete *modulation channel* with coinciding input and output alphabets $\mathcal{X}$ and $\mathcal{Y}$ (for example, $\mathcal{X} = \mathcal{Y} = \{0, 1\}$). The task of the channel code was then to correct the errors that were introduced by the modulation channel.

[1] In the following, the terms "coding", "encoder", and "decoder" always refer to *channel* coding.

A paradigm shift occurred in the 1970s, when Massey suggested that coding and modulation should rather be considered as a single entity [84]. In particular, he studied the gains that could be achieved by *soft* demodulation, i. e., by allowing for a *continuous* channel output $\mathcal{Y}$, which avoids the information loss caused by hard-output demodulation. An immediate consequence of $\mathcal{X} \neq \mathcal{Y}$ is that the notion of an "error probability" of the modulation channel (e. g., an *uncoded* bit error rate) becomes meaningless.

In 1976, shortly after Massey's paper, Ungerböck presented the first practical coding scheme which operates directly on non-binary signal levels (and not on code symbols), the so-called *trellis-coded modulation* (TCM) [135]. However, it was not until 1982 that TCM became widely recognized with the publication of [132]. The essential feature of TCM, which distinguishes it from traditional coding schemes, is that it maximizes the minimum Euclidean distance between signal sequences, rather than the minimum Hamming distance between codewords. In the following years, TCM found widespread application in bandwidth-limited systems, for example in voiceband modems. An accessible overview of TCM can be found in the tutorial papers [133, 134].

A major drawback of TCM is its lack of flexibility: for each modulation alphabet, a separate, well-performing TCM scheme needs to be found. In an attempt to overcome this problem, Viterbi *et al* presented "a pragmatic approach to trellis-coded modulation" in 1989 [139], which allowed to combine industry standard convolutional codes with several M-ary *phase shift keying* (PSK) modulations. The idea was further developed by Zehavi [156], who combined a convolutional code and an M-ary signal constellation via a bank of bit interleavers. At the receiver side, he proposed to compute bitwise soft metrics, which were processed by a soft-input Viterbi decoder. This transceiver concept, which allows combinations of any binary code with any modulation scheme, became later known as *bit-interleaved coded modulation* [19]. Besides its flexibility, BICM also benefits from a larger diversity compared to symbolwise interleaving, which makes it particularly interesting for wireless systems that operate over fading channels.

In the 1990s, coding theory was revolutionized by the invention of *turbo codes* [11, 12] and the subsequent re-discovery of *low-density parity-check* (LDPC) codes [81], which had originally been developed by Gallager in the 1960s [43], but whose potential had not been recognized at that time. Both code families have in common that they are much more "random-like" than traditional codes (for example the highly structured convolutional codes). The downside of this randomness is that it renders optimum *maximum a posteriori* (MAP) decoding infeasible. Nonetheless, it was shown that even with suboptimal *iterative* decoders, these "modern codes" [110] allowed communication over binary-input channels with a vanishingly small gap to the capacity. The roughly simultaneously developed BICM was a simple and pragmatic method to achieve this unprecedented performance also in conjunction with non-binary modulation schemes.

As mentioned above, the demodulator and decoder of a BICM receiver communicate via bitwise soft metrics. BICM can therefore suffer from a performance degradation compared to more general coded modulation (CM) schemes, because bitwise metrics

are unable to capture the statistical dependencies between bits that belong to the same modulation symbol. However, it was demonstrated that the resulting performance loss could be significantly reduced by *iterative detection and decoding* [75,76], a concept that is a direct generalization of the iterative turbo and LDPC decoding algorithms, and that is commonly referred to as BICM-ID. A detailed analysis of BICM, including iterative decoding, is presented in [51].

Today, BICM with modern channel codes is the *de facto* standard in state-of-the-art communication systems. Important examples include the IEEE WLAN standard 802.11n, the cellular *long term evolution* (LTE) system, and *digital video broadcasting* (DVB) services like the terrestrial DVB-T2 and the satellite system DVB-S2. As already mentioned at the beginning of this chapter, it is this outstanding practical relevance which motivates our focus on receiver structures for BICM systems in this thesis.

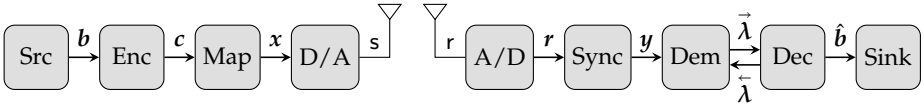


Figure 2.2: Physical layer of a generic BICM transmission system

2.2 A Generic BICM Transmitter

Figure 2.2 provides an overview of a generic BICM-based physical layer. From now on, the *source* does not refer to the original information source as in Figure 2.1 anymore; instead, it should be understood from the perspective of the PHY-layer.

The source emits a *message* in the form of a bit sequence of length N_b:

$$\boldsymbol{b} = (b_0, b_1, \ldots, b_{N_\mathrm{b}-1}). \tag{2.1}$$

We assume that the bits b_i are statistically independent and uniformly distributed, or equivalently, that all 2^{N_b} messages are equally probable:

$$p(\boldsymbol{b}) = 2^{-N_\mathrm{b}}, \qquad \boldsymbol{b} \in \mathbb{F}^{N_\mathrm{b}}. \tag{2.2}$$

For two reasons, this assumption will only hold approximately. On the one hand, practical source encoders are suboptimal (because of latency constraints, for example), so that there will be some remaining redundancy in the message. Furthermore, $\boldsymbol{b}$ also contains packet headers from the higher protocol layers, which exhibit some structure and hence are not uniformly distributed. The redundancy in the message can be exploited by *joint source-channel decoding* (JSCD) techniques [34], but in this thesis, our focus will be limited to the physical layer. Thus, any techniques beyond the channel decoder are out of scope for us. However, JSCD can of course be combined with the PHY-layer that we will discuss in the following—particularly the *iterative source-channel decoding* (ISCD) approaches from [2,52], which belong to the same algorithmic framework as the BICM-ID receiver that will be derived in the next section.

The *encoder* converts the message into a *codeword* of length N_c:

$$\boldsymbol{c} = (c_0, c_1, \ldots, c_{N_\mathrm{c}-1}). \tag{2.3}$$

We will use the terms *information bits* and *code bits* to distinguish between the b_i and c_i. The *rate* of the code is defined as $r = N_\mathrm{b}/N_\mathrm{c}$.

The careful reader will have noticed the lack of an interleaver after the encoder in Figure 2.2. Despite its prominent appearance in the name "BICM", the bit interleaver is (at least for our purposes) just an implementation detail. In the following, the encoder should be understood as an *effective* encoder; a black box which may comprise several subblocks that operate on bit strings. A possible effective encoder is shown

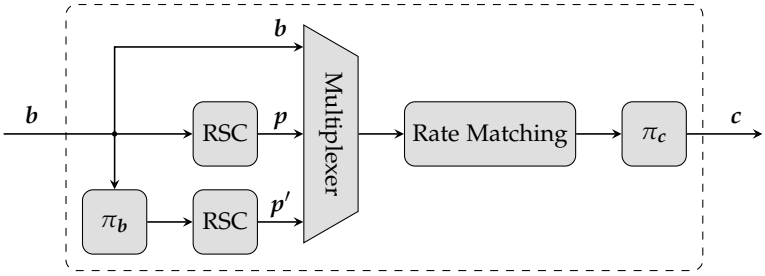


Figure 2.3: Exemplary effective code, consisting of parallelly concatenated RSC encoders, interleavers, and a rate-matching device

in Figure 2.3, built from a rate-1/3 turbo code (which itself consists of two *recursive systematic convolutional* (RSC) encoders, separated by a bit interleaver $\pi_{\boldsymbol{b}}$), a rate-matching device (which could e. g. puncture or repeat some bits), and the BICM interleaver $\pi_{\boldsymbol{c}}$.

The codeword is split into N segments with a length of K bits each (i. e., $N_\mathrm{c} = KN$)

$$\boldsymbol{c} = (\boldsymbol{c}_0, \boldsymbol{c}_1, \ldots, \boldsymbol{c}_{N-1}) \tag{2.4}$$

and the subsequences are mapped to complex data symbols $x \in \mathcal{X}$, where $\mathcal{X}$ is an M-ary modulation alphabet and $M = 2^K$. In the numerical examples that will be presented in later chapters, we will always use *quadrature amplitude modulation* (QAM) schemes like 16-QAM, but the discussed algorithms are of course not restricted to QAM alphabets.

The symbol sequence

$$\boldsymbol{x} = (x_0, x_1, \ldots, x_{N-1}) \tag{2.5}$$

is finally converted to a continuous-time signal s, using linear modulation with a band-limited transmit pulse g and symbol spacing T:

$$\mathsf{s}(t) = \sum_{n=0}^{N-1} x_n \, \mathsf{g}(t - nT). \tag{2.6}$$

Note the convention of using sans-serif letters for continuous-time signals.

We will only consider single-carrier transmissions over frequency-flat channels in the following. This assumption might be rather simplistic, since modern terrestrial systems typically operate over frequency-selective fading channels. However, the basic principles that we will discuss are still applicable. The *de facto* standard for current broadband systems is the multi-carrier technique *orthogonal frequency division multiplexing* (OFDM), where the available bandwidth is split into sufficiently narrow subbands, so that each subcarrier experiences flat fading. Some algorithms carry over unchanged to OFDM systems, for example the MIMO detectors, which then work on a per-subcarrier basis. Others, like synchronization, might need some adaptations, but thanks to the strong modularity of graph-based inference algorithms, these should be fairly straightforward.

2.3 The Outer BICM-ID Receiver

The literature on receiver algorithms sometimes distinguishes between an *inner* and *outer* receiver, but the precise meaning of these terms does not appear to be universally agreed upon. In this thesis, we will draw the line roughly at the boundary between continuous and discrete variables:

- The inner receiver comprises tasks such as *synchronization* and *channel estimation*. It essentially converts the underlying continuous-time into a discrete-time channel.
- The outer receiver consists of the *demapper* (also called *detector*) and the *decoder*. This is the realm of BICM-ID.

In the following, we will briefly derive the standard BICM-ID outer receiver under the assumption of perfect synchronization. The next chapter will then provide a detailed discussion of the inner receiver, and of the connection between inner and outer receiver.

The emphasis of this derivation is on the general concept of message passing algorithms and on the implementation of the soft demapper, while the channel decoder will only be sketched very briefly. The decoder plays of course a central role in the iterative code-aided receiver techniques that this thesis is all about, but it will be sufficient for us to treat it as a black box, which inputs intrinsic L-values of the code bits and returns extrinsic L-values and an estimate $\hat{\boldsymbol{b}}$ of the message. There are many detailed tutorials on soft-in soft-out decoders for turbo and LDPC codes available in the literature, so it will not be necessary to cover this topic here in depth.

As an example, the following derivation assumes a specific modulation and coding scheme (MCS): a turbo code and quadrature phase shift keying (QPSK) modulation. The encoder is structured as shown in Figure 2.3. The message $\boldsymbol{b}$ (whose components will be called *systematic bits*) is encoded by two RSC codes, once in its original shape, and once after a pseudo-random bit interleaver $\pi_{\boldsymbol{b}}$. The output sequences of the two encoders (the *parity bits*) are called $\boldsymbol{p}$ and $\boldsymbol{p}'$. With ρ denoting the rate matching operation and $\pi_{\boldsymbol{c}}$ the BICM interleaver, the final codeword is obtained as

$$\boldsymbol{c} = (c_0, c_1, \ldots, c_{N_c-1}) = \pi_{\boldsymbol{c}}(\rho(\boldsymbol{b}, \boldsymbol{p}, \boldsymbol{p}')). \tag{2.7}$$

The code bits are grouped pairwise

$$\boldsymbol{c} = ((c_{0,0}, c_{0,1}), (c_{1,0}, c_{1,1}), \ldots, (c_{N-1,0}, c_{N-1,1})) \tag{2.8}$$

and for each n, the pair $(c_{n,0}, c_{n,1})$ is mapped to a QPSK data symbol $x_n = \mathcal{M}(c_{n,0}, c_{n,1})$.

Under the assumption of an ideal inner receiver, the outer receiver sees a memoryless discrete-input continuous-output channel that can be described via

$$y_n = g_n x_n + w_n, \qquad w_n \sim \mathcal{CN}(0, N_0) \tag{2.9}$$

where $\boldsymbol{w} = (w_0, \ldots, w_{N-1})$ is additive white Gaussian noise (AWGN). The possibly time-varying channel gain $g_n > 0$ and the noise variance N_0 are estimated by the inner receiver, and are assumed to be known in the remainder of this chapter.

With the channel model (2.9), the average signal-to-noise ratio (SNR) is

$$\mathsf{SNR} \triangleq \frac{\mathbb{E}\left[|gx|^2\right]}{\mathbb{E}\left[|w|^2\right]} = \frac{\mathbb{E}\left[g^2\right] E_\mathrm{s}}{N_0} \tag{2.10}$$

where

$$E_\mathrm{s} \triangleq \frac{1}{M} \sum_{x \in \mathcal{X}} |x|^2 \tag{2.11}$$

denotes the average transmit energy per symbol duration. Without loss of generality, we will assume the normalization $\mathbb{E}[g^2] = 1$, such that $\mathsf{SNR} = E_\mathrm{s}/N_0$.

2.3.1 Standard Derivation via Belief Propagation

It has been shown in [86] that the iterative turbo decoder can be derived as an instance of *belief propagation*[2] (BP), a message passing algorithm for statistical inference. It operates on a graphical representation of some joint probability distribution $p(x_0, \ldots, x_{N-1}, \boldsymbol{y})$, and computes the posterior marginal distributions of the unknown variables x_n, conditioned on the observation $\boldsymbol{y}$. On singly-connected graphs, the algorithm terminates in finite time and returns the exact $p(x_n \,|\, \boldsymbol{y})$. In the presence of cycles, it becomes iterative and only yields approximate results, the so-called *beliefs*, denoted with the letter q in order to distinguish them from the exact marginals:

$$q(x_n) \approx p(x_n \,|\, \boldsymbol{y}). \tag{2.12}$$

In this thesis, we will use *factor graphs*, a particular family of graphical models which appears to be the most common one in the communication literature. Tutorials on belief propagation in conjunction with the classical bipartite factor graphs (containing factor and variable nodes) and with Forney-style factor graphs (consisting only of factor nodes) can be found in [69] and [78], respectively. Furthermore, Appendix B gives an introduction into the more general divergence minimization framework [94] which contains BP as a special case, and which will be used extensively in the remainder of this thesis. See in particular Section B.2 for a brief summary of factor graphs and BP.

The well-known message update equations for belief propagation on bipartite factor graphs are derived in the appendix, c. f. (B.86) and (B.87), and are repeated here for the special case of discrete variables. The message from a variable node x_n to an adjacent factor node f (x_n and f are connected if and only if the function f depends on x_n) is

$$m_{x_n \to f}(x_n) \propto \prod_{g \neq f} m_{g \to x_n}(x_n) \tag{2.13}$$

[2] Of *loopy* belief propagation, to be precise, due to the cycles in the underlying factor graph.

where the product is over all functions other than f that depend on x_n. Furthermore, if $\boldsymbol{x}_f$ is a vector that contains the parameters of f (i. e., $\boldsymbol{x}_f$ is a subvector of $(x_0, \ldots, x_{N-1})$), then the message from factor node f to an adjacent variable node x_n is

$$m_{f \to x_n}(x_n) \propto \sum_{\sim x_n} f(\boldsymbol{x}_f) \prod_{m \neq n} m_{x_m \to f}(x_m) \tag{2.14}$$

where the sum is over all possible assignments to the arguments of f, except of x_n whose value remains fixed. Note that both equations are only defined up to (positive) factors, so we are free to drop any normalization constants.

Finally, the belief of a variable x_n is given as the normalized product of all messages towards its associated variable node:

$$q(x_n) = \frac{1}{Z} \prod_f m_{f \to x_n}(x_n) \tag{2.15}$$

where Z ensures that $\sum_{x_n} q(x_n) = 1$.

In the following, we will apply BP to the factor graph shown in Figure 2.4, which represents the signal model described at the beginning of this section.

Information from the Channel

The likelihood functions $p(y_n \mid x_n)$ depend on x_n and y_n, but the observed variables y_n are not represented by their own variable nodes; they merely parameterize the likelihood functions. The factor node $p(y_n \mid x_n)$ is therefore only connected to the variable node x_n, and the corresponding message is simply

$$\begin{aligned} m_{p(y_n \mid x_n) \to x_n}(x_n) &\propto p(y_n \mid x_n) \\ &\propto \exp\left(-\frac{|y_n - g_n x_n|^2}{N_0} \right) \\ &= \exp\left(-\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 \right) \\ &\propto \mathcal{CN}\left(x_n \mid \frac{y_n}{g_n}, \frac{N_0}{g_n^2} \right) \end{aligned} \tag{2.16}$$

which follows immediately from (2.9).

The variable node x_n is of degree 2,[3] so there is only one factor on the right hand side of (2.13), and thus

$$m_{x_n \to p(x_n \mid c_{n,0}, c_{n,1})}(x_n) \propto m_{p(y_n \mid x_n) \to x_n}(x_n) \propto \mathcal{CN}\left(x_n \mid \frac{y_n}{g_n}, \frac{N_0}{g_n^2} \right). \tag{2.17}$$

[3] The *degree* of a node is the number of edges attached to it.

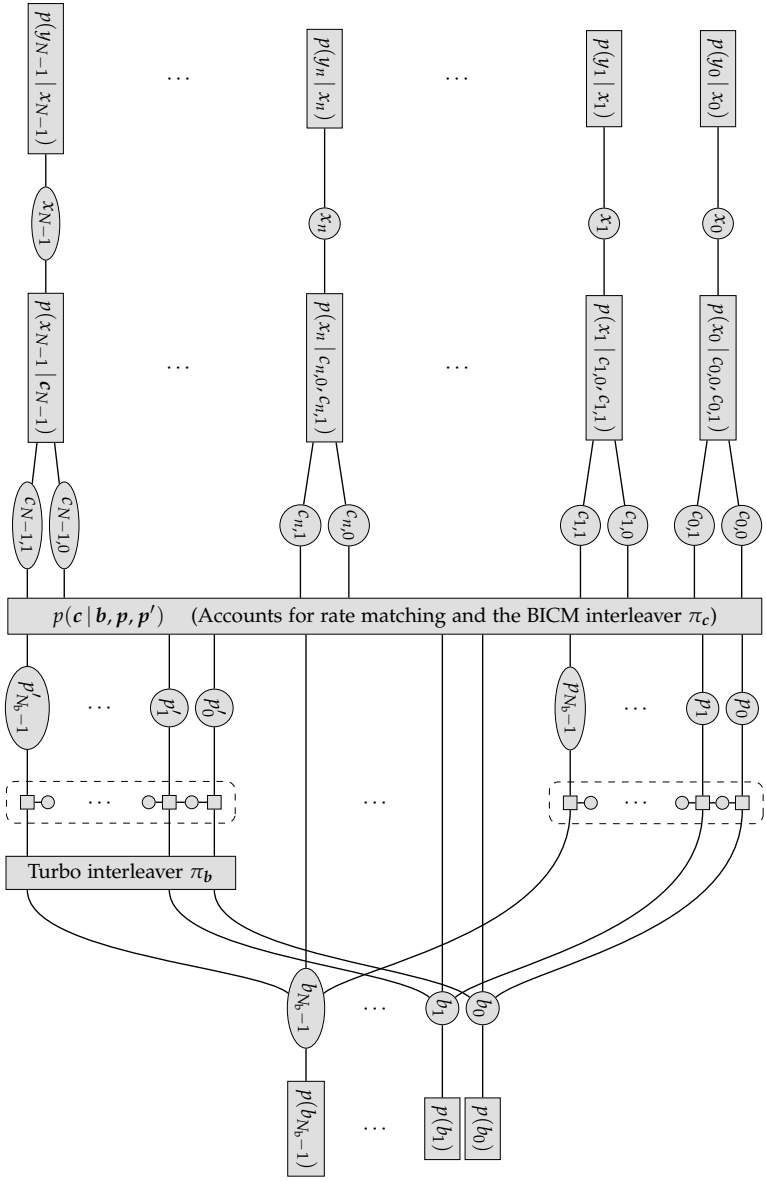


Figure 2.4: Factor graph of a BICM-ID receiver with the code from Figure 2.3, QPSK modulation, and a memoryless channel

Intrinsic, Extrinsic, and Posterior Information

In the literature on BICM-ID, the parameters $\vec{\lambda}$ of the forward messages are often called *intrinsic* L-values, with the rationale that in BPSK-modulated transmissions, where there is only one code bit per channel use, $\vec{\lambda}_n$ describes the information about c_n that is contained in the nth observation y_n.

Similarly, the parameters of the backward messages $\overleftarrow{\lambda}$ are called *extrinsic* L-values, since in BPSK systems, $\overleftarrow{\lambda}_n$ describes (exactly or approximately) the information about c_n contained in all channel uses except of the nth.

Finally, we will refer to $\lambda \triangleq \vec{\lambda} + \overleftarrow{\lambda}$ as the *posterior* L-values, since they parameterize the beliefs of the codebits, which are approximations of the posterior marginal distributions.

It is a general property of degree-2 variable nodes that they just relay their message unchanged, so it makes little sense to distinguish between those two messages at all. Thus, we will usually drop variable nodes of degree 2 altogether and connect their two neighboring factor nodes directly; see the discussion on Tanner vs Forney-style factor graphs on page 236*ff*, and the text box on message nomenclature at the end of this chapter.

Demapping

Now we turn to the derivation of the soft demapper. The function $p(x_n \mid c_{n,0}, c_{n,1})$ describes the deterministic mapping of code bits to data symbols, and is given as

$$p(x_n \mid c_{n,0}, c_{n,1}) = \mathbb{1}\left[x_n = \mathcal{M}(c_{n,0}, c_{n,1})\right] \tag{2.18}$$

where $\mathbb{1}[A]$ is the indicator function: $\mathbb{1}[A] = 1$ if A is true, and $\mathbb{1}[A] = 0$ otherwise.

All messages to and from binary variable nodes are (proportional to) Bernoulli distributions, and can therefore be conveniently represented by *L-values* (see (A.75) on page 214), defined as

$$\lambda = \log \frac{m(1)}{m(0)} \tag{2.19}$$

with inverse

$$m(c) \propto \exp(c\lambda), \qquad c \in \mathbb{F}. \tag{2.20}$$

We denote the L-value of the forward message from code bit $c_{n,k}$ to the decoder as $\vec{\lambda}_{n,k}$, and that of the backward message from the decoder to the demapper as $\overleftarrow{\lambda}_{n,k}$. As illustrated in the text box above, these are also called *intrinsic* and *extrinsic* L-values, respectively.

Now, applying (2.14), the message to the first code bit is

$$\begin{aligned}
&m_{p(x_n \mid c_{n,0},c_{n,1})\to c_{n,0}}(c_{n,0}) \\
&\quad\propto \sum_{c_{n,1}}\sum_{x_n} p(x_n \mid c_{n,0},c_{n,1})\, m_{x_n\to p(x_n \mid c_{n,0},c_{n,1})}(x_n)\, m_{c_{n,1}\to p(x_n \mid c_{n,0},c_{n,1})}(c_{n,1}) \\
&\quad\propto \sum_{c_{n,1}}\sum_{x_n} \mathbb{1}\left[x_n = \mathcal{M}(c_{n,0},c_{n,1})\right] \exp\left(-\frac{g_n^2}{N_0}\left|x_n - \frac{y_n}{g_n}\right|^2\right) \exp\left(c_{n,1}\overleftarrow{\lambda}_{n,1}\right) \\
&\quad= \sum_{c_{n,1}} \exp\left(-\frac{g_n^2}{N_0}\left|\mathcal{M}(c_{n,0},c_{n,1}) - \frac{y_n}{g_n}\right|^2 + c_{n,1}\overleftarrow{\lambda}_{n,1}\right). \qquad (2.21)
\end{aligned}$$

The intrinsic L-value for $c_{n,0}$ is thus

$$\begin{aligned}
\vec{\lambda}_{n,0} &\triangleq \log\frac{m_{p(x_n \mid c_{n,0},c_{n,1})\to c_{n,0}}(1)}{m_{p(x_n \mid c_{n,0},c_{n,1})\to c_{n,0}}(0)} \\
&= \log\frac{\sum\limits_{c_{n,1}} \exp\left(-\frac{g_n^2}{N_0}\left|\mathcal{M}(1,c_{n,1}) - \frac{y_n}{g_n}\right|^2 + c_{n,1}\overleftarrow{\lambda}_{n,1}\right)}{\sum\limits_{c_{n,1}} \exp\left(-\frac{g_n^2}{N_0}\left|\mathcal{M}(0,c_{n,1}) - \frac{y_n}{g_n}\right|^2 + c_{n,1}\overleftarrow{\lambda}_{n,1}\right)} \\
&\overset{(a)}{=} \log\frac{\sum\limits_{c_{n,1}} \exp\left(-\frac{g_n^2}{N_0}\left|\mathcal{M}(1,c_{n,1}) - \frac{y_n}{g_n}\right|^2 + 1\cdot\overleftarrow{\lambda}_{n,0} + c_{n,1}\overleftarrow{\lambda}_{n,1}\right)}{\sum\limits_{c_{n,1}} \exp\left(-\frac{g_n^2}{N_0}\left|\mathcal{M}(0,c_{n,1}) - \frac{y_n}{g_n}\right|^2 + 0\cdot\overleftarrow{\lambda}_{n,0} + c_{n,1}\overleftarrow{\lambda}_{n,1}\right)} - \overleftarrow{\lambda}_{n,0} \\
&\overset{(b)}{=} \log\frac{\sum\limits_{\boldsymbol{c}_n:c_{n,0}=1} \exp\left(-\frac{g_n^2}{N_0}\left|\mathcal{M}(\boldsymbol{c}_n) - \frac{y_n}{g_n}\right|^2 + \boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right)}{\sum\limits_{\boldsymbol{c}_n:c_{n,0}=0} \exp\left(-\frac{g_n^2}{N_0}\left|\mathcal{M}(\boldsymbol{c}_n) - \frac{y_n}{g_n}\right|^2 + \boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right)} - \overleftarrow{\lambda}_{n,0} \qquad (2.22)
\end{aligned}$$

where (a) adds and subtracts $\overleftarrow{\lambda}_{n,0}$, and (b) switches to a more compact vector notation, with $\boldsymbol{c}_n \triangleq (c_{n,0},\ldots,c_{n,K-1})^{\mathsf{T}}$ and $\overleftarrow{\boldsymbol{\lambda}}_n \triangleq (\overleftarrow{\lambda}_{n,0},\ldots,\overleftarrow{\lambda}_{n,K-1})^{\mathsf{T}}$. The operator $\sum_{\boldsymbol{c}_n:c_{n,k}=\beta}$ denotes summation over all bit labels $\boldsymbol{c}_n$ with the value of $c_{n,k}$ fixed to $\beta \in \mathbb{F}$. The last expression (2.22) has the additional advantage that it holds for any K, not just $K = 2$.[4]

Defining $\mathcal{X}_k^\beta$ as the set of symbols whose bit label has a β in the kth position, our final expression of the soft demapper, which avoids the somewhat clumsy $\mathcal{M}(\boldsymbol{c}_n)$, becomes

$$\vec{\lambda}_{n,k} = \log\frac{\sum\limits_{x_n\in\mathcal{X}_k^1} \exp\left(-\frac{g_n^2}{N_0}\left|x_n - \frac{y_n}{g_n}\right|^2 + \boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right)}{\sum\limits_{x_n\in\mathcal{X}_k^0} \exp\left(-\frac{g_n^2}{N_0}\left|x_n - \frac{y_n}{g_n}\right|^2 + \boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right)} - \overleftarrow{\lambda}_{n,k} \qquad (2.23)$$

[4] Though we have not strictly shown this. A more general derivation for arbitrary K will be given in Section 2.3.2.

where $\boldsymbol{c}_n$ is implicitly understood to contain the bit label of the current x_n. This slightly imprecise notation should not cause any confusion.

Computing (2.23) requires a full enumeration of all $x \in \mathcal{X}$, and therefore has a complexity of order $\mathcal{O}(M) = \mathcal{O}(2^K)$. A common trick to reduce the complexity is the *max-log* approximation, where the sums in (2.23) are replaced by their largest terms:

$$\begin{aligned}\vec{\lambda}_{n,k} &\approx \max_{x_n \in \mathcal{X}_k^1} \left(-\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 + \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right) - \max_{x_n \in \mathcal{X}_k^0} \left(-\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 + \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right) - \overleftarrow{\lambda}_{n,k} \\ &= \min_{x_n \in \mathcal{X}_k^0} \left(\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 - \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right) - \min_{x_n \in \mathcal{X}_k^1} \left(\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 - \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right) - \overleftarrow{\lambda}_{n,k}. \end{aligned} \tag{2.24}$$

Naïvely searching for the two minima in (2.24) still requires a full enumeration of the set $\mathcal{X}$, but for specific mapping schemes it is possible to implement the max-log demapper with lower complexity; see for example [3] for QAM and [141] for PSK modulations.

Decoding

As mentioned before, we will only sketch the implementation of the decoder briefly, in an informal way. The box $p(\boldsymbol{c} \,|\, \boldsymbol{b}, \boldsymbol{p}, \boldsymbol{p}')$ in Figure 2.4 can be interpreted as an edge interleaver, which connects the nodes of the systematic and parity bits with the corresponding code bits. If a parity bit is punctured, the respective variable node p_i or p_i' has a degree of 1; it is only connected to the decoder, but not to the demapper. It therefore sends a uniform message to the decoder, i. e., the constant L-value $\vec{\lambda}_i = 0$.

The upper dashed box represents the first RSC decoder, which gets two kinds of input. From the left, it takes the L-values of the parity bits $\boldsymbol{p} = (p_0, \ldots, p_{N_\mathrm{b}-1})$, and from the right, it gets information about the systematic bits $\boldsymbol{b} = (b_0, \ldots, b_{N_\mathrm{b}-1})$. Messages are multiplied at variable nodes, which in the case of binary variables corresponds to a summation of their L-values. Thus, the L-value for b_i which is fed to the RSC decoder is a sum of three other L-values: (a) the information about b_i from the channel, (b) the information from the second RSC decoder, and (c) the prior information, which we assume to be uniform in this thesis, but which can in general be non-uniform, for instance in the presence of an ISCD loop.

The standard implementation of the RSC decoder is via the BCJR algorithm [7], which provides extrinsic L-values of the systematic and the parity bits. The latter are fed back to the demapper for the next BICM-ID iteration, while the former are used in the following run of the second RSC decoder, which proceeds analogously. After one or several turbo iteration (where one turbo iteration consists of running both RSC decoders once), the receiver either continues with another BICM-ID iteration, or it terminates after computing a hard estimate $\hat{\boldsymbol{b}}$ by thresholding the L-values of the systematic bits.

2.3.2 Alternative Derivation via Expectation Propagation

Expectation propagation (EP) is a generalization of belief propagation, where the update equation (2.14) for the factor-to-variable messages is replaced by

$$m_{f\to x_n}(x_n) \propto \frac{1}{m_{x_n\to f}(x_n)} \operatorname{proj}_{\mathcal{Q}_{x_n}}\Bigg[m_{x_n\to f}(x_n) \underbrace{\sum_{\sim x_n} f(\boldsymbol{x}_f) \prod_{m\neq n} m_{x_m\to f}(x_m)}_{\text{belief propagation}}\Bigg]. \tag{2.25}$$

The precise definition of the projection operator $\operatorname{proj}_{\mathcal{Q}}[p]$, where $\mathcal{Q}$ is a set of probability distributions, is given in Appendix B.3.2 on page 242. Informally, it returns the distribution $q \in \mathcal{Q}$ which, in a certain sense, is the best approximation of p. As an ubiquitous example, the projection of some (in general non-Gaussian) p into the set of Gaussians returns the Gaussian distribution with the same mean and variance as p ("moment matching"). In the special case where p itself is an element of $\mathcal{Q}$, the projection becomes the identity operation and (2.25) simplifies to (2.14), which shows that EP really is a generalization of BP.

The problem that EP solves is that the BP messages often become too complex to be used in practical implementations. There are essentially only two kinds of probability networks where BP is applicable without approximations. The first one are problems where all variables are discrete (and with sufficiently small range[5] so that the sums in (2.14) can actually be computed). The BICM-ID receiver belongs into this class. The other example are linear Gaussian networks, where BP leads to the well-known Kalman filter. Due to the projection step, which can be used to control the complexity of the resulting messages, EP is applicable to many further problems where plain BP fails.

We will use EP several times in this thesis to derive novel receiver algorithms; for example, the iterative MIMO detector from Section 4.3 is obtained by replacing the discrete data symbols with Gaussian variables. In the remainder of this chapter, however, we will only re-derive the BICM-ID receiver from the previous section by means of EP. Why would we want to do this? For simplicity and generality, mainly. The factor graph in Figure 2.4 represents a very fine-grained model of the BICM-ID receiver, showing the dependencies between individual bits. This is the right level of detail for the derivation of decoding algorithms, like the turbo decoder in this example. The focus of this thesis, however, is mainly on the inner receiver. Sticking to such a detailed view of the outer receiver would only be distracting for our purposes, and it would also needlessly restrict the generality of the derivations, because our underlying models would always assume one specific modulation and coding scheme.

Compare the detailed model from Figure 2.4 with the more compact representation in Figure 2.5, where the code bits $(c_{n,0}, c_{n,1})$ that belong to one channel use are merged into the bit vector $\boldsymbol{c}_n$, and where the systematic and parity bits are merged into the

[5] The *range* of a random variable is the set of all values that it can assume: $\operatorname{range}(X) \triangleq \{X(\omega) \,|\, \omega \in \Omega\}$.

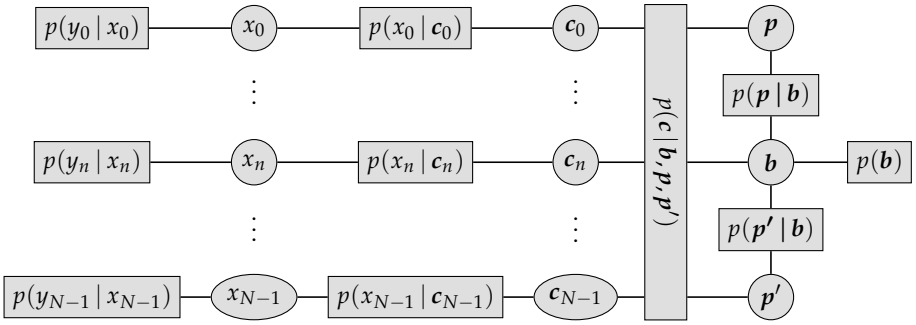


Figure 2.5: Compact representation of the BICM-ID receiver from Figure 2.4, where the factorization of the beliefs down to the individual bits is kept implicit

bit vectors $\boldsymbol{b}$, $\boldsymbol{p}$ and $\boldsymbol{p}'$. The reader will probably agree that Figure 2.5 is much better suited for a high-level overview of the probability network. However, we would run into a problem if we were trying to apply belief propagation to the compact model. Theoretically, it would of course still be possible, and since the subgraph which describes the turbo code is now cycle-free, BP would actually yield an exact, non-iterative decoder. The problem is that the discrete random variables $\boldsymbol{b}$, $\boldsymbol{p}$ and $\boldsymbol{p}'$ each have a range of cardinality 2^{N_b}, so describing their distributions would require $2^{N_\mathrm{b}} - 1$ real numbers, which is far too much for any block size of practical interest. Running BP on Figure 2.5 would essentially boil down to a brute-force algorithm, which would compute the posterior probabilities of *all* possible messages.

When we applied BP to Figure 2.4 in the previous section, such an exponential blowup did not occur. The reason is that the detailed graph contains separate variable nodes for the bits, which forces BP to compute independent bitwise beliefs. The information about $\boldsymbol{b}$ is then given by the product $q(b_0) \cdots q(b_{N_\mathrm{b}-1})$ which can be represented by N_b real numbers, as opposed to the $2^{N_\mathrm{b}} - 1$ real numbers that would in general be required for a description of $q(\boldsymbol{b})$. However, thanks to the projection step in (2.25), we can work with the compact model and still get the same complexity reduction (and the same loss of optimality, of course) using EP; we just need to project the beliefs of the bit vectors into the set of bitwise factorized distributions.

Expectation Propagation with Fully Factorized Beliefs

As derived on page 243, the projection of a multivariate distribution $p(x_0, \ldots, x_{N-1})$ into the set of fully factorized distributions (distributions that can be written in the form $\prod_{n=0}^{N-1} q(x_n)$) is equal to the product of its marginals, i. e., $q(x_n) = p(x_n)$. Therefore, BP on a certain factor graph is equivalent to EP on another graph where some of the variable nodes are merged, as long as the beliefs of the new, "larger" nodes are constrained to be

(a) Factorization $q(x,y) = q(x)\,q(y)$ expressed explicitly by the graph structure

(b) Factorization $q(x,y) = q(x)\,q(y)$ imposed implicitly by the constraint $\mathcal{Q}_{x,y} = \mathcal{P}_x \times \mathcal{P}_y$

Figure 2.6: Simple factor graphs for the comparison of BP and EP

factorized in the same way as in the first graph. The only difference is that BP makes the factorization of the beliefs *explicit* in the graph structure, while EP leaves it *implicit* in the constraints of the beliefs.

Rather than giving a formal proof for this statement, let us consider a simple example, which, however, is general enough to cover all relevant aspects. We start by defining some notation. For a random variable $x \in \mathcal{X}$, let $\mathcal{P}_x$ denote the set of all probability distributions over $\mathcal{X}$. Also, for any x which is represented by its own variable node, let $\mathcal{Q}_x \subseteq \mathcal{P}_x$ be the set of allowed beliefs $q(x)$, i. e., the target set of the projection in (2.25). In the special case of BP, there is no projection, and thus $\mathcal{Q}_x = \mathcal{P}_x$.

Consider the two simple graphs in Figure 2.6. If we apply BP to the left-hand graph, the messages from f to the variables x and y are easily found as

$$m^{\mathrm{BP}}_{f\to x}(x) \propto \sum_y \sum_z f(x,y,z)\, m_{z\to f}(z)\, m_{y\to f}(y) = \sum_y \tilde{f}(x,y)\, m_{y\to f}(y) \tag{2.26}$$

$$m^{\mathrm{BP}}_{f\to y}(y) \propto \sum_x \sum_z f(x,y,z)\, m_{z\to f}(z)\, m_{x\to f}(x) = \sum_x \tilde{f}(x,y)\, m_{x\to f}(x) \tag{2.27}$$

with the intermediate result $\tilde{f}(x,y) \triangleq \sum_z f(x,y,z)\, m_{z\to f}(z)$.

Now we apply EP to the right-hand graph, where x and y are merged into a single variable node. The message from f to the joint variable node (x,y) is in general

$$m^{\mathrm{EP}}_{f\to(x,y)}(x,y) \propto \frac{1}{m_{(x,y)\to f}(x,y)}\, \mathrm{proj}_{\mathcal{Q}_{x,y}}\left[m_{(x,y)\to f}(x,y) \sum_z f(x,y,z)\, m_{z\to f}(z) \right]. \tag{2.28}$$

By setting $\mathcal{Q}_{x,y} = \mathcal{P}_x \times \mathcal{P}_y$, we force the belief $q(x,y)$ into the factorized form $q(x)\,q(y)$. Recall from (2.15) that beliefs are given by the product of all incoming messages. Therefore, the constraint $q(x,y) = q(x)\,q(y)$ implies that all messages to and from the node (x,y) are factorized, too. We can thus write (2.28) as

$$\begin{aligned} m^{\mathrm{EP}}_{f\to x}(x)\, m^{\mathrm{EP}}_{f\to y}(y) &\propto \frac{\mathrm{proj}_{\mathcal{P}_x\times\mathcal{P}_y}\left[m_{x\to f}(x)\, m_{y\to f}(y)\, \tilde{f}(x,y) \right]}{m_{x\to f}(x)\, m_{y\to f}(y)} \\ &\overset{(a)}{=} \frac{\left(\sum_y m_{x\to f}(x)\, m_{y\to f}(y)\, \tilde{f}(x,y)\right)\left(\sum_x m_{x\to f}(x)\, m_{y\to f}(y)\, \tilde{f}(x,y)\right)}{m_{x\to f}(x)\, m_{y\to f}(y)} \end{aligned}$$

$$= \left(\sum_y m_{y\to f}(y)\, \tilde{f}(x,y) \right) \left(\sum_x m_{x\to f}(x)\, \tilde{f}(x,y) \right)$$
$$\propto m^{\mathrm{BP}}_{f\to x}(x)\, m^{\mathrm{BP}}_{f\to y}(y) \tag{2.29}$$

where (a) uses the result that the projection of any $p(x,y)$ into the set of factorized distributions is equal to $p(x)\,p(y)$. We have thus shown that

$$m^{\mathrm{EP}}_{f\to x}(x) \propto m^{\mathrm{BP}}_{f\to x}(x) \qquad \text{and} \qquad m^{\mathrm{EP}}_{f\to y}(y) \propto m^{\mathrm{BP}}_{f\to y}(y) \tag{2.30}$$

and since messages are only defined up to constant factors, this means that BP and EP are equivalent in this simple example. It is straightforward to generalize this result to cases where more than two variables are merged, or where more than one further variable (the z in our example) exists.

Generic Soft Demapping

For completeness, let us now derive the soft demapper by applying EP to Figure 2.5. Compared to the previous derivation based on BP, the following computation has the advantage that it does not make any assumption on the value of K.

Since we want to constrain $q(\boldsymbol{c}_n)$ to the factorized form $q(c_{n,0}) \cdots q(c_{n,K-1})$, we select[6] $\mathcal{Q}_{\boldsymbol{c}_n} = \mathcal{B}^K$ and obtain

$$\begin{aligned}
&m_{p(x_n \mid \boldsymbol{c}_n)\to \boldsymbol{c}_n}(\boldsymbol{c}_n) \\
&\quad\propto \frac{1}{m_{\boldsymbol{c}_n\to p(x_n\mid \boldsymbol{c}_n)}(\boldsymbol{c}_n)} \operatorname{proj}_{\mathcal{B}^K}\left[m_{\boldsymbol{c}_n\to p(x_n\mid \boldsymbol{c}_n)}(\boldsymbol{c}_n) \sum_{x_n} p(x_n \mid \boldsymbol{c}_n)\, m_{x_n\to p(x_n\mid \boldsymbol{c}_n)}(x_n) \right] \\
&\quad\propto \exp\left(-\boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n\right) \operatorname{proj}_{\mathcal{B}^K}\left[\exp\left(\boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n\right) \exp\left(-\frac{g_n^2}{N_0} \left| \mathcal{M}(\boldsymbol{c}_n) - \frac{y_n}{g_n} \right|^2 \right) \right] \\
&\quad\propto \prod_{k=0}^{K-1} \exp\left(-c_{n,k} \overleftarrow{\lambda}_{n,k}\right) \sum_{\sim c_{n,k}} \exp\left(-\frac{g_n^2}{N_0} \left| \mathcal{M}(\boldsymbol{c}_n) - \frac{y_n}{g_n} \right|^2 + \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right)
\end{aligned} \tag{2.31}$$

where the sum in the last line is over all bits $c_{n,k'}$ with $k' \neq k$. The intrinsic L-value $\vec{\lambda}_{n,k}$ is now simply the L-value of the kth factor in (2.31):

$$\vec{\lambda}_{n,k} = \log \frac{\exp(-1\cdot \overleftarrow{\lambda}_{n,k}) \sum\limits_{\boldsymbol{c}_n : c_{n,k}=1} \exp\left(-\frac{g_n^2}{N_0} \left| \mathcal{M}(\boldsymbol{c}_n) - \frac{y_n}{g_n} \right|^2 + \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right)}{\exp(-0\cdot \overleftarrow{\lambda}_{n,k}) \sum\limits_{\boldsymbol{c}_n : c_{n,k}=0} \exp\left(-\frac{g_n^2}{N_0} \left| \mathcal{M}(\boldsymbol{c}_n) - \frac{y_n}{g_n} \right|^2 + \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right)}$$

[6] As defined on page 215, $\mathcal{B}$ denotes the set of Bernoulli distributions (i. e., distributions over the binary field $\mathbb{F} \triangleq \{0,1\}$), and $\mathcal{B}^K$ is the set of bitwise factorized distributions over $\mathbb{F}^K$.

$$= \log \frac{\sum\limits_{x_n \in \mathcal{X}_k^1} \exp\left(-\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 + \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right)}{\sum\limits_{x_n \in \mathcal{X}_k^0} \exp\left(-\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 + \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right)} - \overleftarrow{\lambda}_{n,k} \tag{2.32}$$

which is identical to (2.23). As before, $\boldsymbol{c}_n$ in (2.32) contains the bit label of the current x_n.

Projecting Beliefs or Messages?

As a side note, we can use this derivation to demonstrate the importance of projecting the whole *belief* instead of just the factor-to-variable message; or in other words, the need for multiplying with $m_{x_n \to f}(x_n)$ inside the projection operator in (2.25), and dividing by it after the projection. If we ignored the two occurrences of $m_{x_n \to f}(x_n)$ in the message update equation (2.25), it would reduce to

$$\tilde{m}_{f \to x_n}(x_n) \propto \mathrm{proj}_{\mathcal{Q}_{x_n}} \left[\sum_{\sim x_n} f(\boldsymbol{x}_f) \prod_{m \neq n} m_{x_m \to f}(x_m) \right]. \tag{2.33}$$

Deriving the demapper based on (2.33) would yield the demapper-to-decoder message

$$\begin{aligned} \tilde{m}_{p(x_n \mid \boldsymbol{c}_n) \to \boldsymbol{c}_n}(\boldsymbol{c}_n) &\propto \mathrm{proj}_{\mathcal{B}^K} \left[\sum_{x_n} p(x_n \mid \boldsymbol{c}_n)\, m_{x_n \to p(x_n \mid \boldsymbol{c}_n)}(x_n) \right] \\ &\propto \prod_{k=0}^{K-1} \sum_{\sim c_{n,k}} \exp\left(-\frac{g_n^2}{N_0} \left| \mathcal{M}(\boldsymbol{c}_n) - \frac{y_n}{g_n} \right|^2 \right) \end{aligned} \tag{2.34}$$

and finally the intrinsic L-value

$$\tilde{\overrightarrow{\lambda}}_{n,k} = \log \frac{\sum\limits_{x_n \in \mathcal{X}_k^1} \exp\left(-\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 \right)}{\sum\limits_{x_n \in \mathcal{X}_k^0} \exp\left(-\frac{g_n^2}{N_0} \left| x_n - \frac{y_n}{g_n} \right|^2 \right)}. \tag{2.35}$$

As we can see, the extrinsic L-values have vanished. In the initial BICM-ID iteration, where there is no feedback from the decoder yet, (2.35) happens to coincide with (2.32). In the general case, however, (2.35) just ignores the feedback, which defeats the whole purpose of BICM-ID.

The reason for stressing this point is that there exist several papers whose authors attempt to derive receiver algorithms based on belief propagation, but, when faced with intractable BP messages, approximate them in some *ad hoc* way (like replacing them with Gaussians with the same mean and variance). As we have seen, such an approximation can be put onto solid mathematical ground by means of the projection operator, but

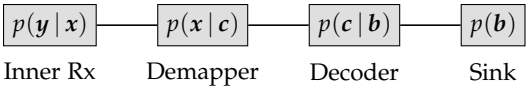


Figure 2.7: Forney-style factor graph of a generic BICM-ID receiver

said papers implicitly use the flawed update equation (2.33) instead of the correct (2.25). Later in this thesis, we will discuss some examples in more detail, where replacing (2.33) with (2.25) leads to a considerable performance gain.

A Fully Generic Model

While Figure 2.5 is a lot more compact than Figure 2.4, it still contains the assumptions of a parallelly connected turbo code and a memoryless channel. As a final step, we can abstract these details away, too, and obtain the fully generic model shown in Figure 2.7, which will be used on several occasions throughout this thesis. In particular, the code structure is now hidden inside the function node

$$p(\boldsymbol{c} \mid \boldsymbol{b}) = \mathbb{1}\left[\boldsymbol{c} = \mathcal{C}(\boldsymbol{b})\right] \tag{2.36}$$

where $\mathcal{C}(\boldsymbol{b})$ is the codeword that is associated with the information bits $\boldsymbol{b}$. Applying EP to this node leads to a similar derivation as (2.31) and (2.32) above, with the expected general result $\overleftarrow{\boldsymbol{\lambda}} = \boldsymbol{\lambda} - \overrightarrow{\boldsymbol{\lambda}}$, where $\boldsymbol{\lambda}$ contains the L-values of the bitwise beliefs.

This concludes our overview of the BICM system model and the iterative outer BICM-ID receiver. We are now equipped with the necessary background and notational conventions for the following discussion of the inner receiver.

Message Nomenclature

The traditional factor graphs are bipartite: every edge is connected to one factor node (say, f) and one variable node (say, x). Accordingly, every message is sent either from a factor to a variable node or from a variable to a factor node, which leads to the generic naming convention $m_{f\to x}(x)$ and $m_{x\to f}(x)$, respectively, which has been used in the present chapter.

However, many factor nodes have a natural interpretation as functional receiver units. In Figure 2.7 for example, the likelihood function $p(\boldsymbol{y} \mid \boldsymbol{x})$ corresponds to the inner receiver, the mapping constraint $p(\boldsymbol{x} \mid \boldsymbol{c})$ to the demapper, and the coding constraint $p(\boldsymbol{c} \mid \boldsymbol{b})$ to the decoder. For better readability, we will usually prefer the mnemonics of the blocks over the mathematical function names, so we would write for example $m_{\mathrm{dem}\to\boldsymbol{c}}(\boldsymbol{c})$ instead of $m_{p(\boldsymbol{x}\mid\boldsymbol{c})\to\boldsymbol{c}}(\boldsymbol{c})$.

In discrete-time models, where synchronization and matched filtering are abstracted away (which has been the case in the present chapter), we will often abbreviate the likelihood function $p(y_n \mid x_n)$ in message names as y_n, so $m_{p(y_n\mid x_n)\to x_n}(x_n)$ will simply be written as $m_{y_n\to x_n}(x_n)$. Since the observations y_n are not represented by their own variable nodes, this notation is unambiguous.

Furthermore, variable nodes with exactly two edges simply pass the messages through unaltered, so for example $m_{\mathrm{dem}\to\boldsymbol{c}}(\boldsymbol{c}) = m_{\boldsymbol{c}\to\mathrm{dec}}(\boldsymbol{c})$. We will thus usually discard the variable nodes of degree 2 (see also the description of Forney-style factor graphs in Appendix B.2.2 on page 236), connect the two neighboring factor nodes directly, and call the remaining message for instance $m_{\mathrm{dem}\to\mathrm{dec}}(\boldsymbol{c})$. The variable is then only determined implicitly by the function argument, which is analogous to the convention of writing $p(x)$ for the probability distribution $p_x(x)$, and should not cause any confusion.

Finally, we will use the terms *forward* and *backward* messages with two different meanings. On the one hand, forward messages are from "earlier" to "later" receiver blocks, e. g., from the inner receiver to the demapper, and from there to the decoder. Backward messages, on the other hand, are sent in the reverse direction. Parameters of these messages are marked with left-to-right and right-to-left arrows, respectively. For example, the intrinsic L-values that are passed from demapper to the decoder (the parameters of the message $m_{\mathrm{dem}\to\mathrm{dec}}(\boldsymbol{c})$) are denoted as $\vec{\boldsymbol{\lambda}}$, whereas $\overleftarrow{\boldsymbol{\lambda}}$ are the extrinsic L-values that are sent back from the decoder to the demapper.

A second meaning of "forward" and "backward" is in the context of temporal models, where forward messages propagate information in positive time direction (filtering), and backward messages in negative time direction (smoothing), all within the same functional block. In this context, the parameters of forward and backward messages are marked with subscripts "f" and "b", respectively.

Chapter 3

Parameter Estimation

In this chapter we will discuss the inner receiver, whose main task is the conversion of the underlying analog channel into an effective discrete-time channel, as given in (2.9) on page 13. Since this thesis is concerned with the design of digital receivers, the "underlying channel" does not only refer to the physical channel, but comprises everything between the D/A converter in the transmitter and the A/D converter in the receiver. In particular, it contains the analog transmit circuit, the physical radio channel, and the analog receiver front-end. We can thus roughly distinguish two kinds of channel parameters, which are *a priori* unknown and need to be estimated by the receiver:

- Multipath propagation caused by reflections of the electromagnetic wave at objects within the propagation path (scatterers) leads to the reception of multiple signal copies with different delays, attenuations, and phase shifts. Due to mobility of the transmitter, the receiver, or any scatterers in the environment, the impulse response of the channel is furthermore time-varying. Therefore, the radio channel is commonly modeled as a linear time-varying (LTV) filter. Static and/or non-dispersive channels are included as special cases in the LTV model. The process of learning and tracking the LTV filter coefficients is called *channel estimation*.

- *Synchronization*, in contrast, refers to the estimation and correction of impairments caused by the analog components. Timing offsets are unavoidable in practice, since the sampling clock in the receiver is unaware of the epoch of the transmitter, and the propagation delay of the radio waves. Other unknown channel parameters include a frequency offset, caused by mismatches between the local oscillators at transmitter and receiver, and phase noise due to oscillator instabilities. Finally, the received signal is superimposed with additive noise, usually modeled as a white Gaussian random process. Its variance also needs to be estimated, since it is required by the baseband receiver (e. g. in the demapper, as we have seen in the previous chapter) and also for higher-layer tasks like link adaptation.

Note, however, that the distinction between synchronization and channel estimation is not always clear-cut. Particularly the task of phase estimation is settled somewhere in between, since a time-varying phase offset is caused both by imperfect oscillators and a varying propagation delay due to mobility.

The focus of this chapter is on the latter class of problems, i. e., the estimation of synchronization parameters. It should be emphasized, however, that the techniques that will be studied in the following are equally applicable to the task of channel estimation. Indeed, I have derived code-aided channel estimators, and integrated them into the simulation framework that has been developed as a part of this research project. The rationale for my decision to restrict this presentation to synchronization algorithms is that they are, in some sense, more unusual and hence more interesting. In particular, channel estimators that are based on the common assumption of a Rayleigh fading channel process can be derived similarly as the phase noise estimator that will be studied in Section 3.6; with the main difference that the von Mises distributions, which are used by the phase noise estimator to encode its probabilistic knowledge about the state of the random process, need to be replaced by the more conventional Gaussian distributions.

This chapter is structured as follows. Section 3.1 begins with a definition of the fundamental problem that we attempt to solve. We then compare several important algorithmic classes, which can be considered as state-of-the-art approaches to parameter estimation, and discuss their respective shortcomings.

Section 3.2 acts as a mathematical interlude, where we introduce the concept of a likelihood function for continuous-time observations. Classically, the likelihood function of an unknown parameter is understood as the density function of the observation, conditioned on the parameter. However, this definition is not suitable for the design of synchronization algorithms, since these operate on random processes, which do not have a probability density. In some sense, it is the very purpose of synchronization to convert the continuous-time observation into a finite-dimensional sufficient statistic.

As a first step towards the derivation of code-aided synchronization algorithms, Section 3.3 focuses on *hard* parameter estimators. While the algorithms from that section are not novel, their new derivation from the principle of divergence minimization leads to some insights that, to my best knowledge, have not been mentioned in the existing literature.

The remaining sections then contain detailed case studies of Bayesian *soft* estimators. Sections 3.4–3.6 are concerned with the problem of phase estimation in several scenarios with increasing complexity. The development starts in Section 3.4 with an estimator for a constant phase offset, which is generalized to Wiener phase noise processes in Section 3.5, and then to general state space models in Section 3.6.

Finally, Section 3.7 closes our discussion of code-aided parameter estimation with the derivation of several SNR estimators, with varying degrees of complexity and algorithmic performance.

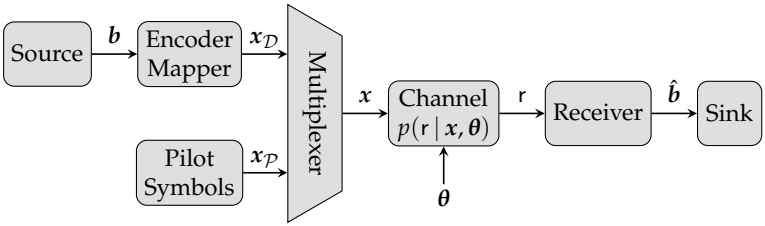


Figure 3.1: Generic system model, including the transmission of pilot symbols

3.1 State-of-the-Art Approaches to Synchronization

In this section, we introduce the synchronization task in a very general setting, and compare different approaches qualitatively. Detailed case studies will be given in the remainder of this chapter. The discussion will be based on the generic transmission system depicted in Figure 3.1, which is similar to the model from the previous chapter, but now also contains pilot symbols that are multiplexed into the stream of data symbols.

The transmitter wishes to transmit the message $\boldsymbol{b} \in \mathbb{F}^{N_\mathrm{b}}$. Via some coding and mapping scheme, it transforms the bit string $\boldsymbol{b}$ into a sequence of N_d complex data symbols $\boldsymbol{x}_\mathcal{D} \in \mathbb{C}^{N_\mathrm{d}}$. N_p pilot symbols $\boldsymbol{x}_\mathcal{P} \in \mathbb{C}^{N_\mathrm{p}}$ are multiplexed into the data stream in order to facilitate synchronization at the receiver. The sets $\mathcal{D}$ and $\mathcal{P}$ contain the time indices of the data and pilot symbols, respectively, so that $\mathcal{D} \cap \mathcal{P} = \emptyset$ and $\mathcal{D} \cup \mathcal{P} = \{0, \dots, N-1\}$, where $N = N_\mathrm{d} + N_\mathrm{p}$ is the total number of channel uses.

The resulting block $\boldsymbol{x}$ is transmitted over the channel, which includes the D/A converter in this system model. The received signal[1] r is a stochastic function which depends on the channel input $\boldsymbol{x}$ and on some synchronization parameters, collected in the vector $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. The stochastic nature of the channel is typically described in terms of the conditional PDF $p(\mathsf{r} \mid \boldsymbol{x}, \boldsymbol{\theta})$ of the channel output, given the input and the unknown parameters. However, this notation is not well defined, because r, being a random process, does not possess a probability density. We will address this problem in Section 3.2; for now, it suffices to think of r as a finite-dimensional vector $\boldsymbol{r}$ which contains the "essential" information about $\boldsymbol{b}$, in the sense that $p(\boldsymbol{b} \mid \boldsymbol{r}) \approx p(\boldsymbol{b} \mid \mathsf{r})$.

The ultimate task of the receiver's PHY-layer is to decide on an estimate $\hat{\boldsymbol{b}}$ of the transmitted message, which is then reported to the higher layers of the protocol stack. The design of the detection algorithm is affected by two competing optimization criteria. In order to maximize the *goodput*, i. e., the net data rate after packet retransmissions, the estimate should be as reliable as possible. At the same time, the computational complexity of the receiver needs to be kept reasonably low due to delay constraints and, particularly for battery-powered devices, in the interest of a low energy consumption.

[1] As introduced in Chapter 2, we denote continuous-time signals with a sans-serif font.

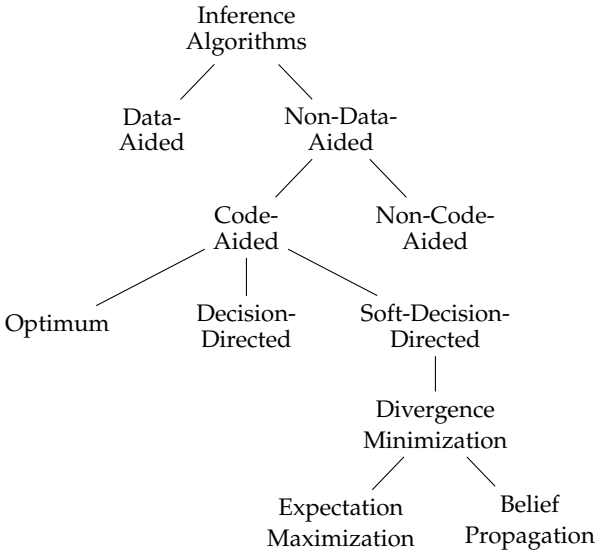


Figure 3.2: Relationship between classes of parameter estimation algorithms

In the following, we review some strategies for the design of receiver algorithms which cover a broad range of the performance-complexity tradeoff, from rather poorly performing but simple, up to optimum but prohibitively complex algorithms. Figure 3.2 shows a tree-structured overview of the algorithmic classes, where children are proper subsets of their parents. An exhaustive treatment of conventional receiver design, including the classes of data-aided (DA), non-code-aided (NCA), and decision-directed (DD) algorithms, can be found in [88]. An overview of the more recently proposed soft-decision-directed (SDD) approaches is given in [54].

Optimum Receivers Due to the two competing optimization criteria *performance* and *complexity*, it would be of ultimate interest to find pareto-optimal receivers. For instance, we may want to specify the maximum allowed complexity, and ask for the best-performing algorithm which does not exceed this complexity. Clearly, attempting to do so would be a daunting endeavor. Most of the practical algorithms that we will encounter in this thesis are able to trade off complexity vs performance, typically by adjusting the number of iterations, but none of them can claim pareto-optimality.

Therefore, we will use the term *optimum receiver* for the best-performing receiver with unconstrained complexity. While being unrealizable in practice for all but the

simplest scenarios, the optimum receiver is a natural starting point for the systematic derivation of suboptimum algorithms with feasible complexity.

In order to be able to speak about the "best-performing receiver," we need to agree on how to quantify the receiver's performance. Since the scope of this thesis is limited to the physical layer, which in particular excludes source coding, the most obvious measure for our purpose is the rate of erroneous decisions after channel decoding. This leaves us with two reasonable choices:

- The receiver that minimizes the *word error rate* (WER) $\mathbb{P}\left[\hat{\boldsymbol{b}} \neq \boldsymbol{b}\right]$, which computes

$$\hat{\boldsymbol{b}} = \underset{\boldsymbol{b} \in \mathbb{F}^{N_\mathrm{b}}}{\arg\max}\, p(\boldsymbol{b} \mid \mathsf{r}). \tag{3.1}$$

- The receiver that minimizes the *bit error rate* (BER) $\mathbb{P}\left[\hat{b}_i \neq b_i\right]$, which computes for each $i \in \{0, \ldots, N_\mathrm{b} - 1\}$

$$\begin{aligned} \hat{b}_i &= \underset{b_i \in \mathbb{F}}{\arg\max}\, p(b_i \mid \mathsf{r}) \\ &= \underset{b_i \in \mathbb{F}}{\arg\max} \sum_{b_j, j \neq i} p(\boldsymbol{b} \mid \mathsf{r}). \end{aligned} \tag{3.2}$$

A key component of both criteria is the posterior distribution $p(\boldsymbol{b} \mid \mathsf{r})$, which is either maximized or marginalized. In the presence of unknown synchronization parameters, this distribution is itself given as the marginal of the joint posterior of the data and the channel parameters:

$$\begin{aligned} p(\boldsymbol{b} \mid \mathsf{r}) &= \int_{\boldsymbol{\Theta}} p(\boldsymbol{b}, \boldsymbol{\theta} \mid \mathsf{r})\, \mathrm{d}\boldsymbol{\theta} \\ &\overset{(a)}{=} \int_{\boldsymbol{\Theta}} \frac{p(\mathsf{r} \mid \boldsymbol{b}, \boldsymbol{\theta})}{p(\mathsf{r})}\, p(\boldsymbol{b})\, p(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} \end{aligned} \tag{3.3}$$

where (a) assumes that the transmitted data and the channel parameters are statistically independent.

On the one hand, the word error rate is the more relevant metric for bidirectional systems like IEEE 802.11 (WLAN) or 3GPP LTE, which are able to detect transmission errors and request the retransmission of erroneously received blocks. On the other hand, solving (3.1) is generally an even harder problem than (3.2) [65, Section 13.1.1], since it involves marginalization and maximization steps, both of which are intractable in general, whereas (3.2) "only" consists of intractable marginalization (the maximization in (3.2) is trivial). We will therefore use (3.2) as a starting point for the derivation of suboptimal algorithms, but evaluate their performance in terms of the word error rate.

Data-Aided Synchronization Arguably the simplest approach to tackle the synchronization problem is by means of *data-aided* (DA) estimation algorithms, where the parameter estimation and data detection stages are decoupled. First, the unknown synchronization parameters are estimated using a training sequence, which is known to the receiver. Then, treating the estimate as the true value of $\boldsymbol{\theta}$, the detector computes a decision of the transmitted payload data.

To formalize this idea, assume that the received signal can be separated into two components $\mathsf{r}_{\mathcal{D}}$ and $\mathsf{r}_{\mathcal{P}}$ which depend only on the data and pilot symbols, respectively:

$$p(\mathsf{r}_{\mathcal{D}}, \mathsf{r}_{\mathcal{P}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta}) = p(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta})\, p(\mathsf{r}_{\mathcal{P}} \,|\, \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta}). \tag{3.4}$$

Then, the joint posterior of the data symbols and the channel parameters can be decomposed as

$$p(\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta} \,|\, \mathsf{r}, \boldsymbol{x}_{\mathcal{P}}) = \frac{1}{Z} \underbrace{p(\mathsf{r}_{\mathcal{P}} \,|\, \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})}_{\text{estimator}} \underbrace{p(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta})\, p(\boldsymbol{x}_{\mathcal{D}})}_{\text{detector}} \tag{3.5}$$

where $Z = p(\mathsf{r} \,|\, \boldsymbol{x}_{\mathcal{P}})$ is a normalization constant.

The synchronizer uses the first part of this decomposition to compute an estimate of the channel parameters, for instance using the *maximum a posteriori* (MAP) criterion

$$\hat{\boldsymbol{\theta}}^{\mathrm{MAP}} \triangleq \operatorname*{arg\,max}_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} p(\mathsf{r}_{\mathcal{P}} \,|\, \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta})\, p(\boldsymbol{\theta}). \tag{3.6}$$

In the absence of useful prior information on $\boldsymbol{\theta}$, it is reasonable to assume a uniform prior. In this case, (3.6) formally reduces to the maximum likelihood (ML) criterion

$$\hat{\boldsymbol{\theta}}^{\mathrm{ML}} \triangleq \operatorname*{arg\,max}_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} p(\mathsf{r}_{\mathcal{P}} \,|\, \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta}). \tag{3.7}$$

Of course, other estimators than MAP or ML could also be employed.

The detector subsequently treats $\hat{\boldsymbol{\theta}}$ as the true value of $\boldsymbol{\theta}$, and decides on the transmitted data by replacing the true posterior of the data in (3.1) or (3.2) (note the one-to-one correspondence between the bits $\boldsymbol{b}$ and the data sequence $\boldsymbol{x}_{\mathcal{D}}$) with[2]

$$p_{\mathsf{r}_{\mathcal{D}} | \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}}(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \hat{\boldsymbol{\theta}})\, p(\boldsymbol{x}_{\mathcal{D}}). \tag{3.8}$$

The concept of using a parameter estimate in lieu of the true value is sometimes called *synchronized detection* in the literature [8, 88]. However, this term is somewhat

[2] The notation $p_{\mathsf{r}_{\mathcal{D}} | \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}}(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \hat{\boldsymbol{\theta}})$ deserves some attention. It denotes the likelihood function $p(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta})$, evaluated at the point $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$. It is common to omit the subscript in such cases and simply write $p(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \hat{\boldsymbol{\theta}})$. However, with the convention that a missing subscript at a PDF implies a subscript that is equal to the argument list of p, this notation would expand to $p_{\mathsf{r}_{\mathcal{D}} | \boldsymbol{x}_{\mathcal{D}}, \hat{\boldsymbol{\theta}}}(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \hat{\boldsymbol{\theta}})$. In this notation, $\hat{\boldsymbol{\theta}}$ would be another random variable, distinct from $\boldsymbol{\theta}$; a subtle, but important difference. Having said this, we will also omit the subscript in the interest of compact notation, but we should always keep the intended meaning of $p(\mathsf{r}_{\mathcal{D}} \,|\, \boldsymbol{x}_{\mathcal{D}}, \hat{\boldsymbol{\theta}})$ in mind.

misleading, because the decision $\hat{\boldsymbol{\theta}}$ will be wrong with probability 1 (assuming that $\boldsymbol{\theta}$ is continuous), and thus the receiver will almost surely *not* be perfectly synchronized. We will therefore use the more accurate term *mismatched detection* in this thesis.

The advantages and disadvantages of the data-aided approach are evident. Since the pilot symbols are known to the receiver, simple closed-form solutions for $\hat{\boldsymbol{\theta}}$ can be found for many cases of interest, e. g. by solving $\partial p(\mathrm{r}_{\mathcal{P}} \mid \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta}) / \partial \boldsymbol{\theta} = \mathbf{0}$. On the negative side, the transmission of training data wastes bandwidth and power. Particularly in fast fading scenarios or short burst transmissions, the training overhead that is necessary for a sufficiently accurate synchronization may become inacceptably large.

Non-Data-Aided Synchronization *Non-data-aided* (NDA) algorithms do not rely on the transmission of known training data, but use the *a priori* unknown payload data instead. (With a slight abuse of terminology, we also include schemes which require a few pilot symbols, e. g. for coarse timing synchronization and phase ambiguity resolution, or for obtaining an initial estimate in iterative algorithms, in the category of NDA algorithms.)

NDA algorithms can be further subdivided into *non-code-aided* (NCA) and *code-aided* (CA) schemes.

Non-Code-Aided Synchronization Non-code-aided estimators disregard the structure of the data sequence that is introduced by the channel code. Rather, they treat the unknown data as an iid symbol sequence. Well-known examples of NCA algorithms are the Viterbi-Viterbi phase estimator for PSK-modulated signals [138], and the Oerder-Meyr timing estimator which exploits the cyclo-stationarity of the received signal [101].

Code-Aided Synchronization While NCA algorithms do succeed in reducing the pilot overhead, their performance is far from optimal since they ignore the statistical dependencies between the data symbols. In this thesis, we are therefore interested in code-aided algorithms, which use the code structure at least to a certain extend (as argued above, truly optimum receivers are practically infeasible for complexity reasons).

In contrast to an optimum receiver, which would solve the synchronization and detection/decoding problems jointly within one unit, virtually all code-aided algorithms that are practically implementable are *iterative*. The intuitive idea is to first run the receiver in DA mode to obtain an initial, but unreliable decision on the transmitted data. Using the detected data sequence as *virtual* pilot symbols, an updated estimate of the synchronization parameters is computed, yielding a new data decision. This process may be repeated until convergence is detected, or for a fixed number of iterations.

Decision-Directed Synchronization The so-called *decision-directed* (DD) approach is a straightforward, but quite *ad hoc* implementation of this idea. Here, the decoder output $\hat{\boldsymbol{b}}$ is re-mapped to yield hard decisions $\hat{\boldsymbol{x}}_{\mathcal{D}}$ on the data symbols. These are treated just

Hard vs Soft Decisions

In this thesis, a *hard decision* about a random variable $x \in \mathcal{X}$ is understood as one particular value $\hat{x} \in \mathcal{X}$. In contrast, a *soft decision* is a probability distribution over $\mathcal{X}$, called $q(x)$ in order to distinguish it from the prior distribution $p(x)$.
To treat both concepts on an equal footing, one can regard hard decisions as Dirac distributions, with $q(x) = \delta(x - \hat{x})$.
It is worth emphasizing that the above definition is valid for both *discrete* and *continuous* random variables. Some authors call an estimate $\hat{x}$ "soft" if x is a continuous variable, but this convention will not be followed in this thesis.

like pilot symbols by the synchronizer, so that standard DA estimation algorithms can be used in a DD receiver.

The fundamental problem with hard decision feedback is *error propagation*. The DD scheme can only be expected to work well as long as the initial parameter estimate is already sufficiently accurate to yield a low symbol error rate. Otherwise, the DD estimator uses many erroneous decisions, and the receiver will likely fail to converge to the correct solution.

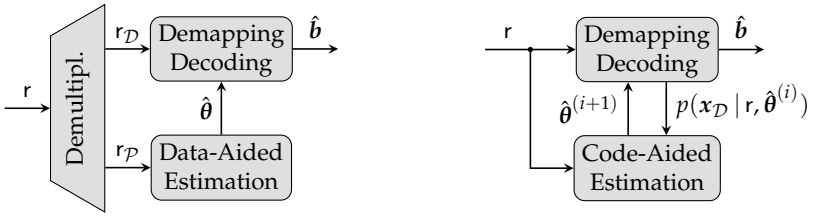
Soft-Decision-Directed Synchronization The key idea to solving this problem is to feed statistical information about the data symbols back to the synchronizer, rather than hard decisions. Intuitively, this allows the synchronizer to use only the reliably decoded data as virtual pilot symbols, and to ignore any unreliable decisions.

Such statistical feedback is referred to as a *soft decision* (see also the text box above), and iterative algorithms using soft feedback are called *soft-decision-directed* (SDD). In the following, some approaches for the systematic derivation of SDD receivers are discussed in more detail.

Expectation Maximization The optimal parameter estimator in terms of the ML criterion is given as

$$\begin{aligned}\hat{\boldsymbol{\theta}}^{\mathrm{ML}} &= \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} p(\mathsf{r} \mid \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \sum_{\boldsymbol{x}_{\mathcal{D}}} p(\mathsf{r} \mid \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta})\, p(\boldsymbol{x}_{\mathcal{D}}).\end{aligned} \tag{3.9}$$

The difficulty in solving (3.9) lies in the summation over the allowed data sequences. The above problem is thus an example of parameter estimation in the presence of *hidden data*: indeed, if $\boldsymbol{x}_{\mathcal{D}}$ were known, the problem would reduce to data-aided synchronization and would have a much simpler solution.



(a) DA synchronization: an estimate $\hat{\boldsymbol{\theta}}$ is obtained from $\mathsf{r}_{\mathcal{P}}$, which is subsequently treated as the true value of $\boldsymbol{\theta}$

(b) EM synchronization: the estimate $\hat{\boldsymbol{\theta}}$ is iteratively updated, using the posterior $p(\boldsymbol{x}_{\mathcal{D}} \mid \mathsf{r}, \boldsymbol{\theta})$ evaluated at the old estimate

Figure 3.3: Comparison of data-aided and EM-based synchronization

A standard approach to solving such estimation problems is the iterative expectation maximization algorithm [32] as described in Section B.5.2 on page 261. By specializing the generic EM parameter update equations (B.89) and (B.90) to the problem at hand, we obtain the following iterative estimator, where i denotes the iteration index:

$$\hat{\boldsymbol{\theta}}^{(i+1)} = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_{p(\boldsymbol{x}_{\mathcal{D}}|\mathsf{r},\hat{\boldsymbol{\theta}}^{(i)})}\left[\log p(\mathsf{r} \mid \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{x}_{\mathcal{P}}, \boldsymbol{\theta})\right]. \tag{3.10}$$

An overview of EM-based synchronization can be found in [100] and references therein.

The EM update is typically presented as a two-step procedure: in the E-step, the expectation of the log-likelihood function is computed, which is then maximized with respect to $\boldsymbol{\theta}$ in the M-step. However, in order to recognize parallels to the DA approach, it is more insightful to consider the computation of the posterior $p(\boldsymbol{x}_{\mathcal{D}} \mid \mathsf{r}, \hat{\boldsymbol{\theta}})$ as one step, and the update of $\hat{\boldsymbol{\theta}}$, using the posterior, as another step. The latter is analogous to the parameter estimator (3.7) from the DA receiver, with the difference that the perfectly known training data of (3.7) is replaced by statistical information about the data sequence. Similarly, computing the posterior $p(\boldsymbol{x}_{\mathcal{D}} \mid \mathsf{r}, \hat{\boldsymbol{\theta}})$, where the current parameter estimate is treated as the true value, corresponds to mismatched data detection. Figure 3.3 illustrates the comparison between DA and EM synchronization.

Unfortunately, an exact implementation of the EM synchronizer is rendered infeasible by the computation of $p(\boldsymbol{x}_{\mathcal{D}} \mid \mathsf{r}, \hat{\boldsymbol{\theta}})$. Since the data sequence $\boldsymbol{x}_{\mathcal{D}}$ is a discrete random variable whose range has cardinality 2^{N_b} (recall that N_b is the number of information bits per codeword), a full description of its distribution requires $2^{N_\mathrm{b}} - 1$ real numbers, which is far too complex for practically relevant block sizes. A common heuristic is to calculate the symbolwise posteriors $p(x_n \mid \mathsf{r}, \hat{\boldsymbol{\theta}})$ instead, and then approximate the sequencewise distribution as

$$p(\boldsymbol{x}_{\mathcal{D}} \mid \mathsf{r}, \hat{\boldsymbol{\theta}}) \approx \prod_{n\in\mathcal{D}} p(x_n \mid \mathsf{r}, \hat{\boldsymbol{\theta}}). \tag{3.11}$$

For simple transmission schemes like a convolutionally coded BPSK modulation, the symbolwise posteriors can be obtained exactly with the BCJR algorithm. But if more complex channel codes (e. g. turbo or LDPC codes) or higher-order mapping schemes (e. g. 16-QAM) are used, even the symbolwise posteriors cannot be computed exactly and need to be approximated, typically by algorithms which are themselves iterative (like turbo/LDPC decoding or BICM-ID).

In order to decrease the computational burden of EM synchronization in conjunction with an iterative decoder, it has been proposed to interweave the EM- with the decoder iterations [99]. Rather than reinitializing the decoder in every EM iteration and running it until convergence, the suggestion is to do only a single iteration inside the decoder in each EM iteration, and to keep the decoder state inbetween. Effectively, this scheme reduces the doubly-iterative algorithm to a singly-iterative one.

Numerical results reported in [99] show that, despite of these approximations, EM-based synchronizers perform well as long as the initial parameter estimate is reasonably close to the true value. It cannot be denied, though, that the approximations described above are fairly *ad hoc*, and that the result is, strictly speaking, not an EM algorithm. In particular, the convergence guarantee which is an often cited advantage of EM-based iterative schemes does not hold anymore. One of the major motivations for the introduction of EM into the synchronization literature has been to provide "A Theoretical Framework for Soft-Information-Based Synchronization in Iterative (Turbo) Receivers" [100]—an endeavor which has succeeded only to a certain extent.

Belief Propagation The algorithms described so far are *mismatched*, in the sense that they compute a hard parameter estimate $\hat{\boldsymbol{\theta}}$ and subsequently treat it as the true value of $\boldsymbol{\theta}$. However, as discussed on page 31, optimum decisions are based on the marginal posterior of the data

$$p(\boldsymbol{b} \,|\, \mathsf{r}) = \int_{\boldsymbol{\Theta}} p(\boldsymbol{b} \,|\, \mathsf{r}, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \,|\, \mathsf{r})\, \mathrm{d}\boldsymbol{\theta} \tag{3.12}$$

which is obtained by integration over the synchronization parameters. As long as the posterior of $\boldsymbol{\theta}$ is sharply peaked, i. e., $p(\boldsymbol{\theta} \,|\, \mathsf{r}) \approx \delta(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$, replacing (3.12) with $p(\boldsymbol{b} \,|\, \mathsf{r}, \hat{\boldsymbol{\theta}})$ is a good approximation. However, in the first (few) iteration(s), this is a very optimistic assumption. Indeed, if the initial estimates were already accurate, the initial data decision would be close to optimum, and iterations between synchronizer and decoder could not lead to any noteworthy gain. In such a situation, it would be reasonable to reduce the number of pilot symbols (minimizing the training overhead is, after all, the main raison d'être of iterative algorithms), which in turn would lead to a less accurate initial parameter estimate. We are therefore interested in algorithms which do not approximate $p(\boldsymbol{\theta} \,|\, \mathsf{r}) \approx \delta(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$, but attempt to solve the original problem (3.12) instead. In essence, these algorithms make *soft* rather than *hard* decisions on $\boldsymbol{\theta}$, which is analogous to the difference between soft and hard data feedback as discussed on page 33.

Belief propagation (BP), also known as the *sum-product algorithm*, is a general approach for solving marginalization problems like (3.12) approximately. It has become famous in the communication and information theory communities after the discovery that the celebrated turbo and LDPC decoding algorithms can be derived as instances of BP [86]. Extending the realm of BP to cover not only channel decoding but also parameter estimation tasks, as proposed in [151], is a consequential generalization.

As discussed in the previous chapter, belief propagation is usually illustrated as a message passing algorithm which operates on a factor graph of the underlying belief network. In principle, extending BP to include parameter estimation is straightforward: we just need to add variable nodes for the unknown parameters and factor nodes for the functions which specify the statistical dependencies between the parameters and the other variables. Then, an iterative estimation and detection algorithm can be instantiated by means of the generic message update equations (B.81) and (B.86) on page 257.

Unfortunately, using BP for parameter estimation is a lot more difficult than for detection/decoding, because the underlying variables are continuous. Hence, the update equations involve integrals rather than sums, which often have no closed-form solution. Furthermore, the messages may become arbitrarily complex functions rather than finite-dimensional vectors. To overcome this problem, [151] proposed the concept of *canonical distributions*, where the messages are restricted to certain families of distributions, whose members are fully specified by a few parameters. One possibility are Dirac distributions which, like the EM algorithm, lead to hard parameter estimates [55]. An alternative which retains soft information are mixtures of Dirac distributions:

$$q(\boldsymbol{\theta}) = \sum_{k=0}^{K-1} w^{(k)}\, \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^{(k)}), \quad \sum_{k=0}^{K-1} w^{(k)} = 1. \tag{3.13}$$

The locations $\boldsymbol{\theta}^{(k)}$ may be predetermined, for instance by uniformly sampling the range of $\boldsymbol{\theta}$, in which case only the probabilities $w^{(k)}$ are variable. This *discretization* of the continuous random variable can lead to tremendous simplifications, since we only have to deal with probability mass instead of density functions, which in particular reduces integrals to finite sums, but it is only feasible for low-dimensional $\boldsymbol{\theta}$. It is furthermore possible to also change the locations $\boldsymbol{\theta}^{(k)}$ at runtime, which gives rise to the so-called *particle filters* (in this context, the probabilities $w^{(k)}$ are interpreted as *weights* of the particles). Applications of these methods to the problem of phase estimation are discussed in [30].

To conclude, using belief propagation for the derivation of iterative synchronizers suffers from similar problems as expectation maximization: it is indeed possible to derive well-working algorithms, but only with additional, rather heuristic approximations. The resulting algorithms are not covered by the theoretical framework.

Divergence Minimization In the remainder of this chapter, we will apply the method of *divergence minimization* (DM) to the synchronization problem. As derived in full detail in Appendix B, divergence minimization is a very general inference framework which includes EM and BP as special cases. We will see that the concept of canonical distributions plays a central role in the derivation of practical algorithms, but the approximations are done in a theoretically sound manner by minimizing well-defined divergence measures between the exact and the approximated probability distributions.

What is the advantage of divergence minimization over the conventional approaches? It turns out that many algorithms that have earlier been derived *approximately* from EM or BP are fully covered by the DM framework. This alone is a good reason to prefer the generalized approach, since it retroactively provides a theoretical justification for their use. Even more interestingly, however, divergence minimization also allows us to derive improvements which, to the best of my knowledge, have not been proposed before.

3.2 The Likelihood Function for Continuous-Time Observations

As a prerequisite for the later development, we first take a closer look at the likelihood function. We mentioned earlier that the stochastic channel is typically described by

$$p(\mathsf{r} \mid \boldsymbol{b}, \boldsymbol{\theta}) \tag{3.14}$$

i. e., the probability density of its output, conditioned on the transmitted data and some channel parameters. After observing a particular realization of r at the receiver, the channel output is fixed, and we may treat (3.14) as a function of the data and the parameters—commonly called the *likelihood function* of $\boldsymbol{b}$ and $\boldsymbol{\theta}$. The maximum likelihood (ML) approach to estimation and detection is now to maximize (3.14). In particular, the *joint* ML parameter estimation and sequence detection problem can be stated as [88]

$$(\hat{\boldsymbol{b}}, \hat{\boldsymbol{\theta}}) = \underset{(\boldsymbol{b},\boldsymbol{\theta}) \in \mathbb{F}^{N_\mathrm{b}} \times \boldsymbol{\Theta}}{\arg\max} \; p(\mathsf{r} \mid \boldsymbol{b}, \boldsymbol{\theta}). \tag{3.15}$$

The problem is that the received signal r is a random *process* (as opposed to a random *vector*), and therefore does not possess a probability density.

A common strategy to circumvent this problem is to sample the continuous-time signal at a sufficiently high rate, and to replace r with a finite-dimensional vector $\boldsymbol{r}$ which is supposed to be a sufficient statistic for $\boldsymbol{b}$ (see e. g. [55, 114]). However, the sampling process is only lossless if the (noise-free) received signal is strictly bandlimited. This is indeed the case under the assumption of a bandlimited transmitted signal and a linear channel, but then the signal is necessarily of infinite duration [70, Section 6.8]. Sampling then merely converts the continuous-time into a discrete-time process, which does not have a density, either. Conversely, we may argue that any practical signal is of finite duration, but then it cannot be bandlimited, and the sampling process is lossy.

If the transmitted signal can be written as a linear combination of some basis functions $(\mathsf{g}_0, \ldots, \mathsf{g}_{N-1})$, for example

$$\mathsf{s}(t) = \sum_{n=0}^{N-1} x_n \, \mathsf{g}_n(t), \quad \mathsf{g}_n(t) = \mathsf{g}(t - nT) \tag{3.16}$$

as in linear modulation schemes, and if the signal is only distorted by AWGN, then

$$\big(\langle \mathsf{r}, \mathsf{g}_0 \rangle, \ldots, \langle \mathsf{r}, \mathsf{g}_{N-1} \rangle \big) \tag{3.17}$$

is a finite-dimensional sufficient statistic for guessing $\boldsymbol{x}$ based on r [70, Section 26.4]. Here, $\langle \mathsf{x}, \mathsf{y} \rangle \triangleq \int_{-\infty}^{\infty} \mathsf{x}(t) \mathsf{y}^*(t) \, \mathrm{d}t$ is the inner product[3] of functions $\mathsf{x}, \mathsf{y} : \mathbb{R} \to \mathbb{C}$. However, a perfect timing and frequency reference is needed in order to compute (3.17), so the above result cannot be used for the purpose of synchronization.

[3] Following the convention of [70], we define the inner product as linear in the first argument and conjugate linear in the second.

It goes without saying that the issue is of rather academic nature, since any practical signal is *virtually* limited both in duration and bandwidth. But it is nonetheless interesting to see how our receiver structure can be cleanly derived from first principles.

The key point here is that it is only the likelihood function $p(\mathsf{r} \,|\, \boldsymbol{b}, \boldsymbol{\theta})$ which is undefined, but this is just an auxiliary object, anyway. What we are really interested in is the posterior distribution $p(\boldsymbol{b} \,|\, \mathsf{r}) = \int_{\boldsymbol{\Theta}} p(\boldsymbol{b}, \boldsymbol{\theta} \,|\, \mathsf{r}) \, \mathrm{d}\boldsymbol{\theta}$, which is a well defined probability mass function. Thus, under the reasonable assumption that all data sequences have a strictly positive prior probability (without loss of generality, any sequence with zero prior probability can be removed from consideration), the ratio

$$\ell_{\mathsf{r}}(\boldsymbol{b}) \triangleq \frac{p(\boldsymbol{b} \,|\, \mathsf{r})}{p(\boldsymbol{b})} = \int_{\boldsymbol{\Theta}} \ell_{\mathsf{r}}(\boldsymbol{b}, \boldsymbol{\theta}) \, p(\boldsymbol{\theta} \,|\, \boldsymbol{b}) \, \mathrm{d}\boldsymbol{\theta} \tag{3.18}$$

with $\ell_{\mathsf{r}}(\boldsymbol{b}, \boldsymbol{\theta}) = \frac{p(\boldsymbol{b}, \boldsymbol{\theta} \,|\, \mathsf{r})}{p(\boldsymbol{b}, \boldsymbol{\theta})}$ is also well defined for all $\boldsymbol{b}$. The symbol ℓ reminds us that $\ell_{\mathsf{r}}(\boldsymbol{b}, \boldsymbol{\theta})$ is similar to the likelihood function $p(\mathsf{r} \,|\, \boldsymbol{b}, \boldsymbol{\theta})$, with the difference that $\ell_{\mathsf{r}}(\boldsymbol{b}, \boldsymbol{\theta})$ exists even if the observation is a random process. In the following, we assume that $\boldsymbol{b}$ and $\boldsymbol{\theta}$ are statistically independent, such that $p(\boldsymbol{\theta} \,|\, \boldsymbol{b}) = p(\boldsymbol{\theta})$. Note that, for a finite-dimensional observation vector $\boldsymbol{r}$, (3.18) simplifies to

$$\ell_{\boldsymbol{r}}(\boldsymbol{b}, \boldsymbol{\theta}) = \frac{p(\boldsymbol{r} \,|\, \boldsymbol{b}, \boldsymbol{\theta})}{p(\boldsymbol{r})} \tag{3.19}$$

by Bayes' rule. In this case, ℓ is proportional to the conventional likelihood function. For a continuous-time observation, however, (3.19) is not applicable.

Let $\mathsf{s}_{\boldsymbol{b}}$ denote the transmitted signal associated with the message $\boldsymbol{b}$. All possible transmit signals are assumed to be integrable,[4] $\mathsf{s}_{\boldsymbol{b}} \in \mathcal{L}_1$, and bandlimited to the bandwidth B_{s}. This implies that the signals are also energy-limited, $\mathsf{s}_{\boldsymbol{b}} \in \mathcal{L}_2$ [70, Note 6.4.12].

Assume that the channel is parameterized by $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, statistically independent of $\boldsymbol{b}$. Denote the noise-free received signal as $\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}$ which is bandlimited to B_{r} (in general, $B_{\mathsf{r}} > B_{\mathsf{s}}$ due to nonlinear effects like a frequency offset between transmitter and receiver). Assume that the channel has a finite gain, so that $\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}$ is also integrable and energy-limited for all $\boldsymbol{b}$ and $\boldsymbol{\theta}$. Let $\boldsymbol{b}_0$ be the transmitted message and $\boldsymbol{\theta}_0$ the true channel parameters. The received signal is then

$$\mathsf{r}(t) = \tilde{\mathsf{r}}_{\boldsymbol{b}_0,\boldsymbol{\theta}_0}(t) + \mathsf{v}(t), \quad -\infty < t < \infty \tag{3.20}$$

where v is a white proper Gaussian noise process of spectral density N_0 with respect to the bandwidth $W > B_{\mathsf{r}}$ (i. e., $\mathrm{Re}(\mathsf{v})$ and $\mathrm{Im}(\mathsf{v})$ are mutually uncorrelated real-valued white Gaussian noise processes of spectral density $N_0/2$, as defined in [70, Section 25.15]). Furthermore, v is assumed to be independent of $\tilde{\mathsf{r}}$.

[4] $\mathcal{L}_1 \triangleq \left\{ \mathsf{x} : \mathbb{R} \to \mathbb{C} \,\middle|\, \int_{-\infty}^{\infty} |\mathsf{x}(t)| \, \mathrm{d}t < \infty \right\}$ is the set of integrable signals, and $\mathcal{L}_2 \triangleq \left\{ \mathsf{x} : \mathbb{R} \to \mathbb{C} \,\middle|\, \int_{-\infty}^{\infty} |\mathsf{x}(t)|^2 \, \mathrm{d}t < \infty \right\}$ the set of energy-limited signals. Furthermore, the symbols B and W denote *one-sided* bandwidths.

As will be shown below, the posterior distribution of the data is then given as

$$p(\boldsymbol{b} \,|\, \mathsf{r}) = p(\boldsymbol{b}) \int_{\boldsymbol{\Theta}} \ell_{\mathsf{r}}(\boldsymbol{b}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} \tag{3.21}$$

where

$$\ell_{\mathsf{r}}(\boldsymbol{b}, \boldsymbol{\theta}) \propto \exp\left(\frac{\operatorname{Re}\left\langle \mathsf{r}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle}{N_0/2} - \frac{\left\| \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\|_2^2}{N_0} \right) \tag{3.22}$$

and the proportionality constant is such that

$$\sum_{\boldsymbol{b}} p(\boldsymbol{b} \,|\, \mathsf{r}) = \sum_{\boldsymbol{b}} p(\boldsymbol{b}) \int_{\boldsymbol{\Theta}} \ell_{\mathsf{r}}(\boldsymbol{b}, \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta} = 1. \tag{3.23}$$

In (3.22), $\langle \mathsf{x}, \mathsf{y} \rangle$ is the inner product as in (3.17), and $\|\mathsf{x}\|_2 \triangleq \sqrt{\langle \mathsf{x}, \mathsf{x} \rangle}$ is the $\mathcal{L}_2$-norm.[5]

Note that both the inner product and the norm in (3.22) are well defined. The norm $\left\| \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\|_2$ exists because $\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \in \mathcal{L}_2$ by assumption. With $\mathsf{r} = \tilde{\mathsf{r}}_{\boldsymbol{b}_0,\boldsymbol{\theta}_0} + \mathsf{v}$, the inner product in (3.22) can be decomposed into a deterministic and a stochastic component:

$$\operatorname{Re}\left\langle \mathsf{r}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle = \operatorname{Re}\left\langle \tilde{\mathsf{r}}_{\boldsymbol{b}_0,\boldsymbol{\theta}_0}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle + \operatorname{Re}\left\langle \mathsf{v}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle. \tag{3.24}$$

The existence of the inner product $\left\langle \tilde{\mathsf{r}}_{\boldsymbol{b}_0,\boldsymbol{\theta}_0}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle$ again follows from the finite-energy-assumption, as can be shown with the Cauchy-Schwarz inequality [70, Section 3.3]

$$|\langle \mathsf{x}, \mathsf{y} \rangle| \leq \|\mathsf{x}\|_2 \, \|\mathsf{y}\|_2, \quad \mathsf{x}, \mathsf{y} \in \mathcal{L}_2. \tag{3.25}$$

The stochastic component $\operatorname{Re}\left\langle \mathsf{v}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle$ is a Gaussian random variable

$$\begin{aligned} \operatorname{Re}\left\langle \mathsf{v}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle &= \left\langle \operatorname{Re}(\mathsf{v}), \operatorname{Re}(\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}) \right\rangle + \left\langle \operatorname{Im}(\mathsf{v}), \operatorname{Im}(\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}) \right\rangle \\ &\sim \mathcal{N}\left(0, \tfrac{N_0}{2} \left\| \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\|_2^2 \right) \end{aligned} \tag{3.26}$$

which follows from [70, Proposition 25.15.2 (i)] because $\tilde{\mathsf{r}}$ is integrable and bandlimited to $B_{\mathsf{r}} < W$, and because $\operatorname{Re}(\mathsf{v})$ and $\operatorname{Im}(\mathsf{v})$ are uncorrelated. In summary, the exponent in (3.22) is a Gaussian random variable with finite mean and variance:

$$\frac{\operatorname{Re}\left\langle \mathsf{r}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle}{N_0/2} - \frac{\left\| \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\|_2^2}{N_0} \sim \mathcal{N}\left(\frac{\operatorname{Re}\left\langle \tilde{\mathsf{r}}_{\boldsymbol{b}_0,\boldsymbol{\theta}_0}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\rangle}{N_0/2} - \frac{\left\| \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\|_2^2}{N_0}, \frac{\left\| \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}} \right\|_2^2}{N_0/2} \right). \tag{3.27}$$

In order to prove (3.22), we first consider a random vector which contains finitely many samples of r, taken at a sufficiently high rate to avoid aliasing, and then take the

[5] Strictly speaking, $\|\cdot\|_2$ is not a norm, because $\|\mathsf{x}\|_2 = 0$ does not imply $\mathsf{x} = 0$. It merely implies that x is *indistinguishable* from the all-zero signal, meaning that the set $\{t \in \mathbb{R} \,|\, \mathsf{x}(t) \neq 0\}$ is of Lebesgue measure zero. However, such mathematical rigor is not needed for our purpose.

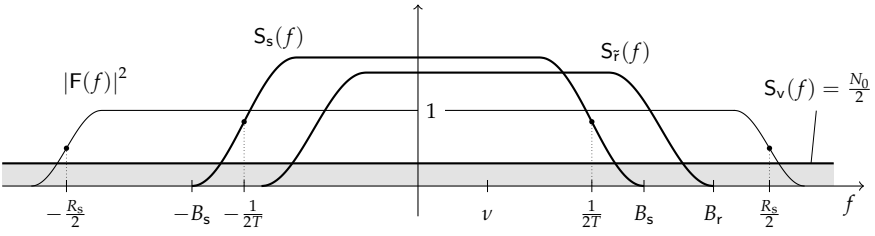


Figure 3.4: Spectral density of the transmitted signal s, the noise-free received signal $\tilde{\mathsf{r}}$ assuming a frequency shift of ν such that $B_\mathsf{r} = B_\mathsf{s} + \nu$, and the Gaussian noise process v; along with the spectrum of a possible anti-aliasing filter f for the sampling rate R_s

limit of an infinite time horizon. To that end, let f be an anti-aliasing filter with the idealized frequency response[6]

$$\mathsf{F}(f) = \begin{cases} 1 & \text{if } |f| < R_\mathrm{s}/2 \\ 0 & \text{otherwise} \end{cases} \tag{3.28}$$

where $B_\mathsf{r} < R_\mathrm{s}/2 < W$. Let $\mathsf{r}_\mathsf{f} = \mathsf{r} \star \mathsf{f}$ denote the filtered received signal, which is subsequently sampled at rate $R_\mathrm{s} = 1/T_\mathrm{s}$. The useful signal is not affected by the filter because $B_\mathsf{r} < R_\mathrm{s}/2$, and thus

$$\mathsf{r}_\mathsf{f}(t) = \tilde{\mathsf{r}}_{\boldsymbol{b}_0,\boldsymbol{\theta}_0}(t) + \mathsf{v}_\mathsf{f}(t) \tag{3.29}$$

with $\mathsf{v}_\mathsf{f} = \mathsf{v} \star \mathsf{f}$. The samples $\mathsf{v}_\mathsf{f}(kT_\mathrm{s})$ form a discrete-time white proper Gaussian noise process with

$$\mathsf{v}_\mathsf{f}(kT_\mathrm{s}) \sim \mathcal{CN}(0, N_0 R_\mathrm{s}), \quad k \in \mathbb{Z} \tag{3.30}$$

which follows from [70, Proposition 25.15.2 (i)] because $\|\mathsf{f}\|_2^2 = R_\mathrm{s}$, as can be shown via Parseval's theorem [70, Theorem 6.2.9] and (3.28).

With the finite sample sequence of length $(2K+1)$

$$\boldsymbol{r}^{(K)} \triangleq \big(\mathsf{r}_\mathsf{f}(-KT_\mathrm{s}), \ldots, \mathsf{r}_\mathsf{f}(0), \ldots, \mathsf{r}_\mathsf{f}(KT_\mathrm{s})\big) \tag{3.31}$$

we can now write

$$\ell_\mathsf{r}(\boldsymbol{b},\boldsymbol{\theta}) = \frac{p(\boldsymbol{b},\boldsymbol{\theta} \,|\, \mathsf{r})}{p(\boldsymbol{b},\boldsymbol{\theta})} = \lim_{K\to\infty} \frac{p(\boldsymbol{r}^{(K)} \,|\, \boldsymbol{b},\boldsymbol{\theta})}{p(\boldsymbol{r}^{(K)})}$$

$$= \lim_{K\to\infty} \frac{p(\boldsymbol{r}^{(K)} \,|\, \boldsymbol{b},\boldsymbol{\theta})}{\sum_{\boldsymbol{b}'} \int_\Theta p(\boldsymbol{b}')\, p(\boldsymbol{\theta}')\, p(\boldsymbol{r}^{(K)} \,|\, \boldsymbol{b}',\boldsymbol{\theta}')\, \mathrm{d}\boldsymbol{\theta}'}$$

[6] Being an ideal low-pass is an overly strict requirement, but it simplifies the following derivation somewhat. See [89] for sufficient and more relaxed conditions that the anti-aliasing filter should fulfill.

$$
\begin{aligned}
&= \lim_{K\to\infty} \frac{\frac{1}{(\pi N_0 R_\mathrm{s})^{2K+1}} \exp\left(-\frac{\sum_{k=-K}^{K} |\mathsf{r}_\mathsf{f}(kT_\mathrm{s}) - \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}(kT_\mathrm{s})|^2}{N_0 R_\mathrm{s}}\right)}{\sum_{\boldsymbol{b}'} \int_{\boldsymbol{\Theta}} p(\boldsymbol{b}')\, p(\boldsymbol{\theta}') \frac{1}{(\pi N_0 R_\mathrm{s})^{2K+1}} \exp\left(-\frac{\sum_{k=-K}^{K} |\mathsf{r}_\mathsf{f}(kT_\mathrm{s}) - \tilde{\mathsf{r}}_{\boldsymbol{b}',\boldsymbol{\theta}'}(kT_\mathrm{s})|^2}{N_0 R_\mathrm{s}}\right) \mathrm{d}\boldsymbol{\theta}'} \\
&\overset{(a)}{=} \frac{\exp\left(\frac{T_\mathrm{s}}{N_0} \sum_{k=-\infty}^{\infty} \left[2\,\mathrm{Re}\big(\mathsf{r}_\mathsf{f}(kT_\mathrm{s})\, \tilde{\mathsf{r}}^*_{\boldsymbol{b},\boldsymbol{\theta}}(kT_\mathrm{s})\big) - |\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}(kT_\mathrm{s})|^2\right]\right)}{\sum_{\boldsymbol{b}'} \int_{\boldsymbol{\Theta}} p(\boldsymbol{b}') p(\boldsymbol{\theta}') \exp\left(\frac{T_\mathrm{s}}{N_0} \sum_{k=-\infty}^{\infty} \left[2\,\mathrm{Re}\big(\mathsf{r}_\mathsf{f}(kT_\mathrm{s})\, \tilde{\mathsf{r}}^*_{\boldsymbol{b}',\boldsymbol{\theta}'}(kT_\mathrm{s})\big) - |\tilde{\mathsf{r}}_{\boldsymbol{b}',\boldsymbol{\theta}'}(kT_\mathrm{s})|^2\right]\right) \mathrm{d}\boldsymbol{\theta}'} \\
&\overset{(b)}{=} \frac{\exp\left(\frac{1}{N_0} \int_{-\infty}^{\infty} \left[2\,\mathrm{Re}\big(\mathsf{r}_\mathsf{f}(t)\, \tilde{\mathsf{r}}^*_{\boldsymbol{b},\boldsymbol{\theta}}(t)\big) - |\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}(t)|^2\right] \mathrm{d}t\right)}{\sum_{\boldsymbol{b}'} \int_{\boldsymbol{\Theta}} p(\boldsymbol{b}')\, p(\boldsymbol{\theta}') \exp\left(\frac{1}{N_0} \int_{-\infty}^{\infty} \left[2\,\mathrm{Re}\big(\mathsf{r}_\mathsf{f}(t)\, \tilde{\mathsf{r}}^*_{\boldsymbol{b}',\boldsymbol{\theta}'}(t)\big) - |\tilde{\mathsf{r}}_{\boldsymbol{b}',\boldsymbol{\theta}'}(t)|^2\right] \mathrm{d}t\right) \mathrm{d}\boldsymbol{\theta}'} \\
&\overset{(c)}{=} \frac{\exp\left(\frac{1}{N_0} \left[2\,\mathrm{Re}\left\langle \mathsf{r}, \tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}\right\rangle - \|\tilde{\mathsf{r}}_{\boldsymbol{b},\boldsymbol{\theta}}\|_2^2\right]\right)}{\sum_{\boldsymbol{b}'} \int_{\boldsymbol{\Theta}} p(\boldsymbol{b}')\, p(\boldsymbol{\theta}') \exp\left(\frac{1}{N_0} \left[2\,\mathrm{Re}\langle \mathsf{r}, \tilde{\mathsf{r}}_{\boldsymbol{b}',\boldsymbol{\theta}'}\rangle - \|\tilde{\mathsf{r}}_{\boldsymbol{b}',\boldsymbol{\theta}'}\|_2^2\right]\right) \mathrm{d}\boldsymbol{\theta}'}.
\end{aligned}
\tag{3.32}
$$

Here, (a) cancels factors that are independent of $\boldsymbol{b}'$ and $\boldsymbol{\theta}'$, and (b) follows from the sampling theorem [70, Section 8.4] because the signals are bandlimited to $R_\mathrm{s}/2$ due to the anti-aliasing filter. Step (c) uses the definition of $\langle\cdot,\cdot\rangle$ and $\|\cdot\|_2$, and exploits that $\tilde{\mathsf{r}}$ is bandlimited to $B_\mathsf{r} < R_\mathrm{s}/2$, so that $\mathsf{r}_\mathsf{f} = \mathsf{r} \star \mathsf{f}$ can be replaced by r without affecting the inner product. Finally, the claim (3.22) follows because the denominator in (3.32) is independent of $\boldsymbol{b}$ and $\boldsymbol{\theta}$, and hence is just a normalization constant. □

A similar calculation, albeit only briefly sketched, has been given in the appendix of [89]. That paper, however, is somewhat inconsequential in that it acknowledges the nonexistence of the likelihood function, but nonetheless advocates the maximum likelihood criterion for receiver design.

Maximum likelihood is a concept from orthodox statistics, which is concerned with the invention of estimators, using a set of *ad hoc* criteria like ML. The central object of orthodox statistics is the likelihood function $p(\boldsymbol{y};\boldsymbol{x})$, which describes the hypothetical distribution of $\boldsymbol{y}$ that would be obtained if the underlying experiment was repeated an infinite number of times with the same deterministic but unknown parameter $\boldsymbol{x}$.

In contrast, we will consequently stick to the Bayesian school in this thesis. Bayesians treat all unknown quantities as random variables, and update their distribution $p(\boldsymbol{x})$ to $p(\boldsymbol{x}\,|\,\boldsymbol{y})$ in the light of new data $\boldsymbol{y}$, using Bayes' theorem $p(\boldsymbol{x}\,|\,\boldsymbol{y}) = p(\boldsymbol{x})\ell_{\boldsymbol{y}}(\boldsymbol{x})$. Among the many advantages of Bayesian statistics are the possibilities to exploit non-uniform prior knowledge, and to compute soft information about $\boldsymbol{x}$ in terms of probability distributions, which can be used as prior knowledge by the next processing step.

Since exact Bayesian inference is infeasible for most practical problems, several approximations have been proposed in the literature. In the following, we will apply the very general divergence minimization (DM) approach, which is discussed in detail in Appendix B, to derive iterative receiver structures for a variety of scenarios.

3.3 Hard Parameter Estimation via Divergence Minimization

In order to demonstrate that previously proposed ML-like[7] code-aided synchronization algorithms are contained as special cases in the DM framework, we start the discussion by re-deriving the results from [100] and [114]. From those papers, we adopt the simple channel model, which essentially consists of an AWGN channel with unknown timing and phase/frequency shifts, and which could for example represent a satellite link.

To derive ML-like estimators from the divergence minimization approach, we constrain the beliefs of the parameters to Dirac distributions, which leads to MAP-like estimators as shown in Section B.4.3 on page 252, and select uniform prior distributions, which makes MAP and ML formally identical. In later sections, soft estimators will be derived from the same basic principles, simply by removing these constraints.

3.3.1 System Model

A message $\boldsymbol{b} \in \mathbb{F}^{N_\mathrm{b}}$ is encoded[8] to a codeword $\boldsymbol{c} \in \mathbb{F}^{N_\mathrm{c}}$, which is split into N_d subvectors $\boldsymbol{c}_n$ of length K. The $\boldsymbol{c}_n$ are transformed to complex symbols according to some M-ary mapping scheme, where $M = 2^K$, yielding a block $\boldsymbol{x}_\mathcal{D}$ of N_d data symbols. Some training data, consisting of N_p pilot symbols $\boldsymbol{x}_\mathcal{P}$, is multiplexed into the symbol stream, which results in a block $\boldsymbol{x}$ of length $N = N_\mathrm{d} + N_\mathrm{p}$.

The transmitted signal s is generated by linear modulation

$$\mathsf{s}(t) = \sum_{n=0}^{N-1} x_n \, \mathsf{g}(t - nT) \tag{3.33}$$

using a transmit pulse g whose properties are summarized in the text box on the facing page.

The channel introduces a delay τ_0, a frequency shift ν_0, a phase offset ϑ_0, a multiplicative gain g_0, and additive Gaussian noise v with power spectral density N_0. The received signal r is then given as

$$\begin{aligned} \mathsf{r}(t) &= g_0 \, \mathsf{s}(t - \tau_0) \, \mathrm{e}^{\mathrm{j}(2\pi\nu_0 t + \vartheta_0)} + \mathsf{v}(t) \\ &= g_0 \sum_{n=0}^{N-1} x_n \, \mathsf{g}(t - nT - \tau_0) \, \mathrm{e}^{\mathrm{j}(2\pi\nu_0 t + \vartheta_0)} + \mathsf{v}(t). \end{aligned} \tag{3.34}$$

A block diagram of the system model is shown in Figure 3.5.

[7] The term *ML-like* indicates that the proposed estimators are derived from the maximum likelihood criterion, but only give approximate results because the presence of the unknown data sequence makes the ML problem intractable.

[8] As explained in Chapter 2, the coding scheme $\mathcal{C}$ should be understood as an *effective* encoder, which contains the channel encoder and also the bit interleaver used in BICM transmitters.

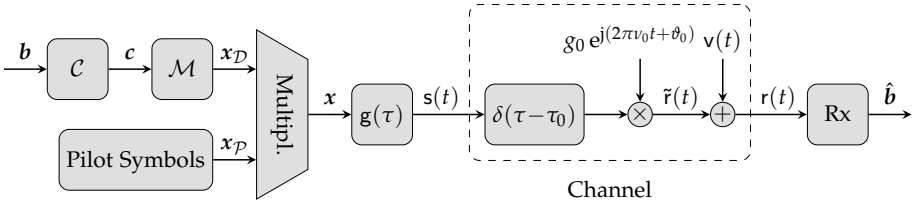


Figure 3.5: Block diagram of a QAM transmitter, AWGN channel, and a generic receiver. The channel is characterized by four unknown parameters: the gain g_0, the time offset τ_0, the frequency offset ν_0, and the phase offset ϑ_0

Properties of the Transmit Pulse

We require the integrable and energy-limited transmit pulse $\mathsf{g} \in \mathcal{L}_1 \cap \mathcal{L}_2$ to be a Nyquist pulse with respect to the symbol duration T, normalized to unit energy, i. e., $\|\mathsf{g}\|_2^2 = 1$. The self-similarity function of g can be compactly written as

$$\mathsf{R}_\mathsf{g}(nT) \triangleq \int_{-\infty}^{\infty} \mathsf{g}(\tau + nT)\, \mathsf{g}^*(\tau)\, \mathrm{d}\tau = \mathbb{1}\left[n = 0\right], \quad n \in \mathbb{Z}. \tag{3.35}$$

Furthermore we assume g to be a baseband pulse whose spectrum is centered around $f = 0$, so in conjunction with the normalization $\|\mathsf{g}\|_2^2 = 1 = \|\mathsf{G}\|_2^2$ (Parseval's theorem), we have

$$\mathsf{G}(f)\Big|_{f=0} = \sqrt{T} = \int_{-\infty}^{\infty} \mathsf{g}(\tau)\, \mathrm{d}\tau. \tag{3.36}$$

Note that the transmit pulse has the somewhat peculiar physical unit $1/\sqrt{\mathsf{s}}$. This is a consequence of the unit-energy normalization, whose rationale is that the energy of a modulated pulse $\mathsf{s}(t) = x\,\mathsf{g}(t)$ is then simply described by $|x|^2$. The physical unit of x is $\sqrt{\mathsf{J}}$, and the unit of the signal $\mathsf{s}(t)$ is $\sqrt{\mathsf{J}/\mathsf{s}} = \sqrt{\mathsf{W}}$.

Collecting the channel parameters in the vector

$$\boldsymbol{\theta} = (g_0, \tau_0, \nu_0, \vartheta_0) \in \boldsymbol{\Theta} \tag{3.37}$$

and assuming that they are independent of the transmitted signal, the joint posterior of all unknown quantities can be factorized as

$$p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x}_\mathcal{D}, \boldsymbol{\theta} \,|\, \mathsf{r}) = p(\boldsymbol{b})\, p(\boldsymbol{c} \,|\, \boldsymbol{b})\, p(\boldsymbol{x}_\mathcal{D} \,|\, \boldsymbol{c})\, p(\boldsymbol{\theta})\, \ell_\mathsf{r}(\boldsymbol{x}_\mathcal{D}, \boldsymbol{\theta}) \tag{3.38}$$

where $\ell_\mathsf{r}(\boldsymbol{x}_\mathcal{D}, \boldsymbol{\theta}) = p(\boldsymbol{x}_\mathcal{D}, \boldsymbol{\theta} \,|\, \mathsf{r}) / p(\boldsymbol{x}_\mathcal{D}, \boldsymbol{\theta})$ as defined in the previous section. A factor graph of (3.38) is shown in Figure 3.6.

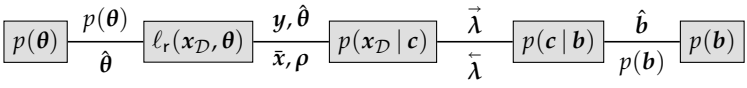


Figure 3.6: Factor graph of the joint posterior distribution (3.38). Parameters above the edges are passed forward, from left to right, while parameters below the edges are passed backward, from right to left

The prior distribution $p(\boldsymbol{b})$ of the transmitted message is assumed to be uniform throughout this thesis. The factors $p(\boldsymbol{c} \mid \boldsymbol{b})$ and $p(\boldsymbol{x}_{\mathcal{D}} \mid \boldsymbol{c})$ represent the deterministic encoding and mapping operations, respectively, and can be written as

$$p(\boldsymbol{c} \mid \boldsymbol{b}) = \mathbb{1}\left[\boldsymbol{c} = \mathcal{C}(\boldsymbol{b})\right] \tag{3.39}$$

$$p(\boldsymbol{x}_{\mathcal{D}} \mid \boldsymbol{c}) = \prod_{n \in \mathcal{D}} p(x_n \mid \boldsymbol{c}_n) = \prod_{n \in \mathcal{D}} \mathbb{1}\left[x_n = \mathcal{M}(\boldsymbol{c}_n)\right]. \tag{3.40}$$

The most interesting factor is the likelihood function, which has been derived in (3.22)

$$\ell_{\mathsf{r}}(\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}) \propto \exp\left(\frac{\operatorname{Re}\langle \mathsf{r}, \tilde{\mathsf{r}}_{\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}} \rangle}{N_0/2} - \frac{\|\tilde{\mathsf{r}}_{\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}}\|_2^2}{N_0}\right). \tag{3.41}$$

In our system model, the squared norm of $\tilde{\mathsf{r}}$ is[9]

$$\begin{aligned}
\|\tilde{\mathsf{r}}_{\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}}\|_2^2 &= \int |\tilde{\mathsf{r}}_{\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}}(t)|^2 \, \mathrm{d}t \\
&= \int \Big| g_0 \sum_{n=0}^{N-1} x_n \, \mathsf{g}(t - nT - \tau_0) \, \mathrm{e}^{\mathrm{j}(2\pi\nu_0 t + \vartheta_0)} \Big|^2 \mathrm{d}t \\
&= g_0^2 \sum_{n=0}^{N-1} \sum_{n'=0}^{N-1} x_n x_{n'}^* \int \mathsf{g}(t - nT - \tau_0) \, \mathsf{g}^*(t - n'T - \tau_0) \, \mathrm{d}t \\
&\overset{(a)}{=} g_0^2 \sum_{n=0}^{N-1} |x_n|^2
\end{aligned} \tag{3.42}$$

where (a) uses (3.35) in the text box. The inner product can similarly be computed as

$$\begin{aligned}
\langle \mathsf{r}, \tilde{\mathsf{r}}_{\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}} \rangle &= \int \mathsf{r}(t) \, \tilde{\mathsf{r}}^*_{\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}}(t) \, \mathrm{d}t \\
&= \int \mathsf{r}(t) \, g_0 \sum_{n=0}^{N-1} x_n^* \, \mathsf{g}^*(t - nT - \tau_0) \, \mathrm{e}^{-\mathrm{j}(2\pi\nu_0 t + \vartheta_0)} \, \mathrm{d}t \\
&= g_0 \, \mathrm{e}^{-\mathrm{j}\vartheta_0} \sum_{n=0}^{N-1} x_n^* \int \mathsf{r}(t) \, \mathsf{g}^*(t - nT - \tau_0) \, \mathrm{e}^{-\mathrm{j}2\pi\nu_0 t} \, \mathrm{d}t
\end{aligned} \tag{3.43}$$

[9] All integrals over time t range from $-\infty$ to ∞, so for simplicity we drop the explicit bounds.

and thus

$$\ell_{\mathsf{r}}(\boldsymbol{x}_{\mathcal{D}},\boldsymbol{\theta}) \propto \exp\left(\frac{2g_0}{N_0}\,\mathrm{Re}\left(\mathrm{e}^{-\mathrm{j}\vartheta_0}\sum_{n=0}^{N-1} x_n^* \int \mathsf{r}(t)\,\mathsf{g}^*(t-nT-\tau_0)\,\mathrm{e}^{-\mathrm{j}2\pi\nu_0 t}\,\mathrm{d}t\right) - \frac{g_0^2}{N_0}\sum_{n=0}^{N-1}|x_n|^2\right). \tag{3.44}$$

The remaining factor is $p(\boldsymbol{\theta})$, the prior of the channel parameters. As explained at the beginning of this section, we assume a uniform prior in order to obtain ML-like estimators. We will revisit this choice in later sections.

In order to apply the DM approach, we still need to define the set $\mathcal{Q}$ from which the auxiliary distribution q is selected, and choose for each message whether it is computed by inclusive or exclusive divergence minimization. First of all, the factor graph in Figure 3.6 implies the factorization $q(\boldsymbol{b},\boldsymbol{c},\boldsymbol{x}_{\mathcal{D}},\boldsymbol{\theta}) = q(\boldsymbol{b})\,q(\boldsymbol{c})\,q(\boldsymbol{x}_{\mathcal{D}})\,q(\boldsymbol{\theta})$. However, to make the receiver algorithm tractable, we break the factorization of the discrete distributions further down to the scalar level, i. e., $q(\boldsymbol{b}) = \prod_{i=0}^{N_\mathrm{b}-1} q(b_i)$, and similarly for $q(\boldsymbol{c})$ and $q(\boldsymbol{x}_{\mathcal{D}})$. As explained in Section 2.3.2, this fine-grained factorization could also have been expressed explicitly in the graphical model by drawing individual variable nodes for each bit and data symbol, and in fact this is done in many papers on turbo or LDPC decoding. Our reason to draw the graph in the compact form as in Figure 3.6, and to impose the factorizations of $q(\boldsymbol{b})$, $q(\boldsymbol{c})$ and $q(\boldsymbol{x}_{\mathcal{D}})$ via implicit constraints, is to maintain a high-level view on the probabilistic network, and to avoid getting entangled in unnecessary details.

As mentioned at the beginning of this section, the belief $q(\boldsymbol{\theta})$ of the synchronization parameters will be constrained to a Dirac distribution in order to obtain hard parameter estimates. A derivation of the generic MAP-like hard estimator in the context of the DM framework is given in Section B.4.3 on page 252, where it is in particular shown that MAP-like hard estimation is based on *exclusive* divergence minimization.

In the discrete part of the belief network, however, we are able to use *inclusive* divergence minimization, which leads to expectation propagation because of the projections that are necessary due to the factorization constraints. Message passing in the discrete part thus boils down to the standard BICM-ID receiver, as discussed in Section 2.3.

In the next two subsections, we will derive the iterative receiver structure, comprising the code-aided parameter estimator and the conventional BICM-ID algorithm. We begin with the forward pass through the receiver, meaning the left-to-right information flow in Figure 3.6, from the parameter estimator through the demapper and decoder towards the sink. The following subsection 3.3.3 then derives the backward pass.

3.3.2 Forward Pass

Parameter Estimation

A straightforward application of (B.84) leads to the generic parameter estimate

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_{q(\boldsymbol{x}_\mathcal{D})}[\log \ell_\mathsf{r}(\boldsymbol{x}_\mathcal{D}, \boldsymbol{\theta})] + \log p(\boldsymbol{\theta}). \tag{3.45}$$

As we assume a uniform prior distribution, the term $\log p(\boldsymbol{\theta})$ is constant and will be ignored in the following. With the likelihood function (3.44), we obtain the following optimization problem:

$$\begin{aligned}(\hat{g}_0, \hat{\tau}_0, \hat{\nu}_0, \hat{\vartheta}_0) &= \arg\max_{g_0,\tau_0,\nu_0,\vartheta_0} \mathbb{E}_{q(\boldsymbol{x}_\mathcal{D})} \left[\frac{2g_0}{N_0} \operatorname{Re}\left(\mathrm{e}^{-\mathrm{j}\vartheta_0} \sum_{n=0}^{N-1} x_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \tau_0)\, \mathrm{e}^{-\mathrm{j}2\pi\nu_0 t}\, \mathrm{d}t \right) \right. \\ &\qquad \left. - \frac{g_0^2}{N_0} \sum_{n=0}^{N-1} |x_n|^2 \right] \\ &= \arg\max_{g_0,\tau_0,\nu_0,\vartheta_0} \frac{2g_0}{N_0} \operatorname{Re}\left(\mathrm{e}^{-\mathrm{j}\vartheta_0} \sum_{n=0}^{N-1} \bar{x}_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \tau_0)\, \mathrm{e}^{-\mathrm{j}2\pi\nu_0 t}\, \mathrm{d}t \right) \\ &\qquad - \frac{g_0^2}{N_0} \sum_{n=0}^{N-1} \rho_n. \end{aligned} \tag{3.46}$$

For data symbols ($n \in \mathcal{D}$), $\bar{x}_n \triangleq \mathbb{E}_{q(x_n)}[x_n]$ and $\rho_n \triangleq \mathbb{E}_{q(x_n)}[|x_n|^2]$ are the first two raw moments of the data symbols, obtained from the current belief $q(x_n)$. For pilot symbols ($n \in \mathcal{P}$), we simply have $\bar{x}_n = x_n$ and $\rho_n = |x_n|^2$.

How are the $q(x_n)$ being computed? In the initial iteration, no information about the data symbols is available, so the $q(x_n)$ are uniform, $\bar{x}_n = 0$, and $\rho_n = E_\mathrm{s}$, where $E_\mathrm{s} \triangleq |\mathcal{X}|^{-1} \sum_{x\in\mathcal{X}} |x|^2$ is the average symbol energy. The code-aided parameter estimation then automatically decays to pilot-aided estimation. In later iterations, non-uniform $q(x_n)$ are obtained based on feedback from later stages of the receiver. This procedure is discussed in more detail in Section 3.3.3 on page 51.

The second term in (3.46) only depends on g_0, which furthermore is non-negative, so

$$(\hat{\tau}_0, \hat{\nu}_0, \hat{\vartheta}_0) = \arg\max_{\tau_0,\nu_0,\vartheta_0} \operatorname{Re}\left(\mathrm{e}^{-\mathrm{j}\vartheta_0} \sum_{n=0}^{N-1} \bar{x}_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \tau_0)\, \mathrm{e}^{-\mathrm{j}2\pi\nu_0 t}\, \mathrm{d}t \right). \tag{3.47}$$

Once $\hat{\tau}_0$ and $\hat{\nu}_0$ are obtained, maximization over ϑ_0 is simple,

$$\hat{\vartheta}_0 = \arg\left(\sum_{n=0}^{N-1} \bar{x}_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \hat{\tau}_0)\, \mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\, \mathrm{d}t \right) \tag{3.48}$$

because then the real part of the term in parentheses coincides with its modulus. The estimates of time and frequency offset are thus

$$(\hat{\tau}_0, \hat{\nu}_0) = \arg\max_{\tau_0, \nu_0} \left| \sum_{n=0}^{N-1} \bar{x}_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \tau_0)\, \mathrm{e}^{-\mathrm{j}2\pi\nu_0 t}\, \mathrm{d}t \right|. \tag{3.49}$$

Finally, the estimate of the channel gain is

$$\begin{aligned} \hat{g}_0 &= \arg\max_{g_0} \frac{2g_0}{N_0} \left| \sum_{n=0}^{N-1} \bar{x}_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \hat{\tau}_0)\, \mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\, \mathrm{d}t \right| - \frac{g_0^2}{N_0} \sum_{n=0}^{N-1} \rho_n \\ &= \frac{\left| \sum_{n=0}^{N-1} \bar{x}_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \hat{\tau}_0)\, \mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\, \mathrm{d}t \right|}{\sum_{n=0}^{N-1} \rho_n} \end{aligned} \tag{3.50}$$

which is easily obtained by setting the derivative with respect to g_0 to zero. Note that, if we combine the channel gain g_0 and the phase offset ϑ_0 into a complex channel weight $h_0 \triangleq g_0\, \mathrm{e}^{\mathrm{j}\vartheta_0}$, its estimate can be written by merging (3.48) and (3.50) as

$$\hat{h}_0 = \frac{\sum_{n=0}^{N-1} \bar{x}_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \hat{\tau}_0)\, \mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\, \mathrm{d}t}{\sum_{n=0}^{N-1} \rho_n}. \tag{3.51}$$

The joint timing and frequency estimator does unfortunately not have such a nice solution. However, (3.49) is not a novel result; strategies for solving the optimization problem have been studied quite extensively in the literature (c. f. [88] and references therein), so we will not discuss this topic any further.

Matched Filtering

The problem of parameter estimation with soft symbol input that we have considered so far corresponds, in the terminology of message passing, to the computation of the message from the factor node $\ell_{\mathsf{r}}(\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta})$ towards $p(\boldsymbol{\theta})$. Now, let us have a look at the message towards the factor node $p(\boldsymbol{x}_{\mathcal{D}} \,|\, \boldsymbol{c})$, i. e., into the discrete part of the receiver. Up to a multiplicative constant, it is given as

$$\begin{aligned} &m_{\mathrm{mf}\to\mathrm{dem}}(\boldsymbol{x}_{\mathcal{D}}) \triangleq m_{\ell_{\mathsf{r}}(\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta}) \to p(\boldsymbol{x}_{\mathcal{D}} \,|\, \boldsymbol{c})}(\boldsymbol{x}_{\mathcal{D}}) \propto \mathbb{E}_{q(\boldsymbol{\theta})}[\ell_{\mathsf{r}}(\boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\theta})] \\ &\overset{(a)}{=} \exp\left(\frac{2\hat{g}_0}{N_0} \operatorname{Re}\left(\mathrm{e}^{-\mathrm{j}\hat{\vartheta}_0} \sum_{n=0}^{N-1} x_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \hat{\tau}_0)\, \mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\, \mathrm{d}t \right) - \frac{\hat{g}_0^2}{N_0} \sum_{n=0}^{N-1} |x_n|^2 \right) \\ &\overset{(b)}{=} \prod_{n=0}^{N-1} \exp\left(\frac{2\hat{g}_0}{N_0} \operatorname{Re}\left(\mathrm{e}^{-\mathrm{j}\hat{\vartheta}_0} x_n^* \int \mathsf{r}(t)\, \mathsf{g}^*(t - nT - \hat{\tau}_0)\, \mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\, \mathrm{d}t \right) - \frac{\hat{g}_0^2}{N_0} |x_n|^2 \right) \\ &\propto \prod_{n=0}^{N-1} m_{\mathrm{mf}\to\mathrm{dem}}(x_n). \end{aligned} \tag{3.52}$$

Here, (a) computes the trivial expectation over $q(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$, and (b) pulls the sum over n out of the exponential function, showing that $m_{\mathrm{mf}\to\mathrm{dem}}(\boldsymbol{x}_{\mathcal{D}})$ decomposes into the product of messages towards the individual data symbols.

Because of the independence of the forward messages, it suffices to consider a single one. Again disregarding constant factors, the message towards x_n becomes

$$\begin{aligned} m_{\mathrm{mf}\to\mathrm{dem}}(x_n) &\propto \exp\left(\frac{2\hat{g}_0}{N_0}\,\mathrm{Re}\left(\mathrm{e}^{-\mathrm{j}\hat{\vartheta}_0} x_n^* \int \mathsf{r}(t)\,\mathsf{g}^*(t-nT-\hat{\tau}_0)\,\mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\,\mathrm{d}t\right) - \frac{\hat{g}_0^2}{N_0}\,|x_n|^2\right) \\ &= \exp\left(-\frac{\hat{g}_0^2}{N_0}\left(|x_n|^2 - 2\,\mathrm{Re}\left(x_n^*\,\frac{\int \mathsf{r}(t)\,\mathsf{g}^*(t-nT-\hat{\tau}_0)\,\mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\,\mathrm{d}t}{\hat{g}_0\,\mathrm{e}^{\mathrm{j}\hat{\vartheta}_0}}\right)\right)\right) \\ &\propto \mathcal{CN}\left(x_n \,\middle|\, \frac{\int \mathsf{r}(t)\,\mathsf{g}^*(t-nT-\hat{\tau}_0)\,\mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\,\mathrm{d}t}{\hat{g}_0\,\mathrm{e}^{\mathrm{j}\hat{\vartheta}_0}}, \frac{N_0}{\hat{g}_0^2}\right) \end{aligned} \tag{3.53}$$

where (3.53) means that for each $x_n \in \mathcal{X}_n$ ($\mathcal{X}_n$ being the discrete modulation alphabet for the nth symbol)

$$m_{\mathrm{mf}\to\mathrm{dem}}(x_n) \propto \exp\left(-\frac{\hat{g}_0^2}{N_0}\left|x_n - \frac{\int \mathsf{r}(t)\,\mathsf{g}^*(t-nT-\hat{\tau}_0)\,\mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\,\mathrm{d}t}{\hat{g}_0\,\mathrm{e}^{\mathrm{j}\hat{\vartheta}_0}}\right|^2\right). \tag{3.54}$$

In the interest of compact notation, it is reasonable to define

$$y_n(\hat{\tau}_0, \hat{\nu}_0) \triangleq \int \mathsf{r}(t)\,\mathsf{g}^*(t-nT-\hat{\tau}_0)\,\mathrm{e}^{-\mathrm{j}2\pi\hat{\nu}_0 t}\,\mathrm{d}t. \tag{3.55}$$

Computing $y_n(\hat{\tau}_0, \hat{\nu}_0)$ from r is commonly called *matched filtering*. It can be implemented as the concatenation of a multiplication of $\mathsf{r}(t)$ with $\exp(-\mathrm{j}2\pi\hat{\nu}_0 t)$ (frequency correction), filtering with $\mathsf{g}^*(-\tau)$, and sampling the result at the time instant $t = nT + \hat{\tau}_0$. We remark that in practice, the timing and frequency correction would rather be implemented in the digital domain, after analog prefiltering and A/D conversion with a sufficiently high sampling rate. However, such implementation aspects are out of scope of this thesis; the interested reader is referred to [88,89].

With the matched filter output $y_n(\hat{\tau}_0, \hat{\nu}_0)$, the forward message towards x_n becomes

$$m_{\mathrm{mf}\to\mathrm{dem}}(x_n) \propto \mathcal{CN}\left(x_n \,\middle|\, \frac{y_n(\hat{\tau}_0, \hat{\nu}_0)}{\hat{g}_0\,\mathrm{e}^{\mathrm{j}\hat{\vartheta}_0}}, \frac{N_0}{\hat{g}_0^2}\right) = \mathcal{CN}\left(x_n \,\middle|\, \frac{y_n(\hat{\tau}_0, \hat{\nu}_0)}{\hat{h}_0}, \frac{N_0}{|\hat{h}_0|^2}\right). \tag{3.56}$$

Demapping and Decoding

The demapper and decoder proceed as derived in Section 2.3, with the sole difference that the input to the demapper, formerly given directly as the channel output as in (2.16), is now replaced by the message (3.56) from the matched filter.

The demapper produces a sequence of intrinsic L-values that serve as input to the soft-in soft-out channel decoder, which in turn computes new beliefs of all code bits c_i and information bits b_i. If the decoder works itself iteratively, the question of scheduling arises. Two canonical choices are as follows.

- On every invocation of the decoder, its internal state is cleared (i. e., the internal messages are reset to uniform distributions), and the decoder is run until convergence. This leads to a doubly-iterative receiver.

- The decoder only runs for a single iteration, keeping its internal state. In this case, the synchronization and decoding iterations are merged, and the receiver remains singly-iterative.

In [99,100], an iterative receiver using the first schedule has been derived by means of the EM algorithm,[10] while the second schedule is proposed as an *ad hoc* approximation with the aim of reducing the overall complexity. Simulation results presented in [99] support the claim that both schedules yield similar performance after convergence of the total receiver, even though the second schedule might require more synchronization iterations. Note that both choices can naturally be obtained from the divergence minimization approach. In contrast to the EM-based derivation, there is no necessity to leave the ground of the theoretical framework by introducing additional approximations.

The simulation results that are presented later have been obtained with the former, doubly-iterative schedule.

After the decoder has run, we either compute an estimate $\hat{\boldsymbol{b}}$ by thresholding the L-values of the information bits, report it to the higher layers of the protocol stack and terminate the receive algorithm, or we continue with another synchronization iteration.

3.3.3 Backward Pass

So far we have discussed the forward pass through the receiver, which consists of parameter estimation using soft information $q(x_n)$ about the data symbols, followed by matched filtering, demapping and decoding. In order to close the loop, it remains to see how the beliefs $q(x_n)$ are computed based on the information that is fed back from the decoder. At first glance, this computation seems quite straightforward, but it contains a subtle aspect that to my knowledge has not been mentioned before in the literature.

Just as every belief, $q(x_n)$ is the product of all messages into the variable node x_n:

$$q(x_n) \propto m_{\mathrm{mf}\to\mathrm{dem}}(x_n)\, m_{\mathrm{dem}\to\mathrm{mf}}(x_n). \tag{3.57}$$

[10] Strictly speaking, even the receiver using the first schedule is just an approximation, because the EM algorithm requires the *exact* posterior distribution of the data, conditioned on the most recent parameter estimate, which is delivered only approximately by iterative decoders like turbo or LDPC.

The forward message, from the matched filter to the demapper, has been given in (3.56), and the backward message is easily computed as

$$\begin{aligned} m_{\mathrm{dem}\to\mathrm{mf}}(x_n) &\propto \sum_{\boldsymbol{c}_n} p(x_n \mid \boldsymbol{c}_n) \prod_{k=0}^{K-1} m_{\mathrm{dec}\to\mathrm{dem}}(c_{n,k}) \\ &\propto \sum_{\boldsymbol{c}_n} \mathbb{1}\left[x_n = \mathcal{M}(\boldsymbol{c}_n)\right] \prod_{k=0}^{K-1} \exp\left(c_{n,k} \overleftarrow{\lambda}_{n,k}\right) \\ &= \exp\left(\sum_{k=0}^{K-1} \mathcal{M}_k^{-1}(x_n)\, \overleftarrow{\lambda}_{n,k}\right) \end{aligned} \tag{3.58}$$

where $\mathcal{M}_k^{-1}(x_n)$ returns the kth bit of the label of x_n. The belief $q(x_n)$ can thus be conveniently computed in the log-domain as

$$\log q(x_n) = \underbrace{-\frac{\hat{g}_0^2}{N_0}\left|x_n - \frac{y_n(\hat{\tau}_0, \hat{\nu}_0)}{\hat{g}_0\, \mathrm{e}^{\mathrm{j}\hat{\vartheta}_0}}\right|^2}_{\text{intrinsic information}} + \underbrace{\sum_{k=0}^{K-1} \mathcal{M}_k^{-1}(x_n)\, \overleftarrow{\lambda}_{n,k}}_{\text{extrinsic information}} + \text{const} \tag{3.59}$$

where the constant term ensures the normalization $\sum_{x_n} q(x_n) = 1$.

Let us briefly reconsider the operation that the demapper carries out. In principle, it takes a discrete distribution of the data symbol as input, which we will call $m(x)$ in the following. Due to the one-to-one correspondence between symbols and bit labels, it can treat $m(x)$ equivalently as a joint distribution $m(c_0, \ldots, c_{K-1})$ of the bits. Then, it computes the bitwise marginal distributions $m(c_k)$. If we attempt to recover $m(x)$ from the $m(c_k)$, the best we can do is to approximate $m(x)$ with the product of marginals, $\prod_k m(c_k)$. Of course, all information about the statistical dependencies between the bits gets lost in this process. This illustrates the validity of the approximation

$$-\frac{\hat{g}_0^2}{N_0}\left|x_n - \frac{y_n(\hat{\tau}_0, \hat{\nu}_0)}{\hat{g}_0\, \mathrm{e}^{\mathrm{j}\hat{\vartheta}_0}}\right|^2 \approx \sum_{k=0}^{K-1} \mathcal{M}_k^{-1}(x_n)\, \vec{\lambda}_{n,k} + \text{const} \tag{3.60}$$

where the left-hand side is the log-domain input to the demapper, and the right-hand side the log-domain product of its output. The approximation consists in disregarding the statistical dependencies between the bits, which are contained in the left-hand side but lost in the right-hand side. We can thus approximate the log-belief (3.59) as

$$\begin{aligned} \log q(x_n) &\approx \sum_{k=0}^{K-1} \mathcal{M}_k^{-1}(x_n)\, \vec{\lambda}_{n,k} + \sum_{k=0}^{K-1} \mathcal{M}_k^{-1}(x_n)\, \overleftarrow{\lambda}_{n,k} + \text{const} \\ &= \sum_{k=0}^{K-1} \mathcal{M}_k^{-1}(x_n)\, \lambda_{n,k} + \text{const}. \end{aligned} \tag{3.61}$$

There exists some confusion in the literature about the question whether extrinsic or posterior L-values should be fed back from the decoder to the earlier stages. The huge interest in iterative receivers that we have witnessed in the last two decades has started with the discovery of turbo codes, followed by the re-discovery of LDPC codes and by the development of iterative demapping and decoding schemes (BICM-ID). All of these are based on belief propagation, which is governed by the *turbo principle*, a term coined by Hagenauer [53] for the concept of *extrinsic* information exchange.

Later, when the idea of iterative receive algorithms was extended to estimation tasks "in front of" the demapper, it was recognized that *posterior* L-value feedback to the synchronizer or channel estimator actually outperforms extrinsic feedback. An explanation of this finding was provided by the derivation of iterative synchronization as an instance of the EM algorithm, which computes the average over the hidden data $\boldsymbol{x}$ based on its posterior distribution $p(\boldsymbol{x} \mid \mathsf{r}; \hat{\boldsymbol{\theta}})$.

However, as the discussion above has shown, posterior L-value feedback is not optimal, either. A better way of computing $q(x_n)$ is given in (3.59), which is in fact based on *extrinsic* rather than posterior feedback from the decoder, but which also considers the available *intrinsic* information in the form of the message $m_{\mathrm{mf}\to\mathrm{dem}}(x_n)$. In order to reduce the complexity of computing $q(x_n)$, it is possible to *approximate* (3.59) by using posterior L-value feedback as in (3.61), disregarding information about the statistical dependencies between the code bits which is contained at least in the intrinsic part.[11] However, one should *not* simply ignore the intrinsic information and compute the feedback into the parameter estimator purely from the extrinsic information. As mentioned above, this is often erroneously done with reference to the turbo principle.

Besides offering a small performance improvement at the cost of a higher computational complexity, (3.59) has another, more important advantage over (3.61): it opens up the possibility for NCA synchronization iterations, which include the demapper but not the decoder. After (re-)estimating the channel parameters, the first term in (3.59) is evaluated for all $x_n \in \mathcal{X}_n$. But instead of running the decoder to get updated extrinsic L-values, a new $q(x_n)$ is computed immediately using the new intrinsic information, but the old extrinsic L-values. Depending on the decoding algorithm, this can lead to a significant complexity reduction at the cost of a decreased algorithmic performance, because this scheme only exploits knowledge about the discrete symbol alphabet, but not the structure that is imposed by the channel code.

Again, we obtain a doubly-iterative structure which leads to the question of scheduling. The two extreme cases are (a) to only iterate between estimator and demapper, and to invoke the decoder just once at the very end, and (b) to run the channel decoder in each synchronization iteration, i. e., to not do any NCA iterations at all. Between those extremes, there is of course some middle ground which offers the opportunity to trade

[11] The extrinsic part does not contain such information anyway if we are using a binary channel code. Indeed, the whole point of non-binary codes over the Galois field $\mathbb{F}_{2^m}$, $m > 1$, is to preserve the statistical dependencies at least among the m bits that constitute a code symbol, but this topic is out of scope of this thesis.

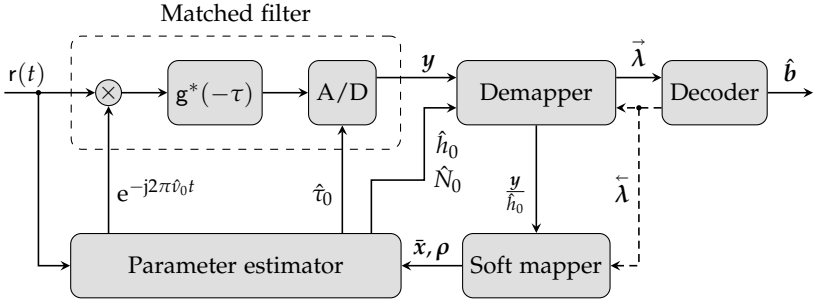


Figure 3.7: Iterative receiver structure with hard parameter estimation for the communication system shown in Figure 3.5 on page 45. The decoder is skipped in NCA iterations, so the dashed feedback line is only updated in CA iterations

off performance vs complexity. In the terminology of Section 3.1, schedule (a) is purely non-code-aided, whereas schedule (b) results in a code-aided receiver.

Note that the split into intrinsic and extrinsic information as in (3.59) is required to derive the idea of iterations between parameter estimator and demapper; the traditional view of posterior L-value feedback as in (3.61) does not allow to bypass the decoder.

3.3.4 Concluding Remarks

In this section, we have used the generic divergence minimization framework to derive an iterative receiver structure as shown in Figure 3.7. It computes hard estimates of the channel parameters, which are subsequently treated as the true parameter values, so the derived structure is a *mismatched* receiver.

The careful reader will notice the estimate $\hat{N}_0$ of the AWGN power spectral density in Figure 3.7. Indeed, information about N_0 is required by the demapper, but it has been taken as perfectly known up to now. Why did we not include N_0 as a fifth component in (3.37) in order to estimate it together with the other parameters? The key difference between N_0 and the other parameters is that the latter are associated with the useful signal, which is a superposition of finitely many finite-energy pulses, so the whole signal has finite energy. It therefore has essentially a finite duration; the only reason for considering an infinite time interval in the derivation is to obtain simpler mathematical expressions. In contrast, N_0 is associated with the noise process, which has finite *power* and therefore, in the limit of an infinite time horizon, contains an infinite amount of energy. Thus, the posterior $p(N_0 \,|\, \mathsf{r})$ is a Dirac distribution, meaning that N_0 can be estimated perfectly from r, but of course this is of no practical relevance. The question of estimating N_0 from the infinitely long observation r is ill-posed. Instead, we will

discuss the problem of SNR estimation in Section 3.7, in the context of a discrete-time transmission model with finitely many channel uses.

So far, the practically oriented engineer, who is only interested in the final algorithms, will not have found many new insights. Non-data-aided synchronization algorithms are studied extensively in [88], and the code-aided parameter estimators (3.48)–(3.51) have been presented for instance in [100] and, for the special case of BPSK modulation, in [114]. *Conceptually*, however, our derivation offers some advantages. The flexibility of DM, which contains both EM and BP as special cases, has allowed us to derive the whole receiver structure systematically in one piece, while the classical EM-based derivation in [100] requires some *ad hoc* approximations to link the code-aided parameter estimator with the outer BICM receiver. Also, we have been able to cleanly answer the question of whether to use posterior or extrinsic L-value feedback to the estimator.

But this is not the end of the story. So far, we have not even attempted to exploit the two main advantages of Bayesian methods over orthodox parameter estimation: the ability to consider prior information, and the possibility to compute soft estimates (approximations of the posterior parameter distributions) instead of hard estimates. For the time and frequency offsets τ_0 and ν_0, hard estimates seem to be a pragmatic choice, since even they do not have closed-form expressions. Computing higher moments of their distributions therefore seems to be practically infeasible. For other parameters, however, soft estimation is well possible and can have a significant positive impact on the algorithmic performance, sometimes with hardly any additional complexity.

Since we will not be discussing timing and frequency synchronization any further, we will abstract the matched filter away in the following, and work with discrete channel models like

$$y_n = g_0\, x_n\, \mathrm{e}^{\mathrm{j}\vartheta_0} + w_n \tag{3.62}$$

where any inter-symbol interference that is caused by residual synchronization errors is assumed to be included in the noise term w_n. For simplicity, we do not make the dependence of the matched filter output y_n on the estimates $\hat{\tau}_0$ and $\hat{\nu}_0$ explicit any more. However, it goes without saying that it is possible to re-estimate τ_0 and ν_0, and to update the y_n accordingly, every time a new belief $q(\boldsymbol{x}_{\mathcal{D}})$ becomes available.

3.4 Soft Carrier Phase Estimation

In this section, we generalize the phase estimator from the previous section to soft output in a very simple and elegant way by removing the Dirac-constraint from the belief $q(\vartheta_0)$. Of course, merely *producing* soft information is not sufficient; we also need to adapt the *consumer* of that information accordingly. To this end, we derive a simple demapping metric which takes the residual phase error into account.

The work presented in this section has partly been published in [117].

3.4.1 System Model

We consider the same BICM transmitter as in the previous section, shown in Figure 3.5. We assume a perfect time and frequency synchronization, so that the transmission can be described with the discrete-time model

$$y_n = g_0\, x_n\, \mathrm{e}^{\mathrm{j}\vartheta_0} + w_n, \qquad n \leq 0 < N \tag{3.63}$$

where w_n is zero-mean white Gaussian noise. We assume knowledge of the channel gain g_0 and the noise variance $N_0 = \mathbb{E}[|w_n|^2]$, so the only unknown channel parameter is the phase offset ϑ_0. In this section, we assume a constant phase shift in order to introduce the basic concepts in a simple scenario. In the following two sections, the model will be generalized to time-varying phase noise processes.

Recall that the sets $\mathcal{D}$ and $\mathcal{P}$ respectively contain the time indices of the data and pilot symbols. The vectors $\boldsymbol{x}_\mathcal{D}$ and $\boldsymbol{x}_\mathcal{P}$ consist of the transmitted data and pilot symbols, and the corresponding observations after matched filtering are collected in $\boldsymbol{y}_\mathcal{D}$ and $\boldsymbol{y}_\mathcal{P}$.

The joint posterior of the unknown quantities factorizes as

$$\begin{aligned} p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x}_\mathcal{D}, \vartheta_0 \,|\, \boldsymbol{y}) &\propto p(\boldsymbol{b})\, p(\boldsymbol{c} \,|\, \boldsymbol{b})\, p(\vartheta_0) \prod_{n\in\mathcal{D}} p(x_n \,|\, \boldsymbol{c}_n) \prod_{n=0}^{N-1} p(y_n \,|\, x_n, \vartheta_0) \\ &\overset{(a)}{=} p(\boldsymbol{b})\, p(\boldsymbol{c} \,|\, \boldsymbol{b})\, p(\vartheta_0)\, p(\boldsymbol{y}_\mathcal{P} \,|\, \boldsymbol{x}_\mathcal{P}, \vartheta_0) \prod_{n\in\mathcal{D}} p(x_n \,|\, \boldsymbol{c}_n)\, p(y_n \,|\, x_n, \vartheta_0) \end{aligned} \tag{3.64}$$

where (a) merges the likelihoods of the pilot symbols into the factor $p(\boldsymbol{y}_\mathcal{P} \,|\, \boldsymbol{x}_\mathcal{P}, \vartheta_0) = \prod_{n\in\mathcal{P}} p(y_n \,|\, x_n, \vartheta_0)$. A graphical model of (3.64) is shown in Figure 3.8.

3.4.2 Soft Phase Estimation

The variable ϑ_0 in Figure 3.8 is connected to three types of nodes: the prior distribution $p(\vartheta_0)$, the likelihood function $p(\boldsymbol{y}_\mathcal{P} \,|\, \boldsymbol{x}_\mathcal{P}, \vartheta_0)$ involving the pilot symbols, and the likelihood functions $p(y_n \,|\, x_n, \vartheta_0), n \in \mathcal{D}$, involving the data symbols. In the following, we consider each of them in turn.

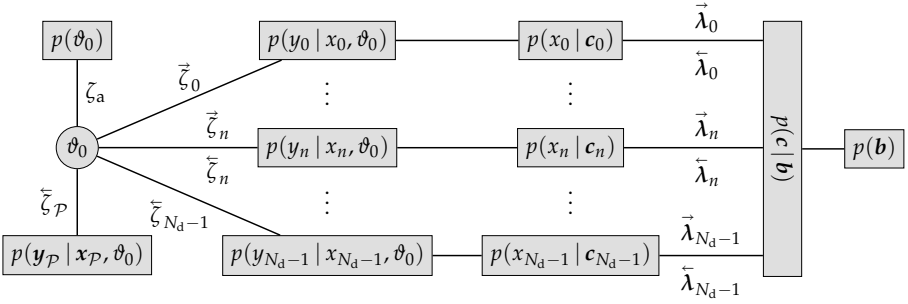


Figure 3.8: Factor graph of the joint posterior distribution (3.64)

Information from the Pilot Symbols

The likelihood function $p(\boldsymbol{y}_{\mathcal{P}} \mid \boldsymbol{x}_{\mathcal{P}}, \vartheta_0)$ is not connected to any other node, since both $\boldsymbol{y}_{\mathcal{P}}$ and $\boldsymbol{x}_{\mathcal{P}}$ are known to the receiver. Abbreviating the message $m_{p(\boldsymbol{y}_{\mathcal{P}} \mid \boldsymbol{x}_{\mathcal{P}}, \vartheta_0) \to \vartheta_0}$ as $m_{\boldsymbol{y}_{\mathcal{P}} \to \vartheta_0}$, the contribution from the pilot symbols to $q(\vartheta_0)$ is thus readily computed as

$$\begin{aligned} m_{\boldsymbol{y}_{\mathcal{P}} \to \vartheta_0}(\vartheta_0) &\propto p(\boldsymbol{y}_{\mathcal{P}} \mid \boldsymbol{x}_{\mathcal{P}}, \vartheta_0) \\ &= \frac{1}{(\pi N_0)^{N_{\mathrm{P}}}} \exp\left(-\frac{\|\boldsymbol{y}_{\mathcal{P}} - g_0 \, \boldsymbol{x}_{\mathcal{P}} \, \mathrm{e}^{\mathrm{j}\vartheta_0}\|^2}{N_0} \right) \\ &\overset{(a)}{\propto} \exp\left(\frac{2 g_0}{N_0} \operatorname{Re}\left(\boldsymbol{x}_{\mathcal{P}}^{\mathsf{H}} \boldsymbol{y}_{\mathcal{P}} \, \mathrm{e}^{-\mathrm{j}\vartheta_0} \right) \right) \\ &\overset{(b)}{\propto} \mathcal{M}\left(\vartheta_0 \,\middle|\, \frac{g_0 \, \boldsymbol{x}_{\mathcal{P}}^{\mathsf{H}} \boldsymbol{y}_{\mathcal{P}}}{N_0/2} \right) \end{aligned} \tag{3.65}$$

where (a) discards factors that are independent of ϑ_0, and (b) expresses the message as a von Mises distribution.

As a quick summary, the von Mises distribution $\mathcal{M}(\zeta)$, parameterized by $\zeta \in \mathbb{C}$, is a distribution that describes angular random variables $\vartheta \in [-\pi, \pi)$. Its PDF is

$$\begin{aligned} p(\vartheta) &= \frac{\exp\big(\kappa \cos(\vartheta - \mu)\big)}{2\pi \, \mathrm{I}_0(\kappa)} \\ &= \frac{\exp\Big(\operatorname{Re}\big(\zeta \mathrm{e}^{-\mathrm{j}\vartheta}\big)\Big)}{2\pi \, \mathrm{I}_0(|\zeta|)} \end{aligned} \tag{3.66}$$

where $\mathrm{I}_0(\cdot)$ is the modified Bessel function of the first kind and zeroth order. The parameter $\mu = \arg \zeta$ determines the mode, and $\kappa = |\zeta|$ is a measure of concentration (inverse variance). More details about the von Mises distribution are given on page 222.

For convenience, we define the parameter in (3.65) as

$$\overleftarrow{\zeta}_{\mathcal{P}} \triangleq \frac{g_0 \boldsymbol{x}_{\mathcal{P}}^{\mathsf{H}} \boldsymbol{y}_{\mathcal{P}}}{N_0/2}. \tag{3.67}$$

Note that this result is intuitively reasonable, because

$$\begin{aligned} \arg \overleftarrow{\zeta}_{\mathcal{P}} &= \arg \left(\boldsymbol{x}_{\mathcal{P}}^{\mathsf{H}} \boldsymbol{y}_{\mathcal{P}} \right) \\ &= \arg \left(g_0 \left\| \boldsymbol{x}_{\mathcal{P}} \right\|^2 \mathrm{e}^{\mathrm{j}\vartheta_0} + \boldsymbol{x}_{\mathcal{P}}^{\mathsf{H}} \boldsymbol{w}_{\mathcal{P}} \right) \end{aligned} \tag{3.68}$$

is an unbiased estimate of ϑ_0, and

$$\begin{aligned} \left| \overleftarrow{\zeta}_{\mathcal{P}} \right| &= \frac{g_0 \left| \boldsymbol{x}_{\mathcal{P}}^{\mathsf{H}} \boldsymbol{y}_{\mathcal{P}} \right|}{N_0/2} \\ &= \frac{\left| g_0^2 \left\| \boldsymbol{x}_{\mathcal{P}} \right\|^2 + \boldsymbol{x}_{\mathcal{P}}^{\mathsf{H}} \boldsymbol{w}_{\mathcal{P}} \, \mathrm{e}^{-\mathrm{j}\vartheta_0} \right|}{N_0/2} \end{aligned} \tag{3.69}$$

is approximately equal to twice the pilot SNR, given as $g_0^2 \left\| \boldsymbol{x}_{\mathcal{P}} \right\|^2 / N_0$. Note that the factor of 2 arises because the estimate of ϑ_0 is only influenced by the angular, but not the radial noise component.

Prior Information

It is usually a good idea to work with *conjugate priors*. These are priors which, when multiplied with the respective likelihood function, yield posterior distributions which have the same parametric form as the prior. Then, updating a prior to a posterior distribution in the light of new data simply requires updating the parameters of the distribution.

In our case, the likelihood function of ϑ_0 is proportional to a von Mises distribution, so we choose

$$p(\vartheta_0) = \mathcal{M}(\vartheta_0 \,|\, \zeta_{\mathrm{a}}). \tag{3.70}$$

Since the multiplication of two von Mises distributions boils down to a summation of their ζ-parameters, which can easily be seen from (3.66), the posterior of ϑ_0 (only taking the pilot symbols into account, not yet the data symbols) is simply $\mathcal{M}(\zeta_{\mathrm{a}} + \overleftarrow{\zeta}_{\mathcal{P}})$.

There is, however, a more fundamental reason for using conjugate priors beyond the mathematical convenience. At the very beginning of a transmission, there is no information about the phase available yet. The only reasonable choice is thus to set $p(\vartheta_0)$ to a uniform distribution on $[-\pi, \pi)$, which is a von Mises distribution with $\zeta_{\mathrm{a}} = 0$. It is then only determined by the likelihood function (i. e., by the channel model, and not by any subjective choices made by the algorithm designer) that the posterior distribution also becomes a von Mises distribution.

Subsequently, this posterior becomes the prior phase distribution for the next data packet, assuming that the phase offset stays constant over several packets. This observation finally justifies the usage of conjugate priors in our Bayesian estimators.

In the following Section 3.5, the same fundamental concept will be used to generalize the phase offset estimator to time-varying phase noise processes.

Information from the Data Symbols

So far, we have derived a purely pilot-aided soft phase estimator. To obtain a code-aided algorithm, it remains to incorporate the information about ϑ_0 that is contained in the data symbols. Since the factor nodes $p(y_n \mid x_n, \vartheta_0)$ are of degree 2, we must decide on the divergence measure that should be used for the computation of the messages $m_{p(y_n \mid x_n, \vartheta_0) \to \vartheta_0}$ (abbreviated as $m_{y_n \to \vartheta_0}$ in the following). We will first show that exclusive DM yields a simple and elegant code-aided soft phase estimator, which is a straightforward generalization of the hard estimator from the previous section. Afterwards we will briefly discuss the problems of inclusive DM and show why it is not applicable to the phase estimation problem (at least not without further approximations).

Exclusive DM With the likelihood function

$$p(y_n \mid x_n, \vartheta_0) = \frac{1}{\pi N_0} \exp\left(-\frac{|y_n - g_0 x_n \mathrm{e}^{\mathrm{j}\vartheta_0}|^2}{N_0} \right) \tag{3.71}$$

we can write

$$\log p(y_n \mid x_n, \vartheta_0) = \frac{g_0 \operatorname{Re}(x_n^* y_n \mathrm{e}^{-\mathrm{j}\vartheta_0})}{N_0/2} - \frac{g_0^2 \, |x_n|^2}{N_0} + \text{const} \tag{3.72}$$

and an application of the message update rule (B.62) from exclusive DM yields

$$\begin{aligned} m_{y_n \to \vartheta_0}(\vartheta_0) &\propto \exp\left(\mathbb{E}_{q(x_n)} \left[\frac{g_0 \operatorname{Re}(x_n^* y_n \mathrm{e}^{-\mathrm{j}\vartheta_0})}{N_0/2} - \frac{g_0^2 \, |x_n|^2}{N_0} \right] \right) \\ &= \exp\left(\frac{g_0 \operatorname{Re}(\bar{x}_n^* y_n \mathrm{e}^{-\mathrm{j}\vartheta_0})}{N_0/2} - \frac{g_0^2 \rho_n}{N_0} \right) \\ &\propto \mathcal{M}(\vartheta_0 \mid \overleftarrow{\zeta}_n) \quad \text{with} \quad \overleftarrow{\zeta}_n \triangleq \frac{g_0 \bar{x}_n^* y_n}{N_0/2}. \end{aligned} \tag{3.73}$$

If we compare (3.73) with (3.65), we see that both have the same functional form, the only difference is that the known value of x_n from (3.65) is replaced by the soft symbol $\bar{x}_n = \mathbb{E}_{q(x_n)}[x_n]$ in (3.73).

Combining the available information about ϑ_0 is done by multiplying all messages towards the variable node ϑ_0, which boils down to summing their ζ-parameters. Thus, $q(\vartheta_0) = \mathcal{M}(\vartheta_0 \,|\, \zeta_\mathrm{p})$ with

$$\begin{aligned}\zeta_\mathrm{p} &\triangleq \zeta_\mathrm{a} + \overleftarrow{\zeta}_{\mathcal{P}} + \sum_{n\in\mathcal{D}} \overleftarrow{\zeta}_n \\ &= \zeta_\mathrm{a} + \sum_{n=0}^{N-1} \overleftarrow{\zeta}_n \\ &= \zeta_\mathrm{a} + \frac{g_0\, \bar{\boldsymbol{x}}^\mathsf{H} \boldsymbol{y}}{N_0/2}\end{aligned} \tag{3.74}$$

where $\bar{\boldsymbol{x}}$ in the last expression contains both pilot and soft data symbols.

Recall the hard phase estimator from the previous section

$$\hat{\vartheta}_0 = \arg\left(\sum_{n=0}^{N-1} \bar{x}_n^* y_n\right) = \arg(\bar{\boldsymbol{x}}^\mathsf{H} \boldsymbol{y}) \tag{3.75}$$

which was given in (3.48). Since the mode of a von Mises distribution $\mathcal{M}(\zeta)$ is equal to $\arg\zeta$, we see that the soft estimator (3.74) is quite similar to (3.48), but is improved in two aspects: it allows for a non-uniform prior, and it additionally yields reliability information in terms of $|\zeta_\mathrm{p}|$. The complexity of both estimators is dominated by the inner product $\bar{\boldsymbol{x}}^\mathsf{H} \boldsymbol{y}$, so the new reliability information essentially comes for free.

Inclusive DM In order to appreciate the elegance of the estimator above, let us see what inclusive DM would yield. Straightforward application of belief propagation gives

$$\begin{aligned} m_{y_n \to \vartheta_0}(\vartheta_0) &\propto \sum_{x_n \in \mathcal{X}_n} m_{x_n \to y_n}(x_n) \exp\left(\frac{g_0 \operatorname{Re}(x_n^* y_n \mathrm{e}^{-\mathrm{j}\vartheta_0})}{N_0/2} - \frac{g_0^2 \,|x_n|^2}{N_0}\right) \\ &\propto \sum_{x_n \in \mathcal{X}_n} m_{x_n \to y_n}(x_n) \exp\left(-\frac{g_0^2 \,|x_n|^2}{N_0}\right) \mathcal{M}\left(\vartheta_0 \,\middle|\, \frac{g_0\, x_n^* y_n}{N_0/2}\right) \end{aligned} \tag{3.76}$$

i. e., a mixture of von Mises distributions with $|\mathcal{X}_n|$ components. The belief $q(\vartheta_0)$, which is the product of all incoming messages, then becomes a mixture of $|\mathcal{X}|^{N_\mathrm{d}}$ components (assuming that all $\mathcal{X}_n$ are equal). This is a typical example of exponential blowup, which makes the estimator intractable for any relevant N_d.

We could circumvent this problem by projecting $q(\vartheta_0)$ into the family of von Mises distributions, yielding expectation propagation rather than belief propagation. We omit the detailed derivation in the interest of space, but it is easy to show that EP results in a complex and numerically unstable algorithm, so this approach is a dead end, too.

Finally, we could apply the projection not to ϑ_0, but to the data symbols x_n, projecting $q(x_n)$ into the family of Gaussians. Assume that $m_{x_n \to y_n}(x_n) \propto \mathcal{CN}(x_n \mid \overleftarrow{\mu}_n, \overleftarrow{\sigma}_n^2)$. Then

$$m_{y_n \to \vartheta_0}(\vartheta_0) \propto \int m_{x_n \to y_n}(x_n) \exp\left(-\frac{|y_n - g_0\, x_n e^{\mathrm{j}\vartheta_0}|^2}{N_0}\right) \mathrm{d}x_n \tag{3.77}$$

which, after a lengthy calculation that we shall omit, becomes

$$m_{y_n \to \vartheta_0}(\vartheta_0) \propto \mathcal{M}(\vartheta_0 \mid \overleftarrow{\zeta}_n) \quad \text{with} \quad \overleftarrow{\zeta}_n = \frac{2 g_0\, \overleftarrow{\mu}_n^* y_n}{g_0^2 \overleftarrow{\sigma}_n^2 + N_0}. \tag{3.78}$$

The assumption of Gaussian data symbols indeed results in a tractable message towards the phase estimator, which is actually quite similar to (3.73). The difference is that (3.78) expresses a larger variance (smaller $|\overleftarrow{\zeta}|$) of the phase estimation error, as it also includes the uncertainty about x_n in terms of its variance $\overleftarrow{\sigma}_n^2$, which is ignored by (3.73).[12] It can thus be expected that the message obtained from inclusive DM yields better results, albeit with an increased complexity. However, simulation results presented in Figure 3.12 below indicate that the performance gain is very small, so the simpler (3.73) is probably the more interesting choice in practice.

If we are willing to use (3.78) rather than (3.73), the remaining question is how to compute $\overleftarrow{\mu}_n$ and $\overleftarrow{\sigma}_n^2$. The authors of [22], who consider the problem of phase estimation purely from a belief propagation perspective, simply replace the discrete BP message $m_{x_n \to y_n}(x_n)$ by a Gaussian with the same mean and variance, without discussing this issue any further. Can we do better? A systematic way to force $q(x_n)$ into a Gaussian form would be expectation propagation, but unfortunately an exact application of EP again yields complicated and thus practically uninteresting expressions. However, with an additional approximation (essentially dropping the division by $m_{n \to k}(x_n)$ in (B.56)) we obtain a tractable expression, which differs from the proposal in [22] in the fact that it takes the mean and variance of the *posterior* information of x_n, computed as in (3.59), whereas [22] only takes the *extrinsic* feedback into account. We will demonstrate via simulations that our approximated EP algorithm offers a non-negligible performance improvement over [22], which should not come as a surprise after the discussion in Section 3.3.3.

To summarize, the soft phase estimate is given by

$$\zeta_\mathrm{p} = \zeta_\mathrm{a} + \sum_{n=0}^{N-1} \overleftarrow{\zeta}_n \tag{3.79}$$

where the $\overleftarrow{\zeta}_n$ are computed either via (3.73) or via (3.78).

[12] In fact, it is a well-known result that exclusive DM tends to underestimate the variance. We will encounter this issue again in Section 4.4, where we discuss MRC-assisted MIMO detectors which are also derived from exclusive DM.

3.4.3 Demapping with Soft Phase Information

In order to put the soft phase information to good use, our final step is the derivation of a demapping metric which takes the uncertainty in the phase estimate into account. Since x_n is a discrete variable, we can use belief propagation for this task:

$$m_{y_n \to \mathrm{dem}}(x_n) \propto \int_{-\pi}^{\pi} m_{\vartheta_0 \to y_n}(\vartheta_0)\, p(y_n \mid x_n, \vartheta_0)\, \mathrm{d}\vartheta_0 \tag{3.80}$$

with the forward message from the phase estimator

$$\begin{aligned} m_{\vartheta_0 \to y_n}(\vartheta_0) &= \frac{q(\vartheta_0)}{m_{y_n \to \vartheta_0}(\vartheta_0)} \\ &\propto \mathcal{M}(\vartheta_0 \mid \vec{\zeta}_n). \end{aligned} \tag{3.81}$$

Here,

$$\vec{\zeta}_n \triangleq \zeta_\mathrm{p} - \overleftarrow{\zeta}_n \tag{3.82}$$

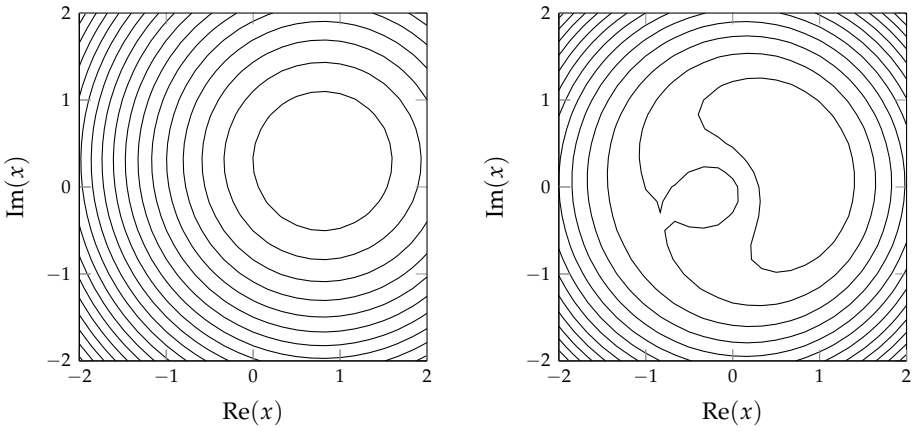
is the *extrinsic* phase information, i. e., the information about ϑ_0 that is contained in the y_m with $m \neq n$ (we will also call $\overleftarrow{\zeta}_n$ the *intrinsic* phase information in the following). We will later show via simulations that using extrinsic phase information in the demapper indeed yields better results than using the whole posterior phase information—a maybe somewhat counterintuitive result that we would not have obtained by deriving the estimator in isolation and connecting it to the outer receiver in some *ad hoc* way, as it is often done in the literature.

For notational convenience, we define the polar decomposition $\vec{\zeta}_n = \vec{\kappa}_n \mathrm{e}^{\mathrm{j}\vec{\mu}_n}$. Then

$$\begin{aligned} m_{y_n \to \mathrm{dem}}(x_n) &\propto \int_{-\pi}^{\pi} \frac{\exp\left(\mathrm{Re}\left(\vec{\zeta}_n^{\,*} \mathrm{e}^{\mathrm{j}\vartheta}\right)\right)}{2\pi\, \mathrm{I}_0(\vec{\kappa}_n)} \exp\left(-\frac{|y_n - g_0\, x_n \mathrm{e}^{\mathrm{j}\vartheta_0}|^2}{N_0}\right) \mathrm{d}\vartheta_0 \\ &= \frac{1}{2\pi\, \mathrm{I}_0(\vec{\kappa}_n)} \exp\left(-\frac{|y_n|^2 + g_0^2\, |x_n|^2}{N_0}\right) \int_{-\pi}^{\pi} \exp\left(\mathrm{Re}\left(\left(\vec{\zeta}_n + \frac{g_0\, x_n^* y_n}{N_0/2}\right)^* \mathrm{e}^{\mathrm{j}\vartheta_0}\right)\right) \mathrm{d}\vartheta_0 \\ &\overset{(a)}{=} \frac{1}{2\pi\, \mathrm{I}_0(\vec{\kappa}_n)} \exp\left(-\frac{|y_n|^2 + g_0^2\, |x_n|^2}{N_0}\right) 2\pi\, \mathrm{I}_0\left(\left|\vec{\zeta}_n + \frac{g_0\, x_n^* y_n}{N_0/2}\right|\right) \\ &\overset{(b)}{=} \exp\left(-\frac{|y_n|^2 + g_0^2\, |x_n|^2}{N_0}\right) \frac{\mathrm{I}_0\left(\left|\vec{\kappa}_n + \frac{g_0\, x_n^* y_n\, \mathrm{e}^{-\mathrm{j}\vec{\mu}_n}}{N_0/2}\right|\right)}{\mathrm{I}_0(\vec{\kappa}_n)} \end{aligned} \tag{3.83}$$

where (a) uses the identity $\int_{-\pi}^{\pi} \exp\left(\mathrm{Re}(\zeta^* \mathrm{e}^{\mathrm{j}\vartheta})\right) \mathrm{d}\vartheta = 2\pi\, \mathrm{I}_0(|\zeta|)$ and (b) multiplies the argument of the abs-function with $\mathrm{e}^{-\mathrm{j}\vec{\mu}_n}$.

The metric (3.83) has already been derived in [22]. However, due to the Bessel functions, evaluation of (3.83) is numerically quite expensive. In the following, we



(a) Mismatched demapping metric, which is equivalent to (3.85) with $\vec{\kappa} \to \infty$

(b) Demapping metric (3.85) with soft phase information, using $\vec{\kappa} = 10$

Figure 3.9: Contour plots of demapping metrics (a) assuming a perfect phase estimate, and (b) taking the phase uncertainty into account

propose a simplification which exploits that the Bessel function $\mathrm{I}_0(z)$ is, for $z \gg 0$, dominated by an exponential behaviour. With $\tilde{\mathrm{I}}_0(z) \triangleq \mathrm{I}_0(z)\,\mathrm{e}^{-z}$ and the notational shortcut

$$\tilde{\zeta}(x_n) \triangleq \vec{\kappa}_n + \frac{g_0\, x_n^* y_n\, \mathrm{e}^{-\mathrm{j}\vec{\mu}_n}}{N_0/2} \tag{3.84}$$

equation (3.83) becomes

$$\begin{aligned} m_{y_n \to \mathrm{dem}}(x_n) &\propto \exp\left(-\frac{|y_n|^2 + g_0^2\,|x_n|^2}{N_0} + \left|\tilde{\zeta}(x_n)\right| - \vec{\kappa}_n\right) \frac{\tilde{\mathrm{I}}_0\left(\left|\tilde{\zeta}(x_n)\right|\right)}{\tilde{\mathrm{I}}_0(\vec{\kappa}_n)} \\ &= \exp\left(-\frac{\left|y_n - g_0\, x_n\, \mathrm{e}^{\mathrm{j}\vec{\mu}_n}\right|^2}{N_0} + \left|\tilde{\zeta}(x_n)\right| - \mathrm{Re}\left(\tilde{\zeta}(x_n)\right)\right) \frac{\tilde{\mathrm{I}}_0\left(\left|\tilde{\zeta}(x_n)\right|\right)}{\tilde{\mathrm{I}}_0(\vec{\kappa}_n)} \\ &\overset{(a)}{\approx} \exp\left(-\frac{\left|y_n - g_0\, x_n\, \mathrm{e}^{\mathrm{j}\vec{\mu}_n}\right|^2}{N_0} + \left|\tilde{\zeta}(x_n)\right| - \mathrm{Re}\left(\tilde{\zeta}(x_n)\right)\right) \end{aligned} \tag{3.85}$$

where (a) drops the ratio of the scaled Bessel functions. I have verified numerically that this simplification does not cause any noteworthy performance degradation.

The first term in the exp-function is the usual Gaussian detection metric as in (2.16), and $\left|\tilde{\zeta}(x)\right| - \mathrm{Re}\left(\tilde{\zeta}(x)\right)$ is a correction term due to the residual phase error. It can be shown that it converges to 0 as $\vec{\kappa} \to \infty$, i. e., in the limit of a perfect phase estimate.

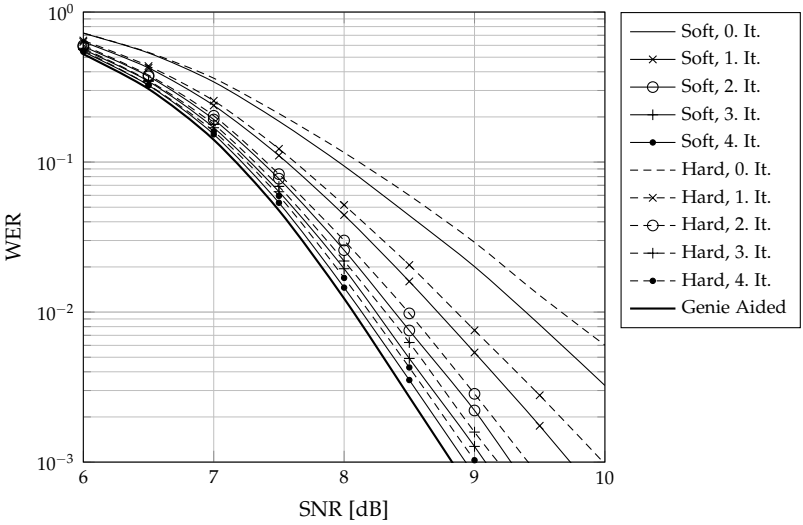


Figure 3.10: Comparison of soft and hard phase estimation

The effect of the correction term is demonstrated in Figure 3.9, which shows contour plots of the demapping metrics with $\vec{\kappa} \to \infty$ and $\vec{\kappa} = 10$, assuming a received symbol of $y = 0.8 + 0.3\mathrm{j}$. Due to the assumption of proper Gaussian noise, the left plot consists of concentric circles around the received symbol. In the right plot, which assumes a zero-mean residual phase error with standard deviation $1/\sqrt{\vec{\kappa}} \approx 18°$, those circles are "smeared" in angular direction, causing the metric to decrease faster in radial than in angular direction.

3.4.4 Numerical Results

We now present a performance evaluation of the proposed soft phase estimator. As a running example, the simulated system uses the 3GPP LTE turbo code of length $N_\mathrm{b} = 128$ bit, which consists of two constituent RSC codes with generator polynomials $(1, 15/13)_8$. The rate-1/3 mother code is punctured to a rate of $1/2$, and the resulting codeword has length $N_\mathrm{c} = 264$ bit (the true rate is slightly lower than $1/2$ because the tail bits, which drive the constituent encoders into the zero state at the end, are also transmitted). The modulator uses a 16-QAM alphabet with Gray mapping, yielding $N_\mathrm{d} = 66$ data symbols. For initial phase estimation, a preamble of $N_\mathrm{p} = 4$ pilot symbols is prepended, so the total block length is $N = 70$ symbols.

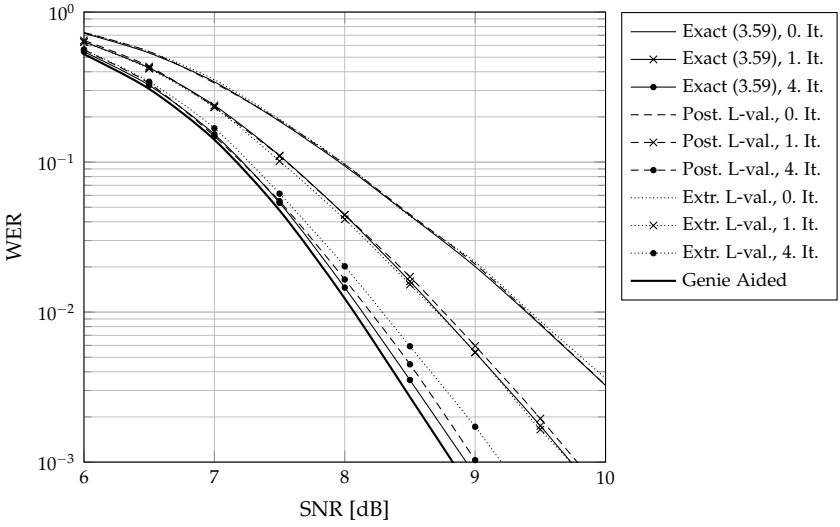


Figure 3.11: Comparison of different ways to compute the feedback $q(\boldsymbol{x}_{\mathcal{D}})$

Soft vs Hard Phase Information First, we study the advantage that we get by using the soft demapping metric, i. e., by considering the residual phase error, versus using the mismatched metric which assumes perfect phase information. Figure 3.10 shows the codeword error rate (WER) as a function of the SNR, for up to four receiver iterations. The plots labeled "0. It." refer to the initial pilot-aided iteration. Note that the mismatched receiver (labeled "hard"), which ignores the phase reliability information in the demapper, is essentially identical to the EM-based phase estimator as in [99], which does not compute reliability information in the first place. Therefore, this simulation compares our proposed estimator with the state of the art. A non-negligible performance gain is clearly visible, particularly in the first iterations where the residual phase error is largest. In later iterations, when the phase estimate becomes more reliable, the soft metric converges to the hard metric and hence both yield similar results.

The following simulations all use the soft demapping metric.

Posterior vs Extrinsic Feedback In a second experiment, we compare the three different ways of computing the feedback $q(\boldsymbol{x}_{\mathcal{D}})$ that we have discussed in Section 3.3.3:

- An exact computation of the posterior information according to (3.59).

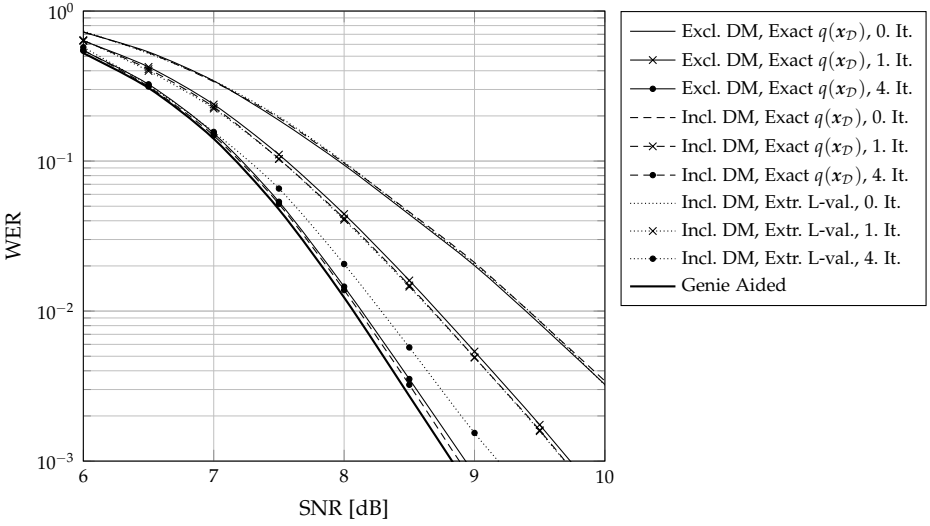


Figure 3.12: Comparison of exclusive (3.73) and inclusive (3.78) DM

- A computation according to (3.61), using the posterior L-values. The approximation relative to the exact computation consists of neglecting statistical dependencies in the intrinsic part of (3.59).
- A computation using only the extrinsic L-values. Here, the approximation consists of neglecting the whole intrinsic information.

Figure 3.11 shows the results after the initial, the first and the fourth iteration in order to keep the figure readable. After the first iteration, the three feedback types are hardly distinguishable. After four iterations, however, the exact computation that we derived in (3.59) clearly leads to the best results. It is closely followed by the posterior L-values, which shows that (3.61) is indeed a good approximation of (3.59). In contrast, using only the extrinsic L-values as feedback into the phase estimator, which is sometimes done in the literature based on the heuristic (and obviously flawed) analogy with iterative channel decoders, causes a larger performance degradation.[13]

Exclusive vs Inclusive DM Next we compare the two proposed ways (3.73) and (3.78) of converting the discrete belief $q(x_n)$ of the nth data symbol into the parameter $\overleftarrow{\zeta}_n$

[13]In this experiment, the difference between the feedback types is rather small. However, we will repeat this comparison in the following section in the context of phase noise estimation, where the difference between posterior and extrinsic feedback is very significant.

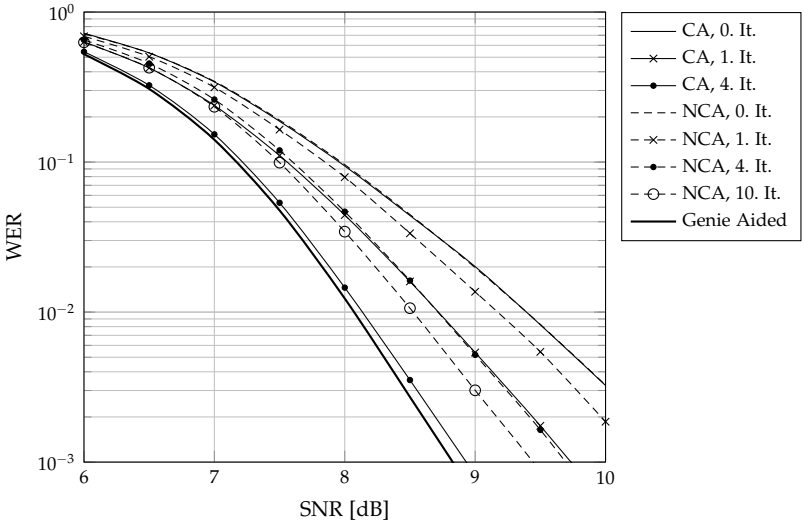


Figure 3.13: Comparison of code-aided and non-code-aided phase estimation

which describes the message into the phase estimator. (The two earlier simulations both used (3.73).) Recall that the two formulas differ in that (3.78), derived via *inclusive* DM, also accounts for the uncertainty of the data symbol, whereas (3.73), obtained by *exclusive* DM, only uses the mean $\bar{x}_n$. Thus, it is expected that (3.78) yields better results, albeit with a larger computational complexity. The comparison in terms of WER is shown in Figure 3.12, again after zero, one and four iterations. For the inclusive case, we present results both for exact computation of $q(\boldsymbol{x}_{\mathcal{D}})$ and for purely extrinsic feedback. Note that the combination of inclusive DM (3.78) with extrinsic L-value feedback has been proposed in [22], where the phase estimator has been derived from belief propagation, and where the extrinsic (and discrete) BP message $m_{\mathrm{dem}\to y_n}(x_n)$ is heuristically replaced with a Gaussian in order to keep the complexity manageable.

The results in Figure 3.12 show that, after four iterations, inclusive DM indeed performs slightly better than exclusive DM as expected—but only for posterior feedback. Using extrinsic feedback, as proposed in [22], leads to a considerable performance degradation. Also, the difference between inclusive and exclusive DM (when using posterior feedback) is rather small, so the higher complexity of (3.78) is probably not justified.

Code-Aided vs Non-Code-Aided Estimation In the last experiment of this section, we compare the code-aided algorithm that has been studied up to now with its non-code-

aided counterpart, as discussed in Section 3.3.3. The NCA formulation relies on the decomposition (3.59) of $q(x_n)$ into an intrinsic part, provided by the demapper for the nth data symbol, and an extrinsic part, provided by the decoder in terms of extrinsic L-values. In an NCA iteration, the decoder is skipped, and thus only the intrinsic part of $q(x_n)$ is updated while the extrinsic L-values remain unaltered. The results after one, four and ten NCA iterations are shown in Figure 3.13. Not surprisingly, the NCA algorithm performs considerably worse than the CA receiver. After four NCA iterations, the results are similar as with one single CA iteration, and after ten NCA iterations, when the algorithm has converged, there remains a non-negligible gap to the CA algorithm after four iterations. However, depending on the relative computational complexity of the demapper and the decoder, an NCA iteration may be significantly cheaper. Also note that the two algorithmic schedules that are compared here are, in a certain sense, the most extreme ones: only CA iterations versus only NCA iterations. Mixed schedules, for instance running the decoder every kth iteration with $k > 1$, might be an interesting compromise between performance and complexity.

A detailed comparison of different schedules depends on many parameters like the modulation and coding scheme and the expected SNR, and is out of scope of this thesis. However, having the option of NCA iterations available is clearly an enrichment of our algorithmic toolbox.

Root-Mean-Squared Estimation Error and Cramér-Rao Lower Bound The quality of a (hard) parameter estimator is often measured in terms of its mean-squared error (MSE). A well-known result from orthodox statistics states that the MSE of any unbiased estimator $\hat{\theta}$ is lower-bounded by the Cramér-Rao Lower Bound (CRLB) [64, Section 3.4]

$$\operatorname{var}\left[\hat{\theta}\right] \geq \frac{1}{I(\theta)} \tag{3.86}$$

with the Fisher information

$$I(\theta) \triangleq -\mathbb{E}\left[\frac{\partial^2 \log p(\boldsymbol{y};\theta)}{\partial \theta^2}\right]. \tag{3.87}$$

Although we are working in a Bayesian setting, we can still derive a hard phase estimate from ζ_p as $\hat{\vartheta}_0 = \arg \zeta_\mathrm{p}$, so it is meaningful to evaluate the MSE of our phase estimator and compare it to the CRLB. Assume a purely pilot-aided estimator, i. e., that the whole block $\boldsymbol{x}$ of length N is known to the receiver. Then the log-likelihood function of ϑ_0 is

$$\log(\boldsymbol{y};\vartheta_0) = -\frac{1}{N_0}\left\|\boldsymbol{y} - g_0 \boldsymbol{x}\, \mathrm{e}^{\mathrm{j}\vartheta_0}\right\|^2 + \text{const} \tag{3.88}$$

with second derivative

$$\frac{\partial^2 \log p(\boldsymbol{y};\vartheta_0)}{\partial \vartheta_0^2} = -\frac{g_0 \operatorname{Re}(\boldsymbol{x}^\mathsf{H} \boldsymbol{y}\, \mathrm{e}^{\mathrm{j}\vartheta_0})}{N_0/2}. \tag{3.89}$$

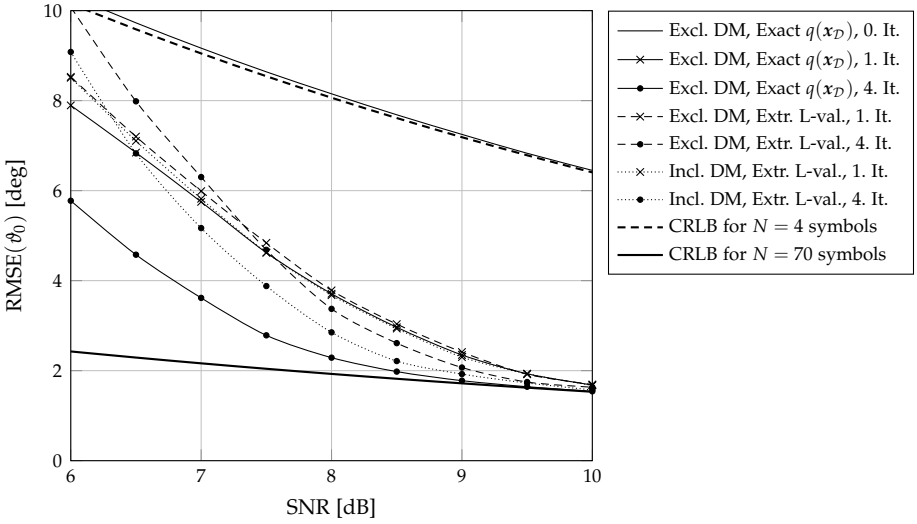


Figure 3.14: Root-mean-squared error of the phase estimate, comparing exact (posterior) and extrinsic feedback. For the case of extrinsic feedback, both exclusive (3.73) and inclusive (3.78) DM are shown

Since $\mathbb{E}[\boldsymbol{y}] = g_0 \boldsymbol{x} e^{j\vartheta_0}$, the Fisher information becomes

$$I(\vartheta_0) = \frac{g_0^2 \|\boldsymbol{x}\|^2}{N_0/2} = 2N\,\mathsf{SNR} \tag{3.90}$$

where

$$\mathsf{SNR} = \frac{g_0^2 E_\mathrm{s}}{N_0} \tag{3.91}$$

is the average signal-to-noise ratio as defined in (2.10) (but with known channel gain g_0). The root-mean-squared error (RMSE), measured in degrees, is then lower-bounded as

$$\mathrm{RMSE}(\hat{\vartheta}_0) \geq \frac{180^\circ}{\pi\sqrt{2N\,\mathsf{SNR}}}. \tag{3.92}$$

Figure 3.14 shows the RMSE for our phase estimator with the following choices: extrinsic feedback with (a) exclusive and (b) inclusive DM, and (c) posterior feedback with exclusive DM (when using posterior feedback, the results for inclusive DM are virtually the same and are therefore omitted in the figure). For reference, the CRLB is plotted with $N = 4$ (the number of pilot symbols in our system), and $N = 70$ (the total block length).

After the zeroth iteration, where the phase estimator works purely pilot-aided, the RMSE is very close to the CRLB for $N = 4$. Note that the three algorithms do not yet differ after the initial pilot-aided iteration, so only one plot is shown in the figure.

The plots for the other, code-aided iterations are more interesting. The lowest RMSE is clearly obtained for posterior feedback, which should not be surprising after the preceding discussion. In high SNR (say, $\geq 9\,\mathrm{dB}$), where the WER drops below 10^{-3}, it reaches the CRLB for $N = 70$ symbols, which is derived under the assumption that the whole block consists only of training data.

The RMSE gets far worse when only extrinsic information is fed back into the channel estimator. When the $\overleftarrow{\zeta}_n$ are derived via exclusive DM, the RMSE even *increases* over the iterations, at least in the low SNR range. The performance gets slightly better when using inclusive DM, but even then, the RMSE is significantly larger than with posterior feedback, which again confirms our result that posterior feedback is the correct choice.

3.5 Wiener Phase Noise Estimation

So far, we have only considered the estimation of a static phase offset. Due to the Bayesian problem formulation, however, the algorithm from the previous section can naturally be extended to time-varying phase noise processes, by adapting the statistical model $p(\boldsymbol{\vartheta})$, where $\boldsymbol{\vartheta} \triangleq (\vartheta_0, \dots, \vartheta_{N-1})$. In the following, we derive an estimator for the simple, but nonetheless practically relevant example of Wiener phase noise, which is a common model for free running oscillators. The subsequent section will then develop an extension to linear state space models, which are sufficiently general for an accurate description of phase-locked loops.

3.5.1 System Model

The only difference to the system model from the previous section is that we now consider the phase ϑ to be time-varying, so the channel model is now

$$y_n = g_0\, x_n\, \mathrm{e}^{\mathrm{j}\vartheta_n} + w_n, \qquad n \leq 0 < N. \tag{3.93}$$

It remains to specify a stochastic model for the phase noise process. Following [31], we write the continuous-time output of a noisy oscillator with frequency f_c as[14]

$$\begin{aligned} \mathsf{z}(t) &= A\, \cos(2\pi f_\mathrm{c}(t + \alpha(t))) \\ &= \frac{A}{2}\left(\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}(t+\alpha(t))} + \mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}(t+\alpha(t))}\right) \end{aligned} \tag{3.94}$$

where $\alpha(t)$ is a stochastic time shift. The autocovariance function of z is

$$\begin{aligned} \mathsf{K}_\mathsf{z}(t,\tau) &= \mathbb{E}\left[\mathsf{z}(t+\tau)\,\mathsf{z}^*(t)\right] \\ &= \frac{A^2}{4}\,\mathbb{E}\Big[\left(\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}(t+\tau+\alpha(t+\tau))} + \mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}(t+\tau+\alpha(t+\tau))}\right) \\ &\qquad \left(\mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}(t+\alpha(t))} + \mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}(t+\alpha(t))}\right)\Big] \\ &= \frac{A^2}{4}\,\mathbb{E}\Big[\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}(\tau+\alpha(t+\tau)-\alpha(t))} + \mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}(2t+\tau+\alpha(t+\tau)+\alpha(t))} \\ &\qquad + \mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}(2t+\tau+\alpha(t+\tau)+\alpha(t))} + \mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}(\tau+\alpha(t+\tau)-\alpha(t))}\Big]. \end{aligned} \tag{3.95}$$

Asymptotically, for $t \to \infty$, the expectations of the second and third term in (3.95) vanish, as shown in [31, Eq. (33)]. This can be intuitively seen because the variance of $\alpha(t)$ increases with time, so asymptotically, the term $\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}\alpha(t)}$ becomes uniformly distributed on the unit circle.

[14] The term $y(t)$ in [31, Eq. (11)], called the *orbital deviation*, is shown in [31] to remain small, and will be ignored in the following derivation.

The difference $x \triangleq \alpha(t+\tau) - \alpha(t)$ asymptotically becomes a Gaussian random variable with linearly increasing variance (see [31, Eqs. (30)–(32)])

$$x \sim \mathcal{N}(0, c\,|\tau|) \tag{3.96}$$

where the constant c is determined by the quality of the oscillator. Thus, the autocovariance function is asymptotically independent of t:

$$\begin{aligned}
\mathsf{K}_\mathsf{z}(\tau) &\triangleq \lim_{t\to\infty} \mathsf{K}_\mathsf{z}(t,\tau) \\
&= \frac{A^2}{4}\left(\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}\tau}\,\mathbb{E}\left[\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}x}\right] + \mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}\tau}\,\mathbb{E}\left[\mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}x}\right]\right) \\
&= \frac{A^2}{4}\left(\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}\tau}\,\varphi_x(2\pi f_\mathrm{c}) + \mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}\tau}\,\varphi_x(-2\pi f_\mathrm{c})\right)
\end{aligned} \tag{3.97}$$

where $\varphi_x(\omega) \triangleq \mathbb{E}_x\left[\mathrm{e}^{\mathrm{j}\omega x}\right]$ is the characteristic function (CF) of x. The CF of a $\mathcal{N}(\mu,\sigma^2)$ random variable is $\varphi(\omega) = \mathrm{e}^{\mathrm{j}\mu\omega - \frac{1}{2}\sigma^2\omega^2}$, and thus

$$\mathsf{K}_\mathsf{z}(\tau) = \frac{A^2}{4}\,\mathrm{e}^{-\frac{1}{2}c|\tau|4\pi^2 f_\mathrm{c}^2}\left(\mathrm{e}^{\mathrm{j}2\pi f_\mathrm{c}\tau} + \mathrm{e}^{-\mathrm{j}2\pi f_\mathrm{c}\tau}\right). \tag{3.98}$$

The asymptotic power spectral density (PSD) of the oscillator process is then

$$\begin{aligned}
\mathsf{S}_\mathsf{z}(f) &= \int_{-\infty}^{\infty} \mathsf{K}_\mathsf{z}(\tau)\,\mathrm{e}^{-\mathrm{j}2\pi f\tau}\,\mathrm{d}\tau \\
&= \frac{A^2}{4}\int_{-\infty}^{\infty} \mathrm{e}^{-|\tau|c2\pi^2 f_\mathrm{c}^2}\left(\mathrm{e}^{-\mathrm{j}2\pi(f-f_\mathrm{c})\tau} + \mathrm{e}^{-\mathrm{j}2\pi(f+f_\mathrm{c})\tau}\right)\mathrm{d}\tau.
\end{aligned} \tag{3.99}$$

Let $\mathcal{L}(0,b)$ denote the zero-mean Laplace distribution, whose density is

$$p(\tau) = \frac{1}{2b}\exp\left(-\frac{|\tau|}{b}\right). \tag{3.100}$$

Then (3.99) can be written as the expectation

$$\begin{aligned}
\mathsf{S}_\mathsf{z}(f) &= \frac{A^2 2b}{4}\,\mathbb{E}_\tau\left[\mathrm{e}^{-\mathrm{j}2\pi(f-f_\mathrm{c})\tau} + \mathrm{e}^{-\mathrm{j}2\pi(f+f_\mathrm{c})\tau}\right] \\
&= \frac{A^2 b}{2}\left(\varphi_\tau(-2\pi(f-f_\mathrm{c})) + \varphi_\tau(-2\pi(f+f_\mathrm{c}))\right)
\end{aligned} \tag{3.101}$$

where $\tau \sim \mathcal{L}(0,b)$ with $b = \frac{1}{c2\pi^2 f_\mathrm{c}^2}$. Its CF is $\varphi_\tau(\omega) = \frac{1}{1+b^2\omega^2}$, so

$$\mathsf{S}_\mathsf{z}(f) = \frac{A^2 b}{2}\left(\frac{1}{1+b^2 4\pi^2(f-f_\mathrm{c})^2} + \frac{1}{1+b^2 4\pi^2(f+f_\mathrm{c})^2}\right). \tag{3.102}$$

As we are only interested in frequencies around the carrier frequency, we can safely neglect the second term, which will be much smaller than the first one. Substituting b, we obtain

$$\mathsf{S}_\mathsf{z}(f) \approx \frac{A^2}{4} \frac{c{f_\mathrm{c}}^2}{\pi^2 c^2 {f_\mathrm{c}}^4 + (f - f_\mathrm{c})^2}. \tag{3.103}$$

The phase noise PSD is usually specified as the single-sided PSD of z, normalized to the carrier power $P = A^2/2$ and taken as a function of $\Delta f \triangleq f - f_\mathrm{c}$ [31, Eq. (40)]

$$\begin{aligned} \mathsf{L}(\Delta f) &\triangleq \frac{2\,\mathsf{S}_\mathsf{z}(f_\mathrm{c} + \Delta f)}{A^2/2} \\ &\approx \frac{c{f_\mathrm{c}}^2}{\pi^2 c^2 {f_\mathrm{c}}^4 + \Delta f^2}. \end{aligned} \tag{3.104}$$

It is typically expressed logarithmically as $10 \log_{10}(\mathsf{L}(\Delta f))$ in units of dBc/Hz.

We can distinguish two regions of the PSD, separated at the corner frequency $\pi c {f_\mathrm{c}}^2$. For very small Δf, the PSD becomes flat with

$$\mathsf{L}(\Delta f \to 0) \approx \frac{1}{\pi^2 c {f_\mathrm{c}}^2} \tag{3.105}$$

whereas for $\Delta f \gg \pi c {f_\mathrm{c}}^2$, the PSD has a slope of -20 dB per decade

$$\mathsf{L}(\Delta f) \approx c \left(\frac{f_\mathrm{c}}{\Delta f} \right)^2. \tag{3.106}$$

Figure 3.15 shows the spectral densities of two simulated phase noise processes (generated via (3.108) below) and the analytic asymptotes (3.105) and (3.106), with numerical values taken from [14]. Note that the corner frequency of the high-quality oscillator with $c = 10^{-18}\,\mathrm{s}$ is at $\pi\, 10^{-18}\,\mathrm{s}\,(200\,\mathrm{MHz})^2 \approx 0.126\,\mathrm{Hz}$, and is therefore outside the plot.

The continuous-time phase noise process $\vartheta(t)$ is related to the time shift $\alpha(t)$ via

$$\vartheta(t) = 2\pi f_\mathrm{c} \alpha(t). \tag{3.107}$$

Because of (3.96), we have $\alpha(nT) - \alpha((n-1)T) \sim \mathcal{N}(0, cT)$, so the discrete-time phase noise process $\vartheta_n \triangleq \vartheta(nT)$ can be modeled as a random walk

$$\vartheta_n = \vartheta_{n-1} + \Delta\vartheta_n \tag{3.108}$$

where the $\Delta\vartheta_n$ are iid zero-mean Gaussian random variables with standard deviation

$$\begin{aligned} \sigma_{\Delta\vartheta} &= 2\pi f_\mathrm{c} \sigma_{\Delta\alpha} \\ &= 2\pi f_\mathrm{c} \sqrt{cT} \end{aligned} \tag{3.109}$$

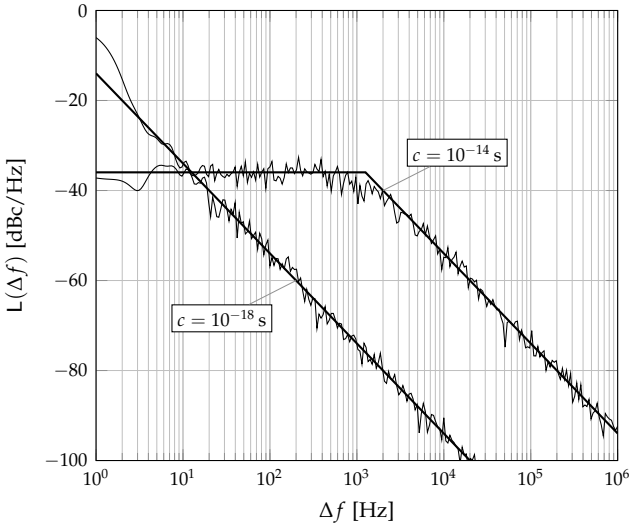


Figure 3.15: Power spectral densities of two simulated phase noise processes with oscillator frequency $f_\mathrm{c} = 200\,\mathrm{MHz}$ and symbol rate $T^{-1} = 10\,\mathrm{MHz}$, along with the analytic asymptotes (3.105) and (3.106) (thick lines)

$$= 2\pi\Delta f\sqrt{\mathsf{L}(\Delta f)T}. \tag{3.110}$$

Above, (3.110) uses the asymptotic expression (3.106), i. e., $\mathsf{L}(\Delta f)$ is the one-sided PSD evaluated at $\Delta f \gg \pi c f_\mathrm{c}$. For example, the standard deviations of the phase increments of the two processes from Figure 3.15 are $\sigma_{\Delta\vartheta} = 360^\circ \cdot 20\,\mathrm{kHz}\sqrt{10^{-6}\,\mathrm{s}/10\,\mathrm{MHz}} \approx 2.28^\circ$ for the low-quality oscillator where $\mathsf{L}(20\,\mathrm{kHz}) = -60\,\mathrm{dBc/Hz}$, and analogously $\sigma_{\Delta\vartheta} \approx 0.0228^\circ$ for the high-quality oscillator.

Similar to (3.64), the posterior distribution of the unknown quantities in the currently considered system can be factorized as

$$p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x}_\mathcal{D}, \boldsymbol{\vartheta} \,|\, \boldsymbol{y}) \propto p(\boldsymbol{b})\, p(\boldsymbol{c} \,|\, \boldsymbol{b})\, p(\boldsymbol{\vartheta}) \prod_{n\in\mathcal{D}} p(x_n \,|\, \boldsymbol{c}_n) \prod_{n=0}^{N-1} p(y_n \,|\, x_n, \vartheta_n) \tag{3.111}$$

where $\boldsymbol{\vartheta} \triangleq (\vartheta_0, \ldots, \vartheta_{N-1})$. Furthermore, because of (3.108), the samples of the phase noise process form a Markov chain:

$$p(\boldsymbol{\vartheta}) = p(\vartheta_0) \prod_{n=1}^{N-1} p(\vartheta_n \,|\, \vartheta_{n-1}) \tag{3.112}$$

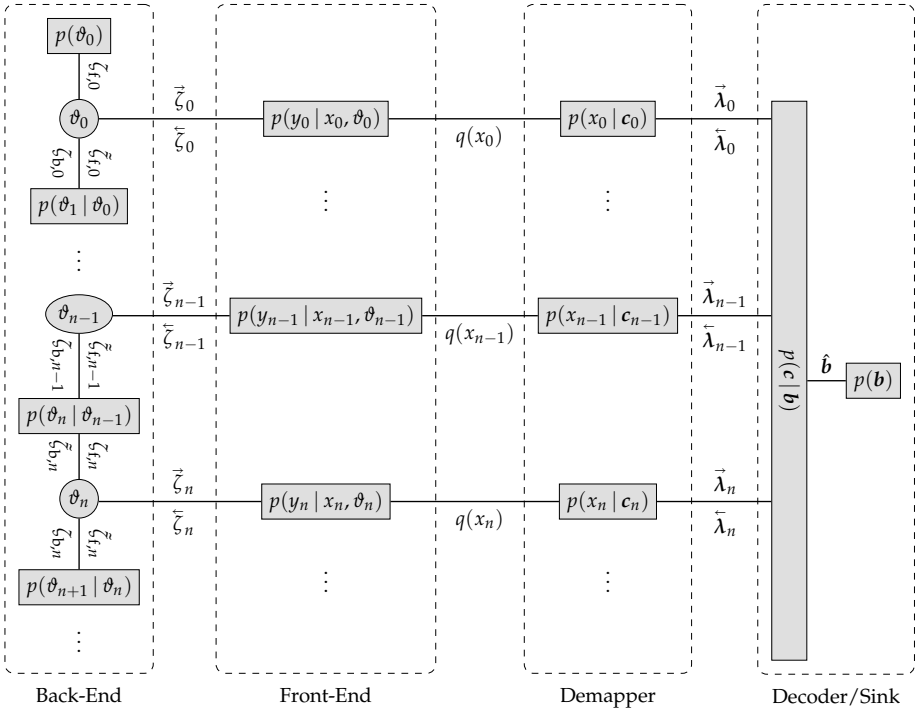


Figure 3.16: Factor graph of (3.113). Parameters above edges refer to forward messages, those below edges to backward messages

where $p(\vartheta_n \mid \vartheta_{n-1}) \propto \exp\left(-\frac{1}{2\sigma^2_{\Delta\vartheta}}(\vartheta_n - \vartheta_{n-1})^2\right)$. With the convention $p(\vartheta_0 \mid \vartheta_{-1}) = p(\vartheta_0)$, (3.111) becomes

$$p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x}_{\mathcal{D}}, \boldsymbol{\vartheta} \mid \boldsymbol{y}) \propto p(\boldsymbol{b})\, p(\boldsymbol{c} \mid \boldsymbol{b}) \prod_{n \in \mathcal{D}} p(x_n \mid \boldsymbol{c}_n) \prod_{n=0}^{N-1} p(\vartheta_n \mid \vartheta_{n-1})\, p(y_n \mid x_n, \vartheta_n). \tag{3.113}$$

The only difference to the factor graph from the previous section, shown in Figure 3.8, lies in the variable nodes for the phase offsets and the corresponding factor nodes which describe their statistics. While Figure 3.8 only contained a single variable node ϑ_0, Figure 3.16 contains separate variable nodes for each ϑ_n, $0 \leq n < N$, which are connected via the functions $p(\vartheta_n \mid \vartheta_{n-1})$ in a chain structure. The rest of the factor graph, particularly the functions $p(y_n \mid x_n, \vartheta_n)$ which connect the data symbols with the phase variables, is the same as in the previous section.

Note that the pilot symbols are not explicitly represented in Figure 3.16. For $n \in \mathcal{P}$, the function node $p(y_n \mid x_n, \vartheta_n)$ is of degree 1, since its only unknown variable is ϑ_n. It is only connected to the left, without requiring feedback from the right-hand part of the graph, and therefore the $\overleftarrow{\zeta}_n$, $n \in \mathcal{P}$, are already available in the initial receiver iteration.

3.5.2 Phase Noise Estimation

Figure 3.16 suggests a natural split of the phase noise estimator into two blocks, which, for lack of better phrasing, shall be called *front-end* and *back-end* in the following. The front-end consists of the functions $p(y_n \mid x_n, \vartheta_n)$, $0 \leq n < N$, and has a twofold task. In the feedback pass, it converts the information $q(x_n)$ about the data symbols, calculated via (3.59) or (3.61), into the *intrinsic phase information*[15] $\overleftarrow{\zeta}_n$ via (3.73) or (3.78), which is then fed into the back-end. In forward direction, it takes the *extrinsic phase information* $\vec{\zeta}_n$ from the back-end, splits it into $\vec{\kappa}_n = |\vec{\zeta}_n|$ and $\vec{\mu}_n = \arg \vec{\zeta}_n$, and computes the soft demapping metric (3.85).

The task of the back-end is somewhat similar to the channel decoder: it takes intrinsic information as input and computes extrinsic information. In the graphical model, it consists of the function $p(\vartheta_0, \ldots, \vartheta_{N-1})$ which describes the statistics of the phase noise process. Hence, different phase noise models give rise to different implementations of the back-end. The front-end, in contrast, is agnostic of the phase noise statistics.

In the following, we derive the back-end for the Wiener phase noise model (3.108). Due to the chain structure, we obtain a forward-backward algorithm which is very similar to the Kalman smoother. We will describe the forward recursion in detail; the backward recursion works analogously because of the symmetry of the functions $p(\vartheta_n \mid \vartheta_{n-1})$. Please refer to Figure 3.16 for the meaning of the various ζ-parameters.

The parameter $\zeta_{\mathrm{f},0}$ is initialized by the prior information $p(\vartheta_0)$, which could be obtained from the previous data block. If no prior information is available, $p(\vartheta_0)$ is a uniform distribution and $\zeta_{\mathrm{f},0} = 0$. At the variable node ϑ_0, the intrinsic information from the front-end is incorporated by simply adding the incoming $\overleftarrow{\zeta}_0$. In the terminology of Kalman filtering, this operation is called the *measurement update* (MU). The *time update* (TU), which corresponds to the operation at the $p(\vartheta_n \mid \vartheta_{n-1})$ nodes, is more complicated. A literal application of (B.81) yields

$$m_{p(\vartheta_1 \mid \vartheta_0) \to \vartheta_1}(\vartheta_1) \propto \int_{-\pi}^{\pi} p(\vartheta_1 \mid \vartheta_0)\, m_{\vartheta_0 \to p(\vartheta_1 \mid \vartheta_0)}(\vartheta_0)\, \mathrm{d}\vartheta_0 \tag{3.114}$$

$$\propto \int_{-\pi}^{\pi} \exp\left(-\frac{1}{2\sigma_{\Delta\vartheta}^2}(\vartheta_1 - \vartheta_0)^2\right) \exp\left(\mathrm{Re}\left(\tilde{\zeta}_{\mathrm{f},0}^{*} \mathrm{e}^{\mathrm{j}\vartheta_0}\right)\right) \mathrm{d}\vartheta_0 \tag{3.115}$$

[15] Note that the terminology of *intrinsic* and *extrinsic* phase information is analogous to that of intrinsic and extrinsic L-values as introduced in the textbox on page 17, but with reversed arrows: $\overleftarrow{\zeta}_n$ describes the information about ϑ_n that is contained in the nth observation y_n, while $\vec{\zeta}_n$ describes the information contained in all other observations.

which is not a von Mises distribution anymore. Fortunately, there is a simple trick to circumvent this problem (which has been independently proposed in [22]): for large enough $|\zeta|$, a von Mises distribution is virtually identical to a Gaussian distribution with mean $\arg\zeta$ and variance $|\zeta|^{-1}$. We therefore approximate the incoming message $m_{\vartheta_0 \to p(\vartheta_1 \,|\, \vartheta_0)}(\vartheta_0)$ with $\mathcal{N}(\vartheta_0 \mid \arg\tilde{\zeta}_{\mathrm{f},0}, |\tilde{\zeta}_{\mathrm{f},0}|^{-1})$. Since the convolution of two Gaussian functions is another Gaussian, where the means and variances simply add up, this yields the outgoing message $m_{p(\vartheta_1 \,|\, \vartheta_0) \to \vartheta_1}(\vartheta_1) \propto \mathcal{N}(\vartheta_1 \mid \arg\tilde{\zeta}_{\mathrm{f},0}, |\tilde{\zeta}_{\mathrm{f},0}|^{-1} + \sigma^2_{\Delta\vartheta})$. Subsequently, this Gaussian message is approximated back to a von Mises distribution with parameter $\zeta_{\mathrm{f},1}$, where $|\zeta_{\mathrm{f},1}| = \left(|\tilde{\zeta}_{\mathrm{f},0}|^{-1} + \sigma^2_{\Delta\vartheta}\right)^{-1}$ and $\arg\zeta_{\mathrm{f},1} = \arg\tilde{\zeta}_{\mathrm{f},0}$:

$$\begin{aligned}\zeta_{\mathrm{f},1} &= \frac{1}{|\tilde{\zeta}_{\mathrm{f},0}|^{-1} + \sigma^2_{\Delta\vartheta}} \, \mathrm{e}^{\mathrm{j}\arg\tilde{\zeta}_{\mathrm{f},0}} \\ &= \frac{|\tilde{\zeta}_{\mathrm{f},0}| \, \mathrm{e}^{\mathrm{j}\arg\tilde{\zeta}_{\mathrm{f},0}}}{1 + |\tilde{\zeta}_{\mathrm{f},0}| \, \sigma^2_{\Delta\vartheta}} \\ &= \frac{\tilde{\zeta}_{\mathrm{f},0}}{1 + |\tilde{\zeta}_{\mathrm{f},0}| \, \sigma^2_{\Delta\vartheta}}. \end{aligned} \tag{3.116}$$

The time update has a simple interpretation: since the innovation sequence $\Delta\vartheta_n$ has zero-mean, it leaves the argument of ζ (the mean of ϑ, i. e., the classical phase estimate) unaltered, but it reduces the modulus of ζ (the reliability of the estimate) by a factor that depends on the variance of the phase noise process $\Delta\vartheta_n$.

One step of the forward recursion can thus be summarized as follows:

$$\tilde{\zeta}_{\mathrm{f},n} = \zeta_{\mathrm{f},n} + \overleftarrow{\zeta}_n \quad \text{(MU)} \tag{3.117}$$

$$\zeta_{\mathrm{f},n+1} = \frac{\tilde{\zeta}_{\mathrm{f},n}}{1 + |\tilde{\zeta}_{\mathrm{f},n}| \, \sigma^2_{\Delta\vartheta}} \quad \text{(TU)}. \tag{3.118}$$

As mentioned above, the backward recursion proceeds analogously. Finally, the extrinsic phase information is obtained as

$$\vec{\zeta}_n = \zeta_{\mathrm{f},n} + \zeta_{\mathrm{b},n}. \tag{3.119}$$

Note that the phase estimator from the previous section is a special case of the Wiener phase noise estimator. A constant phase process is obtained from (3.108) by setting $\sigma^2_{\Delta\vartheta} = 0$. Then, the time update (3.118) reduces to $\zeta_{\mathrm{f},n+1} = \tilde{\zeta}_{\mathrm{f},n}$, and it is easy to show that

$$\vec{\zeta}_n = \zeta_{\mathrm{f},0} + \sum_{n' \neq n} \overleftarrow{\zeta}_{n'} \tag{3.120}$$

which is the same as (3.82) with (3.79), where $\zeta_{\mathrm{f},0}$ now takes the role of the prior phase information ζ_{a}.

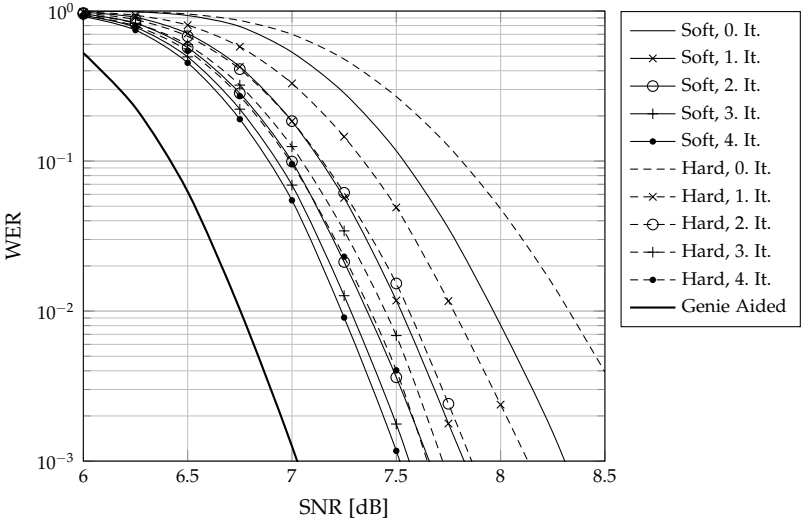


Figure 3.17: Comparison of soft and hard phase estimation

3.5.3 Numerical Results

In this section, we use the LDPC code from the IEEE 802.11n standard with rate 1/2 and a block length of 1944 codebits. A Gray-mapped 16-QAM modulation scheme is used, and pilot symbols are periodically multiplexed into the data stream, with a pilot spacing of 10 symbols. In order to clearly point out the differences between the algorithmic choices, we set $\sigma_{\Delta\vartheta} = 5^\circ$, resulting in very strong phase noise.

The Figures 3.17–3.19 present the same comparisons as Figures 3.10–3.12 from the previous section. The reason for repeating essentially the same simulations is that the results are far more pronounced in the context of heavy phase noise and hence less reliable phase estimates.

Figure 3.17 clearly shows the advantage of soft phase estimation over the classical hard estimate as obtained e. g. by the EM algorithm. Figure 3.18 confirms (a) the significant advantage of posterior vs extrinsic feedback, and (b) that posterior L-values can be used instead of (3.59) without noteworthy performance loss. In Figure 3.19 we finally see that the two ways of converting the feedback $q(x_n)$ to the intrinsic phase information $\overleftarrow{\zeta}_n$, based on exclusive and inclusive DM, hardly differ as long as $q(x_n)$ is based on posterior information, so the computationally simpler (3.73) can be used without performance loss. They *do* differ for purely extrinsic feedback—in this case, using the inclusive (3.78) yields a quite significant performance gain over the

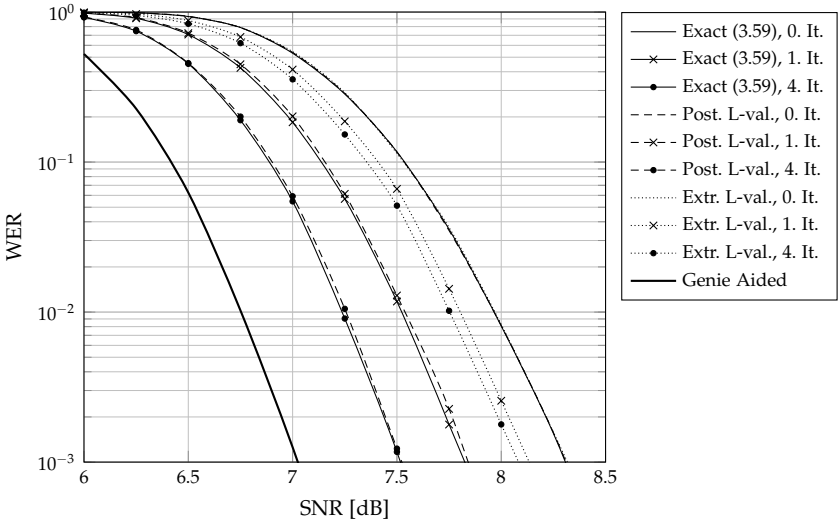


Figure 3.18: Comparison of different ways to compute the feedback $q(\boldsymbol{x}_{\mathcal{D}})$

exclusive (3.73). But of course, this result is not of much practical relevance, since, as we have seen several times now, extrinsic feedback is the wrong choice to begin with.

Note that the combination of inclusive DM and extrinsic L-values (shown with dash-dotted plots in Figure 3.19) corresponds to the algorithm from Colavolpe *et al* [22].

Extrinsic vs Posterior Phase Information What we have not yet compared in the previous section is the usage of extrinsic vs posterior phase information in the demapper. The reason is that dropping a single term in the sum (3.79) does not yield any appreciable difference, even for such a short block length as used above. In the phase noise scenario considered here, however, the coherence time of ϑ is rather short, which makes the extrinsic phase information $\vec{\zeta}$ weaker and the difference between extrinsic and posterior information ($\zeta = \vec{\zeta} + \overleftarrow{\zeta}$) much more pronounced.

The results in Figure 3.20 clearly show the superiority of extrinsic phase information. Since the demapper is derived from standard belief propagation, i. e., inclusive DM, this is exactly what we have expected. This experiment again confirms the practical value of the generic DM approach, from which we have systematically obtained the whole receiver structure which uses *extrinsic* phase information in the demapper, but which feeds *posterior* information about the data symbols back into the phase estimator.

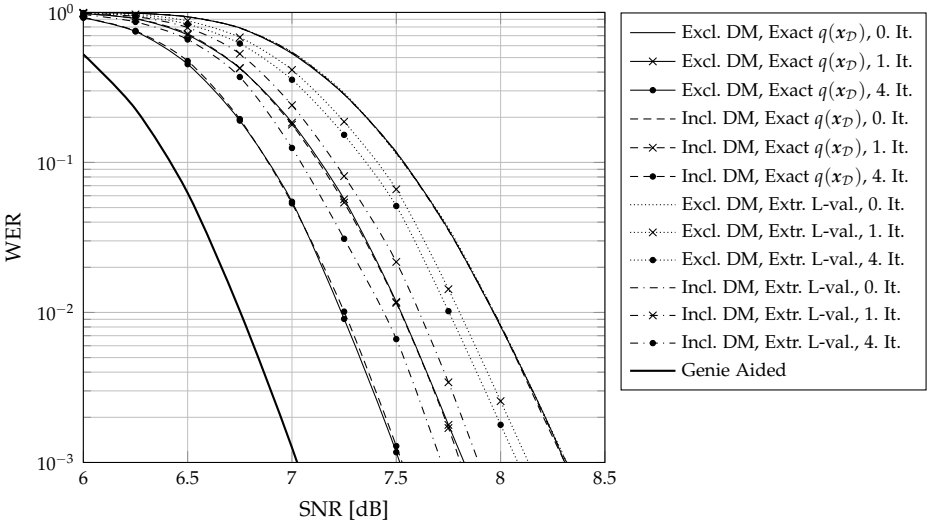


Figure 3.19: Comparison of exclusive (3.73) and inclusive (3.78) DM

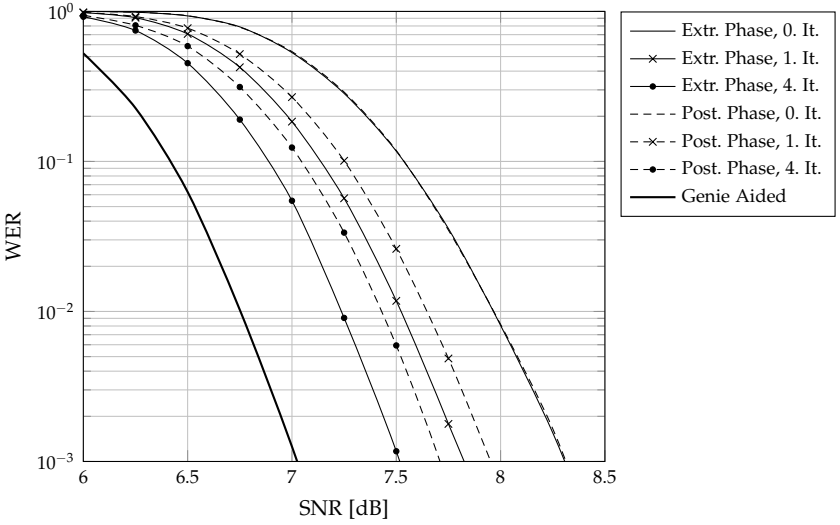
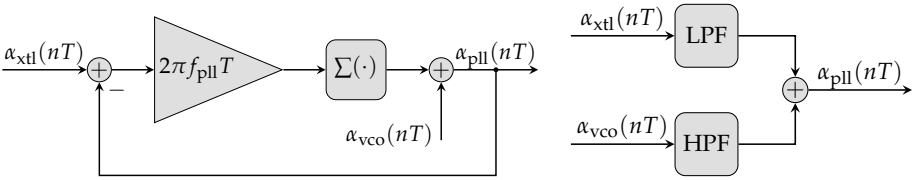


Figure 3.20: Comparison of extrinsic ($\vec{\zeta}$) and posterior ($\zeta = \vec{\zeta} + \overleftarrow{\zeta}$) phase information



(a) Discrete-time first order PLL, adapted from [14] **(b)** Simplified block diagram

Figure 3.21: Block diagrams of a first order phase-locked loop

3.6 General Phase Noise Estimation

The phase noise estimator that we have developed in the previous section is based on the assumption of Wiener phase noise. For free running voltage-controlled oscillators (VCOs), the random-walk model (3.108) is quite accurate [31], but it is too simplistic for many other, practically relevant sources of phase noise. In this section, we finish the discussion of phase noise estimation by generalizing the proposed estimator to linear Gaussian state space models, which are expressive enough to describe a large class of phase noise processes. Phase-locked loops (PLLs) are a particularly important example and will serve as the basis for the following derivation. In order to demonstrate the generality of the proposed estimator, we will briefly consider a different scenario in Section 3.6.4, namely a joint estimation of phase noise and a residual frequency offset.

3.6.1 System Model

The general system setup is the same as in the previous section, the only difference lies in the stochastic model of the phase noise process. In order to keep the derivation simple, we will limit ourselves to first order PLLs. Once the basic principles are developed, extensions to higher order PLLs or other state space models are fairly straightforward.

The following discussion is based on the analysis of PLLs presented in [87] and the derivation of corresponding discrete-time models given in [14]. A general PLL consists of a low-noise reference crystal, abbreviated as XTL, and a voltage controlled oscillator, VCO, with parameters c_{xtl} and c_{vco}, respectively. The output of both oscillators is modeled as Wiener phase noise, i. e., as random walks (3.108) in the discrete-time domain. A phase detector compares the output of the PLL with the reference crystal. The phase difference may be low-pass filtered, and the result serves as input to the VCO.

Figure 3.21a shows a block diagram of a first order PLL (i. e., without a low-pass filter), which is a discrete-time and simplified version of [14, Figure 2b]. Since the low-pass filter h_{LP} is missing, the phase detector constant k_{pd} and the square root of the time constant of the VCO's control node $\sqrt{c_{\mathrm{contr}}}$ are merged into one multiplier $k_{\mathrm{pd}}\sqrt{c_{\mathrm{contr}}} = 2\pi f_{\mathrm{pll}}$. Furthermore, the integrator $\int(\cdot)\,\mathrm{d}t$ is converted to an accumulator

$T\sum(\cdot)$, and the constant T is also merged into the multiplier. The system is linear with an open-loop transfer function $2\pi f_{\text{pll}}T/(z-1)$, so it can be described in the z-domain as

$$A_{\text{pll}}(z) = G_{\text{xtl}}(z)A_{\text{xtl}}(z) + G_{\text{vco}}(z)A_{\text{vco}}(z) \tag{3.121}$$

with the transfer functions

$$G_{\text{xtl}}(z) = \frac{2\pi f_{\text{pll}}T}{z-(1-2\pi f_{\text{pll}}T)} \quad \text{and} \quad G_{\text{vco}}(z) = \frac{z-1}{z-(1-2\pi f_{\text{pll}}T)}. \tag{3.122}$$

Thus, as shown in Figure 3.21b, the time shift processes α of the reference crystal and the VCO are low-pass and high-pass filtered, respectively, both with the same 3 dB corner frequency f_{pll}. For $f > f_{\text{pll}}$, however, the crystal's PSD is negligible anyway compared to the PSD of the VCO, so the low-pass filter can be neglected and we obtain the approximation

$$A_{\text{pll}}(z) \approx A_{\text{xtl}}(z) + \frac{z-1}{z-z_\infty}\,A_{\text{vco}}(z) \quad \text{with} \quad z_\infty \triangleq 1-2\pi f_{\text{pll}}T. \tag{3.123}$$

Since both $\alpha_{\text{xtl}}(nT)$ and $\alpha_{\text{vco}}(nT)$ are random walk processes, their z-transforms are $A_{\text{vco}}(z) = \frac{z}{z-1}\,\Delta A_{\text{vco}}(z)$, where $\Delta A_{\text{vco}}(z)$ is the z-transform of the white innovation sequence, and analogously for $A_{\text{xtl}}(z)$. Thus,

$$A_{\text{pll}}(z) \approx \frac{z}{z-1}\Delta A_{\text{xtl}}(z) + \frac{z}{z-z_\infty}\,\Delta A_{\text{vco}}(z). \tag{3.124}$$

Transforming this expression back to the discrete time domain, and converting the time shift $\alpha_{\text{pll}}(nT)$ into the phase noise process $\vartheta_n = 2\pi f_{\text{c}}\,\alpha_{\text{pll}}(nT)$, yields

$$\vartheta_n = \vartheta_{\text{xtl},n} + \vartheta_{\text{vco},n} \tag{3.125}$$

which is the superposition of two uncorrelated processes: the random walk

$$\vartheta_{\text{xtl},n} = \vartheta_{\text{xtl},n-1} + \Delta\vartheta_{\text{xtl},n} \tag{3.126}$$

whose variance increases linearly with time, and the zero-mean process

$$\vartheta_{\text{vco},n} = z_\infty\vartheta_{\text{vco},n-1} + \Delta\vartheta_{\text{vco},n} \tag{3.127}$$

with constant variance

$$\text{var}\,[\vartheta_{\text{vco}}] = \frac{(2\pi f_{\text{c}})^2 c_{\text{vco}}T}{1-z_\infty^2}. \tag{3.128}$$

Figure 3.22 shows the one-sided PSD of an exemplary PLL phase noise process, with the same parameters as the two Wiener phase noise processes from Figure 3.15. The PSD can roughly be divided into three segments: it starts with a slope of –20 dB/decade, followed by a flat region, followed by another –20 dB/decade slope. This shape, which

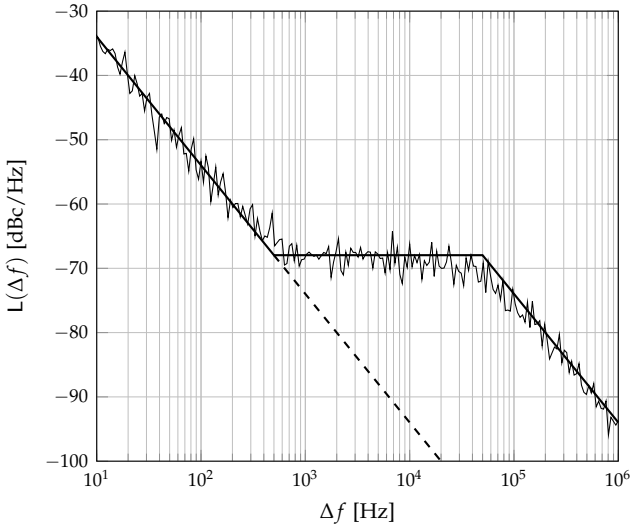


Figure 3.22: Power spectral density of a simulated first-order PLL phase noise process, with $f_c = 200\,\text{MHz}$, $T^{-1} = 10\,\text{MHz}$, $c_{\text{xtl}} = 10^{-18}\,\text{s}$, $c_{\text{vco}} = 10^{-14}\,\text{s}$ and $f_{\text{pll}} = 50\,\text{kHz}$, along with the analytic asymptotes (thick lines)

is characteristic for PLL phase noise processes [87], can be explained as follows. The VCO process is high-pass filtered, so for $f > f_{\text{pll}} = 50\,\text{kHz}$, the PSDs of the PLL process and the free-running VCO process from Figure 3.15 are essentially equal. Below f_{pll} the PSD becomes flat, because the slope of $-20\,\text{dB/decade}$ is cancelled by the first-order high-pass filter. Finally, for $f < \sqrt{c_{\text{xtl}}/c_{\text{vco}}}\, f_{\text{pll}} = 500\,\text{Hz}$, the phase noise from the reference crystal becomes dominant. Notice that at the corner frequency f_{pll}, the PSD of the crystal is, in this example, 40 dB below the VCO, which justifies that we have disregarded the low-pass filter that acts on the crystal's phase noise process.

State Space Model

In order to adapt the factor graph from the previous section to the PLL phase noise model, we need a suitable factorization of $p(\boldsymbol{\vartheta})$. Due to the more involved statistical model, where ϑ_n depends not only on ϑ_{n-1} but on *all* previous phase values, a direct factorization of $p(\boldsymbol{\vartheta})$ would look like

$$p(\vartheta_0, \ldots, \vartheta_{N-1}) = \prod_{n=0}^{N-1} p\left(\vartheta_n \mid \vartheta_0, \ldots, \vartheta_{n-1}\right). \tag{3.129}$$

The corresponding graphical model is fully connected, leading (for realistically large N) to a tremendously complex phase noise estimator.

The standard trick to circumvent this problem is to introduce auxiliary state variables $\boldsymbol{S} \triangleq (\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{N-1})$, such that

$$p(\boldsymbol{\vartheta}) = \int p(\boldsymbol{\vartheta}, \boldsymbol{S}) \, \mathrm{d}\boldsymbol{S} \tag{3.130}$$

with a joint distribution $p(\boldsymbol{\vartheta}, \boldsymbol{S})$ that admits a simple factorization.

For our PLL phase noise model (3.125)–(3.127), a natural state vector $\boldsymbol{s}_n$ is

$$\boldsymbol{s}_n \triangleq \begin{pmatrix} \vartheta_{\mathrm{xtl},n} \\ \vartheta_{\mathrm{vco},n} \end{pmatrix} \tag{3.131}$$

which leads to the state space model

$$\boldsymbol{s}_n = \begin{pmatrix} 1 & 0 \\ 0 & z_\infty \end{pmatrix} \boldsymbol{s}_{n-1} + \boldsymbol{u}_n \tag{3.132}$$

$$\vartheta_n = \begin{pmatrix} 1 & 1 \end{pmatrix} \boldsymbol{s}_n \tag{3.133}$$

where the zero-mean white Gaussian process noise $\boldsymbol{u}_n$ has the covariance matrix

$$\boldsymbol{C}_{\boldsymbol{u}} = \mathbb{E}\left[\boldsymbol{u}_n \boldsymbol{u}_n^{\mathsf{T}}\right] = (2\pi f_{\mathrm{c}})^2 \begin{pmatrix} c_{\mathrm{xtl}} T & 0 \\ 0 & c_{\mathrm{vco}} T \end{pmatrix}. \tag{3.134}$$

The state vectors form a Markov chain, and ϑ_n only depends on $\boldsymbol{s}_n$, so the joint distribution of $\boldsymbol{\vartheta}$ and $\boldsymbol{S}$ factorizes as

$$p(\boldsymbol{\vartheta}, \boldsymbol{S}) = \prod_{n=0}^{N-1} p(\boldsymbol{s}_n \mid \boldsymbol{s}_{n-1}) \, p(\vartheta_n \mid \boldsymbol{s}_n) \tag{3.135}$$

with the convention $p(\boldsymbol{s}_0 \mid \boldsymbol{s}_{-1}) = p(\boldsymbol{s}_0)$.

A factor graph of the receiver is shown in 3.23.

3.6.2 Phase Noise Estimation

To keep the derivation general, we rewrite the state space model (3.132)–(3.133) as

$$\boldsymbol{s}_n = \boldsymbol{A}\boldsymbol{s}_{n-1} + \boldsymbol{u}_n \tag{3.136}$$

$$\vartheta_n = \boldsymbol{c}^{\mathsf{T}} \boldsymbol{s}_n \tag{3.137}$$

with some state-transition matrix $\boldsymbol{A}$ and output vector $\boldsymbol{c}^{\mathsf{T}}$ (we use the letter $\boldsymbol{c}$ because it is standard notation; it should not be confused with the vector of code bits).

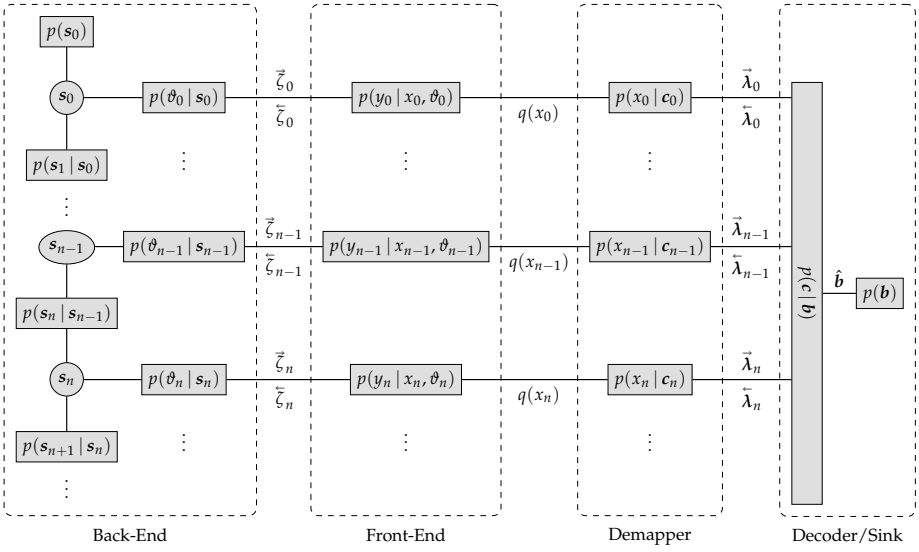


Figure 3.23: Factor graph as in Figure 3.16, but with the back-end of the phase noise estimator implemented according to (3.135)

Since we have a linear Gaussian state space model, the resulting phase noise estimator is almost an instance of the Kalman smoother. The important difference is that the two components of $\boldsymbol{s}$ contain *angular* variables and are therefore only defined up to integer multiples of 2π. This needs to be taken into account when merging two pieces of information; otherwise, averaging for example the two estimates $\hat{\vartheta}_1 \approx 0$ and $\hat{\vartheta}_2 \approx 2\pi$ would yield a total estimate of $\hat{\vartheta} \approx \pi$, which would clearly be wrong.

In the previous section, the estimator was based on von Mises distributions, which circumvented this problem in an elegant way because the averaging took place in the complex plane, in terms of the ζ-parameters. It may look tempting to try an analogous approach here. Since our state vector is two-dimensional, that would mean to track the distribution of $\boldsymbol{s}$ in terms of a bivariate von Mises distribution, rather than a bivariate Gaussian. There is indeed some recent interest in multivariate generalizations of the von Mises distribution (see [83] and references therein), but unfortunately the proposed distributions lack several important properties of the multivariate Gaussian distribution. For example, they are not closed under multiplication, so the measurement update, which involves the multiplication of distributions, must necessarily be approximated. I have tested several multidimensional von-Mises-based algorithms via simulations, but did not succeed in finding an estimator which performs even close to the Gaussian-based algorithm that will be derived in the following.

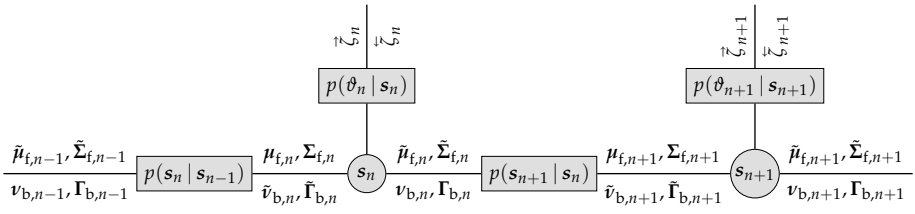


Figure 3.24: Detail of the factor graph from Figure 3.23

The proposed estimator performs Gaussian forward/backward message passing (i. e., Kalman smoothing) along the Markov chain of state vectors $\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{N-1}$. The interface between front-end and back-end of the phase estimator still uses von Mises distributions, i. e., intrinsic and extrinsic ζ-parameters, as shown in Figure 3.23. The conversion between the von Mises and the Gaussian distributions occurs in the $p(\vartheta_n \mid \boldsymbol{s}_n)$ function nodes.

For the purpose of circumventing the above described problem of averaging two phase estimates that might differ by an integer multiple of 2π, we define the function

$$\operatorname{unwrap}_{\phi}(\vartheta) \triangleq \vartheta - 2\pi \left\lceil \frac{\vartheta - \phi}{2\pi} \right\rfloor \tag{3.138}$$

where the square brackets denote rounding to the nearest integer. In words, $\operatorname{unwrap}_{\phi}(\vartheta)$ returns $\vartheta + 2\pi k$, where $k \in \mathbb{Z}$ is chosen such that $\vartheta + 2\pi k \in [\phi - \pi, \phi + \pi)$. We will drop the subscript ϕ if the reference angle is clear from the context.

The back-end of the phase estimator consists of three steps: the forward and backward recursion, and finally the computation of the output, namely the extrinsic phase information. Figure 3.24 shows a detail of the factor graph from Figure 3.23, defining the notation that will be used in the following. The quantities above the edges refer to forward messages, the ones below the edges refer to backward messages.

Forward Recursion

The forward recursion consists of repeated applications of measurement update (MU) and time update (TU). At the nth time instant, starting with the mean vector $\boldsymbol{\mu}_{\mathrm{f},n}$ and the covariance matrix $\boldsymbol{\Sigma}_{\mathrm{f},n}$, we first incorporate the intrinsic phase information $\overleftarrow{\zeta}_n$, yielding intermediate parameters $\tilde{\boldsymbol{\mu}}_{\mathrm{f},n}$ and $\tilde{\boldsymbol{\Sigma}}_{\mathrm{f},n}$. Those are subsequently transformed to $\boldsymbol{\mu}_{\mathrm{f},n+1}$ and $\boldsymbol{\Sigma}_{\mathrm{f},n+1}$, which concludes one step of the forward recursion.

Measurement Update The MU is implemented by the standard message update equation at variable nodes:

$$m_{\boldsymbol{s}_n \to p(\boldsymbol{s}_{n+1} \mid \boldsymbol{s}_n)}(\boldsymbol{s}_n) \propto m_{p(\boldsymbol{s}_n \mid \boldsymbol{s}_{n-1}) \to \boldsymbol{s}_n}(\boldsymbol{s}_n)\, m_{p(\vartheta_n \mid \boldsymbol{s}_n) \to \boldsymbol{s}_n}(\boldsymbol{s}_n). \tag{3.139}$$

The first factor is the forward message $m_{p(\boldsymbol{s}_n \mid \boldsymbol{s}_{n-1}) \to \boldsymbol{s}_n}(\boldsymbol{s}_n) \propto \mathcal{N}(\boldsymbol{s}_n \mid \boldsymbol{\mu}_{\mathrm{f},n}, \boldsymbol{\Sigma}_{\mathrm{f},n})$. The second one is given by the intrinsic phase information as

$$\begin{aligned} m_{p(\vartheta_n \mid \boldsymbol{s}_n) \to \boldsymbol{s}_n}(\boldsymbol{s}_n) &\propto \int p(\vartheta_n \mid \boldsymbol{s}_n)\, \mathcal{M}(\vartheta_n \mid \overleftarrow{\zeta}_n)\, \mathrm{d}\vartheta_n \\ &\propto \int \delta(\vartheta_n - \boldsymbol{c}^{\mathsf{T}} \boldsymbol{s}_n)\, \exp\left(|\overleftarrow{\zeta}_n| \cos(\vartheta_n - \arg \overleftarrow{\zeta}_n) \right) \mathrm{d}\vartheta_n \end{aligned} \tag{3.140}$$

which is non-Gaussian, but can be approximated as a Gaussian by substituting

$$\begin{aligned} \cos(\vartheta_n - \arg \overleftarrow{\zeta}_n) &\overset{(a)}{=} \cos\left(\vartheta_n - \operatorname{unwrap}(\arg \overleftarrow{\zeta}_n) \right) \\ &\overset{(b)}{\approx} -\frac{1}{2} \left(\vartheta_n - \operatorname{unwrap}(\arg \overleftarrow{\zeta}_n) \right)^2 + \text{const.} \end{aligned} \tag{3.141}$$

Above, (b) uses the Taylor series approximation $\cos(x) \approx 1 - x^2/2$ around $x = 0$, where the additive constant 1 in the argument of the exp-function becomes an irrelevant multiplicative constant outside exp. Before that, the unwrap-function introduced in (a), which just adds an integer multiple of 2π to its argument, ensures that the argument of the cosine lies within the interval $(-\pi, \pi]$. Therefore, the above approximation is valid for any ϑ_n and $\arg \overleftarrow{\zeta}_n$, even if they are some 2π apart.[16]

Substituting (3.141) into (3.140) yields

$$\begin{aligned} m_{p(\vartheta_n \mid \boldsymbol{s}_n) \to \boldsymbol{s}_n}(\boldsymbol{s}_n) &\propto \exp\left(-\frac{|\overleftarrow{\zeta}_n|}{2} \left(\boldsymbol{c}^{\mathsf{T}} \boldsymbol{s}_n - \operatorname{unwrap}(\arg \overleftarrow{\zeta}_n) \right)^2 \right) \\ &\propto \exp\left(-\frac{1}{2} \boldsymbol{s}_n^{\mathsf{T}} \boldsymbol{c}\, |\overleftarrow{\zeta}_n|\, \boldsymbol{c}^{\mathsf{T}} \boldsymbol{s}_n + \boldsymbol{s}_n^{\mathsf{T}} \boldsymbol{c}\, |\overleftarrow{\zeta}_n| \operatorname{unwrap}(\arg \overleftarrow{\zeta}_n) \right) \end{aligned} \tag{3.142}$$

which is a degenerate Gaussian with rank-1 inverse covariance matrix

$$\overleftarrow{\boldsymbol{\Sigma}}_n^{-1} = \boldsymbol{c}\, |\overleftarrow{\zeta}_n|\, \boldsymbol{c}^{\mathsf{T}} \tag{3.143}$$

and scaled mean vector

$$\overleftarrow{\boldsymbol{\Sigma}}_n^{-1} \overleftarrow{\boldsymbol{\mu}}_n = \boldsymbol{c}\, |\overleftarrow{\zeta}_n| \operatorname{unwrap}(\arg \overleftarrow{\zeta}_n). \tag{3.144}$$

We can now carry out the computation in (3.139). The multiplication of two exponential family distributions boils down to adding their natural parameters, which in case of Gaussians are the inverse covariance matrix and the scaled mean vector. Thus,

$$\tilde{\boldsymbol{\Sigma}}_{\mathrm{f},n} = \left(\boldsymbol{\Sigma}_{\mathrm{f},n}^{-1} + \boldsymbol{c}\, |\overleftarrow{\zeta}_n|\, \boldsymbol{c}^{\mathsf{T}} \right)^{-1}$$

[16] If the argument of the cosine is somewhere near $-\pi$ or π after resolving the 2π-ambiguity, approximation (3.141) is of course still rather poor. This is an unavoidable error that we introduce by replacing von Mises with Gaussian distributions, and that we just have to accept.

$$\overset{(a)}{=} \boldsymbol{\Sigma}_{\mathrm{f},n} - \frac{|\overleftarrow{\zeta}_n|\,\boldsymbol{\Sigma}_{\mathrm{f},n}\boldsymbol{c}\,\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathrm{f},n}}{1+|\overleftarrow{\zeta}_n|\,\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathrm{f},n}\boldsymbol{c}} \tag{3.145}$$

where (a) uses the matrix inversion lemma, and

$$\begin{aligned}\tilde{\boldsymbol{\mu}}_{\mathrm{f},n} &= \tilde{\boldsymbol{\Sigma}}_{\mathrm{f},n}\left(\boldsymbol{\Sigma}_{\mathrm{f},n}^{-1}\boldsymbol{\mu}_{\mathrm{f},n} + \boldsymbol{c}\,|\overleftarrow{\zeta}_n|\,\mathrm{unwrap}(\arg\overleftarrow{\zeta}_n)\right)\\ &= \boldsymbol{\mu}_{\mathrm{f},n} - \frac{|\overleftarrow{\zeta}_n|\,\boldsymbol{\Sigma}_{\mathrm{f},n}\boldsymbol{c}\,\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\mu}_{\mathrm{f},n}}{1+|\overleftarrow{\zeta}_n|\,\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\Sigma}_{\mathrm{f},n}\boldsymbol{c}} + \tilde{\boldsymbol{\Sigma}}_{\mathrm{f},n}\boldsymbol{c}\,|\overleftarrow{\zeta}_n|\,\mathrm{unwrap}(\arg\overleftarrow{\zeta}_n)\end{aligned} \tag{3.146}$$

where the reference angle for the unwrap-operation is set to the phase estimate $\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\mu}_{\mathrm{f},n}$.

Time Update The derivation of the time update is a lot simpler. From the state transition equation (3.136) it follows immediately that

$$\boldsymbol{\mu}_{\mathrm{f},n+1} = \boldsymbol{A}\tilde{\boldsymbol{\mu}}_{\mathrm{f},n} \tag{3.147}$$

$$\boldsymbol{\Sigma}_{\mathrm{f},n+1} = \boldsymbol{A}\tilde{\boldsymbol{\Sigma}}_{\mathrm{f},n}\boldsymbol{A}^{\mathsf{T}} + \boldsymbol{C}_{\boldsymbol{u}}. \tag{3.148}$$

Initialization The remaining question is how to initialize $\boldsymbol{\mu}_{\mathrm{f},0}$ and $\boldsymbol{\Sigma}_{\mathrm{f},0}$. In a consecutive transmission of several codewords, the obvious choice are the $\boldsymbol{\mu}_{\mathrm{f},N-1}$ and $\boldsymbol{\Sigma}_{\mathrm{f},N-1}$ from the previous codeword. In the lack of prior information, the initialization depends on the particular state space model. Since in our PLL model, the state vector is defined as $\boldsymbol{s}_n = (\vartheta_{\mathrm{xtl},n}, \vartheta_{\mathrm{vco},n})^{\mathsf{T}}$, reasonable choices are $\boldsymbol{\mu}_{\mathrm{f},0} = \boldsymbol{0}$ and

$$\boldsymbol{\Sigma}_{\mathrm{f},0} = \begin{pmatrix} 100 & 0 \\ 0 & \mathrm{var}\,[\vartheta_{\mathrm{vco}}] \end{pmatrix} \tag{3.149}$$

which have been used in the simulation whose result is presented below. Above, $\mathrm{var}\,[\vartheta_{\mathrm{vco}}]$ is given in (3.128), and the value of 100 is a quite arbitrary number, large enough to yield approximately a uniform distribution for $\vartheta_{\mathrm{xtl},0}$ over the interval $[-\pi, \pi)$. The precise value does not matter, because $\tilde{\boldsymbol{\Sigma}}_{\mathrm{f},0}$ after the first measurement update is practically independent of it, anyway.

Backward Recursion

The backward recursion proceeds similarly by alternating measurement and time updates. However, for numerical reasons it is advantageous to describe those updates not in terms of the mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$, but instead using the scaled mean vector $\boldsymbol{\nu} \triangleq \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$ and the inverse covariance matrix $\boldsymbol{\Gamma} \triangleq \boldsymbol{\Sigma}^{-1}$, i. e., the natural parameters of the Gaussian distribution.

Measurement Update This time, the measurement update is the simpler one, because the natural parameters simply add up at the variable node $\boldsymbol{s}_n$. Given $\boldsymbol{\nu}_{\mathrm{b},n}$ and $\boldsymbol{\Gamma}_{\mathrm{b},n}$, the temporary parameters $\tilde{\boldsymbol{\nu}}_{\mathrm{b},n}$ and $\tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n}$ are

$$\tilde{\boldsymbol{\nu}}_{\mathrm{b},n} = \boldsymbol{\nu}_{\mathrm{b},n} + \boldsymbol{c}\,|\overleftarrow{\zeta}_n|\,\mathrm{unwrap}(\arg\overleftarrow{\zeta}_n) \tag{3.150}$$

$$\tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n} = \boldsymbol{\Gamma}_{\mathrm{b},n} + \boldsymbol{c}\,|\overleftarrow{\zeta}_n|\,\boldsymbol{c}^{\mathsf{T}} \tag{3.151}$$

where we have used the natural parameterization of the intrinsic information (3.143) and (3.144). As reference angle for the unwrap-operation, we use the forward phase estimate $\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\mu}_{\mathrm{f},n}$, which has the additional advantage that later, when forward and backward messages are combined to yield the extrinsic phase information, there is no 2π-ambiguity between the forward and backward messages.

Time Update There is no such shortcut for the time update equation, so we derive it from first principles:

$$\begin{aligned}
&m_{p(\boldsymbol{s}_n\,|\,\boldsymbol{s}_{n-1})\to\boldsymbol{s}_{n-1}}(\boldsymbol{s}_{n-1}) \propto \int p(\boldsymbol{s}_n\,|\,\boldsymbol{s}_{n-1})\, m_{\boldsymbol{s}_n\to p(\boldsymbol{s}_n\,|\,\boldsymbol{s}_{n-1})}(\boldsymbol{s}_n)\,\mathrm{d}\boldsymbol{s}_n \\
&\propto \int \exp\left(-\tfrac{1}{2}(\boldsymbol{s}_n - \boldsymbol{A}\boldsymbol{s}_{n-1})^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1}(\boldsymbol{s}_n - \boldsymbol{A}\boldsymbol{s}_{n-1}) - \tfrac{1}{2}\boldsymbol{s}_n^{\mathsf{T}}\tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n}\boldsymbol{s}_n + \boldsymbol{s}_n^{\mathsf{T}}\tilde{\boldsymbol{\nu}}_{\mathrm{b},n}\right)\mathrm{d}\boldsymbol{s}_n \\
&\propto \mathrm{e}^{-\frac{1}{2}\boldsymbol{s}_{n-1}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1}\boldsymbol{A}\boldsymbol{s}_{n-1}} \int \exp\left(-\tfrac{1}{2}\boldsymbol{s}_n^{\mathsf{T}}\left(\boldsymbol{C}_{\boldsymbol{u}}^{-1} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n}\right)\boldsymbol{s}_n + \boldsymbol{s}_n^{\mathsf{T}}\left(\boldsymbol{C}_{\boldsymbol{u}}^{-1}\boldsymbol{A}\boldsymbol{s}_{n-1} + \tilde{\boldsymbol{\nu}}_{\mathrm{b},n}\right)\right)\mathrm{d}\boldsymbol{s}_n \\
&\overset{(a)}{\propto} \exp\Big(-\tfrac{1}{2}\boldsymbol{s}_{n-1}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1}\boldsymbol{A}\boldsymbol{s}_{n-1} \\
&\qquad + \tfrac{1}{2}\left(\boldsymbol{s}_{n-1}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1} + \tilde{\boldsymbol{\nu}}_{\mathrm{b},n}^{\mathsf{T}}\right)\left(\boldsymbol{C}_{\boldsymbol{u}}^{-1} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n}\right)^{-1}\left(\boldsymbol{C}_{\boldsymbol{u}}^{-1}\boldsymbol{A}\boldsymbol{s}_{n-1} + \tilde{\boldsymbol{\nu}}_{\mathrm{b},n}\right)\Big) \\
&\propto \exp\Big(-\tfrac{1}{2}\boldsymbol{s}_{n-1}^{\mathsf{T}}\left(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1}\boldsymbol{A} - \boldsymbol{A}^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1}\left(\boldsymbol{C}_{\boldsymbol{u}}^{-1} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n}\right)^{-1}\boldsymbol{C}_{\boldsymbol{u}}^{-1}\boldsymbol{A}\right)\boldsymbol{s}_{n-1} \\
&\qquad + \boldsymbol{s}_{n-1}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1}\left(\boldsymbol{C}_{\boldsymbol{u}}^{-1} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n}\right)^{-1}\tilde{\boldsymbol{\nu}}_{\mathrm{b},n}\Big) \\
&= \exp\left(-\tfrac{1}{2}\boldsymbol{s}_{n-1}^{\mathsf{T}}\boldsymbol{\Gamma}_{\mathrm{b},n-1}\boldsymbol{s}_{n-1} + \boldsymbol{s}_{n-1}^{\mathsf{T}}\boldsymbol{\nu}_{\mathrm{b},n-1}\right)
\end{aligned} \tag{3.152}$$

where (a) uses the identity

$$\int_{\mathbb{R}^N} \exp\left(-\tfrac{1}{2}\boldsymbol{x}^{\mathsf{T}}\boldsymbol{\Gamma}\boldsymbol{x} + \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\nu}\right)\mathrm{d}\boldsymbol{x} = \sqrt{\frac{(2\pi)^N}{\det\boldsymbol{\Gamma}}}\,\exp\left(\tfrac{1}{2}\boldsymbol{\nu}^{\mathsf{T}}\boldsymbol{\Gamma}^{-1}\boldsymbol{\nu}\right) \tag{3.153}$$

which can be derived from the multivariate Gaussian PDF. The parameters of the message after the time update can be read off as

$$\boldsymbol{\nu}_{\mathrm{b},n-1} = \boldsymbol{A}^{\mathsf{T}}\boldsymbol{C}_{\boldsymbol{u}}^{-1}\left(\boldsymbol{C}_{\boldsymbol{u}}^{-1} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n}\right)^{-1}\tilde{\boldsymbol{\nu}}_{\mathrm{b},n}$$

$$= \boldsymbol{A}^{\mathsf{T}} \left(\boldsymbol{I} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n} \boldsymbol{C}_{\boldsymbol{u}} \right)^{-1} \tilde{\boldsymbol{\nu}}_{\mathrm{b},n} \tag{3.154}$$

and

$$\begin{aligned} \boldsymbol{\Gamma}_{\mathrm{b},n-1} &= \boldsymbol{A}^{\mathsf{T}} \boldsymbol{C}_{\boldsymbol{u}}^{-1} \boldsymbol{A} - \boldsymbol{A}^{\mathsf{T}} \boldsymbol{C}_{\boldsymbol{u}}^{-1} (\boldsymbol{C}_{\boldsymbol{u}}^{-1} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n})^{-1} \boldsymbol{C}_{\boldsymbol{u}}^{-1} \boldsymbol{A} \\ &= \boldsymbol{A}^{\mathsf{T}} \left(\boldsymbol{I} + \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n} \boldsymbol{C}_{\boldsymbol{u}} \right)^{-1} \tilde{\boldsymbol{\Gamma}}_{\mathrm{b},n} \boldsymbol{A}. \end{aligned} \tag{3.155}$$

Note that neither (3.154) nor (3.155) contains the inverse of $\boldsymbol{A}$ or $\boldsymbol{C}_{\boldsymbol{u}}$, so they are well defined also for singular state transition or process noise covariance matrices.

Initialization We assume blockwise processing, where no further observations after the time instant $N-1$ are available. The initial message is therefore an improper uniform distribution, $m_{p(\boldsymbol{s}_N \,|\, \boldsymbol{s}_{N-1}) \to \boldsymbol{s}_{N-1}}(\boldsymbol{s}_{N-1}) \propto 1$, with parameters $\boldsymbol{\nu}_{\mathrm{b},N-1} = \boldsymbol{0}$ and $\boldsymbol{\Gamma}_{\mathrm{b},N-1} = \boldsymbol{0}$.

Computation of Extrinsic Phase Information

After the forward and backward recursions are completed, the final step is to compute the extrinsic phase information $\vec{\zeta}_n$ by combining the forward information $\boldsymbol{\mu}_{\mathrm{f},n}, \boldsymbol{\Sigma}_{\mathrm{f},n}$ and the backward information $\boldsymbol{\nu}_{\mathrm{b},n}, \boldsymbol{\Gamma}_{\mathrm{b},n}$. First, we multiply the two Gaussians to obtain the message $m_{\boldsymbol{s}_n \to p(\vartheta_n \,|\, \boldsymbol{s}_n)}(\boldsymbol{s}_n) \propto \mathcal{N}(\vec{\boldsymbol{\mu}}_n, \vec{\boldsymbol{\Sigma}}_n)$, with the parameters

$$\begin{aligned} \vec{\boldsymbol{\Sigma}}_n &= \left(\boldsymbol{\Sigma}_{\mathrm{f},n}^{-1} + \boldsymbol{\Gamma}_{\mathrm{b},n} \right)^{-1} \\ &= \left(\boldsymbol{I} + \boldsymbol{\Sigma}_{\mathrm{f},n} \boldsymbol{\Gamma}_{\mathrm{b},n} \right)^{-1} \boldsymbol{\Sigma}_{\mathrm{f},n} \end{aligned} \tag{3.156}$$

and

$$\begin{aligned} \vec{\boldsymbol{\mu}}_n &= \vec{\boldsymbol{\Sigma}}_n \left(\boldsymbol{\Sigma}_{\mathrm{f},n}^{-1} \boldsymbol{\mu}_{\mathrm{f},n} + \boldsymbol{\nu}_{\mathrm{b},n} \right) \\ &= \left(\boldsymbol{I} + \boldsymbol{\Sigma}_{\mathrm{f},n} \boldsymbol{\Gamma}_{\mathrm{b},n} \right)^{-1} \boldsymbol{\Sigma}_{\mathrm{f},n} \left(\boldsymbol{\Sigma}_{\mathrm{f},n}^{-1} \boldsymbol{\mu}_{\mathrm{f},n} + \boldsymbol{\nu}_{\mathrm{b},n} \right) \\ &= \left(\boldsymbol{I} + \boldsymbol{\Sigma}_{\mathrm{f},n} \boldsymbol{\Gamma}_{\mathrm{b},n} \right)^{-1} \left(\boldsymbol{\mu}_{\mathrm{f},n} + \boldsymbol{\Sigma}_{\mathrm{f},n} \boldsymbol{\nu}_{\mathrm{b},n} \right). \end{aligned} \tag{3.157}$$

Then, with $\vartheta_n = \boldsymbol{c}^{\mathsf{T}} \boldsymbol{s}_n$, the first two moments of ϑ_n are given as $\mathbb{E}[\vartheta_n] = \boldsymbol{c}^{\mathsf{T}} \vec{\boldsymbol{\mu}}_n$ and $\mathrm{var}[\vartheta_n] = \boldsymbol{c}^{\mathsf{T}} \vec{\boldsymbol{\Sigma}}_n \boldsymbol{c}$. With the usual conversion of Gaussian to von Mises distributions, the extrinsic phase information is thus found as

$$\vec{\zeta}_n = \frac{\mathrm{e}^{\mathrm{j} \boldsymbol{c}^{\mathsf{T}} \vec{\boldsymbol{\mu}}_n}}{\boldsymbol{c}^{\mathsf{T}} \vec{\boldsymbol{\Sigma}}_n \boldsymbol{c}}. \tag{3.158}$$

3.6.3 Numerical Results

We now assess the performance of the derived PLL phase noise estimator. As a benchmark, we compare it to the simple VCO phase noise estimator from the previous section, which assumes a random walk and requires one parameter, the variance of the phase difference $\Delta\vartheta_n = \vartheta_n - \vartheta_{n-1}$. When computing $\mathrm{var}\,[\Delta\vartheta]$ for the PLL phase noise model, we can safely ignore the phase noise of the reference crystal and write

$$\begin{aligned}\Delta\vartheta_n &\approx \vartheta_{\mathrm{vco},n} - \vartheta_{\mathrm{vco},n-1} \\ &\overset{(a)}{=} (z_\infty - 1)\vartheta_{\mathrm{vco},n-1} + \Delta\vartheta_{\mathrm{vco},n}\end{aligned} \tag{3.159}$$

where (a) uses (3.127). Note that $\Delta\vartheta_{\mathrm{vco},n}$ above is the driving noise of the AR process, which is not the same as $\Delta\vartheta_n$ due to the factor z_∞. Thus,

$$\begin{aligned}\mathrm{var}\,[\Delta\vartheta] &= (1 - z_\infty)^2\ \mathrm{var}\,[\vartheta_{\mathrm{vco}}] + \mathrm{var}\,[\Delta\vartheta_{\mathrm{vco}}] \\ &= (1 - z_\infty)^2\ \frac{(2\pi f_{\mathrm{c}})^2 c_{\mathrm{vco}} T}{1 - z_\infty^2} + (2\pi f_{\mathrm{c}})^2 c_{\mathrm{vco}} T \\ &= \frac{(2\pi f_{\mathrm{c}})^2 c_{\mathrm{vco}} T}{1 - \pi f_{\mathrm{pll}} T}.\end{aligned} \tag{3.160}$$

Figure 3.25 shows the results of the comparison, where the same LDPC code as in the previous section has been used, but with a pilot spacing of 30 symbols in order to make the difference between the estimators more pronounced. The phase noise process has been generated with the same parameters as in Figure 3.22.

In the initial iteration, where the estimators work purely pilot-aided, the PLL phase noise estimator outperforms the VCO estimator by around 0.2–0.3 dB. Over the iterations, however, the gap shrinks and is less than 0.1 dB after the fourth iteration. While the specific results of course depend on the phase noise parameters, this experiment at least indicates that the simple Wiener phase noise estimator could be the more interesting choice in practice, even if the true phase noise process is not accurately modeled as a random walk. After all, it has a much lower complexity than the multidimensional Kalman smoother, and it is also more robust since it needs fewer parameters, which have been perfectly known in this simulation, but may need to be estimated in practice.

3.6.4 Concluding Remarks

This section concludes our discussion of phase noise estimation. We have systematically derived a fully-soft phase noise estimator, which takes probabilistic information about the data symbols as input, and provides soft phase information as output. We have proposed to split the estimator conceptually into two blocks, front-end and back-end.

The front-end is agnostic of the phase noise statistics. It converts the decoder feedback into intrinsic phase information, represented by one complex number per time

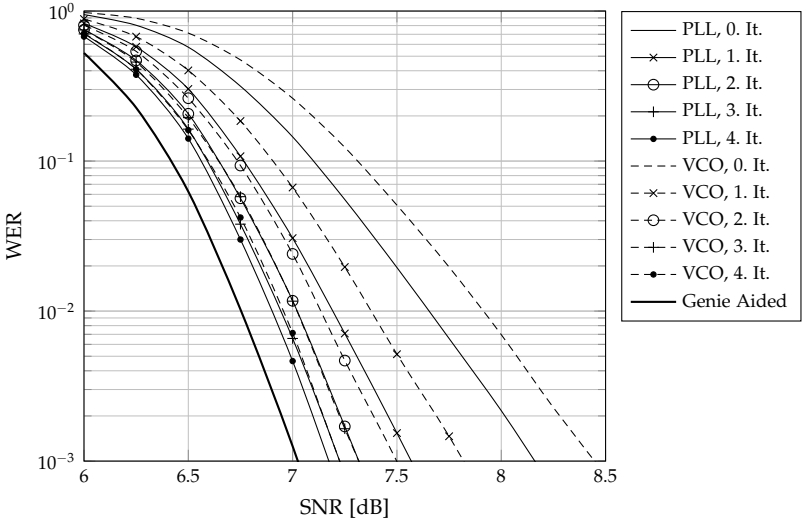


Figure 3.25: PLL phase noise estimation: comparison of the PLL estimator from this section with the estimator from the previous section that assumes a free-running VCO

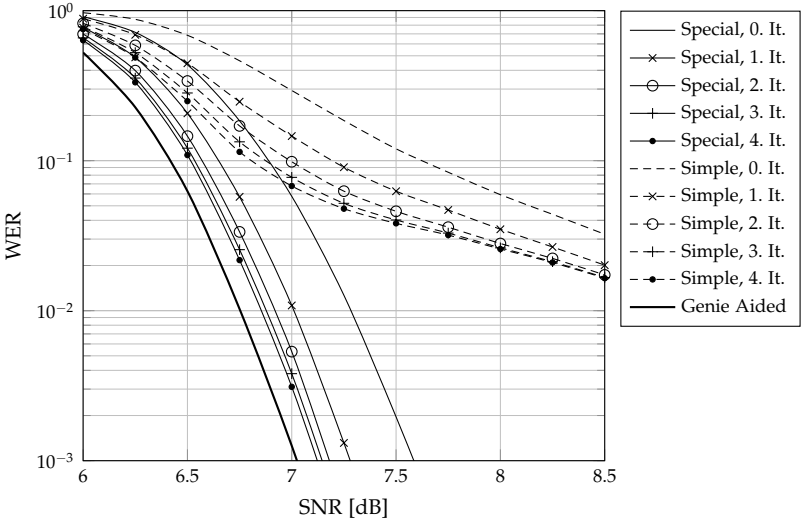
instant which encodes both a hard phase estimate and its reliability (inverse variance). We have shown both theoretically and via simulations that it is necessary to feed back the full posterior information about the data symbols, not just extrinsic information. In earlier, purely pilot-aided estimators, this issue did not arise, but it is all too often neglected even in more recent papers on code-aided phase estimation.

The back-end takes intrinsic phase information from the front-end and computes extrinsic phase information. We have derived three implementations for increasingly general models: a static phase offset, a Wiener phase noise process, and a linear Gaussian state space model. Further implementations are of course possible. The LMMSE filter, for example, which has been proposed in [124] for the pilot-aided case and extended in [49] to code-aided estimation, also fits as a possible back-end into the proposed framework.

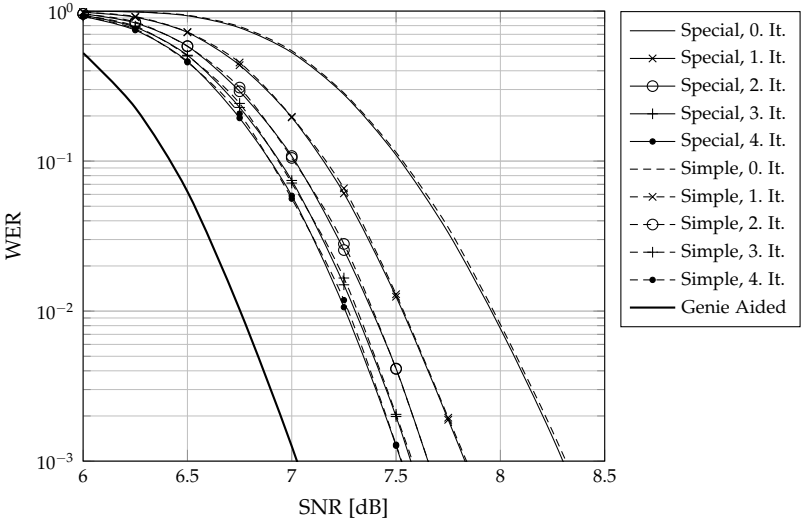
Finally, note that the Gaussian state space model is by no means restricted to PLL phase noise. As another example, consider the joint estimation of phase noise and a residual frequency offset ν, as studied in [9]. Assuming Wiener phase noise[17], and defining $\phi \triangleq 2\pi\nu T$, the phase process can be modeled as

$$\vartheta_n = \vartheta_{n-1} + \phi + \Delta\vartheta_n \tag{3.161}$$

[17]This is without loss of generality; the extension to PLL phase noise is straightforward.



(a) Frequency offset ϕ significant compared to phase noise ($\sigma_{\Delta\vartheta} = 2°, \sigma_\phi = 2°$)



(b) Frequency offset ϕ negligible compared to phase noise ($\sigma_{\Delta\vartheta} = 5°, \sigma_\phi = 1°$)

Figure 3.26: Estimation of a Wiener phase noise process including a constant frequency offset: comparison of specialized estimator based on (3.162) vs the simple random walk estimator from Section 3.5

which can be cast into a state space model by setting

$$\boldsymbol{s}_n = \begin{pmatrix} \vartheta_n \\ \phi \end{pmatrix} \quad \boldsymbol{A} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad \boldsymbol{c} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \boldsymbol{C_u} = \begin{pmatrix} \sigma_{\Delta\vartheta}^2 & 0 \\ 0 & 0 \end{pmatrix}. \tag{3.162}$$

It is interesting to note that [9] proposes a kind of hybrid estimator for this problem. The phase process ϑ_n, on the one hand, is estimated by means of tracking von Mises distributions. This comes as no surprise since [9] is a follow-up paper of [22], where the von-Mises-based phase noise estimator was first proposed, and which emphasizes the significant complexity reduction that it achieves compared to the (at that time) state-of-the-art approach of quantizing ϑ_n and treating it as a discrete variable. But the normalized frequency offset ϕ, on the other hand, *is* handled by discretizing it into L hypotheses despite that insight. Essentially, [9] invokes the Wiener phase noise estimator L times with different trial values $\hat{\phi}_l$. Our Gaussian state space estimator is a consequential further development which completely avoids the artificial discretization of continuous random variables.

Figure 3.26 compares the Gaussian state space estimator based on (3.162) with the simple Wiener phase noise estimator which does not assume an additional frequency offset. For each codeword, one realization $\phi \sim \mathcal{N}(0, \sigma_\phi^2)$ is drawn. For a fair comparison, the simple Wiener phase noise estimator is not parameterized with $\sigma_{\Delta\vartheta}^2$, but rather with $\sigma_\phi^2 + \sigma_{\Delta\vartheta}^2$ since this is the variance of the phase increment $\vartheta_n - \vartheta_{n-1}$.

In Figure 3.26a, the parameters are set to $\sigma_{\Delta\vartheta} = \sigma_\phi = 2^\circ$. The effect of the frequency offset is therefore quite significant compared to the Wiener phase noise, which is reflected by the huge performance difference of the two estimators. The simple random-walk-estimator is obviously not able to track the phase process. In contrast, Figure 3.26b shows results for $\sigma_{\Delta\vartheta} = 5^\circ$ and $\sigma_\phi = 1^\circ$, so that the frequency offset is now almost negligible compared to the Wiener phase noise. Here, the advantage of the specialized estimator is so tiny that, up the accuracy of the Monte Carlo simulation, both estimators are virtually equivalent.

The bottom line is that the Kalman smoother can indeed perform better than the von-Mises-based estimator, since it is able to model the phase noise statistics more accurately. Having said this, one should evaluate carefully whether the additional complexity is really well spent. Due to its simplicity and robustness, the von Mises estimator seems to be an interesting choice for many scenarios, even in cases where the random walk model is just a rough approximation of the true phase noise statistics.

3.7 SNR Estimation

In the previous sections, the power spectral density N_0 of the AWGN process has been assumed to be known at the receiver. Also, we have only briefly touched upon the estimation of the channel gain g_0 in Section 3.3.2, in the context of traditional (ML-like) hard estimation. In this section, we conclude our discussion of code-aided parameter estimation with a derivation of soft estimators for both g_0 and N_0. For conciseness, we refer to the joint estimation of g_0 and N_0 as *SNR estimation*, but it should be emphasized that we actually compute separate estimates for the two parameters. If an estimate of

$$\mathsf{SNR} = \frac{g_0^2 E_\mathrm{s}}{N_0} \tag{3.163}$$

itself is required (for example as a quality indicator used for link adaptation), it can easily be obtained from $\hat{g}_0$ and $\hat{N}_0$.

For simplicity, we will restrict the discussion to *coherent* SNR estimation, that is, we assume that the phase has already been perfectly recovered. Thanks to the modularity of graph-based receivers, however, it is fairly straightforward to combine the previously introduced phase estimators with the coherent SNR estimators in an iterative manner. An initial SNR estimate, required to bootstrap the algorithm, could then be obtained with an envelope-based estimator, which discards the phase of the observations. Examples are the non-coherent EM algorithm from [45] and the M_2M_4-estimator [85, 103] which uses the second and fourth moment of the observations.

This section is structured as follows. After an introduction of the system model in Section 3.7.1, we apply our main algorithmic families as listed in Section B.4.4 to the problem of SNR estimation, from "simple and moderately well performing" up to "complex, but virtually optimal." We start in Section 3.7.2 with classical hard estimation, in particular with the ML estimator for pilot-aided estimation, and its iterative code-aided generalization based on expectation maximization. Those estimators can be regarded as state-of-the-art, and will be used both as a starting point for the derivation of soft estimators, and as a benchmark for performance comparisons.

Recall that EM, if derived as a special case of DM, uses *exclusive* divergence minimization, followed by a projection into the family of Dirac distributions. As a natural generalization to soft output, Section 3.7.3 derives a VMP-type estimator by switching to a Bayesian setting and by dropping the Dirac-constraint. We will see that the resulting algorithm looks formally quite similar to the EM-type hard estimator (and in particular has almost the same computational complexity), but performs remarkably better if the number of pilot symbols is small, i. e., if the initial estimate is unreliable.[18]

[18]Note that this procedure is very similar to the development in Section 3.4, where we have derived the soft VMP-type phase estimator (3.74), which is a generalization of the conventional EM-type hard estimator (3.75).

Both the EM and the VMP estimators require some training data[19] to obtain an initial guess, which is then iteratively refined. However, it is well possible to invent SNR estimators which are purely non-data-aided, i. e., do not require any pilots at all. Now, it would be quite disappointing if those more or less *ad hoc* algorithms had desirable properties which were out of reach for our systematically derived DM-based estimators. Fortunately, it turns out that *inclusive* divergence minimization also leads to a pilot-less SNR estimator. In Section 3.7.4, we derive an EP-type algorithm (with some approximations) which gives virtually perfect results without training data and without iterations over demapper and decoder. As usual, its drawback with respect to the VMP-type estimator from Section 3.7.3 is its higher computational complexity.

The work presented in this section has partly been published in [120].

3.7.1 System Model

As stated above, we will only consider coherent SNR estimation in the following, so we will work with the channel model

$$y_n = g_0 x_n + w_n, \quad 0 \leq n < N \tag{3.164}$$

where

$$w_n \sim \mathcal{CN}(0, N_0). \tag{3.165}$$

Both the real-valued channel gain $g_0 \geq 0$ and the variance N_0 of the discrete-time AWGN process are unknown to the receiver. Rather than estimating the noise variance N_0 itself, however, we will derive estimators for the *noise precision* instead, defined as

$$\gamma_0 \triangleq \frac{1}{N_0}. \tag{3.166}$$

The reason is that the conjugate prior for the noise precision is a Gamma distribution, which leads to simpler expressions than the *inverse* Gamma distribution, which would arise if we were estimating N_0 directly.[20]

Combining the channel model (3.164) with the standard BICM transmitter as used in the previous sections, the joint posterior of the unknown quantities factorizes as

$$\begin{aligned} &p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x}_\mathcal{D}, g_0, \gamma_0 \,|\, \boldsymbol{y}) \\ &\quad \propto p(\boldsymbol{b})\, p(\boldsymbol{c} \,|\, \boldsymbol{b})\, p(\boldsymbol{x}_\mathcal{D} \,|\, \boldsymbol{c})\, p(g_0)\, p(\gamma_0)\, p(\boldsymbol{y}_\mathcal{D} \,|\, \boldsymbol{x}_\mathcal{D}, g_0, \gamma_0)\, p(\boldsymbol{y}_\mathcal{P} \,|\, \boldsymbol{x}_\mathcal{P}, g_0, \gamma_0). \end{aligned} \tag{3.167}$$

A graphical model of (3.167) is shown in Figure 3.27, which serves as the basis for the following derivations. Note that some factors could be broken down further into the individual time instants, for instance $p(\boldsymbol{y}_\mathcal{D} \,|\, \boldsymbol{x}_\mathcal{D}, g_0, \gamma_0) = \prod_{n \in \mathcal{D}} p(y_n \,|\, x_n, g_0, \gamma_0)$, but the

[19] Unless we are willing to initialize the algorithm with some rather arbitrary values as in [142].

[20] And furthermore, the demapper actually requires the inverse of N_0, so by estimating N_0^{-1} we turn the division by N_0 into a multiplication with γ_0, which is much cheaper in a VLSI implementation.

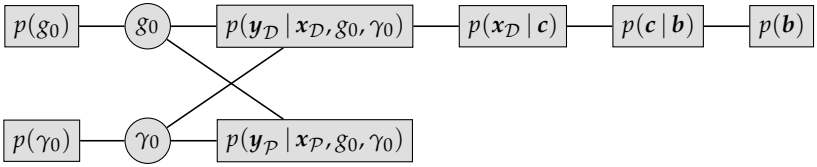


Figure 3.27: Factor graph of the joint posterior distribution (3.167)

compact representation in Figure 3.27 is sufficient for the estimators based on exclusive DM, which are the topic of the following Sections 3.7.2 and 3.7.3. Only in Section 3.7.4 on inclusive DM will we need the more detailed factorization, c. f. Figure 3.30 on page 109.

3.7.2 Hard SNR Estimation

We start with a discussion of classical hard estimators, namely the ML and EM estimators for data- and code-aided estimation, respectively. Those approaches are not new; there are several papers which apply the EM algorithm to the SNR estimation problem (see for instance [26, 142, 152]). However, their derivation is nonetheless interesting for us, since they serve as a basis for the subsequent generalization to Bayesian soft estimators, and as a performance benchmark.

The following derivations are rooted in orthodox statistics, meaning that (a) we compute hard estimates, and (b) that g_0 and γ_0 are modeled as deterministic quantities. Thus, for the purpose of this section, $p(g_0)$ and $p(\gamma_0)$ would need to be removed from Figure 3.27; or, formally equivalently, they could be understood as improper (non-normalizable) uniform priors: $p(g_0) \propto 1$ and $p(\gamma_0) \propto 1$.

Data-Aided Estimation

If $N_\mathrm{p} > 0$, the ML estimator based on the training sequence is easily derived. The log-likelihood function is

$$\log p(\boldsymbol{y}_\mathcal{P} \mid \boldsymbol{x}_\mathcal{P}, g_0, \gamma_0) = N_\mathrm{p} \log\left(\frac{\gamma_0}{\pi}\right) - \gamma_0 \left\|\boldsymbol{y}_\mathcal{P} - g_0 \boldsymbol{x}_\mathcal{P}\right\|^2 \tag{3.168}$$

and by setting its derivatives with respect to g_0 and γ_0 to zero, the ML estimates are found as

$$\hat{g}_0 = \frac{\mathrm{Re}(\boldsymbol{x}_\mathcal{P}^\mathsf{H} \boldsymbol{y}_\mathcal{P})}{\left\|\boldsymbol{x}_\mathcal{P}\right\|^2} \tag{3.169}$$

$$\hat{\gamma}_0 = \frac{N_\mathrm{p}}{\left\|\boldsymbol{y}_\mathcal{P} - \hat{g}_0 \boldsymbol{x}_\mathcal{P}\right\|^2}. \tag{3.170}$$

Since $\hat{g}_0$ does not depend on $\hat{\gamma}_0$, the ML estimates can be computed directly, without iterations between $\hat{g}_0$ and $\hat{\gamma}_0$.

Note that it is possible for $\hat{g}_0$ to become negative, even though we know that $g_0 \geq 0$. The difficulty of exploiting such prior knowledge is a well-known problem of orthodox statistics, and we need to resort to some *ad hoc* procedure in order to ensure that $\hat{g}_0 \geq 0$. For example, we could modify the estimate of g_0 as proposed in [64, Section 10.3]:

$$\hat{g}_0' \triangleq \max\{0, \hat{g}_0\}. \tag{3.171}$$

This could be viewed as a MAP estimate with an improper uniform prior $\mathcal{U}[0,\infty)$ over the non-negative reals. But of course, this does not solve our problem at all: an estimate $\hat{g}_0' = 0$ might be physically more meaningful than $\hat{g}_0 < 0$, but it is equally useless. Indeed, the ability to exploit the prior knowledge $g_0 \geq 0$ properly is one of the major advantages of the Bayesian approach that we will pursue in Section 3.7.3.

Code-Aided Estimation

In order to also use the information that is contained in the data symbols, we will now derive an iterative EM-like[21] estimator from the divergence minimization framework, by constraining the beliefs of g_0 and γ_0 to Dirac distributions: $q(g_0) = \delta(g_0 - \hat{g}_0)$ and $q(\gamma_0) = \delta(\gamma_0 - \hat{\gamma}_0)$. The general receiver structure, including the BP-based pass through the demapper and decoder, as well as the computation of the feedback $q(\boldsymbol{x}_\mathcal{D})$, has been described in Section 3.3, so in the following we only derive the update equations for the parameter estimates $\hat{g}_0$ and $\hat{\gamma}_0$. In the interest of readability, we will not annotate the beliefs $q(g_0)$, $q(\gamma_0)$, and $q(\boldsymbol{x}_\mathcal{D})$ with an iteration index; it should be clear that all three are updated once per EM iteration.

Update of $q(g_0)$ Specializing the generic hard estimator (B.84) to $\hat{g}_0$ yields

$$\begin{aligned}
\hat{g}_0 &= \arg\max_{g_0} \mathbb{E}_{q(\gamma_0)\,q(\boldsymbol{x}_\mathcal{D})}\left[\log p(\boldsymbol{y} \mid \boldsymbol{x}, g_0, \gamma_0)\right] \\
&= \arg\max_{g_0} \mathbb{E}_{q(\gamma_0)\,q(\boldsymbol{x}_\mathcal{D})}\left[N \log\left(\frac{\gamma_0}{\pi}\right) - \gamma_0 \left\|\boldsymbol{y} - g_0\boldsymbol{x}\right\|^2\right] \\
&= \arg\max_{g_0} \mathbb{E}_{q(\boldsymbol{x}_\mathcal{D})}\left[-\hat{\gamma}_0\left(\|\boldsymbol{y}\|^2 - 2g_0\,\mathrm{Re}(\boldsymbol{x}^\mathsf{H}\boldsymbol{y}) + g_0^2\,\|\boldsymbol{x}\|^2\right)\right] \\
&= \arg\min_{g_0}\left(\|\boldsymbol{y}\|^2 - 2g_0\,\mathrm{Re}(\bar{\boldsymbol{x}}^\mathsf{H}\boldsymbol{y}) + g_0^2\,\mathbb{E}\left[\|\boldsymbol{x}\|^2\right]\right)
\end{aligned} \tag{3.172}$$

where $\bar{\boldsymbol{x}}$ contains the pilot symbols and the expectations of the data symbols (soft symbols) as defined below (3.46), and $\mathbb{E}\left[\|\boldsymbol{x}\|^2\right] = \sum_{n=0}^{N-1}\rho_n$ is an estimate of the total transmitted energy.

[21] As explained at the beginning of the chapter, the resulting algorithm is a true instance of EM only in those special cases where we are able to evaluate $p(\boldsymbol{x}_\mathcal{D} \mid \boldsymbol{y}_\mathcal{D}, \hat{g}_0, \hat{\gamma}_0)$ exactly.

Setting the derivative of (3.172) to zero, we finally obtain

$$\hat{g}_0 = \frac{\mathrm{Re}(\bar{\boldsymbol{x}}^{\mathsf{H}}\boldsymbol{y})}{\mathbb{E}\left[\|\boldsymbol{x}\|^2\right]}. \tag{3.173}$$

Update of $q(\gamma_0)$ The update equation for $\hat{\gamma}_0$ is found similarly as

$$\begin{aligned}\hat{\gamma}_0 &= \arg\max_{\gamma_0} \mathbb{E}_{q(g_0)q(\boldsymbol{x}_\mathcal{D})}\left[\log p(\boldsymbol{y} \mid \boldsymbol{x}, g_0, \gamma_0)\right] \\ &= \arg\max_{\gamma_0} \mathbb{E}_{q(\boldsymbol{x}_\mathcal{D})}\left[N \log\left(\frac{\gamma_0}{\pi}\right) - \gamma_0 \|\boldsymbol{y} - \hat{g}_0\boldsymbol{x}\|^2\right] \\ &= \arg\max_{\gamma_0} \left(N \log\left(\frac{\gamma_0}{\pi}\right) - \gamma_0 \left(\|\boldsymbol{y}\|^2 - 2\hat{g}_0 \,\mathrm{Re}(\bar{\boldsymbol{x}}^{\mathsf{H}}\boldsymbol{y}) + \hat{g}_0^2\, \mathbb{E}\left[\|\boldsymbol{x}\|^2\right]\right)\right)\end{aligned} \tag{3.174}$$

and finally

$$\hat{\gamma}_0 = \frac{N}{\|\boldsymbol{y}\|^2 - 2\hat{g}_0 \,\mathrm{Re}(\bar{\boldsymbol{x}}^{\mathsf{H}}\boldsymbol{y}) + \hat{g}_0^2 \mathbb{E}\left[\|\boldsymbol{x}\|^2\right]}. \tag{3.175}$$

Note the similarity between the code-aided update equations (3.173), (3.175) and their data-aided couterparts (3.169) and (3.170). Basically the only difference consists in the expectation over the data symbols (the denominator in (3.175) is just the expectation of $\|\boldsymbol{y} - \hat{g}_0\boldsymbol{x}\|^2$).

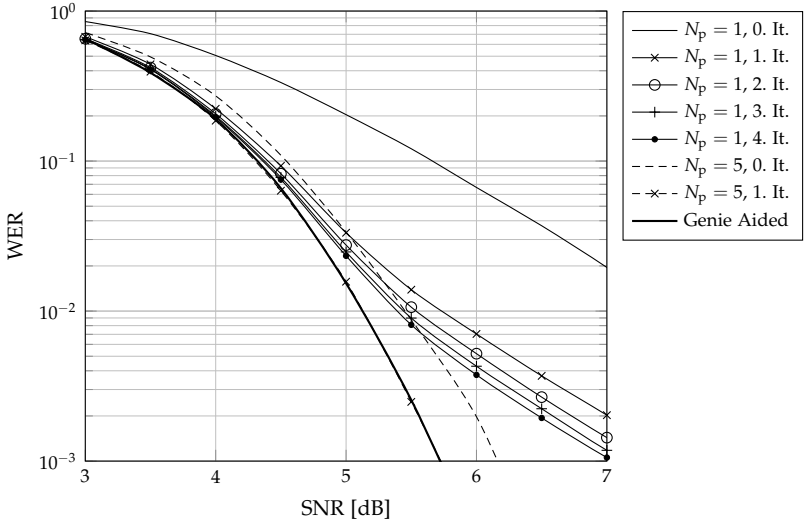
Numerical Results

We now assess the performance of the EM-based hard SNR estimator. Here and in the rest of this section, we use the rate-1/3 turbo code from 3GPP LTE with a length of 128 information bits, which are mapped to 396 code bits.[22] In order to make the receiver sensitive to SNR estimation errors, we use a 16-QAM modulation scheme,[23] such that we obtain a block of $N_\mathrm{d} = 99$ data symbols, into which N_p pilot symbols are multiplexed. The initial iteration uses the ML estimates (3.169) and (3.170), which rely only on the pilot symbols. Later iterations then use the EM estimates (3.173) and (3.175).

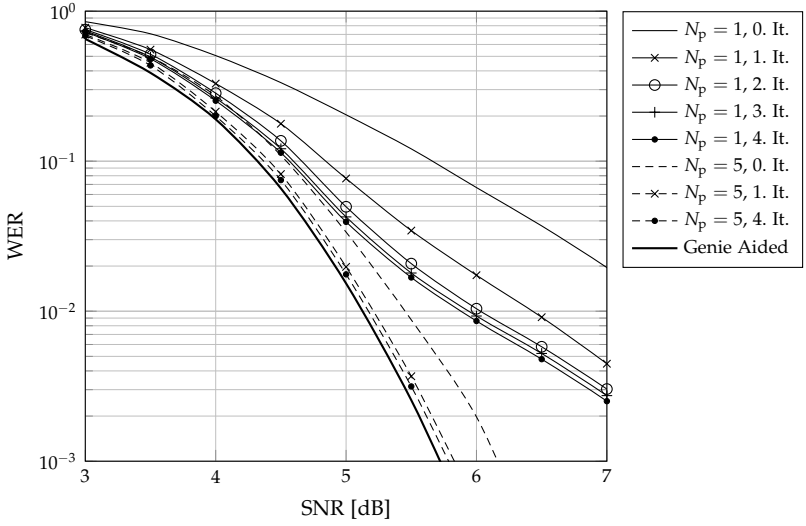
Figure 3.28a shows the word error rate for $N_\mathrm{p} = 1$ and $N_\mathrm{p} = 5$ pilot symbols. For the longer training sequence, one single iteration over the decoder suffices to yield the same results (up to the simulation accuracy) as the genie aided receiver which has perfect knowledge of g_0 and γ_0. The short training sequence of length 1, however, is insufficient; the plots show a kind of error floor. The main reason for this behaviour is the above

[22] The actual code rate is slightly lower than 1/3 due to the tail bits which drive the RSC encoders into the zero state, and which are also transmitted.

[23] In [150] it has been shown that turbo decoders are quite insensitive to SNR estimation errors. However, [150] only considers BPSK. For higher-order QAM modulations, a reliable estimate of the channel gain is indeed required.



(a) Code-Aided: both demapper and decoder are invoked in each iteration



(b) Non-Code-Aided: only the demapper is invoked, but not the decoder

Figure 3.28: Performance of the EM-based SNR estimator, with 1 and 5 pilot symbols

discussed problem that the initial estimate $\hat{g}_0$ can become negative, in which case the whole frame is lost. For $N_\mathrm{p} = 1$, this happens with a non-negligible probability.

For complexity reasons, we may choose to iterate only over the demapper, yielding an NCA parameter estimator. In this case, the decoder is only invoked once using the final SNR estimate. Results obtained with the NCA estimator are shown in Figure 3.28b. With the long training sequence, a single iteration over the demapper still suffices to approach the genie aided receiver up to around 0.1 dB. And as expected, the error floor in the case $N_\mathrm{p} = 1$ is even more pronounced than before.

The bottom line is that the EM-based estimator works well as long as the initial estimate is sufficiently reliable (which is a well-known characteristic of EM algorithms). For short training sequences, however, the EM algorithm fails to work reliably. In the remainder of this chapter, we derive two soft SNR estimators, based on exclusive and inclusive DM, respectively, which are able to fix this problem.

3.7.3 Soft SNR Estimation Based on Exclusive DM

We now switch to a Bayesian model, i. e., we regard g_0 and γ_0 as random variables and assign prior distributions to them. Furthermore, we remove the Dirac-constraint from the beliefs $q(g_0)$ and $q(\gamma_0)$ in order to produce soft estimates.

Prior Distributions

Since we are now working in a fully Bayesian setting, we need to assign prior distributions to g_0 and γ_0. For the same reasons as discussed in Section 3.4.2 on phase estimation, we will choose *conjugate priors*, i. e., priors which lead to posteriors from the same parametric family. Statistical inference then reduces to updating the parameters (as opposed to tracking arbitrarily evolving distributions, which is generally infeasible).

Prior $p(g_0)$ Consider the likelihood function

$$p(\boldsymbol{y} \,|\, \boldsymbol{x}, g_0, \gamma_0) = \left(\frac{\gamma_0}{\pi}\right)^N \exp\left(-\gamma_0 \,\|\boldsymbol{y} - g_0\boldsymbol{x}\|^2\right). \tag{3.176}$$

Taken as a function of g_0, it has the parametric form

$$\ell(g_0) \propto \exp\left(-a\, g_0^2 + b\, g_0\right) \tag{3.177}$$

for some $a > 0$ and $b \in \mathbb{R}$, which is the form of a Gaussian distribution. But, as mentioned above, we also need to encode the physical constraint $g_0 \geq 0$ in the prior. We therefore assign zero probability to $g_0 < 0$ and set the prior of g_0 to a *truncated Gaussian*

$$p(g_0) = \mathcal{TN}(g_0 \,|\, m_\mathrm{a}, s_\mathrm{a}^2). \tag{3.178}$$

Briefly, the truncated Gaussian distribution is proportional to a Gaussian, but with the tail $g_0 < 0$ cut off. Note that the parameters m and s^2 are not equal to the mean and variance of the distribution; instead, they refer to the mean and variance of the underlying non-truncated Gaussian. A detailed description is given on page 216.

It can easily be verified that the truncated Gaussian is indeed a conjugate prior, since the normalized product of a truncated Gaussian and a (non-truncated) Gaussian is again a truncated Gaussian.

If there is no prior knowledge available beyond the non-negativity constraint, we can set $m_\mathrm{a} = 0$ and take the limit $s_\mathrm{a} \to \infty$, yielding an improper uniform prior $\mathcal{U}[0, \infty)$.

Prior $p(\gamma_0)$ If we view (3.176) as a function of γ_0, we get the parametric form

$$\ell(\gamma_0) \propto \gamma_0^a \exp(-\gamma_0 b) \tag{3.179}$$

for some $a > 0$ and $b > 0$, which is the form of a Gamma distribution. Since the class of Gamma distributions is an exponential family (see page 218 in the appendix for details), it is closed under multiplication, so

$$p(\gamma_0) = \mathcal{G}(\gamma_0 \,|\, \alpha_\mathrm{a}, \beta_\mathrm{a}) \tag{3.180}$$

is a conjugate prior.

In this case, there are two obvious ways to choose a non-informative prior. On the one hand, taking the limits $\alpha_\mathrm{a} \to 1$, $\beta_\mathrm{a} \to 0$ again results in an improper $\mathcal{U}[0, \infty)$ prior. On the other hand, the limits $\alpha_\mathrm{a} \to 0$, $\beta_\mathrm{a} \to 0$ yield the improper *Jeffrey's prior*

$$p(\gamma_0) \propto \mathbb{1}[\gamma_0 > 0] \, \frac{1}{\gamma_0} \tag{3.181}$$

which is often said to be a more natural prior for scale parameters; see [82, Chapter 24] or [62, Chapter 12]. We will return to this issue below.

Update Equations

Now that we have fixed the priors to

$$p(g_0) = \mathcal{TN}(g_0 \,|\, m_\mathrm{a}, s_\mathrm{a}^2) \quad \text{and} \quad p(\gamma_0) = \mathcal{G}(\gamma_0 \,|\, \alpha_\mathrm{a}, \beta_\mathrm{a}) \tag{3.182}$$

we can derive the update equations for the beliefs $q(g_0)$ and $q(\gamma_0)$. By design they have the same parametric form as the priors, and we will write them as

$$q(g_0) = \mathcal{TN}(g_0 \,|\, m_\mathrm{p}, s_\mathrm{p}^2) \quad \text{and} \quad q(\gamma_0) = \mathcal{G}(\gamma_0 \,|\, \alpha_\mathrm{p}, \beta_\mathrm{p}). \tag{3.183}$$

Note that m_a, s_a, α_a and β_a stay constant, whereas m_p, s_p, α_p and β_p are iteratively updated during the inference algorithm. This could be emphasized with iteration indices, but we will omit them in the interest of readability.

Update of $q(g_0)$ Given the latest beliefs $q(\gamma_0)$ and $q(\boldsymbol{x}_\mathcal{D})$ about the noise precision and the data symbols, the update equation for $q(g_0)$ is given as

$$\begin{aligned}
q(g_0) &\propto p(g_0)\, \exp\left(\mathbb{E}_{q(\gamma_0)\,q(\boldsymbol{x}_\mathcal{D})}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, g_0, \gamma_0)]\right) \\
&\propto \mathbb{1}\left[g_0 \geq 0\right] \exp\left(-\frac{(g_0 - m_\mathrm{a})^2}{2s_\mathrm{a}^2} - \mathbb{E}_{q(\gamma_0)\,q(\boldsymbol{x}_\mathcal{D})}\left[\gamma_0 \left(g_0^2 \left\|\boldsymbol{x}\right\|^2 - 2g_0\, \mathrm{Re}(\boldsymbol{x}^\mathsf{H}\boldsymbol{y}) + \left\|\boldsymbol{y}\right\|^2\right)\right]\right) \\
&\propto \mathbb{1}\left[g_0 \geq 0\right] \exp\left(-\frac{g_0^2}{2}\left(\frac{1}{s_\mathrm{a}^2} + 2\bar{\gamma}_0\, \mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]\right) + g_0\left(\frac{m_\mathrm{a}}{s_\mathrm{a}^2} + 2\bar{\gamma}_0\, \mathrm{Re}(\bar{\boldsymbol{x}}^\mathsf{H}\boldsymbol{y})\right)\right) \\
&\propto \mathcal{TN}\left(g_0 \,\middle|\, \frac{m_\mathrm{a} + 2\bar{\gamma}_0 s_\mathrm{a}^2\, \mathrm{Re}(\bar{\boldsymbol{x}}^\mathsf{H}\boldsymbol{y})}{1 + 2\bar{\gamma}_0 s_\mathrm{a}^2\, \mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]}, \frac{s_\mathrm{a}^2}{1 + 2\bar{\gamma}_0 s_\mathrm{a}^2\, \mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]}\right)
\end{aligned} \tag{3.184}$$

where $\bar{\gamma}_0 \triangleq \frac{\alpha_\mathrm{p}}{\beta_\mathrm{p}}$ is the mean of $q(\gamma_0)$, as derived in (A.97). Therefore, it can be viewed as an MMSE-like estimate of γ_0.

If we take the limit $s_\mathrm{a} \to \infty$ to obtain a non-informative prior, (3.184) simplifies to

$$q(g_0) = \mathcal{TN}\left(g_0 \,\middle|\, \frac{\mathrm{Re}(\bar{\boldsymbol{x}}^\mathsf{H}\boldsymbol{y})}{\mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]}, \frac{1}{2\bar{\gamma}_0\, \mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]}\right). \tag{3.185}$$

From either (3.184) or (3.185), we can now compute the moments $\bar{g}_0 \triangleq \mathbb{E}\left[g_0\right]$ and $\overline{g_0^2} \triangleq \mathbb{E}\left[g_0^2\right]$ using (A.90) and (A.91) on page 217. They will be needed in the following update of $q(\gamma_0)$.

Update of $q(\gamma_0)$ The update equation for $q(\gamma_0)$ is found similarly as

$$\begin{aligned}
q(\gamma_0) &\propto p(\gamma_0)\, \exp\left(\mathbb{E}_{q(g_0)\,q(\boldsymbol{x}_\mathcal{D})}[\log p(\boldsymbol{y} \mid \boldsymbol{x}, g_0, \gamma_0)]\right) \\
&\propto \mathbb{1}\left[\gamma_0 \geq 0\right] \gamma_0^{\alpha_\mathrm{a}-1} \exp(-\gamma_0\beta_\mathrm{a})\, \gamma_0^N \exp\left(-\gamma_0\, \mathbb{E}_{q(g_0)\,q(\boldsymbol{x}_\mathcal{D})}\left[\left\|\boldsymbol{y} - g_0\boldsymbol{x}\right\|^2\right]\right) \\
&= \mathbb{1}\left[\gamma_0 \geq 0\right] \gamma_0^{\alpha_\mathrm{a}+N-1} \exp\left(-\gamma_0\left(\beta_\mathrm{a} + \left\|\boldsymbol{y}\right\|^2 - 2\bar{g}_0\, \mathrm{Re}(\bar{\boldsymbol{x}}^\mathsf{H}\boldsymbol{y}) + \overline{g_0^2}\, \mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]\right)\right) \\
&\propto \mathcal{G}\left(\gamma_0 \mid \alpha_\mathrm{a} + N, \beta_\mathrm{a} + \left\|\boldsymbol{y}\right\|^2 - 2\bar{g}_0\, \mathrm{Re}(\bar{\boldsymbol{x}}^\mathsf{H}\boldsymbol{y}) + \overline{g_0^2}\, \mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]\right).
\end{aligned} \tag{3.186}$$

The updated estimate of the noise precision is thus

$$\bar{\gamma}_0 = \frac{\alpha_\mathrm{a} + N}{\beta_\mathrm{a} + \left\|\boldsymbol{y}\right\|^2 - 2\bar{g}_0\, \mathrm{Re}(\bar{\boldsymbol{x}}^\mathsf{H}\boldsymbol{y}) + \overline{g_0^2}\, \mathbb{E}\left[\left\|\boldsymbol{x}\right\|^2\right]} \tag{3.187}$$

which, as we have seen above, is actually all that we need. We do not have to track the parameters α_p and β_p separately.

If we assume an improper uniform prior for γ_0, (3.187) simplifies to

$$\bar{\gamma}_0 = \frac{N+1}{\|\boldsymbol{y}\|^2 - 2\bar{g}_0 \operatorname{Re}(\bar{\boldsymbol{x}}^{\mathsf{H}}\boldsymbol{y}) + \overline{g_0^2}\,\mathbb{E}\left[\|\boldsymbol{x}\|^2\right]} \tag{3.188}$$

and with Jeffrey's prior we obtain

$$\bar{\gamma}_0 = \frac{N}{\|\boldsymbol{y}\|^2 - 2\bar{g}_0 \operatorname{Re}(\bar{\boldsymbol{x}}^{\mathsf{H}}\boldsymbol{y}) + \overline{g_0^2}\,\mathbb{E}\left[\|\boldsymbol{x}\|^2\right]}. \tag{3.189}$$

Iterations and Initialization Since (3.184) and (3.185) depend on $\bar{\gamma}_0$,[24] it looks like the receiver with soft SNR estimation is doubly-iterative, with inner iterations between the estimators for g_0 and γ_0 in addition to the usual outer iterations through the demapper and decoder. Fortunately, the dependence of $q(g_0)$ on $\bar{\gamma}_0$ is quite weak. I have verified by simulations that it suffices to update $q(g_0)$ and $\bar{\gamma}_0$ just once per (outer) iteration. Doing more than one inner iteration, i. e., re-computing $q(g_0)$ with the new $\bar{\gamma}_0$, does not result in any performance improvement.

In order to bootstrap the algorithm, a possible initialization is

$$\bar{\gamma}_0 = \frac{\alpha_{\mathrm{a}} + N_{\mathrm{p}}}{\beta_{\mathrm{a}} + \|\boldsymbol{y}_{\mathcal{P}} - \hat{g}_0 \boldsymbol{x}_{\mathcal{P}}\|^2} \tag{3.190}$$

as obtained by combining a pilot-only version of (3.187) with the ML estimate (3.169).

Message to the Demapper The messages from the SNR estimator to the demapper are computed as

$$\begin{aligned} m_{y_n \to x_n}(x_n) &\propto \exp\left(\mathbb{E}_{q(g_0)\,q(\gamma_0)}\left[\log p(y_n \mid x_n, g_0, \gamma_0)\right]\right) \\ &\propto \exp\left(-\bar{\gamma}_0\,\overline{g_0^2}\left|x_n - \frac{\bar{g}_0\, y_n}{\overline{g_0^2}}\right|^2\right) \\ &\propto \mathcal{CN}\left(x_n \,\middle|\, \frac{\bar{g}_0\, y_n}{\overline{g_0^2}}, \frac{1}{\bar{\gamma}_0\,\overline{g_0^2}}\right) \qquad (3.191) \\ &\overset{(a)}{\approx} \mathcal{CN}\left(x_n \,\middle|\, \frac{y_n}{\bar{g}_0}, \frac{1}{\bar{\gamma}_0\,\bar{g}_0^2}\right) \qquad (3.192) \end{aligned}$$

where (a) approximates $\overline{g_0^2} = \bar{g}_0^2 + \operatorname{var}[g_0]$ with just $\bar{g}_0^2$, i. e., it disregards $\operatorname{var}[g_0]$. Again, this does not cause any appreciable performance degradation, as verified by simulations.

[24] Note that this is in contrast to the hard SNR estimator, where $\hat{g}_0$ in (3.173) does not depend on $\hat{\gamma}_0$.

Computational Complexity

The EM estimator (3.173) and (3.175) is dominated by the computations of $\mathrm{Re}(\bar{\boldsymbol{x}}^{\mathsf{H}}\boldsymbol{y})$ and $\mathbb{E}\left[\|\boldsymbol{x}\|^2\right]$, which require for each $n \in \mathcal{D}$: (a) the evaluation of the belief $q(x_n)$, (b) the computation of $\mathrm{Re}(\mathbb{E}\left[x_n^*\right] y_n)$, and (c) the computation of $\mathbb{E}[|x_n|^2]$.

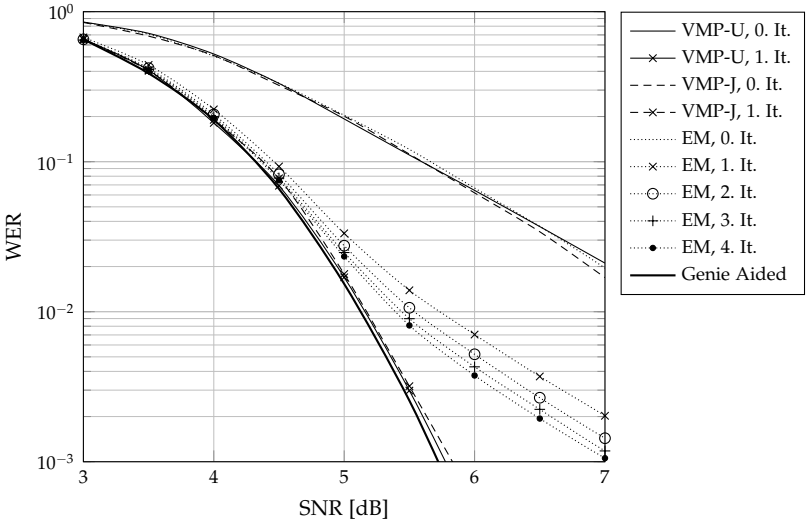
The update equations (3.184) and (3.187) for the soft estimator actually require the same expectations. Virtually the only difference to the EM estimator is the need to compute the first two moments of g_0 using (A.90) and (A.91), which involves a single evaluation of $\lambda(\frac{m_{\mathrm{p}}}{s_{\mathrm{p}}})$. We can therefore conclude that the additional complexity of the proposed soft estimator, compared to the EM estimator, is practically negligible.

Numerical Results

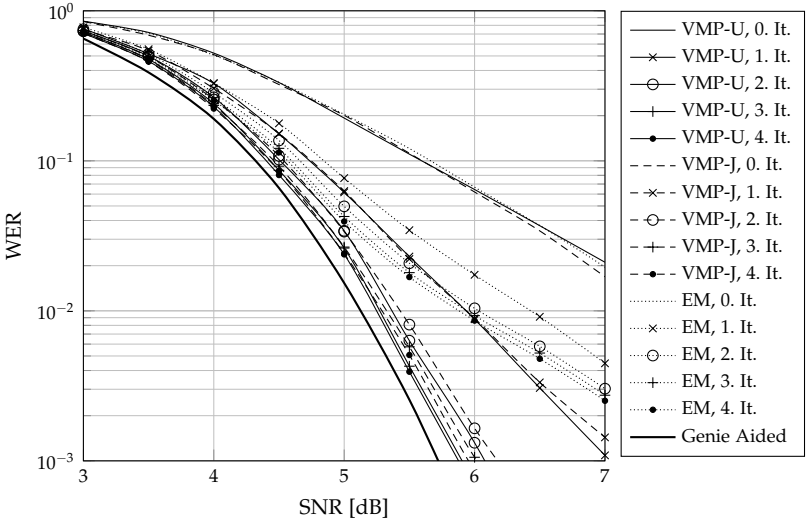
We now compare the proposed soft SNR estimator with the hard EM-type estimator from the previous section. We have seen in Figure 3.28 that the EM estimator performs very well as long as the initial estimate is good enough. We will therefore focus on the particularly challenging case $N_{\mathrm{p}} = 1$, where the initial estimate is so poor that the EM-type receiver shows a significant error floor.

In contrast, the proposed VMP-type soft estimator yields practically perfect results after just a single CA-iteration, as shown in Figure 3.29a. And as we see in Figure 3.29b, the soft estimator even performs quite close to the genie-aided receiver when it is run in an NCA-manner, though it takes three or four iterations in this case.

Both plots show results with a uniform and with Jeffrey's prior for the noise precision γ_0, and it is evident from the figures that the choice hardly matters. This is good news! Bayesian inference is sometimes criticized by orthodox statisticians for its requirement of assigning prior distributions to the unknowns, which, so the argument goes, is not possible in an objective manner. Therefore, the results of the inference process supposedly depend on subjective decisions made by the algorithm designer. At least for the particular case studied here, however, we have seen that the Bayesian approach is robust against the choice of the prior distribution—the data speaks for itself. And independently of the chosen prior, the Bayesian estimator outperforms the orthodox EM algorithm quite significantly. As discussed above, the main reason for this performance difference is the ability to express the constaint $g_0 \geq 0$ in the prior $p(g_0)$, so that the gain estimator is guaranteed to give a physically meaningful output. The orthodox estimator, on the other hand, may compute a negative gain, in which case the whole frame is lost.



(a) Code-Aided: both demapper and decoder are invoked in each iteration



(b) Non-Code-Aided: only the demapper is invoked, but not the decoder

Figure 3.29: Performance of the soft VMP-type SNR estimator, with uniform (U) and Jeffrey's (J) prior for γ_0, compared to the hard EM-type estimator, with $N_\mathrm{p} = 1$

3.7.4 Soft SNR Estimation Based on Inclusive DM

The soft VMP-type SNR estimator derived in the previous section clearly improves on the hard EM-type one. However, one pilot symbols is still required in order for it to bootstrap (as long as no informative prior is available). The reason is seen in (3.185): in the initial iteration, the expectations of the data symbols are all equal to zero, so if no pilot symbols were transmitted, $\bar{\boldsymbol{x}}$ would be the all-zero vector and $m_{\mathrm{p}} = 0$.

For phase estimation (and more generally, estimation of fading channels), the requirement of having at least one reference symbol, if only for resolving phase ambiguities caused by symmetries in the modulation alphabet [35], is fairly intuitive.[25] Coherent SNR estimation, however, should be possible in a purely NDA manner. To see this, consider a simple BPSK transmission with $\mathcal{X} = \{-1, +1\}$. Assume we receive $y = g_0 x + w = 2$. We do not know which $x \in \mathcal{X}$ was transmitted, so there are two hypotheses. Either $x = -1$, in which case the noise-free received signal $g_0 x$ would necessarily be negative, and w would have to be greater than 2. If the transmitted symbol were $x = +1$, however, a noise realization of $w \approx 0$ could explain our observation. The hypothesis $x = +1$ is thus more likely (assuming that the AWGN variance is not too large). This breaks the symmetry of $\mathcal{X}$, and we conclude that $g_0 \approx 2$, give or take a bit due to the noise.

Now that we have intuitively motivated that purely non-data-aided SNR estimation should be possible, we would like to derive an appropriate algorithm in a systematic manner. Interestingly, an NDA estimator emerges quite naturally in a *non-coherent* setting [45]. Consider for a moment the transmission model

$$y = g_0 \, x \, \mathrm{e}^{\mathrm{j}\vartheta_0} + w \tag{3.193}$$

and assume that the phase is completely unknown, so $p(\vartheta_0) = \frac{1}{2\pi}$. Then the likelihood function for g_0 and γ_0 is

$$\begin{aligned} p(y \mid x, g_0, \gamma_0) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} p(y \mid x, g_0, \gamma_0, \vartheta_0) \, \mathrm{d}\vartheta_0 \\ &= \frac{\gamma_0}{2\pi^2} \int_{-\pi}^{\pi} \exp\Big(-\gamma_0 \big(|y|^2 + g_0^2 \, |x|^2 - 2 g_0 \, \mathrm{Re}(y x^* \mathrm{e}^{-\mathrm{j}\vartheta_0})\big)\Big) \\ &= \frac{\gamma_0}{\pi} \exp\Big(-\gamma_0 \big(|y|^2 + g_0^2 \, |x|^2\big)\Big) \, \mathrm{I}_0 \left(2 g_0 \gamma_0 \, |y| \, |x|\right) \\ &\propto\!\!\!\!\propto \gamma_0 \, \exp\Big(-\gamma_0 \big(|y| - g_0 \, |x|\big)^2\Big) \end{aligned} \tag{3.194}$$

where $\propto\!\!\!\!\propto$ denotes "approximately proportional to", and the last line uses $\mathrm{I}_0(x) \propto\!\!\!\!\propto \exp(x)$, similarly as in the derivation of (3.85). The non-coherent likelihood function depends on

[25] A more relaxed requirement is that there are no two transmit sequences which differ only by a constant phase shift; formally $(\exists i, j, \vartheta : \boldsymbol{x}_i = \boldsymbol{x}_j \, \mathrm{e}^{\mathrm{j}\vartheta}) \Rightarrow i = j$. Having a fixed reference symbol is just one particularly simple way of ensuring that this condition holds.

the unknown symbol x only through its modulus. In a PSK modulation scheme, $|x|$ is known *a priori*, so the data symbols can be used for SNR estimation as if they were pilots. And even in non-constant-modulus transmissions, (3.194) can be used by replacing $|x|$ with its mean, since the uncertainty in $|x|$ will get averaged out for sufficiently long block lengths.

But what about the coherent signal model? In this case, we could still *pretend* that the phase was unknown and use (3.194) nonetheless. In all likelihood, this would be a pragmatically reasonable choice, since the resulting algorithm has a rather low complexity and can be expected to yield accurate results. Nonetheless, having to resort to such tricks is unsatisfactory from a theoretical point of view. We will therefore finish this section by deriving a non-data-aided SNR estimator which is based on the coherent signal model. The approach uses inclusive divergence minimization, so the resulting algorithm is of EP-type.[26] The following derivation is, admittedly, mainly an academic exercise, with the main purpose of demonstrating the generality of the DM framework. The algorithm does have a potential practical advantage, though: due to its sequential and Bayesian nature, it can easily be extended to time-varying parameters.

An Expectation Propagation Approach to Soft SNR Estimation

How can we derive an SNR estimator using inclusive DM? A straightforward application of belief propagation to the factor graph in Figure 3.27 is infeasible due to the presence of discrete ($\boldsymbol{x}_\mathcal{D}$) and continuous ($g_0, \gamma_0$) variables. Summing $p(\boldsymbol{y}_\mathcal{D} \,|\, \boldsymbol{x}_\mathcal{D}, g_0, \gamma_0)$ over $\boldsymbol{x}_\mathcal{D}$ leads to mixture distributions in g_0 and γ_0. For complexity reasons, we need to collapse them to simple distributions in the exponential family by means of projections, so we will use *expectation* rather than *belief* propagation.

However, applying EP to the structure in Figure 3.27 is still infeasible due to the huge range of $\boldsymbol{x}_\mathcal{D}$, whose cardinality is exponential in the block length. We will therefore process the observations sequentially, so that at each time instant, we only operate on mixture distributions with M components (recall that $M = |\mathcal{X}|$ is the size of the modulation alphabet). To that end, we replace the scalar variables g_0 and γ_0 with processes $\boldsymbol{g} = (g_0, \ldots, g_{N-1})$ and $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{N-1})$. We still assume that the gain and noise precision remain constant during one block, so we assign to them the prior PDFs

$$p(\boldsymbol{g}) = p(g_0) \prod_{n=1}^{N-1} \delta(g_n - g_{n-1}) \tag{3.195}$$

$$p(\boldsymbol{\gamma}) = p(\gamma_0) \prod_{n=1}^{N-1} \delta(\gamma_n - \gamma_{n-1}) \tag{3.196}$$

[26] Actually, the algorithm is an instance of *assumed density filtering* (ADF), because we only sweep through the data once, without going back and refining the estimates iteratively. As explained in [91], this iterative refinement is the generalization which distinguishes EP from the older ADF. For simplicity, however, we will nonetheless use the name *expectation propagation* in the following.

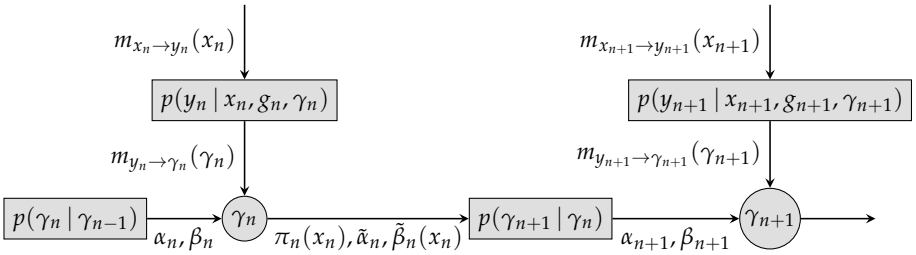


Figure 3.30: Detail of the factor graph which shows the relevant parts for the γ-estimator

but in principle, we could also allow for time-varying parameters by selecting the conditional distributions $p(g_n \mid g_{n-1})$ and $p(\gamma_n \mid \gamma_{n-1})$ appropriately. The model is very similar to that of the phase noise estimator from Section 3.5, where we have derived a Kalman-filter-like algorithm which, however, operated on von Mises distributions rather than Gaussians. The estimators for channel gain and noise precision that we will discuss in the following are again similar to the Kalman filter, but this time, they are based on truncated Gaussian and Gamma distributions, respectively. In contrast to the phase noise estimator, we will only perform a forward pass through the data, i. e., derive a filter rather than a smoother. This saves us from some mathematical trouble, and is perfectly sufficient to yield excellent results.

Estimating the Noise Precision

We begin with the noise precision estimator. The gain is assumed to be known in the following. The graphical model of the whole receiver is quite similar to that shown in Figure 3.16 on page 75, but with the Markov chain for $(\vartheta_0, \ldots, \vartheta_{N-1})$ replaced by that for $(\gamma_0, \ldots, \gamma_{N-1})$. We therefore only show a relevant detail of the model in Figure 3.30.

Measurement Update The measurement update at time instant n combines the forward message $m_{p(\gamma_n \mid \gamma_{n-1}) \to \gamma_n}(\gamma_n)$ with the information $m_{y_n \to \gamma_n}(\gamma_n)$ contained in the observation. The former is (proportional to) a Gamma distribution with parameters α_n and β_n. In particular, for $n = 0$, the forward recursion is initialized with the prior distribution $p(\gamma_0) = \mathcal{G}(\gamma_0 \mid \alpha_0, \beta_0)$, which could be chosen as a non-informative prior, just as we did in the previous section for the VMP-type estimator.

The message from the observation y_n is computed in the usual way, by marginalizing the likelihood function over x_n:

$$m_{y_n \to \gamma_n}(\gamma_n) \propto \sum_{x_n} m_{x_n \to y_n}(x_n)\, p(y_n \mid x_n, g_n, \gamma_n)$$

$$= \sum_{x_n} m_{x_n \to y_n}(x_n) \frac{\gamma_n}{\pi} \exp\left(-\gamma_n \left|y_n - g_n x_n\right|^2\right). \tag{3.197}$$

The two messages are merged at the variable node γ_n by multiplication:

$$\begin{aligned}
m_{\gamma_n \to p(\gamma_{n+1} \,|\, \gamma_n)}(\gamma_n) &\propto m_{p(\gamma_n \,|\, \gamma_{n-1}) \to \gamma_n}(\gamma_n)\, m_{y_n \to \gamma_n}(\gamma_n) \\
&= \frac{\beta_n^{\alpha_n}}{\Gamma(\alpha_n)} \gamma_n^{\alpha_n - 1} \exp(-\gamma_n \beta_n) \sum_{x_n} m_{x_n \to y_n}(x_n) \frac{\gamma_n}{\pi} \exp\left(-\gamma_n \left|y_n - g_n x_n\right|^2\right) \\
&\propto \sum_{x_n} m_{x_n \to y_n}(x_n)\, \gamma_n^{\alpha_n} \exp\left(-\gamma_n \left(\beta_n + \left|y_n - g_n x_n\right|^2\right)\right) \\
&\overset{(a)}{=} \sum_{x_n} m_{x_n \to y_n}(x_n)\, \gamma_n^{\tilde{\alpha}_n - 1} \exp\left(-\gamma_n \tilde{\beta}_n(x_n)\right) \\
&= \sum_{x_n} m_{x_n \to y_n}(x_n) \frac{\Gamma(\tilde{\alpha}_n)}{\tilde{\beta}_n(x_n)^{\tilde{\alpha}_n}} \frac{\tilde{\beta}_n(x_n)^{\tilde{\alpha}_n}}{\Gamma(\tilde{\alpha}_n)} \gamma_n^{\tilde{\alpha}_n - 1} \exp\left(-\gamma_n \tilde{\beta}_n(x_n)\right) \\
&\overset{(b)}{\propto} \sum_{x_n} \pi_n(x_n)\, \mathcal{G}\left(\gamma_n \,|\, \tilde{\alpha}_n, \tilde{\beta}_n(x_n)\right).
\end{aligned} \tag{3.198}$$

In (a) we have introduced the auxiliary parameters

$$\tilde{\alpha}_n \triangleq \alpha_n + 1 \tag{3.199}$$

$$\tilde{\beta}_n(x_n) \triangleq \beta_n + \left|y_n - g_n x_n\right|^2 \tag{3.200}$$

and in (b) the discrete distribution

$$\pi_n(x_n) \triangleq \frac{1}{Z} \frac{m_{x_n \to y_n}(x_n)}{\tilde{\beta}_n(x_n)^{\tilde{\alpha}_n}} \tag{3.201}$$

where Z is a normalization constant which ensures that $\sum_{x_n} \pi_n(x_n) = 1$.

As we see in (3.198), the forward message after the measurement update is a Gamma-mixture with M components, whose weights are given by the PMF $\pi_n(x_n)$.

Time Update The second part of the recursion step is the time update, where we convert the Gamma-mixture $m_{\gamma_n \to p(\gamma_{n+1} \,|\, \gamma_n)}(\gamma_n)$ into a single Gamma distribution $m_{p(\gamma_{n+1} \,|\, \gamma_n) \to \gamma_{n+1}}(\gamma_{n+1}) \propto \mathcal{G}(\gamma_{n+1} \,|\, \alpha_{n+1}, \beta_{n+1})$. Since we only perform forward filtering, the backward message $m_{\gamma_{n+1} \to p(\gamma_{n+1} \,|\, \gamma_n)}(\gamma_{n+1})$ is constant and can be ignored.

The EP message update (B.82) thus simplifies to

$$\begin{aligned}
m_{p(\gamma_{n+1} \,|\, \gamma_n) \to \gamma_{n+1}}(\gamma_{n+1}) &\propto \operatorname{proj}_{\mathcal{G}}\left[\int_0^\infty p(\gamma_{n+1} \,|\, \gamma_n)\, m_{\gamma_n \to p(\gamma_{n+1} \,|\, \gamma_n)}(\gamma_n)\, \mathrm{d}\gamma_n\right] \\
&= \operatorname{proj}_{\mathcal{G}}\left[\int_0^\infty \delta(\gamma_{n+1} - \gamma_n) \sum_{x_n} \pi_n(x_n)\, \mathcal{G}\left(\gamma_n \,|\, \tilde{\alpha}_n, \tilde{\beta}_n(x_n)\right)\, \mathrm{d}\gamma_n\right]
\end{aligned}$$

$$= \operatorname{proj}_{\mathcal{G}} \left[\sum_{x_n} \pi_n(x_n)\, \mathcal{G}\left(\gamma_{n+1} \mid \tilde{\alpha}_n, \tilde{\beta}_n(x_n)\right) \right]. \tag{3.202}$$

The sufficient statistics of the Gamma distribution are $\log\gamma$ and γ (c. f. (A.94)). According to the moment matching equation (B.39), we would therefore have to compute $\mathbb{E}[\log\gamma_{n+1}]$ and $\mathbb{E}[\gamma_{n+1}]$, and then select α_{n+1} and β_{n+1} by inverting (A.96) and (A.97). Unfortunately, however, $\mathbb{E}[\log\gamma]$ involves the digamma function, which precludes a closed-form expression for α_{n+1} and β_{n+1}. We will therefore approximate the projection step by matching the first and second moment instead, as if we were projecting into the family of Gaussians. We will see later by simulations that the algorithm still gives very good results.

Since $\mathbb{E}[\gamma] = \frac{\alpha}{\beta}$ and $\operatorname{var}[\gamma] = \frac{\alpha}{\beta^2}$, the second moment of the Gamma distribution is

$$\mathbb{E}\left[\gamma^2\right] = \frac{\alpha^2}{\beta^2} + \frac{\alpha}{\beta^2} = \frac{\alpha^2+\alpha}{\beta^2} \tag{3.203}$$

and a simple calculation shows that the parameters of a Gamma distribution with given first and second moment are

$$\alpha = \frac{(\mathbb{E}[\gamma])^2}{\mathbb{E}[\gamma^2] - (\mathbb{E}[\gamma])^2} \tag{3.204}$$

$$\beta = \frac{\mathbb{E}[\gamma]}{\mathbb{E}[\gamma^2] - (\mathbb{E}[\gamma])^2}. \tag{3.205}$$

Due to the linearity of the expectation operator, the first two moments of the mixture distribution in (3.202) are

$$\mathbb{E}[\gamma_{n+1}] = \sum_{x_n} \pi_n(x_n)\, \frac{\tilde{\alpha}_n}{\tilde{\beta}_n(x_n)} = \tilde{\alpha}_n \sum_{x_n} \frac{\pi_n(x_n)}{\tilde{\beta}_n(x_n)} \tag{3.206}$$

$$\mathbb{E}\left[\gamma_{n+1}^2\right] = \sum_{x_n} \pi_n(x_n)\, \frac{\tilde{\alpha}_n^2 + \tilde{\alpha}_n}{\tilde{\beta}_n(x_n)} = (\tilde{\alpha}_n^2 + \tilde{\alpha}_n) \sum_{x_n} \frac{\pi_n(x_n)}{\tilde{\beta}_n(x_n)} \tag{3.207}$$

and inserting those into (3.204) and (3.205) yields α_{n+1} and β_{n+1}, which concludes one step of the forward recursion.

When the forward recursion is finished, the algorithm outputs the soft estimate $\mathcal{G}(\alpha_N, \beta_N)$, or, if sufficient, the MMSE-like hard estimate $\bar{\gamma}_N = \frac{\alpha_N}{\beta_N}$.

Estimating the Channel Gain

The channel gain estimator works analogously, but it tracks truncated Gaussians instead of Gamma distributions. In the following, we assume that the noise precision is known.

Measurement Update The forward message $m_{p(g_n \mid g_{n-1}) \to g_n}(g_n)$ is a truncated Gaussian with parameters m_n and s_n^2. At the variable node g_n, it is multiplied with

$$\begin{aligned} m_{y_n \to g_n}(g_n) &\propto \sum_{x_n} m_{x_n \to y_n}(x_n)\, p(y_n \mid x_n, g_n, \gamma_n) \\ &\propto \sum_{x_n} m_{x_n \to y_n}(x_n)\, \exp\left(-\gamma_n \left|y_n - g_n x_n\right|^2\right) \end{aligned} \tag{3.208}$$

which results in a mixture of truncated Gaussians:

$$\begin{aligned} & m_{g_n \to p(g_{n+1} \mid g_n)}(g_n) \propto m_{p(g_n \mid g_{n-1}) \to g_n}(g_n)\, m_{y_n \to g_n}(g_n) \\ & \propto \mathbb{1}\left[g_n \geq 0\right] \exp\left(-\frac{(g_n - m_n)^2}{2s_n^2}\right) \sum_{x_n} m_{x_n \to y_n}(x_n)\, \exp\left(-\gamma_n \left|y_n - g_n x_n\right|^2\right) \\ & = \mathbb{1}\left[g_n \geq 0\right] \sum_{x_n} m_{x_n \to y_n}(x_n)\, \exp\left(-\frac{g_n^2}{2}\left(\frac{1}{s_n^2} + 2\gamma_n \left|x_n\right|^2\right) + g_n\left(\frac{m_n}{s_n^2} + 2\gamma_n \operatorname{Re}(x_n^* y_n)\right)\right) \\ & \qquad \times \exp\left(-\frac{m_n^2}{2s_n^2} - \gamma_n \left|y_n\right|^2\right) \\ & \overset{(a)}{=} \mathbb{1}\left[g_n \geq 0\right] \sum_{x_n} m_{x_n \to y_n}(x_n)\, \exp\left(-\frac{g_n^2}{2\tilde{s}_n^2(x_n)} + g_n \frac{m_n + 2\gamma_n s_n^2 \operatorname{Re}(x_n^* y_n)}{s_n^2} - \frac{m_n^2 + 2\gamma_n s_n^2 \left|y_n\right|^2}{2s_n^2}\right) \\ & \overset{(b)}{=} \mathbb{1}\left[g_n \geq 0\right] \sum_{x_n} m_{x_n \to y_n}(x_n)\, \exp\left(-\frac{(g_n - \tilde{m}_n(x_n))^2}{2\tilde{s}_n^2(x_n)}\right) \\ & \qquad \times \exp\left(-\frac{1}{2\tilde{s}_n^2(x_n)}\left(\frac{m_n^2 + 2\gamma_n s_n^2 \left|y_n\right|^2}{1 + 2\gamma_n s_n^2 \left|x_n\right|^2} - \tilde{m}_n^2(x_n)\right)\right) \\ & = \mathbb{1}\left[g_n \geq 0\right] \sum_{x_n} m_{x_n \to y_n}(x_n) \frac{F_{\mathcal{N}}\left(\frac{\tilde{m}_n(x_n)}{\tilde{s}_n(x_n)}\right) \tilde{s}_n(x_n)}{F_{\mathcal{N}}\left(\frac{\tilde{m}_n(x_n)}{\tilde{s}_n(x_n)}\right) \tilde{s}_n(x_n)}\, \exp\left(-\frac{(g_n - \tilde{m}_n(x_n))^2}{2\tilde{s}_n^2(x_n)}\right) \\ & \qquad \times \exp\left(-\frac{\tilde{s}_n^2(x_n)}{s_n^2}\gamma_n \left|y_n - m_n x_n\right|^2 - 2\gamma_n^2 \tilde{s}_n^2(x_n)\left(\operatorname{Im}(x_n^* y_n)\right)^2\right) \\ & \overset{(c)}{\propto} \sum_{x_n} \pi_n(x_n)\, \mathcal{TN}\left(g_n \mid \tilde{m}_n(x_n), \tilde{s}_n^2(x_n)\right). \end{aligned} \tag{3.209}$$

Here, $F_{\mathcal{N}}(x)$ is the CDF of the $\mathcal{N}(0,1)$ distribution, (a) and (b) introduce the auxiliary parameters

$$\tilde{s}_n^2(x_n) \triangleq \frac{s_n^2}{1 + 2\gamma_n s_n^2 \left|x_n\right|^2} \tag{3.210}$$

$$\tilde{m}_n(x_n) \triangleq \frac{m_n + 2\gamma_n s_n^2 \operatorname{Re}(x_n^* y_n)}{1 + 2\gamma_n s_n^2 \left|x_n\right|^2} \tag{3.211}$$

and (c) defines the discrete distribution

$$\pi_n(x_n) = \frac{1}{Z}\, m_{x_n \to y_n}(x_n)\, F_{\mathcal{N}}\left(\frac{\tilde{m}_n(x_n)}{\tilde{s}_n(x_n)}\right) \tilde{s}_n(x_n) \times \exp\left(-\frac{\tilde{s}_n^2(x_n)}{s_n^2}\gamma_n \left|y_n - m_n x_n\right|^2 - 2\gamma_n^2 \tilde{s}_n^2(x_n)\left(\operatorname{Im}(x_n^* y_n)\right)^2\right) \tag{3.212}$$

where Z is again a normalization constant.

Time Update Analogously to the γ-estimator discussed above, the time update collapses the mixture of truncated Gaussians (3.209) to a single truncated Gaussian:

$$\begin{aligned} m_{p(g_{n+1}\,|\,g_n)\to g_{n+1}}(g_{n+1}) &\propto \operatorname{proj}_{\mathcal{TN}}\left[\int_0^\infty p(g_{n+1}\,|\,g_n)\, m_{g_n \to p(g_{n+1}\,|\,g_n)}(g_n)\,\mathrm{d}g_n\right] \\ &= \operatorname{proj}_{\mathcal{TN}}\left[\sum_{x_n} \pi_n(x_n)\, \mathcal{TN}\left(g_{n+1}\,|\,\tilde{m}_n(x_n), \tilde{s}_n^2(x_n)\right)\right]. \end{aligned} \tag{3.213}$$

To that end, we compute the first two moments of g_{n+1} via

$$\mathbb{E}\left[g_{n+1}\right] = \sum_{x_n} \pi_n(x_n)\left(\tilde{m}_n(x_n) + \tilde{s}_n(x_n)\,\lambda\left(\frac{\tilde{m}_n(x_n)}{\tilde{s}_n(x_n)}\right)\right) \tag{3.214}$$

$$\mathbb{E}\left[g_{n+1}^2\right] = \sum_{x_n} \pi_n(x_n)\left(\tilde{m}_n^2(x_n) + \tilde{s}_n^2(x_n) + \tilde{m}_n(x_n)\,\tilde{s}_n(x_n)\,\lambda\left(\frac{\tilde{m}_n(x_n)}{\tilde{s}_n(x_n)}\right)\right) \tag{3.215}$$

with $\lambda(x)$ defined in (A.92) on page 217. Now, we would have to find m_{n+1} and s_{n+1}^2 by inverting (A.90) and (A.91), but for complexity reasons, we approximate this step and simply set

$$m_{n+1} = \mathbb{E}\left[g_{n+1}\right] \tag{3.216}$$

$$s_{n+1}^2 = \mathbb{E}\left[g_{n+1}^2\right] - \left(\mathbb{E}\left[g_{n+1}\right]\right)^2. \tag{3.217}$$

After the forward recursion, the algorithm outputs the soft estimate $\mathcal{TN}(m_N, s_N^2)$.

Iterations and Initialization

The estimator for the noise precision assumes perfect knowledge of the channel gain and vice versa. We could thus iterate between the two estimators, but it turns out that estimating γ and g just once (in that order) gives already very good results.

The gain estimator uses the freshly estimated $\bar{\gamma}$, so the only remaining question is how to initialize the estimate of g which is used by the noise precision estimator. In the

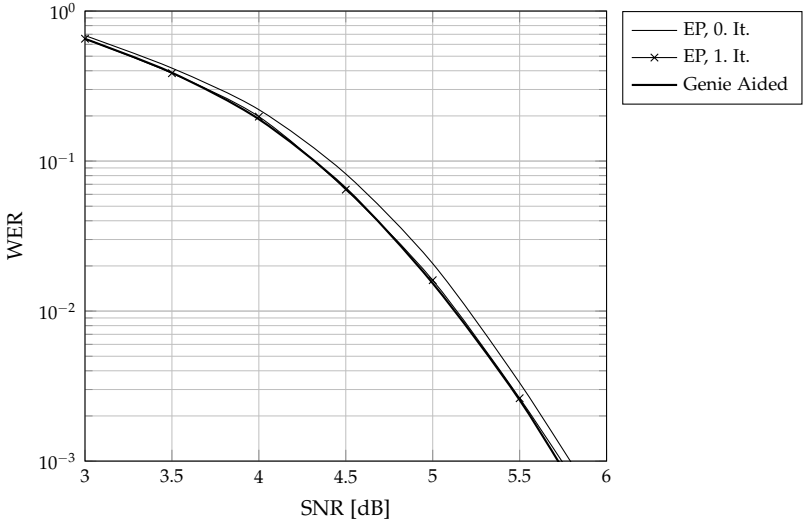


Figure 3.31: Performance of the EP-based SNR estimator, with $N_\mathrm{p} = 0$

following simulation, we use a simple *ad hoc* estimator: the expected squared norm of the received data is

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{y}\|^2\right] &= g_0^2\mathbb{E}\left[\|\boldsymbol{x}\|^2\right] + \mathbb{E}\left[\|\boldsymbol{w}\|^2\right] \\ &= g_0^2 N E_\mathrm{s} + N/\gamma_0. \end{aligned} \tag{3.218}$$

Ignoring the noise power, and replacing the expectation of $\|\boldsymbol{y}\|^2$ with its observed realization, a coarse estimate of g_0 is found as

$$\hat{g}_0 = \sqrt{\frac{\|\boldsymbol{y}\|^2}{NE_\mathrm{s}}}. \tag{3.219}$$

There may be more sophisticated strategies, but as the following simulation shows, any attempts to improve the proposed estimator further are likely to be in vain.

Numerical Results

Figure 3.31 shows simulation results using the rate-1/3 turbo code and 16-QAM modulation as in the previous sections, but this time without any training data, $N_\mathrm{p} = 0$.

We see that even in the initial iteration (without feedback from the decoder), the gap to the genie aided receiver is less than 0.1 dB. And after a single re-estimation with

decoder feedback, the performance is not distinguishable anymore from the genie-aided lower bound.

3.7.5 Concluding Remarks

We conclude our case study of SNR estimation with a brief wrap-up.

Arguably the state-of-the-art approaches to code-aided SNR estimation are based on the EM algorithm, with several publications having appeared just recently [26,45,142,152]. We have therefore started this case study in Section 3.7.2 by deriving an EM-based estimator for the coherent channel model.

As discussed in Section B.5.5, the hard-output EM algorithm is closely related to the soft-output VMP framework, with two key differences: (a) VMP is a Bayesian algorithm and thus models the unknown parameters as random variables, and (b) EM reduces the beliefs to point estimates by taking their argmax. The EM-type SNR estimator can therefore be converted to a Bayesian soft-output algorithm in a natural and elegant way. We have seen in Section 3.7.3 that the VMP-type estimator has virtually the same computational complexity as its EM-type counterpart, but it allows us to reduce the number of pilot symbols that are required in order to obtain close-to-optimal results.

VMP is a special case of the generic DM framework, which minimizes the *exclusive* KL divergence. In Section 3.7.4, we have finally switched to *inclusive* DM and derived EP-type estimators for the noise precision and the channel gain. Both estimators bear some similarity to the Kalman filter, but they operate on Gamma and truncated Gaussian distributions, respectively, rather than plain Gaussians. The proposed receiver is able to work in a purely NDA-manner, i. e., without any training data. And as we have shown by simulations, it gives close-to-optimal results already in the initial NCA iteration, where it only exploits the discrete modulation alphabet, but not the coding constraints. Its drawback, compared to the VMP-type estimator, is a significantly higher computational complexity.

Chapter 4

MIMO Detection

In the previous chapter, we have studied parameter estimators for conventional wireless systems, where both the transmitter and receiver are equipped with a single antenna. Since the radio channel of these systems has one input and one output port, they are commonly called *single-input single-output* (SISO) systems. Owing to the development of modern channel codes and iterative receivers, it is now possible to operate those systems fairly close to the channel capacity. This means that any further progress in the area of receiver design can only be expected to reap diminishing gains in the achievable data rates, since the channel capacity constitutes a fundamental upper bound.

It is thus necessary to modify the channel itself, and hence its capacity, in order to improve the data rates of wireless systems by any significant factor. One possibility is to increase the bandwidth of the radio channel over which the system operates; and indeed, the trend towards wider frequency bands can be witnessed in the evolution of both cellular mobile systems and wireless LAN standards. For example, while the 3GPP *Universal Mobile Telecommunications System* (UMTS) uses a bandwidth of 5 MHz, the *Long Term Evolution* (LTE) standard, release 8, is specified for a maximum of 20 MHz. And a major feature of release 10, known as *LTE Advanced*, it a technique called *carrier aggregation*, which allows the system to bundle up to five release-8 component carriers to obtain a total bandwidth of 100 MHz [24, Section 7.3.1].

However, spectrum is a scarce and expensive resource that cannot grow ad infinitum. It is therefore also important to improve the spectral *efficiency* in order to utilize the available bandwidth as well as possible. A major breakthrough in this regard has been the development of multi-antenna techniques. *Multiple-input multiple-output* (MIMO) systems, where both ends of the link are equipped with several antenna elements,[1] are able to transmit multiple data streams in parallel over the same frequency band by exploiting the spatial dimension of the radio channel. Due to their significant

[1] In the special cases where only the transmitter or the receiver has more than one antenna, we also speak of *MISO* and *SIMO* systems, respectively.

performance benefits which do not necessitate an increase of resources like bandwidth or transmit power, MIMO techniques are a central component of modern wireless systems like the above mentioned 3GPP LTE, or the IEEE WLAN standard 802.11n.

From the receiver's point of view, the transition from SISO to MIMO does not introduce any fundamental obstacles. Both the transmitted and the received signal become vector-valued, but the BICM receiver from Chapter 2 can easily be adapted to the new channel model (see Section 4.1.3). Instead, the main challenge that MIMO receivers face is a very practical one. As shown in (2.23) on page 18, the BICM demapper involves an enumeration of all potentially transmitted signals. In a SISO system which sends an M-QAM-modulated data stream, there are M possible signals per symbol period, typically with $M \in \{4, 16, 64\}$, which is sufficiently small so that (2.23) can be implemented verbatim.[2] However, in a MIMO system that transmits N_s parallel data streams, there are M^{N_s} possible transmit signals (assuming for simplicity that all data streams share the same modulation alphabet). Even for moderate N_s of, say, three or four, this number is so large that a real-time implementation of (2.23) becomes infeasible.

For this reason, the past years have witnessed a tremendous amount of research on suboptimal MIMO detection schemes with lower complexity. *Linear* detectors are a family of particularly simple algorithms. They separate the spatially superimposed data streams (to some extent) by applying a linear filter to the received signal. Subsequently, the streams are processed by a bank of N_s independently working scalar demappers.

While the idea of using linear filters for the purpose of MIMO detection is certainly far from being novel, the existing literature derives them in a very *ad hoc* fashion: the general structure of the algorithm is fixed in advance, the filter matrix is derived based on some heuristic optimization criterion (usually the LMMSE criterion), and the output of the filter is somehow connected to the rest of the receiver. In the context of conventional non-iterative receivers, this approach works sufficiently well, and the lack of a solid theoretical fundament could be regarded as a purely academic issue. However, the problem of this approach becomes apparent as soon as one attempts to use a linear detector in an iterative BICM-ID receiver, as we shall see later within this chapter.

Since the root cause is the isolated design of the linear MIMO detectors, which are retroactively plugged into the BICM-ID algorithm, the aim of this chapter is to recover a holistic system perspective which will allow us to derive the receiver in one piece. This gives rise to an interesting question: how can approximations be selectively introduced into a systematic algorithm design? Clearly, sticking to belief propagation will be insufficient, as this only leads to the standard BICM-ID receiver.

Divergence minimization is a more general and flexible approach to the design of approximate Bayesian inference algorithms (see Appendix B), which we have successfully used in the previous chapter to derive several novel parameter estimators. Confirmed by these results, we shall now explore the applicability of DM to detection problems.

[2] For many practically relevant modulation schemes, there also exist closed expressions for the demapper which have an even lower complexity.

This chapter is structured as follows. After providing a brief overview of multi-antenna communication systems, Section 4.1 introduces the MIMO channel model and the spatial multiplexing technique, which will serve as the fundament for the following derivations. It then sketches the above mentioned modifications that are necessary to adapt the BICM-ID receiver from Chapter 2 to multiple data streams.

Section 4.2 presents some important conventional low-complexity MIMO detectors, which will serve as a performance benchmark for the novel detectors. Also, working through the traditional derivation helps us to appreciate the benefits of the DM approach.

The main contribution of this chapter is presented in Section 4.3. We derive an iterative MIMO detector where an affine front-end, consisting of interference cancellation and LMMSE filtering, and a bank of streamwise demappers are connected by a loop whose structure has, to my best knowledge, not been published before. The proposed algorithm is an instance of expectation propagation, since the complexity reduction with respect to the exact MIMO detector is achieved by constraining the belief of the data symbols to Gaussian distributions, i. e., by projecting the discrete symbolwise distributions into the set of proper Gaussians. This turns the sum over all possible transmitted signals as in (2.23) into an integral which admits a closed-form solution. Furthermore, we briefly discuss a possible extension to *widely* linear filters.

Finally, in Section 4.4, we try a different approximation of the symbolwise beliefs: we modify the objective function of the underlying optimization problem rather than the feasible set, and minimize the exclusive instead of the inclusive KL divergence. The resulting algorithm is thus an instance of variational message passing. Its most obvious difference with respect to the detector from Section 4.3 is that the LMMSE filter gets replaced by a simple maximum ratio combiner. But as we will see, there is also a second, more subtle difference, namely that the symbol statistics which are required by the affine front-end must now be computed based on *a posteriori* information. If we had just substituted the filter in an *ad hoc* way, with the intention of reducing the computational complexity of the front-end, we would most likely have missed this necessary modification. This case study thus nicely demonstrates the benefit of the holistic system perspective that the divergence minimization approach provides us with.

In order to keep things as simple as possible, the developments in this chapter assume a perfectly synchronized system, so that we can focus on the detection problem. However, it goes without saying that the proposed MIMO detectors can be combined with the estimation techniques from the previous chapter, yielding iterative receivers which perform synchronization, detection, and decoding jointly. Modularity is, after all, an inherent characteristic of our graph-based Bayesian inference framework.

4.1 Introduction to Multi-Antenna Communication Systems

SIMO and MISO systems have been studied for several decades. The papers [47,48] from 1997 provide an accessible overview of traditional multi-antenna techniques, which can roughly be categorized into beamforming and diversity approaches. In the former case, one end of the link (typically the base station) is equipped with an antenna array, which is used to concentrate the radiated energy into a certain direction, determined by the phase difference of the signals at the individual antenna elements. By steering the beam towards a particular mobile receiver, it increases the received power and hence the SNR of that link, while at the same time reducing the interference experienced by other users. Due to reciprocity, the same principle can be applied at the receiver side: by coherently combining the individual received signals, the sensitivity of the antenna array is focused in the direction of the desired signal, while interfering signals that arrive from different directions are suppressed. Beamforming thus works best when the link is dominated by a single propagation path, for example a line-of-sight (LOS) path.

Diversity techniques, on the other hand, aim to increase the reliability of the radio link by exploiting the multipath propagation that occurs in rich scattering environments. They, too, can be used at both the transmitter and the receiver, but exploiting *receive diversity* is arguably simpler. It is a well-known phenomenon that multipath propagation causes variations of the received signal power by several orders of magnitude along distances on the order of half a wavelength (i. e., just a few centimeters for the frequencies that are used by mobile systems). Thus, if the antennas are sufficiently separated, they receive the signal with approximately uncorrelated fading coefficients, and the probability that all links are simultaneously in outage decreases with an increasing number of antennnas. A simple *maximum ratio combiner* (MRC) can then be used to convert the SIMO channel into an effective scalar channel, which has the maximum effective SNR among all linear receivers.[3] Exploiting *transmit diversity* in systems without *channel state information* (CSI) at the transmitter is less straightforward. A practically relevant approach which has spawned a lot of research are the so-called *space-time codes* like for example the famous Alamouti scheme [4].

While MISO and SIMO techniques certainly have a positive impact on the quality of the radio transmission, it is the idea to employ multiple antennas at *both* ends of the link that has led to a major paradigm shift in the wireless communication world. The crucial property of MIMO systems is that they provide additional degrees of freedom, which leads to a significantly increased channel capacity, particularly in the medium-to-high SNR range. This can be explained as follows. The beamforming techniques mentioned above achieve a *power gain*, which can be as large as the number of antennas: if N antennas are used at either the transmitter or the receiver, the SNR can be increased by a factor of up to N compared to a SISO system. If the system operates in the *power-limited*

[3] A detailed discussion of MRC receivers in the MIMO context is given in Section 4.4, but essentially the MRC filter is just a multiplication of the received signal with the Hermitian of the channel matrix.

regime (i. e., at low SNR), the channel capacity scales roughly linearly with the SNR, and beamforming can lead to an N-fold increase of the data rate. In modern cellular systems, this is a relevant scenario for cell-edge users, who can thus benefit from beamforming. However, in the *bandwidth-limited* regime (whose practical importance is witnessed by the fact that mobile operators are willing to spend billions of Euros on a few megahertz of licensed spectrum), the channel capacity only scales logarithmically with the SNR, and the impact of the power gain becomes negligible. In this case, the additional degrees of freedom provided by MIMO become relevant: as a rule of thumb, if the transmitter employs N_t antennas and the receiver N_r antennas, then the capacity of the MIMO channel scales linearly with $\min(N_\mathrm{t}, N_\mathrm{r})$. Detailed studies of the MIMO capacity in different scenarios can be found in the landmark papers by Foschini and Gans [39] and Telatar [128], and in an overview paper by Goldsmith *et al* [50]. Furthermore, the textbook [131] by Tse and Viswanath contains a well readable treatment of many important theoretical and practical aspects of MIMO systems.

We continue this section with an introduction of the MIMO channel, followed by a discussion of the well-known diversity-multiplexing tradeoff, and an explanation of why we will focus on spatial multiplexing (SM) in the remainder of this chapter. Finally, we derive the exact MIMO detector, whose prohibitive complexity motivates the study of low-complexity detection algorithms in the subsequent sections.

4.1.1 The MIMO Channel

We consider a communication system with N_t antennas at the transmitter, and N_r antennas at the receiver. We will refer to this configuration as an $N_\mathrm{t} \times N_\mathrm{r}$ MIMO system. We assume a perfect synchronization in time and frequency. As explained at the end of Section 3.3, any residual inter-symbol interference due to imperfect synchronization is understood to be included in the noise term (the ISI power is assumed to be sufficiently small, such that the noise-plus-interference term is still approximately Gaussian distributed). We furthermore assume a non-dispersive (frequency-flat) fading channel, but the following derivations are also valid for OFDM transceivers over frequency-selective channels, where the detectors operate on a per-subcarrier basis. The single-antenna channel model, which will be generalized to multiple antennas in the following, is thus

$$y_n = h_n x_n + w_n, \qquad w_n \sim \mathcal{CN}(0, N_0) \tag{4.1}$$

which is similar to (3.62), but where we have merged the channel gain g and the phase offset ϑ to a complex channel weight h, which is furthermore allowed to be time-varying. The reason is that the spatial multiplexing technique, which will be described in more detail below, does not work well in line-of-sight scenarios, where the channel gain is constant and only the phase is varying due to mobility and/or phase noise. In such a scenario, the MIMO channel matrix would be poorly conditioned, and rank-1 transmission would become the optimal choice (which can be shown by waterfilling

over the eigenmodes of the channel; see e. g. [131, Section 7.1]). Instead, we will work with the standard Rayleigh fading model, which is based on the assumption of many independent propagation paths, such that $h_n \sim \mathcal{CN}(0, \sigma_h^2)$ due to the central limit theorem. For simplicity, we assume the normalization $\sigma_h^2 = 1$, so that the received energy per symbol duration is, on average, equal to the transmitted energy E_s.[4]

In the presence of multiple transmit and receive antennas, each link can be described by a complex channel weight $h_{n,j,i}$ with $0 \leq i < N_\mathrm{t}$ and $0 \leq j < N_\mathrm{r}$. Assuming a linear propagation channel, the signal received by the jth antenna at time n is

$$y_{n,j} = \sum_{i=0}^{N_\mathrm{t}-1} h_{n,j,i} x_{n,i} + w_{n,j} \tag{4.2}$$

where $x_{n,i}$ is the symbol that is transmitted from the ith antenna at time n.

Collecting the symbols that are transmitted from all N_t transmit antennas at time n into the vector $\boldsymbol{x}_n \triangleq (x_{n,0}, \ldots, x_{n,N_\mathrm{t}-1})^\mathsf{T}$, and the received signals similarly into the vector $\boldsymbol{y}_n \triangleq (y_{n,0}, \ldots, y_{n,N_\mathrm{r}-1})^\mathsf{T}$, we obtain the standard matrix-vector description of a flat-fading MIMO channel

$$\boldsymbol{y}_n = \boldsymbol{H}_n \boldsymbol{x}_n + \boldsymbol{w}_n. \tag{4.3}$$

The additive noise is assumed to be temporally and spatially uncorrelated, so that $\boldsymbol{w}_n \sim \mathcal{CN}(\boldsymbol{0}_{N_\mathrm{r}}, N_0 \boldsymbol{I}_{N_\mathrm{r}})$, where $\boldsymbol{0}_N$ is the all-zero vector of dimension N.

Due to the Rayleigh fading assumption mentioned above, the time-varying channel matrix is modeled as a Gaussian random process. It remains to specify the correlation between the channel coefficients, both spatially and temporally.

Spatial Correlation The spatial correlation of the channel coefficients, i. e., the correlation between the entries of the channel matrix $\boldsymbol{H}_n$ for one time instant n, depends on the angular spread of the arriving multipath components. Line-of-sight channels are an extreme case: due to the single dominating path, the angular spread is close to zero and the channel coefficients are maximally correlated. Rich scattering environments, where the radio waves arrive omnidirectionally, are the other extreme. Here, the entries of $\boldsymbol{H}_n$ are practically uncorrelated.

As argued above, the spatial multiplexing technique does not work well in LOS scenarios; it requires a well-conditioned channel matrix. For the purpose of the following discussion, we can thus safely assume that the receiver operates in a rich scattering environment with low spatial correlation. For simplicity, all simulation results that are presented in this chapter have been obtained with spatially uncorrelated channels.

[4] Note that we do *not* normalize the symbol energy E_s or the noise power spectral density N_0 to unity, as is often done in the literature. The difference between h on the one hand, and x and w on the other hand, is that h is a unit-less multiplicative gain, while x and w have the physical unit $\sqrt{\mathrm{J}}$, or in other words, their variances E_s and N_0 are energies, measured in Joule. By setting, say, $E_\mathrm{s} = 1$ and thus discarding the J, we would needlessly eliminate the ability to do quick sanity-checks on our results by looking at the units. This is not just an academic exercise in mathematical rigor, but has a very practical relevance: I have seen several erroneous derivations in the literature which could have easily been spotted by checking the physical units of the equations.

Temporal Correlation A standard model for the temporal channel correlation, which is based on the assumption of omnidirectionally arriving multipath components, is given by *Clarke's model* [21]. Its discrete-time autocorrelation function of each individual link is

$$R_h(m) \triangleq \mathbb{E}\left[h_{n+m}h_n^*\right] = \mathrm{J}_0(2\pi\nu_{\max}Tm) \tag{4.4}$$

where $\nu_{\max}T \in [0, 0.5]$ is the normalized Doppler spread, and $\mathrm{J}_0(x)$ is the Bessel function of the first kind and zeroth order. Note that the channel is assumed to be wide-sense stationary, so that R_h does not depend on n.

It is common practice to concentrate on the two extreme cases of *quasi-static* and *ergodic* channels. Quasi-static means that the channel stays approximately constant during the transmission of each codeword, so that each transmission only experiences a single realization of the fading process. In this scenario, the receiver will be able to feed the instantaneous channel coefficients back to the transmitter with acceptable overhead. But if CSI is available at the transmitter, it can use *eigen-beamforming* which decomposes the MIMO channel into a set of mutually independent SISO channels (c. f. (4.6) below), and the problem that we will study in this chapter, namely the *joint* detection of all simultaneously transmitted data symbols, does not arise to begin with.

We will thus focus on the fast-fading regime, where the channel varies significantly over the duration of one codeword, and instantaneous CSI feedback becomes infeasible. In the limit of a very rapidly fading process (relative to the codeword length), the transmission of every codeword experiences, in some sense, all possible channel matrices, and the temporal average of the fading process approaches the ensemble average. In this case, we speak of an *ergodic* channel, and it becomes meaningful to study the Shannon capacity as an upper bound on the achievable data rate, which can be used as a benchmark for practical transmission schemes.[5] Due to the bit interleaver that is used in BICM systems, adjacent bits are mapped to data symbols which are transmitted at some temporal distance, and which are likely to experience independent channel realizations. Thus, we use temporally uncorrelated channels in the numerical perfomance evaluations, which is a standard assumption in the literature on MIMO detection.

4.1.2 The MIMO Transmitter

Having established the fundamental characteristics of the MIMO channel, the question arises how the multiple available transmit and receive antennas should be used. As motivated in the previous section, we will only be interested in fast fading scenarios, where the transmitter does not know the channel coefficients. The receiver, on the other hand, is assumed to have perfect knowledge of the instantaneous channel coefficients.[6]

[5] In the quasi-static regime, the Shannon capacity of a Rayleigh fading channel is zero, and one must resort to other, more appropriate measures, like for example the outage probability.

[6] Which is of course an unrealistic assumption. More accurately, the receiver is assumed to be *mismatched* in the terminology from Section 3.1, meaning that it estimates the channel coefficients and pretends that the estimates were the true values, ignoring the inevitable estimation errors.

Diversity vs Multiplexing

In the introduction to this chapter, we mentioned that traditional SIMO and MISO systems provide a diversity gain (assuming sufficiently uncorrelated channel coefficients). In SIMO systems, the receiver observes N_r approximately independently faded replicas of the transmitted signal, and can achieve a diversity order of N_r by using a simple MRC filter. In the MISO case, the transmitter emits N_t signals, which fade independently and then superimpose at the receiver. Those systems can realize a diversity order of N_t by using appropriate signaling schemes like space-time codes.

These diversity techniques can be generalized to MIMO systems, which offer a diversity gain of up to $N_\mathrm{t}N_\mathrm{r}$, the number of links between transmitter and receiver. However, as mentioned above, MIMO systems also offer additional degrees of freedom, which can be used to increase the data rate by multiplexing up to $\min(N_\mathrm{t}, N_\mathrm{r})$ data streams in the spatial domain; but this technique, known as *spatial multiplexing* (SM), does not provide any diversity gain. It thus seems that the system designer must choose between using the antennas for a lower error probability, but with the same data rate as an otherwise equivalent SISO system, or for a higher data rate, but with a poor link reliability. In fact, those two choices are the extreme ends of the well-known *diversity-multiplexing tradeoff* [158], which quantifies the maximum diversity that can be reached for a given multiplexing gain, and *vice versa*.

Now, which point on the diversity-multiplexing tradeoff curve should one attempt to realize? This question has been studied in the recent work [79, 80] by Lozano and Jindal. They argue that "in the context of most modern wireless systems and for the operating points of interest, transmission techniques that utilize all available spatial degrees of freedom for multiplexing outperform techniques that explicitly sacrifice spatial multiplexing for diversity" [79, Abstract]. Their rationale is concisely summarized by the following bullet points, quoted from [79, Section I.B]:

- "Modern systems use link adaptation to maintain a target error probability and there is essentially no benefit in operating below this target. This makes diversity metrics, which quantify the speed at which error probability is driven to zero with the signal-to-noise ratio, beside the point.

- Wireless channels in modern systems generally exhibit a notable amount of time and frequency selectivity, which is naturally converted into diversity benefits through coding and interleaving. This renders additional transmit diversity superfluous.

- Block error probability is the relevant measure of reliability. Since the channel codes featured in contemporary systems allow for operation close to information-theoretic limits, such block error probability is well approximated by the mutual information outages. Although uncoded error probability is often quantified, this

is only an indirect performance measure and incorrect conclusions can be reached by considering only uncoded performance."

As argued above in the context of temporal channel correlation, the focus of this work is on fast fading channels. Our target systems thus offer some amount of time diversity by assumption, in addition to the frequency diversity that is inherently present in modern broadband systems like 3GPP LTE and IEEE 802.11n. We will thus follow the reasoning from Lozano and Jindal, and only consider spatial multiplexing.

Spatial Multiplexing

The MIMO system model used in this chapter is a straightforward extension of the single-antenna BICM system defined in Section 2.2 on page 11. The channel encoder and QAM mapper remain unaltered. The symbol sequence (2.5) produced by the mapper is demultiplexed into N_s data streams,[7] with $N_\mathrm{s} \leq \min(N_\mathrm{t}, N_\mathrm{r})$. The nth data symbol vector $\tilde{\boldsymbol{x}}_n \in \mathbb{C}^{N_\mathrm{s}}$ is linearly mapped to the vector $\boldsymbol{x}_n = \boldsymbol{P}\boldsymbol{G}\tilde{\boldsymbol{x}}_n \in \mathbb{C}^{N_\mathrm{t}}$ which is transmitted at discrete time n. Here, $\boldsymbol{G}$ is a diagonal matrix containing real non-negative gains, whose purpose is to adjust the transmit powers of the data streams, and $\boldsymbol{P} \in \mathbb{C}^{N_\mathrm{t} \times N_\mathrm{s}}$ is a precoding matrix with $\boldsymbol{P}^\mathsf{H}\boldsymbol{P} = \boldsymbol{I}_{N_\mathrm{s}}$.

How should the transmit powers and the precoding matrix be chosen? If the transmitter has access to instantaneous CSI, it can compute the singular value decomposition

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\mathsf{H} \tag{4.5}$$

of the channel matrix, where $\boldsymbol{U}^\mathsf{H}\boldsymbol{U} = \boldsymbol{V}^\mathsf{H}\boldsymbol{V} = \boldsymbol{I}_{\min(N_\mathrm{t},N_\mathrm{r})}$, and where the diagonal matrix $\boldsymbol{\Sigma}$ contains the singular values of $\boldsymbol{H}$. If the transmitter now precodes the symbol vector $\tilde{\boldsymbol{x}}$ with $\boldsymbol{V}$, and the receiver filters the received signal $\boldsymbol{y}$ with $\boldsymbol{U}^\mathsf{H}$, we obtain

$$\begin{aligned} \tilde{\boldsymbol{y}} &\triangleq \boldsymbol{U}^\mathsf{H}\boldsymbol{y} \\ &= \boldsymbol{U}^\mathsf{H}(\boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}) \\ &= \boldsymbol{\Sigma}\boldsymbol{G}\tilde{\boldsymbol{x}} + \tilde{\boldsymbol{w}} \end{aligned} \tag{4.6}$$

where the effective noise vector $\tilde{\boldsymbol{w}} \triangleq \boldsymbol{U}^\mathsf{H}\boldsymbol{w} \sim \mathcal{CN}(\boldsymbol{0}_{N_\mathrm{r}}, N_0\boldsymbol{I}_{N_\mathrm{r}})$ has the same distribution as $\boldsymbol{w}$. Since both $\boldsymbol{\Sigma}\boldsymbol{G}$ and $\mathrm{cov}[\tilde{\boldsymbol{w}}]$ are diagonal, (4.6) describes a set of $\min(N_\mathrm{t}, N_\mathrm{r})$ mutually independent scalar channels. It has been shown by Telatar [128] that this scheme, known as *eigen-beamforming*, is capacity-achieving if the data symbols are iid circular Gaussian, with transmit powers chosen by water-filling.

But since the data symbols in (4.6) can be detected separately, this scenario is not relevant for the following discussion, which will be about the *joint* detection of MIMO

[7] This transmitter structure, where a single codeword is transmitted over all antennas, is called *vertical coding*. An alternative is *horizontal coding*, where the spatial data streams are produced by separate encoders. All simulation results that will be presented later have been obtained with vertical coding, but the proposed MIMO detectors can of course also be employed in a horizontally coded system.

symbol vectors. As motivated above, we will thus focus on the scenario with CSI at the receiver, but not at the transmitter. In this case, the capacity-achieving input covariance is a scaled identity matrix [128, Theorem 1], which implies $N_\mathrm{s} = N_\mathrm{t}$, $\boldsymbol{P} = \boldsymbol{I}$, and $\boldsymbol{G} = \alpha\boldsymbol{I}$ for some constant $\alpha > 0$. Without loss of generality, we can assume $\alpha = 1$ and thus $\tilde{\boldsymbol{x}} = \boldsymbol{x}$ by incorporating the transmit power into the data symbols. With the symbol energy E_s as defined in (2.11) on page 14, the total transmit power becomes

$$P = \frac{N_\mathrm{t} E_\mathrm{s}}{T}. \tag{4.7}$$

One could argue that the constraint $N_\mathrm{s} = N_\mathrm{t}$ is overly restrictive, because the channel might not support N_t data streams. This is clearly the case if $N_\mathrm{t} > N_\mathrm{r}$, but also for ill-conditioned channel matrices, or in low-SNR scenarios where water-filling would assign zero power to some modes. However, modern wireless standards allow for the feedback of low-rate channel quality information, like the maximum number of spatial streams or the highest supported modulation and coding scheme. The 3GPP LTE system, for instance, defines a *rank indication*, a *precoder matrix indication*, and a *channel-quality indication*, which are sent from the terminals back to the base stations [24, Section 13.2.5]. It is therefore a realistic assumption that the number of data streams is not excessively large, even in a fast fading scenario without full CSI at the transmitter. In the interest of a simple notation, we will not distinguish between N_s and N_t, and stick with the simple channel model (4.3) where the entries of $\boldsymbol{x}_n$ are QAM symbols without any precoding. This simplification is without loss of generality, because we can always interpret $\boldsymbol{H}_n$ as the $N_\mathrm{r} \times N_\mathrm{s}$ product of the physical channel and the precoding matrix.

To summarize, the following derivations will be based on the channel equation

$$\boldsymbol{y}_n = \boldsymbol{H}_n \boldsymbol{x}_n + \boldsymbol{w}_n, \qquad 0 \le n < N \tag{4.8}$$

where

$$\operatorname{cov}[\boldsymbol{x}_n] = E_\mathrm{s} \boldsymbol{I}_{N_\mathrm{t}} \qquad \text{and} \qquad \operatorname{cov}[\boldsymbol{w}_n] = N_0 \boldsymbol{I}_{N_\mathrm{r}}. \tag{4.9}$$

With the transmit power as given in (4.7), a noise power after matched filtering of N_0/T at each receive antenna, and assuming the normalization $\sigma_h^2 = 1$ as mentioned in Section 4.1.1, the average signal-to-noise ratio per receive antenna is

$$\mathsf{SNR} = \frac{\sum_{i=0}^{N_\mathrm{t}-1} \mathbb{E}[|h_{n,j,i}\, x_{n,i}|^2]}{\mathbb{E}[|w_{n,j}|^2]} = \frac{N_\mathrm{t} E_\mathrm{s}}{N_0}. \tag{4.10}$$

A block diagram of our MIMO system model, including the BICM-ID receiver that will be derived next, is shown in Figure 4.1.

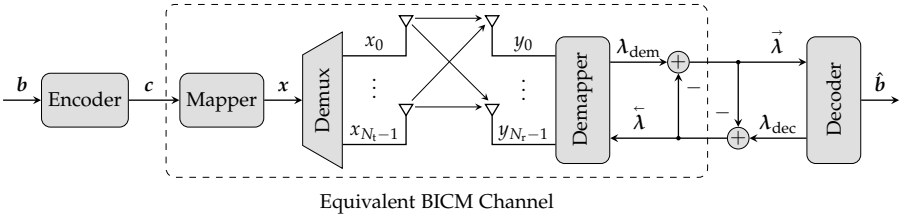


Figure 4.1: Overview of a MIMO transmission system with spatial multiplexing and an iterative BICM-ID receiver. The dashed box marks the equivalent BICM channel that will be discussed in Section 4.1.4.

4.1.3 The Outer MIMO Receiver

As a starting point for the following discussion, we now derive the exact BICM-ID receiver[8] for MIMO systems with spatial multiplexing, which is a very straightforward extension of the single-antenna BICM-ID receiver from Section 2.3. Due to its more compact graphical representation, we will use expectation propagation with fully factorized beliefs as presented in Section 2.3.2, but the more common derivation from loopy belief propagation is mathematically equivalent.

As usual, we start with the joint posterior distribution

$$p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x} \,|\, \boldsymbol{y}) \propto p(\boldsymbol{b})\, p(\boldsymbol{c} \,|\, \boldsymbol{b}) \prod_{n=0}^{N-1} p(\boldsymbol{x}_n \,|\, \boldsymbol{c}_n)\, p(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n) \tag{4.11}$$

where $\boldsymbol{c}_n$ contains the code bits that are mapped to the data vector $\boldsymbol{x}_n$. For later reference, we also define the sub-vectors $\boldsymbol{c}_{n,i}$ which contain only those bits that are associated with the symbol $x_{n,i}$. Note that we assume without loss of generality that all N_t data streams use the same M-ary modulation alphabet, so all $\boldsymbol{c}_{n,i}$ have the same length $K = \log_2(M)$.

Figure 4.2 shows a comb-structured graphical model of (4.11). The left-most nodes are of degree one, so the messages from the matched filter (represented by the nodes $p(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n)$) to the MIMO demapper (represented by the nodes $p(\boldsymbol{x}_n \,|\, \boldsymbol{c}_n)$) are simply given by the likelihood functions

$$\begin{aligned} m_{\mathrm{mf}\to\mathrm{dem}}(\boldsymbol{x}_n) &\propto \exp\left(-\frac{\|\boldsymbol{y}_n - \boldsymbol{H}_n \boldsymbol{x}_n\|^2}{N_0}\right) \\ &\propto \exp\left(-\boldsymbol{x}_n^\mathsf{H} \boldsymbol{H}_n^\mathsf{H} N_0^{-1} \boldsymbol{H}_n \boldsymbol{x}_n + 2\,\mathrm{Re}(\boldsymbol{x}_n^\mathsf{H} \boldsymbol{H}_n^\mathsf{H} N_0^{-1} \boldsymbol{y}_n)\right) \end{aligned} \tag{4.12}$$

[8] Note that this is *not* the globally optimal recevier which minimizes the bit or word error rate. The approximation made in BICM-ID receivers, the bitwise information exchange between detector and decoder, is still in effect. The word "exact" only refers to the MIMO detector, and should be contrasted to the suboptimal detectors with lower complexity that we will study in the following.

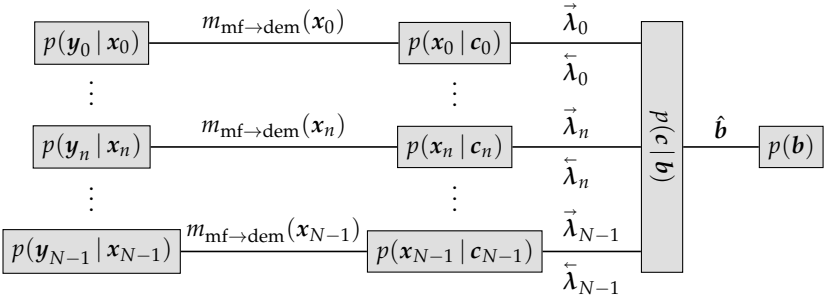


Figure 4.2: Forney-style factor graph of the joint posterior (4.11)

which are (possibly degenerate) circular Gaussian distributions with scaled mean vectors $\boldsymbol{\nu}_n = \boldsymbol{H}_n^{\mathsf{H}} N_0^{-1} \boldsymbol{y}_n$ and precision (inverse covariance) matrices $\boldsymbol{\Gamma}_n = \boldsymbol{H}_n^{\mathsf{H}} N_0^{-1} \boldsymbol{H}_n$. This is the MIMO-generalization of (2.16) from page 15, which was guaranteed to be normalizable as long as $g_n \neq 0$.

Note (a) that the messages $m_{\mathrm{mf}\to\mathrm{dem}}(\boldsymbol{x}_n)$ have been computed via inclusive DM, and (b) that the beliefs $q(\boldsymbol{x}_n)$ of the data symbols are unconstrained. The messages are thus essentially obtained via belief propagation, and since the $p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)$ are leaf nodes in the graphical model in Figure 4.2, they remain static during the inference algorithm. Only the messages between demapper and decoder, i. e., $m_{\mathrm{dem}\to\mathrm{dec}}(\boldsymbol{c}_n)$ and $m_{\mathrm{dec}\to\mathrm{dem}}(\boldsymbol{c}_n)$, are iteratively updated. This is the key difference between the exact MIMO detector from this section and the low-complexity algorithms that will be derived later from the generalized divergence minimization perspective: the LMMSE-based MIMO detector from Section 4.3 constrains the beliefs $q(\boldsymbol{x}_n)$ to Gaussians, hence turning BP into EP, and the MRC-based detector from Section 4.4 computes $m_{\mathrm{mf}\to\mathrm{dem}}(\boldsymbol{x}_n)$ via exclusive DM, yielding a VMP-type algorithm. In both cases, the $m_{\mathrm{mf}\to\mathrm{dem}}(\boldsymbol{x}_n)$ are not static anymore; they are iteratively refined, too. This gives rise to a second loop between the demapper nodes $p(\boldsymbol{x}_n \mid \boldsymbol{c}_n)$ and the MIMO front-end $p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)$, in addition to the standard BICM loop between demapper and decoder, which will be derived next.

Following the central concept behind BICM, we assume fully factorized beliefs

$$q(\boldsymbol{c}_n) = \prod_{i=0}^{N_{\mathrm{t}}-1} \prod_{k=0}^{K-1} q(c_{n,i,k}) \tag{4.13}$$

i. e., we project $q(\boldsymbol{c}_n)$ into the set $\mathcal{B}^{N_{\mathrm{t}}K}$ of bitwise factorized distributions over $\mathbb{F}^{N_{\mathrm{t}}K}$. In the following, the vector $\boldsymbol{\lambda}_n \in \mathbb{R}^{N_{\mathrm{t}}K}$ contains the L-values associated with the belief $q(\boldsymbol{c}_n)$, and $\vec{\boldsymbol{\lambda}}_n$ and $\overleftarrow{\boldsymbol{\lambda}}_n$ denote its intrinsic and extrinsic component, respectively.

With the extrinsic feedback from the decoder to the demapper

$$m_{\text{dec}\to\text{dem}}(\boldsymbol{c}_n) \propto \exp\left(\boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right) = \prod_{i=0}^{N_{\mathrm{t}}-1}\prod_{k=0}^{K-1}\exp\left(c_{n,i,k}\overleftarrow{\lambda}_{n,i,k}\right) \tag{4.14}$$

the messages from the demapper to the decoder become

$$\begin{aligned} &m_{\text{dem}\to\text{dec}}(\boldsymbol{c}_n) \\ &\quad\propto \frac{1}{m_{\text{dec}\to\text{dem}}(\boldsymbol{c}_n)}\,\text{proj}_{\mathcal{B}^{N_{\mathrm{t}}K}}\left[m_{\text{dec}\to\text{dem}}(\boldsymbol{c}_n)\sum_{\boldsymbol{x}_n} p(\boldsymbol{x}_n \mid \boldsymbol{c}_n)\, m_{\text{mf}\to\text{dem}}(\boldsymbol{x}_n)\right] \\ &\quad\propto \prod_{i=0}^{N_{\mathrm{t}}-1}\prod_{k=0}^{K-1}\exp\left(-c_{n,i,k}\overleftarrow{\lambda}_{n,i,k}\right)\sum_{\sim c_{n,i,k}}\exp\left(-\frac{\|\boldsymbol{y}_n-\boldsymbol{H}_n\mathcal{M}(\boldsymbol{c}_n)\|^2}{N_0}+\boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right) \end{aligned} \tag{4.15}$$

and the intrinsic L-values $\vec{\lambda}_{n,i,k}$ that are sent forward from the demapper to the decoder can be computed as

$$\vec{\lambda}_{n,i,k} = \log\frac{\sum_{\boldsymbol{x}_n\in\mathcal{X}_{i,k}^1}\exp\left(-\frac{\|\boldsymbol{y}_n-\boldsymbol{H}_n\boldsymbol{x}_n\|^2}{N_0}+\boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right)}{\sum_{\boldsymbol{x}_n\in\mathcal{X}_{i,k}^0}\exp\left(-\frac{\|\boldsymbol{y}_n-\boldsymbol{H}_n\boldsymbol{x}_n\|^2}{N_0}+\boldsymbol{c}_n^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_n\right)} - \overleftarrow{\lambda}_{n,i,k}. \tag{4.16}$$

Here, $\mathcal{X}_{i,k}^{\beta}$ is the set of N_{t}-dimensional symbol vectors, where the bit label of the component x_i contains a $\beta\in\mathbb{F}$ at position k. The two equations above are straightforward generalizations of (2.31) and (2.32). As before in (2.32), $\boldsymbol{c}_n$ in (4.16) implicitly contains the bit label of the symbol vector $\boldsymbol{x}_n$.

The soft decoder, i. e., the computation of $\overleftarrow{\boldsymbol{\lambda}}$ and $\hat{\boldsymbol{b}}$ from $\vec{\boldsymbol{\lambda}}$, is the same as before and will not be discussed further.

The Max-Log Approximation and the Sphere Detector

The two sums in (4.16) contain $2^{N_{\mathrm{t}}K-1}$ terms each, so the exact MIMO demapper has a complexity that is exponential in the number of transmit streams. Even for a moderate system configuration of, say, $N_{\mathrm{t}}=4$ and a 16-QAM modulation, the squared norm in (4.16) must be evaluated 2^{16} times, which cannot be done online for practically relevant symbol durations T. For this reason, the past years have witnessed a large amount of research on low-complexity, suboptimal MIMO detection schemes.

A common starting point that we briefly mentioned on page 19 is the max-log approximation, which relies on the fact that the sums in (4.16) are often dominated by just a few terms. Replacing the sums with their maximum terms yields the approximation

$$\vec{\lambda}_{n,i,k} \approx \min_{\boldsymbol{x}_n \in \mathcal{X}_{i,k}^0} \left(\frac{\|\boldsymbol{y}_n - \boldsymbol{H}_n \boldsymbol{x}_n\|^2}{N_0} - \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right) - \min_{\boldsymbol{x}_n \in \mathcal{X}_{i,k}^1} \left(\frac{\|\boldsymbol{y}_n - \boldsymbol{H}_n \boldsymbol{x}_n\|^2}{N_0} - \boldsymbol{c}_n^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}_n \right) - \overleftarrow{\lambda}_{n,i,k}. \tag{4.17}$$

A brute-force implementation of (4.17) would also need to evaluate all $2^{N_t K}$ squared norms. However, it has been recognized that the discrete minimization problem $\min_{\boldsymbol{x} \in \mathcal{X}^{N_t}} \|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x}\|^2$ (ignoring the term $\boldsymbol{c}^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}$ for a moment) can be formulated as the search for the closest point in a lattice [25], which can be solved via branch-and-bound techniques that are known as *sphere detectors* (SD) in the context of MIMO receivers. Also, modified sphere detectors that make use of the prior information $\boldsymbol{c}^{\mathsf{T}} \overleftarrow{\boldsymbol{\lambda}}$ have been developed, so that they can be employed as a building block in BICM-ID receivers [58].

While sphere detectors are much simpler than brute-force receivers which enumerate all possible transmit vectors as in (4.16), their complexity is non-constant (it depends on the realizations of the fading and noise processes) which imposes difficulties for VLSI implementations, and the average complexity is still exponential in the number of spatial data streams [61]. This motivates our interest in MIMO detectors with even lower (and in particular constant) complexity, which will be the topic of the rest of this chapter. We will, however, use the max-log detector as a benchmark in the numerical performance evaluations. When reading those figures, it should be kept in mind that practical implementations of the max-log detector using the SD technique will not reach the optimal max-log performance, since they will require further approximations. An extensive comparison of different practical sphere detectors, however, would justify a thesis on its own, so we will stick to the pure max-log detector in the comparisons.

4.1.4 Information Theoretic Bounds and EXIT Charts

We continue with a brief introduction to a concept that will play an important role in the performance evaluations: the equivalent BICM channel and its associated mutual information. The performance of MIMO detectors is usually evaluated in terms of bit or word error rates. This approach, however, has the drawback that it is dependent on the specific channel code and the applied decoding algorithm. In a recent paper [37], the authors propose to benchmark MIMO detectors based on the mutual information over an equivalent BICM channel. This equivalent channel, marked in Figure 4.1 with a dashed box, consists of the symbol mapper, the physical channel, and the detector, i.e., it is a binary-input continuous-output channel, with code bits c as inputs and L-values $\vec{\lambda}$ as outputs.[9] The mutual information $I(c; \vec{\lambda})$ over this channel depends on the detection algorithm, but not on the channel code.

[9] Note that this BICM channel is not the same as the modulation channel that we discussed in Section 2.1: the latter takes complex data symbols as inputs, while the BICM channel takes bits and outputs bitwise L-values.

Assuming a uniform input distribution, $p(c) = 0.5$ for $c \in \mathbb{F}$, the mutual information over the BICM channel can be calculated as follows:

$$\begin{aligned} I(c;\vec{\lambda}) &\triangleq \mathbb{H}[c] - \mathbb{H}[c \,|\, \vec{\lambda}] \\ &= 1 - \mathbb{E}_{c,\vec{\lambda}}\left[\log_2 \frac{1}{p(c \,|\, \vec{\lambda})}\right] \\ &= 1 - \mathbb{E}_{c,\vec{\lambda}}\left[\log_2 \frac{p(\vec{\lambda} \,|\, 0) + p(\vec{\lambda} \,|\, 1)}{p(\vec{\lambda} \,|\, c)}\right] \end{aligned} \tag{4.18}$$

where $\mathbb{H}[x]$ is the entropy of x, and $\mathbb{H}[x \,|\, y]$ the conditional entropy of x given y. For most cases of practical interest, there is no analytical expression available for (4.18), but the conditional densities and the expectation can be approximated by Monte Carlo methods. The results that are presented in this chapter have been obtained with the algorithm described in [37, Appendix A].

In general, the bitwise mutual information can depend on k, the index of the bit within the label of the data symbol. For example, consider the 4-PAM scheme with symbols $-3, -1, +1, +3$, and associated bit labels $00, 01, 11, 10$. It is intuitively reasonable that the first bit, which determines the sign of x, is better protected than the second bit, and thus $I(c_0;\vec{\lambda}_0) > I(c_1;\vec{\lambda}_1)$. The information that can be transmitted over the MIMO channel per symbol duration is then upper-bounded by the sum of $I(c_k;\vec{\lambda}_k)$ over all bits

$$\begin{aligned} C^{\text{bicm}} &\triangleq N_{\text{t}} \sum_{k=0}^{K-1} I(c_k;\vec{\lambda}_k) \\ &= N_{\text{t}}\, K\, I_E \end{aligned} \tag{4.19}$$

which is called the *BICM capacity*. In (4.19), we have introduced the average mutual information

$$I_E \triangleq \frac{1}{K} \sum_{k=0}^{K-1} I(c_k;\vec{\lambda}_k). \tag{4.20}$$

So far, we have considered non-iterative receivers. In the presence of feedback $\overleftarrow{\lambda}$ from the decoder, there is some *a priori* information about the bits available at the detector, whose average is defined analogously to (4.20) as

$$I_A \triangleq \frac{1}{K} \sum_{k=0}^{K-1} I(c_k;\overleftarrow{\lambda}_k). \tag{4.21}$$

In general, the availability of $\overleftarrow{\lambda}$ helps the detector to compute more reliable $\vec{\lambda}$, so that I_E is a non-decreasing function of I_A, called an *extrinsic information transfer* (EXIT) function. The graph of $I_E(I_A)$ in a $[0,1] \times [0,1]$ diagram is known as an *EXIT chart* [129].

The *area theorem* [5] states that the *CM capacity*, which upper-bounds the data rates that are achievable with BICM-ID receivers, is

$$C^{\mathrm{cm}} \triangleq N_{\mathrm{t}} K \int_0^1 I_E(I_A)\,\mathrm{d}I_A \tag{4.22}$$

where the integral is equal to the area in the EXIT chart below the function plot.

Strictly speaking, the area theorem only holds if the *extrinsic channel* (a hypothetical construct which models the relationship between c_k and $\overleftarrow{\lambda}_k$) is a binary erasure channel. But due to the robustness of EXIT charts, (4.22) is reasonably accurate also for other, more realistic channels [5]. In the numerical results presented in this chapter, we follow the usual convention and model the extrinsic channel as a binary-input AWGN channel [130, Section II.B].

A concise introduction to EXIT charts and the area theorem can also be found in [51, Sections 5.3 and 5.4].

4.1.5 Numerical Results

We conclude this introductory section with some numerical results. Figure 4.3 compares the EXIT characteristics of MIMO systems with 1, 2, and 4 spatial data streams. In all cases, a Gray-mapped 16-QAM modulation and an SNR of 6 dB was used.

For $N_{\mathrm{t}} = N_{\mathrm{r}} = 1$, i. e., a conventional single-antenna system, the EXIT function is almost constant, which explains the well-known fact that BICM-ID hardly yields any gains in Gray-mapped SISO systems. The early research on BICM-ID has thus been accompanied by studies of non-Gray-mapping schemes that are optimized for use in BICM-ID systems (see for example [20]). Those mapping schemes, however, have the drawback that they perform poorly in conjunction with strong channel codes like turbo or LDPC [112]. Thus, plain Gray mappings seem to be the most reasonable choice for SISO systems, which in turn makes single-antenna BICM-ID receivers less attractive.

This changes once we turn to multi-antenna systems. As seen in Figure 4.3, the EXIT characteristics for 2 and 4 data streams are not horizontal anymore, so it is reasonable to assume that iterations between the detector and decoder will yield non-negligible performance gains, even if pragmatic Gray-mapped modulations are used.

To verify this assumptions, we plot in Figure 4.4 the CM and BICM capacities of a 4×4 MIMO system (i. e., $N_{\mathrm{t}} = N_{\mathrm{r}} = 4$). We see that there is a significant gap of around 1 to 3 dB between the two curves, which corresponds to the potential gain that can be achieved with iterative receivers.

As an upper bound, the figure also shows the Shannon capacity that was computed by numerical quadrature of [128, Equation (8)]. It can be seen that, below a certain SNR, the CM capacities are quite close to the Shannon capacity, while they necessarily branch off and converge to $N_{\mathrm{t}} \log_2 M$ for higher SNR. This shows that BICM-ID systems have

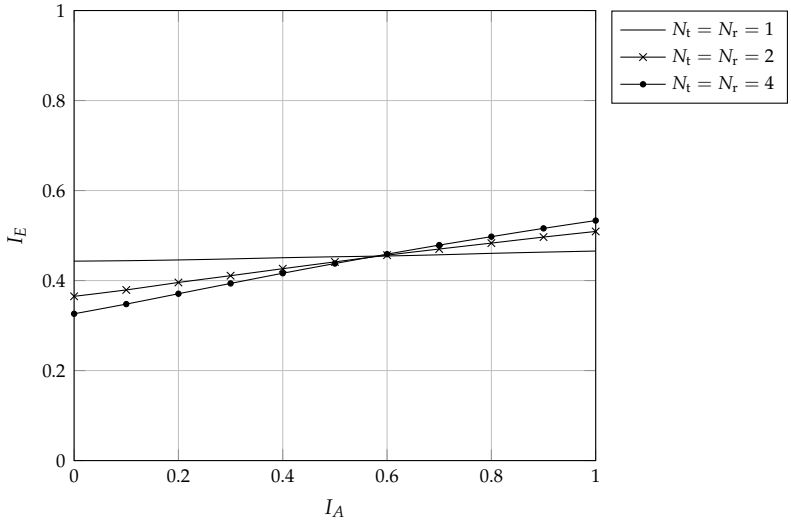


Figure 4.3: EXIT chart, using a Gray-mapped 16-QAM modulation, a symmetrical antenna configuration, and a max-log sphere detector, at SNR = 6 dB

the potential to operate close to capacity, provided that not too much information is lost by the suboptimal, low-complexity MIMO detectors.

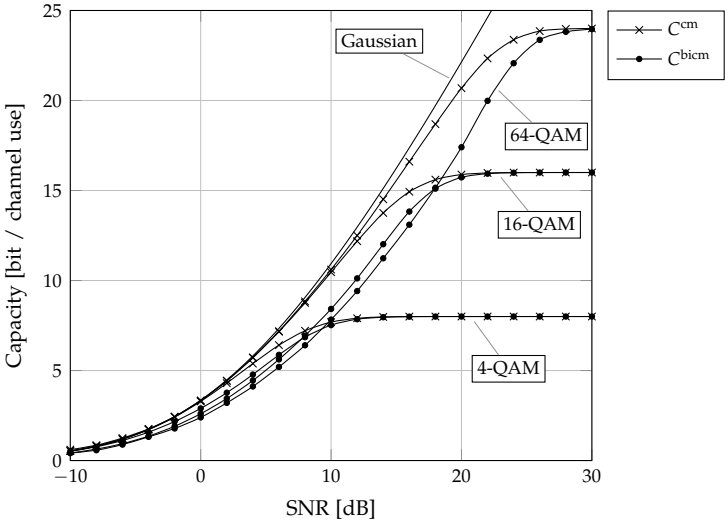


Figure 4.4: CM and BICM capacity of a 4×4 MIMO system, with Gray-mapped 4-, 16-, and 64-QAM modulation schemes. Also shown is the Shannon capacity, achieved with a Gaussian input distribution, as an upper bound for C^{cm} and C^{bicm}

4.2 Conventional Low-Complexity MIMO Detectors

In this section, we briefly introduce a few conventional low-complexity MIMO detectors. They are quite heuristic, but they still provide some interesting context for the novel iterative MIMO detectors that we will derive later from the diveregence minimization framework. Besides serving as a performance benchmark for the proposed detectors, we will also encounter open questions regarding the connection between the MIMO detector and the other receiver components, that arise from the *ad hoc* derivations and whose answers will be derived from the systematic approach.

Since the MIMO detectors process each data vector separately, we will drop the time index n in this section, to simplify the notation.

4.2.1 ZF Detection

One of the simplest MIMO detection methods is the *zero forcing* (ZF) detector (e. g. [18]). It proceeds in two steps. First, the spatial interference between the data streams is removed by filtering the received signal with the (pseudo)inverse of the channel matrix. Then, the N_{t} interference-free signals are processed by conventional scalar demappers.

Applying the ZF filter $\boldsymbol{H}^{+} \triangleq (\boldsymbol{H}^{\mathsf{H}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathsf{H}}$ to the received signal yields

$$\begin{aligned}\tilde{\boldsymbol{x}}_{\mathrm{ZF}} &\triangleq \boldsymbol{H}^{+}\boldsymbol{y} \\ &= (\boldsymbol{H}^{\mathsf{H}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathsf{H}}(\boldsymbol{H}\boldsymbol{x}+\boldsymbol{w}) \\ &= \boldsymbol{x} + \tilde{\boldsymbol{w}}_{\mathrm{ZF}}\end{aligned} \tag{4.23}$$

with an effective noise vector $\tilde{\boldsymbol{w}}_{\mathrm{ZF}} = (\boldsymbol{H}^{\mathsf{H}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathsf{H}}\boldsymbol{w}$ which is circular Gaussian with $\mathrm{cov}\,[\tilde{\boldsymbol{w}}_{\mathrm{ZF}}] = N_0(\boldsymbol{H}^{\mathsf{H}}\boldsymbol{H})^{-1}$.

Note that the ZF filter only exists for $N_{\mathrm{r}} \geq N_{\mathrm{t}}$, but this is not a relevant limitation in practice. As argued earlier on page 126, $\boldsymbol{H}$ should in general be understood as the product of the $N_{\mathrm{r}} \times N_{\mathrm{t}}$ physical channel and the $N_{\mathrm{t}} \times N_{\mathrm{s}}$ precoder; and the system can always choose to transmit $N_{\mathrm{s}} \leq N_{\mathrm{r}}$ streams, even if $N_{\mathrm{t}} > N_{\mathrm{r}}$.

Equation (4.23) can be broken down into the scalar equations

$$\tilde{x}_{\mathrm{ZF},i} = x_i + \tilde{w}_{\mathrm{ZF},i} \qquad 0 \leq i < N_{\mathrm{t}} \tag{4.24}$$

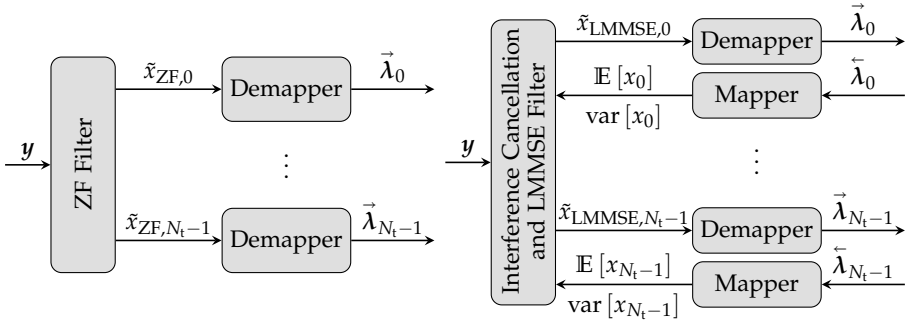
each of which describes an AWGN channel with $\mathrm{var}\left[\tilde{w}_{\mathrm{ZF},i}\right] = N_0\boldsymbol{e}_i^{\mathsf{H}}(\boldsymbol{H}^{\mathsf{H}}\boldsymbol{H})^{-1}\boldsymbol{e}_i$. Here, $\boldsymbol{e}_i$ is the ith unit vector which contains a one in the ith position, and zeros otherwise. The outputs of those virtual AWGN channels are fed to a bank of N_{t} scalar demappers, which compute the L-values that are passed to the decoder. It should be noted that the suboptimality of the ZF detector is not caused by the ZF filter, which is information-preserving (as long as the pseudoinverse of $\boldsymbol{H}$ exists). Rather, information is lost because the scalar demappers work independently of each other, and thus do not consider the correlation between the filtered noise samples; i. e., they neglect the off-diagonal elements of $\mathrm{cov}\,[\tilde{\boldsymbol{w}}_{\mathrm{ZF}}]$.

An overview of the ZF detector is shown in Figure 4.5a. It is a very simple and rather poorly performing algorithm, but we will see in the further development that the more advanced detectors exhibit the same general structure: a filter followed by a bank of scalar demappers. An important difference, which also becomes apparent from Figure 4.5a, is that the ZF detector works in a purely feed-forward manner; it is not able to exploit feedback from the channel decoder. The detectors that we will discuss in the following use the non-uniform prior information about the data symbols for some kind of *interference cancellation* (IC). The ZF detector, however, removes the whole interference anyway, so pre-subtracting an estimated interference vector before applying the ZF filter would not have any effect.

4.2.2 MRC Detection

Assume for a moment that we were only interested in receiving the ith spatial stream. We would then decompose the received signal as

$$\boldsymbol{y} = \boldsymbol{h}_i x_i + \sum_{i' \neq i} \boldsymbol{h}_{i'} x_{i'} + \boldsymbol{w}$$



(a) ZF Detector **(b)** LMMSE Detector

Figure 4.5: Overview of the ZF and LMMSE detectors

$$= \boldsymbol{h}_i x_i + \boldsymbol{H}_{\setminus i}\boldsymbol{x}_{\setminus i} + \boldsymbol{w} \tag{4.25}$$

where $\boldsymbol{h}_i$ is the ith column of $\boldsymbol{H}$, $\boldsymbol{H}_{\setminus i}$ is obtained from $\boldsymbol{H}$ by removing the ith column, and $\boldsymbol{x}_{\setminus i}$ is obtained from $\boldsymbol{x}$ by removing the ith symbol x_i. The second term in (4.25) is the spatial interference[10] that acts on the ith data stream. For later reference, we define the interference-plus-noise term

$$\begin{aligned}\boldsymbol{v}_i &\triangleq \sum_{i' \neq i} \boldsymbol{h}_{i'} x_{i'} + \boldsymbol{w} \\ &= \boldsymbol{H}_{\setminus i}\boldsymbol{x}_{\setminus i} + \boldsymbol{w}\end{aligned} \tag{4.26}$$

and write (4.25) more compactly as

$$\boldsymbol{y} = \boldsymbol{h}_i x_i + \boldsymbol{v}_i. \tag{4.27}$$

As we have seen above, the ZF filter is designed to maximize the signal-to-interference power ratio (SIR) without considering the noise vector (where "signal" only refers to the desired signal component $\boldsymbol{h}_i x_i$, not the interference). The *maximum ratio combining* (MRC) filter represents the other extreme: it maximizes the signal-to-noise ratio without considering the interference term. It is well known that the SNR-maximizing filter for a SIMO channel $\boldsymbol{y} = \boldsymbol{h}x + \boldsymbol{w}$ is (proportional to) the conjugate transpose of the channel vector [59]. Collecting the MRC estimates of all N_t streams in one vector thus yields

$$\tilde{\boldsymbol{x}}_{\mathrm{MRC}} \triangleq \boldsymbol{H}^{\mathsf{H}}\boldsymbol{y} \tag{4.28}$$

[10]Since we are considering a very simple channel model without any other sources of interference (like inter-symbol-, inter-carrier-, or multi-user-interference, we will simply speak of *the* interference in the following.

where each component can be written as the output of an effective AWGN channel

$$\tilde{x}_{\mathrm{MRC},i} = \|\boldsymbol{h}_i\|^2 x_i + \tilde{v}_{\mathrm{MRC},i} \qquad 0 \leq i < N_\mathrm{t} \tag{4.29}$$

with gain $\|\boldsymbol{h}_i\|^2$ and filtered interference-plus-noise term

$$\tilde{v}_{\mathrm{MRC},i} = \boldsymbol{h}_i^\mathsf{H} \boldsymbol{v}_i \tag{4.30}$$

$$= \boldsymbol{h}_i^\mathsf{H} \boldsymbol{H}_{\setminus i} \boldsymbol{x}_{\setminus i} + \boldsymbol{h}_i^\mathsf{H} \boldsymbol{w}. \tag{4.31}$$

An advantage of MRC over ZF, and in particular over the LMMSE filter from the next section, is the extremely low computational complexity of (4.28), since it does not involve any matrix inversions. Its obvious drawback is that it does not even attempt to suppress the interference, so it can only be expected to work well if the filtered noise term $\boldsymbol{h}_i^\mathsf{H} \boldsymbol{w}$ is much stronger than the filtered interference term $\boldsymbol{h}_i^\mathsf{H} \boldsymbol{H}_{\setminus i} \boldsymbol{x}_{\setminus i}$. This condition is trivially fulfilled for rank-1 transmissions, since they do not experience spatial interference at all. Hence, as briefly mentioned at the beginning of this chapter, the MRC receiver is commonly encountered in the literature on beamforming.

Another scenario where MRC could turn out to be an interesting option is the uplink of so-called *massive MIMO* systems which have gained a lot of interest recently (see e. g. [71]). They are characterized by huge antenna arrays at the base station (research papers envision several hundred elements), such that $N_\mathrm{r} \gg N_\mathrm{t}$. The channel matrix then becomes very tall, and its columns are almost orthogonal with high probability.

Finally, another possible reason why the interference could be small compared to the noise is the presence of feedback in iterative receivers. Using the non-uniform prior information about the data symbols, it becomes possible to compute an estimate of the spatial interference, and to subtract it from the received signal. However, based on our development so far, we would have to resort to heuristics, as it is not yet apparent how such an *interference cancellation* (IC) step would have to be implemented correctly. We will return to this topic in Section 4.4.

4.2.3 LMMSE Detection

The *linear minimum mean-squared error* (LMMSE) filter strikes a balance between ZF and MRC by minimizing the mean-squared Euclidean distance between $\boldsymbol{x}$ and its estimate $\tilde{\boldsymbol{x}}_{\mathrm{LMMSE}}$, thus taking both the noise and the interference into account. Since it will play an important role in the following section, we will take the time and derive the LMMSE detector from first principles.

The LMMSE filter depends on the first two moments of $\boldsymbol{x}$. In non-iterative receivers, these would be $\mathbb{E}[\boldsymbol{x}] = \boldsymbol{0}_{N_\mathrm{t}}$ and $\mathrm{cov}[\boldsymbol{x}] = E_\mathrm{s} \boldsymbol{I}_{N_\mathrm{t}}$, but since we are interested in iterative algorithms, where the detector has some prior information about $\boldsymbol{x}$ after the first iteration,

we will not make this assumption to keep the derivation general. In order to account for the possibly non-zero mean, we include a constant term $\boldsymbol{a}$ in the estimator, and write

$$\tilde{\boldsymbol{x}}_{\text{LMMSE}} \triangleq \boldsymbol{Q}\boldsymbol{y} + \boldsymbol{a}. \tag{4.32}$$

Strictly speaking, (4.32) is an *affine* rather than a linear equation, but we will stick to the more common terminology. The two parameters $\boldsymbol{Q}$ and $\boldsymbol{a}$ are now chosen to minimize the mean-squared estimation error:

$$\begin{aligned}(\boldsymbol{Q}, \boldsymbol{a}) &= \arg\min_{\boldsymbol{Q},\boldsymbol{a}} \mathbb{E}\left[\|\boldsymbol{Q}(\boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}) + \boldsymbol{a} - \boldsymbol{x}\|^2\right] \\ &= \arg\min_{\boldsymbol{Q},\boldsymbol{a}} \mathbb{E}\left[\|(\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\boldsymbol{x} + \boldsymbol{Q}\boldsymbol{w} + \boldsymbol{a}\|^2\right].\end{aligned} \tag{4.33}$$

Taking the derivative with respect to $\boldsymbol{a}^{\mathsf{H}}$, and setting it to zero, yields

$$\begin{aligned}\boldsymbol{0} &\stackrel{!}{=} \frac{\partial}{\partial \boldsymbol{a}^{\mathsf{H}}} \mathbb{E}\left[\|(\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\boldsymbol{x} + \boldsymbol{Q}\boldsymbol{w} + \boldsymbol{a}\|^2\right] \\ &\stackrel{(a)}{=} \mathbb{E}\left[(\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\boldsymbol{x} + \boldsymbol{Q}\boldsymbol{w} + \boldsymbol{a}\right] \\ &= (\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\mathbb{E}\left[\boldsymbol{x}\right] + \boldsymbol{a}\end{aligned} \tag{4.34}$$

and thus

$$\boldsymbol{a} = -(\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\mathbb{E}\left[\boldsymbol{x}\right]. \tag{4.35}$$

In (a) above, we have swapped the derivative and the expectation operator.

We proceed by inserting (4.35) into (4.33) and obtain

$$\begin{aligned}\boldsymbol{Q} &= \arg\min_{\boldsymbol{Q}} \mathbb{E}\left[\|(\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})(\boldsymbol{x} - \mathbb{E}\left[\boldsymbol{x}\right]) + \boldsymbol{Q}\boldsymbol{w}\|^2\right] \\ &= \arg\min_{\boldsymbol{Q}} \operatorname{tr}\left((\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\operatorname{cov}\left[\boldsymbol{x}\right](\boldsymbol{H}^{\mathsf{H}}\boldsymbol{Q}^{\mathsf{H}} - \boldsymbol{I}) + \boldsymbol{Q}N_0\boldsymbol{Q}^{\mathsf{H}}\right).\end{aligned} \tag{4.36}$$

Again, we find the optimum by setting the derivative to zero

$$\begin{aligned}\boldsymbol{0} &\stackrel{!}{=} \frac{\partial}{\partial \boldsymbol{Q}^{\mathsf{H}}} \operatorname{tr}\left((\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\operatorname{cov}\left[\boldsymbol{x}\right](\boldsymbol{H}^{\mathsf{H}}\boldsymbol{Q}^{\mathsf{H}} - \boldsymbol{I}) + \boldsymbol{Q}N_0\boldsymbol{Q}^{\mathsf{H}}\right) \\ &= (\boldsymbol{Q}\boldsymbol{H} - \boldsymbol{I})\operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}} + \boldsymbol{Q}N_0 \\ &= \boldsymbol{Q}\left(\boldsymbol{H}\operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}} + N_0\boldsymbol{I}\right) - \operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}}\end{aligned} \tag{4.37}$$

which yields the result

$$\boldsymbol{Q} = \operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}}\left(\boldsymbol{H}\operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}} + N_0\boldsymbol{I}\right)^{-1}. \tag{4.38}$$

Substituting both (4.35) and (4.38) into (4.32), we finally arrive at the LMMSE estimate

$$\begin{aligned}\tilde{\boldsymbol{x}}_{\text{LMMSE}} &= \boldsymbol{Q}\left(\boldsymbol{y} - \boldsymbol{H}\mathbb{E}\left[\boldsymbol{x}\right]\right) + \mathbb{E}\left[\boldsymbol{x}\right] \\ &= \operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}}\left(\boldsymbol{H}\operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}} + N_0\boldsymbol{I}\right)^{-1}\left(\boldsymbol{y} - \boldsymbol{H}\mathbb{E}\left[\boldsymbol{x}\right]\right) + \mathbb{E}\left[\boldsymbol{x}\right] \\ &= \boldsymbol{C}_{\boldsymbol{x}}\boldsymbol{H}^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\left(\boldsymbol{y} - \boldsymbol{H}\mathbb{E}\left[\boldsymbol{x}\right]\right) + \mathbb{E}\left[\boldsymbol{x}\right] \end{aligned} \tag{4.39}$$

where we have introduced the shorthands $\boldsymbol{C}_{\boldsymbol{x}} = \operatorname{cov}\left[\boldsymbol{x}\right]$ and

$$\boldsymbol{C}_{\boldsymbol{y}} = \boldsymbol{H}\operatorname{cov}\left[\boldsymbol{x}\right]\boldsymbol{H}^{\mathsf{H}} + N_0\boldsymbol{I}. \tag{4.40}$$

Replacing $\boldsymbol{y}$ with $\boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}$, the elements of $\tilde{\boldsymbol{x}}_{\text{LMMSE}}$ can be written as

$$\begin{aligned}\tilde{x}_{\text{LMMSE},i} &= \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\left(\boldsymbol{H}\boldsymbol{x} + \boldsymbol{w} - \boldsymbol{H}\mathbb{E}\left[\boldsymbol{x}\right]\right) + \mathbb{E}\left[x_i\right] \\ &= \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\left(\boldsymbol{h}_i\left(x_i - \mathbb{E}\left[x_i\right]\right) + \boldsymbol{H}_{\setminus i}\left(\boldsymbol{x}_{\setminus i} - \mathbb{E}[\boldsymbol{x}_{\setminus i}]\right) + \boldsymbol{w}\right) + \mathbb{E}\left[x_i\right] \\ &= \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\boldsymbol{h}_i\, x_i + \left(1 - \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\boldsymbol{h}_i\right)\mathbb{E}\left[x_i\right] + \tilde{v}_{\text{LMMSE},i} \\ &= \tilde{g}_i\, x_i + \left(1 - \tilde{g}_i\right)\mathbb{E}\left[x_i\right] + \tilde{v}_{\text{LMMSE},i} \end{aligned} \tag{4.41}$$

where we have defined the effective channel gain

$$\tilde{g}_i \triangleq \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\boldsymbol{h}_i \tag{4.42}$$

and the filtered interference-plus-noise component

$$\tilde{v}_{\text{LMMSE},i} \triangleq \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\left(\boldsymbol{H}_{\setminus i}\left(\boldsymbol{x}_{\setminus i} - \mathbb{E}[\boldsymbol{x}_{\setminus i}]\right) + \boldsymbol{w}\right). \tag{4.43}$$

The covariance matrix of the vector in parentheses is $\boldsymbol{C}_{\boldsymbol{y}} - \boldsymbol{h}_i \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}$, so the variance of the interference-plus-noise term can be written as

$$\begin{aligned}\operatorname{var}\left[\tilde{v}_{\text{LMMSE},i}\right] &= \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\left(\boldsymbol{C}_{\boldsymbol{y}} - \boldsymbol{h}_i \operatorname{var}\left[x_i\right]\boldsymbol{h}_i^{\mathsf{H}}\right)\boldsymbol{C}_{\boldsymbol{y}}^{-1}\boldsymbol{h}_i \operatorname{var}\left[x_i\right] \\ &= \operatorname{var}\left[x_i\right]\tilde{g}_i\left(1 - \tilde{g}_i\right). \end{aligned} \tag{4.44}$$

Biased vs Unbiased LMMSE Detection The LMMSE estimate $\tilde{x}_{\text{LMMSE},i}$ shown in (4.41) is biased due to the presence of the effective channel gain $\tilde{g}_i < 1$. It is sometimes proposed in the literature (see e. g. [23]) to use an unbiased LMMSE estimator instead, which would be obtained by multiplying both sides of (4.41) with $\tilde{g}_i^{-1}$.

This modification is indeed relevant for hard-output MIMO detectors, which would slice $\tilde{x}_{\text{LMMSE},i}$ to the closest valid constellation point. This is intuitively reasonable because, for example, a very small $\tilde{g}_i$ would cause the detector to only decide on the constellation points that are closest to the origin of the complex plane, while never picking the outer constellation points.

However, the soft-output detectors that we are studying are not affected at all by a multiplication with $\tilde{g}_i^{-1}$ (with any non-zero constant, for that matter). To see this, consider the canonical AWGN channel model $y = gx + w$ with $\mathrm{var}[w] = N_0$, where the computation of the L-values is based on the ratio $\frac{|y-gx|^2}{N_0}$. The equivalent "unbiased" channel model $\frac{y}{g} = x + \frac{w}{g}$ with $\mathrm{var}\left[\frac{w}{g}\right] = \frac{N_0}{g^2}$ would accordingly lead to the ratio $\frac{|y/g-x|^2}{N_0/g^2} = \frac{g^2}{N_0}\left|x - \frac{y}{g}\right|^2$, which happens to be the formulation that we used in (2.23), but which is obviously the same as $\frac{|y-gx|^2}{N_0}$ above. In fact, it is a simple consequence of the data processing inequality that multiplying an observation by some constant cannot improve the final result, as long as the further postprocessing is lossless. We will therefore not be concerned with the issue of biased vs unbiased LMMSE detection.

Extrinsic vs Posterior Feedback In the derivation above, we have assumed that the MIMO detector knows the first two moments $\mathbb{E}[\boldsymbol{x}]$ and $\mathrm{cov}[\boldsymbol{x}]$ of the data symbol vector, which are presumably calculated based on feedback from the channel decoder (or are simply set to $\mathbf{0}$ and $E_s\boldsymbol{I}$, respectively, in the first iteration or in non-iterative receivers). But despite their importance, we have not yet given a precise equation for calculating the two moments. However, this is not due to an oversight, but because of a fundamental issue of the derivation that is pervasive in the literature, and that we have followed so far. The problem is that the conventional LMMSE detector is based on the simplistic system model $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w}$, which only includes the transmission of a single data vector, and completely ignores the presence of other components like the channel code. Thus, if we want to embed it into an iterative receiver structure, we need to connect it to the rest of the system in an *ad hoc* manner.

The approach that is taken in virtually every paper on iterative LMMSE-based MIMO detection is to start with the standard BICM-ID receiver as developed in Section 4.1.3 and as shown in Figure 4.1 on page 127, and to replace the exact, but computationally infeasible detector with the suboptimal LMMSE detector, leaving the rest of the system untouched. In particular, this implies that the LMMSE detector receives feedback in terms of extrinsic L-values $\overleftarrow{\boldsymbol{\lambda}}$. Then, faced with the problem of converting these into the two moments $\mathbb{E}[\boldsymbol{x}]$ and $\mathrm{cov}[\boldsymbol{x}]$, the only obvious way to proceed is to compute them from the auxiliary prior distribution $p(\boldsymbol{x}; \overleftarrow{\boldsymbol{\lambda}}) \propto \exp\left(\boldsymbol{c}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}\right)$, where $\boldsymbol{c}$ is the bit label of $\boldsymbol{x}$.

Already in 2002, however, it was observed by means of numerical simulations that the overall performance improves if the channel decoder passes not just the extrinsic, but the full posterior information back to the MIMO detector [148] (i. e., if one removes the lower subtraction node in Figure 4.1). Since posterior L-values are the sum of the extrinsic and intrinsic L-values, and since the intrinsic L-values have been computed by the MIMO detector itself in the preceding iteration, replacing extrinsic with posterior feedback essentially means that some information is passed back to its own producer.

This is a remarkable contradiction to the famous *turbo principle* [53], which states that only "new" information should be conveyed to other receiver components in order to prevent double counting. Despite the high practical relevance of this phenomenon, the publication [148] has to my best knowledge not been followed by any attempts to substantiate it with a theoretical explanation.

In the following section, we will develop an LMMSE-based MIMO receiver from a holistic system perspective, which not only yields the detector itself, but also its connections to the remaining system components. On the one hand, the proposed algorithm is able to outperform the conventional LMMSE detector (even if the latter gets posterior feedback as in [148]). And furthermore, the novel derivation also sheds light on the remarkable and so far unexplained performance gain that LMMSE detectors experience with posterior information feedback.

4.2.4 LMMSE-SIC Detection

The new algorithm mentioned above improves on the conventional LMMSE detector by exploiting the knowledge that the data symbols are chosen from a discrete alphabet, and hence cannot have arbitrary values. We will thus briefly introduce one last conventional MIMO detector which is based on a similar idea: in order to exploit the discrete symbol constellation, it augments the linear filter with a nonlinear interference cancellation step.

A *successive interference cancellation* (SIC) detector processes the N_t data streams sequentially in the order $(\pi(0), \pi(1), \dots, \pi(N_\mathrm{t}-1))$, where π is a permutation of the set $\{0, \dots, N_\mathrm{t}-1\}$. It starts by computing the filtered estimate $\tilde{x}_{\pi(0)}$ using one of the linear filters from the previous sections (we will only consider LMMSE, but ZF could also be used). While the L-values for that stream are computed based on $\tilde{x}_{\pi(0)}$, it also obtains a hard estimate

$$\hat{x}_{\pi(0)} = Q(\tilde{x}_{\pi(0)}) \tag{4.45}$$

where $Q(\tilde{x}) \triangleq \arg\min_{\hat{x}\in\mathcal{X}} |\hat{x} - \tilde{x}|$ quantizes $\tilde{x}$ to the closest constellation point (as discussed on page 139, this should be done using the *unbiased* LMMSE estimate).

For the detection of the next data stream $\pi(1)$, the estimated interference from the already detected stream, $\boldsymbol{h}_{\pi(0)}\hat{x}_{\pi(0)}$, is cancelled from the observation, yielding

$$\begin{aligned}\tilde{\boldsymbol{y}}_1 &\triangleq \boldsymbol{y} - \boldsymbol{h}_{\pi(0)}\hat{x}_{\pi(0)} \\ &= \boldsymbol{H}_{\setminus\pi(0)}\boldsymbol{x}_{\setminus\pi(0)} + \boldsymbol{h}_{\pi(0)}(x_{\pi(0)} - \hat{x}_{\pi(0)}) + \boldsymbol{w}\end{aligned} \tag{4.46}$$

which is filtered with a new LMMSE filter (since $\boldsymbol{C}_{\tilde{\boldsymbol{y}}_1} \neq \boldsymbol{C}_{\boldsymbol{y}}$), resulting in $\tilde{x}_{\pi(1)}$. The filtering, slicing, and interference-cancellation steps are then iterated until all data streams have been detected.

This slightly informal description of the LMMSE-SIC detector shall suffice for us; a more detailed explanation can be found e. g. in [73].

The possible advantage of LMMSE-SIC over the plain LMMSE receiver is obvious: if the hard estimates $\hat{x}$ happen to be correct, the remaining data symbols are affected by less interference and can thus be detected more reliably. But the approach also has its downside. A problem which is inherent to the hard-estimation step is the possible error propagation, which happens if the guess $\hat{x}$ is wrong, and which can actually lead to *less* reliable decisions downstream. A related issue is the question of how to choose the detection ordering. In order to minimize the probability of error propagation, one should proceed from the strongest to the weakest stream (in terms of the signal-to-interference-plus-noise ratio, SINR). This is the ordering used in the so-called *V-BLAST* algorithm [149]. However, determining the stream with the highest SINR in each iteration is a complex operation, which might dominate the total runtime.

Intuitively, a superior approach would be to estimate and cancel the interference in a *parallel* rather than sequential manner. This would avoid the whole problem of ordering, and it would also mean that *every* data stream would benefit from interference cancellation from *all* other streams. Also, more efficient hardware implementations would be possible, since they could process the streams in parallel, rather than one after the other (and not having to sort the data streams would further simplify VLSI implementations, since sorting is quite expensive in hardware).

There exist some publications about LMMSE detection with *parallel interference cancellation* (PIC), for example [125]. Note, however, that this MMSE-PIC algorithm is *not* what we are discussing here, since its PIC-step is solely based on the decoder feedback. In fact, the affine MMSE detector that we have derived in Section 4.2.3 is nothing but the MMSE-PIC algorithm; the eponymous interference cancellation is simply the subtraction $\boldsymbol{y} - \boldsymbol{H}\mathbb{E}[\boldsymbol{x}]$ in (4.39). In contrast, the LMMSE-Soft-PIC algorithm that will be the topic of the following section is closer to being a dual of the LMMSE-SIC detector, as it estimates the interference based on information that is intrinsically computed by the MIMO detector (and hence also works in non-iterative receivers, though it can certainly be used in BICM-ID receivers, too, by combining this intrinsic information with the extrinsic decoder feedback).

4.2.5 Numerical Results

We close this section with a performance comparison of the low-complexity MIMO detectors that we have introduced so far. As a standard configuration that will be used not only in this section, but for most numerical results in this chapter, we use a 4×4 MIMO system with a Gray-mapped 16-QAM modulation.

Several publications compare MIMO detectors by their uncoded bit error rate performance. In order to demonstrate the flaws of this metric, we begin with an uncoded BER plot in Figure 4.6. In all figures within this section, "ordered LMMSE-SIC" uses the V-BLAST ordering from [149], while "unordered LMMSE-SIC" processes the streams in their natural order, i. e., uses the identity function for the permutation π.

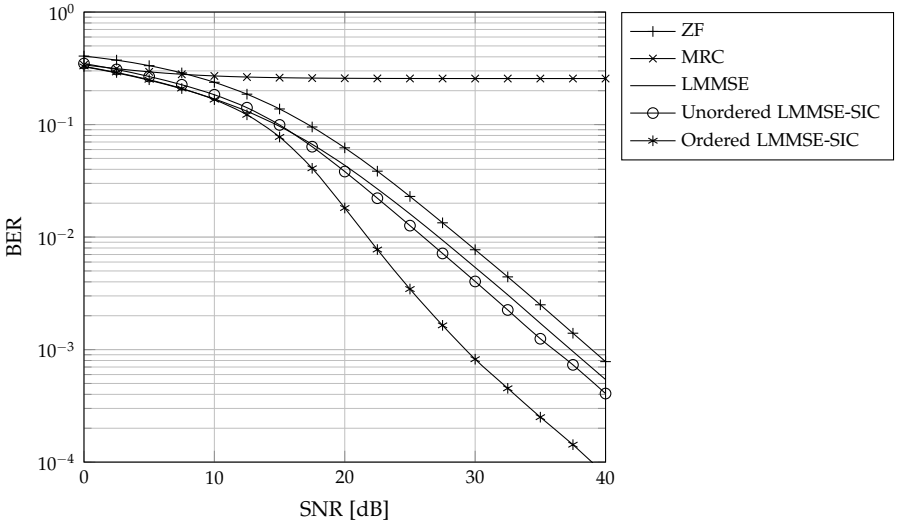


Figure 4.6: Uncoded bit error rates of a 4×4 MIMO system with a Gray-mapped 16-QAM modulation

The first obvious result is that the performance of the MRC receiver is completely uninteresting. Only in the very low SNR range does it perform comparable to LMMSE (with and without SIC) and better than ZF. But at around 7.5 dB, it crosses the performance plot of ZF, and stays at a BER of around 0.25 even in high SNR. This behaviour is not surprising, however; it is an immediate consequence of the fact that MRC ignores the spatial interference and only considers the noise component, so it cannot be expected to perform well in high SNR, where the total distortion is dominated by the interference.

Among the other algorithms, ZF is clearly the worst, with a gap of around 1.5 dB to LMMSE. Between LMMSE and unordered LMMSE-SIC, there is no clear winner. Below approximately 16 dB, the SIC step makes so many erroneous decisions that it actually degrades the BER compared to plain LMMSE. In high SNR, on the other hand, unordered LMMSE-SIC outperforms LMMSE by around 1.3 dB. By far the best algorithm, however, is the ordered LMMSE-SIC, with a gain of around 8 dB compared to LMMSE in high SNR.

Figure 4.7 compares the same MIMO receivers based on their maximum data rates $R_{\max}$, which are obtained from the mutual information over the equivalent BICM channel as discussed in Section 4.1.4. Just as above, MRC shows the most striking behaviour, its spectral efficiency converges to just around 5 bit / channel use (bpcu) in the high SNR range, whereas the system configuration would allow for a maximum spectral efficiency

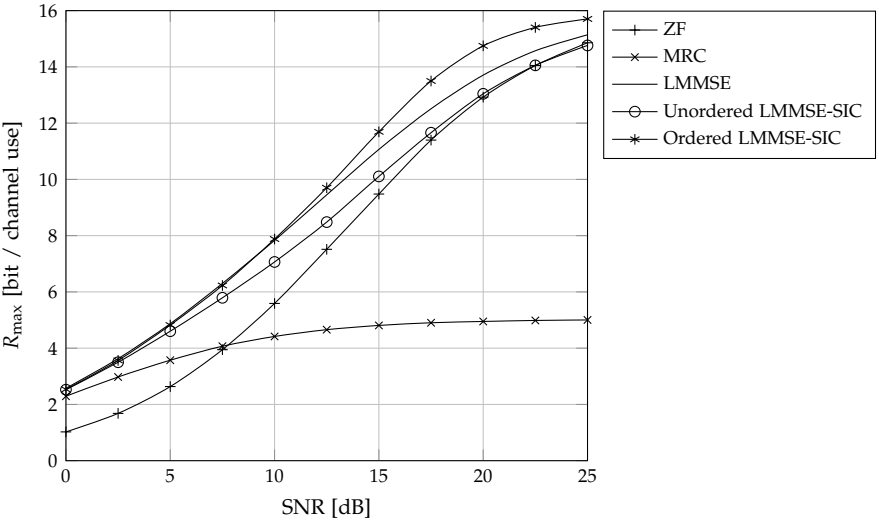


Figure 4.7: Upper bounds on the achievable data rates of a 4×4 MIMO system with a Gray-mapped 16-QAM modulation, for different low-complexity detectors

of 16 bpcu. In contrast to the uncoded BER plot, however, unordered LMMSE-SIC is now worse than plain LMMSE over the whole SNR range of interest; in high SNR, it approaches the rather suboptimal performance of ZF. Finally, the comparison between LMMSE and ordered LMMSE-SIC is far from being as clear-cut as it was in the uncoded BER plot. Both algorithms show virtually the same results in low to medium SNR; only in high SNR can the ordered LMMSE-SIC receiver outperform plain LMMSE.

In a last simulation, we compare the detectors in a coded system, using the practically relevant LDPC code from the IEEE 802.11n standard with rate $1/2$ and a block length of 1944 code bits. As seen in Figure 4.8, the gap between ordered LMMSE-SIC and plain LMMSE has diminished to less than 0.1 dB, which can be neglected for most practical purposes. ZF, on the other hand, now performs more than 2.5 dB worse than LMMSE, and the error propagation in the unordered LMMSE-SIC receiver is so severe that this algorithm is even worse than ZF, let alone plain LMMSE without interference cancellation. The maximum ratio combiner is omitted from the figure, since its WER is 1 over the whole SNR range. Again, this is not surprising, since we have seen in Figure 4.7 that its spectral efficiency hardly exceeds 5 bpcu, whereas the system that has been simulated in Figure 4.8 operates at a rate of 8 bpcu (a raw bit rate of 16 bpcu, times a code rate of $1/2$).

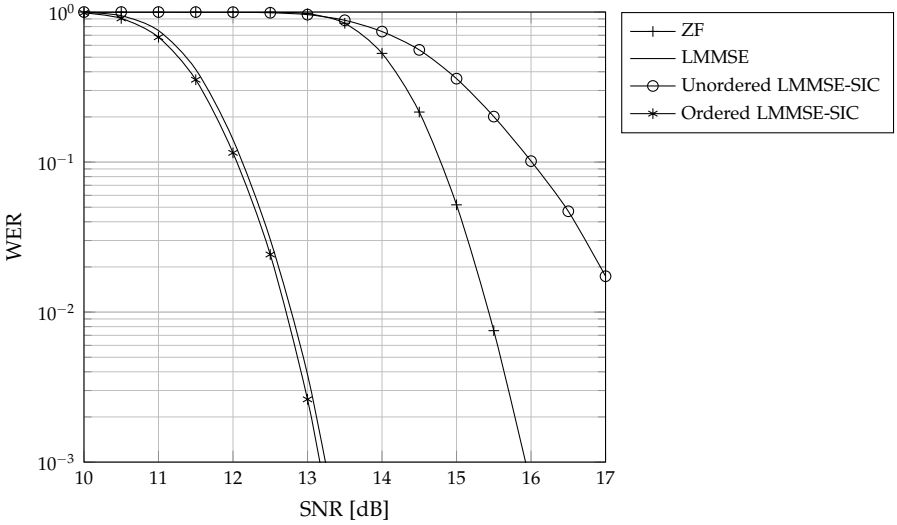


Figure 4.8: Word error rates of an LDPC-coded 4×4 MIMO system, for different detectors

One conclusion of this study is that uncoded BER plots should be interpreted with care, since one can easily draw misleading (or plain wrong) conclusions from them. Reading only Figure 4.6, one would be tempted to think that the complexity that is required for ordering the streams by their SINR is well spent—it yields a gain of 8 dB over LMMSE without SIC, after all. The more practical results from Figure 4.8, based on an LDPC-coded system, are disillusioning however. One reason for this difference between coded and uncoded results is that uncoded BER plots tend to cover an unrealistically[11] high SNR range of, say, 30–40 dB, where most hard decisions made by the interference estimator are correct. The problem of error propagation, which dominates the system performance in the lower SNR range, is thus hardly reflected by the uncoded results.

Another result is that, since the ZF and unordered LMMSE-SIC receivers perform significantly worse than LMMSE with a comparable (ZF) or even higher (unordered LMMSE-SIC) complexity, we will drop them from our further considerations. The MRC receiver, on the other hand, is much simpler than LMMSE as it does not require any matrix inversion. Its complexity is only $\mathcal{O}(N_\mathrm{t}N_\mathrm{r})$ rather than $\mathcal{O}({N_\mathrm{r}}^2(N_\mathrm{t}+N_\mathrm{r}))$, or, for symmetric systems, quadratic rather than cubic. We will therefore return to it in Section 4.4, despite the devastating performance that we have observed in this study.

[11] "Unrealistic" in the sense that one would not operate the system that we have studied here in such an high SNR. Indeed, we have seen in Figure 4.4 that the 16-QAM modulation becomes a performance bottleneck already at around 15 dB, so at 30 dB, one would certainly switch to a higher-order modulation scheme.

4.3 MIMO Detection Based on Inclusive DM: Iterative LMMSE Receivers

Let us recap our discussion of MIMO detectors so far. We have started in Section 4.1.3 with the "exact" BICM-ID MIMO receiver, which is a simple generalization of the scalar BICM-ID detector from Section 2.3 to multi-dimensional data symbols and vector-valued observations. It is obtained by applying the well-known loopy belief propagation (BP) algorithm to the joint posterior

$$p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x} \mid \boldsymbol{y}) \propto p(\boldsymbol{b})\, p(\boldsymbol{c} \mid \boldsymbol{b}) \prod_{n=0}^{N-1} p(\boldsymbol{y}_n \mid \boldsymbol{x}_n) \prod_{i=0}^{N_\mathrm{t}-1} p(x_{n,i} \mid \boldsymbol{c}_{n,i}) \tag{4.47}$$

which fully describes the transmission system (assuming perfect CSI at the receiver). Application of BP to (4.47) thus derives the whole iterative receiver "in one go"; it not only yields descriptions of the two major algorithmic blocks (the detector and decoder, corresponding to the factors $p(\boldsymbol{y}_n \mid \boldsymbol{x}_n) \prod_{i=0}^{N_\mathrm{t}-1} p(x_{n,i} \mid \boldsymbol{c}_{n,i})$ and $p(\boldsymbol{c} \mid \boldsymbol{b})$, respectively), but also of their interaction (exchange of extrinsic L-values).

We have then introduced three kinds of low-complexity MIMO detectors, which all share a common structure: a concatenation of a linear filter, which (more or less) separates the spatially superimposed data streams, followed by scalar detectors. We have furthermore seen an example of a nonlinear preprocessor, which attempts to improve on the linear filters by exploiting information about the discrete modulation alphabet. A common characteristic of these low-complexity detectors is that they are *ad hoc* replacements of the exact detector, which, for lack of alternatives, are usually connected to the decoder in the same way as the exact detector: via extrinsic L-values. But, as shown in the already mentioned paper [148], providing the LMMSE-based detector with posterior L-values can yield significantly better results than with extrinsic L-values. This proves that it is insufficient to just replace the implementation of an algorithmic component with a heuristic approximation, while leaving its interface unaltered.

In order to understand the phenomenon discovered in [148], we will now approach the design of low-complexity MIMO receivers from a system perspective. We have seen in Chapter 3 that divergence minimization (DM), which unifies several seemingly unrelated techniques like belief propagation and expectation maximization under a general framework, is able to yield a variety of novel parameter estimators, covering several practically interesting points on the performance-complexity tradeoff curve. In this and the following section, we will explore whether DM is also applicable to the design of *detection* algorithms.[12]

The work presented in this section has partly been published in [118].

[12]Strictly speaking, we already know that the answer is in the affirmative, since the exact BICM-ID detector is based on BP, which is a special case of DM. The question we are really interested in is thus, whether we can use the additional degrees of freedom that the general DM approach provides to derive suboptimal, but less complex algorithms, which however are still built upon a solid theoretical fundament.

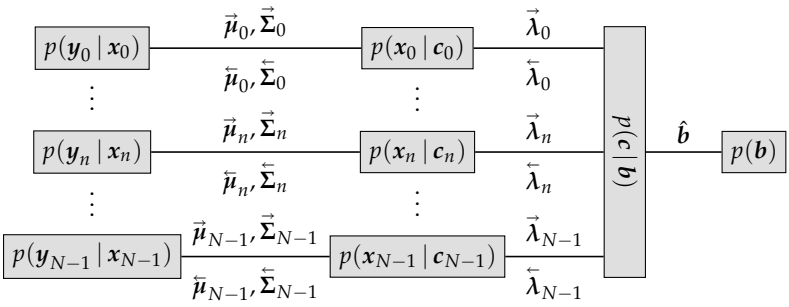


Figure 4.9: Forney-style factor graph of the joint posterior (4.47), and the information that flows along its edges

4.3.1 LMMSE Detection with Soft Parallel Interference Cancellation

The fundamental principle of divergence minimization is to approximate the posterior distribution p of interest with a simpler distribution q, which is found by minimizing some divergence measure between p and q. The traditional BICM-ID receiver, for example, can be derived by choosing a fully factorized auxiliary distribution

$$q(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x}) = \prod_{j=0}^{N_\mathrm{b}-1} q(b_j) \prod_{n=0}^{N-1} \prod_{i=0}^{N_\mathrm{t}-1} q(x_{n,i}) \prod_{k=0}^{K-1} q(c_{n,i,k}) \tag{4.48}$$

and refining the approximations of the individual factors iteratively, according to loopy belief propagation (BP; see Section B.5.1). As we have seen, the huge complexity of the exact BP-based MIMO detector is caused by the fact that the factors $q(x_{n,i})$ are discrete distributions over the modulation alphabet $\mathcal{X}$, which leads to BP message updates that consist of sums with exponentially many terms.

Expectation propagation (EP; see Section B.5.2) is a generalization of BP which allows us to impose further constraints on the auxiliary distribution. In order to remove the computational bottleneck caused by the discrete variables $x_{n,i}$, we exploit this additional degree of freedom by constraining the factors $q(x_{n,i})$ to proper Gaussian distributions:

$$q(x_{n,i}) \stackrel{!}{=} \mathcal{CN}(x_{n,i} \mid \mu_{n,i}, \sigma_{n,i}^2). \tag{4.49}$$

Equivalently, we constrain the vectorwise distributions $q(\boldsymbol{x}_n)$ to proper Gaussians with mean vector $\boldsymbol{\mu}_n$ and diagonal covariance matrix $\boldsymbol{\Sigma}_n$. Both perspectives will be useful in the following derivation of the EP-based MIMO detector. The underlying graphical model has been shown on page 128 and is repeated in Figure 4.9 for convenience.

Interference Cancellation and Spatial Filtering

We begin our derivation with the message from $p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)$ to $p(\boldsymbol{x}_n \mid \boldsymbol{c}_n)$. We will refer to these nodes as the *equalizer*[13] and *demapper* nodes, respectively, so the messages between them will be called $m_{\text{equ}\to\text{dem}}(\boldsymbol{x}_n)$ and $m_{\text{dem}\to\text{equ}}(\boldsymbol{x}_n)$. Due to the Gaussian constraint on $q(\boldsymbol{x}_n)$, both of these messages are Gaussian, too (up to constant factors), whose parameters are marked with left-to-right and right-to-left arrows.

According to the general EP message update rule (B.56), the computation at the equalizer nodes can be derived from

$$m_{\text{equ}\to\text{dem}}(\boldsymbol{x}_n) = \frac{\text{proj}_{\mathcal{CN}}\left[Z^{-1}\, p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)\, m_{\text{dem}\to\text{equ}}(\boldsymbol{x}_n)\right]}{m_{\text{dem}\to\text{equ}}(\boldsymbol{x}_n)} \tag{4.50}$$

where Z is a normalization constant, and where the projection is into the family of proper Gaussian distributions with diagonal covariance matrix. Up to constant factors, the argument of the projection is

$$\begin{aligned}
&p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)\, m_{\text{dem}\to\text{equ}}(\boldsymbol{x}_n) \\
&\propto \exp\left(-\frac{\|\boldsymbol{y}_n - \boldsymbol{H}_n\boldsymbol{x}_n\|^2}{N_0} - (\boldsymbol{x}_n - \overleftarrow{\boldsymbol{\mu}}_n)^{\mathsf{H}}\overleftarrow{\boldsymbol{\Sigma}}_n^{-1}(\boldsymbol{x}_n - \overleftarrow{\boldsymbol{\mu}}_n)\right) \\
&\propto \exp\left(-\boldsymbol{x}_n^{\mathsf{H}}\left(\boldsymbol{H}_n^{\mathsf{H}}N_0^{-1}\boldsymbol{H}_n + \overleftarrow{\boldsymbol{\Sigma}}_n^{-1}\right)\boldsymbol{x}_n + 2\,\text{Re}\left(\boldsymbol{x}_n^{\mathsf{H}}\left(\boldsymbol{H}_n^{\mathsf{H}}N_0^{-1}\boldsymbol{y}_n + \overleftarrow{\boldsymbol{\Sigma}}_n^{-1}\overleftarrow{\boldsymbol{\mu}}_n\right)\right)\right)
\end{aligned} \tag{4.51}$$

with $\overleftarrow{\boldsymbol{\Sigma}}_n = \text{diag}\left(\overleftarrow{\sigma}_{n,0}^2, \ldots, \overleftarrow{\sigma}_{n,N_{\text{t}}-1}^2\right)$. After normalization (multiplication with Z^{-1}), this becomes a Gaussian distribution $\mathcal{CN}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$ with covariance matrix

$$\begin{aligned}
\boldsymbol{\Sigma}_n &= \left(\boldsymbol{H}_n^{\mathsf{H}}N_0^{-1}\boldsymbol{H}_n + \overleftarrow{\boldsymbol{\Sigma}}_n^{-1}\right)^{-1} \\
&\overset{(a)}{=} \overleftarrow{\boldsymbol{\Sigma}}_n - \overleftarrow{\boldsymbol{\Sigma}}_n\boldsymbol{H}_n^{\mathsf{H}}\left(\boldsymbol{H}_n\overleftarrow{\boldsymbol{\Sigma}}_n\boldsymbol{H}_n^{\mathsf{H}} + N_0\boldsymbol{I}_{N_{\text{r}}}\right)^{-1}\boldsymbol{H}_n\overleftarrow{\boldsymbol{\Sigma}}_n \\
&\overset{(b)}{=} \overleftarrow{\boldsymbol{\Sigma}}_n - \overleftarrow{\boldsymbol{\Sigma}}_n\boldsymbol{H}_n^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}\boldsymbol{H}_n\overleftarrow{\boldsymbol{\Sigma}}_n
\end{aligned} \tag{4.52}$$

and mean vector

$$\begin{aligned}
\boldsymbol{\mu}_n &= \left(\overleftarrow{\boldsymbol{\Sigma}}_n - \overleftarrow{\boldsymbol{\Sigma}}_n\boldsymbol{H}_n^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}\boldsymbol{H}_n\overleftarrow{\boldsymbol{\Sigma}}_n\right)\left(\boldsymbol{H}_n^{\mathsf{H}}N_0^{-1}\boldsymbol{y}_n + \overleftarrow{\boldsymbol{\Sigma}}_n^{-1}\overleftarrow{\boldsymbol{\mu}}_n\right) \\
&= \overleftarrow{\boldsymbol{\mu}}_n + \overleftarrow{\boldsymbol{\Sigma}}_n\boldsymbol{H}_n^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}\left(\boldsymbol{y}_n - \boldsymbol{H}_n\overleftarrow{\boldsymbol{\mu}}_n\right)
\end{aligned} \tag{4.53}$$

where (a) uses the matrix inversion lemma, and (b) introduces the shorthand notation

$$\boldsymbol{C}_{\boldsymbol{y}_n} = \boldsymbol{H}_n\overleftarrow{\boldsymbol{\Sigma}}_n\boldsymbol{H}_n^{\mathsf{H}} + N_0\boldsymbol{I}_{N_{\text{r}}}. \tag{4.54}$$

[13]The rationale for this name is that the equalizer node separates the spatially superimposed data streams, just as a conventional equalizer for frequency-selective channels separates the temporally superimposed data symbols.

Note that (4.53) consists of parallel interference cancellation using $\overleftarrow{\boldsymbol{\mu}}_n$ (the latest estimate of the transmitted signal $\boldsymbol{x}_n$), followed by an LMMSE filter. In fact, it is essentially the same expression as (4.39). The small, but important difference it that the terms $\mathbb{E}[\boldsymbol{x}]$ and $\boldsymbol{C}_{\boldsymbol{x}}$ in (4.39) are only vaguely defined, as we argued on page 140. In (4.53), they are replaced by the well defined message parameters $\overleftarrow{\boldsymbol{\mu}}$ and $\overleftarrow{\boldsymbol{\Sigma}}$, whose computation will be derived in the next subsection.

We have seen that the argument of the projection operator is already Gaussian, but with a non-diagonal covariance matrix (4.52). Since we know that the projection of a joint distribution into the set of fully factorized distributions is equal to the product of its marginals (see the derivation on page 243), the result of our projection into the Gaussians with diagonal covariance matrix simply consists of neglecting the off-diagonal elements of (4.52). The parameters of the updated symbolwise beliefs $q(x_{n,i})$ thus become

$$\mu_{n,i} = \overleftarrow{\mu}_{n,i} + \overleftarrow{\sigma}^2_{n,i} \boldsymbol{h}^{\mathsf{H}}_{n,i} \boldsymbol{C}^{-1}_{\boldsymbol{y}_n} (\boldsymbol{y}_n - \boldsymbol{H}_n \overleftarrow{\boldsymbol{\mu}}_n) \tag{4.55}$$

$$\sigma^2_{n,i} = \overleftarrow{\sigma}^2_{n,i} - \overleftarrow{\sigma}^4_{n,i} \boldsymbol{h}^{\mathsf{H}}_{n,i} \boldsymbol{C}^{-1}_{\boldsymbol{y}_n} \boldsymbol{h}_{n,i} \tag{4.56}$$

where $\boldsymbol{h}_{n,i}$ is the ith column of the channel matrix $\boldsymbol{H}_n$. Note that the above equations are obtained by taking the ith coefficient of the mean vector (4.53), and the ith diagonal element of the covariance matrix (4.52).

Finally, the new message $m_{\text{equ}\to\text{dem}}(x_{n,i})$ from the equalizer to the demapper is computed by dividing the updated belief $q(x_{n,i})$ by the incoming message $m_{\text{dem}\to\text{equ}}(x_{n,i})$. Since the product of two members of an exponential family is obtained by summing their natural parameters, which in the case of proper Gaussians are $\mu\,\sigma^{-2}$ and σ^{-2} (up to constant factors; c. f. page 215), the parameters of the forward and backward symbolwise messages are related via

$$\frac{1}{\sigma^2} = \frac{1}{\vec{\sigma}^2} + \frac{1}{\overleftarrow{\sigma}^2} \qquad \text{and} \qquad \frac{\mu}{\sigma^2} = \frac{\vec{\mu}}{\vec{\sigma}^2} + \frac{\overleftarrow{\mu}}{\overleftarrow{\sigma}^2} \tag{4.57}$$

from which the parameters of $m_{\text{equ}\to\text{dem}}(x_{n,i})$ are easily found as

$$\vec{\mu}_{n,i} = \overleftarrow{\mu}_{n,i} + \frac{\boldsymbol{h}^{\mathsf{H}}_{n,i} \boldsymbol{C}^{-1}_{\boldsymbol{y}_n} (\boldsymbol{y}_n - \boldsymbol{H}_n \overleftarrow{\boldsymbol{\mu}}_n)}{\boldsymbol{h}^{\mathsf{H}}_{n,i} \boldsymbol{C}^{-1}_{\boldsymbol{y}_n} \boldsymbol{h}_{n,i}} \tag{4.58}$$

$$\vec{\sigma}^2_{n,i} = \left(\boldsymbol{h}^{\mathsf{H}}_{n,i} \boldsymbol{C}^{-1}_{\boldsymbol{y}_n} \boldsymbol{h}_{n,i} \right)^{-1} - \overleftarrow{\sigma}^2_{n,i}. \tag{4.59}$$

Computing the Symbol Statistics

We continue with the operation at the demapper node $p(x_{n,i} \mid \boldsymbol{c}_{n,i})$.

The backward message $m_{\text{dem}\to\text{equ}}(x_{n,i})$, which we derive in this section, essentially consists of the first two moments $\overleftarrow{\mu}_{n,i}$ and $\overleftarrow{\sigma}^2_{n,i}$ of the symbol $x_{n,i}$, which are required

for the interference cancellation and LMMSE filtering, respectively, as shown in (4.58) and (4.59). Starting again from the generic EP message update rule (B.56), we obtain

$$m_{\text{dem}\to\text{equ}}(x_{n,i}) = \frac{\text{proj}_{\mathcal{CN}}\left[Z^{-1}\sum_{\boldsymbol{c}_{n,i}} p(x_{n,i}\,|\,\boldsymbol{c}_{n,i})\, m_{\text{equ}\to\text{dem}}(x_{n,i})\, m_{\text{dec}\to\text{dem}}(\boldsymbol{c}_{n,i})\right]}{m_{\text{equ}\to\text{dem}}(x_{n,i})}. \tag{4.60}$$

The message $m_{\text{equ}\to\text{dem}}(x_{n,i}) = \mathcal{CN}(x_{n,i}\,|\,\vec{\mu}_{n,i}, \vec{\sigma}^2_{n,i})$ comes from the spatial equalizer with the parameters computed according to (4.58) and (4.59), and the feedback from the decoder to the demapper consists of the usual extrinsic information

$$m_{\text{dec}\to\text{dem}}(\boldsymbol{c}_{n,i}) \propto \prod_{k=0}^{K-1} \exp\left(c_{n,i,k}\overleftarrow{\lambda}_{n,i,k}\right) \tag{4.61}$$

as in (4.14). With the function $p(x_{n,i}\,|\,\boldsymbol{c}_{n,i}) = \mathbb{1}\left[x_{n,i} - \mathcal{M}(\boldsymbol{c}_{n,i})\right]$, which represents the deterministic symbol mapping, we can write the argument of the projection as

$$f(x_{n,i}) = Z^{-1}\exp\left(-\frac{|x_{n,i}-\vec{\mu}_{n,i}|^2}{\vec{\sigma}^2_{n,i}} + \sum_{k=0}^{K-1}\mathcal{M}_k^{-1}(x_{n,i})\overleftarrow{\lambda}_{n,i,k}\right) \tag{4.62}$$

where $\mathcal{M}_k^{-1}(x_{n,i})$ denotes the kth bit of the bit label associated to $x_{n,i}$ as in (3.58), and where the normalization constant Z is chosen such that $\sum_{x_{n,i}\in\mathcal{X}} f(x_{n,i}) = 1$.

According to the principle of moment matching, the projection of $f(x_{n,i})$ into the family of proper Gaussians consists of computing the first two moments of $f(x_{n,i})$

$$\mu_{n,i} = \sum_{x_{n,i}\in\mathcal{X}} f(x_{n,i})\, x_{n,i} \tag{4.63}$$

$$\sigma^2_{n,i} = \sum_{x_{n,i}\in\mathcal{X}} f(x_{n,i})\, |x_{n,i}-\mu_{n,i}|^2 \tag{4.64}$$

which are finally used to determine the updated parameters of $m_{\text{dem}\to\text{equ}}(x_{n,i})$ as

$$\overleftarrow{\mu}_{n,i} = \frac{\mu_{n,i}\,\vec{\sigma}^2_{n,i} - \vec{\mu}_{n,i}\,\sigma^2_{n,i}}{\vec{\sigma}^2_{n,i} - \sigma^2_{n,i}} \tag{4.65}$$

$$\overleftarrow{\sigma}^2_{n,i} = \frac{\vec{\sigma}^2_{n,i}\,\sigma^2_{n,i}}{\vec{\sigma}^2_{n,i} - \sigma^2_{n,i}} \tag{4.66}$$

where we have again used (4.57).

It can happen that $\vec{\sigma}^2_{n,i} \leq \sigma^2_{n,i}$, which would yield an infinite or negative variance $\overleftarrow{\sigma}^2_{n,i}$; clearly a nonsensical result. However, this does not reveal a mistake in the derivation; it is known that the division in the EP message update step can result in unnormalizable

densities [65, Section 14.3.3.2].[14] To the best of my knowledge, there is no systematic way to handle this situation. In this particular case, I simply propose to feed the full belief back to the equalizer, i.e., to set $\overleftarrow{\mu}_{n,i} = \mu_{n,i}$ and $\overleftarrow{\sigma}^2_{n,i} = \sigma^2_{n,i}$. Clearly, this is just a heuristic, and there may be more sophisticated ways to prevent $\overleftarrow{\sigma}^2_{n,i}$ from becoming negative. However, this situation occurs quite rarely, typically in a fraction of 10^{-5} to 10^{-3} of all processed symbol vectors, so that I do not expect that this heuristic causes a noteworthy performance degradation.

Symbolwise Demapping

For completeness, we finish this derivation with $m_{\mathrm{dem}\to\mathrm{dec}}(c_{n,i,k}) \propto \mathcal{B}(c_{n,i,k} \mid \vec{\lambda}_{n,i,k})$, the message from the demapper to the decoder. It is parameterized by the intrinsic L-value

$$\vec{\lambda}_{n,i,k} = \log \frac{\sum\limits_{x_{n,i}\in\mathcal{X}_k^1} \exp\left(-\frac{|x_{n,i}-\vec{\mu}_{n,i}|^2}{\vec{\sigma}^2_{n,i}} + \boldsymbol{c}_{n,i}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_{n,i}\right)}{\sum\limits_{x_{n,i}\in\mathcal{X}_k^0} \exp\left(-\frac{|x_{n,i}-\vec{\mu}_{n,i}|^2}{\vec{\sigma}^2_{n,i}} + \boldsymbol{c}_{n,i}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}_{n,i}\right)} - \overleftarrow{\lambda}_{n,i,k} \tag{4.67}$$

where, analogously to (2.23), $\boldsymbol{c}_{n,i}$ contains the bit label of the current $x_{n,i}$.

4.3.2 Discussion of the Proposed LMMSE-Soft-PIC Receiver

In Section 4.2.4 about the LMMSE-SIC detector, we argued that *parallel* interference cancellation would be a more promising approach than the sequential one, because the detection of every stream would benefit from IC of all other streams, and because the problem of ordering would be avoided. As we see in (4.58), the detector that we have derived above indeed performs a parallel IC by means of the subtraction $\boldsymbol{y} - \boldsymbol{H}\overleftarrow{\boldsymbol{\mu}}$. Furthermore, the interference cancellation is *soft* in the sense that the estimate $\overleftarrow{\boldsymbol{\mu}}$ of the transmitted signal is not obtained by hard-quantizing the LMMSE result to the closest constellation point like in (4.45), but rather by computing a kind of soft symbol as shown in (4.63)–(4.65). Due to this soft-IC approach, the proposed detector also avoids the problem of error propagation. Because of these properties, the algorithm will be called *LMMSE-Soft-PIC* in the following.

Comparison with the Conventional LMMSE-PIC Receiver

We also mentioned in Section 4.2.4 that the name *MMSE-PIC* is sometimes used in the literature (e. g. [125]) to refer to the iterative MMSE detector that we have derived in Section 4.2.3. While both receivers include a parallel IC step, they differ in the way how

[14] As a side note, this situation cannot occur in (4.59). By writing $\boldsymbol{C}_{\boldsymbol{y}}$ as $\boldsymbol{h}_i\overleftarrow{\sigma}_i^2\boldsymbol{h}_i^{\mathsf{H}} + \boldsymbol{H}_{\setminus i}\overleftarrow{\boldsymbol{\Sigma}}_{\setminus i}\boldsymbol{H}_{\setminus i}^{\mathsf{H}} + N_0\boldsymbol{I}_{N_{\mathrm{T}}}$ and applying the matrix inversion lemma, it can easily be shown that $\vec{\sigma}_i^2 > 0$.

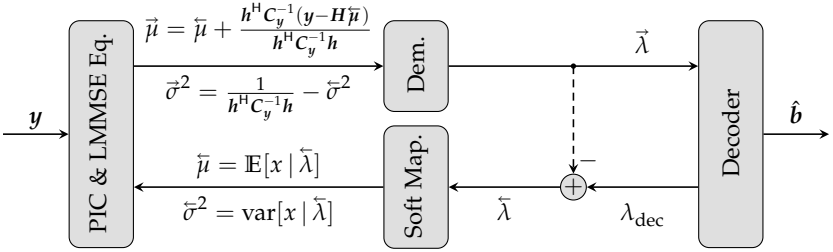


Figure 4.10: Block diagram of the conventional iterative LMMSE-PIC receiver

they compute the estimate $\overleftarrow{\mu}$ of the interference and the associated variance $\overleftarrow{\sigma}^2$. Let us now compare the two algorithms, and in particular their feedback paths, in more detail.

Figure 4.10 shows a block diagram of the conventional algorithm, where all time and stream incides have been omitted for simplicity. The linear (or affine) front-end, labeled "PIC & LMMSE Eq." in the figure, is identical to the one that we have derived above using expectation propagation. It receives the first two moments of the data symbols, $\overleftarrow{\mu}$ and $\overleftarrow{\sigma}^2$, as inputs. In the initial iteration, those are simply given as $\overleftarrow{\mu} = 0$ and $\overleftarrow{\sigma}^2 = E_{\mathrm{s}}$, and in the later iterations, they are computed based on the decoder feedback.

Using these symbol statistics, the front-end computes $\vec{\mu}$ and $\vec{\sigma}^2$ according to the formulas given in the figure, and passes them to the streamwise demapper. Note that the given expressions for $\vec{\mu}$ and $\vec{\sigma}^2$ are taken from (4.58) and (4.59), but it is easy to show that the conventional derivation (4.41)–(4.44) of the LMMSE detector is equivalent.

The demapper then determines the L-values $\vec{\lambda}$ in the usual manner, and passes them to the decoder, which computes the posterior information λ_{dec} and the extrinsic L-values $\overleftarrow{\lambda} = \lambda_{\mathrm{dec}} - \vec{\lambda}$. Finally, the extrinsic L-values are processed by a soft mapper, yielding updated symbol statistics $\overleftarrow{\mu}$ and $\overleftarrow{\sigma}^2$, which closes the loop.

Using the projection operator, the computation at the soft mapper can be written as

$$\mathcal{CN}(\overleftarrow{\mu}, \overleftarrow{\sigma}^2) = \mathrm{proj}_{\mathcal{CN}}\Bigg[\underbrace{\exp\left(\boldsymbol{c}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}\right)}_{\substack{\text{only extrinsic information} \\ \text{from decoder considered}}} \Bigg] \tag{4.68}$$

which emphasizes that its input consists only of the extrinsic L-values from the decoder.

As already mentioned, it has been observed in numerical simulations [148] that the performance of the iterative LMMSE MIMO detector can be improved by ignoring the turbo principle and feeding back the full posterior information from the decoder to the demapper, i. e., by removing the dashed line in Figure 4.10. Then, writing the posterior

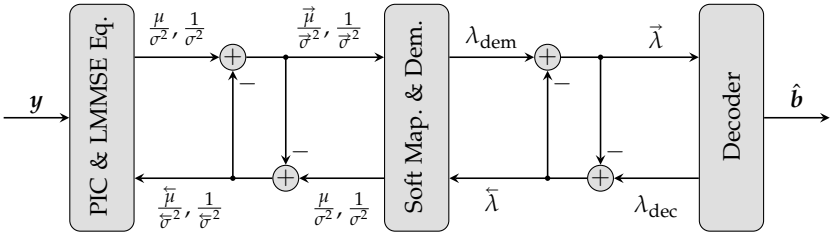


Figure 4.11: Block diagram of the proposed doubly iterative LMMSE-Soft-PIC receiver

L-values as a sum of intrinsic and extrinsic L-values, the operation at the soft mapper can be expressed analogously to (4.68) as

$$\mathcal{CN}(\overleftarrow{\mu}, \overleftarrow{\sigma}^2) = \text{proj}_{\mathcal{CN}}\Big[\underbrace{\exp\left(\boldsymbol{c}^{\mathsf{T}}\vec{\boldsymbol{\lambda}}\right)}_{\substack{\text{intrinsic information} \\ \text{from LMMSE filter}}} \quad \underbrace{\exp\left(\boldsymbol{c}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}\right)}_{\substack{\text{extrinsic information} \\ \text{from decoder}}} \Big] \tag{4.69}$$

where the comment mentions that the intrinsic information originates from the spatial equalizer, even though the arrow labeled with $\vec{\lambda}$ is rooted in the demapper block. Ultimately, all that the demapper does is to take a symbolwise PMF of x, which can equivalently be considered as a joint PMF of the associated bits $(c_0, \ldots, c_{K-1})$, and turns it into a product of bitwise marginal distributions, discarding any information about the correlation between the bits.

Now, contrast the conventional LMMSE-PIC receiver with the proposed EP-based algorithm that is depicted in Figure 4.11. It consists of two similarly structured loops: (a) between the streamwise demapper and the linear (affine) front-end, which performs the interference cancellation and the spatial equalization, and (b) between the demapper and the decoder.

The right-hand loop between demapper and decoder is the standard BICM-ID loop. Here, the two connected components exchange probabilistic information about the code bits in terms of Bernoulli distributions, parameterized by L-values. The left-hand loop between the demapper and the LMMSE-Soft-PIC front-end is conceptually similar. Here, information about the complex data symbols is exchanged in terms of proper Gaussian distributions. The two subtraction blocks in this loop are equivalent to a division of the corresponding Gaussian densities, just as the subtractions of the L-values in the right-hand loop are equivalent to divisions of the corresponding Bernoulli distributions.

We are now in a position to give a first explanation of the phenomenon discovered in [148]. Analogously to (4.68) and (4.69), we can express the computation of the symbol

statistics for the affine front-end, including both the soft mapper & demapper block and the subsequent subtraction point in Figure 4.11, as

$$\mathcal{CN}(\vec{\mu},\vec{\sigma}^2) = \mathrm{proj}_{\mathcal{CN}}\Bigg[\underbrace{\mathcal{CN}(\vec{\mu},\vec{\sigma}^2)}_{\substack{\text{intrinsic information}\\ \text{from LMMSE filter}}} \quad \underbrace{\exp\left(\boldsymbol{c}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}\right)}_{\substack{\text{extrinsic information}\\ \text{from decoder}}} \Bigg] \quad \underbrace{\frac{1}{\mathcal{CN}(\vec{\mu},\vec{\sigma}^2)}}_{\substack{\text{removal of intrinsic}\\ \text{information after proj.}}} . \tag{4.70}$$

Comparing this with (4.68), we see that the difference between the conventional and the proposed LMMSE receivers boils down to one small, but crucial detail: in the terminology of message passing algorithms, the conventional algorithm projects only the backward message, whereas the proposed algorithm projects the whole belief, including the intrinsic information, and subsequently removes the intrinsic information from the result of the projection. Due to the nonlinearity of the projection operator, the two equations will in general yield different results, and the derivation of the generic message update equations in Section B.4.1 on page 249 has shown that (4.70) is the "correct" way.[15]

Equation (4.69) lies somewhat in between (4.68) and (4.70). On the one hand, it does include the intrinsic information in the projection operation, which explains the better performance that can be achieved with posterior information feedback. However, it differs from the proposed equation (4.70) in two points. First, it uses the intrinsic information after it has been processed by the demapper, which, as mentioned above, essentially discards the information about the correlation between the bits. The proposed algorithm, in contrast, uses the intrinsic information in terms of the complex symbol x, which still contains the bitwise correlations. Second, it complies with the turbo principle by removing the intrinsic information after the projection. We will see in the simulation results that are presented later that these differences cause a further performance gain of the proposed LMMSE-Soft-PIC receiver over the conventional LMMSE-PIC receiver with posterior feedback.

Note that very similar considerations have arisen earlier in this thesis. On page 24, in the context of deriving the standard single-antenna BICM-ID algorithm, we have already demonstrated the importance of projecting beliefs rather than messages. And the comparison of extrinsic and posterior feedback has also played a central role in Chapter 3 on parameter estimation; see in particular the discussion of the generic backward pass in Section 3.3.3 on page 51. It is one of the major benefits of the divergence minimization approach to the design of iterative receivers that, while breaking down a complex system into smaller algorithmic components, it still maintains the holistic perspective that is required for also deriving the *interactions* of those components in a systematic manner.

[15] Up to the fact that DM is itself just a heuristic approach to statistical inference, so that the term "correct" should be understood in a very informal sense; hence the quotation marks.

Scheduling

A question that is *not* answered by the divergence minimization framework is that of scheduling. As soon as the receiver contains more than one loop, it is not apparent anymore in which temporal order the algorithmic blocks should be invoked. Finding the optimal schedule is in general a very hard problem, and it is an active field of research (see e. g. the recent publications [154, 157]). In fact, even finding a reasonable optimization criterion is far from obvious. The algorithmic performance in terms of error probabilities is of course an important point, but hardware aspects like runtime, energy consumption, and occupied chip area of the functional units also need to be considered for VLSI implementations.

Due to the complexity of this field of research, I do not attempt to optimize the scheduling of the proposed LMMSE-Soft-PIC receiver. Instead, I have evaluated the performance of several schedules by computer simulations, and have found that one particularly simple schedule (in terms of the overall runtime and the regularity of the schedule) yields very promising results.

As we see in Figure 4.11, the LMMSE-Soft-PIC receiver is doubly-iterative: there is the left-hand loop between the spatial equalizer and the streamwise demappers, and the right-hand BICM-ID loop between demapper and decoder. Since the left and middle block in Figure 4.11 together implement the functionality of the MIMO demapper from Figure 4.1, we will refer to the left-hand loop as *inner* iterations, as they are conceptually confined to the demapper. Correspondingly, the iterations between demapper and decoder will be called *outer* iterations.

My proposal for a simple, but efficient schedule is as follows. During the initial outer iteration, when no feedback from the decoder is available yet, the MIMO detector should perform two inner iterations. The first one is equivalent to the conventional LMMSE MIMO detector, while the second iteration already benefits from interference cancellation, leading to more reliable L-values for the channel decoder. In the later outer iterations,[16] a single inner iteration is sufficient, as I have verified by computer simulations. The computational complexity of the later iterations of the proposed schedule is thus comparable to that of the conventional LMMSE detector, but some memory (one complex and one real number per scalar data symbol) is required in order to store the left-to-right message parameters $\vec{\mu}$ and $\vec{\sigma}^2$ between the outer iterations.

An alternative schedule, which does not require that memory at the cost of additional computations, is the one that I have proposed in [118]. Here, the $\vec{\mu}$ and $\vec{\sigma}^2$ are not stored; instead, the detector always performs two inner iterations, where the first one approximately recovers the $\vec{\mu}$ and $\vec{\sigma}^2$ from the previous outer iteration. This schedule thus treats every outer iteration like the first one. In the low-to-medium SNR range,

[16] Note, however, that it is not necessary to perform outer iterations in order to benefit from the Soft-PIC detector. Due to the two initial inner iterations, the performance gain over the conventional LMMSE detector is already observable in the first outer iteration, i. e., in a plain BICM (rather than BICM-ID) receiver.

both schedules show practically the same algorithmic performance. But as we will see in Section 4.3.4, the former schedule (which stores the $\vec{\mu}$ and $\vec{\sigma}^2$ between outer iterations) yields better results in high SNR, so it should be preferred if the additional memory can be afforded.

Another question that we will study in Section 4.3.4 is whether it makes sense to perform more than two inner iterations. Details will be presented below, but the short answer is no. In fact, we will see that a third inner iteration can even deteriorate the detection results.

Before we turn to the numerical results, however, we will discuss a modification which has been shown to improve the algorithmic performance of the conventional LMMSE detector in BICM-ID receivers, and study its applicability to the LMMSE-Soft-PIC detector: using *widely* rather than *strictly* linear filters.

4.3.3 Widely Linear MMSE Receivers

Preliminaries

In the real-valued world, an estimator is called *linear* if it can be expressed as a matrix multiplication[17]

$$\tilde{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{y} \tag{4.71}$$

with $\tilde{\boldsymbol{x}} \in \mathbb{R}^M, \boldsymbol{y} \in \mathbb{R}^N$, and $\boldsymbol{A} \in \mathbb{R}^{M \times N}$.

A straightforward extension to complex quantities is to keep the expression (4.71), but to allow for complex-valued coefficients in the observation $\boldsymbol{y}$, the filter matrix $\boldsymbol{A}$, and the estimate $\tilde{\boldsymbol{x}}$. However, there exists another, more general extension, which additionally considers the complex conjugate of the observation:

$$\tilde{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{B}\boldsymbol{y}^*. \tag{4.72}$$

It is obvious that (4.72) is at least as expressive as (4.71), since we can choose $\boldsymbol{B} = \boldsymbol{0}$. That it is strictly more expressive can be seen with a simple one-dimensional example. Assume that y is the superposition of x and zero-mean complex-valued noise. If we know that x is real-valued, the imaginary part of y consists solely of noise, and an intuitively reasonable estimator would be $\tilde{x} = \operatorname{Re} y$. With (4.71), such an estimator cannot be expressed, because there is no $a \in \mathbb{C}$ such that $\forall y \in \mathbb{C} : a\,y = \operatorname{Re} y$. With the formulation (4.72), in contrast, this estimator is easily obtained: by choosing $a = b = \frac{1}{2}$, we get $\tilde{x} = \frac{1}{2}y + \frac{1}{2}y^* = \operatorname{Re} y$.

A filter of the form (4.72) is called *widely linear*, and the "normal" linear filter (4.71) will be called *strictly linear* if we want to emphasize the difference. An introduction to complex-valued statistical signal processing and widely linear estimators can be found in the textbook [115].

[17] For simplicity, we assume zero-mean variables in this section.

A natural question to ask is whether our LMMSE MIMO detectors can benefit from the additional expressiveness of (4.72), i. e., whether *widely linear minimum mean-squared error* (WLMMSE) detectors have the potential to outperform their LMMSE counterparts.

In order to answer this question, we first define some vocabulary from the field of complex-valued statistical signal processing. Let $\boldsymbol{x}$ be a complex random vector with mean $\bar{\boldsymbol{x}} \triangleq \mathbb{E}[\boldsymbol{x}]$. The regular covariance matrix of $\boldsymbol{x}$ is defined as

$$\boldsymbol{C}_{\boldsymbol{x}} \triangleq \mathbb{E}\left[(\boldsymbol{x}-\bar{\boldsymbol{x}})(\boldsymbol{x}-\bar{\boldsymbol{x}})^{\mathsf{H}}\right]. \tag{4.73}$$

In addition, we can define the *complementary covariance matrix* as [115, Section 2.2]

$$\tilde{\boldsymbol{C}}_{\boldsymbol{x}} \triangleq \mathbb{E}\left[(\boldsymbol{x}-\bar{\boldsymbol{x}})(\boldsymbol{x}-\bar{\boldsymbol{x}})^{\mathsf{T}}\right]. \tag{4.74}$$

Similarly, the cross-covariance and complementary cross-covariance matrices between two complex random vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are respectively defined as

$$\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{y}} \triangleq \mathbb{E}\left[(\boldsymbol{x}-\bar{\boldsymbol{x}})(\boldsymbol{y}-\bar{\boldsymbol{y}})^{\mathsf{H}}\right] \tag{4.75}$$

and

$$\tilde{\boldsymbol{C}}_{\boldsymbol{x}\boldsymbol{y}} \triangleq \mathbb{E}\left[(\boldsymbol{x}-\bar{\boldsymbol{x}})(\boldsymbol{y}-\bar{\boldsymbol{y}})^{\mathsf{T}}\right] \tag{4.76}$$

with $\bar{\boldsymbol{y}} \triangleq \mathbb{E}[\boldsymbol{y}]$. A random vector $\boldsymbol{x}$ is called *proper* if its complementary covariance matrix vanishes ($\tilde{\boldsymbol{C}}_{\boldsymbol{x}} = \boldsymbol{0}$); otherwise $\boldsymbol{x}$ is called *improper* [115, Definition 2.1]. Two random vectors are *cross-proper* if their complementary cross-covariance matrix vanishes.

Now let $\boldsymbol{y}$ be an observation, and $\boldsymbol{x}$ a signal that we want to estimate based on $\boldsymbol{y}$. As shown in [115, Result 5.4], the WLMMSE and LMMSE estimates of $\boldsymbol{x}$ are identical if and only if

$$\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{y}}\boldsymbol{C}_{\boldsymbol{y}}^{-1}\tilde{\boldsymbol{C}}_{\boldsymbol{y}} = \tilde{\boldsymbol{C}}_{\boldsymbol{x}\boldsymbol{y}}. \tag{4.77}$$

A sufficient condition for (4.77) to hold is that the observation is proper ($\tilde{\boldsymbol{C}}_{\boldsymbol{y}} = \boldsymbol{0}$), and that the signal and observation are cross-proper ($\tilde{\boldsymbol{C}}_{\boldsymbol{x}\boldsymbol{y}} = \boldsymbol{0}$).

Application to MIMO Detectors

For the linear channel model defined in (4.3)

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{w} \tag{4.78}$$

with $\boldsymbol{w} \sim \mathcal{CN}(\boldsymbol{0}_{N_{\mathrm{r}}}, N_0\boldsymbol{I}_{N_{\mathrm{r}}})$ and perfectly known $\boldsymbol{H}$, the complementary covariance matrix of the observation is

$$\tilde{\boldsymbol{C}}_{\boldsymbol{y}} = \mathbb{E}\left[\big(\boldsymbol{H}(\boldsymbol{x}-\bar{\boldsymbol{x}})+\boldsymbol{w}\big)\big(\boldsymbol{H}(\boldsymbol{x}-\bar{\boldsymbol{x}})+\boldsymbol{w}\big)^{\mathsf{T}}\right] = \boldsymbol{H}\tilde{\boldsymbol{C}}_{\boldsymbol{x}}\boldsymbol{H}^{\mathsf{T}} + \tilde{\boldsymbol{C}}_{\boldsymbol{w}} \tag{4.79}$$

and the complementary cross-covariance matrix between signal and observation is

$$\tilde{\boldsymbol{C}}_{\boldsymbol{xy}} = \mathbb{E}\left[(\boldsymbol{x}-\bar{\boldsymbol{x}})\big(\boldsymbol{H}(\boldsymbol{x}-\bar{\boldsymbol{x}})+\boldsymbol{w}\big)^{\mathsf{T}}\right] = \tilde{\boldsymbol{C}}_{\boldsymbol{x}}\boldsymbol{H}^{\mathsf{T}}. \tag{4.80}$$

The noise process is proper by assumption ($\tilde{\boldsymbol{C}}_{\boldsymbol{w}} = \boldsymbol{0}$). Thus, the WLMMSE estimate is identical to the strict LMMSE estimate if the signal is proper ($\tilde{\boldsymbol{C}}_{\boldsymbol{x}} = \boldsymbol{0}$). Assuming that the data symbols that belong to one symbol vector are statistically independent (which can be justified due to the bit interleaver in BICM systems), this reduces to the condition that all of the scalar data symbols are proper.

Decomposing the complex scalar x into its real and imaginary parts, $x = u + \mathrm{j}v$, the complementary variance of x can be written as

$$\begin{aligned}\mathbb{E}\left[(x-\bar{x})^2\right] &= \mathbb{E}\left[\big(u-\bar{u}+\mathrm{j}(v-\bar{v})\big)^2\right] \\ &= \mathbb{E}\left[(u-\bar{u})^2\right] - \mathbb{E}\left[(v-\bar{v})^2\right] + 2\mathrm{j}\,\mathbb{E}\left[(u-\bar{u})(v-\bar{v})\right] \\ &= \operatorname{var}[u] - \operatorname{var}[v] + 2\mathrm{j}\,\operatorname{cov}[u,v]\,. \end{aligned} \tag{4.81}$$

Thus, a complex random variable is proper if and only if its real and imaginary parts are uncorrelated and have the same variance.

This condition indeed holds for the usual modulation alphabets like QAM and PSK,[18] assuming that the symbols are uniformly distributed. For that reason, WLMMSE filters cannot provide a performance gain over LMMSE filters in conventional, non-iterative BICM receivers. However, as first noticed in [147], the situation changes once we consider iterative receivers.[19] Due to the information that is provided by the channel decoder, the data symbols are not uniformly distributed anymore, and they become improper in general. As a simple example, consider a QPSK system, where one of the two bits per data symbol determines its real part, and the other its imaginary part. Now assume that, due to the decoder feedback, the first bit is known with a high reliability (its L-value has a large magnitude), whereas the L-value of the second bit is close to 0. Then, $\operatorname{var}[\operatorname{Re} x] < \operatorname{var}[\operatorname{Im} x]$, and the symbol is improper.

The WLMMSE-Soft-PIC Detector

Clearly, the same situation arises in the doubly-iterative LMMSE-Soft-PIC detector, where we do not even need decoder feedback; the symbols already become improper after the first inner iteration. It is therefore an interesting question whether we can modify its EP-based derivation such that we obtain a *widely* linear MMSE detector

[18] A noteworthy exception are purely real-valued alphabets like BPSK. Since the imaginary part is always zero, it is *a priori* known to the receiver, and thus has a variance of zero.

[19] The main focus of [147] are signaling schemes where both x and its complex conjugate x^* are transmitted, like for example the well-known Alamouti scheme. However, the paper also mentions the applicability of widely linear filters to iterative receivers.

instead. And indeed, this is easily possible; the only required modification is that we project the data symbols into the family of *improper* instead of *proper* Gaussian distributions.[20] Formally, we replace the constraint (4.49) with

$$q(x_{n,i}) \stackrel{!}{=} \mathcal{EN}(x_{n,i} \,|\, \mu_{n,i}, \sigma_{n,i}^2, \tilde{\sigma}_{n,i}^2). \tag{4.82}$$

In order to simplify the derivation of the widely linear filter, it is advisable to work with *augmented* vectors and matrices as introduced on page 205. With the augmented quantities

$$\underline{\boldsymbol{x}}_n \triangleq \begin{pmatrix} \boldsymbol{x}_n \\ \boldsymbol{x}_n^* \end{pmatrix}, \quad \underline{\boldsymbol{\mu}}_n \triangleq \begin{pmatrix} \boldsymbol{\mu}_n \\ \boldsymbol{\mu}_n^* \end{pmatrix}, \quad \underline{\boldsymbol{\Sigma}}_n \triangleq \begin{pmatrix} \boldsymbol{\Sigma}_n & \tilde{\boldsymbol{\Sigma}}_n \\ \tilde{\boldsymbol{\Sigma}}_n^* & \boldsymbol{\Sigma}_n^* \end{pmatrix} \tag{4.83}$$

the PDF of the N_t-dimensional improper Gaussian vector $\boldsymbol{x}_n$ takes on the form

$$q(\boldsymbol{x}_n) = \frac{\exp\left(-\frac{1}{2}(\underline{\boldsymbol{x}}_n - \underline{\boldsymbol{\mu}}_n)^\mathsf{H} \underline{\boldsymbol{\Sigma}}_n^{-1} (\underline{\boldsymbol{x}}_n - \underline{\boldsymbol{\mu}}_n)\right)}{\pi^{N_\mathrm{t}} \sqrt{\det \underline{\boldsymbol{\Sigma}}_n}} \tag{4.84}$$

as derived in (A.122). The marginal distribution of $x_{n,i}$ can similarly be expressed using

$$\underline{x}_{n,i} \triangleq \begin{pmatrix} x_{n,i} \\ x_{n,i}^* \end{pmatrix}, \quad \underline{\mu}_{n,i} \triangleq \begin{pmatrix} \mu_{n,i} \\ \mu_{n,i}^* \end{pmatrix}, \quad \underline{\Sigma}_{n,i} \triangleq \begin{pmatrix} \sigma_{n,i}^2 & \tilde{\sigma}_{n,i}^2 \\ \tilde{\sigma}_{n,i}^{*2} & \sigma_{n,i}^2 \end{pmatrix} \stackrel{(a)}{=} \sigma_{n,i}^2 \begin{pmatrix} 1 & \rho_{n,i} \\ \rho_{n,i}^* & 1 \end{pmatrix} \tag{4.85}$$

as

$$\begin{aligned} q(x_{n,i}) &= \frac{\exp\left(-\frac{1}{2}(\underline{x}_{n,i} - \underline{\mu}_{n,i})^\mathsf{H} \underline{\Sigma}_{n,i}^{-1} (\underline{x}_{n,i} - \underline{\mu}_{n,i})\right)}{\pi \sqrt{\det \underline{\Sigma}_{n,i}}} \\ &= \frac{1}{\pi \sigma_{n,i}^2 \sqrt{1 - |\rho_{n,i}|^2}} \exp\left(-\frac{|x_{n,i} - \mu_{n,i}|^2 - \mathrm{Re}\left(\rho_{n,i}^* (x_{n,i} - \mu_{n,i})^2\right)}{\sigma_{n,i}^2 \left(1 - |\rho_{n,i}|^2\right)}\right). \end{aligned} \tag{4.86}$$

In (4.85), (a) introduces the *complex correlation coefficient*

$$\rho \triangleq \frac{\tilde{\sigma}^2}{\sigma^2} \tag{4.87}$$

with $|\rho| \leq 1$. For $\rho = 0$, the Gaussian distribution is proper, while for $|\rho| = 1$, it is maximally improper, meaning that the support of the distribution degenerates to a line.

With the notion of using augmented vectors and matrices, the derivation of the WLMMSE-Soft-PIC detector becomes very similar to that of its strictly linear counterpart. We will therefore only sketch it very briefly. Analogously to (4.50), it starts with

$$m_{\mathrm{equ}\to\mathrm{dem}}(\boldsymbol{x}_n) = \frac{\mathrm{proj}_{\mathcal{EN}}\left[Z^{-1}\, p(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n)\, m_{\mathrm{dem}\to\mathrm{equ}}(\boldsymbol{x}_n)\right]}{m_{\mathrm{dem}\to\mathrm{equ}}(\boldsymbol{x}_n)} \tag{4.88}$$

[20] The improper Gaussian distribution is introduced on page 223 in the appendix. We denote it as $\mathcal{EN}$, which is supposed to mean *elliptical*, in contrast to the *circular* Gaussian distribution $\mathcal{CN}$.

but this time, the incoming message from the demapper node is an improper Gaussian

$$m_{\text{dem}\to\text{equ}}(\boldsymbol{x}_n) \propto \mathcal{EN}(\boldsymbol{x}_n \,|\, \overleftarrow{\boldsymbol{\mu}}_n, \overleftarrow{\boldsymbol{\Sigma}}_n, \overleftarrow{\tilde{\boldsymbol{\Sigma}}}_n) \tag{4.89}$$

where $\overleftarrow{\boldsymbol{\Sigma}}_n = \operatorname{diag}\big(\overleftarrow{\sigma}^2_{n,0}, \ldots, \overleftarrow{\sigma}^2_{n,N_\text{t}-1}\big)$ and $\overleftarrow{\tilde{\boldsymbol{\Sigma}}}_n = \operatorname{diag}\big(\overleftarrow{\tilde{\sigma}}^2_{n,0}, \ldots, \overleftarrow{\tilde{\sigma}}^2_{n,N_\text{t}-1}\big)$. With the augmented matrices

$$\underline{\boldsymbol{H}}_n = \begin{pmatrix} \boldsymbol{H}_n & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H}_n^* \end{pmatrix}, \quad \underline{\overleftarrow{\boldsymbol{\Sigma}}}_n = \begin{pmatrix} \overleftarrow{\boldsymbol{\Sigma}}_n & \overleftarrow{\tilde{\boldsymbol{\Sigma}}}_n \\ \overleftarrow{\tilde{\boldsymbol{\Sigma}}}_n^* & \overleftarrow{\boldsymbol{\Sigma}}_n^* \end{pmatrix} \quad \text{and} \quad \underline{\boldsymbol{C}}_{\underline{\boldsymbol{y}}_n} = \underline{\boldsymbol{H}}_n \underline{\overleftarrow{\boldsymbol{\Sigma}}}_n \underline{\boldsymbol{H}}_n^\mathsf{H} + N_0 \boldsymbol{I}_{2N_\text{r}} \tag{4.90}$$

the argument of the projection operator is the improper Gaussian $\mathcal{EN}(\underline{\boldsymbol{\mu}}_n, \underline{\boldsymbol{\Sigma}}_n)$ with augmented mean

$$\underline{\boldsymbol{\mu}}_n = \underline{\overleftarrow{\boldsymbol{\mu}}}_n + \underline{\overleftarrow{\boldsymbol{\Sigma}}}_n \underline{\boldsymbol{H}}_n^\mathsf{H} \underline{\boldsymbol{C}}_{\underline{\boldsymbol{y}}_n}^{-1} (\underline{\boldsymbol{y}}_n - \underline{\boldsymbol{H}}_n \underline{\overleftarrow{\boldsymbol{\mu}}}_n) \tag{4.91}$$

and augmented covariance matrix

$$\underline{\boldsymbol{\Sigma}}_n = \underline{\overleftarrow{\boldsymbol{\Sigma}}}_n - \underline{\overleftarrow{\boldsymbol{\Sigma}}}_n \underline{\boldsymbol{H}}_n^\mathsf{H} \underline{\boldsymbol{C}}_{\underline{\boldsymbol{y}}_n}^{-1} \underline{\boldsymbol{H}}_n \underline{\overleftarrow{\boldsymbol{\Sigma}}}_n. \tag{4.92}$$

Note that these equations are essentially the same as (4.52) and (4.53), the only difference is that now all vectors and matrices are augmented.

The projection of $\mathcal{EN}(\underline{\boldsymbol{\mu}}_n, \underline{\boldsymbol{\Sigma}}_n)$ into the set of fully factorized improper Gaussians is the product of its marginals, and the parameters $\underline{\mu}_{n,i}$ and $\underline{\Sigma}_{n,i}$ of the ith marginal are

$$\underline{\mu}_{n,i} = \begin{pmatrix} \underline{\boldsymbol{\mu}}_n(i) \\ \underline{\boldsymbol{\mu}}_n(N_\text{t}+i) \end{pmatrix} \quad \text{and} \quad \underline{\Sigma}_{n,i} = \begin{pmatrix} \underline{\boldsymbol{\Sigma}}_n(i,i) & \underline{\boldsymbol{\Sigma}}_n(i,N_\text{t}+i) \\ \underline{\boldsymbol{\Sigma}}_n(N_\text{t}+i,i) & \underline{\boldsymbol{\Sigma}}_n(N_\text{t}+i,N_\text{t}+i) \end{pmatrix} \tag{4.93}$$

where $\boldsymbol{a}(i)$ returns the ith coefficient of the vector $\boldsymbol{a}$, and $\boldsymbol{A}(i,j)$ the (i,j)th coefficient the matrix $\boldsymbol{A}$.

Next, the forward messages $m_{\text{equ}\to\text{dem}}(x_{n,i})$ are computed by dividing the belief $q(x_{n,i})$ by $m_{\text{dem}\to\text{equ}}(x_{n,i})$. The natural parameters of the improper Gaussian distribution are the inverse augmented covariance matrix $\underline{\Sigma}^{-1}$ and the weighted mean $\underline{\Sigma}^{-1}\underline{\mu}$, so the parameters of the forward message $m_{\text{equ}\to\text{dem}}(x_{n,i}) \propto \mathcal{EN}(x_{n,i} \,|\, \underline{\vec{\mu}}_{n,i}, \underline{\vec{\Sigma}}_{n,i})$ are given as

$$\underline{\vec{\mu}}_{n,i} = \underline{\vec{\Sigma}}_{n,i} \left(\underline{\Sigma}_{n,i}^{-1} \underline{\mu}_{n,i} - \underline{\overleftarrow{\Sigma}}_{n,i}^{-1} \underline{\overleftarrow{\mu}}_{n,i} \right) \quad \text{and} \quad \underline{\vec{\Sigma}}_{n,i} = \left(\underline{\Sigma}_{n,i}^{-1} - \underline{\overleftarrow{\Sigma}}_{n,i}^{-1} \right)^{-1}. \tag{4.94}$$

These are finally used by the demapper node to compute either new $\underline{\overleftarrow{\mu}}_{n,i}$, $\overleftarrow{\sigma}^2_{n,i}$ and $\overleftarrow{\tilde{\sigma}}^2_{n,i}$, using essentially (4.62)–(4.66), or new L-values for the decoder, using essentially (4.67), which completes the inner iteration. The main difference is that the quadratic term $-\frac{|x_{n,i}-\overrightarrow{\mu}_{n,i}|^2}{\overrightarrow{\sigma}^2_{n,i}}$ in (4.62) and (4.67) is replaced by $-\frac{|x_{n,i}-\overrightarrow{\mu}_{n,i}|^2 - \operatorname{Re}\left(\overrightarrow{p}^*_{n,i}(x_{n,i}-\overrightarrow{\mu}_{n,i})^2\right)}{\overrightarrow{\sigma}^2_{n,i}\left(1-|\overrightarrow{p}_{n,i}|^2\right)}$ as in (4.86).

Note that the above presented equations should not be used verbatim for an implementation of the WLMMSE detector. While augmented vectors and matrices lead to elegant mathematical expressions, they also contain redundancy since their lower halves are just the complex conjugates of their upper halves. Besides avoiding redundant computations, it is furthermore possible to eliminate the explicit computation of the parameters in (4.93) and the subsequent division (4.94) by merging both steps, similar to (4.58)–(4.59). Not only does this modification reduce the number of operations, but it also improves the numerical stability of the algorithm, since the inversions of the augmented covariance matrices in (4.94) are not required anymore. This is important since those matrices can become close to singular, for instance when the information about $\mathrm{Re}\, x$ is much more reliable than that of $\mathrm{Im}\, x$, in which case the shape of the distribution of x would degenerate to a thin vertical ellipse.

Despite their practical importance, however, implementation aspects are not in the focus of this thesis. We will therefore not explore this issue any further.

4.3.4 Numerical Results

We proceed with a numerical performance evaluation of the proposed MIMO detector. As earlier in this chapter, the following simulation results are based on a 4×4 MIMO system with a Gray-mapped 16-QAM modulation.

Impact of the Number of Inner Iterations

As mentioned before, the LMMSE-Soft-PIC detector is itself iterative; it exchanges information between the spatial equalizer and the streamwise demappers. In conjunction with a BICM-ID receiver, the total algorithm thus becomes doubly iterative. In the section on scheduling, I proposed to perform two inner iterations during the initial outer iteration. To justify this proposal, this section starts with a performance comparison of LMMSE-Soft-PIC detectors doing 1, 2, and 3 inner iterations. There is no feedback from the decoder, so the simulations correspond to non-iterative BICM receivers, or to the initial iteration in BICM-ID receivers.

Figure 4.12 shows uncoded BER results for both the LMMSE and WLMMSE versions of the detector. As expected, LMMSE and WLMMSE show exactly the same performance after the first inner iteration, because without feedback, the symbols are proper and the WLMMSE filter becomes strictly linear. In fact, with a single inner iteration, both detectors are identical to the conventional LMMSE detector.

The second inner iteration leads to a significantly better performance of the proposed algorithm, compared to the conventional detector. At an uncoded BER of 10^{-2}, the LMMSE receiver performs about 5 dB, and its widely linear version even 5.7 dB better than the non-iterative detectors. And, as one would intuitively expect, a third iteration yields a further improvement, albeit by much less than the second one.

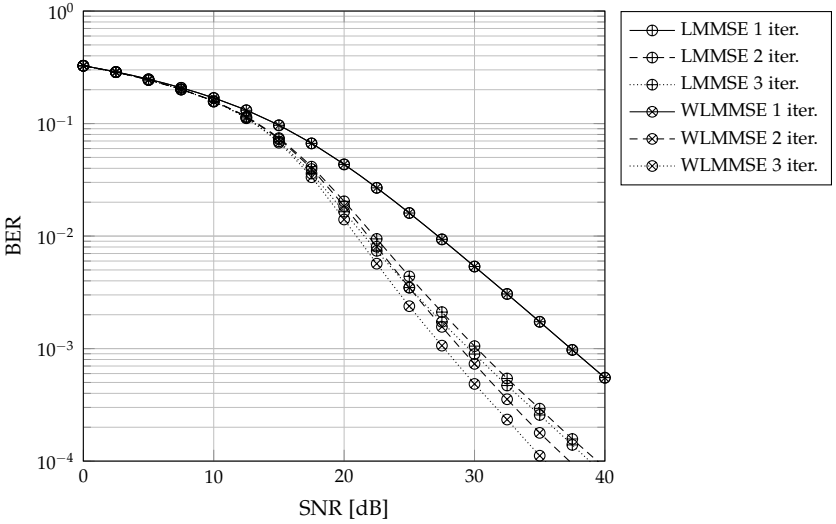


Figure 4.12: Uncoded bit error rates of a 4×4 MIMO system with a Gray-mapped 16-QAM modulation, using the (W)LMMSE-Soft-PIC detector with 1, 2, and 3 inner iterations

However, as we have seen in Section 4.2.5, uncoded error rates can be misleading. Figure 4.13 therefore present a similar study, but using an LDPC code of rate $1/2$ and a length of 1944 code bits. This time, the results are more interesting: While the second inner iteration still provides a considerable gain over the non-iterative detector, the third iteration actually *decreases* the performance.

What is the reason for this unexpected behaviour? We have seen that the uncoded BER, which only depends on the *sign* of the L-values, does profit from a third inner iteration. Therefore, the third iteration must have a negative impact on the *magnitude* of the L-values. In order to investigate this issue, Figure 4.14 shows scatter diagrams of L-values that have been obtained by a Monte Carlo simulation. Each dot corresponds to one transmitted bit; the horizontal coordinate is equal to its exact L-value (4.16), while the vertical coordinate corresponds to the output of the LMMSE-Soft-PIC detector. Thus, dots that are concentrated around the main diagonal of the diagram indicate an accurately working LMMSE detector. The transmitted bits are iid, drawn from $\mathbb{F} = \{0, 1\}$, but the sign of the L-values that belong to $c = 0$ have been flipped, so that a positive L-value always correspond to a correct, and a negative L-value to a wrong hard decision.

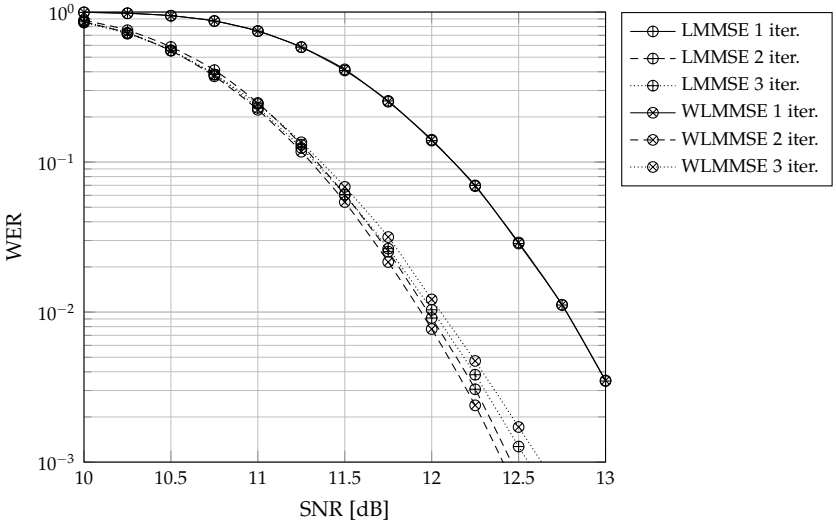


Figure 4.13: Word error rates of an LDPC-coded 4×4 MIMO system, using the (W)LMMSE-Soft-PIC detector with 1, 2, and 3 inner iterations

Low-SNR Behaviour The figure shows nine subplots: the three rows correspond to SNR values of 0, 10, and 20 dB, while the different columns show the results after 1, 2, and 3 inner iterations (in particular, the left column corresponds to the conventional LMMSE detector). From the top row, we see that the LMMSE detector works very well in low SNR, as the L-values are tightly concentrated along the main diagonal. This is already true for the conventional LMMSE detector, and even more so for the Soft-PIC detector after two or three inner iterations.

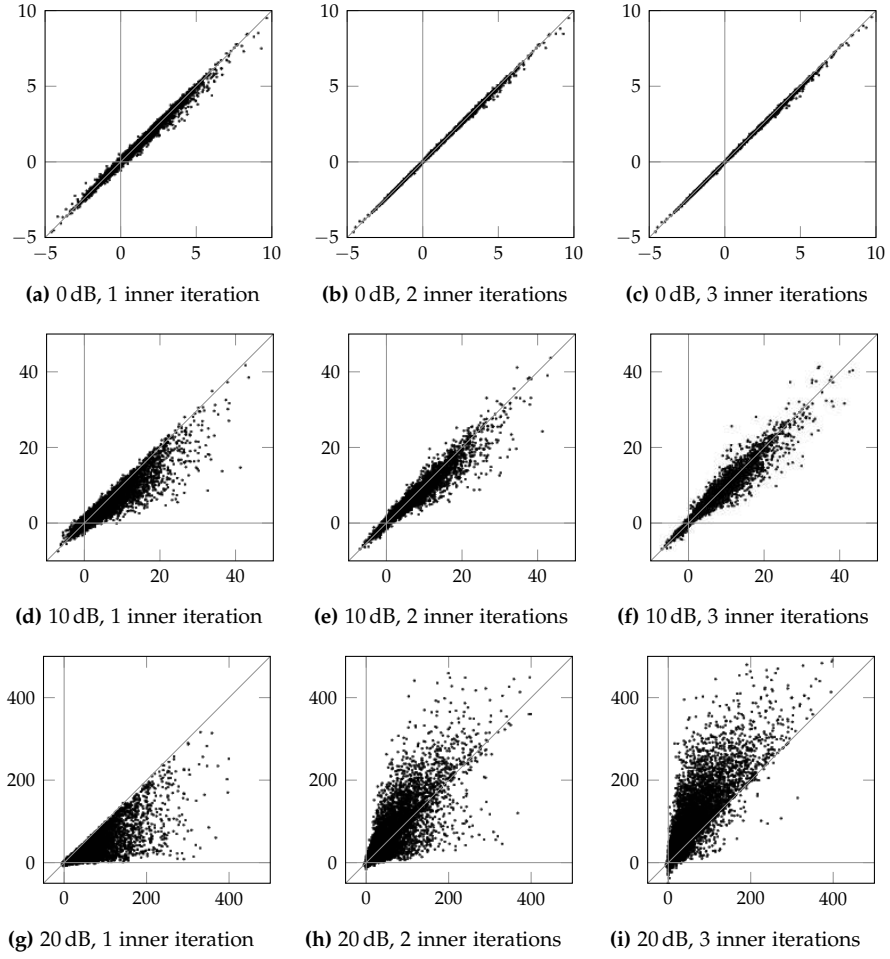
In order to understand the good low-SNR performance, it is instructive to consider the formulation introduced in (4.26) and (4.27)

$$\boldsymbol{y} = \boldsymbol{h}_i x_i + \boldsymbol{v}_i \tag{4.95}$$

which focuses on the detection of the ith data symbol, and merges the interference from all other signals and the noise into an interference-plus-noise term

$$\boldsymbol{v}_i \triangleq \boldsymbol{H}_{\setminus i} \boldsymbol{x}_{\setminus i} + \boldsymbol{w}. \tag{4.96}$$

The vector $\boldsymbol{v}_i$ is the sum of $N_\mathrm{t} - 1$ discrete and one Gaussian random vector, which are mutually independent. Its density is thus equal to the convolution of the individual densities, namely a Gaussian mixture. Now, the only approximation that is introduced by the LMMSE detector is that it treats $\boldsymbol{v}_i$ as a Gaussian random vector with the



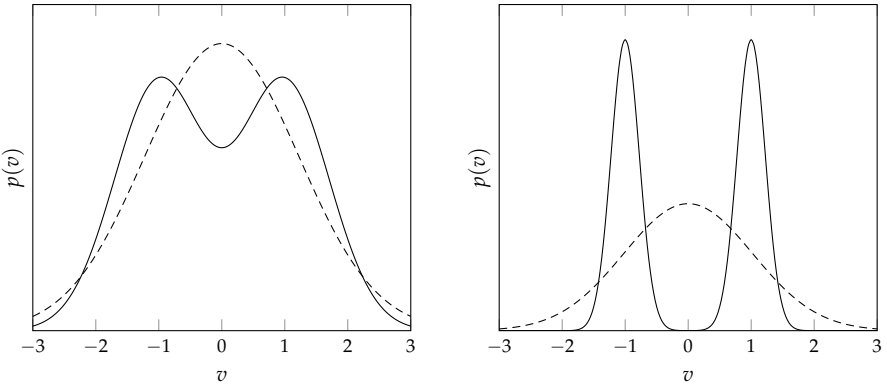
(a) 0 dB, 1 inner iteration **(b)** 0 dB, 2 inner iterations **(c)** 0 dB, 3 inner iterations

(d) 10 dB, 1 inner iteration **(e)** 10 dB, 2 inner iterations **(f)** 10 dB, 3 inner iterations

(g) 20 dB, 1 inner iteration **(h)** 20 dB, 2 inner iterations **(i)** 20 dB, 3 inner iterations

Figure 4.14: Scatter diagrams of L-values in a 4×4 MIMO system with 16-QAM modulation and two different MIMO detectors: an exact detector (horizontal axis) and an LMMSE-Soft-PIC detector (vertical axis) with 1, 2, and 3 inner iterations

same mean and covariance; if v_i were truly proper (improper) Gaussian, the LMMSE (WLMMSE) detector would be exact. Intuitively, this approximation is quite accurate in low SNR, where the individual components of the Gaussian mixture are so wide that they essentially merge into one mode. In high SNR, on the other hand, the individual



(a) SNR = 0 dB **(b)** SNR = 10 dB

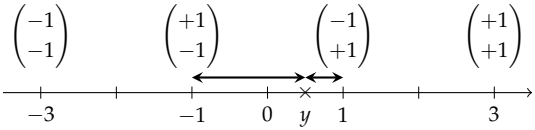
Figure 4.15: Probability density function of the interference-plus-noise term $v = x + w$, assuming a BPSK-modulated interferer $x \in \{-1, +1\}$ and $w \sim \mathcal{N}(0, N_0/2)$, and its Gaussian approximation (dashed)

components hardly overlap, so that approximating the mixture distribution with a single Gaussian necessarily leads to a severe error. This is visualized in Figure 4.15, which depicts the distribution of a scalar interference-plus-noise term, assuming one BPSK-modulated interferer, for SNR values of 0 and 10 dB. The dashed lines in the same figure show the corresponding Gaussian approximation. It should be obvious that the left-hand distribution (SNR = 0 dB) can be much better approximated with a Gaussian than the right-hand one (SNR = 10 dB).

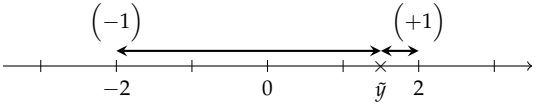
To summarize, even the non-iterative LMMSE detector is fairly accurate in low SNR, so that we can only expect a small gain by doing more than one inner iteration.

High-SNR Behaviour The three scatter diagrams in the bottom row of Figure 4.14 (SNR = 20 dB) look entirely different. In the leftmost plot, practically all dots are below the main diagonal, which means that the conventional LMMSE detector underestimates the magnitudes of the L-values. This can be explained as follows. It is well known that the Gaussian distribution has the largest entropy among all distributions with a fixed covariance matrix (see Appendix A.2). Thus, replacing $\boldsymbol{v}_i$ by a Gaussian with the same covariance necessarily increases the entropy of the effective noise term, which causes the detector to output conservative soft decisions (i. e., L-values with underestimated magnitudes).

After two iterations (middle column), some L-values are below and some above the main diagonal, and after the third inner iteration (right column), the characteristic of the



(a) Without prior information about the data symbols



(b) After interference cancellation with $x_0 = -1$

Figure 4.16: Received signal space of a 2×1 MISO system with BPSK modulation, a transmitted signal of $\boldsymbol{x} = (-1,\ +1)^{\mathsf{T}}$, a channel of $\boldsymbol{H} = (1,\ 2)$, and a received signal of $y = \boldsymbol{H}\boldsymbol{x} + w = 0.5$

results is reversed: now, almost all L-values are overestimated, which explains that the WER performance after just two iterations is actually better than after three iterations.

The overestimation of the L-values is (somewhat paradoxically) caused by the virtually perfect prior information about the spatial interference that is available to the detector in the third iteration. To understand this phenomenon, consider a simple example which is visualized in Figure 4.16. We have a 2×1 MISO system, where both streams are BPSK-modulated. Assume that the symbol vector $(x_0,\ x_1)^{\mathsf{T}} = (-1,\ +1)^{\mathsf{T}}$ has been transmitted, that the channel matrix is $\boldsymbol{H} = (1,\ 2)$, and that the observation is $y = h_0 x_0 + h_1 x_1 + w = 0.5$. Furthermore, assume a noise power spectral density of $N_0 = 1$ for simplicity.

Now, we want to compute the L-value for x_1 using the max-log approximation. Without any prior information, the four hypotheses for the noise-free received signal $(-3, -1, 1, 3)$ are equally probable. We search for the two hypotheses (one with $x_1 = -1$, one with $x_1 = +1$) that are closest to $y = 0.5$, and find them to be -1 and 1, respectively, as indicated with the arrows in Figure 4.16a. The L-value is thus

$$\lambda_1 = (y - (-1))^2 - (y - (+1))^2 \tag{4.97}$$

$$= 1.5^2 - (-0.5)^2 \tag{4.98}$$

$$= 2. \tag{4.99}$$

Now assume that we know from previous iterations that $x_0 = -1$. We subtract the interference from the observation and obtain $\tilde{y} \triangleq y - h_0 x_0 = 1.5$. As indicated in Figure 4.16b, the only remaining receive hypotheses are -2 and 2, and computing the L-value from the interference-free observation yields

$$\tilde{\lambda}_1 = (\tilde{y} - (-2))^2 - (\tilde{y} - (+2))^2 \tag{4.100}$$

$$= 3.5^2 - (-0.5)^2 \tag{4.101}$$
$$= 12. \tag{4.102}$$

Thus, the removal of the interference has led to a severe overestimation of the L-value.

Methods for correcting such erroneously computed L-values (at least to some extend) are an active field of research. While working on this thesis, I have tried to apply the technique proposed in [126] which is based on a saddlepoint approximation of the L-value distribution,[21] but unfortunately without success. However, this study at least explains the surprising fact that a third iteration within the LMMSE-Soft-PIC detector causes a performance degradation.

In the remainder of this section, the LMMSE-Soft-PIC detector will be operated according to the schedule described on page 155, i. e., with two inner iterations in the first outer iteration, and a single inner iteration in all subsequent outer iterations.

Maximum Data Rates

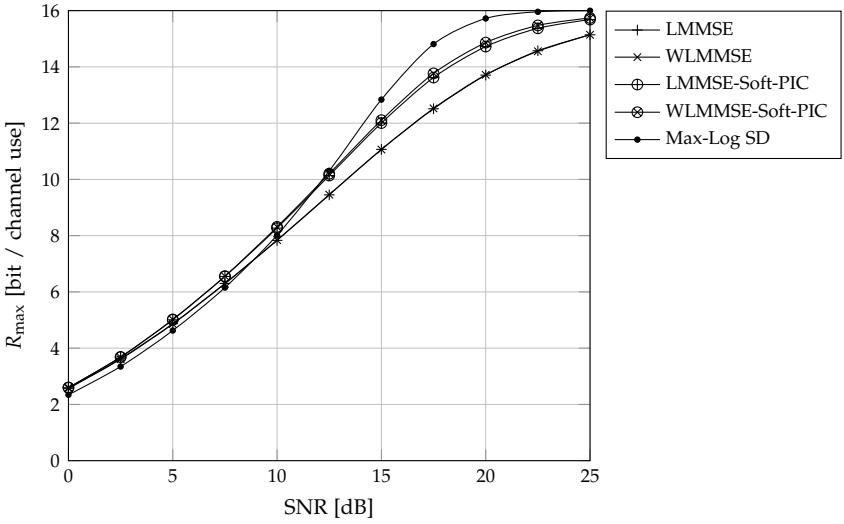
We proceed with an evaluation of the maximal achievable rates (see Section 4.1.4) of the conventional LMMSE detector and the proposed LMMSE-Soft-PIC detector, along with their WLMMSE versions. As a performance benchmark, we also include the max-log detector (4.17) in the comparison, implemented via the sphere detection (SD) algorithm.

The results are presented in Figure 4.17, which is split into two subfigures to improve the readability: the upper plot shows the achievable data rates of a non-iterative receiver,[22] while the lower plot contains the results for a BICM-ID system.

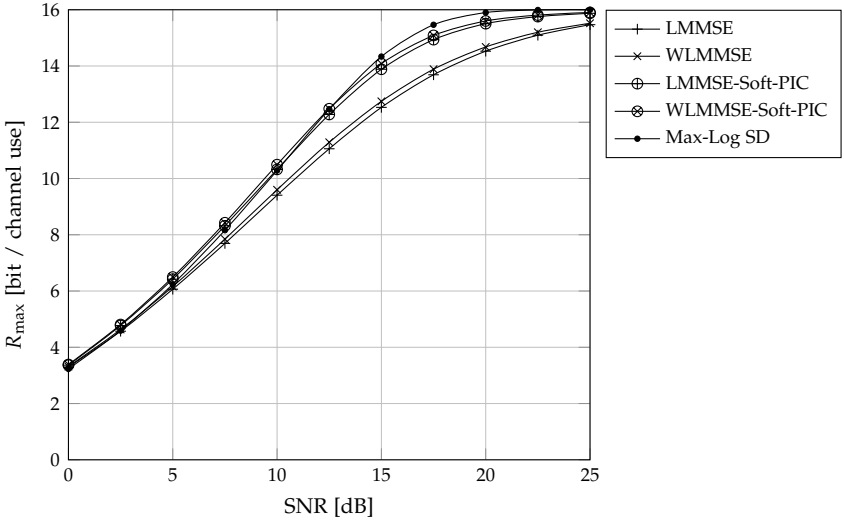
First consider the non-iterative case. In the low SNR range, there is hardly any difference between LMMSE and LMMSE-Soft-PIC, as we would have expected after discussing Figure 4.14. But with increasing SNR, the gain that is provided by the inner iteration within the LMMSE-Soft-PIC becomes quite significant. At a data rate of 8 bit / channel use (bpcu), corresponding to a code rate of $1/2$, the gap between the two receivers is around 0.7 dB, and at 12 bpcu, it already exceeds 1.5 dB. The max-log detector performs worse than both LMMSE-based detectors in the low SNR region, where the max-log approximation is particularly inaccurate. In high SNR, however, the approximation becomes very tight, and max-log SD converges to the exact MIMO detector. While both LMMSE detectors are clearly suboptimal in high SNR, the LMMSE-Soft-PIC at least reduces the gap to the max-log SD by more than 50% (in terms of dB), compared to the conventional LMMSE detector.

[21] In collaborative work with the INRS Montreal, Canada, presented in [121], we have shown that the saddlepoint approximation can also be used to derive a MIMO receiver which resembles the LMMSE detector, but yields more accurate results because it approximates the distribution of the interference-plus-noise term (c. f. Figure 4.15) with a non-Gaussian and in general multi-modal distribution. However, due to space constraints, we will not investigate this approach any further in this thesis.

[22] In the sense that there are no iterations between MIMO detector and decoder. The (W)LMMSE-Soft-PIC detector still does two internal iterations.



(a) Non-iterative BICM receivers



(b) Iterative BICM-ID receivers

Figure 4.17: Upper bounds on the achievable data rates of a 4×4 MIMO system with a Gray-mapped 16-QAM modulation, comparing the conventional and the proposed iterative (W)LMMSE detectors

As expected, there is no difference between the conventional LMMSE and WLMMSE detectors in non-iterative receivers, because without feedback, the data symbols are proper and both filters are identical. In case of the Soft-PIC detectors, there is still hardly any difference in the low SNR range, but in high SNR, we can see a small gap of around 0.2–0.3 dB. The reason for this behaviour is that only the spatial interference can become improper, while the noise $\boldsymbol{w}$ always remains proper. Thus, the higher the SNR, the "more improper" can the interference-plus-noise term (4.96) get in the presence of feedback, which increases the potential benefit of widely linear filters.

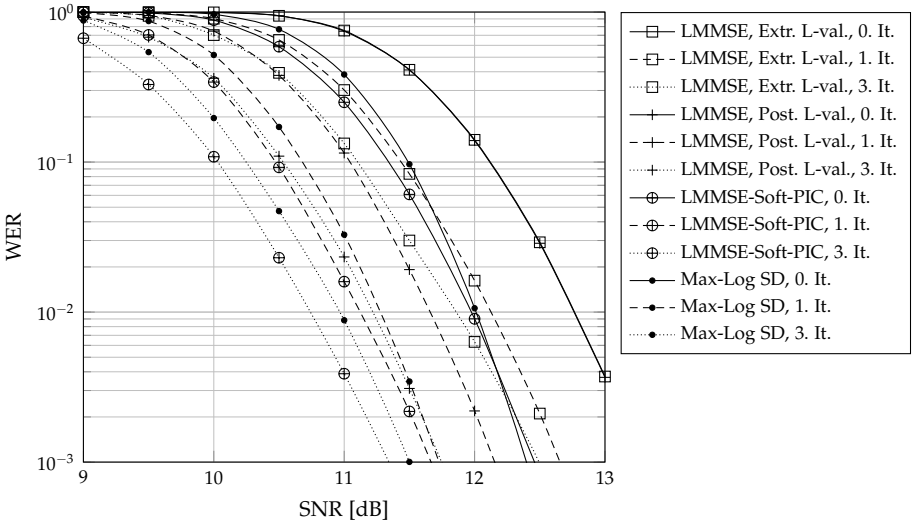
In iterative BICM-ID receivers (Figure 4.17b), the advantage of LMMSE-Soft-PIC over the conventional LMMSE detector is even more pronounced. Up to a spectral efficiency of around 12 bpcu, its performance is close to that of the max-log detector, while the conventional LMMSE is already more than 2 dB worse at that rate. Only in very high SNR, there remains some gap between LMMSE-Soft-PIC and max-log SD, and it can be argued that this high SNR range, where the $R_{\max}$ curves start to flatten out and approach their limit of $N_{\mathrm{t}} \log_2 M$, is rather uninteresting from a practical point of view. The reason is that modern systems adapt their modulation and coding scheme to the channel conditions, and would switch to a larger constellation like 64-QAM as soon as the current modulation alphabet became a performance bottleneck.

LDPC-Coded Word Error Rates

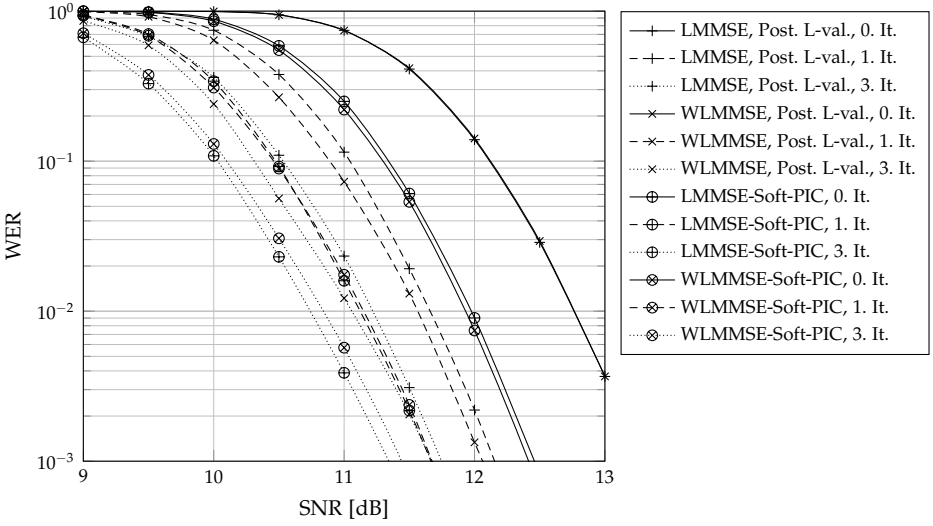
In order to verify the soundness of the information theoretic study, we close this section with some word error rate plots.

Medium SNR The results reported in Figure 4.18 have been obtained using the LDPC code from the IEEE 802.11n standard, with a block length of 1944 code bits and code rate $r = 1/2$. The upper subfigure compares the conventional LMMSE with extrinsic and with posterior decoder feedback, the proposed LMMSE-Soft-PIC, and as a reference the max-log sphere detector. For each algorithm, word error rates are shown after the initial pass through the receiver (the "zeroth" iteration, corresponding to a non-iterative BICM receiver), and after one and three BICM-ID iterations.

It can clearly be seen that the LMMSE detector with posterior feedback performs significantly better than with extrinsic feedback. This verifies the observation from [148] which has been the initial motivation for the work presented in this section. After just one iteration, the receiver which uses posterior feedback yields better results than the version with extrinsic feedback after three iterations. However, it is equally clearly outperformed by the LMMSE-Soft-PIC: it, too, yields a lower WER after a single iteration than the conventional LMMSE (with posterior feedback) after three iterations. Comparing both algorithms after three iterations, the gap is around 0.5 dB at a WER of 10^{-2}. The LMMSE-Soft-PIC detector even outperforms our reference algorithm, the max-log sphere detector, by just under 0.2 dB at a WER of 10^{-2}.



(a) Comparison of the conventional LMMSE detector (with extrinsic and posterior feedback), the LMMSE-Soft-PIC detector, and the max-log SD



(b) Comparison of strictly and widely linear MMSE detectors

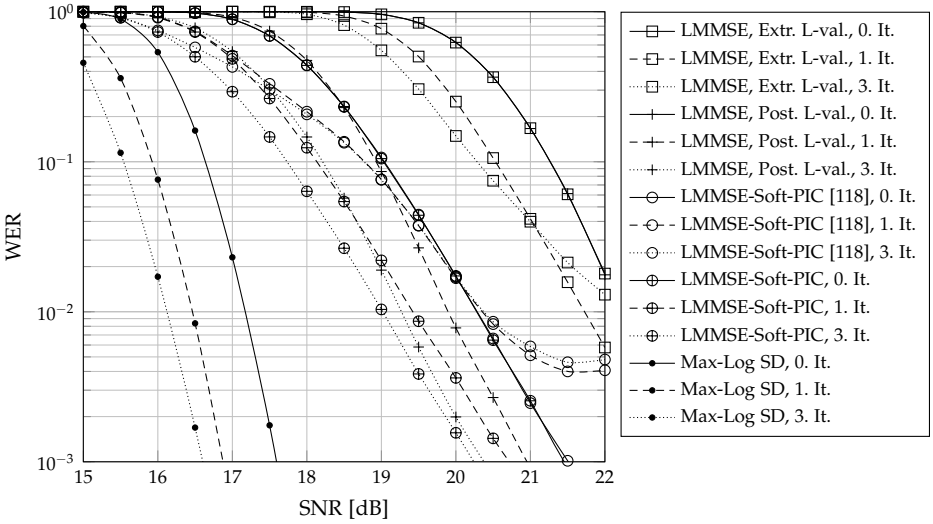
Figure 4.18: Word error rates of an LDPC-coded 4×4 MIMO system with a Gray-mapped 16-QAM modulation, using a rate-1/2 code

The lower subfigure compares both the conventional LMMSE and the LMMSE-Soft-PIC with their widely linear variants. Used in the conventional detector, the widely linear filter offers a non-negligible performance gain of around 0.17 dB. In the case of the Soft-PIC detector, however, the difference is much smaller and amounts to a mere 0.05 dB in non-iterative receivers, which makes the additional complexity of the WLMMSE filter hardly worthwile. In iterative receivers, the stricly linear filter even performs slightly better than the widely linear version. There is of course no fundamental reason for this behaviour—theoretically, the WLMMSE receiver should never yield worse results than LMMSE. However, as briefly mentioned above, the WLMMSE filter can run into numerical problems when the complex correlation coefficient approaches the unit circle, i. e., when the symbol distributions become maximally improper. Because of the rather small performance difference between the strictly and widely linear MMSE detectors, I have not invested much effort into improving the numerical stability of the algorithm; it is well possible that better implementations of the WLMMSE filter can be found. Still, quantization errors are clearly an important issue that must be investigated carefully when designing a VLSI implementation, especially when the data is represented with fixed point numbers (all results that are shown in this thesis have been computed with double precision floating point numbers).

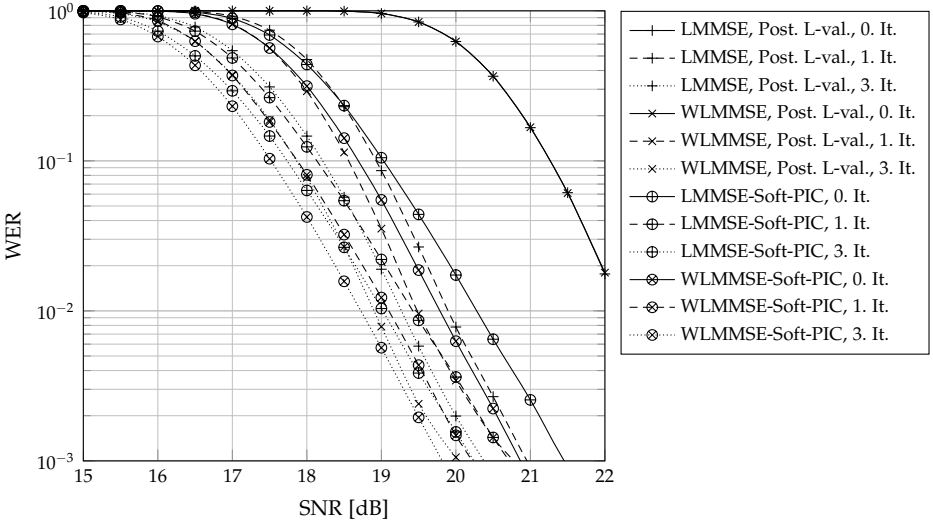
High SNR The last Figure 4.19 shows the results of essentially the same simulation, but with a code rate of $r = 5/6$, in order to evaluate the MIMO detectors in high SNR.

The upper subplot, which again compares extrinsic and posterior feedback in conjunction with the conventional LMMSE detector, shows an interesting effect: the WER curves of the receiver with extrinsic feedback after one and after three iterations intersect at around 21 dB; in higher SNR, doing more than one iteration actually degrades the performance. This result shows once more that using a plain LMMSE receiver with extrinsic feedback is fundamentally flawed. A similar phenomenon can also be observed with the LMMSE-Soft-PIC detector, when it operates according to the schedule that I have described in [118] (the detector is always run for two inner iterations, and the internal parameters $\vec{\mu}$ and $\vec{\sigma}^2$ are not stored between the outer iterations). In the medium SNR range, this schedule yields essentially the same results as the one proposed in this work (storing the internal parameters, and doing just a single inner iteration in all but the initial outer iteration), which is why the corresponding results have not been shown in the previous Figure 4.18. In high SNR, however, using the schedule from [118] results in an error floor after ≈ 20 dB, which is avoided by the schedule proposed in this work. (In some sense, this is also the "more correct" schedule according to the expectation propagation algorithm. Resetting the internal detector messages is just an *ad hoc* trick to save some memory; it is not theoretically backed up by the EP equations.)

However, while the proposed LMMSE-Soft-PIC detector performs best among the LMMSE-based algorithms, it should be acknowledged that all of them are clearly



(a) Comparison of the conventional LMMSE detector (with extrinsic and posterior feedback), the LMMSE-Soft-PIC detector, and the max-log SD



(b) Comparison of strictly and widely linear MMSE detectors

Figure 4.19: Word error rates of an LDPC-coded 4×4 MIMO system with a Gray-mapped 16-QAM modulation, using a rate-5/6 code

outperformed by the max-log SD. This should not come as a surprise, since this result is in agreement with the information theoretic study from Figure 4.17.

The lower subplot of Figure 4.19 again compares the strictly and widely linear receivers. This time, the results of the WLMMSE detector are around 0.3 dB better than those of LMMSE (after three iterations and at a WER of 10^{-2}). Obviously, the potential performance advantage of widely linear receivers is large enough in this high SNR to overcompensate any numerical issues. Nonetheless, the practical relevance of the WLMMSE-Soft-PIC detector stays questionable in my opinion. A performance gain of around 0.3 dB is certainly not negligible, but not that significant, either. And even this moderate gain is only present in a rather unrealistically high SNR range, where the max-log sphere detector (or a variant thereof) would be a superior option, anyway.

4.3.5 Concluding Remarks

In this section, we have systematically derived a new low-complexity MIMO detector as an instance of the expectation propagation framework. The proposed algorithm is very similar to the well-known LMMSE-based MIMO detector, but it achieves a significant performance gain by exchanging information internally between the spatial equalizer and the streamwise demappers, in a fashion that is conceptually closely related to the exchange of L-values between demapper and decoder in BICM-ID systems. Due to this modification, the proposed detector is able to exploit its knowledge about the discrete modulation of the interfering signals; the conventional non-iterative LMMSE detector, in contrast, treats the interference as Gaussian and hence discards this information.

The performance of the LMMSE-Soft-PIC detector has been assessed in terms of the maximal data rate that could be achieved with optimal outer coding, and in terms of word error rates that have been obtained with an LDPC code. In both metrics, the proposed algorithm clearly outperforms the conventional LMMSE detector (except in very low SNR, where even the conventional detector yields almost optimal results).

In the practically relevant medium SNR range (corresponding to code rates of, say, 1/3 to 2/3), the new algorithm performs comparably to the max-log SD, which has a much higher (and in particular non-deterministic) computational complexity. In high SNR, the proposed detector still improves considerably on the conventional LMMSE receiver, but it is clearly worse than the max-log SD, whose results are virtually optimal in high SNR. Further research should thus focus on the high SNR range, in order to decrease the remaining gap to capacity as far as possible. We have discussed one potential technique, the adoption of widely rather than strictly linear filters, but more engineering work is required to optimize the numerical stability of the algorithm. However, a more interesting topic for further research, in my opinion, are methods that aim at correcting the overestimated L-values that arise after the second, and particularly after the third inner iteration. At first glance, being able to cancel the spatial

interference thanks to very reliable feedback seems to be a curious problem, but to my best knowledge, no simple remedy has been found yet.

I have intentionally not presented a detailed complexity analysis of the LMMSE-Soft-PIC detector—I gladly leave this work to my fellow researchers who are specialized on VLSI implementations, and who will be able to present vastly more accurate numbers than I could within the scope of this thesis. Having said that, I am fairly confident that the proposed algorithm can be implemented with only a moderately increased complexity, compared to a conventional LMMSE receiver. According to the scheduling that I have proposed, the LMMSE-Soft-PIC detector only performs two inner iterations in the initial outer iteration—and this additional computation is well spent, considering the significant performance gain that it yields. In all subsequent BICM-ID iterations, the proposed detector runs for just a single inner iteration. Assuming that the complexity of computing and applying the LMMSE filter is dominated by the inversion (or decomposition) of the covariance matrix $\boldsymbol{C}_{\boldsymbol{y}}$, those later iterations should be comparable complexity-wise to a conventional LMMSE receiver.

A potential problem is that the proposed detector requires memory to store the parameters $\vec{\mu}$ and $\vec{\sigma}^2$ across the outer iterations. I am unable to offer a realistic judgement on how severe this issue might be for a VLSI implementation. In recent work [6], an implementation of a variant of the proposed detector has been presented, which avoids this additional memory at the cost of some algorithmic performance. The conclusion of [6] is that "lower area and higher throughput combined with significant SNR gains over regular linear MMSE detection is possible," which strengthens my confidence that the proposed LMMSE-Soft-PIC detector, or at least the fundamental idea behind it, will be a promising option for future iterative MIMO receivers.

Aside from its practical value, the work presented in this section is also interesting from a theoretical perspective. The empirical observation that LMMSE detectors work better with posterior than with extrinsic feedback, as reported in [148], has been quite remarkable due to its apparent contradiction to the turbo principle. To my best knowledge, the present work is the first to provide an explanation to this phenomenon.

4.4 MIMO Detection Based on Exclusive DM: Iterative MRC Receivers

The iterative LMMSE-based MIMO detector from the previous section is an instance of expectation propagation (EP), an algorithmic framework for approximate statistical inference. As discussed in Appendix B, EP can be derived as a special case of the more general divergence minimization (DM) approach: it is obtained by minimizing the *inclusive* Kullback-Leibler divergence $D_{\mathrm{KL}}(p \,\|\, q)$ with respect to q, where p is an (intractable) target distribution, and where q is constrained to a set of simpler distributions.

Variational message passing (VMP) is another statistical inference technique which can be derived in a similar manner as EP; the main difference is that VMP attempts to minimize the *exclusive* Kullback-Leibler divergence $D_{\mathrm{KL}}(q \,\|\, p)$. In Chapter 3 on iterative synchronization, we have successfully used both inclusive and exclusive DM to derive novel parameter estimators. The general outcome of that study is that algorithms that are based on inclusive DM tend to yield better results, but are also more complex than the ones derived from exclusive DM.

Now, a natural question to ask is whether exclusive DM can also be applied to the problem of MIMO detection. If so, we would expect to obtain a simpler but worse performing algorithm compared to the LMMSE-Soft-PIC detector. Depending on the extent of the performance loss and the achieved complexity reduction, such an algorithm could well have its scope of practical application. On the following pages, we shall investigate this question.

4.4.1 MRC Detection with Soft Parallel Interference Cancellation

As with all DM-based message passing algorithms, the following derivation starts with a suitable factorization of the joint posterior distribution. For the system that we are studying in this chapter, i. e., MIMO detection with perfect channel state information, the joint posterior has been given in (4.47) and is repeated here for convenience:

$$p(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x} \,|\, \boldsymbol{y}) \propto p(\boldsymbol{b})\, p(\boldsymbol{c} \,|\, \boldsymbol{b}) \prod_{n=0}^{N-1} p(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n) \prod_{i=0}^{N_{\mathrm{t}}-1} p(x_{n,i} \,|\, \boldsymbol{c}_{n,i}). \tag{4.103}$$

As in the previous section (c. f. (4.48)), we approximate (4.103) with a fully factorized auxiliary distribution

$$q(\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{x}) = \prod_{j=0}^{N_{\mathrm{b}}-1} q(b_j) \prod_{n=0}^{N-1} \prod_{i=0}^{N_{\mathrm{t}}-1} q(x_{n,i}) \prod_{k=0}^{K-1} q(c_{n,i,k}) \tag{4.104}$$

but this time, we use exclusive divergence minimization, i. e., variational message passing (see Section B.5.5 on page 265), for a subset of the local approximations.

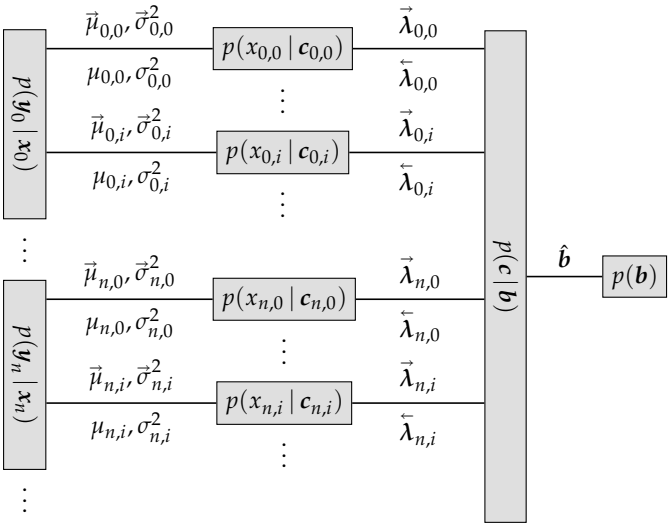


Figure 4.20: Forney-style factor graph of the joint posterior (4.103), and the information that flows along its edges

First of all, we need to determine which of the factors lend themselves to VMP. The local functions $p(\boldsymbol{c} \mid \boldsymbol{b})$ and $p(x_{n,i} \mid \boldsymbol{c}_{n,i})$ encode deterministic mappings (from messages to codewords, and from blocks of code bits to data symbols, respectively) and hence are Dirac distributions. As shown in the appendix, the VMP message update involves an integration over the logarithm of the local factor. Since the logarithm of a Dirac distribution is not a well defined function, the functions $p(\boldsymbol{c} \mid \boldsymbol{b})$ and $p(x_{n,i} \mid \boldsymbol{c}_{n,i})$ are not suitable for VMP.

This leaves us with the factors $p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)$ which encode the stochastic channel model. Using VMP, we will now approximate them with the fully factorized expression

$$p(\boldsymbol{y}_n \mid \boldsymbol{x}_n) \approx \prod_{i=0}^{N_\mathrm{t}-1} m_{\mathrm{equ}\to\mathrm{dem}}(x_{n,i}). \tag{4.105}$$

The underlying factor graph of the message passing algorithm is sketched in Figure 4.20. It is very similar to Figure 4.9 from the previous section; the only difference is that Figure 4.20 contains separate nodes for the individual data symbols, while Figure 4.9 kept $p(\boldsymbol{x}_n \mid \boldsymbol{c}_n) = \prod_{i=0}^{N_\mathrm{t}-1} p(x_{n,i} \mid \boldsymbol{c}_{n,i})$ as a single node. Figure 4.20 is thus a more accurate representation of the factorization (4.103). In fact, we could also have based the derivation of the LMMSE-Soft-PIC detector on the more detailed graph, but the math would have been slightly more involved. Hence, in the interest of simplicity,

we have preferred to work with Figure 4.9 instead, which has been possible due to the exponential family constraint that is enforced by EP's inherent projection step. By choosing a fully factorized constraint $q(\boldsymbol{x}_n) \stackrel{!}{=} \prod_{i=0}^{N_\mathrm{t}-1} q(x_{n,i})$, the graphs 4.9 and 4.20 essentially become equivalent. As mentioned in Section B.4.2, however, imposing an exponential family constraint on the solution of exclusive DM is generally intractable (because the resulting optimization problem is non-convex in general), so we will work with the graph from Figure 4.20 in the following.

Interference Cancellation and Spatial Filtering

We proceed by deriving the VMP updates of the equalizer-to-demapper messages given in (4.105). Starting from the generic update equation (B.62), we obtain

$$\begin{aligned}
m_{\mathrm{equ}\to\mathrm{dem}}(x_{n,i}) &\propto \exp\left(\mathbb{E}_{q(\boldsymbol{x}_{n,\setminus i})}[\log p(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n)]\right) \\
&\propto \exp\left(-\tfrac{1}{N_0}\,\mathbb{E}_{q(\boldsymbol{x}_{n,\setminus i})}\left[\|\boldsymbol{y}_n - \boldsymbol{H}_n\boldsymbol{x}_n\|^2\right]\right) \\
&\propto \exp\left(-\tfrac{1}{N_0}\,\mathbb{E}_{q(\boldsymbol{x}_{n,\setminus i})}\left[(\boldsymbol{y}_n^{\mathsf{H}} - \boldsymbol{x}_{n,\setminus i}^{\mathsf{H}}\boldsymbol{H}_{n,\setminus i}^{\mathsf{H}} - x_{n,i}^*\boldsymbol{h}_{n,i}^{\mathsf{H}})(\boldsymbol{y}_n - \boldsymbol{H}_{n,\setminus i}\boldsymbol{x}_{n,\setminus i} - \boldsymbol{h}_{n,i}x_{n,i})\right]\right) \\
&\overset{(a)}{\propto} \exp\left(-\tfrac{1}{N_0}\,\mathbb{E}_{q(\boldsymbol{x}_{n,\setminus i})}\left[|x_{n,i}|^2\,\|\boldsymbol{h}_{n,i}\|^2 - 2\,\mathrm{Re}\left(x_{n,i}^*\boldsymbol{h}_{n,i}^{\mathsf{H}}(\boldsymbol{y}_n - \boldsymbol{H}_{n,\setminus i}\,\boldsymbol{x}_{n,\setminus i})\right)\right]\right) \\
&\overset{(b)}{\propto} \exp\left(-\frac{\|\boldsymbol{h}_{n,i}\|^2}{N_0}\left(|x_{n,i}|^2 - 2\,\mathrm{Re}\left(x_{n,i}^*\,\frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}(\boldsymbol{y}_n - \boldsymbol{H}_{n,\setminus i}\,\boldsymbol{\mu}_{n,\setminus i})}{\|\boldsymbol{h}_{n,i}\|^2}\right)\right)\right) \\
&\propto \mathcal{CN}\left(x_{n,i}\,\middle|\,\frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}(\boldsymbol{y}_n - \boldsymbol{H}_{n,\setminus i}\,\boldsymbol{\mu}_{n,\setminus i})}{\|\boldsymbol{h}_{n,i}\|^2}, \frac{N_0}{\|\boldsymbol{h}_{n,i}\|^2}\right)
\end{aligned} \tag{4.106}$$

where (a) drops all terms that are independent of $x_{n,i}$, and (b) introduces the mean

$$\boldsymbol{\mu}_{n,\setminus i} \triangleq \mathbb{E}_{q(\boldsymbol{x}_{n,\setminus i})}\left[\boldsymbol{x}_{n,\setminus i}\right] \tag{4.107}$$

of the interfering data symbols, computed with respect to the current belief

$$q(\boldsymbol{x}_{n,\setminus i}) = \prod_{j\neq i} q(x_{n,j}). \tag{4.108}$$

When we derived the LMMSE-Soft-PIC detector in the previous section, a central step was the projection of $q(x_{n,i})$ into the family of proper Gaussians, which led to the crucial complexity reduction compared to the exact MIMO detector. This time, applying the generic VMP update equation immediately results in a proper Gaussian message, as evidenced by (4.106). Let us have a closer look at both its mean and its variance.

Mean of $m_{\mathbf{equ}\to\mathbf{dem}}(x_{n,i})$ The mean of the equalizer-to-demapper message is

$$\begin{aligned}\vec{\mu}_{n,i} &\triangleq \frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}(\boldsymbol{y}_n - \boldsymbol{H}_{n,\setminus i}\,\boldsymbol{\mu}_{n,\setminus i})}{\left\|\boldsymbol{h}_{n,i}\right\|^2} \\ &= \mu_{n,i} + \frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}(\boldsymbol{y}_n - \boldsymbol{H}_n\boldsymbol{\mu}_n)}{\left\|\boldsymbol{h}_{n,i}\right\|^2}. \end{aligned} \tag{4.109}$$

Note the similarity of (4.109) with the corresponding (4.58) from the previous section. Both consist of an interference cancellation step, followed by a filter which reduces the vector-valued observation to a scalar. There are, however, two key differences:

- The estimate of the interference in (4.58) is computed based on the *extrinsic* information $\overleftarrow{\mu}$ from the demapper (indicated by the right-to-left arrow). The corresponding estimate in (4.109), in contrast, is computed from the current belief, i. e., the *posterior* information (indicated by the lack of an arrow).

- The LMMSE filter $\frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}}{\boldsymbol{h}_{n,i}^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}\boldsymbol{h}_{n,i}}$ in (4.58) is replaced by the simpler expression $\frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}}{\|\boldsymbol{h}_{n,i}\|^2}$ which is simply the maximum ratio combiner (MRC) that we have briefly mentioned in Section 4.2.2.

Since the derived MIMO detector consists of soft parallel interference cancellation followed by MRC filtering, we will call it *MRC-Soft-PIC* in the following.

Variance of $m_{\mathbf{equ}\to\mathbf{dem}}(x_{n,i})$ The variance of the equalizer-to-demapper message as derived via VMP is

$$\vec{\sigma}_{n,i}^2 \triangleq \frac{N_0}{\left\|\boldsymbol{h}_{n,i}\right\|^2} \tag{4.110}$$

which is the inverse SNR of the ith data stream. In Section 4.2.2 above, we mentioned that the MRC filter maximizes the SNR while ignoring the interference from the other data streams. In some sense, it is thus not surprising that the variance of the VMP message, whose mean is given by MRC filtering, also disregards the spatial interference.

Let us now compute the true variance of the MRC output, including the interference term. On substituting $\boldsymbol{y}_n = \boldsymbol{H}_n\boldsymbol{x}_n + \boldsymbol{w}_n$ in (4.109), we obtain

$$\begin{aligned}\vec{\mu}_{n,i} &= \frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}\left(\boldsymbol{H}_n\boldsymbol{x}_n + \boldsymbol{w}_n - \boldsymbol{H}_{n,\setminus i}\,\boldsymbol{\mu}_{n,\setminus i}\right)}{\left\|\boldsymbol{h}_{n,i}\right\|^2} \\ &= x_{n,i} + \frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}\left(\boldsymbol{H}_{n,\setminus i}\left(\boldsymbol{x}_{n,\setminus i} - \boldsymbol{\mu}_{n,\setminus i}\right) + \boldsymbol{w}_n\right)}{\left\|\boldsymbol{h}_{n,i}\right\|^2}. \end{aligned} \tag{4.111}$$

Clearly, this is an unbiased estimate, $\mathbb{E}[\vec{\mu}_{n,i}] = x_{n,i}$, since $\boldsymbol{w}_n$ has zero mean, and since $\mathbb{E}[\boldsymbol{x}_{n,\setminus i}] = \boldsymbol{\mu}_{n,\setminus i}$ by definition (4.107). The variance of the estimate (4.111) is

$$\begin{aligned}
\operatorname{var}\left[\vec{\mu}_{n,i}\right] &= \frac{\mathbb{E}\left[\boldsymbol{h}_{n,i}^{\mathsf{H}}\left(\boldsymbol{H}_{n,\setminus i}\left(\boldsymbol{x}_{n,\setminus i}-\boldsymbol{\mu}_{n,\setminus i}\right)+\boldsymbol{w}_n\right)\left(\left(\boldsymbol{x}_{n,\setminus i}^{\mathsf{H}}-\boldsymbol{\mu}_{n,\setminus i}^{\mathsf{H}}\right)\boldsymbol{H}_{n,\setminus i}^{\mathsf{H}}+\boldsymbol{w}_n^{\mathsf{H}}\right)\boldsymbol{h}_{n,i}\right]}{\left\|\boldsymbol{h}_{n,i}\right\|^4} \\
&= \frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}\left(\boldsymbol{H}_{n,\setminus i}\boldsymbol{\Sigma}_{n,\setminus i}\boldsymbol{H}_{n,\setminus i}^{\mathsf{H}}+N_0\boldsymbol{I}_{N_{\mathrm{r}}}\right)\boldsymbol{h}_{n,i}}{\left\|\boldsymbol{h}_{n,i}\right\|^4} \\
&= \frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}\left(\boldsymbol{H}_n\boldsymbol{\Sigma}_n\boldsymbol{H}_n^{\mathsf{H}}\right)\boldsymbol{h}_{n,i}}{\left\|\boldsymbol{h}_{n,i}\right\|^4}-\sigma_{n,i}^2+\frac{N_0}{\left\|\boldsymbol{h}_{n,i}\right\|^2}
\end{aligned} \tag{4.112}$$

with $\sigma_{n,i}^2 \triangleq \mathbb{E}[\left|x_{n,i}-\mu_{n,i}\right|^2]$ and $\boldsymbol{\Sigma}_n \triangleq \operatorname{diag}(\sigma_{n,0}^2,\ldots,\sigma_{n,N_{\mathrm{t}}-1}^2)$.

It can be shown by simulations that using the "correct" variance (4.112) yields vastly improved results compared to the expression (4.110) that we have derived by applying the VMP message update equations. We will discuss this issue briefly in Section 4.4.4.

All simulation results that will be presented in Section 4.4.3 are based on (4.112).

Soft Mapping and Demapping

So far, we have only discussed the operation at the equalizer nodes (labeled $p(\boldsymbol{y}_n \,|\, \boldsymbol{x}_n)$ in Figure 4.20), which compute the left-to-right parameters $\vec{\mu}_{n,i}$ and $\vec{\sigma}_{n,i}^2$ from the posterior information $\mu_{n,i}$ and $\sigma_{n,i}^2$ as provided by the demapper nodes. The demapper nodes $p(x_{n,i} \,|\, \boldsymbol{c}_{n,i})$ themselves do not require a new derivation; as argued above, they are still based on inclusive DM and hence operate as before. In the forward direction, they compute the intrinsic L-values $\vec{\boldsymbol{\lambda}}_{n,i}$ using $\vec{\mu}_{n,i}$ and $\vec{\sigma}_{n,i}^2$ from the MRC filter and $\overleftarrow{\boldsymbol{\lambda}}_{n,i}$ from the decoder, as specified in (4.67) on page 151. In the backward direction, they compute $\mu_{n,i}$ and $\sigma_{n,i}^2$ by projecting the posterior information into the family of proper Gaussians

$$\mathcal{CN}(\mu,\sigma^2) = \operatorname{proj}_{\mathcal{CN}}\Bigg[\underbrace{\mathcal{CN}(\vec{\mu},\vec{\sigma}^2)}_{\substack{\text{intrinsic information}\\ \text{from MRC filter}}} \qquad \underbrace{\exp\left(\boldsymbol{c}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}\right)}_{\substack{\text{extrinsic information}\\ \text{from decoder}}}\Bigg] \tag{4.113}$$

similarly to (4.70), but without the subsequent division by $\mathcal{CN}(\vec{\mu},\vec{\sigma}^2)$ since the MRC filter requires the full posterior information. The implementation complexity of (4.113) can be further reduced with the approximation

$$\mathcal{CN}(\mu,\sigma^2) \approx \operatorname{proj}_{\mathcal{CN}}\left[\exp\left(\boldsymbol{c}^{\mathsf{T}}\vec{\boldsymbol{\lambda}}\right)\,\exp\left(\boldsymbol{c}^{\mathsf{T}}\overleftarrow{\boldsymbol{\lambda}}\right)\right]$$

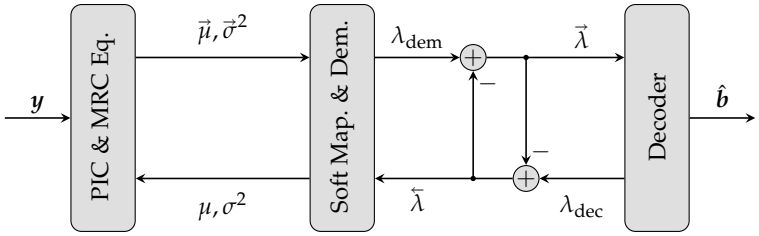


Figure 4.21: Block diagram of the proposed doubly iterative MRC-Soft-PIC receiver

$$= \text{proj}_{\mathcal{CN}}\left[\exp\left(\boldsymbol{c}^{\mathsf{T}}\boldsymbol{\lambda}_{\text{dec}}\right)\right] \tag{4.114}$$

which is just a soft symbol mapper that uses the posterior L-values from the decoder as input. In particular, this approximation gets rid of the memory that is required to save the parameters $\vec{\mu}$ and $\vec{\sigma}^2$ during the outer iterations. The background of (4.114) is the same as in Section 3.3.3 on page 51, where we have discussed this approximation in the context of iterative parameter estimation.

Overview of the MRC-Soft-PIC Receiver

Figure 4.21 shows a block diagram of the VMP-based MIMO receiver. Just as in Figure 4.11 before, the resulting algorithm is doubly iterative. On the right hand side, between demapper and decoder, we recognize the by now familiar BICM-ID loop. The left-hand loop between the spatial equalizer and the demapper is different, however: the subtraction points, which correspond to divisions of probabilities, are missing in Figure 4.21, since the MRC filter is fed with posterior rather than extrinsic feedback.

According to computer simulations that will be presented in Section 4.4.3 below, doing three to five inner iterations (between MRC filter and demapper) for each outer BICM-ID iteration seems to be a reasonable schedule.

4.4.2 A Combined LMMSE-MRC-Soft-PIC Receiver

In the present and the previous section, we have derived two similar MIMO detectors, which exchange information iteratively between an affine front-end (which performs interference cancellation and spatial equalizing) and a bank of streamwise symbol demappers. In the former algorithm, the spatial equalizer is realized as an LMMSE filter, and in the latter, as an MRC filter. Both approaches have their respective advantages and disadvantages.

On the one hand, the benefit of the MRC filter is its computational simplicity. As shown in (4.109), the result of the interference canceller is multiplied with the filter vector

$\frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}}{\|\boldsymbol{h}_{n,i}\|^2}$, and this operation has a complexity of $\mathcal{O}(N_\mathrm{r})$. Since the product needs to be evaluated for each stream, the spatial equalizer has a total complexity of $\mathcal{O}(N_\mathrm{t}N_\mathrm{r})$, i. e., a quadratic complexity if we assume $N_\mathrm{t} \propto N_\mathrm{r}$.

In contrast, the computation of the LMMSE filter vector $\frac{\boldsymbol{h}_{n,i}^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}}{\boldsymbol{h}_{n,i}^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}\boldsymbol{h}_{n,i}}$ as given in (4.58) is dominated by the inversion (or decomposition) of the covariance matrix (4.54)

$$\boldsymbol{C}_{\boldsymbol{y}_n} = \boldsymbol{H}_n \overleftarrow{\boldsymbol{\Sigma}}_n \boldsymbol{H}_n^{\mathsf{H}} + N_0 \boldsymbol{I}_{N_\mathrm{r}}. \tag{4.115}$$

Both the computation and the inversion of $\boldsymbol{C}_{\boldsymbol{y}_n}$ have a cubic complexity.

The benefit of the LMMSE filter, on the other hand, is its superior algorithmic performance. As discussed above, the LMMSE filter is designed to maximize the signal-to-interference-plus-noise ratio, whereas the MRC filter ignores the spatial interference and only maximizes the signal-to-noise ratio. It is thus to be expected that the performance gap between LMMSE and MRC depends on the power of the spatial interference: the stronger the interference (compared to the noise power), the heavier the performance degradation of the MRC filter. In particular, the performance gap will be especially pronounced in the initial iteration of the algorithm, when there is no interference cancellation possible yet. Subsequently, the power of the residual interference decreases during the iterations, and the difference between MRC and LMMSE becomes smaller.

Moreover, it should be noted that the covariance matrix (4.115) reduces to

$$\boldsymbol{C}_{\boldsymbol{y}_n} = E_\mathrm{s} \boldsymbol{H}_n \boldsymbol{H}_n^{\mathsf{H}} + N_0 \boldsymbol{I}_{N_\mathrm{r}} \tag{4.116}$$

in the initial iteration, and thus only depends on the channel. In most practical systems, the channel coefficients can be expected to stay approximately constant for at least a few symbol periods (or subcarriers, in OFDM systems). Otherwise, their estimates would become very poor and coherent communication would be impossible in the first place. It can thus be expected that the LMMSE filter matrix for the initial iteration can be used during several symbol periods, which reduces the average complexity per channel use accordingly. In the later iterations, however, the LMMSE filter must be recomputed for each symbol vector, even if the channel stays constant, since the covariance matrices $\overleftarrow{\boldsymbol{\Sigma}}_n$ will vary independently of each other.

For the important case of $N_\mathrm{r} > N_\mathrm{t}$,[23] there is a second reason why the LMMSE filter has a lower complexity in the first iteration than in all further iterations: the LMMSE filter vector for the ith data stream is (up to a normalization constant) given as $\boldsymbol{h}_{n,i}^{\mathsf{H}}\boldsymbol{C}_{\boldsymbol{y}_n}^{-1}$. Collecting the filter outputs for all streams into one vector, the LMMSE filter

[23] We will see in Section 4.4.3 that for $N_\mathrm{r} = N_\mathrm{t}$, the spatial interference is so strong that the MRC receiver breaks down completely. It is therefore a fair assumption that MRC-based receivers will only ever be operated in systems with $N_\mathrm{r} > N_\mathrm{t}$.

thus essentially needs to compute the expression $\boldsymbol{H}_n^{\mathsf{H}} \boldsymbol{C}_{\boldsymbol{y}_n}^{-1} \tilde{\boldsymbol{y}}_n$, where $\tilde{\boldsymbol{y}}_n = \boldsymbol{y}_n - \boldsymbol{H}_n \bar{\boldsymbol{\mu}}_n$ is the received signal after interference cancellation. Using the identity

$$\boldsymbol{H}^{\mathsf{H}} (\boldsymbol{H} \boldsymbol{A} \boldsymbol{H}^{\mathsf{H}} + \boldsymbol{B})^{-1} = \boldsymbol{A}^{-1} (\boldsymbol{A}^{-1} + \boldsymbol{H}^{\mathsf{H}} \boldsymbol{B}^{-1} \boldsymbol{H})^{-1} \boldsymbol{H}^{\mathsf{H}} \boldsymbol{B}^{-1} \tag{4.117}$$

for invertible $\boldsymbol{A}$ and $\boldsymbol{B}$, which can easily be derived using the matrix inversion lemma, the computation at the LMMSE filter can be written as

$$\begin{aligned} \boldsymbol{H}_n^{\mathsf{H}} \boldsymbol{C}_{\boldsymbol{y}_n}^{-1} \tilde{\boldsymbol{y}}_n &= \boldsymbol{H}_n^{\mathsf{H}} \left(\boldsymbol{H}_n \overleftarrow{\boldsymbol{\Sigma}}_n \boldsymbol{H}_n^{\mathsf{H}} + N_0 \boldsymbol{I}_{N_{\mathrm{r}}} \right)^{-1} \tilde{\boldsymbol{y}}_n \\ &= \overleftarrow{\boldsymbol{\Sigma}}_n^{-1} \left(\boldsymbol{H}_n^{\mathsf{H}} \boldsymbol{H}_n + \overleftarrow{\boldsymbol{\Sigma}}_n^{-1} N_0 \right)^{-1} \boldsymbol{H}_n^{\mathsf{H}} \tilde{\boldsymbol{y}}_n. \end{aligned} \tag{4.118}$$

In the first iteration, $\overleftarrow{\boldsymbol{\Sigma}}_n = E_{\mathrm{s}} \boldsymbol{I}_{N_{\mathrm{t}}}$, and (4.118) can be simplified to

$$\boldsymbol{H}_n^{\mathsf{H}} \boldsymbol{C}_{\boldsymbol{y}_n}^{-1} \tilde{\boldsymbol{y}}_n = \left(E_{\mathrm{s}} \boldsymbol{H}_n^{\mathsf{H}} \boldsymbol{H}_n + N_0 \boldsymbol{I}_{N_{\mathrm{t}}} \right)^{-1} \boldsymbol{H}_n^{\mathsf{H}} \tilde{\boldsymbol{y}}_n. \tag{4.119}$$

We have thus reduced the dimension of the matrix that needs to be inverted from $N_{\mathrm{r}} \times N_{\mathrm{r}}$ to $N_{\mathrm{t}} \times N_{\mathrm{t}}$. However, this trick only works in the initial iteration. As soon as prior information about the data symbols is available, the variances on the main diagonal of $\overleftarrow{\boldsymbol{\Sigma}}_n$ can get arbitrarily small, and (4.118) becomes numerically unstable, even though $\boldsymbol{H}_n^{\mathsf{H}} \boldsymbol{C}_{\boldsymbol{y}_n}^{-1} \tilde{\boldsymbol{y}}_n$ itself remains well defined.

To summarize, we have argued that in the initial iteration, the LMMSE receiver is computationally cheaper than in the later iterations; and that at the same time, its performance advantage compared to the MRC receiver is largest in the first iteration. Now, in an attempt to combine the advantages of both filters, we can implement a receiver which uses an LMMSE filter initially, when no interference cancellation is possible yet, but which switches to an MRC filter in the later iterations, where the interference can at least partially be cancelled due to the availability of non-uniform prior information from the channel decoder.

In terms of the divergence minimization framework, this means that the first computation of the equalizer-to-demapper messages is based on inclusive DM, and all later message updates on exclusive DM. While changing the divergence measure that is being minimized dynamically at runtime might be unusual (and all algorithms that we have discussed so far in this thesis keep the choice of divergence measure static), nothing in the general derivation of DM prevents us from doing so. It is admittedly a rather heuristic procedure, but a well motivated one. At the end of the day, every approximate solution of NP-hard problems is heuristic; the only property that matters is the tradeoff between performance and complexity that a particular algorithm realizes.

In order to assess this tradeoff for the plain MRC and the combined LMMSE-MRC Soft-PIC receivers, we proceed by presenting some numerical results.

4.4.3 Numerical Results

We begin with an information-theoretic study of the maximum data rates, as introduced in Section 4.1.4. Figure 4.22 shows $R_{\max}$ for non-iterative (upper plot) and iterative (lower plot) BICM receivers, using a 3×4 MIMO system with a Gray-mapped 16-QAM modulation. The receiver employs an MRC-Soft-PIC detector with 0 up to 6 inner iterations, where "0" refers to a pure feed-forward algorithm. Results for the max-log SD are also shown for comparison.

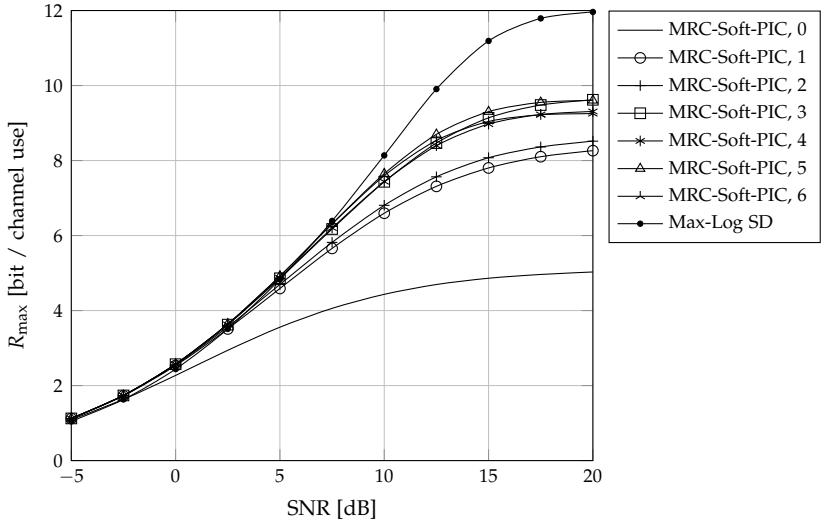
In both subfigures, we can clearly see that doing a single inner iteration already realizes a major part of the gain that can be achieved with the MRC-Soft-PIC compared to the plain non-iterative MRC detector. If the channel decoder is not part of the feedback loop (Figure 4.22a), the performance saturates after around five inner iterations; and in BICM-ID receivers (Figure 4.22b), three inner iterations are already sufficient. Furthermore, the performance of MRC-Soft-PIC is comparable to the max-log sphere detector in the low to medium SNR range: for non-iterative BICM receivers, up to a code rate of around $1/2$ (6 bpcu), and for BICM-ID receivers, even up to a code rate of more than $2/3$ (8 bpcu).

In the remainder of this section, the MRC-Soft-PIC detector performs four inner iterations in every outer iteration. Also, all WER results presented in the following have been obtained using a Gray-mapped 16-QAM modulation, and the LDPC code from the IEEE 802.11n standard with rate $1/2$ and a block length of 1944 codebits.

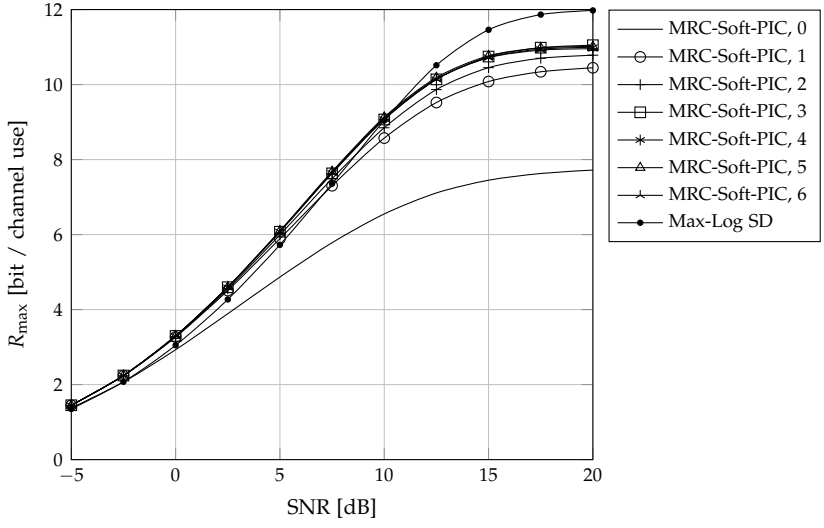
Note that the above results are for a 3×4 MIMO system, while all previous computer simulations in this chapter have been based on a symmetric 4×4 system. For a meaningful study of MRC receivers, however, we need to consider asymmetric systems with $N_{\mathrm{r}} > N_{\mathrm{t}}$ in this section. Since the MRC filter ignores the spatial interference and only maximizes the streamwise SNR, the iterative receiver only has a chance to converge to the correct solution if the spatial interference is sufficiently small, even in the initial iteration where the interference cannot be mitigated yet. The MRC filter output after the first iteration is (up to normalization constants)

$$\boldsymbol{x}_{\mathrm{MRC}} = \boldsymbol{H}^{\mathsf{H}}\boldsymbol{y} = \boldsymbol{H}^{\mathsf{H}}\boldsymbol{H}\boldsymbol{x} + \boldsymbol{H}^{\mathsf{H}}\boldsymbol{w} \tag{4.120}$$

from which we see that the diagonal elements of the Gramian matrix $\boldsymbol{H}^{\mathsf{H}}\boldsymbol{H}$ must be much larger (in magnitude) than the off-diagonal elements in order for the MRC filter to yield viable results, and the probability for this precondition to hold increases with the number of receive antennas. In fact, MRC-based MIMO receivers have to the best of my knowlege only been considered for massive MIMO systems, where the base station is equipped with hundreds of antenna elements, so that the uplink is characterized by a very tall channel matrix. For example, [71] notes that "While ZF also works fairly well for a conventional or moderately-sized MIMO system, MRC generally does not. The reason for why MRC works so well for massive MIMO is that the channel responses



(a) Non-iterative BICM receivers



(b) Iterative BICM-ID receivers

Figure 4.22: Upper bounds on the achievable data rates of a 3×4 MIMO system with a Gray-mapped 16-QAM modulation. The numbers in the legend refer to the number of inner iterations within the MRC-Soft-PIC detector

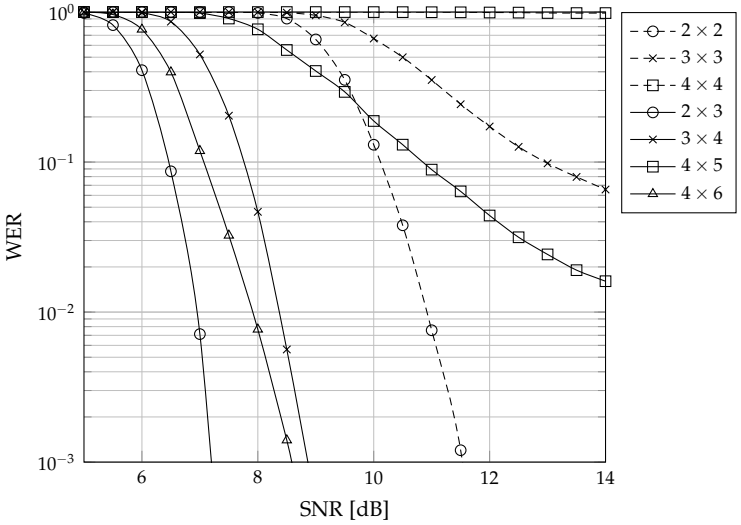


Figure 4.23: Word error rates of $N_t \times N_r$ MIMO systems with a Gray-mapped 16-QAM modulation and a rate-1/2 LDPC code, using the MRC-Soft-PIC receiver with 3 outer and 4 inner iterations

associated with different terminals tend to be nearly orthogonal when the number of base station antennas is large".

This statement of course refers to conventional, non-iterative MRC receivers. In order to check its validity for *iterative* MRC receivers, Figure 4.23 shows the word error rates of MIMO systems with different antenna configurations, using the MRC-Soft-PIC detector with 3 outer and 4 inner iterations. Out of the symmetric configurations (dashed lines), only the 2×2 system exhibits a distinct turbo-cliff, and even that only occurs at a very high SNR, with a gap of around 4 dB to the 2×3 system. The WER results of the 3×3 and 4×4 configurations are completely uninteresting. However, if we increase the number of receive antennas by just one or two, the results look entirely different. For two and three data streams, $N_r = N_t + 1$ is already enough to recover the turbo-cliff. For $N_t = 4$, five receive antennas are not yet sufficient, but the 4×6 system again works fairly well. These results show that the above cited statement from [71] is too pessimistic if one allows for iterative receivers. There is no need to enter the realm of massive MIMO and to increase the number of receive antennas into the hundreds; the MRC-Soft-PIC detector can well be employed in conventional MIMO systems with only mildly asymmetric antenna configurations.

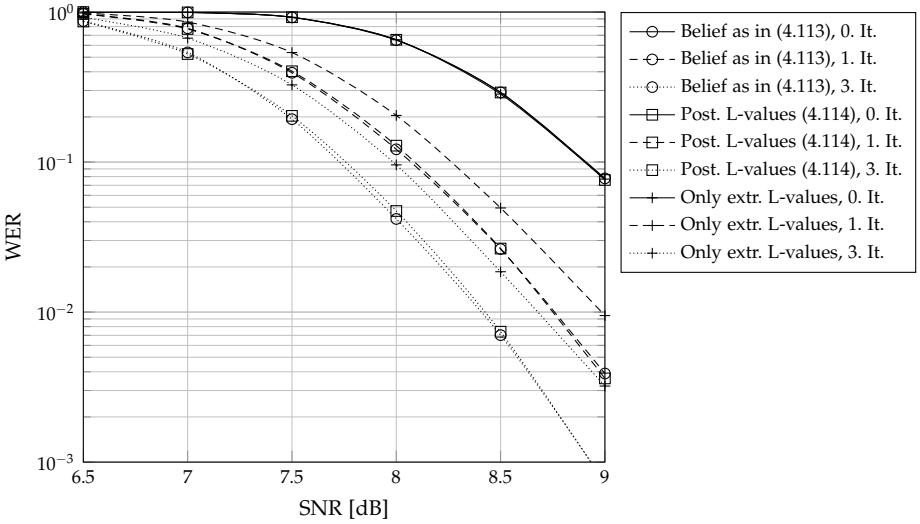
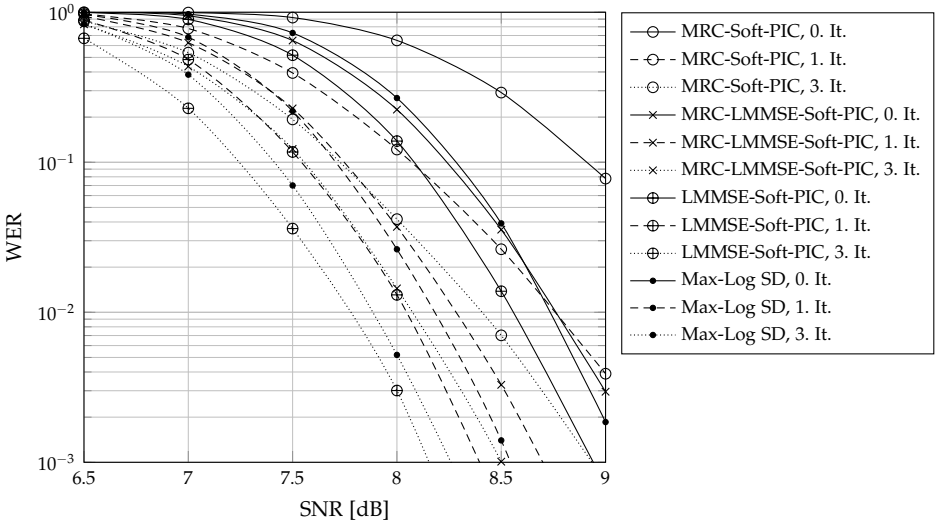


Figure 4.24: Performance of the MRC-Soft-PIC detector with different kinds of information that is being passed to the interference canceller and the MRC filter

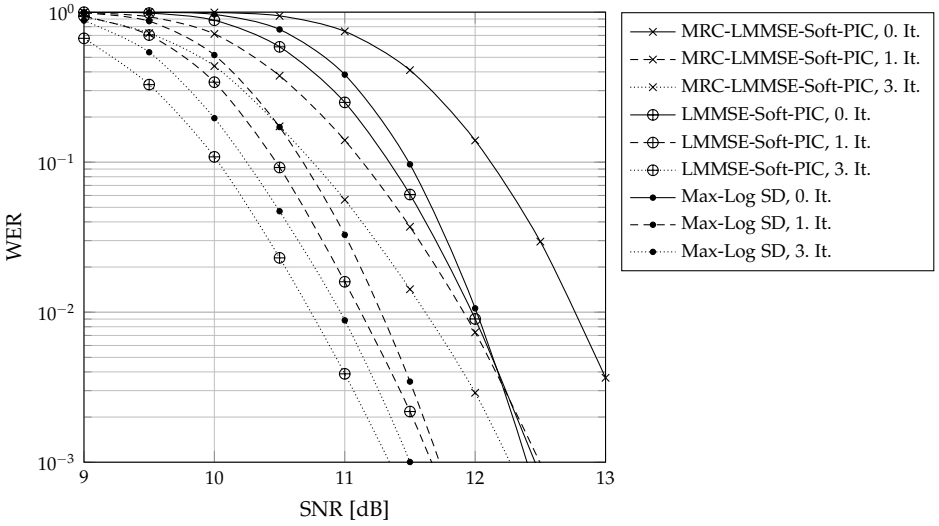
Next, we study the influence of the feedback type on the algorithmic performance. Since the MRC-Soft-PIC detector is an instance of exclusive DM, theory predicts that the symbolwise mean and variance should be computed using the available posterior information as shown in (4.113). Also, as mentioned before, computing these symbol statistics from the posterior L-values as in (4.114) instead is an approximation which reduces the computational complexity and the memory requirements. Both types of feedback are compared in Figure 4.24, and we see that the performance penalty incurred by the approximation (4.114) is absolutely negligible.

Moreover, Figure 4.24 also shows results with purely extrinsic feedback, i. e., similar to (4.114), but using $\overleftarrow{\boldsymbol{\lambda}}$ instead of $\boldsymbol{\lambda}_{\mathrm{dec}}$. Note that these are by no means theoretically motivated; they are only included in order to validate once more that the DM framework gives us reliable instructions how to connect the components of an iterative receiver. In fact, the figure shows that the MRC-Soft-PIC detector yields distinctly worse results when being fed with extrinsic instead of posterior information; with a gap of around 0.25 dB at a WER of 10^{-2} in this example.

In the final Figure 4.25, we compare the MRC-Soft-PIC detector with its LMMSE-based counterpart that we have derived in Section 4.3, and with the combined MRC-LMMSE-Soft-PIC detector from Section 4.4.2. Remember that the latter algorithm uses



(a) Asymmetric 3×4 MIMO System



(b) Symmetric 4×4 MIMO System

Figure 4.25: Comparison of the iterative MRC and LMMSE detectors, the combined MRC-LMMSE-Soft-PIC, and the max-log sphere detector

the LMMSE filter only in the initial iteration, which makes it only slightly more complex than the pure MRC filter.

The upper Figure 4.25a shows results for a 3×4 MIMO system. Clearly, the performance that can be gained by using LMMSE in the first iteration (where its interference rejection capabilities are most urgently needed) is quite significant. At a WER of 10^{-2}, and after three outer iterations, the gap between the pure MRC and LMMSE detectors is larger than 0.6 dB, which the combined MRC-LMMSE algorithm reduces by more than 50% to around 0.3 dB. And compared to the max-log sphere detector, the low-complexity MRC-LMMSE detector only has a gap of less than 0.2 dB.

For the sake of completeness, Figure 4.25b shows analogous results for a symmetric 4×4 MIMO system. The plain MRC-Soft-PIC detector is omitted from this figure, since its WER stays at essentially 1 over the whole SNR range that is shown in the figure. Of course, this had to be expected after the discussion of Figure 4.23. The combined MRC-LMMSE detector clearly improves on the catastrophic performance of plain MRC, but it still shows significant gaps of almost 1 dB compared to the LMMSE-Soft-PIC detector, and of 0.65 dB compared to the max-log SD (again after three outer iterations, and at a WER of 10^{-2}). Also, after three outer iterations, it only performs around 0.35 dB better than the LMMSE-Soft-PIC detector without any BICM-ID iterations.

Thus, while the MRC-LMMSE-Soft-PIC detector is in principle applicable to the examined symmetric 4×4 system (in contrast to the plain MRC-Soft-PIC detector), it is probably not a good choice in practice. For slightly asymmetric systems, however, both MRC-based detectors achieve an interesting tradeoff between algorithmic performance and computational complexity, which could make them viable options in particular for specialized hardware that is optimized for low power consumption.

4.4.4 Concluding Remarks

In this section, we have used the divergence minimization principle to derive another low-complexity MIMO detector in a similar vein as in Section 4.3, but with one crucial difference: whereas the LMMSE-Soft-PIC detector from the previous section has been obtained by minimizing the *inclusive* KL divergence at the factor nodes $p(\boldsymbol{y} \mid \boldsymbol{x})$, the present derivation has been based on minimizing the *exclusive* KL divergence.

The resulting MRC-Soft-PIC detector exhibits a clearly worse performance than its LMMSE-based counterpart. Of course, this does not come as a surprise: as shown in Section B.3.2, the inclusive KL divergence is in some sense the "correct" divergence measure to minimize.[24] It is hence to be expected that switching to a different divergence measure has a negative impact on the algorithmic performance. On the positive side,

[24] In the sense that a receiver which solves the global optimization problem (B.35) on page 244, where $\mathcal{F}$ is the set of fully factorized distributions, obtains the exact bitwise marginals of the posterior distribution p, and hence achieves the minimum BER among all possible receivers.

however, the computational complexity of the new detector is also significantly lower, since it does not require the inversion of the observation's covariance matrix.

Using MRC filters to combine multiple observations in wireless receivers is not a novel idea, but to the best of my knowledge, these filters have only been considered for massive MIMO systems in the literature, where the columns of the channel matrix are almost orthogonal with high probability (and for SIMO systems, where there is no spatial interference to begin with). In contrast, the proposed MRC-Soft-PIC detector can also be employed in moderately asymmetric MIMO systems, as we have shown by computer simulations. Its crucial advantage compared to ordinary MRC-based receivers is its ability to exploit knowledge about the discrete modulation alphabet, by means of an iterative information exchange between the affine front-end (including interference cancellation and MRC filtering) and the scalar demappers. We have verified numerically that it is better to use posterior rather than extrinsic information in the feedback path towards the front-end. This is in accordance with the results of the generic divergence minimization framework, since the proposed detector is based on exclusive DM and is hence an instance of variational message passing.

However, we have also seen that a literal application of VMP can result in a quite suboptimal algorithm. Deriving the equalizer-to-demapper message from the generic VMP update equation (B.62) results in a Gaussian function with variance (4.110) which is too small, because it only considers the noise but disregards the power of the residual spatial interference. Using the correct variance (4.112) instead is crucial for the MRC-Soft-PIC detector to work properly; in fact, I have refrained from presenting numerical results based on (4.110) at all, since the resulting word error rate is essentially equal to one over the whole SNR range of interest.

Note that we have encountered this issue already in Section 3.4 on phase estimation: the ζ-parameter of the soft phase estimate (3.73), obtained via exclusive DM, has a larger magnitude than in (3.78) obtained by inclusive DM, since the latter includes an additional term in its denominator which accounts for the uncertainty of the data symbol. As the variance of a von Mises distribution is (approximately) inversely proportional to the modulus of ζ, we see that the soft estimate (3.73) has a smaller variance than (3.78). In the context of phase estimation, however, it hardly makes a difference whether one uses the correct variance or the one obtained from VMP, as shown in Figure 3.12. This is in stark contrast to the MRC-based MIMO detector from the present section.

It is not a coincidence that we have observed the same phenomenon in two quite different scenarios. Underestimating the variance is actually an inherent characteristic of exclusive DM, which has already been described in [94, Section 2], and which can be explained as follows. As discussed in Appendix B.4.4 on page 255, the difference between inclusive and exclusive DM is that the former spreads its probability mass across the whole support of the target distribution p, whereas the latter only covers a single mode of p. This behaviour, which is illustrated in Figure B.8, is indeed the very origin of the names "inclusive" and "exclusive" divergence. Now, recall that the projection

(which is defined via *inclusive* DM) of an arbitrary p into the exponential family is found by matching the moments of the sufficient statistics, which in particular means that the Gaussian approximation obtained by inclusive DM preserves the mean and variance of p exactly. But since exclusive DM leads to a more compact approximation, it must necessarily underestimate the variance of p.

The bottom line is that the algorithms that are derived via divergence minimization should not be taken too literally. They are heuristic approximations to NP-hard problems, after all, and it certainly does not hurt to investigate possibilities for manually tweaking the mechanically derived equations. But despite its imperfections, the generic DM framework has again proven to be a valuable tool for the systematic design of composite receivers, which are built from local components that belong to different algorithmic classes, connected in a theoretically sound manner to a global iterative structure.

Chapter 5

Summary and Outlook

The development of the iteratively decoded turbo and LDPC codes has arguably been one of the key discoveries in the area of wireless communication. Motivated by their unprecedented performance, researchers soon tried to extend the concept of iterative information processing to other receiver components beyond the channel decoder.

Two different algorithmic frameworks have become the standard mathematical groundwork of iterative receivers. Belief propagation, which was discovered to be the theoretical fundament of the turbo and LDPC decoders, has successfully been applied to detection problems, yielding in particular the well-known iterative BICM-ID algorithm. However, BP turned out to be less suitable for the estimation of continuous variables. Instead, expectation maximization has emerged as the state-of-the-art algorithmic framework for the derivation of parameter estimators. But since BP and EM are two unrelated classes of algorithms (at least in their original formulations), it is necessary to derive the EM-based parameter estimators and the BP-based detectors and decoders separately, and to connect them in some heuristic manner.

The topic of this research project has been the study of a more generic approach to the design of iterative receivers. The fundamental concept has been developed in the machine learning community as a method for finding approximate solutions to intractable statistical inference problems. The idea is to replace the joint probability distribution, which describes the relations between all variables in the problem domain, with an auxiliary distribution which is subsequently used to answer any probability queries. The process of selecting the auxiliary distribution involves two conflicting goals. On the one hand, it should resemble the true distribution as closely as possible, since any discrepancy may degrade the quality of the final results. But on the other hand, it must be simple enough so that it can actually be used to answer the probability queries, or nothing would have been gained. These two criteria give rise to a constrained optimization problem. The most appropriate auxiliary distribution is the one which minimizes a given measure of dissimilarity with respect to the true distribution; selected

from a set of feasible solutions which only contains sufficiently simple distributions. Since the objective function which quantifies the difference between two probability distributions is usually called a *divergence* measure, the general approach to statistical inference that is described above is called *divergence minimization* (DM) in this thesis.

Unfortunately, solving these divergence minimization problems exactly in practice is also infeasible. They are hence broken down into smaller interconnected subproblems, whose provisional solutions are iteratively refined according to some fixed-point scheme. In the context of receiver design, these subproblems correspond to the functional units of wireless receivers, like parameter estimators, MIMO detectors, and channel decoders.

The generalized formulation of DM provides the designer with a great deal of flexibility. In particular, it contains several more specific algorithms, among them the above mentioned BP and EM, as special cases, which can be obtained by choosing certain divergence measures and feasible sets. Now, a key characteristic of the generic DM approach is that it is possible to apply different objective functions and feasible sets to the individual subproblems. This allows for the systematic derivation of entire receiver structures from a global optimization criterion, where every functional component is based on the algorithmic technique that is most appropriate for its task.

Estimation

In the first part of this thesis, we have used the DM framework to derive approximate solutions to estimation problems. By constraining the feasible set of the parameter distributions to Dirac deltas, one obtains "hard" estimates as known from orthodox statistics. We have seen that the derived structure is formally equivalent to state-of-the-art hybrid receivers, which combine EM-based estimators with BP-based detection and decoding units. Since the estimators decide on one particular (and almost surely wrong) parameter value, which is subsequently used for data detection, this technique is known as *mismatched detection*.

At this point, we have been able to start reaping the benefits of the DM approach. By relaxing the Dirac-constraint of the parameter distributions, we have derived "soft" estimators, which compute not just a single value, but entire probability distributions that encode the remaining uncertainty about the parameter. The resulting algorithms thus follow the Bayesian paradigm: they start with prior distributions for the variables, and update them to posterior distributions as soon as new information becomes available.

Bayesian statistics has occasionally been criticized for being subjective, due to the necessity of assigning prior distributions to the variables, whereas orthodox statistics supposedly lets the data speak for itself. However, we have seen that *conjugate priors* offer an objective choice which is entirely determined by the probabilistic model.

In order to appreciate the benefits of the DM approach, we have conducted an exhaustive case study on carrier phase estimation. Starting from a simple model with a phase offset that remains constant during the transmission, we have derived an elegant

soft-input soft-output phase estimator, which computes von Mises distributions that are parameterized by a single complex number. Information about the carrier phase that has been obtained from independent sources can be optimally combined by taking the sum of the corresponding complex parameters, which leads to a simple and intuitive expression for the soft-output estimator. By also modifying the detection metric such that it takes the residual phase error into account, we have developed a Bayesian receiver which outperforms its traditional EM-based counterpart with only a mildly increased computational complexity.

A key benefit of the holistic design approach, compared to a separate derivation of the individual receiver components, is that is also specifies the connections between the functional units. Soft information exchange has thus been a recurring topic in this thesis. Earlier publications have suggested that the estimators should be provided with extrinsic information about the data symbols, with a reference to the "turbo principle" which states that an algorithmic unit should only pass "new" information to its neighbors. Our derivation however has shown that the estimators require all available information, which should ideally be computed by combining the intrinsic information with the extrinsic L-values that are fed back from the channel decoder. Simply mapping the posterior L-values to soft symbols, as often done in the literature, is just an approximation; albeit a fairly good one which is justified by its significantly lower complexity with only a small performance penalty. On the other hand, the data detector with soft phase input should only consider the extrinsic phase information; i. e., when detecting the nth symbol, the phase knowledge that has been obtained from the nth observation should be ignored. These results are certainly not obvious, and demonstrate the practical value of the proposed methodology.

A further benefit of the Bayesian approach is that estimators for static parameters can easily be generalized to time-varying processes. Bayesian inference essentially consists of updating prior to posterior distributions in the light of new data. If the unknown parameter varies significantly between two symbols, one can update its estimate after every received symbol, using the posterior distribution from the previous symbol as the prior for the next. This naturally leads to forward-backward algorithms which resemble the Kalman smoother, but which may generally operate on non-Gaussian densities.

We have demonstrated this concept by deriving a phase-noise estimator, which we have broken down into two parts, a front-end and a back-end. The front-end is independent of the process statistics. It connects the back-end with the rest of the receiver by converting the soft information from the data symbols into likelihood functions with respect to the time-varying parameter, and by converting the parameter estimates back into metrics that are consumed by the data detector. The back-end, on the other hand, is immediately dependent on the assumed statistics which describe the behaviour of the random process. We have derived two such back-ends with different complexity: a simple one which tracks the output of a free-running oscillator, and a more complex one based on a generic state space model. The latter is sufficiently generic so that it can be

used for phase-locked loops, or for the estimation of a residual frequency offset which leads to a linearly increasing phase shift.

While we have focused this discussion on phase-noise processes, it should be noted that the same principle can also be used for code-aided channel estimation. Different channel models, like autoregressive or basis expansion models, give rise to different implementations of the back-end, whereas the front-end essentially turns classical data-aided channel estimators into iterative code-aided algorithms, by providing the back-end with soft information that is contained in the data symbols.

In a final case study, we have examined the joint estimation of channel gain and noise variance. Here, the major point has been the importance of considering all available information in the algorithm design. It is a natural constraint that the channel gain must be a non-negative quantity. However, orthodox estimators like the EM algorithm treat the unknown parameters as deterministic, which makes it difficult to incorporate prior knowledge. The Bayesian formulation, in contrast, allows the exclusion of physically impossible values simply by assigning them zero prior probability. We have seen that the proposed estimator offers a performance gain with respect to the EM algorithm, which is due to the fact that the latter sometimes computes a nonsensical, negative gain.

Detection

In the second part of this thesis, we have applied the DM approach to detection problems. As a case study, we have considered the joint detection of spatially superimposed data streams in multi-antenna systems. The proposed techniques are however more widely applicable, since the underlying model of a linear channel plus Gaussian noise is very general. It occurs for instance in the context of multi-user detection in CDMA systems, and turbo equalization for frequency-selective channels.

Conceptually, MIMO detection is a fairly simple problem. Practically, however, it suffers from a computational complexity that increases exponentially with the number of data streams. For many practically relevant applications, solving the detection problem exactly in real time is prohibitively hard, which motivates our interest in suboptimal, low-complexity approximations.

Linear detectors are a class of algorithms with particularly appealing complexity characteristics. They work in two stages. First, a linear filter is applied to the received vector-valued signal in order to separate the superimposed data streams. The results are then processed by mutually independent scalar detectors.

The most important filter for linear detectors is the LMMSE filter which minimizes the total distortion, taking both noise and inter-symbol interference into account. However, due to the linearity constraint, even the LMMSE filter is unable to exploit the receiver's knowledge about the discrete modulation alphabet, which can lead to a significant performance degradation compared to the exact detector.

To mitigate this problem, successive interference cancellation techniques have been proposed, but they suffer from error propagation due to erroneous hard decisions, which can be so severe that SIC can actually degrade rather than improve the results.

Another strategy is to solve the detection and decoding problems jointly, by means of the aforementioned iterative BICM-ID technique. This algorithm is based on belief propagation, and proceeds by exchanging extrinsic information about the transmitted bits between the detector and the channel decoder. However, it has been observed by computer simulations that iterative receivers which employ a linear MIMO detector actually work better if they feed the full posterior information back to the detector, rather than just extrinsic L-values. This phenomenon seems to violate the turbo principle, and clearly demonstrates the necessity of a more thorough theoretical understanding.

Once again, the fundamental issue consists in the separate design of the individual receiver components, which are then connected in some heuristic manner. By applying the DM framework to the global inference problem, we have been able to derive an iterative receiver structure with an affine MIMO front-end, consisting of parallel interference cancellation followed by an LMMSE filter, and a conventional BICM-ID loop. The crucial step in our derivation has been the introduction of a Gaussian constraint on the auxiliary distributions of the data symbols, which has turned the summations over exponentially many terms into integrals which admit closed-form solutions. Due to this constraint, the resulting algorithm is an instance of expectation propagation rather than belief propagation, since only the first two moments of the data symbols are exchanged between the LMMSE filter and the scalar detectors.

We have seen that the proposed receiver exhibits a novel doubly-iterative structure. On the one hand, there is the traditional BICM-ID loop between the scalar detectors and the channel decoder, which exchange extrinsic information about the code bits. For numerical reasons, implementations of the BICM-ID algorithm represent this information in the logarithmic domain in terms of L-values, but fundamentally, the information that is being exchanged consists of Bernoulli distributions (i. e., distributions over $\{0, 1\}$). The subtraction, which computes the extrinsic L-values as the difference of the posterior and prior L-values, is in fact a division of the posterior and prior Bernoulli distributions.

The novel loop, between the LMMSE filter and the detectors, is structurally identical. The only difference is that these components exchange information about the complex data symbols in terms of Gaussian distributions, i. e., in terms of mean and variance. In particular, the detectors first compute the symbolwise posterior distributions by combining the intrinsic information from the LMMSE filter and the extrinsic L-values that are provided by the channel decoder. The resulting discrete distributions are then reduced to Gaussians by computing their first two moments; but before this information is passed back to the LMMSE filter, the intrinsic component which has originated in the LMMSE filter itself is removed by dividing the two Gaussian functions, which in practice is implemented as a subtraction of their natural parameters.

There are two aspects that should be emphasized. First, the proposed receiver indeed uses the full posterior information (and not just the extrinsic L-values) for computing the discrete, symbolwise distributions. This explains the remarkable observation that iterative receivers with linear MIMO detectors work better with posterior instead of extrinsic feedback. Secondly, however, the algorithm is fully compatible with the turbo principle, because it removes the information that has been contributed by the LMMSE filter before the remaining information is passed back to the affine front-end. We have seen that this final step leads to a further performance gain with respect to the heuristic LMMSE detector with posterior L-value feedback. In the original, exact formulation of the MIMO detector, which does not reduce the discrete symbolwise distributions to Gaussians, adding and removing the intrinsic information are inverse operations, so they cancel each other out, and the remaining feedback solely consists of the extrinsic L-values. Due to the nonlinear reduction step, however, these two operations are not inverse anymore in LMMSE-based receivers, and computing the feedback to the LMMSE filter solely from the extrinsic L-values causes a significant performance degradation compared to our proposed method.

We have shown by simulations that the proposed iterative LMMSE detector with soft parallel interference cancellation is able to perform comparably to the sphere detector over a wide SNR range; from very low SNR where even the conventional LMMSE receiver yields close to optimal results, up to medium SNR which corresponds to a code rate of, say, $1/2$ to $2/3$. In the high SNR regime, however, the algorithm tends to overestimate the magnitudes of the L-values, which causes quite a performance loss compared to the sphere detector. While the reason for this phenomenon is understood, we have unfortunately not been able to find a remedy. Nonetheless, the proposed detector seems to be an interesting option for future mobile receivers, thanks to its good performance and its fairly low and in particular constant complexity (in contrast to the sphere detector, whose runtime depends on the channel and noise realizations).

By introducing a different modification to the divergence minimization problem, this time simplifying the objective function instead of the feasible set, we have finally derived another linear MIMO detector. The new algorithm uses an MRC instead of an LMMSE filter, which leads to an even simpler implementation since the MRC filter does not require any matrix inversions; it merely consists of multiplying the received signal with the adjoint of the channel matrix. We have furthermore shown that the new filter requires a different kind of feedback: it must be fed with posterior information, i. e., the division step that we have described above needs to be omitted. If we had simply exchanged the filter heuristically, with the intention of reducing the complexity of our receiver, we would certainly have missed this necessary modification. This demonstrates once more the practical value of the systematic design that is offered by the DM approach.

Outlook

Solving the generic DM equations often leads to reliable expressions, in the sense that they offer a superior algorithmic performance compared to more heuristic approaches. However, we have also seen examples of quite suboptimal results. In the derivation of the iterative MRC-based MIMO detector, a literal application of the DM equations yields a significantly underestimated variance (and hence an overestimated reliability of the detected data) because the resulting expression does not consider the residual interference from the superimposed data streams. In this example, it has been fairly simple to derive the correct variance after interference cancellation and filtering, and the new expression indeed causes a significant performance gain. For the above mentioned problem of the iterative LMMSE detector, which leads to overestimated L-values in the high-SNR regime, I have however not been able to propose a simple correction. Investigating the root causes of these problems and developing countermeasures would certainly be an interesting research project. If it were possible to correct the mismatched L-values of the iterative LMMSE detector with a moderate computational complexity, the algorithm would yield excellent results across the whole SNR range.

The focus of this thesis has exclusively been on deterministic algorithms, which optimize a simple parametric distribution to match the true posterior as well as possible. However, these methods will necessarily yield suboptimal results for involved models. For example, replacing a significantly non-Gaussian distribution with a Gaussian will always be a poor fit, no matter how well the parameters of the approximating density have been optimized. Stochastic inference algorithms, on the other hand, hardly make any assumptions at all on the target distribution, which has pros and cons. In those scenarios where deterministic algorithms work well, stochastic algorithms will likely require more computational power to yield similar results, since they must draw many samples in order to represent the target distribution faithfully, whereas the deterministic algorithm only needs to tune a few parameters. But with an increasing complexity of the underlying model, the lack of assumptions becomes an asset, and a well designed stochastic algorithm will converge to the correct solution in the limit of an infinite runtime. Besides studying deterministic algorithms, I have also worked on stochastic approaches to MIMO detection in the course of this research project; c. f. [116,119,122]. In particular, [116] presents a Markov chain Monte Carlo method for MIMO detection in the presence of residual channel estimation errors. Since the computational power of mobile receivers will continue to grow in the foreseeable future, it is predictable that stochastic algorithms will gain in importance. Fortunately, the methods that we have discussed in this thesis will not become obsolete by these developments, since the two approaches are by no means mutually exclusive. For example, the aforementioned stochastic MIMO detector could well be coupled with a DM-based channel estimator. It would be interesting to explore which tasks of modern receivers can profit the most from stochastic algorithms, and which ones are better handled by deterministic methods.

Appendix A

Mathematical Background

A.1 Differential Calculus

A common task in signal processing applications is the optimization of some cost function f, which is necessarily scalar and real-valued. Its parameters, however, may be complex, and in this thesis we are often concerned with functions $f : \mathbb{C}^N \to \mathbb{R}$.

The derivative of complex functions $f : \mathcal{D} \to \mathbb{C}$ with domain $\mathcal{D} \subseteq \mathbb{C}$

$$f'(z_0) \triangleq \lim_{z \to z_0} \frac{f(z) - f(z_0)}{z - z_0} \tag{A.1}$$

as studied in complex analysis, however, is not applicable to the problem of finding stationary points of real-valued functions. Indeed, it is a well known result that a *holomorphic* (complex-analytic) function f must satisfy the Cauchy-Riemann equations

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \quad \text{and} \quad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x} \tag{A.2}$$

where $z = x + \mathrm{j}y$ and $f = u + \mathrm{j}v$. If f is real-valued, $\frac{\partial v}{\partial x} = \frac{\partial v}{\partial y} = 0$, so any real-valued holomorphic function must be constant.

However, a non-holomorphic function can still be *real* differentiable when viewed as a function of two real variables $f(x, y)$, so the traditional real gradient can be used to find stationary points of f. The downside of this approach is that the resulting formulas tend to be awkward, since the inherent coupling of x and y as real and imaginary part of one complex variable gets lost.

An elegant alternative is the Wirtinger calculus [146], also called $\mathbb{CR}$-calculus [67], which defines two generalized complex derivatives, applicable to non-holomorphic (and in particular real-valued) functions. While the original work of Wilhelm Wirtinger dates back to 1927, it appears that the English speaking engineering community has only taken

notice of it through Brandwood in 1983, who defined a complex gradient operator [16]. Later, this work was extended by von den Bos [136], who derived a complex gradient with respect to the augmented[1] vector

$$\underline{z} \triangleq \begin{pmatrix} z \\ z^* \end{pmatrix} \tag{A.3}$$

and used it to define a complex Hessian. At about the same time, the work of Picinbono on statistical signal processing of complex data was published [106–109].

The purpose of this section is to give a brief introduction into the topic of complex and multivariate calculus, both in order to establish the notation used in this thesis, and to serve as a collection of some basic definitions and results that are needed by the main text. Recommended further reading includes the text books [57, 115] and the articles [1, 36, 56, 67, 74, 102]. The section closes with a brief discussion of variational calculus.

A.1.1 Univariate Complex Calculus

Consider some complex univariate function[2] $f : \mathbb{C} \to \mathbb{C}$. By splitting both its domain and codomain into real and imaginary part, we can view f also as a function $\mathbb{R}^2 \to \mathbb{R}^2$. With a slight abuse of notation, we use the same symbol for both functions and write $f(z) = f(x + \mathrm{j}y) = f(x, y) = u(x, y) + \mathrm{j}v(x, y)$. The complex function $f(z)$ is called *real-differentiable* if the four partial derivatives $\frac{\partial u}{\partial x}$, $\frac{\partial u}{\partial y}$, $\frac{\partial v}{\partial x}$ and $\frac{\partial v}{\partial y}$ exist, in which case we define the partial derivatives of f straightforwardly as

$$\frac{\partial f}{\partial x} \triangleq \frac{\partial u}{\partial x} + \mathrm{j}\frac{\partial v}{\partial x} \quad \text{and} \quad \frac{\partial f}{\partial y} \triangleq \frac{\partial u}{\partial y} + \mathrm{j}\frac{\partial v}{\partial y}. \tag{A.4}$$

The differential of f, denoted as $\mathrm{d}f$ and defined as the linear part of the difference $f(x + \mathrm{d}x, y + \mathrm{d}y) - f(x, y)$ (and analogously for the other differentials that we encounter later in this section), is then given in terms of its partial derivatives by the usual formula

$$\mathrm{d}f = \frac{\partial f}{\partial x}\,\mathrm{d}x + \frac{\partial f}{\partial y}\,\mathrm{d}y. \tag{A.5}$$

The two generalized complex derivatives, which are the central ingredients of the Wirtinger calculus, are now defined as

$$\frac{\partial f}{\partial z} \triangleq \frac{1}{2}\left(\frac{\partial f}{\partial x} - \mathrm{j}\frac{\partial f}{\partial y}\right) \quad \text{and} \quad \frac{\partial f}{\partial z^*} \triangleq \frac{1}{2}\left(\frac{\partial f}{\partial x} + \mathrm{j}\frac{\partial f}{\partial y}\right) \tag{A.6}$$

[1] He did not use the term *augmented*, though, which to my best knowledge was coined by Schreier and Scharf [115].
[2] To simplify the notation, we will assume without loss of generality that f is defined on the whole $\mathbb{C}$ or $\mathbb{C}^N$.

from which

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial z} + \frac{\partial f}{\partial z^*} \quad \text{and} \quad \frac{\partial f}{\partial y} = \mathrm{j}\left(\frac{\partial f}{\partial z} - \frac{\partial f}{\partial z^*}\right) \tag{A.7}$$

are easily obtained.

Definition (A.6) might seem arbitrary at first. It is justified by the fact that it yields, in conjunction with the definitions

$$\mathrm{d}z \triangleq \mathrm{d}x + \mathrm{j}\,\mathrm{d}y \quad \text{and} \quad \mathrm{d}z^* \triangleq \mathrm{d}x - \mathrm{j}\,\mathrm{d}y \tag{A.8}$$

an expression for the differential of f that is analogous to (A.5)

$$\mathrm{d}f = \frac{\partial f}{\partial z}\,\mathrm{d}z + \frac{\partial f}{\partial z^*}\,\mathrm{d}z^*. \tag{A.9}$$

It is easy to verify that $\frac{\partial f^*}{\partial z^*} = \left(\frac{\partial f}{\partial z}\right)^*$, which implies that in general $\frac{\partial f}{\partial z^*} \neq \left(\frac{\partial f}{\partial z}\right)^*$. Only in the special case of a real valued function, we have $f = f^*$ and thus $\frac{\partial f}{\partial z^*} = \left(\frac{\partial f}{\partial z}\right)^*$, in which case the differential of f becomes $\mathrm{d}f = 2\,\mathrm{Re}\left(\frac{\partial f}{\partial z}\,\mathrm{d}z\right)$.

If the only way to obtain $\frac{\partial f}{\partial z}$ and $\frac{\partial f}{\partial z^*}$ was to compute the derivatives with respect to real and imaginary part and then applying (A.6), we would not have gained much. The elegance of the Wirtinger calculus lies in the fact that the complex derivatives can be computed very naturally by treating z and z^* formally as independent variables [16, Theorem 1]. In particular, we have

$$\frac{\partial z}{\partial z} = \frac{\partial z^*}{\partial z^*} = 1 \quad \text{and} \quad \frac{\partial z}{\partial z^*} = \frac{\partial z^*}{\partial z} = 0. \tag{A.10}$$

If $f : \mathbb{C} \to \mathbb{R}$ is real-valued and real-differentiable, then both

$$\frac{\partial f}{\partial z} = 0 \quad \text{and} \quad \frac{\partial f}{\partial z^*} = 0 \tag{A.11}$$

are necessary and sufficient conditions for f to have a stationary point [16, Theorem 2].

It can be shown that the complex derivatives $\frac{\partial}{\partial z}$ and $\frac{\partial}{\partial z^*}$ obey the same sum and product rules as the standard real derivatives. The chain rule, however, is different, as the inner function needs to be considered both in non-conjugate and conjugate form [36]

$$\frac{\partial f \circ g}{\partial z} = \frac{\partial f}{\partial g}\,\frac{\partial g}{\partial z} + \frac{\partial f}{\partial g^*}\,\frac{\partial g^*}{\partial z} \quad \text{and} \quad \frac{\partial f \circ g}{\partial z^*} = \frac{\partial f}{\partial g}\,\frac{\partial g}{\partial z^*} + \frac{\partial f}{\partial g^*}\,\frac{\partial g^*}{\partial z^*}. \tag{A.12}$$

Later we will see how this inconvenience can be circumvented with the notion of augmented vectors.

Example To appreciate the simplicity of the Wirtinger calculus, consider the function

$$f(z) = |z|^2 = x^2 + y^2. \tag{A.13}$$

With $\frac{\partial f}{\partial x} = 2x$ and $\frac{\partial f}{\partial y} = 2y$, we obtain via (A.6)

$$\frac{\partial f}{\partial z} = \tfrac{1}{2}\,(2x - \mathrm{j}2y) = x - \mathrm{j}y = z^* \quad \text{and} \quad \frac{\partial f}{\partial z^*} = \tfrac{1}{2}\,(2x + \mathrm{j}2y) = x + \mathrm{j}y = z. \tag{A.14}$$

By writing $f(z, z^*) = zz^*$ and treating the two parameters as independent, we immediately get $\frac{\partial f}{\partial z} = z^*$ and $\frac{\partial f}{\partial z^*} = z$. Even in this small example, the simplification that the Wirtinger calculus offers is evident.

A.1.2 Multivariate Real Calculus

Before we extend the discussion to multivariate complex calculus, we introduce some basic definitions for the real-valued case.

Jacobian Let $f : \mathbb{R}^N \to \mathbb{R}$ be a real-valued function of N real parameters. We define its derivative with respect to the vector $\boldsymbol{x}$ as the *row* vector of partial derivatives

$$\frac{\partial f}{\partial \boldsymbol{x}} \triangleq \begin{pmatrix} \frac{\partial f}{\partial x_0} & \cdots & \frac{\partial f}{\partial x_{N-1}} \end{pmatrix}. \tag{A.15}$$

We furthermore introduce the shorthand notation $\frac{\partial f}{\partial \boldsymbol{x}^\mathsf{T}} \triangleq \left(\frac{\partial f}{\partial \boldsymbol{x}}\right)^\mathsf{T}$.

Note that many authors, particularly from the engineering community, define $\frac{\partial f}{\partial \boldsymbol{x}}$ as a *column* vector and call it the *gradient* of f. A discussion of why (A.15) is a more natural definition is given in [66] (see also references therein), where $\frac{\partial f}{\partial \boldsymbol{x}}$ is called the *cogradient* of f. Here, the following pragmatic reasons shall suffice: with $\frac{\partial f}{\partial \boldsymbol{x}}$ defined as a row vector, the differential of f can be written in the natural form

$$\mathrm{d}f = \frac{\partial f}{\partial \boldsymbol{x}}\,\mathrm{d}\boldsymbol{x} = \sum_{n=0}^{N-1} \frac{\partial f}{\partial x_n}\,\mathrm{d}x_n. \tag{A.16}$$

Furthermore, the derivatives themselves become simpler, with fewer transpose-operators. To appreciate the difference, consider the simple example $f(\boldsymbol{x}) = \boldsymbol{a}^\mathsf{T}\boldsymbol{x} = \boldsymbol{x}^\mathsf{T}\boldsymbol{a}$. With the definitions above, we obtain $\frac{\partial \boldsymbol{a}^\mathsf{T}\boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{a}^\mathsf{T}$ and $\frac{\partial \boldsymbol{x}^\mathsf{T}\boldsymbol{a}}{\partial \boldsymbol{x}^\mathsf{T}} = \boldsymbol{a}$. Contrast this with $\frac{\partial \boldsymbol{a}^\mathsf{T}\boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{a}$ and $\frac{\partial \boldsymbol{x}^\mathsf{T}\boldsymbol{a}}{\partial \boldsymbol{x}^\mathsf{T}} = \boldsymbol{a}^\mathsf{T}$ which would be the derivatives if $\frac{\partial f}{\partial \boldsymbol{x}}$ was defined as a column vector.

In order to avoid confusion, we will use the clearly defined partial derivatives instead of the term *gradient* and the symbol ∇.

Defining the derivative of a scalar function as a row vector leads to a straightforward generalization to vector-valued functions. Let $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^M$. The derivative of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$, often called the *Jacobian*, is defined as

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \triangleq \begin{pmatrix} \frac{\partial f_0}{\partial \boldsymbol{x}} \\ \vdots \\ \frac{\partial f_{M-1}}{\partial \boldsymbol{x}} \end{pmatrix} = \begin{pmatrix} \frac{\partial f_0}{\partial x_0} & \cdots & \frac{\partial f_0}{\partial x_{N-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{M-1}}{\partial x_0} & \cdots & \frac{\partial f_{M-1}}{\partial x_{N-1}} \end{pmatrix}. \tag{A.17}$$

The derivative of a scalar function $f : \mathbb{R}^{N \times K} \to \mathbb{R}$ with respect to a matrix parameter of dimension $N \times K$ is the $K \times N$ matrix of partial derivatives

$$\frac{\partial f}{\partial \boldsymbol{X}} \triangleq \begin{pmatrix} \frac{\partial f}{\partial x_{0,0}} & \cdots & \frac{\partial f}{\partial x_{N-1,0}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial x_{0,K-1}} & \cdots & \frac{\partial f}{\partial x_{N-1,K-1}} \end{pmatrix}. \tag{A.18}$$

Again we define the shorthand $\frac{\partial f}{\partial \boldsymbol{X}^\mathsf{T}} \triangleq \left(\frac{\partial f}{\partial \boldsymbol{X}}\right)^\mathsf{T}$ and note that the differential of f is in this case given as $\mathrm{d}f = \mathrm{tr}\left(\frac{\partial f}{\partial \boldsymbol{X}}\,\mathrm{d}\boldsymbol{X}\right)$, where $\mathrm{tr}(\cdot)$ denotes the trace of a square matrix.

Hessian The *Hessian* of a scalar function is the Jacobian of its first derivative $\frac{\partial f}{\partial \boldsymbol{x}^\mathsf{T}}$:

$$\frac{\partial^2 f}{\partial \boldsymbol{x}\,\partial \boldsymbol{x}^\mathsf{T}} = \begin{pmatrix} \frac{\partial^2 f}{\partial x_0^2} & \cdots & \frac{\partial^2 f}{\partial x_{N-1}\,\partial x_0} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_0\,\partial x_{N-1}} & \cdots & \frac{\partial^2 f}{\partial x_{N-1}^2} \end{pmatrix}. \tag{A.19}$$

The Hessian is symmetric, and more generally $\frac{\partial^2 f}{\partial \boldsymbol{x}\,\partial \boldsymbol{y}^\mathsf{T}} = \left(\frac{\partial^2 f}{\partial \boldsymbol{y}\,\partial \boldsymbol{x}^\mathsf{T}}\right)^\mathsf{T}$.

A.1.3 Multivariate Complex Calculus

Jacobian and Conjugate Jacobian Now, let $f : \mathbb{C}^N \to \mathbb{C}$ be a function of a complex vector $\boldsymbol{z}$, and define $\boldsymbol{x} = \mathrm{Re}(\boldsymbol{z})$ and $\boldsymbol{y} = \mathrm{Im}(\boldsymbol{z})$. Using the derivative (A.15) with respect to a real vector, the generalized complex derivatives (A.6) can be extended to the multivariate case as

$$\frac{\partial f}{\partial \boldsymbol{z}} \triangleq \frac{1}{2}\left(\frac{\partial f}{\partial \boldsymbol{x}} - \mathrm{j}\frac{\partial f}{\partial \boldsymbol{y}}\right) \quad \text{and} \quad \frac{\partial f}{\partial \boldsymbol{z}^*} \triangleq \frac{1}{2}\left(\frac{\partial f}{\partial \boldsymbol{x}} + \mathrm{j}\frac{\partial f}{\partial \boldsymbol{y}}\right) \tag{A.20}$$

which are respectively called the *Jacobian* and *conjugate Jacobian* [67]. The differential of f is then given by the straightforward generalization of (A.9)

$$\mathrm{d}f = \frac{\partial f}{\partial \boldsymbol{z}}\,\mathrm{d}\boldsymbol{z} + \frac{\partial f}{\partial \boldsymbol{z}^*}\,\mathrm{d}\boldsymbol{z}^*. \tag{A.21}$$

We also define the shorthand notation $\frac{\partial f}{\partial \boldsymbol{z}^\mathsf{T}} \triangleq \left(\frac{\partial f}{\partial \boldsymbol{z}}\right)^\mathsf{T}$ and $\frac{\partial f}{\partial \boldsymbol{z}^\mathsf{H}} \triangleq \left(\frac{\partial f}{\partial \boldsymbol{z}^*}\right)^\mathsf{T}$. Note that in general $\frac{\partial f}{\partial \boldsymbol{z}^\mathsf{H}} \neq \left(\frac{\partial f}{\partial \boldsymbol{z}}\right)^\mathsf{H}$. Only in the special case of a real-valued f, $\frac{\partial f}{\partial \boldsymbol{z}^\mathsf{H}} = \left(\frac{\partial f}{\partial \boldsymbol{z}}\right)^\mathsf{H}$. Also note that with above definitions, $\frac{\partial f}{\partial \boldsymbol{z}}$ and $\frac{\partial f}{\partial \boldsymbol{z}^*}$ are row vectors, while $\frac{\partial f}{\partial \boldsymbol{z}^\mathsf{T}}$ and $\frac{\partial f}{\partial \boldsymbol{z}^\mathsf{H}}$ are column vectors. As an example of the simple notation which emerges from this convention, consider the quadratic form $f(\boldsymbol{z}) = \boldsymbol{z}^\mathsf{H}\boldsymbol{A}\boldsymbol{z}$, whose derivatives are $\frac{\partial f}{\partial \boldsymbol{z}^\mathsf{H}} = \boldsymbol{A}\boldsymbol{z}$ and $\frac{\partial f}{\partial \boldsymbol{z}} = \boldsymbol{z}^\mathsf{H}\boldsymbol{A}$.

Definition (A.20) can be extended to vector-valued functions $\boldsymbol{f} : \mathbb{C}^N \to \mathbb{C}^M$:

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}} \triangleq \begin{pmatrix} \frac{\partial f_0}{\partial z_0} & \cdots & \frac{\partial f_0}{\partial z_{N-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{M-1}}{\partial z_0} & \cdots & \frac{\partial f_{M-1}}{\partial z_{N-1}} \end{pmatrix} \quad \text{and} \quad \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}^*} \triangleq \begin{pmatrix} \frac{\partial f_0}{\partial z_0^*} & \cdots & \frac{\partial f_0}{\partial z_{N-1}^*} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{M-1}}{\partial z_0^*} & \cdots & \frac{\partial f_{M-1}}{\partial z_{N-1}^*} \end{pmatrix}. \tag{A.22}$$

The chain rule (A.12) also has a straightforward generalization to the multivariate case. Let $\boldsymbol{f} : \mathbb{C}^M \to \mathbb{C}^K$ and $\boldsymbol{g} : \mathbb{C}^N \to \mathbb{C}^M$, such that $\boldsymbol{f} \circ \boldsymbol{g} : \mathbb{C}^N \to \mathbb{C}^K$. Then,

$$\frac{\partial \boldsymbol{f} \circ \boldsymbol{g}}{\partial \boldsymbol{z}} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{g}}\,\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{z}} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{g}^*}\,\frac{\partial \boldsymbol{g}^*}{\partial \boldsymbol{z}} \quad \text{and} \quad \frac{\partial \boldsymbol{f} \circ \boldsymbol{g}}{\partial \boldsymbol{z}^*} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{g}}\,\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{z}^*} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{g}^*}\,\frac{\partial \boldsymbol{g}^*}{\partial \boldsymbol{z}^*}. \tag{A.23}$$

Hessian Just as there are two Jacobians, we can identify four Hessians of a scalar function $f : \mathbb{C}^N \to \mathbb{C}$

$$\frac{\partial^2 f}{\partial \boldsymbol{z}\,\partial \boldsymbol{z}^\mathsf{H}} = \begin{pmatrix} \frac{\partial^2 f}{\partial z_0 \partial z_0^*} & \cdots & \frac{\partial^2 f}{\partial z_{N-1} \partial z_0^*} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial z_0 \partial z_{N-1}^*} & \cdots & \frac{\partial^2 f}{\partial z_{N-1} \partial z_{N-1}^*} \end{pmatrix} \quad \frac{\partial^2 f}{\partial \boldsymbol{z}^*\,\partial \boldsymbol{z}^\mathsf{H}} = \begin{pmatrix} \frac{\partial^2 f}{\partial z_0^* \partial z_0^*} & \cdots & \frac{\partial^2 f}{\partial z_{N-1}^* \partial z_0^*} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial z_0^* \partial z_{N-1}^*} & \cdots & \frac{\partial^2 f}{\partial z_{N-1}^* \partial z_{N-1}^*} \end{pmatrix}$$

$$\frac{\partial^2 f}{\partial \boldsymbol{z}\,\partial \boldsymbol{z}^\mathsf{T}} = \begin{pmatrix} \frac{\partial^2 f}{\partial z_0 \partial z_0} & \cdots & \frac{\partial^2 f}{\partial z_{N-1} \partial z_0} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial z_0 \partial z_{N-1}} & \cdots & \frac{\partial^2 f}{\partial z_{N-1} \partial z_{N-1}} \end{pmatrix} \quad \frac{\partial^2 f}{\partial \boldsymbol{z}^*\,\partial \boldsymbol{z}^\mathsf{T}} = \begin{pmatrix} \frac{\partial^2 f}{\partial z_0^* \partial z_0} & \cdots & \frac{\partial^2 f}{\partial z_{N-1}^* \partial z_0} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial z_0^* \partial z_{N-1}} & \cdots & \frac{\partial^2 f}{\partial z_{N-1}^* \partial z_{N-1}} \end{pmatrix}. \tag{A.24}$$

For general complex-valued f, the Hessians satisfy the three equations

$$\frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{H}}} = \left(\frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{T}}}\right)^{\mathsf{T}}, \quad \frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{T}}} = \left(\frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{T}}}\right)^{\mathsf{T}}, \quad \frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{H}}} = \left(\frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{H}}}\right)^{\mathsf{T}}. \tag{A.25}$$

In the special case of real-valued f, we have furthermore

$$\frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{H}}} = \left(\frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{H}}}\right)^{\mathsf{H}}, \quad \frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{T}}} = \left(\frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{T}}}\right)^{\mathsf{H}}, \quad \frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{T}}} = \left(\frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{H}}}\right)^{\mathsf{H}} \tag{A.26}$$

and, combining (A.26) with (A.25), also

$$\frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{H}}} = \left(\frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{T}}}\right)^{*} \quad \text{and} \quad \frac{\partial^2 f}{\partial z^*\,\partial z^{\mathsf{H}}} = \left(\frac{\partial^2 f}{\partial z\,\partial z^{\mathsf{T}}}\right)^{*}. \tag{A.27}$$

Augmented Vectors and Matrices

Many computations that involve complex variables $z \in \mathbb{C}^N$ can be vastly simplified by introducing the *augmented vector* [115][3]

$$\underline{z} \triangleq \begin{pmatrix} z \\ z^* \end{pmatrix} \in \mathbb{C}^{2N}_* \tag{A.28}$$

where, informally, $\mathbb{C}^{2N}_* \subset \mathbb{C}^{2N}$ is the set of all vectors whose upper N entries are complex conjugates of the lower N entries. Consider the real vector

$$w \triangleq \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \mathrm{Re}(z) \\ \mathrm{Im}(z) \end{pmatrix} \in \mathbb{R}^{2N} \tag{A.29}$$

containing the real and imaginary part of z. The two vectors are related via

$$\underline{z} = J_N w \quad \text{and} \quad w = J_N^{-1}\underline{z} \tag{A.30}$$

with the transformation matrix

$$J_N \triangleq \begin{pmatrix} I_N & \mathrm{j}I_N \\ I_N & -\mathrm{j}I_N \end{pmatrix}. \tag{A.31}$$

Some useful identities are

$$J_N^{-1} = \tfrac{1}{2} J_N^{\mathsf{H}} \tag{A.32}$$

$$\det J_N = (-2\mathrm{j})^N \tag{A.33}$$

$$\det(J_N J_N^{\mathsf{H}}) = 4^N. \tag{A.34}$$

[3] In this thesis, we adopt the convention from [115] and underline all augmented vectors and matrices.

Jabobian We can now define the Jacobian of a function $\boldsymbol{f} : \mathbb{C}^N \to C^M$ with respect to the augmented parameter vector as

$$\frac{\partial \boldsymbol{f}}{\partial \underline{\boldsymbol{z}}} \triangleq \begin{pmatrix} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}} & \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}^*} \end{pmatrix} = \begin{pmatrix} \frac{\partial \boldsymbol{f}}{\partial z_0} & \cdots & \frac{\partial \boldsymbol{f}}{\partial z_{N-1}} & \frac{\partial \boldsymbol{f}}{\partial z_0^*} & \cdots & \frac{\partial \boldsymbol{f}}{\partial z_{N-1}^*} \end{pmatrix}. \tag{A.35}$$

For the special case of a scalar function $f : \mathbb{C} \to \mathbb{C}$, we also define

$$\frac{\partial f}{\partial \underline{\boldsymbol{z}}^{\mathsf{H}}} \triangleq \begin{pmatrix} \frac{\partial f}{\partial \boldsymbol{z}^{\mathsf{H}}} \\ \frac{\partial f}{\partial \boldsymbol{z}^{\mathsf{T}}} \end{pmatrix}. \tag{A.36}$$

We can furthermore augment the function itself, and write the Jacobian as

$$\frac{\partial \underline{\boldsymbol{f}}}{\partial \underline{\boldsymbol{z}}} = \begin{pmatrix} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}} & \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}^*} \\ \frac{\partial \boldsymbol{f}^*}{\partial \boldsymbol{z}} & \frac{\partial \boldsymbol{f}^*}{\partial \boldsymbol{z}^*} \end{pmatrix} = \begin{pmatrix} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}} & \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}^*} \\ \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}^*}\right)^* & \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{z}}\right)^* \end{pmatrix} \in \mathbb{C}_*^{2M \times 2N} \tag{A.37}$$

where $\mathbb{C}_*^{2M \times 2N} \subset \mathbb{C}^{2M \times 2N}$ is the set of complex matrices whose upper-left block is the complex conjugate of its lower-right block, and whose upper-right block is the complex conjugate of its lower-left block. Matrices which satisfy this pattern are called *augmented matrices*.

The differential of $\boldsymbol{f}$ is

$$\mathrm{d}\boldsymbol{f} = \frac{\partial \boldsymbol{f}}{\partial \underline{\boldsymbol{z}}}\, \mathrm{d}\underline{\boldsymbol{z}} \quad \text{or} \quad \mathrm{d}\underline{\boldsymbol{f}} = \frac{\partial \underline{\boldsymbol{f}}}{\partial \underline{\boldsymbol{z}}}\, \mathrm{d}\underline{\boldsymbol{z}} \tag{A.38}$$

which is an example of the fact that using augmented complex vectors typically results in expressions that are formally identical (or at least very similar) to their real-valued counterparts.

Hessian The Hessian of a scalar function $f : \mathbb{C}^N \to \mathbb{C}$ with respect to the augmented parameter vector contains all four Hessians from (A.24):

$$\frac{\partial^2 f}{\partial \underline{\boldsymbol{z}}\, \partial \underline{\boldsymbol{z}}^{\mathsf{H}}} = \begin{pmatrix} \frac{\partial^2 f}{\partial \boldsymbol{z}\, \partial \boldsymbol{z}^{\mathsf{H}}} & \frac{\partial^2 f}{\partial \boldsymbol{z}^*\, \partial \boldsymbol{z}^{\mathsf{H}}} \\ \frac{\partial^2 f}{\partial \boldsymbol{z}\, \partial \boldsymbol{z}^{\mathsf{T}}} & \frac{\partial^2 f}{\partial \boldsymbol{z}^*\, \partial \boldsymbol{z}^{\mathsf{T}}} \end{pmatrix}. \tag{A.39}$$

Using (A.27), we see that the Hessian of a real-valued f is an augmented matrix.

The Jacobian with respect to $\underline{\boldsymbol{z}}$ as defined above, and the standard real Jacobian with respect to $\boldsymbol{w}$ are related via

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}} = \frac{\partial \boldsymbol{f}}{\partial \underline{\boldsymbol{z}}}\, \boldsymbol{J}_N \quad \text{and} \quad \frac{\partial \boldsymbol{f}}{\partial \underline{\boldsymbol{z}}} = \frac{1}{2} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}}\, \boldsymbol{J}_N^{\mathsf{H}}. \tag{A.40}$$

Similarly, the Hessians of a scalar function are related via

$$\frac{\partial^2 f}{\partial \boldsymbol{w}\,\partial \boldsymbol{w}^{\mathsf{T}}} = \boldsymbol{J}_N^{\mathsf{H}}\,\frac{\partial^2 f}{\partial \underline{\boldsymbol{z}}\,\partial \underline{\boldsymbol{z}}^{\mathsf{H}}}\,\boldsymbol{J}_N \quad \text{and} \quad \frac{\partial^2 f}{\partial \underline{\boldsymbol{z}}\,\partial \underline{\boldsymbol{z}}^{\mathsf{H}}} = \frac{1}{4}\,\boldsymbol{J}_N\,\frac{\partial^2 f}{\partial \boldsymbol{w}\,\partial \boldsymbol{w}^{\mathsf{T}}}\,\boldsymbol{J}_N^{\mathsf{H}}. \tag{A.41}$$

It is important to note that differentiating with respect to $\underline{\boldsymbol{z}}$ or $\underline{\boldsymbol{z}}^*$ does *not* obey the same rules as differentiating with respect to $\boldsymbol{z}$ or $\boldsymbol{z}^*$. For example, consider the quadratic form $f(\underline{\boldsymbol{z}}) = \underline{\boldsymbol{z}}^{\mathsf{H}}\boldsymbol{M}\underline{\boldsymbol{z}}$ with $\boldsymbol{M} \in \mathbb{C}^{2N\times 2N}$. Then, $\frac{\partial f}{\partial \underline{\boldsymbol{z}}}$ is *not* equal to $\underline{\boldsymbol{z}}^{\mathsf{H}}\boldsymbol{M}$, as it would be if $\underline{\boldsymbol{z}}$ were an unpatterned vector from $\mathbb{C}^{2N}$. Instead, by expanding the function as

$$f(\boldsymbol{z}, \boldsymbol{z}^*) = \begin{pmatrix} \boldsymbol{z}^{\mathsf{H}} & \boldsymbol{z}^{\mathsf{T}} \end{pmatrix} \begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{pmatrix} \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{z}^* \end{pmatrix} \tag{A.42}$$

and applying the definitions above, we obtain the first two derivatives as

$$\frac{\partial f}{\partial \underline{\boldsymbol{z}}^{\mathsf{H}}} = \begin{pmatrix} \boldsymbol{A}+\boldsymbol{D}^{\mathsf{T}} & \boldsymbol{B}+\boldsymbol{B}^{\mathsf{T}} \\ \boldsymbol{C}+\boldsymbol{C}^{\mathsf{T}} & \boldsymbol{D}+\boldsymbol{A}^{\mathsf{T}} \end{pmatrix} \underline{\boldsymbol{z}} \quad \text{and} \quad \frac{\partial^2 f}{\partial \underline{\boldsymbol{z}}\,\partial \underline{\boldsymbol{z}}^{\mathsf{H}}} = \begin{pmatrix} \boldsymbol{A}+\boldsymbol{D}^{\mathsf{T}} & \boldsymbol{B}+\boldsymbol{B}^{\mathsf{T}} \\ \boldsymbol{C}+\boldsymbol{C}^{\mathsf{T}} & \boldsymbol{D}+\boldsymbol{A}^{\mathsf{T}} \end{pmatrix}. \tag{A.43}$$

It is easy to show that f is real-valued if and only if $\boldsymbol{M} \in \mathbb{C}_*^{2N\times 2N}$, i. e., $\boldsymbol{A} = \boldsymbol{D}^*$ and $\boldsymbol{B} = \boldsymbol{C}^*$. Then, the derivatives simplify to

$$\frac{\partial f}{\partial \underline{\boldsymbol{z}}^{\mathsf{H}}} = \left(\underline{\boldsymbol{M}} + \underline{\boldsymbol{M}}^{\mathsf{H}}\right) \underline{\boldsymbol{z}} \quad \text{and} \quad \frac{\partial^2 f}{\partial \underline{\boldsymbol{z}}\,\partial \underline{\boldsymbol{z}}^{\mathsf{H}}} = \underline{\boldsymbol{M}} + \underline{\boldsymbol{M}}^{\mathsf{H}} \tag{A.44}$$

which shows that the Hessian is a Hermitian augmented matrix.

Equations (A.35) and (A.39) demonstrate an important simplification that the notion of augmented vector provides: instead of having to deal with two Jacobians and four Hessians as in (A.22) and (A.24), it is sufficient to consider one Jacobian and Hessian with respect to $\underline{\boldsymbol{z}}$, which respectively contain all first and second order derivatives of f.

Chain Rule The notion of augmented vectors also simplifies the chain rule (A.23). As above, let $\boldsymbol{f} : \mathbb{C}^M \to \mathbb{C}^K$ and $\boldsymbol{g} : \mathbb{C}^N \to \mathbb{C}^M$, and define the augmented vector functions $\underline{\boldsymbol{f}}$ and $\underline{\boldsymbol{g}}$ as in (A.37). Then

$$\begin{aligned} \frac{\partial \underline{\boldsymbol{f}} \circ \underline{\boldsymbol{g}}}{\partial \underline{\boldsymbol{z}}} &= \begin{pmatrix} \frac{\partial f \circ \underline{g}}{\partial z} & \frac{\partial f \circ \underline{g}}{\partial z^*} \\ \frac{\partial f^* \circ \underline{g}}{\partial z} & \frac{\partial f^* \circ \underline{g}}{\partial z^*} \end{pmatrix} \\ &= \begin{pmatrix} \frac{\partial f}{\partial g}\frac{\partial g}{\partial z} + \frac{\partial f}{\partial g^*}\frac{\partial g^*}{\partial z} & \frac{\partial f}{\partial g}\frac{\partial g}{\partial z^*} + \frac{\partial f}{\partial g^*}\frac{\partial g^*}{\partial z^*} \\ \frac{\partial f^*}{\partial g}\frac{\partial g}{\partial z} + \frac{\partial f^*}{\partial g^*}\frac{\partial g^*}{\partial z} & \frac{\partial f^*}{\partial g}\frac{\partial g}{\partial z^*} + \frac{\partial f^*}{\partial g^*}\frac{\partial g^*}{\partial z^*} \end{pmatrix} \\ &= \begin{pmatrix} \frac{\partial f}{\partial g} & \frac{\partial f}{\partial g^*} \\ \frac{\partial f^*}{\partial g} & \frac{\partial f^*}{\partial g^*} \end{pmatrix} \begin{pmatrix} \frac{\partial g}{\partial z} & \frac{\partial g}{\partial z^*} \\ \frac{\partial g^*}{\partial z} & \frac{\partial g^*}{\partial z^*} \end{pmatrix} \end{aligned}$$

$$= \frac{\partial \underline{f}}{\partial \underline{g}} \frac{\partial \underline{g}}{\partial \underline{z}} \in \mathbb{C}_*^{2K \times 2N}. \tag{A.45}$$

Thus, by working with augmented quantities, the chain rule for complex calculus becomes formally identical to the standard chain rule of real calculus.

A.1.4 Variational Calculus

Let us go back to the discussion of multivariate real calculus, where we have studied functions $f : \mathbb{R}^N \to \mathbb{R}$ with argument $\boldsymbol{x} = (x_0, \ldots, x_{N-1})^\mathsf{T} \in \mathbb{R}^N$. We now view this vector itself as a function $g : \{0, \ldots, N-1\} \to \mathbb{R}$ with $g(n) = x_n$, such that $f : (\{0, \ldots, N-1\} \to \mathbb{R}) \to \mathbb{R}$ becomes a higher-order function or *functional*. A straightforward generalization is to replace the discrete domain of g with a continuous one, say $\mathbb{C}^K$. In the following, we will briefly discuss the calculus of functionals $F : (\mathbb{C}^K \to \mathbb{R}) \to \mathbb{R}$, which map real-valued functions g to real numbers.

Analogous to (A.16), the functional derivative $\frac{\delta F}{\delta g}$ is defined via

$$\begin{aligned} \mathrm{d}F &= \left\langle \frac{\delta F}{\delta g}, \mathrm{d}g \right\rangle \\ &\triangleq \int_{\mathbb{C}^K} \frac{\delta F}{\delta g}(\boldsymbol{z})\, \mathrm{d}g(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \end{aligned} \tag{A.46}$$

where the differential $\mathrm{d}F$ is the part of $F(g + \mathrm{d}g) - F(g)$ that is linear in $\mathrm{d}g$.

As a very simple example, consider the linear functional

$$F(g) = \sum_{i=0}^{I-1} w_i\, g(\boldsymbol{z}_i) \tag{A.47}$$

which evaluates its argument only at a discrete set of points $\{\boldsymbol{z}_0, \ldots, \boldsymbol{z}_{I-1}\}$. Then

$$\begin{aligned} \mathrm{d}F &= \sum_{i=0}^{I-1} w_i (g(\boldsymbol{z}_i) + \mathrm{d}g(\boldsymbol{z}_i)) - \sum_{i=0}^{I-1} w_i\, g(\boldsymbol{z}_i) \\ &= \sum_{i=0}^{I-1} w_i\, \mathrm{d}g(\boldsymbol{z}_i) \end{aligned} \tag{A.48}$$

and a comparison with (A.46) implies that

$$\frac{\delta F}{\delta g}(\boldsymbol{z}) = \sum_{i=0}^{I-1} w_i\, \delta(\boldsymbol{z} - \boldsymbol{z}_i). \tag{A.49}$$

Of course, such a functional is hardly interesting, as it can be reduced to a regular function $f : \mathbb{R}^I \to \mathbb{R}$ with $f(\boldsymbol{g}) = \boldsymbol{w}^\mathsf{T} \boldsymbol{g}$ and $\boldsymbol{g} = (g(\boldsymbol{z}_0), \ldots, g(\boldsymbol{z}_{I-1}))^\mathsf{T}$. Instead, we are interested in functionals which evaluate their argument at a continuum of points

$$F(g) = \int h(g(\boldsymbol{z}))\, \mathrm{d}\boldsymbol{z} \tag{A.50}$$

with some differentiable function $h : \mathbb{R} \to \mathbb{R}$. With $\simeq$ meaning "equal up to first order terms in $\mathrm{d}g$," we have

$$\mathrm{d}F = \int \frac{\delta F}{\delta g}(z)\, \mathrm{d}g(z)\, \mathrm{d}z \simeq \int h(g(z) + \mathrm{d}g(z))\, \mathrm{d}z - \int h(g(z))\, \mathrm{d}z \tag{A.51}$$

which holds in general if

$$\forall z \in \mathbb{C}^K : \quad \frac{\delta F}{\delta g}(z)\, \mathrm{d}g(z) \simeq h(g(z) + \mathrm{d}g(z)) - h(g(z)). \tag{A.52}$$

With $h'(x) \triangleq \frac{\mathrm{d}h(x)}{\mathrm{d}x}$, the functional derivative is thus found as

$$\frac{\delta F}{\delta g}(z) = h'(g(z)). \tag{A.53}$$

Example Consider the functional

$$F(p) = -\int p(z) \log p(z)\, \mathrm{d}z \tag{A.54}$$

which maps a probability distribution $p(z)$ to its differential entropy. This functional is in the form (A.50) with $h(x) = -x \log x$. Using $h'(x) = -(1 + \log x)$, the functional derivative of F is found as

$$\frac{\delta F}{\delta p}(z) = -(1 + \log p(z)). \tag{A.55}$$

A.2 Exponential Families

Exponential families are an essential ingredient of many modern inference algorithms and thus play a major role throughout this thesis. Informally, an exponential family is a set of probability distributions which share the same parametric form, and whose individual elements are identified by a parameter vector $\boldsymbol{\theta}$.

Exponential families possess two key properties which make them invaluable for the design and implementation of inference algorithms:

- Bounded space complexity. Representing an arbitrary probability density function would generally require an infinite amount of information. A member of an exponential family, however, is fully determined by the finite-dimensional parameter vector $\boldsymbol{\theta}$. (Strictly speaking, an exact representation of a continuous parameter also requires an infinite amount of memory, but it can be truncated to an appropriate fixed point data format with an arbitrarily small quantization error.)

- Bounded time complexity. Exponential families are closed under multiplication; the product of two members of an exponential family with parameters $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ is, up to a normalization constant, another member of the exponential family with parameter $\boldsymbol{\theta} + \boldsymbol{\theta}'$. This is important in practice, because an ubiquitous task in statistical inference is to combine information about an unobserved quantity obtained from different (presumably independent) sources, which requires multiplication of the corresponding distributions.
  As a very basic example, assume we are interested in an unobserved quantity x, and let y_1 denote all information available to us at a certain time instant. Our state of knowledge about x is then fully described by the distribution $p(x \mid y_1)$. Now assume that we obtain additional information y_2. Using Bayes' theorem, we update our state of knowledge about x as

$$\begin{aligned} p(x \mid y_1, y_2) &= \frac{p(y_2 \mid x, y_1)\, p(x \mid y_1)}{p(y_2 \mid y_1)} \\ &\overset{(a)}{=} \frac{p(y_2 \mid x)\, p(x \mid y_1)}{p(y_2 \mid y_1)} \end{aligned} \tag{A.56}$$

  where (a) presumes that y_1 and y_2 are statistically independent given x. In general, this computation could be arbitrarily complex. However, if $p(x \mid y_1)$ is a member of an exponential family, and if $p(y_2 \mid x)$ when taken as a function of x is (up to a constant factor) in the same exponential family, then we know that $p(x \mid y_1, y_2)$ is also a member of this exponential family, and its parameter is merely the sum of the parameters of $p(x \mid y_1)$ and $p(y_2 \mid x)$. In particular, there is no need to evaluate the normalization constant $p(y_2 \mid y_1) = \int p(y_2 \mid x)\, p(x \mid y_1)\, \mathrm{d}x$.

Of course, those favorable properties cannot *per se* justify the use of exponential families. The fundamental reason for their use is that they emerge from the *principle of maximum entropy* [62, Chapter 11–12], which states that, "subject to precisely stated prior data [...], the probability distribution which best represents the current state of knowledge is the one with largest entropy" [143].

To derive this connection, assume that we have observed some data $(x_0, \dots, x_{N-1})$ that are drawn iid from an unknown distribution $p(x)$. In order to estimate this distribution, we determine the empirical moments of some functions of the data, collected in the vector $\boldsymbol{\phi}(x) = (\phi_0(x), \dots, \phi_{K-1}(x))^\mathsf{T}$. That is, we compute

$$\boldsymbol{m} \triangleq \frac{1}{N} \sum_{n=0}^{N-1} \boldsymbol{\phi}(x_n) \tag{A.57}$$

and then discard the data. How can we obtain an estimate of $p(x)$ from $\boldsymbol{m}$? Clearly, it is desirable to select a distribution whose moments

$$\boldsymbol{\mu} \triangleq \mathbb{E}\left[\boldsymbol{\phi}(x)\right] \tag{A.58}$$

match the empirical moments $\boldsymbol{m}$, but there will generally be infinitely many such distributions. According to the principle of maximum entropy, we should then pick the one distribution which has the largest (differential) entropy among all distributions that are compatible with $\boldsymbol{m}$. The rationale is that, since entropy is a measure of randomness, this choice results in the distribution which contains the least amount of information beyond that provided by the available data.

Formally, we obtain the following optimization problem:

$$\text{maximize} \int_{\mathcal{X}} -p(x) \log p(x) \, \mathrm{d}x \tag{A.59a}$$

$$\text{subject to} \int_{\mathcal{X}} p(x) \, \mathrm{d}x = 1 \tag{A.59b}$$

$$\int_{\mathcal{X}} p(x) \boldsymbol{\phi}(x) \, \mathrm{d}x = \boldsymbol{m} \tag{A.59c}$$

where $\mathcal{X}$ is the set from which the data is drawn. Typically this is $\mathbb{R}$ or $\mathbb{C}$, but if we have additional knowledge about the possible values of x, for instance a non-negativity constraint, this can be incorporated here.

In order to solve (A.59) we introduce a Langrange multiplier λ associated to the normalization constraint (A.59b), and a vector-valued Lagrange multiplier $\boldsymbol{\theta}$ associated to the moment matching constraint (A.59c). The Lagrangian is then

$$L(p) = \int_{\mathcal{X}} p(x) \left(-\log p(x) + \lambda + \boldsymbol{\theta}^\mathsf{T} \boldsymbol{\phi}(x) \right) \, \mathrm{d}x \tag{A.60}$$

with functional derivative (see Section A.1.4)

$$\frac{\delta L}{\delta p}(x) = \left(-\log p(x) + \lambda + \boldsymbol{\theta}^\mathsf{T} \boldsymbol{\phi}(x) - 1 \right). \tag{A.61}$$

Setting $\frac{\delta L}{\delta p}(x) = 0$ for $x \in \mathcal{X}$, and with $p(x) = 0$ for $x \notin \mathcal{X}$, the maximum entropy solution is found as

$$\hat{p}(x) = \mathbb{1}[x \in \mathcal{X}] \exp\left(\boldsymbol{\theta}^{\mathsf{T}} \boldsymbol{\phi}(x) + \lambda - 1\right). \tag{A.62}$$

Ignoring the normalization constant and the constraint $x \in \mathcal{X}$ for a moment, we see that our estimate of the unknown distribution $p(x)$ has the functional form

$$\hat{p}(x) \propto \exp\left(\boldsymbol{\theta}^{\mathsf{T}} \boldsymbol{\phi}(x)\right) = \exp\left(\sum_{k=0}^{K-1} \theta_k \phi_k(x)\right) \tag{A.63}$$

where the parameters $\boldsymbol{\theta}$ must be chosen such that $\boldsymbol{\mu} = \boldsymbol{m}$. For example, we will later see that the Gaussian distribution is characterized by $\phi_0(x) = x$ and $\phi_1(x) = x^2$. Thus, if all that we know about a random variable x are its first two moments, our best guess of $p(x)$ is a Gaussian distribution.

In the remainder of this section, we will formally introduce exponential families. Then, with the concepts from the previous section on complex calculus, we will show how the notion of exponential families can easily be generalized to complex random variables and parameters, without having to resort to the isomorphism $\mathbb{C}^N \cong \mathbb{R}^{2N}$, i. e., treating N complex variables as $2N$ unrelated real variables. As argued in the previous section, this would mathematically be equivalent, but formally much more intricate. Along the way, we will discuss some important examples of exponential families which are used in this thesis.

Recommended books about statistical inference and exponential families include [13, 65, 140].

A.2.1 Univariate Distributions

An exponential family is a set of probability distributions which can be written in the form

$$p(x; \boldsymbol{\theta}) = h(x) \exp\left(\sum_{k=0}^{K-1} \theta_k \phi_k(x) - A(\boldsymbol{\theta})\right). \tag{A.64}$$

In general, $h(x)$ can be an arbitrary non-negative function. However, we will only allow h to take values in $\{0, 1\}$, because otherwise the family would not be closed under multiplication. But as argued above, this is an essential property for our purposes. The function h can thus only be used to restrict the range of x, which we denote as

$$\mathcal{X} \triangleq \{x \,|\, h(x) = 1\}. \tag{A.65}$$

To simplify the notation, we drop the explicit $h(x)$ in the following, with the implicit understanding that $p(x; \boldsymbol{\theta}) = 0$ for $x \notin \mathcal{X}$. Also, integrals $\int(\cdot)\,\mathrm{d}x$ (or sums $\sum_x(\cdot)$ in case of discrete random variables) are implicitly understood to range over the set $\mathcal{X}$.

The functions $\phi_k : \mathcal{X} \to \mathbb{R}$ are the *sufficient statistics* of the family, and the associated θ_k are called *natural* or *canonical parameters*. The claim from the beginning of this section that $p(x;\boldsymbol{\theta})p(x;\boldsymbol{\theta}') \propto p(x;\boldsymbol{\theta}+\boldsymbol{\theta}')$, i. e., that the product of two members of an exponential family is obtained by adding their natural parameters, easily follows from (A.64).

The *log-partition function* $A : \mathbb{R}^K \to \mathbb{R}$, which serves to normalize the distribution, is given by the integral

$$A(\boldsymbol{\theta}) = \log \int \exp\left(\sum_{k=0}^{K-1} \theta_k \phi_k(x)\right) \mathrm{d}x. \tag{A.66}$$

We are of course interested in natural parameters which produce normalizable distributions, i. e., parameters $\boldsymbol{\theta} \in \Omega$, where

$$\Omega \triangleq \{\boldsymbol{\theta} \mid A(\boldsymbol{\theta}) < \infty\} \tag{A.67}$$

is called the *natural parameter space*. Below it is shown that the Hessian of A is the covariance matrix of $\boldsymbol{\phi}$, so it is necessarily positive semidefinite. Thus, A is a convex function in $\boldsymbol{\theta}$, which implies that Ω is a convex set.

Besides natural parameters, exponential families can alternatively be parameterized in terms of *mean parameters*, which are defined as the expectations of the sufficient statistics

$$\boldsymbol{\mu} \triangleq \mathbb{E}\left[\boldsymbol{\phi}(x)\right] \in \mathbb{R}^K. \tag{A.68}$$

The derivative of the log-partition function provides a mapping from natural to mean parameters

$$\boldsymbol{\mu} = \frac{\partial A}{\partial \boldsymbol{\theta}^\mathsf{T}} \tag{A.69}$$

which can be shown by the following calculation

$$\begin{aligned}
\frac{\partial A}{\partial \theta_k} &= \frac{\partial}{\partial \theta_k} \log \int \exp\left(\sum_{k'=0}^{K-1} \theta_{k'} \phi_{k'}(x)\right) \mathrm{d}x \\
&= \frac{\int \exp\left(\sum_{k'=0}^{K-1} \theta_{k'} \phi_{k'}(x)\right) \phi_k(x)\, \mathrm{d}x}{\int \exp\left(\sum_{k'=0}^{K-1} \theta_{k'} \phi_{k'}(x)\right) \mathrm{d}x} \\
&= \int \exp\left(\sum_{k'=0}^{K-1} \theta_{k'} \phi_{k'}(x) - A(\boldsymbol{\theta})\right) \phi_k(x)\, \mathrm{d}x \\
&= \mathbb{E}\left[\phi_k(x)\right] = \mu_k.
\end{aligned} \tag{A.70}$$

Also, the Hessian of A equals the covariance matrix of the sufficient statistics

$$\frac{\partial^2 A}{\partial \boldsymbol{\theta}\, \partial \boldsymbol{\theta}^\mathsf{T}} = \mathrm{cov}\left[\boldsymbol{\phi}\right] \tag{A.71}$$

as can be shown continuing from (A.70)

$$\begin{aligned}\frac{\partial^2 A}{\partial\theta_l\,\partial\theta_k} &= \frac{\partial}{\partial\theta_l}\int \exp\left(\sum_{k'=0}^{K-1}\theta_{k'}\phi_{k'}(x) - A(\boldsymbol{\theta})\right)\phi_k(x)\,\mathrm{d}x \\ &= \exp\left(\sum_{k'=0}^{K-1}\theta_{k'}\phi_{k'}(x) - A(\boldsymbol{\theta})\right)\phi_k(x)\left(\phi_l(x)-\mu_l\right)\,\mathrm{d}x \\ &= \mathbb{E}\left[\phi_k(x)\phi_l(x)\right] - \mu_k\mu_l \\ &= \operatorname{cov}\left[\phi_k(x),\phi_l(x)\right]. \end{aligned} \tag{A.72}$$

The Bernoulli Distribution

The Bernoulli distribution is ubiquitous in digital communication algorithms, because it describes the statistics of binary random variables, i. e., bits. It is defined over the set $\mathcal{X} = \mathbb{F} = \{0,1\}$ and is fully described by the mean parameter

$$\mu \triangleq \mathbb{E}\left[x\right] = p(1) \in [0,1]. \tag{A.73}$$

In order to identify the natural parameters, sufficient statistics, and the log-partition function of the Bernoulli distribution, we write $p(x)$ in the form required by (A.64):

$$\begin{aligned} p(x) &= \mu^x(1-\mu)^{1-x} \\ &= \exp\left(\log(\mu)x + \log(1-\mu)(1-x)\right) \\ &= \exp\left(\log\left(\frac{\mu}{1-\mu}\right)x + \log(1-\mu)\right) \\ &= \exp\left(\lambda x - \log(1+\mathrm{e}^{\lambda})\right). \end{aligned} \tag{A.74}$$

In (A.74) we have defined the *L-value*

$$\lambda \triangleq \log\frac{\mu}{1-\mu} = \log\frac{p(1)}{p(0)} \tag{A.75}$$

from which

$$\mu = p(1) = \frac{\mathrm{e}^{\lambda}}{1+\mathrm{e}^{\lambda}} \tag{A.76}$$

$$1-\mu = p(0) = \frac{1}{1+\mathrm{e}^{\lambda}} \tag{A.77}$$

and thus

$$p(x) = \frac{\mathrm{e}^{x\lambda}}{1+\mathrm{e}^{\lambda}} \tag{A.78}$$

are easily obtained.

By comparing (A.64) and (A.74) we find that the Bernoulli distribution has $K = 1$ sufficient statistic $\phi(x) = x$, which justifies in retrospect the usage of the symbol μ for the mean of x. The corresponding natural parameter θ_0 is the L-value λ, and the natural parameter space is $\Omega = \mathbb{R} \cup \{-\infty, +\infty\}$, where $\lambda \in \{-\infty, +\infty\}$ corresponds to the degenerate cases $\mu \in \{0, 1\}$ where x is perfectly known.

The natural parameter λ and the mean parameter μ are two equivalent choices for describing a Bernoulli distribution. For practical implementations, however, the L-value is preferable, both for numerical reasons and because the combination of (presumably independent) information about a bit boils down to a summation of the L-values.

Finally, the log-partition function is $A(\lambda) = \log(1 + \mathrm{e}^{\lambda})$. By taking its first two derivatives, we find the mean and variance of x as

$$\mathbb{E}[x] = \frac{\mathrm{d}A}{\mathrm{d}\lambda} = \frac{\mathrm{e}^{\lambda}}{1 + \mathrm{e}^{\lambda}} = \mu \tag{A.79}$$

$$\mathrm{var}[x] = \frac{\mathrm{d}^2 A}{\mathrm{d}\lambda^2} = \frac{\mathrm{e}^{\lambda}}{(1 + \mathrm{e}^{\lambda})^2} = \mu(1 - \mu). \tag{A.80}$$

We denote the Bernoulli distribution with natural parameter λ as $\mathcal{B}(\lambda)$. Also, $\mathcal{B}^N(\boldsymbol{\lambda})$ with $\boldsymbol{\lambda} \in \mathbb{R}^N$ denotes the distribution of a random vector $\boldsymbol{x} \in \mathbb{F}^N$ with statistically independent components. That is, if $\boldsymbol{x} \sim \mathcal{B}^N(\boldsymbol{\lambda})$, then its joint distribution is

$$p(\boldsymbol{x}) = \prod_{n=0}^{N-1} p(x_n) = \prod_{n=0}^{N-1} \mathcal{B}(x_n \,|\, \lambda_n). \tag{A.81}$$

The Gaussian Distribution

The density of a Gaussian random variable $x \in \mathcal{X} = \mathbb{R}$ with mean μ and variance σ^2 can be written in the form (A.64) as follows

$$\begin{aligned} p(x) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \\ &= \exp\left(\frac{\mu}{\sigma^2}x - \frac{1}{2\sigma^2}x^2 - \frac{\mu^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)\right) \\ &= \exp\left(\theta_0 x + \theta_1 x^2 + \frac{\theta_0^2}{4\theta_1} + \frac{1}{2}\log\left(-\frac{\theta_1}{\pi}\right)\right). \end{aligned} \tag{A.82}$$

Thus, the Gaussian distribution is described by the $K = 2$ sufficient statistics

$$\boldsymbol{\phi}(x) = \begin{pmatrix} x \\ x^2 \end{pmatrix} \tag{A.83}$$

with natural parameters $\theta_0 = \frac{\mu}{\sigma^2}$ and $\theta_1 = -\frac{1}{2\sigma^2}$. The natural parameter space is $\Omega = \mathbb{R} \times [-\infty, 0)$, where $\theta_1 = -\infty$ corresponds to the degenerate case of $\sigma^2 = 0$.

The log-partition function can be read off from (A.82) as

$$A(\theta_0, \theta_1) = -\frac{\theta_0^2}{4\theta_1} - \frac{1}{2}\log\left(-\frac{\theta_1}{\pi}\right) \tag{A.84}$$

whose derivatives yield the expected results

$$\mathbb{E}[x] = \frac{\partial A}{\partial \theta_0} = -\frac{\theta_0}{2\theta_1} = \mu \tag{A.85}$$

$$\mathbb{E}\left[x^2\right] = \frac{\partial A}{\partial \theta_1} = \frac{\theta_0^2}{4\theta_1^2} - \frac{1}{2\theta_1} = \mu^2 + \sigma^2. \tag{A.86}$$

We denote the Gaussian distribution with mean μ and variance σ^2 as $\mathcal{N}(\mu, \sigma^2)$.

The Truncated Gaussian Distribution

A truncated Gaussian distribution is obtained by restricting the range of a Gaussian variable to some interval. In this thesis, only one kind of truncation is used, namely the restriction to $\mathcal{X} = [0, \infty)$, so we will simply speak of *the* truncated Gaussian distribution.

Let

$$f_{\mathcal{N}}(x) \triangleq \frac{\exp\left(-\frac{1}{2}x^2\right)}{\sqrt{2\pi}} \quad \text{and} \quad F_{\mathcal{N}}(x) \triangleq \int_{-\infty}^{x} f_{\mathcal{N}}(\xi)\,\mathrm{d}\xi \tag{A.87}$$

respectively denote the PDF and CDF of the $\mathcal{N}(0,1)$ distribution. Then, the PDF of the truncated Gaussian distribution with parameters m and s^2 is

$$\begin{aligned} p(x) &= \frac{\mathbb{1}[x \geq 0]}{F_{\mathcal{N}}(\frac{m}{s})\sqrt{2\pi s^2}} \exp\left(-\frac{(x-m)^2}{2s^2}\right) \\ &= \mathbb{1}[x \geq 0] \exp\left(\frac{m}{s^2}x - \frac{1}{2s^2}x^2 - \frac{m^2}{2s^2} - \frac{1}{2}\log(2\pi s^2) - \log F_{\mathcal{N}}\left(\frac{m}{s}\right)\right) \\ &= \mathbb{1}[x \geq 0] \exp\left(\theta_0 x + \theta_1 x^2 + \frac{\theta_0^2}{4\theta_1} + \frac{1}{2}\log\left(-\frac{\theta_1}{\pi}\right) - \log F_{\mathcal{N}}\left(\frac{\theta_0}{\sqrt{-2\theta_1}}\right)\right) \end{aligned} \tag{A.88}$$

which has the same sufficient statistics as the non-truncated Gaussian distribution, but a different range (i. e., a different h). The natural parameters are analogously to the non-truncated case $\theta_0 = \frac{m}{s^2}$ and $\theta_1 = -\frac{1}{2s^2}$. The log-partition function

$$A(\theta_0, \theta_1) = -\frac{\theta_0^2}{4\theta_1} - \frac{1}{2}\log\left(-\frac{\theta_1}{\pi}\right) + \log F_{\mathcal{N}}\left(\frac{\theta_0}{\sqrt{-2\theta_1}}\right) \tag{A.89}$$

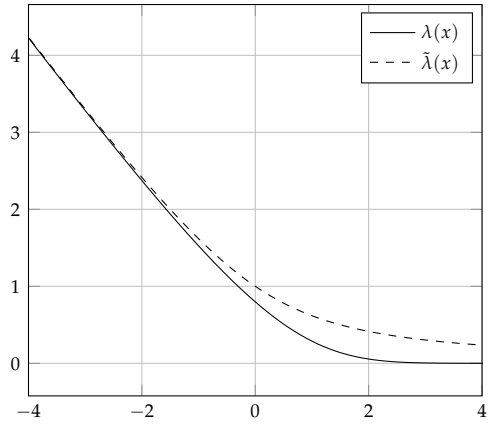


Figure A.1: The function $\lambda(x) = \frac{f_{\mathcal{N}}(x)}{F_{\mathcal{N}}(x)}$ needed for evaluating the moments of the truncated Gaussian distribution, and its approximation $\tilde{\lambda}(x) = \frac{1}{2}\sqrt{x^2+4} - \frac{x}{2}(x \lesssim -3)$

contains an additional term, compared to the non-truncated case. Its derivatives yield the mean parameters

$$\begin{aligned}\mathbb{E}[x] = \frac{\partial A}{\partial \theta_0} &= -\frac{\theta_0}{2\theta_1} + \frac{1}{\sqrt{-2\theta_1}} \frac{f_{\mathcal{N}}\left(\frac{\theta_0}{\sqrt{-2\theta_1}}\right)}{F_{\mathcal{N}}\left(\frac{\theta_0}{\sqrt{-2\theta_1}}\right)} \\ &= m + s\,\lambda\left(\frac{m}{s}\right) \end{aligned} \tag{A.90}$$

and

$$\begin{aligned}\mathbb{E}\left[x^2\right] = \frac{\partial A}{\partial \theta_1} &= \frac{\theta_0^2}{4\theta_1^2} - \frac{1}{2\theta_1} - \frac{\theta_0}{2\theta_1\sqrt{-2\theta_1}} \frac{f_{\mathcal{N}}\left(\frac{\theta_0}{\sqrt{-2\theta_1}}\right)}{F_{\mathcal{N}}\left(\frac{\theta_0}{\sqrt{-2\theta_1}}\right)} \\ &= m^2 + s^2 + m\,s\,\lambda\left(\frac{m}{s}\right) \end{aligned} \tag{A.91}$$

with

$$\lambda(x) \triangleq \frac{f_{\mathcal{N}}(x)}{F_{\mathcal{N}}(x)}. \tag{A.92}$$

For $x \lesssim -3$, a good approximation is given by $\tilde{\lambda}(x) \triangleq \frac{1}{2}\sqrt{x^2+4} - \frac{x}{2}$, and for $x \gtrsim 3$, $\lambda(x) \approx 0$. In between, for $|x| < 3$, the function can be implemented with a small lookup-table. A plot of λ and $\tilde{\lambda}$ is shown in Figure A.1.

We denote the truncated Gaussian with parameters m and s^2 as $\mathcal{TN}(m, s^2)$.

The Gamma Distribution

The gamma distribution is a density on $\mathcal{X} = [0, \infty)$. It arises for instance in SNR estimation problems, because it is the conjugate prior of the precision (inverse variance) of the Gaussian distribution. Its PDF is given as

$$\begin{aligned} p(x) &= \mathbb{1}[x \geq 0] \, \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} \mathrm{e}^{-\beta x} \\ &= \mathbb{1}[x \geq 0] \exp\big((\alpha - 1)\log x - \beta x + \alpha \log \beta - \log \Gamma(\alpha)\big) \\ &= \mathbb{1}[x \geq 0] \exp\big(\theta_0 \log x + \theta_1 x + (\theta_0 + 1)\log(-\theta_1) - \log \Gamma(\theta_0 + 1)\big) \end{aligned} \tag{A.93}$$

which is described by the $K = 2$ sufficient statistics

$$\boldsymbol{\phi}(x) = \begin{pmatrix} \log x \\ x \end{pmatrix} \tag{A.94}$$

with natural parameters $\theta_0 = \alpha - 1$ and $\theta_1 = -\beta$. The natural parameter space is $\Omega = (-1, \infty) \times (-\infty, 0)$, corresponding to the conditions $\alpha > 0$ and $\beta > 0$.

The log-partition function is

$$A(\theta_0, \theta_1) = \log \Gamma(\theta_0 + 1) - (\theta_0 + 1)\log(-\theta_1) \tag{A.95}$$

from which we obtain

$$\mathbb{E}[\log x] = \frac{\partial A}{\partial \theta_0} = \frac{\frac{\mathrm{d}}{\mathrm{d}\theta_0}\Gamma(\theta_0 + 1)}{\Gamma(\theta_0 + 1)} - \log(-\theta_1) = \psi(\alpha) - \log \beta \tag{A.96}$$

$$\mathbb{E}[x] = \frac{\partial A}{\partial \theta_1} = \frac{\theta_0 + 1}{-\theta_1} = \frac{\alpha}{\beta} \tag{A.97}$$

$$\operatorname{var}[x] = \frac{\partial^2 A}{\partial \theta_1^2} = \frac{\theta_0 + 1}{\theta_1^2} = \frac{\alpha}{\beta^2} \tag{A.98}$$

where $\psi(\alpha) \triangleq \frac{\frac{\mathrm{d}}{\mathrm{d}\alpha}\Gamma(\alpha)}{\Gamma(\alpha)}$ is the digamma function.

We denote the gamma distribution with parameters α and β as $\mathcal{G}(\alpha, \beta)$.

A.2.2 Multivariate Distributions

In principle, (A.64) also includes multivariate distributions. We just take x as a random vector (and thus write it in boldface in the following), and define the sufficient statistic vector appropriately. For instance, a Gaussian random vector $\boldsymbol{x} \in \mathbb{R}^N$ would be described by $K = N + \frac{1}{2}N(N+1)$ sufficient statistics, including the linear terms $x_0, \ldots, x_{N-1}$, the quadratic terms $x_0^2, \ldots, x_{N-1}^2$, and the mixed terms $x_m x_n$ for $m < n$.

Notationally, however, it is more convenient to allow for vector- or matrix-valued sufficient statistics and natural parameters, and rewrite (A.64) as

$$p(\boldsymbol{x};\boldsymbol{\theta}_0,\ldots,\boldsymbol{\theta}_{K-1}) = h(\boldsymbol{x}) \exp\left(\sum_{k=0}^{K-1} \langle \boldsymbol{\theta}_k, \boldsymbol{\phi}_k(\boldsymbol{x})\rangle - A(\boldsymbol{\theta}_0,\ldots,\boldsymbol{\theta}_{K-1})\right) \tag{A.99}$$

with the Euclidean inner product $\langle \boldsymbol{x},\boldsymbol{y}\rangle \triangleq \boldsymbol{y}^\mathsf{T}\boldsymbol{x}$, or for matrices $\langle \boldsymbol{X},\boldsymbol{Y}\rangle \triangleq \operatorname{tr}(\boldsymbol{X}\boldsymbol{Y}^\mathsf{T})$, whose derivatives are simply

$$\frac{\partial \langle \boldsymbol{x},\boldsymbol{y}\rangle}{\partial \boldsymbol{x}^\mathsf{T}} = \boldsymbol{y} \quad \text{and} \quad \frac{\partial \langle \boldsymbol{X},\boldsymbol{Y}\rangle}{\partial \boldsymbol{X}^\mathsf{T}} = \boldsymbol{Y}. \tag{A.100}$$

As a straightforward generalization of the univariate case, the vector derivatives of A yield a mapping from natural to mean parameters

$$\begin{aligned}
\frac{\partial A}{\partial \boldsymbol{\theta}_k^\mathsf{T}} &= \frac{\partial}{\partial \boldsymbol{\theta}_k^\mathsf{T}} \log \int \exp\left(\sum_{k'=0}^{K-1} \langle \boldsymbol{\theta}_{k'}, \boldsymbol{\phi}_{k'}(\boldsymbol{x})\rangle\right) \mathrm{d}\boldsymbol{x} \\
&= \frac{\int \exp\left(\sum_{k'=0}^{K-1} \langle \boldsymbol{\theta}_{k'}, \boldsymbol{\phi}_{k'}(\boldsymbol{x})\rangle\right) \boldsymbol{\phi}_k(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}{\int \exp\left(\sum_{k'=0}^{K-1} \langle \boldsymbol{\theta}_{k'}, \boldsymbol{\phi}_{k'}(\boldsymbol{x})\rangle\right) \mathrm{d}\boldsymbol{x}} \\
&= \mathbb{E}\left[\boldsymbol{\phi}_k(\boldsymbol{x})\right] = \boldsymbol{\mu}_k
\end{aligned} \tag{A.101}$$

while the Hessians of A are the covariance matrices of the sufficient statistics

$$\begin{aligned}
\frac{\partial^2 A}{\partial \boldsymbol{\theta}_l\,\partial \boldsymbol{\theta}_k^\mathsf{T}} &= \frac{\partial}{\partial \boldsymbol{\theta}_l} \int \exp\left(\sum_{k'=0}^{K-1} \langle \boldsymbol{\theta}_{k'}, \boldsymbol{\phi}_{k'}(\boldsymbol{x})\rangle - A(\boldsymbol{\theta}_0,\ldots,\boldsymbol{\theta}_{K-1})\right) \boldsymbol{\phi}_k(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\
&= \int p(\boldsymbol{x}) \boldsymbol{\phi}_k(\boldsymbol{x}) \left(\boldsymbol{\phi}_l^\mathsf{T}(\boldsymbol{x}) - \frac{\partial A}{\partial \boldsymbol{\theta}_l}\right) \mathrm{d}\boldsymbol{x} \\
&= \mathbb{E}\left[\boldsymbol{\phi}_k(\boldsymbol{x})\boldsymbol{\phi}_l^\mathsf{T}(\boldsymbol{x})\right] - \boldsymbol{\mu}_k\boldsymbol{\mu}_l^\mathsf{T} \\
&= \operatorname{cov}\left[\boldsymbol{\phi}_k(\boldsymbol{x}), \boldsymbol{\phi}_l^\mathsf{T}(\boldsymbol{x})\right].
\end{aligned} \tag{A.102}$$

An important subtlety here is that above derivation assumes that the parameters $\boldsymbol{\theta}_k$ are *unpatterned*. Thus, in order to find the mean parameters, unpatterned derivatives of A need to be computed, even if the parameters happen to be patterned. This will be the case in the following example, where the matrix paramter $\boldsymbol{\Theta}_1$ is constrained to be negative definite.

The Multivariate Gaussian Distribution

The density of a Gaussian random vector $\boldsymbol{x} \in \mathbb{R}^N$ with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ can be written in the canonical form (A.99) as follows

$$\begin{aligned} p(\boldsymbol{x}) &= \frac{\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)}{(2\pi)^{\frac{N}{2}}\sqrt{\det\boldsymbol{\Sigma}}} \\ &= \exp\left(\boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{x} - \tfrac{1}{2}\operatorname{tr}(\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\boldsymbol{x}^\mathsf{T}) - \tfrac{1}{2}\boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} - \tfrac{1}{2}\log\det\boldsymbol{\Sigma} - \tfrac{N}{2}\log(2\pi)\right) \\ &= \exp\left(\langle\boldsymbol{\theta}_0, \boldsymbol{x}\rangle + \left\langle\boldsymbol{\Theta}_1, \boldsymbol{x}\boldsymbol{x}^\mathsf{T}\right\rangle + \tfrac{1}{4}\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{\Theta}_1^{-1}\boldsymbol{\theta}_0 + \tfrac{1}{2}\log\det(-2\boldsymbol{\Theta}_1) - \tfrac{N}{2}\log(2\pi)\right). \end{aligned} \tag{A.103}$$

As in the univariate case, we have $K = 2$ sufficient statistics, the vector-valued $\boldsymbol{\phi}_0 = \boldsymbol{x}$ and the matrix-valued $\boldsymbol{\Phi}_1 = \boldsymbol{x}\boldsymbol{x}^\mathsf{T}$. The corresponding natural parameters are $\boldsymbol{\theta}_0 = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$ and $\boldsymbol{\Theta}_1 = -\frac{1}{2}\boldsymbol{\Sigma}^{-1}$, from which $\boldsymbol{\mu} = -\frac{1}{2}\boldsymbol{\Theta}_1^{-1}\boldsymbol{\theta}_0$ and $\boldsymbol{\Sigma} = -\frac{1}{2}\boldsymbol{\Theta}_1^{-1}$ are easily obtained. The natural parameter space is $\Omega = \mathbb{R}^N \times \mathbb{R}^{N\times N}_{\prec 0}$, where $\mathbb{R}^{N\times N}_{\prec 0}$ is the set of negative definite matrices of dimension $N \times N$.

The log-partition function can be read off as

$$A(\boldsymbol{\theta}_0, \boldsymbol{\Theta}_1) = -\tfrac{1}{4}\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{\Theta}_1^{-1}\boldsymbol{\theta}_0 - \tfrac{1}{2}\log\det(-2\boldsymbol{\Theta}_1) + \tfrac{N}{2}\log(2\pi). \tag{A.104}$$

With the matrix derivatives (see e. g. [90])

$$\frac{\partial \boldsymbol{a}^\mathsf{T}\boldsymbol{X}^{-1}\boldsymbol{a}}{\partial \boldsymbol{X}} = -\boldsymbol{X}^{-1}\boldsymbol{a}\boldsymbol{a}^\mathsf{T}\boldsymbol{X}^{-1} \quad \text{and} \quad \frac{\partial \log\det \boldsymbol{X}}{\partial \boldsymbol{X}} = \boldsymbol{X}^{-1} \tag{A.105}$$

and noting that $\boldsymbol{\Theta}_1 = \boldsymbol{\Theta}_1^\mathsf{T}$, the derivatives of the log-partition function yield the moments

$$\mathbb{E}[\boldsymbol{x}] = \frac{\partial A}{\partial \boldsymbol{\theta}_0^\mathsf{T}} = -\tfrac{1}{2}\boldsymbol{\Theta}_1^{-1}\boldsymbol{\theta}_0 = \boldsymbol{\mu} \tag{A.106}$$

$$\operatorname{cov}[\boldsymbol{x}] = \frac{\partial^2 A}{\partial \boldsymbol{\theta}_0 \, \partial \boldsymbol{\theta}_0^\mathsf{T}} = -\tfrac{1}{2}\boldsymbol{\Theta}_1^{-1} = \boldsymbol{\Sigma} \tag{A.107}$$

$$\mathbb{E}\left[\boldsymbol{x}\boldsymbol{x}^\mathsf{T}\right] = \frac{\partial A}{\partial \boldsymbol{\Theta}_1^\mathsf{T}} = \tfrac{1}{4}\boldsymbol{\Theta}_1^{-1}\boldsymbol{\theta}_0\boldsymbol{\theta}_0^\mathsf{T}\boldsymbol{\Theta}_1^{-1} - \tfrac{1}{2}\boldsymbol{\Theta}_1^{-1} = \boldsymbol{\mu}\boldsymbol{\mu}^\mathsf{T} + \boldsymbol{\Sigma}. \tag{A.108}$$

Note that the standard unpatterned derivative has been used in the last equation, even though $\boldsymbol{\Theta}_1$ is negative definite and thus symmetric.

We denote the multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ as $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

A.2.3 Extension to Complex Random Variables

In this thesis, we are often concerned with complex-valued quantities. Since the distribution of a complex random vector $\boldsymbol{z} \in \mathbb{C}^N$ is defined via the joint distribution of its real and imaginary part, (A.99) is sufficient for describing the distribution of $\boldsymbol{z}$; we just have to set $\boldsymbol{x} = (\mathrm{Re}(\boldsymbol{z})^\mathsf{T}, \mathrm{Im}(\boldsymbol{z})^\mathsf{T})^\mathsf{T} \in \mathbb{R}^{2N}$. However, doing so typically results in awkward expressions since it discards the tight coupling of the real and imaginary parts. We are therefore interested in a description of exponential families for complex random variables directly in the complex domain.

As discussed in Section A.1.3 on complex calculus, the notion of augmented vectors constitutes a systematic way of extending expressions from the real to the complex case. Recall that the augmented vector for some $\boldsymbol{z} \in \mathbb{C}^N$ is defined as

$$\underline{\boldsymbol{z}} \triangleq \begin{pmatrix} \boldsymbol{z} \\ \boldsymbol{z}^* \end{pmatrix} = \begin{pmatrix} \boldsymbol{I}_N & \mathrm{j}\boldsymbol{I}_N \\ \boldsymbol{I}_N & -\mathrm{j}\boldsymbol{I}_N \end{pmatrix} \begin{pmatrix} \mathrm{Re}(\boldsymbol{z}) \\ \mathrm{Im}(\boldsymbol{z}) \end{pmatrix} = \boldsymbol{J}_N \boldsymbol{w} \tag{A.109}$$

where $\boldsymbol{J}_N$ and $\boldsymbol{w} \in \mathbb{R}^{2N}$ are implicitly defined via the above equation.

In this thesis, the inner product of complex vectors or matrices is defined as linear in the first and conjugate-linear in the second argument. Thus, for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^N$ and $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{C}^{N \times M}$ we have

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle \triangleq \boldsymbol{y}^\mathsf{H} \boldsymbol{x} \quad \text{and} \quad \langle \boldsymbol{X}, \boldsymbol{Y} \rangle \triangleq \mathrm{tr}(\boldsymbol{X}\boldsymbol{Y}^\mathsf{H}). \tag{A.110}$$

The inner product of augmented vectors $\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}} \in \mathbb{C}_*^{2N}$ or matrices $\underline{\boldsymbol{X}}, \underline{\boldsymbol{Y}} \in \mathbb{C}_*^{2N \times 2M}$ is real-valued, thus commutative, and given as

$$\langle \underline{\boldsymbol{x}}, \underline{\boldsymbol{y}} \rangle = 2\,\mathrm{Re}(\langle \boldsymbol{x}, \boldsymbol{y} \rangle) \quad \text{and} \quad \langle \underline{\boldsymbol{X}}, \underline{\boldsymbol{Y}} \rangle = 2\,\mathrm{Re}\left(\langle \boldsymbol{X}, \boldsymbol{Y} \rangle + \langle \tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}} \rangle\right) \tag{A.111}$$

where

$$\underline{\boldsymbol{x}} = \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{x}^* \end{pmatrix}, \quad \underline{\boldsymbol{y}} = \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{y}^* \end{pmatrix}, \quad \underline{\boldsymbol{X}} = \begin{pmatrix} \boldsymbol{X} & \tilde{\boldsymbol{X}} \\ \tilde{\boldsymbol{X}}^* & \boldsymbol{X}^* \end{pmatrix}, \quad \underline{\boldsymbol{Y}} = \begin{pmatrix} \boldsymbol{Y} & \tilde{\boldsymbol{Y}} \\ \tilde{\boldsymbol{Y}}^* & \boldsymbol{Y}^* \end{pmatrix}. \tag{A.112}$$

Equipped with this inner product, a straightforward generalization of (A.99) to complex random variables is

$$p(\underline{\boldsymbol{z}}; \underline{\boldsymbol{\theta}}_0, \ldots, \underline{\boldsymbol{\theta}}_{K-1}) = h(\underline{\boldsymbol{z}}) \exp\left(\sum_{k=0}^{K-1} \left\langle \underline{\boldsymbol{\theta}}_k, \underline{\boldsymbol{\phi}}_k(\underline{\boldsymbol{z}}) \right\rangle - A(\underline{\boldsymbol{\theta}}_0, \ldots, \underline{\boldsymbol{\theta}}_{K-1}) \right). \tag{A.113}$$

With definition (A.36), the derivatives of the inner products in (A.111) are

$$\frac{\partial \langle \underline{\boldsymbol{x}}, \underline{\boldsymbol{y}} \rangle}{\partial \underline{\boldsymbol{x}}^\mathsf{H}} = \underline{\boldsymbol{y}} \quad \text{and} \quad \frac{\partial \langle \underline{\boldsymbol{X}}, \underline{\boldsymbol{Y}} \rangle}{\partial \underline{\boldsymbol{X}}^\mathsf{H}} = \underline{\boldsymbol{Y}} \tag{A.114}$$

analogously to the real case. Therefore, Equations (A.101) and (A.102) carry over to the complex case as

$$\frac{\partial A}{\partial \underline{\boldsymbol{\theta}}_k^{\mathsf{H}}} = \mathbb{E}\left[\underline{\boldsymbol{\phi}}_k(\underline{z})\right] = \underline{\boldsymbol{\mu}}_k \quad \text{and} \quad \frac{\partial^2 A}{\partial \underline{\boldsymbol{\theta}}_l \, \partial \underline{\boldsymbol{\theta}}_k^{\mathsf{H}}} = \operatorname{cov}\left[\underline{\boldsymbol{\phi}}_k(\underline{z}), \underline{\boldsymbol{\phi}}_l^{\mathsf{H}}(\underline{z})\right] \tag{A.115}$$

which can be shown analogously to (A.101) and (A.102).

The von Mises Distribution

The von Mises distribution, also called the circular normal distribution, often occurs naturally when we are dealing with random variables that describe angles. In this thesis, it is used for the estimation of phase noise processes.

A real random variable x with domain $\mathcal{X} = [-\pi, \pi)$ (or any interval of length 2π) is von Mises distributed if its density is

$$p(x) = \frac{\exp\left(\kappa \cos(x - \mu)\right)}{2\pi \, \mathrm{I}_0(\kappa)} \tag{A.116}$$

where $\mathrm{I}_n(x)$ is the modified Bessel function of the first kind and nth order. The parameter $\mu \in [-\pi, \pi)$ specifies the mode of the distribution, while $\kappa \geq 0$ determines its variance. For $\kappa = 0$, $p(x)$ becomes a uniform distribution (note that $\mathrm{I}_0(0) = 1$). For large κ (say, $\kappa > 5$), the von Mises distribution converges to a Gaussian distribution with mean μ and variance κ^{-1}, which can be seen by using the approximations $\cos(x) \approx 1 - \frac{x^2}{2}$ and $\mathrm{I}_0(x) \approx \frac{\exp(x)}{\sqrt{2\pi x}}$ in (A.116).

Rather than working with the real-valued *phase* x, it is often more convenient to consider the complex *phasor* $z \triangleq \mathrm{e}^{\mathrm{j}x}$ on the unit circle instead. To this end, we also merge the real parameters κ and μ into the complex parameter

$$\zeta \triangleq \kappa \, \mathrm{e}^{\mathrm{j}\mu} \tag{A.117}$$

and rewrite (A.116) as

$$\begin{aligned} p(z) &= \frac{\exp\left(\kappa \operatorname{Re}\left(\mathrm{e}^{\mathrm{j}(x-\mu)}\right)\right)}{2\pi \, \mathrm{I}_0(\kappa)} \\ &= \frac{\exp\left(\operatorname{Re}(\langle \zeta, z \rangle)\right)}{2\pi \, \mathrm{I}_0(|\zeta|)}. \end{aligned} \tag{A.118}$$

With augmented vectors $\underline{z} = (z, z^*)^{\mathsf{T}}$ and $\underline{\zeta} = (\zeta, \zeta^*)^{\mathsf{T}}$ we finally obtain the canonical form (A.113)

$$p(\underline{z}) = \exp\left(\left\langle \tfrac{1}{2}\underline{\zeta}, \underline{z} \right\rangle - \log \mathrm{I}_0(\sqrt{\zeta\zeta^*}) - \log(2\pi)\right) \tag{A.119}$$

with $K = 1$ sufficient statistic $\underline{\phi}(\underline{z}) = \underline{z}$. The corresponding natural parameter is $\underline{\theta} = \frac{1}{2}\underline{\zeta}$ with natural parameter space $\Omega = \mathbb{C}_*^2$.

From the log-partition function $A(\underline{\zeta}) = \log \mathrm{I}_0(\sqrt{\zeta\zeta^*}) - \log(2\pi)$, and using the derivative $\frac{\mathrm{d}}{\mathrm{d}x}\mathrm{I}_0(x) = \mathrm{I}_1(x)$, we find the first moment of $\underline{z}$ as

$$\mathbb{E}[\underline{z}] = \frac{\partial A}{\partial \underline{\theta}^{\mathsf{H}}} = \begin{pmatrix} \mathbb{E}[z] \\ \mathbb{E}[z^*] \end{pmatrix} = \begin{pmatrix} 2\frac{\partial A}{\partial \zeta^*} \\ 2\frac{\partial A}{\partial \zeta} \end{pmatrix} = \begin{pmatrix} \frac{\mathrm{I}_1(|\zeta|)}{\mathrm{I}_0(|\zeta|)}\frac{\zeta}{|\zeta|} \\ \frac{\mathrm{I}_1(|\zeta|)}{\mathrm{I}_0(|\zeta|)}\frac{\zeta^*}{|\zeta|}. \end{pmatrix} \tag{A.120}$$

We denote the von Mises distribution with parameter $\zeta = \kappa\,\mathrm{e}^{\mathrm{j}\mu}$ as $\mathcal{M}(\zeta)$.

The Complex Gaussian Distribution

We first derive the density of a general (i. e., possibly improper) complex Gaussian random vector [137]. Let $\boldsymbol{w} \in \mathbb{R}^{2N}$ be a real Gaussian random vector with mean $\boldsymbol{\nu} \triangleq \mathbb{E}[\boldsymbol{w}]$ and covariance matrix $\boldsymbol{\Gamma} \triangleq \mathbb{E}\left[(\boldsymbol{w}-\boldsymbol{\nu})(\boldsymbol{w}-\boldsymbol{\nu})^{\mathsf{T}}\right]$, whose density is

$$p(\boldsymbol{w}) = \frac{\exp\left(-\frac{1}{2}(\boldsymbol{w}-\boldsymbol{\nu})^{\mathsf{T}}\boldsymbol{\Gamma}^{-1}(\boldsymbol{w}-\boldsymbol{\nu})\right)}{(2\pi)^N\sqrt{\det\boldsymbol{\Gamma}}}. \tag{A.121}$$

Now we take $\boldsymbol{w} = \boldsymbol{J}_N^{-1}\underline{\boldsymbol{z}} = \frac{1}{2}\boldsymbol{J}_N^{\mathsf{H}}\underline{\boldsymbol{z}}$ as in (A.30) and define the mean $\underline{\boldsymbol{\mu}} \triangleq \mathbb{E}[\underline{\boldsymbol{z}}] = \boldsymbol{J}_N\boldsymbol{\nu}$ and the covariance $\underline{\boldsymbol{\Sigma}} \triangleq \mathbb{E}\left[(\underline{\boldsymbol{z}}-\underline{\boldsymbol{\mu}})(\underline{\boldsymbol{z}}-\underline{\boldsymbol{\mu}})^{\mathsf{H}}\right] = \boldsymbol{J}_N\boldsymbol{\Gamma}\boldsymbol{J}_N^{\mathsf{H}}$. From this definition we obtain $\boldsymbol{\Gamma}^{-1} = \boldsymbol{J}_N^{\mathsf{H}}\underline{\boldsymbol{\Sigma}}^{-1}\boldsymbol{J}_N$ and $\det\boldsymbol{\Gamma} = \det\boldsymbol{J}_N^{-1}\det\underline{\boldsymbol{\Sigma}}\det\boldsymbol{J}_N^{-\mathsf{H}} = 4^{-N}\det\underline{\boldsymbol{\Sigma}}$. Plugging everything into (A.121) yields the density of the augmented complex Gaussian vector

$$p(\underline{\boldsymbol{z}}) = \frac{\exp\left(-\frac{1}{2}(\underline{\boldsymbol{z}}-\underline{\boldsymbol{\mu}})^{\mathsf{H}}\underline{\boldsymbol{\Sigma}}^{-1}(\underline{\boldsymbol{z}}-\underline{\boldsymbol{\mu}})\right)}{\pi^N\sqrt{\det\underline{\boldsymbol{\Sigma}}}}. \tag{A.122}$$

$\underline{\boldsymbol{\Sigma}}$ is called the *augmented covariance matrix* of $\boldsymbol{z}$ [115] because $\underline{\boldsymbol{\Sigma}} \in \mathbb{C}_*^{2N\times 2N}$. This can be seen by writing it as a block matrix

$$\underline{\boldsymbol{\Sigma}} = \begin{pmatrix} \mathbb{E}\left[(\boldsymbol{z}-\boldsymbol{\mu})(\boldsymbol{z}-\boldsymbol{\mu})^{\mathsf{H}}\right] & \mathbb{E}\left[(\boldsymbol{z}-\boldsymbol{\mu})(\boldsymbol{z}-\boldsymbol{\mu})^{\mathsf{T}}\right] \\ \mathbb{E}\left[(\boldsymbol{z}-\boldsymbol{\mu})^*(\boldsymbol{z}-\boldsymbol{\mu})^{\mathsf{H}}\right] & \mathbb{E}\left[(\boldsymbol{z}-\boldsymbol{\mu})^*(\boldsymbol{z}-\boldsymbol{\mu})^{\mathsf{T}}\right] \end{pmatrix} = \begin{pmatrix} \boldsymbol{\Sigma} & \tilde{\boldsymbol{\Sigma}} \\ \tilde{\boldsymbol{\Sigma}}^* & \boldsymbol{\Sigma}^* \end{pmatrix}. \tag{A.123}$$

$\boldsymbol{\Sigma}$ is the regular covariance matrix of $\boldsymbol{z}$, while $\tilde{\boldsymbol{\Sigma}}$ is the cross-covariance of $\boldsymbol{z}$ and $\boldsymbol{z}^*$, and is called the *complementary covariance matrix* [72] or the *pseudo-covariance matrix* [98]. If $\tilde{\boldsymbol{\Sigma}} = \boldsymbol{0}$, $\boldsymbol{z}$ is *proper* [98], and the density (A.122) simplifies to the well-known form

$$p(\boldsymbol{z}) = \frac{\exp\left(-(\boldsymbol{z}-\boldsymbol{\mu})^{\mathsf{H}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu})\right)}{\pi^N\det\boldsymbol{\Sigma}}. \tag{A.124}$$

The density (A.122) can be expressed in the canonical form (A.113) as

$$\begin{aligned} p(\underline{\boldsymbol{z}}) &= \exp\left(\underline{\boldsymbol{\mu}}^{\mathsf{H}}\underline{\boldsymbol{\Sigma}}^{-1}\underline{\boldsymbol{z}} - \tfrac{1}{2}\operatorname{tr}(\underline{\boldsymbol{\Sigma}}^{-1}\underline{\boldsymbol{z}}\underline{\boldsymbol{z}}^{\mathsf{H}}) - \tfrac{1}{2}\underline{\boldsymbol{\mu}}^{\mathsf{H}}\underline{\boldsymbol{\Sigma}}^{-1}\underline{\boldsymbol{\mu}} - \tfrac{1}{2}\log\det\underline{\boldsymbol{\Sigma}} - N\log\pi\right) \\ &= \exp\left(\langle\underline{\boldsymbol{\theta}}_0, \underline{\boldsymbol{z}}\rangle + \left\langle\underline{\boldsymbol{\Theta}}_1, \underline{\boldsymbol{z}}\underline{\boldsymbol{z}}^{\mathsf{H}}\right\rangle + \tfrac{1}{4}\underline{\boldsymbol{\theta}}_0^{\mathsf{H}}\underline{\boldsymbol{\Theta}}_1^{-1}\underline{\boldsymbol{\theta}}_0 + \tfrac{1}{2}\log\det(-2\underline{\boldsymbol{\Theta}}_1) - N\log\pi\right) \end{aligned} \tag{A.125}$$

with sufficient statistics $\underline{\boldsymbol{\phi}}_0 = \underline{\boldsymbol{z}}$ and $\underline{\boldsymbol{\Phi}}_1 = \underline{\boldsymbol{z}}\underline{\boldsymbol{z}}^{\mathsf{H}}$, and corresponding natural parameters $\underline{\boldsymbol{\theta}}_0 = \underline{\boldsymbol{\Sigma}}^{-1}\underline{\boldsymbol{\mu}}$ and $\underline{\boldsymbol{\Theta}}_1 = -\frac{1}{2}\underline{\boldsymbol{\Sigma}}^{-1}$. Note the similarity of (A.103) and (A.125), which is a consequence of the usage of augmented quantities.

The matrix derivatives (A.105) have straightforward extensions to the complex case

$$\frac{\partial \underline{\boldsymbol{a}}^{\mathsf{H}}\underline{\boldsymbol{Z}}^{-1}\underline{\boldsymbol{a}}}{\partial \underline{\boldsymbol{Z}}} = -\underline{\boldsymbol{Z}}^{-1}\underline{\boldsymbol{a}}\underline{\boldsymbol{a}}^{\mathsf{H}}\underline{\boldsymbol{Z}}^{-1} \quad \text{and} \quad \frac{\partial \log\det\underline{\boldsymbol{Z}}}{\partial \underline{\boldsymbol{Z}}} = \underline{\boldsymbol{Z}}^{-1} \tag{A.126}$$

for any $\underline{\boldsymbol{Z}} \in \mathbb{C}_*^{2N\times 2N}$ and $\underline{\boldsymbol{a}} \in \mathbb{C}_*^{2N}$. With the log-partition function

$$A(\underline{\boldsymbol{\theta}}_0, \underline{\boldsymbol{\Theta}}_1) = -\tfrac{1}{4}\underline{\boldsymbol{\theta}}_0^{\mathsf{H}}\underline{\boldsymbol{\Theta}}_1^{-1}\underline{\boldsymbol{\theta}}_0 - \tfrac{1}{2}\log\det(-2\underline{\boldsymbol{\Theta}}_1) + N\log\pi \tag{A.127}$$

we find the moments using (A.115) as

$$\mathbb{E}\left[\underline{\boldsymbol{z}}\right] = \frac{\partial A}{\partial \underline{\boldsymbol{\theta}}_0^{\mathsf{H}}} = -\tfrac{1}{2}\underline{\boldsymbol{\Theta}}_1^{-1}\underline{\boldsymbol{\theta}}_0 = \underline{\boldsymbol{\mu}} \tag{A.128}$$

$$\operatorname{cov}\left[\underline{\boldsymbol{z}}\right] = \frac{\partial^2 A}{\partial \underline{\boldsymbol{\theta}}_0\, \partial \underline{\boldsymbol{\theta}}_0^{\mathsf{H}}} = -\tfrac{1}{2}\underline{\boldsymbol{\Theta}}_1^{-1} = \underline{\boldsymbol{\Sigma}} \tag{A.129}$$

$$\mathbb{E}\left[\underline{\boldsymbol{z}}\underline{\boldsymbol{z}}^{\mathsf{H}}\right] = \frac{\partial A}{\partial \underline{\boldsymbol{\Theta}}_1^{\mathsf{H}}} = \tfrac{1}{4}\underline{\boldsymbol{\Theta}}_1^{-1}\underline{\boldsymbol{\theta}}_0\underline{\boldsymbol{\theta}}_0^{\mathsf{H}}\underline{\boldsymbol{\Theta}}_1^{-1} - \tfrac{1}{2}\underline{\boldsymbol{\Theta}}_1^{-1} = \underline{\boldsymbol{\mu}}\underline{\boldsymbol{\mu}}^{\mathsf{H}} + \underline{\boldsymbol{\Sigma}} \tag{A.130}$$

once more verifying the validity of the mapping from natural to mean parameters in the complex case. Note that in the last equation $\frac{\partial A}{\partial \underline{\boldsymbol{\Theta}}_1^{\mathsf{H}}} = \frac{\partial A}{\partial \underline{\boldsymbol{\Theta}}_1}$, because $\underline{\boldsymbol{\Theta}}_1$ is Hermitian and A is real-valued.

We denote the improper Gaussian distribution with mean $\underline{\boldsymbol{\mu}} = \begin{pmatrix}\boldsymbol{\mu}\\ \boldsymbol{\mu}^*\end{pmatrix}$ and covariance matrix $\underline{\boldsymbol{\Sigma}} = \begin{pmatrix}\boldsymbol{\Sigma} & \tilde{\boldsymbol{\Sigma}}\\ \tilde{\boldsymbol{\Sigma}}^* & \boldsymbol{\Sigma}^*\end{pmatrix}$ as $\mathcal{EN}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \tilde{\boldsymbol{\Sigma}})$ or $\mathcal{EN}(\underline{\boldsymbol{\mu}}, \underline{\boldsymbol{\Sigma}})$.

If $\tilde{\boldsymbol{\Sigma}} = \boldsymbol{0}$, the distribution is proper, and is denoted as $\mathcal{CN}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Appendix B

Approximate Statistical Inference

All detection and estimation tasks that a receiver needs to execute are instances of *statistical inference* problems, which thus play a central role in this thesis.

Historically, statistics has been split into *orthodox* and *Bayesian* schools. A detailed comparison of both approaches is out of scope of this work; the interested reader is referred to the very enlightening discussion in [62] instead. The bottom line, however, is that nowadays it seems generally recognized [33,77,82] that orthodox statistics suffers from some severe flaws, whereas Bayesian statistics offers a systematic methodology which gives meaningful results for all well-posed problems, including corner cases where the orthodox approach breaks down (examples of which can be found in [93]).

For this reason, this thesis is based on the Bayesian approach to statistical inference. In particular, we treat all unknown quantities as *random variables*, and express our imperfect knowledge about those variables in terms of probability distributions (which may decay to Dirac distributions). The related concept of *deterministic but unknown* parameters, which is ubiquitous in orthodox statistics, will not be used in this thesis.

This chapter is organized as follows. After a brief introduction in Section B.1, the topic of *structured probability distributions* is covered in Section B.2. This is a central concept, since exploiting the available structure is a key step towards the derivation of efficient inference algorithms. Section B.3 introduces the *divergence minimization* principle, a framework for the systematic derivation of a broad variety of inference algorithms. An iterative method for finding approximate solutions to the generic divergence minimization problem is presented in Section B.4. Finally, Section B.5 derives several state-of-the-art algorithms like *belief propagation* and *expectation maximization* as special cases of the divergence minimization framework.

The most influential paper for this chapter has been [94], which proposes the generic message passing scheme that will be discussed later on. More recently, similar work has been presented in [111], which approaches the problem of unifying belief propagation

and variational message passing from a different angle, but arrives at essentially the same result.

An exhaustive treatment of graphical models and inference algorithms can be found in [65], while [69] and [78] are well readable tutorial papers on factor graphs and belief propagation. A comparison of inference algorithms that work on graphical models is given in [40]. Further recommended books on statistical inference include [13] and [140].

B.1 Statistical Inference

Statistical inference is the process of reasoning about unknown quantities, based on the observation of other, stochastically related quantities. At the heart of statistical inference lies a joint probability distribution p which models the stochastic dependencies between all quantities that are of interest to the problem at hand. The task of finding a suitable model, which describes the underlying physical phenomena with sufficient accuracy while being simple enough to allow for mathematical tractability, is an interesting problem in its own and is outside the scope of this thesis. In the context of wireless communication, this problem mainly involves the modeling of the stochastic channel, including both the propagation of the radio waves and the behaviour of analog components in transmitter and receiver. In this thesis, we will only consider simple, widely accepted models such as Rayleigh fading and Wiener phase noise. In the following we assume the availability of a sufficiently accurate model, encoded in the distribution p.

When a suitable model has been identified, it can be used to answer several kinds of questions. For our purpose, the most important kind is what [65] calls a *marginal MAP query*, which finds the most probable value of a particular unobserved variable, conditioned on the values of the observed quantities. To formalize this task, we split the set of variables into $\boldsymbol{x}$ and $\boldsymbol{y}$: the observations are collected in the vector $\boldsymbol{y}$, whereas

$$\boldsymbol{x} = (x_0, \dots, x_{N-1})^{\mathsf{T}} \in \mathcal{X} \triangleq \mathcal{X}_0 \times \cdots \times \mathcal{X}_{N-1} \tag{B.1}$$

contains the unknown variables. The marginal MAP query for x_n is then defined as

$$\begin{aligned}
\hat{x}_n &\triangleq \operatorname*{arg\,max}_{x_n \in \mathcal{X}_n} p(x_n \mid \boldsymbol{y}) \\
&\overset{(a)}{=} \operatorname*{arg\,max}_{x_n \in \mathcal{X}_n} \int_{\mathcal{X}_{\setminus n}} p(\boldsymbol{x} \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{x}_{\setminus n} \\
&\overset{(b)}{=} \operatorname*{arg\,max}_{x_n \in \mathcal{X}_n} \int_{\mathcal{X}_{\setminus n}} \frac{p(\boldsymbol{x})\, p(\boldsymbol{y} \mid \boldsymbol{x})}{p(\boldsymbol{y})} \, \mathrm{d}\boldsymbol{x}_{\setminus n} \\
&\overset{(c)}{=} \operatorname*{arg\,max}_{x_n \in \mathcal{X}_n} \int_{\mathcal{X}_{\setminus n}} p(\boldsymbol{x})\, p(\boldsymbol{y} \mid \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}_{\setminus n}
\end{aligned} \tag{B.2}$$

where $\boldsymbol{x}_{\setminus n} \triangleq (x_0, \dots, x_{n-1}, x_{n+1}, \dots, x_{N-1})^{\mathsf{T}}$ contains all unknown variables except x_n.

In (B.2), (a) involves marginalization over $\boldsymbol{x}_{\setminus n}$, (b) is an application of Bayes' rule, and (c) holds because $p(\boldsymbol{y})$ is a positive constant with respect to $\boldsymbol{x}$, hence multiplication with $p(\boldsymbol{y})$ does not change the location of the maximum. We will often make use of the freedom to ignore constant factors, as it will lead to some considerable simplifications.[1]

[1] This is not to say that constant factors can *always* be neglected. In the problem of *model selection*, where we do not have one but several candidate models, encoded in the distributions p_i, the so-called *evidence* $p_i(\boldsymbol{y})$ quantifies how well the competing models can explain the observed data. It can be used either for deciding on the best model, or for averaging over all models. In this thesis, we will not be concerned with model selection.

In principle, (B.2) can be solved via repeated application of the two basic rules of probability theory: the marginalization or *sum rule*

$$p(y) = \int p(x, y)\,\mathrm{d}x \tag{B.3}$$

and the chain rule or *product rule*

$$p(x, y) = p(x)\,p(y \mid x). \tag{B.4}$$

In practice, however, exact solutions are only available for some very simple models. In more complex models, we are facing two kinds of obstacles:

- *Space complexity:* An accurate representation of probability densities can require an arbitrary amount of memory. But even for discrete distributions over a finite set $\mathcal{X}_n$, which could theoretically be stored as a vector of dimension $|\mathcal{X}_n| - 1$, the space complexity may become problematic, as $|\mathcal{X}_n|$ is often very large in practice.

- *Time complexity:* The integrals that arise in the context of continuous variables may not have closed solutions. And if $\mathcal{X}_n$ is discrete, such that $\int_{\mathcal{X}_n}(\cdot)\,\mathrm{d}x_n$ turns into a sum, the number $|\mathcal{X}_n|$ of terms may be so large that its evaluation is still practically impossible.

When faced with such complex models, we have no other choice than to introduce approximations. The available algorithms for approximate inference can be broadly split into two categories.

- *Parametric methods:* If the distribution that encodes our model is sufficiently "nice" in a certain sense (e. g., it is a member of a sufficiently structured exponential family; see Section A.2), chances are that we can solve (B.2) exactly. If it is not, we can still *approximate* it with an appropriate member of the exponential family, and then answer our MAP queries using the approximate distribution instead of the true one. If the approximation is good, we can expect to obtain results that are close to the exact answers. The problem is then to tweak the parameters of the approximate distribution such that it fits the true distribution as closely as possible. The integration problem (B.2) thus turns into an optimization problem.

- *Particle methods:* Besides the parametric approach, another option to approximate a complex distribution is with a set of values $x^{(k)}$, the so-called *particles*, which may have an associated *weight* $w^{(k)}$:

$$p(x_n) \approx \sum_{k=0}^{K-1} w^{(k)} \delta(x_n - x_n^{(k)}) \quad \text{with} \quad \sum_{k=0}^{K-1} w^{(k)} = 1. \tag{B.5}$$

In this case, the approximated density is represented by the pairs of particles and weights $(x_n^{(k)}, w^{(k)})_{k=0}^{K-1}$, and integrals over x_n decay to finite sums:

$$\int_{\mathcal{X}_n} p(x_n)\, f(x_n)\, \mathrm{d}x_n \approx \sum_{k=0}^{K-1} w^{(k)} f(x_n^{(k)}). \tag{B.6}$$

The problem is then to find a set of particles that represents the true distribution accurately. In practice, this is often done by drawing random samples from the target distribution (or an approximation thereof), giving rise to *stochastic algorithms*.

The remainder of this chapter gives a deeper introduction to the parametric paradigm, as it is the basis for the main part of this thesis. The dominance of parametric algorithms can be explained by the stringent complexity constraints of mobile receivers, which are due to the real-time requirements and the scarce supply of energy in battery-driven devices. Since particle methods make fewer assumptions about the target distribution, they usually require more computations to yield satisfactory results than parametric methods; a property which currently makes them less interesting for practical applications. However, presuming that Moore's law continues to hold for some time, it is well possible that this will change in the future. This is because parameteric methods can suffer from systematic errors, while well designed particle-based approaches do not. If we approximate a multimodal density with, say, a Gaussian, then the approximation is necessarily imperfect, no matter how much computation time we spend on optimizing the parameters of the Gaussian distribution. Particle methods, on the other hand, can approximate any distribution more closely when we increase the number of samples. In this case, the lack of assumptions turns into an advantage. In the limit of an infinite amount of available computation power, a well designed particle-based algorithm would therefore almost surely converge to the exact result, whereas parametric algorithms would run into a kind of error floor.

Finally, it should be emphasized that parametric and particle-based approaches are by no means mutually exclusive. As we discuss in the following section, most distributions of practical interest are structured in certain ways, such that inference problems naturally decompose into smaller, local problems, each of which can be solved with a different algorithm. Graphical models provide a global view on the problem at hand, and suggest a systematic way for breaking it down into local, interconnected subtasks. A discussion of how particle methods fit into this framework is given in [29].

In some sense, hard parameter estimation, which approximates the distribution of x_n by a Dirac distribution $\delta(x_n - \hat{x}_n)$, can be regarded as a degenerate particle method with $K = 1$. However, the particle $\hat{x}_n$ is usually not found by stochastic algorithms, but by maximizing (an approximation of) the posterior distribution of x_n. We will return to this aspect in Section B.4.3.

B.2 Structured Distributions and Graphical Models

In order to solve (B.2) as efficiently as possible (be it exactly or approximately), it is essential to exploit the structure in the form of conditional independences that the distribution $p(\boldsymbol{x} \mid \boldsymbol{y})$ provides. In this section, we first have a closer look on how conditional independence can make marginalization tractable. We then introduce the concept of probabilistic graphical models and discuss some simple, but important examples of particularly strongly structured distributions.

For the following development, it is immaterial that $p(\boldsymbol{x} \mid \boldsymbol{y})$ is a posterior distribution. In order to simplify the notation, we thus drop the explicit dependence on $\boldsymbol{y}$ (which is just a fixed parameter at runtime) and denote the distribution simply as $p(\boldsymbol{x})$. We also assume that all random variables x_n are discrete, and that their ranges $|\mathcal{X}_n|$ are small enough such that summations over $\mathcal{X}_n$ are practically feasible.

B.2.1 Conditional Independence

By repeated application of the chain rule (B.4), it is easy to show that any distribution over N variables can be factorized as

$$p(x_0, \ldots, x_{N-1}) = \prod_{n=0}^{N-1} p(x_n \mid x_0, \ldots, x_{n-1}) \tag{B.7}$$

where the first factor should be read as the marginal distribution $p(x_0)$. Of course, the order in (B.7) does not matter, so $p(x_0, \ldots, x_{N-1})$ can be expanded in $N!$ different ways.

Fortunately, most distributions that arise in practice are simpler than (B.7), in the sense that some of the factors $p(x_n \mid x_0, \ldots, x_{n-1})$ only depend on a subset of $\{x_i \mid i < n\}$. The deeper reason is that typically, only few variables interact directly with each other in the underlying physical process; most interactions are indirect. As a simple example, consider an uncoded transmission (i. e., of mutually independent data symbols) over a discrete-time channel described by $y_n = h_n x_n + w_n$. By this equation, the four variables y_n, h_n, x_n and w_n interact directly. Under the standard assumption of white noise $\boldsymbol{w}$ and correlated fading $\boldsymbol{h}$, the observation y_n also contains information about the data symbols $x_m, m \neq n$, but only *indirectly* through the fading process $\boldsymbol{h}$. A factorization along the lines of (B.7), but neglecting as many unnecessary conditions as possible, is $p(\boldsymbol{x}, \boldsymbol{h}, \boldsymbol{y}) = p(\boldsymbol{h}) \prod_{n=0}^{N-1} p(y_n \mid x_n, h_n)\, p(x_n)$. Thus, while y_{n+1} contains information about x_n, it does not contribute *further* information assuming that we already know h_n. This gives rise to the notion of conditional independence, formally defined as follows.

Let $\mathcal{N} \triangleq \{0, \ldots, N-1\}$. In the following, we write $\boldsymbol{x}_{\mathcal{I}} \triangleq (x_i \mid i \in \mathcal{I})$, where $\mathcal{I} \subseteq \mathcal{N}$ is an index set, to denote a subvector of $\boldsymbol{x}$. Let $\mathcal{A}, \mathcal{B}, \mathcal{C} \subset \mathcal{N}$ be three mutually disjoint index sets. Then, $\boldsymbol{x}_{\mathcal{A}}$ and $\boldsymbol{x}_{\mathcal{B}}$ are *conditionally independent* given $\boldsymbol{x}_{\mathcal{C}}$, written $(\boldsymbol{x}_{\mathcal{A}} \perp \boldsymbol{x}_{\mathcal{B}} \mid \boldsymbol{x}_{\mathcal{C}})$, if and only if

$$p(\boldsymbol{x}_{\mathcal{A}}, \boldsymbol{x}_{\mathcal{B}} \mid \boldsymbol{x}_{\mathcal{C}}) = p(\boldsymbol{x}_{\mathcal{A}} \mid \boldsymbol{x}_{\mathcal{C}})\, p(\boldsymbol{x}_{\mathcal{B}} \mid \boldsymbol{x}_{\mathcal{C}}) \tag{B.8}$$

or equivalently

$$p(\boldsymbol{x}_\mathcal{A} \,|\, \boldsymbol{x}_\mathcal{B}, \boldsymbol{x}_\mathcal{C}) = p(\boldsymbol{x}_\mathcal{A} \,|\, \boldsymbol{x}_\mathcal{C}). \tag{B.9}$$

In the special case $\mathcal{C} = \emptyset$, meaning that $\boldsymbol{x}_\mathcal{C}$ vanishes from above equations, $\boldsymbol{x}_\mathcal{A}$ and $\boldsymbol{x}_\mathcal{B}$ are *marginally independent*, written $(\boldsymbol{x}_\mathcal{A} \perp \boldsymbol{x}_\mathcal{B})$.

Fully Factorized Distributions Exploiting conditional independence is crucial for the derivation of efficient inference algorithms, because it allows us to discard irrelevant information. In the simplest imaginable case, the distribution decomposes as

$$p(x_0, \ldots, x_{N-1}) = \frac{1}{Z} \prod_{n=0}^{N-1} f_n(x_n) \tag{B.10}$$

where the $f_n : \mathcal{X}_n \to [0, \infty)$ are non-negative functions, and Z is a normalization constant. In order to obtain the marginal distribution $p(x_n)$, we simply take $f_n(x_n)$ and normalize it to a valid distribution; the factorization allows us to ignore all other $x_m, m \neq n$. A distribution which can be decomposed as (B.10) is called *fully factorized*. Despite their simplicity, fully factorized distributions emerge as a key ingredient in the development of approximate inference algorithms.

Markov Chains A somewhat more complex example is a *Markov chain*, where the only direct interactions are between adjacent variables:

$$p(x_0, \ldots, x_{N-1}) = \frac{1}{Z} \prod_{n=1}^{N-1} f_n(x_{n-1}, x_n) \tag{B.11}$$

which implies the conditional independences $(x_0, \ldots, x_{n-1} \perp x_{n+1}, \ldots, x_{N-1} \,|\, x_n)$. Thus, when reasoning about, say, x_0, we do not have to sum over all possible assignments to $(x_1, \ldots, x_{N-1})$ because

$$\begin{aligned} p(x_0) &= \frac{1}{Z} \sum_{x_1} \cdots \sum_{x_{N-1}} \prod_{n=1}^{N-1} f_n(x_{n-1}, x_n) \\ &= \frac{1}{Z} \sum_{x_1} f_1(x_0, x_1) \sum_{x_2} \cdots \sum_{x_{N-1}} \prod_{n=2}^{N-1} f_n(x_{n-1}, x_n) \\ &= \frac{1}{Z} \sum_{x_1} f_1(x_0, x_1)\, \psi(x_1). \end{aligned} \tag{B.12}$$

Since x_0 depends on the other variables only through x_1, it is possible to merge all information that the other variables contain about x_0 into a unary function $\psi(x_1)$.

(a) Fully factorized distribution **(b)** Markov chain

Figure B.1: Simple Markov networks with $N = 4$ variables

B.2.2 Graphical Models and Message Passing

In general, it might not be obvious how to extract conditional independences from a given factorization of the joint distribution p. Graphical models provide a visualization of p, and are therefore valuable tools for the identification of its structure. As we will see later, they are also useful as a basis for the systematic derivation of a broad variety of inference algorithms.

Markov Networks

Assume that we have some generic factorization

$$p(\boldsymbol{x}) = \frac{1}{Z} \prod_{k=0}^{K-1} f_k(\boldsymbol{x}_{\mathcal{N}_k}) \tag{B.13}$$

where Z is a normalization constant. For all k, f_k is a non-negative function and $\mathcal{N}_k \subseteq \{0, \dots, N-1\}$, such that the argument list $\boldsymbol{x}_{\mathcal{N}_k}$ of f_k is a subvector of $\boldsymbol{x}$. The functions f_k could be, but don't have to be, conditional distributions as in the decomposition (B.7).

The *Markov network* associated with the factorization (B.13) is a graph with N vertices, one for each variable x_0 through x_{N-1}.[2] It contains an edge between the vertices x_m and x_n if and only if $\exists k : \{m, n\} \subseteq \mathcal{N}_k$ (i. e., at least one of the functions f_k depends on both x_m and x_n). A *path* is a sequence $(x_{\pi(0)}, x_{\pi(1)}, \dots, x_{\pi(M)})$ of vertices, such that for all $m \in \{0, \dots, M-1\}$ there exists an edge between $x_{\pi(m)}$ and $x_{\pi(m+1)}$. Two vertices x_m and x_n are *connected* if a path exists between them, otherwise they are *disconnected*. They are *adjacent* if they are connected via a single edge. Adjacent vertices are called *neighbors*.

It can be shown [65, Section 4.3] that $(\boldsymbol{x}_\mathcal{A} \perp \boldsymbol{x}_\mathcal{B} \mid \boldsymbol{x}_\mathcal{C})$ if all paths from any $x_a, a \in \mathcal{A}$ to any $x_b, b \in \mathcal{B}$ contain an $x_c, c \in \mathcal{C}$. In other words, if all $\boldsymbol{x}_\mathcal{C}$ and their adjacent edges were removed from the graph, the sets $\boldsymbol{x}_\mathcal{A}$ and $\boldsymbol{x}_\mathcal{B}$ would become disconnected. Intuitively, information about the variables flows only along the edges of the Markov network. If a variable (say, y) is observed, the information flow gets blocked because the subnetwork "behind" y could only contribute information to the variables "in front of" y through y itself, which however is perfectly known already.

Figure B.1 shows Markov networks for the two simple examples from the previous section. The fully factorized distribution (B.10) does not contain any factor which depends on two or more variables. Thus, its Markov network is the fully disconnected

[2] With slight abuse of notation, we will not distinguish between the vertices and their corresponding variables.

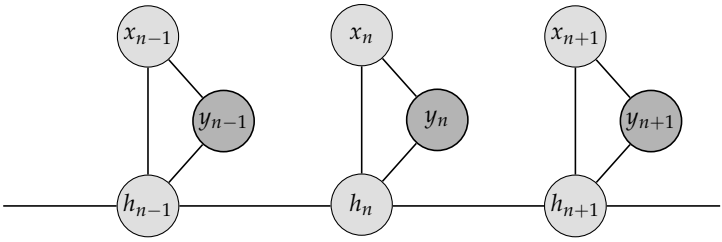


Figure B.2: Section of the Markov network for the factorization (B.14)

graph as shown in Figure B.1a. The fact that all variables in (B.10) are statistically independent is reflected by the lack of edges in the network, which means that it is impossible for any variable to provide information about any other.

The Markov chain (B.11) contains only pairwise factors with variables (x_{n-1}, x_n), so each factor contributes one edge to the Markov network shown in Figure B.1b. Obviously, the graph is connected, so all variables are marginally dependent. Observing one variable, say x_n, splits the chain into two conditionally independent subgraphs $(x_0, \ldots, x_{n-1})$ and $(x_{n+1}, \ldots, x_{N-1})$.

In order to demonstrate how graphical models can help us to identify conditional independences, consider again the example of an uncoded transmission over a discrete-time channel as discussed at the beginning of Section B.2.1. For simplicity, assume furthermore that the fading process $\boldsymbol{h}$ can be modeled as a Markov chain. A natural factorization of this model is

$$p(\boldsymbol{x}, \boldsymbol{h}, \boldsymbol{y}) = \prod_{n=0}^{N-1} p(y_n \,|\, x_n, h_n)\, p(x_n)\, p(h_n \,|\, h_{n-1}) \tag{B.14}$$

with $p(h_0 \,|\, h_{-1}) = p(h_0)$. A section of the corresponding Markov network is shown in Figure B.2, where observed variables are indicated by a dark shade. From the graphical model it is obvious that any path from $y_m, m \neq n$ to x_n necessarily contains h_n, from which we conclude that the only information that the observations $y_m, m \neq n$ can provide about the data symbol x_n is via the state of the fading channel. If $\boldsymbol{h}$ were known, the observations $y_m, m \neq n$ would not contain any further information, so the demapping of x_n could be based on observation y_n alone. This leads naturally to a decomposition of the receiver into a channel estimator and N independent demappers. Even in this simple example, extracting this structure directly from the factorization (B.14) would arguably be more cumbersome.

Finally, note that a factorization of a given distribution is not unique, so the Markov network is not unique, either. A complete graph (which contains edges between all pairs of vertices) is trivially compatible with any distribution, but it does not reveal any

structure. It is therefore worthwile to look for a *minimal* network which contains as few edges as possible. The interesting information that graphical models convey is not contained in the edges that are *present* in the graph, but in the edges that are *missing*.

Message Passing

Let us now turn to the problem of marginalizing structured distributions. Consider again the Markov chain with $N = 4$ variables as shown in Figure B.1b, and assume we are interested in finding $p(x_0)$, the marginal of the first variable. Inserting (B.11) into the definition of the marginal yields

$$p(x_0) = \frac{1}{Z} \sum_{x_1} \sum_{x_2} \sum_{x_3} f_1(x_0, x_1)\, f_2(x_1, x_2)\, f_3(x_2, x_3). \tag{B.15}$$

Exploiting the distributive law, we shift the sums as far to the right as possible and assign descriptive names to the intermediate results:

$$\begin{aligned}
p(x_0) &= \frac{1}{Z} \sum_{x_1} f_1(x_0, x_1) \sum_{x_2} f_2(x_1, x_2) \underbrace{\sum_{x_3} f_3(x_2, x_3)}_{\triangleq m_{3\to 2}(x_2)} \\
&= \frac{1}{Z} \sum_{x_1} f_1(x_0, x_1) \underbrace{\sum_{x_2} f_2(x_1, x_2)\, m_{3\to 2}(x_2)}_{\triangleq m_{2\to 1}(x_1)} \\
&= \frac{1}{Z} \underbrace{\sum_{x_1} f_1(x_0, x_1)\, m_{2\to 1}(x_1)}_{\triangleq m_{1\to 0}(x_0)} \\
&= \frac{1}{Z} m_{1\to 0}(x_0).
\end{aligned} \tag{B.16}$$

The intermediate functions $m_{n+1\to n}(x_n)$ can be computed recursively, and each computation only involves the summation over a single variable.

Now assume we want to find the last marginal $p(x_3)$. By symmetry, we can proceed analogously:

$$\begin{aligned}
p(x_3) &= \frac{1}{Z} \sum_{x_2} f_3(x_2, x_3) \sum_{x_1} f_2(x_1, x_2) \underbrace{\sum_{x_0} f_1(x_0, x_1)}_{\triangleq m_{0\to 1}(x_1)} \\
&= \frac{1}{Z} \sum_{x_2} f_3(x_2, x_3) \underbrace{\sum_{x_1} f_2(x_1, x_2)\, m_{0\to 1}(x_1)}_{\triangleq m_{1\to 2}(x_2)}
\end{aligned}$$

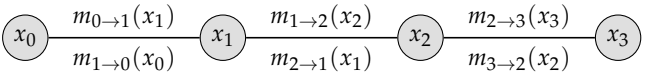


Figure B.3: Forward and backward messages passed along a Markov chain

$$= \frac{1}{Z} \underbrace{\sum_{x_2} f_3(x_2, x_3)\, m_{1\to2}(x_2)}_{\triangleq m_{2\to3}(x_3)}$$

$$= \frac{1}{Z} m_{2\to3}(x_3). \tag{B.17}$$

What about the interior variables, say x_1? With the same approach, we find

$$p(x_1) = \frac{1}{Z} \underbrace{\sum_{x_0} f_1(x_0, x_1)}_{m_{0\to1}(x_1)} \underbrace{\sum_{x_2} f_2(x_1, x_2) \sum_{x_3} f_3(x_2, x_3)}_{m_{2\to1}(x_1)} \tag{B.18}$$

where we have reused intermediate results from (B.16) and (B.17). Thus, once we have computed the intermediate functions in (B.16) and (B.17), we can find *all* marginals as

$$p(x_n) = \frac{1}{Z} m_{n-1\to n}(x_n)\, m_{n+1\to n}(x_n). \tag{B.19}$$

In order to make this equation well defined for all $n \in \{0, N-1\}$, it makes sense to define the constant functions $m_{-1\to0}(x_0) \equiv 1 \equiv m_{N\to N-1}(x_{N-1})$.

A standard interpretation of this algorithm is *message passing*: the functions m are referred to as *messages* which are exchanged between adjacent variables. Figure B.3 shows the Markov chain, where the edges are labeled with the messages that are passed along them. The messages above the edges are called *forward messages*, and their computation (B.17) is called the *forward recursion*. Similarly, the messages below the edges are the *backward messages*, computed by the *backward recursion* (B.16).

The method of finding the marginals of a Markov chain that we just derived is a special case of a more general algorithm called *belief propagation*. It works over any *singly-connected* (i. e., *tree-structured*) belief network, and returns the exact marginals after a finite number of operations. Starting at the leaves (nodes with only one edge), it recursively computes $2E$ messages, where E is the number of edges in the network. Then, the marginals are obtained as

$$p(x_n) = \frac{1}{Z} \prod_{n'} m_{n'\to n}(x_n) \tag{B.20}$$

where the product is over all neighbors of x_n.

Unfortunately, most probability distributions of practical interest do not have simple[3] singly-connected networks. In order to perform statistical inference in such distributions, we do not have much choice but to introduce approximations. A common heuristic is based on the fact that belief propagation only performs *local* computations which can be implemented regardless of whether the *global* graph structure is tree-like or not. By ignoring the loops and running belief propagation nonetheless, we obtain an iterative algorithm that is commonly called *loopy belief propagation*. Empirical evidence shows that if the messages converge, the approximated marginals that are obtained as in (B.20) are sufficiently accurate for many practical problems. Arguably the most famous examples in the field of digital communication are the highly successful turbo and LDPC decoders, which are known to be instances of loopy belief propagation [86].

As long as the variables are discrete and have a sufficiently small range (like the binary variables in decoding algorithms), there is not much left to say. However, loops are not the only reason why we might need to introduce approximations. Beyond any structural aspects, the factors f_k in (B.13) themselves may be too complex for an exact implementation of belief propagation. This happens frequently if the variables are continuous (the all-Gaussian case being an important exception), but also if the variables are discrete with an exponential number of states.

In the remainder of this chapter, we will discuss a systematic approach to the derivation of approximate inference algorithms which plays a central role in this thesis. The *divergence minimization* approach contains loopy belief propagation as a special case, but also encompasses other techniques which are better suited for continuous variables.

Factor Graphs

While Markov networks are a useful tool for the identification of conditional independences, it turns out that *factor graphs* are more suitable for the purpose of deriving inference algorithms. Like Markov networks, they represent the factorization (B.13) of a distribution

$$p(\boldsymbol{x}) = \frac{1}{Z} \prod_{k=0}^{K-1} f_k(\boldsymbol{x}_{\mathcal{N}_k}) \tag{B.21}$$

but in contrast to Markov networks, factor graphs make the factorization *explicit*. In addition to the N vertices that represent the variables x_n, the factor graph for (B.21) contains K vertices representing the functions f_k. The two types of vertices are commonly called *variable nodes* and *factor nodes*, respectively. An edge exists between the variable node x_n and the factor node f_k if and only if $n \in \mathcal{N}_k$ (i. e., the function f_k depends on x_n).

[3] There exist graph transformations which can be used to remove cycles, for example by merging variables. However, the range of the variables increases exponentially (e. g., if M binary variables are merged, the new variable has 2^M states), so the practical applicability of this technique is rather limited. As an extreme example, it is always possible to merge *all* variables into the vector $\boldsymbol{x}$. The resulting Markov network contains just a single node, so it is clearly singly-connected, but it is also useless because it defeats the purpose of identifying and exploiting structure.

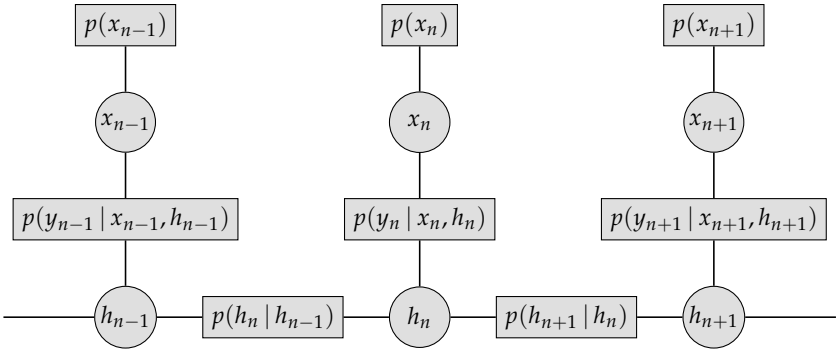


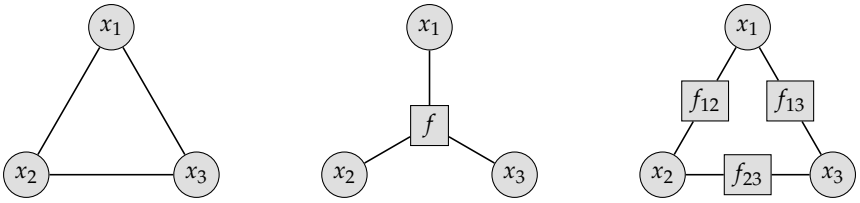
Figure B.4: Section of the factor graph for the factorization (B.14)

Since all edges in a factor graph connect a variable with a factor node, factor graphs are *bipartite*. The set $\mathcal{N}_k$ consists of the indices of all variables that f_k depends on. Similarly, the set $\mathcal{K}_n = \{k \mid n \in \mathcal{N}_k\}$ contains the indices of all functions that depend on x_n. In the following, we adhere to the convention of drawing variable nodes as circles and factor nodes as rectangles.

As a simple example, Figure B.4 shows a section of the factor graph for the discrete-time transmission as described by (B.14). The corresponding Markov network has been given on page 233. Note that only the unobserved variables are explicitly represented by nodes in the factor graph. The observations y_n are considered as fixed parameters of the functions $f(x_n, h_n) = p(y_n \mid x_n, h_n)$, which at runtime only depend on x_n and h_n.

One advantage of factor graphs over Markov network is their higher expressive power, as demonstrated by the standard example in Figure B.5. The network consists of three variables which are all directly connected, i. e., there are no conditional independences. Nonetheless, the distribution might admit a factorization into pairwise functions as $p(x_1, x_2, x_3) = \frac{1}{Z} f_{12}(x_1, x_2)\, f_{13}(x_1, x_3)\, f_{23}(x_2, x_3)$. The Markov network is not able to distinguish such a factorization from the trivial $p(x_1, x_2, x_3) = \frac{1}{Z} f(x_1, x_2, x_3)$. In contrast, the factor graph expresses the factorization explicitly and is therefore able to distinguish between the two cases.

The bipartite factor graphs discussed so far are also known as *Tanner graphs* [127]. Besides them, there exists another flavor of factor graphs in the literature, called *normal graphs* [42] or *Forney-style factor graphs* (FFGs) [78]. The difference is that FFGs *only* contain factor nodes. The variables are not represented by nodes, but are associated with *edges* instead. Since every edge is connected to exactly two nodes, it might seem that FFGs only exist for factorizations where every variable occurs in exactly two factors, but with a simple trick, FFGs can be drawn for any factorization. To handle variables which occur just once, FFGs may also contain *half edges* which are only connected to a



(a) Complete Markov network over three variables

(b) Factor graph for the trivial factorization $p \propto f$

(c) Factor graph for the more structured factorization $p \propto f_{12} f_{13} f_{23}$

Figure B.5: Comparison of the complete Markov network over three variables with two compatible factor graphs, expressing different amounts of structure

single node. To handle variables which occur $d \geq 3$ times, one introduces $d - 1$ auxiliary variables and an equality-constraint function, which forces all d copies of the variable to have the same value.

As an example, consider the factorization

$$f(x, y, z)\, g(y, z)\, h(z). \tag{B.22}$$

Figure B.6a shows the classical Tanner graph of (B.22). The Forney-style graph, shown in Figure B.6b, is obtained from the Tanner graph by the following steps: (a) Remove all variable nodes of degree 1, and convert their adjacent edges into half edges. (b) Remove all variable nodes of degree 2, and merge their two adjacent edges. (c) For every variable node of degree $d > 2$, replace the variable node by an equality constraint node, and introduce $d - 1$ copies of the original variable. Due to these auxiliary variables, the FFG describes strictly speaking a different factorization, in our example

$$f(x, y, z)\, g(y, z')\, h(z'')\, \boxed{=}(z, z', z'') \tag{B.23}$$

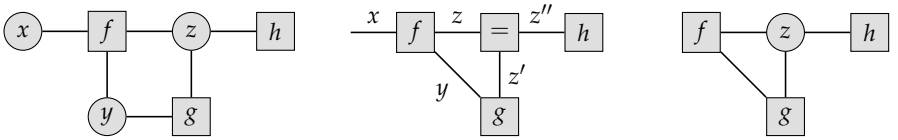
with the equality constraint function

$$\boxed{=}(z, z', z'') \triangleq \delta(z - z')\, \delta(z - z''). \tag{B.24}$$

Both factorizations are equivalent, however, in the sense that integrating over the auxiliary variables reproduces the original factorization. In particular, this implies that both factorizations have the same marginals.

It is obvious that the conversion from a Tanner graph to an FFG described above also works in the other direction. Thus, the two styles are isomorphic; both have exactly the same expressive power. It is then mainly a matter of taste which style one uses.

For the derivation of the generic divergence minimization approach that follows below, I consider it helpful to distinguish clearly between factor-to-variable and variable-to-factor messages. Therefore, Tanner graphs will be used in the remainder of this



(a) Classical bipartite factor graph

(b) Forney-style factor graph

(c) Hybrid factor graph used in this thesis

Figure B.6: Comparison of different factor graph styles

appendix. The main text, however, mostly uses a kind of hybid style as shown in Figure B.6c, which in my opinion combines the advantages of both approaches: Variables with degree ≥ 3 will be drawn as standard variable nodes, so we don't need to introduce any auxiliary variables and equality constraints. Variable nodes with degree 2, however, will not be drawn, just as in FFGs. The rationale is that degree-2 variable nodes are quite uninteresting from a computational perspective: all they do is pass the messages through unaltered. In the example from Figure B.6a, we would have $m_{f\to y}(y) = m_{y\to g}(y)$ and $m_{g\to y}(y) = m_{y\to f}(y)$. It thus leads to simpler notation, both mathematically and graphically, to omit the degree-2 variable nodes, and to denote the messages through them simply as $m_{f\to g}(y)$ and $m_{g\to f}(y)$, in this example. Finally, variable nodes of degree 1 will also be omitted, along with their adjacent edges. The reason is that variables of degree 1 will always be *observed*, and thus fixed at runtime. A standard example is the above mentioned $p(y_n \mid x_n, h_n)$ which describes the relation between channel input and output. The observation y_n does not appear in any other factor, so it is of degree 1, but since it is known, it makes sense to think of it as a fixed parameter and consider $p(y_n \mid x_n, h_n)$ as a function of just x_n and h_n.

It goes without saying that this hybrid style can trivially be converted to either Tanner or Forney-style graphs. If you prefer to work with Tanner graphs, just add degree-2 variable nodes where necessary. And if you prefer FFGs, just convert every variable node into an equality constraint function for that particular variable.

B.3 The Divergence Minimization Approach

While the problem of marginalizing an arbitrary distribution is generally intractable, we have seen in the previous section that there are certain classes of structured distributions which do allow for exact marginalization. We have identified two conditions that such simple distributions should fulfill:

- the distribution should have a factorization whose graphical representation is singly-connected, and
- whose factors are sufficiently simple functions. If all arguments are discrete, this basically means that the ranges of the variables are small enough such that a full enumeration of all states is possible. In the continuous case, the functions should have a simple parametric representation, the most important example being Gaussian functions.

Unfortunately, many distributions that arise in practical problems do not fulfill these conditions, which motivates our interest in *approximate* inference techniques.

Divergence minimization (DM) is a generic approach to the systematic derivation of approximate inference algorithms. The central idea is to approximate the true[4] distribution with a simpler one, which is subsequently used to answer the probability queries of interest. If the auxiliary distribution closely resembles the true distribution, it can be expected that those answers are good approximations of the exact results.

In the remainder of this section, we formalize this idea and show how the original marginalization problem can be transformed into an optimization problem. Then, in the next section, we will develop a generic method for solving this optimization problem approximately via fixed-point iterations. As we will see, state-of-the-art techniques like belief propagation and expectation maximization can be expressed as special cases of divergence minimization.

B.3.1 Statistical Inference via Optimization

We start by introducing some notation. As above, the vector $\boldsymbol{x} = (x_0, \ldots, x_{N-1})^\mathsf{T}$ contains all *unobserved* (also called *hidden* or *latent*) variables. The notation should be understood in a somewhat generalized sense, because the components $x_n \in \mathcal{X}_n$ are not necessarily real or complex scalars, but could be compound objects like vectors or matrices themselves. From the perspective of the DM framework, however, the x_n are atomic. In the language of graphical models, each x_n corresponds to one variable node.

[4] We use the terms *true distribution* or *target distribution* to refer to the distribution which encodes our model assumptions. The word "true" should not be understood as a statement about the accuracy of this model. As discussed at the beginning of this chapter, the problem of finding an appropriate model is outside the scope of this thesis.

The observations are collected in the vector $\boldsymbol{y}$, which stays fixed during the execution of the inference algorithm. The target distribution that we want to approximate is the posterior $p(\boldsymbol{x} \mid \boldsymbol{y}) \propto p(\boldsymbol{x}, \boldsymbol{y})$. As discussed above, the normalization constant $p(\boldsymbol{y})$ can usually be neglected.

In order to distinguish cleanly between the target distribution and its approximation, the former is always denoted as p, while the auxiliary distribution will be called q. Note that q is an approximation to the *posterior* distribution, but for simplicity, its dependence on $\boldsymbol{y}$ is not made explicit. Our task is thus to find a q such that

$$q(\boldsymbol{x}) \approx p(\boldsymbol{x} \mid \boldsymbol{y}). \tag{B.25}$$

The first design decision that we need to make is to determine a set $\mathcal{Q}$ of "simple" distributions, from which the approximation q is picked. This choice entails a tradeoff between accuracy and complexity: the larger[5] we choose $\mathcal{Q}$, the higher the probability that it contains an element q which approximates p well, but the problem of *finding* a good $q \in \mathcal{Q}$ typically becomes more complex.

In order to formalize this problem, we need a divergence measure $D(p \,\|\, q)$ which quantifies the dissimilarity of p and q. It should be a non-negative function, with $D(p \,\|\, q) = 0$ if and only if $p = q$. Again, the choice of D contains an accuracy-complexity tradeoff.

When we have selected a set $\mathcal{Q}$ and a measure D, we can express the problem of finding a $q \approx p$ formally as

$$q = \operatorname*{arg\,min}_{\tilde{q} \in \mathcal{Q}} D(p \,\|\, \tilde{q}). \tag{B.26}$$

Once the optimum q is found, approximations to the marginals of interest are computed as

$$q_n(x_n) = \int q(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}_{\setminus n} \tag{B.27}$$

where $\boldsymbol{x}_{\setminus n} = (x_0, \ldots, x_{n-1}, x_{n+1}, \ldots, x_{N-1})^{\mathsf{T}}$, and $\int (\cdot) \, \mathrm{d}\boldsymbol{x}_{\setminus n}$ denotes marginalization over all components of $\boldsymbol{x}$ except of x_n. Since $\mathcal{Q}$ only contains sufficiently simple distributions, (B.27) is exactly solvable by construction and requires no further discussion. It then remains to find a solution of (B.26).

As we will examine in more detail below, the elements of $\mathcal{Q}$ are typically members of an exponential family (see Section A.2) and are therefore indexed by a parameter vector $\boldsymbol{\theta} \in \Omega$, where Ω is the natural parameter space of the exponential family. Then, the optimization problem (B.26) can equivalently be written as

$$\boldsymbol{\theta} = \operatorname*{arg\,min}_{\tilde{\boldsymbol{\theta}} \in \Omega} D(p \,\|\, q_{\tilde{\boldsymbol{\theta}}}). \tag{B.28}$$

[5] "Larger" should be understood as "being a superset of," rather than "having a larger cardinality," because even the simplest $\mathcal{Q}$ used in practice consist of uncountably many elements.

In Section A.2 we have shown that the natural parameter space of an exponential family is always a convex set. If the divergence measure is furthermore chosen such that $D(p \,\|\, q_{\boldsymbol{\theta}})$ is convex in $\boldsymbol{\theta}$, (B.28) is a convex optimization problem [15]. Now, the value of the preceding derivation becomes clear. Recall that our starting point has been the marginalization problem

$$p(x_n \,|\, \boldsymbol{y}) = \int p(\boldsymbol{x} \,|\, \boldsymbol{y}) \,\mathrm{d}\boldsymbol{x}_{\setminus n} \tag{B.29}$$

which, by assumption, involves intractable sums or integrals. From the formulation (B.29) it is not apparent how approximations could be introduced in a systematic manner. By transforming the integration into an optimization problem, we gain access to the powerful tools of convex optimization theory.

So far, we have not yet discussed how $\mathcal{Q}$ and D should be chosen. A natural question to ask is whether there exist choices which lead to exact marginals, $q_n(x_n) = p(x_n \,|\, \boldsymbol{y})$. As we will see in the following, this is indeed the case. Note that this result does not imply that we are able to solve the integration problem (B.29) exactly after all, because we will generally not be able to find the exact solution of the corresponding optimization problem (B.26). However, it still serves as an interesting starting point for the further development.

B.3.2 Projections of Probability Distributions

Arguably the most important divergence measure is the Kullback-Leibler divergence

$$D_{\mathrm{KL}}(p \,\|\, q) \triangleq \int p(x) \,\log \frac{p(x)}{q(x)} \,\mathrm{d}x \tag{B.30}$$

which is well defined as long as $p(x) > 0 \implies q(x) > 0$. It is a well-known property that $D_{\mathrm{KL}}(p \,\|\, q) \geq 0$ with equality if and only if $p = q$. Also, by substituting a generic exponential family $q(x) = \exp\left(\boldsymbol{\theta}^{\mathsf{T}} \boldsymbol{\phi}(x) - A(\boldsymbol{\theta})\right)$ into (B.30) and using the result that A is always a convex function, it can easily be shown that $D_{\mathrm{KL}}(p \,\|\, q_{\boldsymbol{\theta}})$ is convex in $\boldsymbol{\theta}$.

An operation that will be useful for the following derivation is the *projection* of a probability distribution p into a set $\mathcal{Q}$ of distributions, formally defined as

$$\begin{aligned} \mathrm{proj}_{\mathcal{Q}}\,[p] &\triangleq \operatorname*{arg\,min}_{q \in \mathcal{Q}} D_{\mathrm{KL}}(p \,\|\, q) \\ &= \operatorname*{arg\,max}_{q \in \mathcal{Q}} \int p(x) \,\log q(x) \,\mathrm{d}x. \end{aligned} \tag{B.31}$$

Figure B.7 illustrates the projection of p into $\mathcal{Q}$. Distributions are represented as points in a Euclidean space, and the set $\mathcal{Q}$ is visualized by the gray surface. Every $q \in \mathcal{Q}$ has a certain non-negative divergence from p, represented in the figure by the Euclidean distance. Then, $q^* = \mathrm{proj}_{\mathcal{Q}}\,[p]$ is the element of $\mathcal{Q}$ which is closest to p.

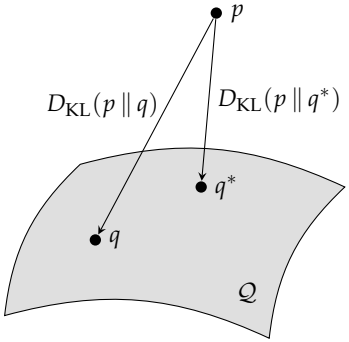


Figure B.7: Visualization of $q^* = \mathrm{proj}_{\mathcal{Q}}[p]$, the projection of p into the set $\mathcal{Q}$

Note that the Kullback-Leibler divergence is not symmetric, so a different projection could be defined as $\arg\min_{q\in\mathcal{Q}} D_{\mathrm{KL}}(q \,\|\, p)$. In [65], this latter projection is called the *information projection*, while (B.31) is called the *moment projection*. While we will indeed discuss minimization problems of the type $\arg\min_{q\in\mathcal{Q}} D_{\mathrm{KL}}(q \,\|\, p)$ below, we will reserve the term *projection*, and in particular the proj-operator, for moment projection.

Two special kinds of projection are of particular importance for us.

Projections into the Set of Fully Factorized Distributions

Consider a random vector $\boldsymbol{x} = (x_0, \ldots, x_{N-1})^{\mathsf{T}}$ where, as discussed at the beginning of Section B.3.1, the x_n are not necessarily scalars, but constitute an arbitrary partition of all random variables of interest. The set of fully factorized distributions (with respect to the given partition) is defined as

$$\mathcal{F} \triangleq \left\{ q \,\middle|\, q(\boldsymbol{x}) = \prod_{n=0}^{N-1} q_n(x_n) \right\} \tag{B.32}$$

where no constraints are imposed on the marginals q_n, beyond the necessary $q_n(x_n) \geq 0$ and $\int q_n(x_n)\,\mathrm{d}x_n = 1$.

The projection of an arbitrary distribution p into $\mathcal{F}$ can be computed analytically:

$$\begin{aligned} q &= \arg\max_{\tilde{q}\in\mathcal{F}} \int p(\boldsymbol{x})\, \log \tilde{q}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\ &\overset{(a)}{=} \arg\max_{\tilde{q}_0,\ldots,\tilde{q}_{N-1}} \int p(\boldsymbol{x}) \sum_{n=0}^{N-1} \log \tilde{q}_n(x_n)\,\mathrm{d}\boldsymbol{x} \\ &\overset{(b)}{=} \arg\max_{\tilde{q}_0,\ldots,\tilde{q}_{N-1}} \sum_{n=0}^{N-1} \int p_n(x_n)\, \log \tilde{q}_n(x_n)\,\mathrm{d}x_n \end{aligned} \tag{B.33}$$

where (a) uses the constraint $\tilde{q} \in \mathcal{F}$, (b) follows from the linearity of the integral, and p_n is the nth marginal of the joint distribution p. The sum is maximized by maximizing each term individually, and hence

$$\begin{aligned} q_n &= \arg\max_{\tilde{q}_n} \int p_n(x_n) \, \log \tilde{q}_n(x_n) \, \mathrm{d}x_n \\ &= \arg\min_{\tilde{q}_n} D_{\mathrm{KL}}(p_n \,\|\, \tilde{q}_n) \\ &= p_n. \end{aligned} \tag{B.34}$$

In words, the projection of an arbitrary p into the set of fully factorized distributions is equal to the product of marginals of p. This means in particular that by choosing $\mathcal{Q} = \mathcal{F}$ and $D = D_{\mathrm{KL}}$ in (B.26), we obtain an optimization problem

$$q = \arg\min_{\tilde{q} \in \mathcal{F}} D_{\mathrm{KL}}(p \,\|\, \tilde{q}) \tag{B.35}$$

whose solution consists of the *exact* marginals of p. Unfortunately, solving (B.35) is an intractable problem in itself for most practical distributions, but this result nonetheless motivates our focus on fully factorized approximations (which furthermore have the advantage that they lead to a trivial post-processing step (B.27)). In order to finally obtain a suboptimal, but practically implementable algorithm, we can apply several kinds of approximations to (B.35):

- A potential problem is that the marginals q_n can become arbitrarily complex, particularly for continuous variables. This can be remedied by imposing further constraints on the individual marginals, the most important example being the constraint to some exponential family. This will be the topic of the following subsection.

- We can change the divergence measure. In principle, there is an endless supply of candidates (for instance, [94] discusses the class of α-divergences which includes $D_{\mathrm{KL}}(p \,\|\, q)$ and $D_{\mathrm{KL}}(q \,\|\, p)$ as special cases), but in this thesis we limit ourselves to the Kullback-Leibler divergence, or *KL divergence* for short. We will see, however, that replacing $D_{\mathrm{KL}}(p \,\|\, q)$ by $D_{\mathrm{KL}}(q \,\|\, p)$ often leads to a significant complexity reduction (albeit usually at the expense of a reduced algorithmic performance). For reasons that are explained in Section B.4.4, $D_{\mathrm{KL}}(p \,\|\, q)$ is called the *inclusive* KL divergence, and $D_{\mathrm{KL}}(q \,\|\, p)$ the *exclusive* KL divergence.

- Rather than trying to find a one-shot solution to (B.35), which is hopeless for most distributions of practical interest anyway, we can exploit the structure of p (see Section B.2). By maintaining separate approximations for the individual factors of p, we obtain a set of equations which we can attempt to solve by iterative methods. Section B.4 presents a detailed study of an approach using fixed-point iterations.

Projections into an Exponential Family

In this subsection, we address the first point from the list above. If the true marginals are too complex, we can impose a constraint $q_n \in \mathcal{Q}_n$ on them, where

$$\mathcal{Q}_n = \left\{ q_n \mid \log q_n(x_n) = \boldsymbol{\theta}_n^\mathsf{T} \boldsymbol{\phi}_n(x_n) - A_n(\boldsymbol{\theta}_n) \right\} \tag{B.36}$$

is a suitable exponential family with sufficient statistics $\boldsymbol{\phi}_n$ and natural parameters $\boldsymbol{\theta}_n \in \Omega_n$, as introduced in Section A.2.

In order to incorporate this constraint into (B.35), we use again the projection operator as defined in (B.31). Solving for $q_n = \operatorname{proj}_{\mathcal{Q}_n}[p_n]$ boils down to finding the natural parameters $\boldsymbol{\theta}_n$ of q_n:

$$\begin{aligned} \boldsymbol{\theta}_n &= \operatorname*{arg\,max}_{\tilde{\boldsymbol{\theta}}_n \in \Omega_n} \int p_n(x_n) \, \log q_n(x_n) \, \mathrm{d}x_n \\ &= \operatorname*{arg\,max}_{\tilde{\boldsymbol{\theta}}_n \in \Omega_n} \underbrace{\int p_n(x_n) \left(\tilde{\boldsymbol{\theta}}_n^\mathsf{T} \boldsymbol{\phi}_n(x_n) - A_n(\tilde{\boldsymbol{\theta}}_n) \right) \mathrm{d}x_n}_{\triangleq f_n(\tilde{\boldsymbol{\theta}}_n)} . \end{aligned} \tag{B.37}$$

A unique stationary point of f_n is found by setting its derivative to zero:

$$\frac{\partial f_n}{\partial \tilde{\boldsymbol{\theta}}_n^\mathsf{T}} = \int p_n(x_n) \, \boldsymbol{\phi}_n(x_n) \, \mathrm{d}x_n - \frac{\partial A_n}{\partial \tilde{\boldsymbol{\theta}}_n^\mathsf{T}} \stackrel{!}{=} \mathbf{0} \tag{B.38}$$

$$\iff \mathbb{E}_{p_n}[\boldsymbol{\phi}_n] = \mathbb{E}_{q_n}[\boldsymbol{\phi}_n] \tag{B.39}$$

where (B.39) follows from (A.69), which states that the derivative of the log-partition function of an exponential family is equal to the expectation of its sufficient statistics.

The Hessian of f_n can be found using (A.71) as

$$\frac{\partial^2 f_n}{\partial \tilde{\boldsymbol{\theta}}_n \, \partial \tilde{\boldsymbol{\theta}}_n^\mathsf{T}} = -\frac{\partial^2 A_n}{\partial \tilde{\boldsymbol{\theta}}_n \, \partial \tilde{\boldsymbol{\theta}}_n^\mathsf{T}} = -\operatorname{cov}_{q_n}[\boldsymbol{\phi}_n] \tag{B.40}$$

which is negative semidefinite. This proves that the stationary point is a maximum.

Equation (B.39) is referred to as *moment matching*. For example, since the Gaussian distribution is characterized by the sufficient statistics $\boldsymbol{\phi}(x) = (x, x^2)^\mathsf{T}$, the projection of p into the family of Gaussian distributions is the particular Gaussian with the same mean and variance as p.

B.4 Divergence Minimization via Fixed-Point Iterations

In Section B.2 we have seen that exploiting the structure of the target distribution is a key step towards the derivation of efficient inference algorithms. There, our focus has been on singly-connected belief networks, which allow for exact marginalization via belief propagation. In this section, we will discuss how the structure of p can be used to derive iterative algorithms which yield approximate solutions of the divergence minimization problem (B.35).

The notion of structure essentially refers to a factorization of p,

$$p(\boldsymbol{x} \mid \boldsymbol{y}) = \frac{1}{Z} \prod_{k=0}^{K-1} f_k(\boldsymbol{x}_{\mathcal{N}_k}) \tag{B.41}$$

where each factor f_k usually depends on just a small subset of $\boldsymbol{x}$. Note that some of the f_k may depend on $\boldsymbol{y}$, but for simplicity we do not make this dependence explicit.

Recall that our aim is to find a simple auxiliary distribution q which approximates p. In order to exploit the factorization of p, we impose an analogous factorization as (B.41) on q and write

$$q(\boldsymbol{x}) = \frac{1}{\tilde{Z}} \prod_{k=0}^{K-1} \tilde{f}_k(\boldsymbol{x}_{\mathcal{N}_k}) \tag{B.42}$$

with the implicit meaning that each $\tilde{f}_k$ is an approximation of the corresponding f_k. Furthermore, q is constrained to be fully factorized, so it can also be written as

$$q(\boldsymbol{x}) = \prod_{n=0}^{N-1} q_n(x_n). \tag{B.43}$$

Since $q(\boldsymbol{x})$ is an approximation of the posterior $p(\boldsymbol{x} \mid \boldsymbol{y})$, each $q_n(x_n)$ can be interpreted as an approximation of the marginal posterior $p(x_n \mid \boldsymbol{y})$. The distributions q_n are also called *beliefs*, as they encode our belief, i. e., our probabilistic knowledge, about x_n.

By comparing (B.42) and (B.43), we see that each $\tilde{f}_k$ necessarily decomposes into a product of unary factors

$$\tilde{f}_k(\boldsymbol{x}_{\mathcal{N}_k}) \propto \prod_{n \in \mathcal{N}_k} m_{k \to n}(x_n) \tag{B.44}$$

and thus[6]

$$\begin{aligned} q(\boldsymbol{x}) &\propto \prod_{k=0}^{K-1} \prod_{n \in \mathcal{N}_k} m_{k \to n}(x_n) \\ &= \prod_{n=0}^{N-1} \prod_{k \in \mathcal{K}_n} m_{k \to n}(x_n). \end{aligned} \tag{B.45}$$

[6] As defined earlier, $\mathcal{N}_k$ contains the indices of the variables that appear in the argument list of f_k, and $\mathcal{K}_n$ contains the indices of the functions that depend on x_n.

The two expressions above merely differ in the grouping of the factors $m_{k\to n}$. Then, comparing (B.45) with (B.43), we obtain

$$q_n(x_n) \propto \prod_{k\in\mathcal{K}_n} m_{k\to n}(x_n). \tag{B.46}$$

If the $m_{k\to n}$ are members of an exponential family, this multiplication boils down to a summation of their natural parameters.

Note that the equalities above are only given up to constant factors (with respect to $\boldsymbol{x}$, which in particular includes the evidence $p(\boldsymbol{y})$). As already mentioned at the beginning of this chapter, it is not necessary to keep track of multiplicative constants, because (a) we are often just interested in $\arg\max q_n(x_n)$, which is invariant under multiplication with positive constants, and (b) we know that q_n is a distribution and therefore must be normalized. We can always recover the normalization constant from (B.46) if required.

For later use, we also define functions of the type $m_{n\to k}$ as

$$\begin{aligned} m_{n\to k}(x_n) &\propto \prod_{k'\in\mathcal{K}_n\setminus\{k\}} m_{k'\to n}(x_n) \\ &\overset{(a)}{\propto} \frac{q_n(x_n)}{m_{k\to n}(x_n)} \end{aligned} \tag{B.47}$$

where (a) holds as long as $\forall x_n \in \mathcal{X}_n : m_{k\to n}(x_n) \neq 0$, but this limitation is not relevant to us for the following reason. Without loss of generality, we can restrict $\mathcal{X}_n$ to those values which have a non-zero prior: $\mathcal{X}_n = \{x_n \mid p_n(x_n) > 0\}$. And our ubiquitous assumption that the observations can be modeled as a superposition of some function of $\boldsymbol{x}$ and additive Gaussian noise guarantees that the likelihood is strictly positive: the received data may render some values $x_n \in \mathcal{X}_n$ extremely unlikely, but never impossible. Thus, all $x_n \in \mathcal{X}_n$ necessarily have a strictly positive belief, and because of (B.46), this implies that all $m_{k\to n}$ are also strictly positive.[7]

Note that (B.47) implies that for any $k \in \mathcal{K}_n$

$$q_n(x_n) \propto m_{n\to k}(x_n)\, m_{k\to n}(x_n). \tag{B.48}$$

The iterative divergence minimization algorithm now works as follows:

- The algorithm maintains a factor $m_{k\to n}$ for each pair (k, n) with $n \in \mathcal{N}_k$.
- At the beginning, the factors are initialized to constant functions: $m_{k\to n}(x_n) \equiv 1$.
- According to some schedule, the functions $m_{k\to n}$ are updated by minimizing a *local* divergence measure. This restriction to local optimization problems is the key step which makes the algorithm practically feasible, but also suboptimal. It will be discussed in more detail below.

[7] An exception occurs in the context of hard decisions (see Section B.4.3), where the belief q_n is a Dirac distribution. In this case, $\forall k \in \mathcal{K}_n : m_{n\to k} = q_n$.

- At any time, estimates $q_n(x_n)$ of the marginals $p(x_n \mid \boldsymbol{y})$ can be obtained via (B.46).

- After a certain stopping criterion has been reached (e. g., after a fixed number of iterations, or when convergence has been detected), the algorithm terminates and outputs the most recent beliefs.

We now examine the central step, the update of the functions $m_{k\to n}$. Ideally, we would like to solve

$$\begin{aligned} m_{k\to n}^{\text{new}} &= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} D(p \,\|\, q) \\ &= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} D\left(\frac{1}{Z} \prod_{k'=0}^{K-1} f_{k'} \,\middle\|\, \frac{1}{\tilde{Z}} \prod_{k'=0}^{K-1} \tilde{f}_{k'} \right). \end{aligned} \tag{B.49}$$

The subscript of the arg min operator indicates that we minimize over $m_{k\to n}$ with the constraint $q_n = m_{n\to k}\, m_{k\to n} \in \mathcal{Q}_n$. The set $\mathcal{Q}_n$ is either a certain restricted set of distributions, e. g., an exponential family, or simply the set of *all* distributions over $\mathcal{X}_n$.

However, solving the global problem (B.49) directly is intractable by assumption. The trick for obtaining a feasible approximation is to assume that all $\tilde{f}_{k'}$, $k' \neq k$, are already good approximations of the respective $f_{k'}$, so that the local problem[8]

$$m_{k\to n}^{\text{new}} = \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} D\left(\frac{1}{Z'} f_k \prod_{k'\neq k} \tilde{f}_{k'} \,\middle\|\, \frac{1}{\tilde{Z}} \tilde{f}_k \prod_{k'\neq k} \tilde{f}_{k'} \right) \tag{B.50}$$

is a good approximation of (B.49).

The validity of the assumption $f_{k'} \approx \tilde{f}_{k'}$ is of course far from guaranteed. A typical characteristic of iterative schemes is the existence of a certain region of attraction around the global optimum. As long as the algorithm is initialized within this region, the updates will move the beliefs closer to the optimum and the algorithm will eventually converge. In digital communication systems, a standard means of improving the initial parameter estimates and hence increasing the probability of convergence to the global optimum is the transmission of some training data. However, an analytical characterization of the size or shape of the region of attraction is a hard problem, which is not studied in this thesis.

It remains to solve the local optimization problem (B.50) for particular choices of D. As mentioned earlier, we restrict ourselves to the inclusive and exclusive Kullback-Leibler divergences, which allow for analytic solutions of (B.50).

[8] Here and in the following, we simply write $k' \neq k$ and leave the bounds $0 \leq k' < K$ implicit.

B.4.1 Inclusive Kullback-Leibler Divergence

On setting $D(p \,\|\, q) = D_{\mathrm{KL}}(p \,\|\, q)$, (B.50) becomes

$$\begin{aligned} m_{k\to n}^{\text{new}} &= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} D_{\mathrm{KL}}\left(\frac{1}{Z'} f_k \prod_{k'\neq k} \tilde{f}_{k'} \,\middle\|\, \frac{1}{\tilde{Z}} \tilde{f}_k \prod_{k'\neq k} \tilde{f}_{k'} \right) \\ &= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k'\neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}}) \log \frac{f_k(\boldsymbol{x}_{\mathcal{N}_k})}{\tilde{f}_k(\boldsymbol{x}_{\mathcal{N}_k})} \,\mathrm{d}\boldsymbol{x} \\ &\overset{(a)}{=} \operatorname*{arg\,max}_{m_{k\to n}: q_n \in \mathcal{Q}_n} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k'\neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}}) \log m_{k\to n}(x_n) \,\mathrm{d}\boldsymbol{x} \end{aligned} \tag{B.51}$$

where (a) holds because the objective function depends on $m_{k\to n}$ only through $\tilde{f}_k$.

Let us first assume that $\mathcal{Q}_n$ is the set of all distributions over $\mathcal{X}_n$, so that the only constraints are $q_n(x_n) \geq 0$ and $\int q_n(x_n) \,\mathrm{d}x_n = 1$. With a similar argumentation as given after (B.47), we can assume that the optimum occurs at an interior point, i. e., that q_n is strictly positive. Therefore, we can solve (B.51) by setting the derivative of its Lagrangian to zero. The non-negativity constraint is satisfied anyway, so we only introduce a Lagrange multiplier λ for the normalization constraint, yielding

$$L(m_{k\to n}) = \int \Big(\int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k'\neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}}) \log m_{k\to n}(x_n) \,\mathrm{d}\boldsymbol{x}_{\setminus n} + \lambda \, m_{n\to k}(x_n) \, m_{k\to n}(x_n) \Big) \mathrm{d}x_n. \tag{B.52}$$

Its functional derivative (see Section A.1.4 on page 208) is

$$\frac{\delta L}{\delta m_{k\to n}} = \int \frac{f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k'\neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}})}{m_{k\to n}(x_n)} \,\mathrm{d}\boldsymbol{x}_{\setminus n} + \lambda \, m_{n\to k}(x_n) \tag{B.53}$$

which becomes zero if

$$\begin{aligned} m_{k\to n}^{\text{new}}(x_n) &\propto \frac{1}{m_{n\to k}(x_n)} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k'\neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}}) \,\mathrm{d}\boldsymbol{x}_{\setminus n} \\ &\overset{(a)}{\propto} \frac{1}{m_{n\to k}(x_n)} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k'\neq k} \prod_{n'\in\mathcal{N}_{k'}} m_{k'\to n'}(x_{n'}) \,\mathrm{d}\boldsymbol{x}_{\setminus n} \\ &\overset{(b)}{=} \frac{1}{m_{n\to k}(x_n)} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n'=0}^{N-1} \prod_{k'\in\mathcal{K}_{n'}\setminus\{k\}} m_{k'\to n'}(x_{n'}) \,\mathrm{d}\boldsymbol{x}_{\setminus n} \\ &\overset{(c)}{\propto} \frac{1}{m_{n\to k}(x_n)} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n'\in\mathcal{N}_k} \prod_{k'\in\mathcal{K}_{n'}\setminus\{k\}} m_{k'\to n'}(x_{n'}) \prod_{n''\in\mathcal{N}_k\setminus\{n\}} \mathrm{d}x_{n''} \\ &\overset{(d)}{=} \frac{1}{m_{n\to k}(x_n)} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n'\in\mathcal{N}_k} m_{n'\to k}(x_{n'}) \prod_{n''\in\mathcal{N}_k\setminus\{n\}} \mathrm{d}x_{n''} \end{aligned}$$

$$\overset{(e)}{=} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n' \in \mathcal{N}_k \setminus \{n\}} m_{n' \to k}(x_{n'}) \, \mathrm{d}x_{n'}. \tag{B.54}$$

Here, (a) uses (B.44), (b) swaps the product operators, (c) follows from integration over all $x_{n'}$ with $n' \notin \mathcal{N}_k$, which implies that $k \notin \mathcal{K}_{n'}$, (d) uses (B.47), and (e) cancels the function $m_{n \to k}$.

If we impose further constraints on q_n (i. e., if $\mathcal{Q}_n$ is a strict subset of the set of all distributions over $\mathcal{X}_n$), the optimization problem does not have a closed solution as (B.54) since it must of course depend on the particular constraint. However, we can derive a similar-looking solution with the help of the projection operator (B.31). To that end, we write $m^{\text{new}}_{k \to n}(x_n) \propto q^{\text{new}}_n(x_n) / m_{n \to k}(x_n)$ with the implicit understanding that q_n is varied only through $m_{k \to n}$, while $m_{n \to k}$ stays constant. Continuing from (B.51), the update equation for q_n can be derived as

$$\begin{aligned}
q^{\text{new}}_n &= \operatorname*{arg\,max}_{q_n \in \mathcal{Q}_n} \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k' \neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}}) \, \log q_n(x_n) \, \mathrm{d}\boldsymbol{x} \\
&= \operatorname*{arg\,max}_{q_n \in \mathcal{Q}_n} \int \left(\int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k' \neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}}) \, \mathrm{d}\boldsymbol{x}_{\setminus n} \right) \log q_n(x_n) \, \mathrm{d}x_n \\
&\overset{(a)}{=} \operatorname*{arg\,max}_{q_n \in \mathcal{Q}_n} \int \left(m_{n \to k}(x_n) \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n' \in \mathcal{N}_k \setminus \{n\}} m_{n' \to k}(x_{n'}) \, \mathrm{d}x_{n'} \right) \log q_n(x_n) \, \mathrm{d}x_n \\
&= \operatorname*{arg\,min}_{q_n \in \mathcal{Q}_n} D_{\mathrm{KL}} \left(\frac{1}{Z} \, m_{n \to k}(x_n) \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n' \in \mathcal{N}_k \setminus \{n\}} m_{n' \to k}(x_{n'}) \, \mathrm{d}x_{n'} \,\middle\|\, q_n \right) \\
&= \operatorname{proj}_{\mathcal{Q}_n} \left[\frac{1}{Z} \, m_{n \to k}(x_n) \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n' \in \mathcal{N}_k \setminus \{n\}} m_{n' \to k}(x_{n'}) \, \mathrm{d}x_{n'} \right]
\end{aligned} \tag{B.55}$$

where (a) comprises essentially the same steps as (a)–(d) in (B.54), and where Z is a normalization constant that is required because the KL divergence and the projection operator are only defined for normalized distributions.

The update equation for $m_{k \to n}$ is then

$$m^{\text{new}}_{k \to n}(x_n) \propto \frac{1}{m_{n \to k}(x_n)} \operatorname{proj}_{\mathcal{Q}_n} \left[\frac{1}{Z} \, m_{n \to k}(x_n) \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n' \in \mathcal{N}_k \setminus \{n\}} m_{n' \to k}(x_{n'}) \, \mathrm{d}x_{n'} \right] \tag{B.56}$$

which can be seen as a generalization of (B.54): if the projection is unnecessary, either because q_n is unconstrained or because the projection argument is already in $\mathcal{Q}_n$, then proj becomes the identity operator, the functions $m_{n \to k}$ cancel out, and (B.56) reduces to (B.54). In the general case, however, proj is a nonlinear operator, and canceling the $m_{n \to k}$ is not allowed.

B.4.2 Exclusive Kullback-Leibler Divergence

Using the exclusive KL divergence $D(p \,\|\, q) = D_{\mathrm{KL}}(q \,\|\, p)$ in (B.50), we obtain

$$\begin{aligned}
m_{k\to n}^{\text{new}} &= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} D_{\mathrm{KL}}\left(\frac{1}{\tilde{Z}} \tilde{f}_k \prod_{k' \neq k} \tilde{f}_{k'} \,\middle\|\, \frac{1}{Z'} f_k \prod_{k' \neq k} \tilde{f}_{k'}\right) \\
&= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} \int \tilde{f}_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{k' \neq k} \tilde{f}_{k'}(\boldsymbol{x}_{\mathcal{N}_{k'}}) \log \frac{\tilde{f}_k(\boldsymbol{x}_{\mathcal{N}_k})}{f_k(\boldsymbol{x}_{\mathcal{N}_k})} \,\mathrm{d}\boldsymbol{x} \\
&= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} \int q_n(x_n) \prod_{n' \neq n} q_{n'}(x_{n'}) \log \frac{\prod_{n'' \in \mathcal{N}_k} m_{k\to n''}(x_{n''})}{f_k(\boldsymbol{x}_{\mathcal{N}_k})} \,\mathrm{d}\boldsymbol{x} \\
&= \operatorname*{arg\,min}_{m_{k\to n}: q_n \in \mathcal{Q}_n} \int m_{n\to k}(x_n)\, m_{k\to n}(x_n) \prod_{n' \neq n} q_{n'}(x_{n'}) \log \frac{m_{k\to n}(x_n)}{f_k(\boldsymbol{x}_{\mathcal{N}_k})} \,\mathrm{d}\boldsymbol{x}. \qquad \text{(B.57)}
\end{aligned}$$

As above, we solve this problem by introducing a Lagrange multiplier λ for the normalization constraint, yielding the Lagrangian

$$\begin{aligned}
L(m_{k\to n}) = \int \Bigg(\int m_{n\to k}(x_n)\, m_{k\to n}(x_n) \prod_{n' \neq n} q_{n'}(x_{n'}) \log \frac{m_{k\to n}(x_n)}{f_k(\boldsymbol{x}_{\mathcal{N}_k})} \,\mathrm{d}\boldsymbol{x}_{\setminus n} \\
+ \lambda\, m_{n\to k}(x_n)\, m_{k\to n}(x_n) \Bigg) \mathrm{d}x_n. \qquad \text{(B.58)}
\end{aligned}$$

Its functional derivative is

$$\begin{aligned}
\frac{\delta L}{\delta m_{k\to n}} = \int \Big(m_{n\to k}(x_n) \prod_{n' \neq n} q_{n'}(x_{n'}) \log \frac{m_{k\to n}(x_n)}{f_k(\boldsymbol{x}_{\mathcal{N}_k})} + m_{n\to k}(x_n) \prod_{n' \neq n} q_{n'}(x_{n'}) \Big) \,\mathrm{d}\boldsymbol{x}_{\setminus n} \\
+ \lambda\, m_{n\to k}(x_n) \qquad \text{(B.59)}
\end{aligned}$$

which becomes zero if

$$\int \log \frac{m_{k\to n}^{\text{new}}(x_n)}{f_k(\boldsymbol{x}_{\mathcal{N}_k})} \prod_{n' \in \mathcal{N}_k \setminus \{n\}} q_{n'}(x_{n'}) \,\mathrm{d}x_{n'} = \text{const} \qquad \text{(B.60)}$$

$$\iff \log m_{k\to n}^{\text{new}}(x_n) = \int \log\left(f_k(\boldsymbol{x}_{\mathcal{N}_k})\right) \prod_{n' \in \mathcal{N}_k \setminus \{n\}} q_{n'}(x_{n'}) \,\mathrm{d}x_{n'} + \text{const} \qquad \text{(B.61)}$$

$$\iff m_{k\to n}^{\text{new}}(x_n) \propto \exp\left(\int \log\left(f_k(\boldsymbol{x}_{\mathcal{N}_k})\right) \prod_{n' \in \mathcal{N}_k \setminus \{n\}} q_{n'}(x_{n'}) \,\mathrm{d}x_{n'} \right). \qquad \text{(B.62)}$$

This derivation assumes an unconstrained q_n (besides the necessary normalization constraint). In principle, we could impose an exponential-family constraint on q_n, just as we did for the inclusive KL divergence. However, the resulting optimization problem does not have a closed-form solution, because the exclusive KL divergence

$D_{\mathrm{KL}}(\exp(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{\phi} - A(\boldsymbol{\theta})) \,\|\, p)$ is in general not convex in $\boldsymbol{\theta}$. (In contrast, the inclusive KL divergence $D_{\mathrm{KL}}(p \,\|\, \exp(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{\phi} - A(\boldsymbol{\theta})))$ *is* convex in $\boldsymbol{\theta}$, independently of p and $\boldsymbol{\phi}$, since its Hessian is equal to the Hessian of A, which is a positive semidefinite matrix.) The good news is that such a constraint is rarely needed in practice, since the result of (B.62) is typically quite simple already.

An important exception is *hard* estimation of x_n, which is included in the divergence minimization framework by constraining q_n to the family of Dirac distributions.

B.4.3 Hard Parameter Estimation

Two problems that arise frequently when dealing with continuous variables are (a) the problem of storing a probability density in finite memory, and (b) doing further computations with the density, typically involving the evaluation of expectations of certain functions with respect to the (potentially very complex) density. As we have seen, exponential families are often able to solve both problems.

Another approach for handling continuous parameters is to abandon *soft* information (expressing uncertainty by means of probability distributions) entirely, and to simply compute *hard* estimates (also called *point* estimates) instead. The concept of hard estimators plays a central role in orthodox statistics, where they are invented in a rather *ad hoc* fashion, based on heuristic criteria like *maximum likelihood* (ML) or unbiasedness. But hard estimators are also found in the context of Bayesian statistics, the most important examples being the *maximum a posteriori* (MAP) and the *minimum mean squared error* (MMSE) estimators. They can be systematically integrated into our Bayesian framework by treating the estimate $\hat{x}_n$ of x_n as the Dirac distribution

$$q_n(x_n) = \delta(x_n - \hat{x}_n). \tag{B.63}$$

Hard estimates solve the two problems mentioned above even better than exponential families: their representation only involves storing a single value $\hat{x}_n$, and the expectation of any function $f(x_n)$ with respect to a Dirac distribution is trivially $f(\hat{x}_n)$. The obvious disadvantage is that hard estimates generally lead to a worse algorithmic performance since they discard any reliability information about the estimate.

There are two situations where using hard estimates is appropriate:

- When there is sufficient data available, so that the variable can be estimated with high reliability. If the true posterior distribution is a thin spike, then replacing it with a Dirac will have a negligible impact on the algorithmic performance, and the complexity reduction makes the usage of hard estimates worthwile.

- As a last resort, when the true posterior is so involved that even the projection into an exponential family is intractable.

In this section, we discuss how hard estimates fit into the divergence minimization framework. The following notation will be used: the set of all Dirac distributions over $\mathcal{X}_n$ is denoted as $\Delta(\mathcal{X}_n)$, while $\delta_{\hat{x}_n} \in \Delta(\mathcal{X}_n)$, defined via

$$\delta_{\hat{x}_n}(x_n) \triangleq \delta(x_n - \hat{x}_n) \tag{B.64}$$

is the distribution which assigns all probability mass to the value $\hat{x}_n \in \mathcal{X}_n$.

We begin by considering the divergence minimization problem

$$\operatorname*{arg\,min}_{q_n \in \Delta(\mathcal{X}_n)} D_{\mathrm{KL}}(q_n \,\|\, Z^{-1} f) \tag{B.65}$$

with some non-negative function $f : \mathcal{X}_n \to \mathbb{R}$, and $Z \triangleq \int f(x_n)\,\mathrm{d}x_n$. Strictly speaking, the Kullback-Leibler divergence is undefined for Dirac distributions. We can circumvent this problem by instead constraining q_n to Gaussian distributions with fixed variance $\varepsilon > 0$, and defining (B.65) as the limit for $\varepsilon \to 0$. For simplicity, the following derivation assumes that x_n is a real scalar, but the generalization to other objects like vectors or matrices is straightforward.

With the constraint[9] $q_n = \mathcal{N}(\mu, \varepsilon)$, where ε is fixed and μ variable, we obtain the following optimization problem:

$$\begin{aligned} q_n &= \operatorname*{arg\,min}_{q_n = \mathcal{N}(\mu,\varepsilon)} D_{\mathrm{KL}}(q_n \,\|\, Z^{-1} f) \\ &= \operatorname*{arg\,min}_{q_n = \mathcal{N}(\mu,\varepsilon)} \int q_n(x_n)\, \log \frac{q_n(x_n)}{Z^{-1} f(x_n)}\,\mathrm{d}x_n \\ &= \operatorname*{arg\,max}_{q_n = \mathcal{N}(\mu,\varepsilon)} \int q_n(x_n)\, \log f(x_n)\,\mathrm{d}x_n + \tfrac{1}{2}\log(2\pi\mathrm{e}\varepsilon) - \log Z. \end{aligned} \tag{B.66}$$

The last two terms in (B.66) are independent of μ. Taking the limit $\varepsilon \to 0$, it remains to solve

$$\begin{aligned} \mu &= \lim_{\varepsilon \to 0} \operatorname*{arg\,max}_{\tilde{\mu}} \mathbb{E}_{\mathcal{N}(\tilde{\mu},\varepsilon)}[\log f(x_n)] \\ &= \operatorname*{arg\,max}_{\tilde{\mu}} \log f(\tilde{\mu}) \end{aligned} \tag{B.67}$$

since $\mathbb{E}_{\mathcal{N}(\mu,\varepsilon)}[\log f(x_n)] \to \log f(\mu)$ for $\varepsilon \to 0$. Thus, we define (B.65) as

$$\operatorname*{arg\,min}_{q_n \in \Delta(\mathcal{X}_n)} D_{\mathrm{KL}}(q_n \,\|\, Z^{-1} f) \triangleq \delta_{\hat{x}_n} \quad \text{with} \quad \hat{x}_n = \operatorname*{arg\,max}_{x_n \in \mathcal{X}_n} f(x_n). \tag{B.68}$$

[9] $\mathcal{N}(\mu, \varepsilon)$ is the Gaussian distribution with mean μ and variance ε. It should not be confused with $\mathcal{N}_k$, the set of variables from the argument list of f_k.

In order to incorporate the hard estimation of variable x_n into the DM framework, we must slightly modify the procedure that we discussed above. The problem is that the decomposition (B.46)

$$q_n(x_n) \propto \prod_{k \in \mathcal{K}_n} m_{k \to n}(x_n) \tag{B.69}$$

is not meaningful if q_n is a Dirac distribution. Thus, rather than updating the $m_{k \to n}$ separately based on this factorization, we update q_n directly by minimizing the exclusive KL divergence:

$$\begin{aligned}
q_n^{\text{new}} &= \underset{q_n \in \Delta(\mathcal{X}_n)}{\arg\min}\, D_{\text{KL}}(q \,\|\, p) \\
&= \underset{q_n \in \Delta(\mathcal{X}_n)}{\arg\min}\, D_{\text{KL}}\left(\prod_{n'=0}^{N-1} q_{n'} \,\middle\|\, \frac{1}{Z} \prod_{k=0}^{K-1} f_k \right) \\
&= \underset{q_n \in \Delta(\mathcal{X}_n)}{\arg\min} \int q_n(x_n) \prod_{n' \neq n} q_{n'}(x_{n'}) \, \log \frac{q_n(x_n) \prod_{n' \neq n} q_{n'}(x_{n'})}{\prod_k f_k(\boldsymbol{x}_{\mathcal{N}_k})} \, \mathrm{d}\boldsymbol{x} \\
&= \underset{q_n \in \Delta(\mathcal{X}_n)}{\arg\min} \int q_n(x_n) \, \mathbb{E}_{\prod_{n' \neq n} q_{n'}} \left[\log \frac{q_n(x_n)}{\prod_k f_k(\boldsymbol{x}_{\mathcal{N}_k})} \right] \mathrm{d}x_n \\
&= \underset{q_n \in \Delta(\mathcal{X}_n)}{\arg\min}\, D_{\text{KL}} \left(q_n \,\middle\|\, \frac{1}{Z''} \mathbb{E}_{\prod_{n' \neq n} q_{n'}} \left[\log \prod_{k=0}^{K-1} f_k(\boldsymbol{x}_{\mathcal{N}_k}) \right] \right) \\
&= \delta_{\hat{x}_n}
\end{aligned} \tag{B.70}$$

where

$$\begin{aligned}
\hat{x}_n &= \underset{x_n \in \mathcal{X}_n}{\arg\max}\, \mathbb{E}_{\prod_{n' \neq n} q_{n'}} \left[\log \prod_{k=0}^{K-1} f_k(\boldsymbol{x}_{\mathcal{N}_k}) \right] \\
&= \underset{x_n \in \mathcal{X}_n}{\arg\max} \sum_{k \in \mathcal{K}_n} \mathbb{E}_{\prod_{n' \in \mathcal{N}_k \setminus \{n\}} q_{n'}} \left[\log f_k(\boldsymbol{x}_{\mathcal{N}_k}) \right].
\end{aligned} \tag{B.71}$$

If the functions are so involved that there is no closed-form solution for the maximization problem (B.71), gradient ascent methods may be employed as proposed in [28].

Note that $\hat{x}_n$ can be seen as an approximation of the MAP estimate

$$\hat{x}_n^{\text{MAP}} \triangleq \underset{x_n \in \mathcal{X}_n}{\arg\max}\, p(x_n \,|\, \boldsymbol{y}) \tag{B.72}$$

because

$$\begin{aligned}
\hat{x}_n &= \underset{x_n \in \mathcal{X}_n}{\arg\max} \int \prod_{n' \neq n} q_{n'}(x_{n'}) \, \log p(\boldsymbol{x} \,|\, \boldsymbol{y}) \, \mathrm{d}x_{n'} \\
&= \underset{x_n \in \mathcal{X}_n}{\arg\max} \, \log p(x_n \,|\, \boldsymbol{y}) + \int \prod_{n' \neq n} q_{n'}(x_{n'}) \, \log p(\boldsymbol{x}_{\setminus n} \,|\, x_n, \boldsymbol{y}) \, \mathrm{d}x_{n'}.
\end{aligned} \tag{B.73}$$

Under the assumption that the x_n are almost independent given the data $\boldsymbol{y}$, the term $\log p(x_n \mid \boldsymbol{y})$ is much more sensitive to changes in x_n than the integral, so $\hat{x}_n \approx \hat{x}_n^{\mathrm{MAP}}$.

In the derivation above, we have always used the exclusive KL divergence. Could we also use the inclusive KL divergence for hard estimation? In order to define the projection into the set of Dirac distributions, we can write similarly to (B.66) for variable μ and fixed ε

$$\begin{aligned} q_n &= \operatorname*{arg\,min}_{q_n = \mathcal{N}(\mu,\varepsilon)} D_{\mathrm{KL}}(Z^{-1} f \,\|\, q_n) \\ &= \operatorname*{arg\,max}_{q_n = \mathcal{N}(\mu,\varepsilon)} \int Z^{-1} f(x_n)\, \log q_n(x_n)\, \mathrm{d}x_n \\ &= \operatorname*{arg\,min}_{q_n = \mathcal{N}(\mu,\varepsilon)} \int Z^{-1} f(x_n)\, (x_n - \mu)^2\, \mathrm{d}x_n. \end{aligned} \tag{B.74}$$

By setting the derivative with respect to μ to zero, it is easy to show that the minimum is attained for

$$\mu = \mathbb{E}_{Z^{-1} f}[x_n] \tag{B.75}$$

independently of ε, so it particularly holds in the limit $\varepsilon \to 0$. Thus, it makes sense to define

$$\operatorname*{arg\,min}_{q_n \in \Delta(\mathcal{X}_n)} D_{\mathrm{KL}}(Z^{-1} f \,\|\, q_n) \triangleq \delta_{\bar{x}_n} \quad \text{where} \quad \bar{x}_n = \mathbb{E}_{Z^{-1} f}[x_n]\,. \tag{B.76}$$

We can incorporate this projection into the DM framework analogously as above, yielding MMSE hard estimates

$$\bar{x}_n = \bar{x}_n^{\mathrm{MMSE}} \triangleq \mathbb{E}_{p(x_n \mid \boldsymbol{y})}[x_n]\,. \tag{B.77}$$

At first glance, this seems to be a nice complement to the MAP-like estimates obtained via exclusive divergence minimization. However, it does not appear to have any practical relevance. If we use hard estimates in order to decrease the computational complexity when the true posterior is narrowly spiked (the first bullet on page 252), then MAP and MMSE yield virtually the same estimates, anyway. And if we use them as a last resort because even projections into an exponential family are too involved, then computing the MMSE estimate is probably intractable, either. Conversely, if we were able to compute the mean of a distribution, we could usually also compute its variance, so the projection into the Gaussians would be an option. (Even if the true distribution is very non-Gaussian, a Gaussian approximation would still be better than a Dirac distribution.)

In this thesis, we always use the exclusive KL divergence for hard estimation.

B.4.4 Comparison of Inclusive vs Exclusive Divergence Minimization

Above we have derived different ways to approximate the functions f_k, based on minimizing the inclusive and exclusive Kullback-Leibler divergence. But what is the difference between them, and when should we use which one?

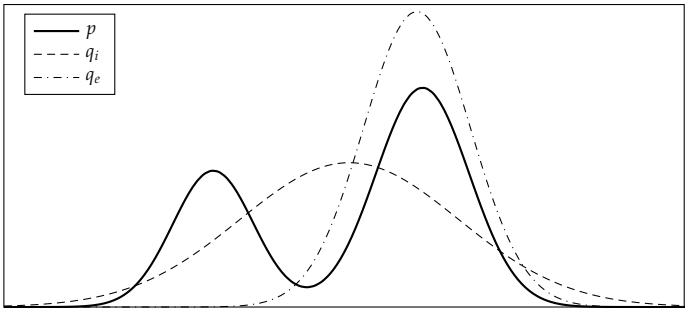


Figure B.8: Projection of a bimodal Gaussian mixture distribution p into the family of Gaussians, where $q_i = \arg\min D_{\mathrm{KL}}(p \,\|\, q)$ is the result of minimizing the *inclusive* KL divergence, while $q_e = \arg\min D_{\mathrm{KL}}(q \,\|\, p)$ is obtained by minimizing the *exclusive* KL divergence

In order to answer the first question, we first note that if the approximating family allows for a close fit to the true distribution, then we expect that minimizing either divergence will give about the same result. So in order to appreciate the difference, we need to study an example where the true distribution is quite different than even the best approximations. Inspired by the example given in [94], Figure B.8 shows the approximation of a bimodal Gaussian mixture p with a single Gaussian. The Gaussian labeled q_i is the result of minimizing the inclusive KL divergence, i. e., the result of the projection as defined in (B.31). Its mean and variance have been obtained by moment matching (B.39), i. e., by setting them to the mean and variance of the Gaussian mixture. The result is a Gaussian which covers both modes of p, even though it necessarily has its own mode in the valley between the modes of p, in a region where p is rather small. Conversely, the Gaussian labeled q_e, which has been obtained by minimizing the exclusive KL divergence, covers only the larger mode of p and almost ignores the other mode. Since there is no closed-form solution, the mean and variance of q_e have been computed by numerical optimization.

This behaviour can be explained as follows. From the definition of the inclusive KL divergence

$$D_{\mathrm{KL}}(p \,\|\, q_i) = \int p(x) \, \log \frac{p(x)}{q_i(x)} \, \mathrm{d}x \tag{B.78}$$

we see that

$$p(x) > 0 \implies q_i(x) > 0. \tag{B.79}$$

A violation of this condition (except possibly on a set of measure zero) would lead to an infinite KL divergence. Thus, q_i includes *all* regions where p is large.

Conversely, when minimizing the exclusive KL divergence,

$$p(x) = 0 \implies q_e(x) = 0 \tag{B.80}$$

so q_e cannot assign a large density to a region where p is small; it can only cover one mode while excluding the other. As a side note, this example also illustrates that minimizing the exclusive KL divergence with an exponential family constraint is in general a non-convex problem: the global minimum is attained when q_e covers the bigger mode (in terms of bigger mass, not higher density at the maximum), but there is also a local minimum when q_e covers the smaller mode. In between, there is a local maximum where q_e covers only the valley between the modes of p.

So when shall we use which divergence? Since there are no hard and fast criteria, all we can do is give some rules of thumb. A coarse ordering of the approximations from "good but complex" down to "poor but simple" is

- Inclusive KL divergence minimization using (B.54):

$$m_{k\to n}^{\text{new}}(x_n) \propto \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n'\in\mathcal{N}_k\setminus\{n\}} m_{n'\to k}(x_{n'}) \,\mathrm{d}x_{n'} \tag{B.81}$$

  As discussed on page 244, projecting p into the set of fully factorized distributions, without further constraints on the marginals q_n, would result in the exact marginals $q_n = p_n$. Of course, we are not able to solve the projection exactly for sufficiently complex p, but nonetheless, the approximate result that we obtain via the fixed-point iterations is usually still quite good. Since (B.81) does not impose an exponential family constraint, it can in practice only be used if the result of (B.81) is already sufficiently simple, e. g., if x_n is discrete. In the context of digital receivers, important examples are demapping and decoding, since those tasks involve data symbols and bits, respectively. Famous applications of (B.81) are the iterative turbo and LDPC decoding algorithms, as well as iterative detection and decoding (BICM-ID).

- Inclusive KL divergence with an exponential family constraint, using (B.56):

$$m_{k\to n}^{\text{new}}(x_n) \propto \frac{1}{m_{n\to k}(x_n)} \operatorname{proj}_{\mathcal{Q}_n}\left[\frac{1}{Z}\, m_{n\to k}(x_n) \int f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n'\in\mathcal{N}_k\setminus\{n\}} m_{n'\to k}(x_{n'}) \,\mathrm{d}x_{n'}\right] \tag{B.82}$$

  Projecting the belief into a family of simple distributions can sometimes lead to a tremendous complexity reduction. The impact on the algorithmic performance obviously depends on the accuracy of the projection. An important application of (B.82) in the context of digital receivers are LMMSE filters, used e. g. for MIMO detection, and iterative code-aided channel estimation.

- Exclusive KL divergence minimization using (B.62):

$$m_{k\to n}^{\text{new}}(x_n) \propto \exp\left(\int \log\left(f_k(\boldsymbol{x}_{\mathcal{N}_k})\right) \prod_{n'\in\mathcal{N}_k\setminus\{n\}} q_{n'}(x_{n'})\,\mathrm{d}x_{n'}\right) \tag{B.83}$$

  While discrete variables can usually be handled with the inclusive KL divergence, (B.83) often leads to significantly simpler (and also numerically more robust) formulas than (B.82). In this thesis, it is used e. g. for SNR estimation and the tracking of phase noise processes.

- Hard estimation using (B.70) and (B.71):

$$q_n^{\text{new}} = \delta_{\hat{x}_n} \quad \text{where} \quad \hat{x}_n = \arg\max_{x_n\in\mathcal{X}_n} \sum_{k\in\mathcal{K}_n} \int \log\left(f_k(\boldsymbol{x}_{\mathcal{N}_k})\right) \prod_{n'\in\mathcal{N}_k\setminus\{n\}} q_{n'}(x_{n'})\,\mathrm{d}x_{n'} \tag{B.84}$$

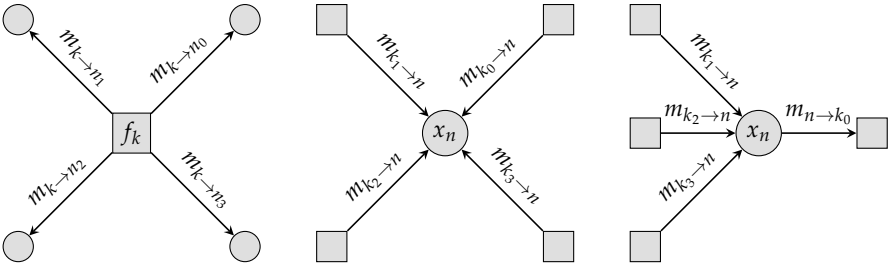
  Clearly, reducing a distribution over $\mathcal{X}_n$ to just a single value $\hat{x}_n \in \mathcal{X}_n$ is the simplest possible approximation, but also the poorest one.

Due to the lack of a thorough, systematic way for selecting the best approximation strategy for a given inference task, there is not much choice but to exercise pragmatism: do the math, implement the algorithms in a simulation environment, compare them in terms of performance and complexity, and finally come to an educated decision.

B.4.5 A Message-Passing Interpretation of Divergence Minimization

As discussed in Section B.2.2, belief propagation is often presented as a message passing algorithm, which operates on a graphical model of the belief network. While this interpretation is by no means of any fundamental importance, it is still a nice and intuitive way which helps the algorithm designer to identify the structure of the underlying problem, and to visualize the exchange of information between the variables. Also, some properties are expressed more naturally with graphical models than using plain mathematics. For instance, it is well known that loops of length 4 are detrimental to the performance of LDPC codes, and hence should be avoided. Expressing this statement (and identifying the problem in the first place) without the aid of graphical models would be more intricate.

In [94] it has been shown that the generic divergence minimization framework also lends itself to such an interpretation. As long as the approximating distribution is fully factorized (as we have assumed throughout the derivation above), *factor graphs* are well suited for this task. Remember that a factor graph is bipartite. It contains a factor node for each function f_k, $k \in \{0,\dots,K-1\}$, a variable node for each variable x_n, $n \in \{0,\dots,N-1\}$, and an edge connecting the nodes f_k and x_n if and only if f_k



(a) The fully factorized approximation of $\tilde{f}_k$ is the product of all outgoing messages from the factor node f_k

(b) The belief q_n is given by the product of all incoming messages at the variable node x_n as in (B.85)

(c) An outgoing message from a variable node is the product of the incoming messages along all other edges (B.86)

Figure B.9: Messages, beliefs, and fully factorized approximations in a factor graph

depends on x_n. Thus, $\mathcal{N}_k$ can be seen as a function which returns the set of neighbors of the factor node f_k, while $\mathcal{K}_n$ returns the set of neighbors of the variable node x_n.

Associated with each edge are two functions $m_{k\to n}(x_n)$ and $m_{n\to k}(x_n)$, which are respectively interpreted as the message sent from factor node f_k to variable node x_n, and the message in the reverse direction, from variable to factor node. The message passing algorithm proceeds by iteratively refining the factor-to-variable messages according to the update equations that are summarized in the previous subsection. The beliefs q_n and variable-to-factor messages $m_{n\to k}$, which appear in those update equations, are just shortcuts for products of factor-to-variable messages, defined in (B.46) and (B.47):

$$q_n(x_n) \propto \prod_{k\in\mathcal{K}_n} m_{k\to n}(x_n) \tag{B.85}$$

$$m_{n\to k}(x_n) \propto \prod_{k'\in\mathcal{K}_n\setminus\{k\}} m_{k'\to n}(x_n) \propto \frac{q_n(x_n)}{m_{k\to n}(x_n)}. \tag{B.86}$$

Hard constraints are an exception: if $q_n = \delta_{\hat{x}_n}$, then $\forall k \in \mathcal{K}_n : m_{n\to k} = \delta_{\hat{x}_n}$ as well.

B.5 Special Cases of Divergence Minimization

We conclude this chapter with a brief overview of some state-of-the-art techniques for statistical inference. We will see that they are all included in the generic divergence minimization framework as special cases. They also serve as building blocks for the receiver algorithms that are discussed in this thesis. The virtue of the generic divergence minimization approach is that it provides a systematic way for deriving composite receivers whose subparts belong to different algorithmic classes. In the context of digital receiver design, the ability to mix different inference techniques is of vital importance, since this domain naturally comprises both discrete (bits, data symbols) and continuous variables (channel weights, synchronization parameters). There are several examples in the communication literature where such algorithmic blends have been derived heuristically, and where the DM framework enables us to propose improvements for these *ad hoc* algorithms, or to explain phenomena which have been observed experimentally, but which so far have lacked a theoretical understanding.

B.5.1 Loopy Belief Propagation (BP)

In Section B.2.2, *belief propagation* [104] has been presented as an exact marginalization technique for tree-structured graphical models.[10] Essentially, it is a systematic way for minimizing the number of arithmetic operations by means of the distributive law, exploiting the structure of the belief network. Belief propagation terminates in finite time and returns the exact marginals.

While BP in its strict form can only be applied to single-connected graphs, it purely consists of local computations which are ignorant of the global structure, so nothing prevents us from running BP in belief networks with loops. The resulting algorithm, known as *loopy belief propagation*, is both iterative and approximate (since information can flow in circles and eventually gets double-counted), which caused it to be largely ignored at first [41]. It became widely appreciated only after the discovery of turbo codes [11, 12], when researchers recognized that it provides the theoretical background of the celebrated turbo decoder [68, 86]. This finding then sparked research on the application of loopy BP to other inference problems than channel decoding [97].

While loopy belief propagation has traditionally been presented as a heuristic application of BP to graphs with cycles, it is interesting that the divergence minimization approach leads to precisely the same algorithm. In fact, (B.86) and (B.81) are nothing but the variable-to-function and function-to-variable message update equations of

[10] In his paper [104], Pearl used a different kind of graphical model, called *Bayesian network*, but the gist of BP on Markov and Bayesian networks is the same. See also [105] and [65].

loopy belief propagation in factor graphs (c. f. (5) and (6) in [69]), particularly if we replace (B.81) by its discrete version

$$m_{k\to n}^{\text{new}}(x_n) \propto \sum_{\boldsymbol{x}_{\mathcal{N}_k\setminus\{n\}}} f_k(\boldsymbol{x}_{\mathcal{N}_k}) \prod_{n'\in\mathcal{N}_k\setminus\{n\}} m_{n'\to k}(x_{n'}). \tag{B.87}$$

B.5.2 Expectation Propagation (EP)

Expectation propagation is a technique for approximating posteriors in belief networks, which has originally been proposed as a generalization of *assumed density filtering* (ADF) [91, 92]. Later it has been formulated as a message passing algorithm [94], which has made the tight connection of EP and BP obvious. In fact, the only difference is that EP uses (B.82) rather than (B.81) for the function-to-variable message update, i. e., that it performs a projection of the beliefs into an exponential family.

The fact that projecting a distribution p into an exponential family with sufficient statistics $\boldsymbol{\phi}$ boils down to computing the expectation $\mathbb{E}_p[\boldsymbol{\phi}(x)]$ explains the name: While BP propagates the whole *beliefs* of the variables through the network, EP only propagates the *expectations* of their sufficient statistics.

B.5.3 Expectation Maximization (EM, BEM)

Expectation maximization is an iterative method for solving intractable maximum likelihood (ML) estimation problems. It has originally been presented in [32], and an accessible overview of the algorithm can be found in [96].

Assume we are given some data y, drawn from a distribution $p(y;\theta)$ which depends on a parameter θ. The original formulation of the EM algorithm is rooted in orthodox statistics and thus models θ as a deterministic quantity rather than a random variable; in particular, it does not assign a prior distribution to θ. We wish to solve the maximum likelihood estimation problem

$$\hat{\theta}^{\text{ML}} \triangleq \arg\max_{\theta} p(y;\theta). \tag{B.88}$$

The EM algorithm assumes the existence of some *hidden data* x, such that, if x were known, maximizing $p(x,y;\theta)$ over θ would be easy. (Note that the bold $\boldsymbol{x}$ in [96], the so-called *complete data*, comprises both x and y in our notation.)

Starting from an initial estimate $\hat{\theta}^{(0)}$, where the superscript denotes an iteration index, the EM algorithm proceeds by iteratively applying the following two steps.

- Expectation step (E-step):

$$Q(\theta;\hat{\theta}^{(i)}) = \mathbb{E}_{p(x\,|\,y;\hat{\theta}^{(i)})}[\log p(x,y;\theta)] \tag{B.89}$$

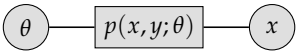


Figure B.10: Factor graph for the EM algorithm (note that y is known and is therefore not explicitly represented by a variable node)

- Maximization step (M-step):

$$\hat{\theta}^{(i+1)} = \arg\max_{\theta} Q(\theta; \hat{\theta}^{(i)}). \tag{B.90}$$

To see how this algorithm can be derived as a special case of DM, we start by drawing the (admittedly not very exciting) factor graph shown in Figure B.10. The belief q_x is unconstrained, whereas q_θ is constrained to Dirac distributions, $q_\theta^{(i)} = \delta_{\hat{\theta}^{(i)}}$.

We start by initializing $\hat{\theta}^{(0)}$ (how to choose the initial estimate is application-specific and not prescribed by the EM algorithm). One iteration of DM then proceeds as follows. First, we update q_x according to

$$\begin{aligned} q_x^{(i)}(x) &\propto m_{p\to x}^{(i)}(x) \\ &= \mathbb{E}_{q_\theta^{(i)}}[p(x, y; \theta)] \\ &= p(x, y; \hat{\theta}^{(i)}). \end{aligned} \tag{B.91}$$

Normalization of $q_x^{(i)}$ is trivial:

$$q_x^{(i)}(x) = p(x \mid y; \hat{\theta}^{(i)}). \tag{B.92}$$

Then, we update q_θ by applying (B.84)

$$\begin{aligned} \hat{\theta}^{(i+1)} &= \arg\max_{\theta} \int \log\big(p(x, y; \theta)\big)\, q_x^{(i)}(x)\, \mathrm{d}x \\ &= \arg\max_{\theta} \mathbb{E}_{p(x \mid y; \hat{\theta}^{(i)})}[\log p(x, y; \theta)]\,. \end{aligned} \tag{B.93}$$

One EM-iteration can be broken down into three steps:

1. Compute $p(x \mid y; \hat{\theta}^{(i)})$
2. Compute $\mathbb{E}_{p(x \mid y; \hat{\theta}^{(i)})}[\log p(x, y; \theta)]$
3. Compute $\hat{\theta}^{(i+1)} = \arg\max_\theta \mathbb{E}_{p(x \mid y; \hat{\theta}^{(i)})}[\log p(x, y; \theta)]$.

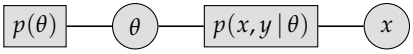


Figure B.11: Factor graph for the Bayesian EM algorithm

The traditional presentation of the EM algorithm merges the first two steps into the E-step, while the last one is the M-step. In the derivation via the DM framework, 1. is taken as a separate step (the update of q_x), while 2. and 3. together constitute another step (the update of q_θ). The overall algorithm, however, is identical.

The *Bayesian EM* algorithm (BEM) [46] is a variant of EM, which models θ as a random variable and thus assigns it a prior distribution p_θ. It is easily derived that the only difference to the classical EM algorithm is to account for the log-prior when updating $\hat{\theta}$:

$$\hat{\theta}^{(i+1)} = \arg\max_{\theta} \mathbb{E}_{p(x \mid y, \hat{\theta}^{(i)})}[\log p(x, y \mid \theta)] + \log p(\theta). \tag{B.94}$$

Note that the classical EM algorithm can be regarded as a special case of Bayesian EM (at least formally); it is obtained by assuming a uniform prior for θ.

B.5.4 Space-Alternating Generalized EM (SAGE)

The space-alternating generalized expectation maximization algorithm [38] is a generalization of EM which assumes a parameter vector $\boldsymbol{\theta}$. In each iteration, it only updates a subset of components of $\boldsymbol{\theta}$. As long as the components can be split into a partition $(\theta_0, \ldots, \theta_{N-1})$, where exactly one θ_n is updated per iteration, SAGE can be obtained as a special case of DM by introducing separate variable nodes for each θ_n.

As an example, consider the problem of estimating static, frequency selective channels in a multiple-input-single-output (MISO) OFDM system, as discussed in [153]. The transmitter is equipped with N_t antennas. The link from the nth transmit antenna to the receiver is characterized by the channel impulse response $\boldsymbol{h}_n$ of length L. The OFDM system uses K subcarriers, $K > L$, so the frequency domain channel response is $\boldsymbol{F}\boldsymbol{h}_n$, where $\boldsymbol{F}$ is a $K \times L$ matrix, consisting of the first L columns of the $K \times K$ FFT matrix. The nth transmit antenna sends the frequency-domain data symbols $\boldsymbol{x}_n$, which are assumed to be known at the receiver (they are either training data, or the receiver operates in a decision-directed mode). Assuming perfect synchronization, the received signal in the frequency domain is

$$\boldsymbol{y} = \sum_{n=0}^{N_\mathrm{t}-1} \boldsymbol{X}_n \boldsymbol{F} \boldsymbol{h}_n + \boldsymbol{w} \tag{B.95}$$

where $\boldsymbol{X}_n = \mathrm{diag}(\boldsymbol{x}_n)$ and $\boldsymbol{w} \sim \mathcal{CN}(\boldsymbol{0}, N_0 \boldsymbol{I}_K)$. Our task is to estimate the channels $\boldsymbol{h}_n$.

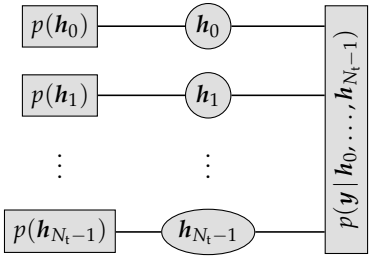


Figure B.12: Factor graph for the application of SAGE to MISO-OFDM channel estimation

If we model the channel weights as deterministic (equivalently, assign improper uniform priors), a one-shot solution is given by the method of least squares as

$$\hat{\boldsymbol{h}}^{\mathrm{LS}} = (\boldsymbol{X}^{\mathsf{H}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\mathsf{H}}\boldsymbol{y} = \boldsymbol{X}^{+}\boldsymbol{y} \tag{B.96}$$

where $\boldsymbol{h} = (\boldsymbol{h}_0^{\mathsf{T}}, \ldots, \boldsymbol{h}_{N_t-1}^{\mathsf{T}})^{\mathsf{T}}$ and $\boldsymbol{X} = (\boldsymbol{X}_0\boldsymbol{F}, \ldots, \boldsymbol{X}_{N_t-1}\boldsymbol{F})$. However, this requires the computation of the pseudo-inverse of a $K \times N_t L$ matrix (assuming $K \geq N_t L$).

In order to decrease the computational complexity, [153] proposes to apply the SAGE algorithm to this problem. The vector $\boldsymbol{h}$ of unknown parameters (corresponding to $\boldsymbol{\theta}$ above) is partitioned into the $N = N_t$ links $\boldsymbol{h}_0, \ldots, \boldsymbol{h}_{N_t-1}$, and per iteration, one estimate $\hat{\boldsymbol{h}}_n$ is updated. The comb-structured factor graph of this problem is shown in Figure B.12. The update equations are easily derived as follows

$$\begin{aligned}
\hat{\boldsymbol{h}}_n^{\mathrm{new}} &= \arg\max_{\boldsymbol{h}_n \in \mathbb{C}^L} \mathbb{E}_{\prod_{n' \neq n} q(\boldsymbol{h}_{n'})}[\log p(\boldsymbol{y} \mid \boldsymbol{h}_0, \ldots, \boldsymbol{h}_{N_t-1})] \\
&= \arg\min_{\boldsymbol{h}_n \in \mathbb{C}^L} \Big\| \underbrace{\boldsymbol{y} - \sum_{n' \neq n} \boldsymbol{X}_{n'}\boldsymbol{F}\hat{\boldsymbol{h}}_{n'}}_{\triangleq \tilde{\boldsymbol{y}}_n} - \boldsymbol{X}_n\boldsymbol{F}\boldsymbol{h}_n \Big\|^2 \\
&= (\boldsymbol{F}^{\mathsf{H}}\boldsymbol{X}_n^{\mathsf{H}}\boldsymbol{X}_n\boldsymbol{F})^{-1}\boldsymbol{F}^{\mathsf{H}}\boldsymbol{X}_n^{\mathsf{H}}\tilde{\boldsymbol{y}}_n
\end{aligned} \tag{B.97}$$

which only involves the inversion of an $L \times L$ matrix.

Note that the DM approach has allowed us to derive (B.97) in a simple and systematic manner, without the need for artificial concepts like "complete" and "incomplete" data spaces. Having understood the general principle, it is easy to derive further modifications. For example, if the inversion of an $L \times L$ matrix is still too expensive, we can break the parameter vector $\boldsymbol{h}$ down into smaller pieces. As an extreme case, we could split $\boldsymbol{h}$ into its $N_t L$ scalar coefficients, introducing a separate variable node

for each of them. The update equation for $\hat{h}_{n,l}$ is derived analogously as above (see also [155])

$$\hat{h}_{n,l}^{\text{new}} = \underset{h_{n,l}\in\mathbb{C}}{\arg\min} \Big\| \underbrace{\boldsymbol{y} - \sum_{n'\neq n} \boldsymbol{X}_{n'} \boldsymbol{F} \hat{\boldsymbol{h}}_{n'} - \boldsymbol{X}_n \sum_{l'\neq l} \boldsymbol{f}_{l'} \hat{h}_{n,l'}}_{\triangleq \tilde{\boldsymbol{y}}_{n,l}} - \boldsymbol{X}_n \boldsymbol{f}_l h_{n,l} \Big\|^2$$

$$= \frac{\boldsymbol{f}_l^{\mathsf{H}} \boldsymbol{X}_n^{\mathsf{H}} \tilde{\boldsymbol{y}}_{n,l}}{\boldsymbol{f}_l^{\mathsf{H}} \boldsymbol{X}_n^{\mathsf{H}} \boldsymbol{X}_n \boldsymbol{f}_l} \tag{B.98}$$

which only requires a scalar division. Above, $\boldsymbol{f}_l$ denotes the lth column of $\boldsymbol{F}$.

Further generalizations that are straightforward within the DM framework include the assumption of Gaussian rather than uniform priors, and the extension to imperfect knowledge of the transmitted data.

B.5.5 Mean Field (MF) and Variational Message Passing (VMP)

The EM algorithm distinguishes two unknown quantities: the parameter θ that we wish to estimate, and the hidden data x. They are treated asymmetrically in two ways.

- The parameter θ is modeled as deterministic and therefore does not have a prior distribution, while the hidden data x is modeled as a random variable. Thus, even if prior knowledge about θ is available, it cannot be used by the classical EM algorithm. As discussed above, this issue is remedied by the Bayesian EM algorithm, which treats all unknowns as random variables.

- The more important difference is that both EM and BEM make hard decisions about θ, while soft knowledge about x is retained in terms of the posterior distribution $p(x \mid y, \hat{\theta})$.

In the original formulation of EM, it is θ that we are ultimately interested in, while x is an auxiliary quantity, sometimes essentially made up out of thin air, introduced with the sole purpose of simplifying the estimation of θ. However, the second difference outlined above has the somewhat paradoxical consequence that the EM algorithm harvests more information about the nuisance data than about the parameter of interest!

As an example, consider again the problem of channel estimation, but this time with *unknown* data symbols (joint iterative channel estimation and data detection). Without doubt, we are ultimately interested in the data symbols, whereas the channel coefficients are nuisance parameters. Thus, we should identify the data symbols with θ and the channel weights with x. But this inevitably leads to hard data decisions, which makes this scheme unsuitable for coded transmissions, since modern channel decoders (like turbo and LDPC decoders) require soft input. Hence, it is actually preferable to identify θ with the channel weights and x with the data symbols (see [44] and references therein).

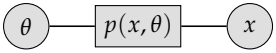


Figure B.13: Factor graph for the MF algorithm with two variables

The lack of an objective rule for mapping θ and x to the quantities of the problem domain can cause the derivation of EM algorithms degenerate to trial-and-error. Clearly, there is something fundamentally wrong with the distinction between parameters and nuisance data, which is artificial and unsuitable for (at least some) practical problems.

With the identification of EM as a special case of DM, as discussed in Section B.5.2, the solution is obvious: we just remove the Dirac-constraint from q_θ and hence retain soft information for both θ and x. Based on the factor graph shown in Figure B.13, and minimizing the exclusive KL divergence (which is also the basis of EM), the update equations for the two beliefs are easily derived as

$$q_x^{\text{new}}(x) = \exp\left(\mathbb{E}_{q_\theta}[\log p(x,\theta)]\right) \tag{B.99}$$

$$q_\theta^{\text{new}}(\theta) = \exp\left(\mathbb{E}_{q_x}[\log p(x,\theta)]\right) \tag{B.100}$$

which are iterated until convergence. Note that θ and x are now treated on equal footing, so for simplicity we drop the different notation and just call them x_0 and x_1. An obvious generalization is then to allow for more than two variables, which finally leads us to the general update equation

$$q_n^{\text{new}}(x_n) = \exp\left(\mathbb{E}_{\prod_{n'\neq n} q_{n'}}[\log p(x_0,\ldots,x_{N-1})]\right). \tag{B.101}$$

This is nothing but the well-known *naïve mean field*[11] algorithm, whose name is rooted in statistical physics [17]. In the context of statistical inference, mean field algorithms are also called *variational methods* [10, 60, 63].

If p factorizes as in (B.13), the naïve mean field algorithm can be implemented by local computations, giving rise to a message-passing interpretation, where, since we are minimizing the exclusive KL divergence, the messages are computed by (B.83). The resulting algorithm is known as *variational message passing*, and has originally been proposed in the context of Bayesian networks [144, 145]. Later, a formulation for factor graphs has been given in [27].

[11]The opposite of *naïve* mean field is *structured* mean field. The difference is that naïve MF uses fully factorized auxiliary distributions, while structured MF allows for more involved approximations like chains or trees [113]. As a side note, similar generalizations are possible for inclusive KL divergence minimization, see e. g. [95].

Bibliography

[1] T. Adalı, P. J. Schreier, and L. L. Scharf, "Complex-valued signal processing: The proper way to deal with impropriety," *IEEE Transactions on Signal Processing*, vol. 59, no. 11, pp. 5101–5125, Nov. 2011.

[2] M. Adrat, T. Clevorn, and L. Schmalen, "Iterative source-channel decoding & turbo decodulation," in *Advances in Digital Speech Transmission*, R. Martin, U. Heute, and C. Antweiler, Eds. John Wiley & Sons, Ltd, 2008, ch. 13, pp. 365–398.

[3] E. Akay and E. Ayanoglu, "Low complexity decoding of bit-interleaved coded modulation for M-ary QAM," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Paris, France, Jun. 2004.

[4] S. M. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, pp. 1451–1458, Oct. 1998.

[5] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2657–2673, Nov. 2004.

[6] D. Auras, R. Leupers, and G. Ascheid, "VLSI implementation of linear MIMO detection with boosted communications performance," in *Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI)*, Houston, TX, USA, May 2014.

[7] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, Mar. 1974.

[8] J. Baltersee, G. Fock, and H. Meyr, "An information theoretic foundation of synchronized detection," *IEEE Transactions on Communications*, vol. 49, no. 12, pp. 2115–2123, Dec. 2001.

[9] A. Barbieri, G. Colavolpe, and G. Caire, "Joint iterative detection and decoding in the presence of phase noise and frequency offset," *IEEE Transactions on Communications*, vol. 55, no. 1, pp. 171–179, Jan. 2007.

[10] M. J. Beal, "Variational algorithms for approximate Bayesian inference," Ph.D. dissertation, University of London, 2003.

[11] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261–1271, Oct. 1996.

[12] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Geneva, Switzerland, May 1993.

[13] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[14] S. Bittner, S. Krone, and G. Fettweis, "Tutorial on discrete time phase noise modeling for phase locked loops." https://mns.ifn.et.tu-dresden.de/personalSites/stefan.krone/Documents/Bittner_S_PN_Tutorial.pdf

[15] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[16] B. H. Brandwood, "A complex gradient operator and its application in adaptive array theory," *Microwaves, Optics and Antennas, IEE Proceedings H*, vol. 130, no. 1, pp. 11–16, Feb. 1983.

[17] E. Brézin, *Introduction to Statistical Field Theory*. Cambridge, UK: Cambridge University Press, 2010.

[18] M. R. G. Butler and I. B. Collings, "A zero-forcing approximate log-likelihood receiver for MIMO bit-interleaved coded modulation," *IEEE Communications Letters*, vol. 8, no. 2, pp. 105–107, Feb. 2004.

[19] G. Caire, G. Taricco, and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 927–946, May 1998.

[20] A. Chindapol and J. A. Ritcey, "Design, analysis, and performance evaluation for BICM-ID with square QAM constellations in Rayleigh fading channels," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 5, pp. 944–957, May 2001.

[21] R. H. Clarke, "A statistical theory of mobile-radio reception," *Bell System Technical Journal*, vol. 47, pp. 957–1000, Jul. 1968.

[22] G. Colavolpe, A. Barbieri, and G. Caire, "Algorithms for iterative decoding in the presence of strong phase noise," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 9, pp. 1748–1757, Sep. 2005.

[23] I. B. Collings, M. R. G. Butler, and M. R. McKay, "Low complexity receiver design for MIMO bit-interleaved coded modulation," in *Proceedings of the IEEE International Symposium on Spread Spectrum Techniques and Applications (ISSSTA)*, Sydney, Australia, Aug. 2004.

[24] E. Dahlman, S. Parkvall, and J. Sköld, *4G—LTE / LTE-Advanced for Mobile Broadband*. Academic Press, 2011.

[25] M. O. Damen, H. E. Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Transactions on Information Theory*, vol. 49, no. 10, pp. 2389–2402, Oct. 2003.

[26] A. Das, "NDA SNR estimation: CRLBs and EM based estimators," in *Proceedings of the IEEE Region 10 Conference (TENCON)*, Hyderabad, India, Nov. 2008.

[27] J. Dauwels, "On variational message passing on factor graphs," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Nice, France, Jun. 2007.

[28] J. Dauwels, S. Korl, and H.-A. Loeliger, "Steepest descent as message passing," in *Proceedings of the IEEE Information Theory Workshop (ITW)*, Rotorua, New Zealand, Aug. 2005.

[29] J. Dauwels, S. Korl, and H.-A. Loeliger, "Particle methods as message passing," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Seattle, WA, USA, Jul. 2006.

[30] J. Dauwels and H.-A. Loeliger, "Phase estimation by message passing," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Paris, France, Jun. 2004.

[31] A. Demir, A. Mehrotra, and J. Roychowdhury, "Phase noise in oscillators: A unifying theory and numerical methods for characterization," *IEEE Transactions on Circuits and Systems—Part I: Fundamental Theory and Applications*, vol. 47, no. 5, pp. 655–674, May 2000.

[32] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, Jan. 1977.

[33] Z. Dienes, "Bayesian versus orthodox statistics: Which side are you on?" *Perspectives on Psychological Science*, vol. 6, no. 3, pp. 274–290, May 2011.

[34] P. Duhamel and M. Kieffer, *Joint Source-Channel Decoding: A Cross-Layer Perspective with Applications in Video Broadcasting over Mobile and Wireless Networks*. Oxford, UK: Academic Press, 2010.

[35] M. Dörpinghaus, G. Ascheid, H. Meyr, and R. Mathar, "Optimal PSK signaling over stationary Rayleigh fading channels," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Toronto, Canada, Jul. 2008.

[36] J. Eriksson, E. Ollila, and V. Koivunen, "Essential statistics and tools for complex random variables," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5400–5408, Oct. 2010.

[37] P. Fertl, J. Jaldén, and G. Matz, "Performance assessment of MIMO-BICM demodulators based on mutual information," *IEEE Transactions on Signal Processing*, vol. 60, no. 3, pp. 1366–1382, Mar. 2012.

[38] J. A. Fessler and A. O. Hero, "Space-alternating generalized expectation-maximization algorithm," *IEEE Transactions on Signal Processing*, vol. 42, no. 10, pp. 2664–2677, Oct. 1994.

[39] G. J. Foschini and M. J. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, no. 3, pp. 311–335, Mar. 1998.

[40] B. J. Frey and N. Jojic, "A comparison of algorithms for inference and learning in probabilistic graphical models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 9, pp. 1392–1416, Sep. 2005.

[41] B. J. Frey and D. J. C. MacKay, "A revolution: Belief propagation in graphs with cycles," in *Advances in Neural Information Processing Systems 10*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds. Cambridge, MA, USA: MIT Press, 1998, pp. 479–485.

[42] J. G. David Forney, "Codes on graphs: Normal realizations," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 520–548, Feb. 2001.

[43] R. G. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[44] A. S. Gallo and G. M. Vitetta, "Soft-in soft-output detection in the presence of parametric uncertainty via the Bayesian EM algorithm," *EURASIP Journal on Wireless Communications and Networking*, vol. 2005, no. 2, pp. 100–116, Apr. 2005.

[45] W. Gappmair, R. López-Valcarce, and C. Mosquera, "Cramér-Rao lower bound and EM algorithm for envelope-based SNR estimation of nonconstant modulus constellations," *IEEE Transactions on Communications*, vol. 57, no. 6, pp. 1622–1627, Jun. 2009.

[46] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian Data Analysis*, 2nd ed. Boca Raton, FL, USA: Chapman and Hall/CRC, 2004.

[47] L. C. Godara, "Application of antenna arrays to mobile communications, part II: Beamforming and direction-of-arrival considerations," *Proceedings of the IEEE*, vol. 85, no. 8, pp. 1195–1245, Aug. 1997.

[48] L. C. Godara, "Applications of antenna arrays to mobile communications, part I: Performance improvement, feasibility, and system considerations," *Proceedings of the IEEE*, vol. 85, no. 7, pp. 1031–1060, Jul. 1997.

[49] S. Godtmann, N. Hadaschik, A. Pollok, G. Ascheid, and H. Meyr, "Iterative code-aided phase noise synchronization based on the LMMSE criterion," in *Proceedings of the IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Helsinki, Finland, Jun. 2007.

[50] A. Goldsmith, S. A. Jafar, N. Jindal, and S. Vishwanath, "Capacity limits of MIMO channels," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 5, pp. 684–702, Jun. 2003.

[51] A. Guillén i Fàbregas, A. Martinez, and G. Caire, "Bit-interleaved coded modulation," *Foundations and Trends in Communications and Information Theory*, vol. 5, no. 1–2, pp. 1–153, Jan. 2008.

[52] N. Görtz, "On the iterative approximation of optimal joint source-channel decoding," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 9, pp. 1662–1670, Sep. 2001.

[53] J. Hagenauer, "The turbo principle in mobile communications," in *Proceedings of the International Symposium on Nonlinear Theory and its Applications (NOLTA)*, Xi'an, China, Oct. 2002.

[54] C. Herzet, N. Noels, V. Lottici, H. Wymeersch, M. Luise, M. Moeneclaey, and L. Vandendorpe, "Code-aided turbo synchronization," *Proceedings of the IEEE*, vol. 95, no. 6, pp. 1255–1271, Jun. 2007.

[55] C. Herzet, V. Ramon, and L. Vandendorpe, "A theoretical framework for iterative synchronization based on the sum-product and the expectation-maximization algorithms," *IEEE Transactions on Signal Processing*, vol. 55, no. 5, pp. 1644–1658, May 2007.

[56] A. Hjørunges and D. Gesbert, "Complex-valued matrix differentiation: Techniques and key results," *IEEE Transactions on Signal Processing*, vol. 55, no. 6, pp. 2740–2746, Jun. 2007.

[57] A. Hjørungnes, *Complex-Valued Matrix Derivatives*. Cambridge, UK: Cambridge University Press, 2011.

[58] B. M. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Transactions on Communications*, vol. 51, no. 3, pp. 389–399, Mar. 2003.

[59] A. Ibing, D. Kühling, and H. Boche, "On the relation of MIMO APP detection and SIMO maximum ratio combining," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Honolulu, HI, USA, Nov. 2009.

[60] T. S. Jaakkola and M. I. Jordan, "Bayesian parameter estimation via variational methods," *Statistics and Computing*, vol. 10, no. 1, pp. 25–37, 2000.

[61] J. Jaldén and B. Ottersten, "On the complexity of sphere decoding in digital communications," *IEEE Transactions on Signal Processing*, vol. 53, no. 4, pp. 1474–1484, Apr. 2005.

[62] E. T. Jaynes, *Probability Theory: The Logic of Science*. Cambridge, UK: Cambridge University Press, 2003.

[63] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine Learning*, vol. 37, no. 2, pp. 183–233, 1999.

[64] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ, USA: Prentice Hall, 1993, vol. 1.

[65] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: MIT Press, 2009.

[66] K. Kreutz-Delgado, "Real vector derivatives, gradients, and nonlinear least-squares," University of California, San Diego, CA, USA, 2008. http://dsp.ucsd.edu/~kreutz/PEI-05SupportFiles/RealVectorDerivativesFall2008.pdf

[67] K. Kreutz-Delgado, "The complex gradient operator and the CR-calculus," University of California, San Diego, CA, USA, 2009. http://arxiv.org/abs/0906.4835v1

[68] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 219–230, Feb. 1998.

[69] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[70] A. Lapidoth, *A Foundation in Digital Communication*. New York, NY, USA: Cambridge University Press, 2009.

[71] E. G. Larsson, F. Tufvesson, O. Edfors, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[72] E. A. Lee and D. G. Messerschmitt, *Digital Communication*, 2nd ed. Boston, MA, USA: Kluwer, 1994.

[73] H. Lee, B. Lee, and I. Lee, "Iterative detection and decoding with an improved V-BLAST for MIMO-OFDM systems," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3, pp. 504–513, Mar. 2006.

[74] H. Li and T. Adalı, "Complex-valued adaptive signal processing using nonlinear functions," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, no. 1, pp. 1–9, Jan. 2008.

[75] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding," *IEEE Communications Letters*, vol. 1, no. 6, pp. 169–171, Nov. 1997.

[76] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding using soft feedback," *IET Electronic Letters*, vol. 34, no. 10, pp. 942–943, May 1998.

[77] D. V. Lindley, "The future of statistics—A Bayesian 21st century," *Advances in Applied Probability*, vol. 7, pp. 106–115, Sep. 1975.

[78] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 28–41, Jan. 2004.

[79] A. Lozano and N. Jindal, "Transmit diversity vs. spatial multiplexing in modern MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 9, no. 1, pp. 186–197, Jan. 2010.

[80] A. Lozano and N. Jindal, "Are yesterday's information-theoretic fading models and performance metrics adequate for the analysis of today's wireless systems?" *IEEE Communications Magazine*, vol. 50, no. 11, pp. 210–217, Nov. 2012.

[81] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[82] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge, UK: Cambridge University Press, 2003.

[83] K. V. Mardia, G. Hughes, C. C. Taylor, and H. Singh, "A multivariate von Mises distribution with applications to bioinformatics," *Canadian Journal of Statistics*, vol. 36, no. 1, pp. 99–109, 2008.

[84] J. L. Massey, "Coding and modulation in digital communications," in *Proceedings of the International Zurich Seminar on Digital Communications*, Zurich, Switzerland, Mar. 1974.

[85] R. Matzner, "An SNR estimation algorithm for complex baseband signals using higher order statistics," *Facta Universitatis (Niš)*, vol. 6, no. 1, pp. 41–52, 1993.

[86] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as an instance of Pearl's "belief propagation" algorithm," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 140–152, Feb. 1998.

[87] A. Mehrotra, "Noise analysis of phase-locked loops," *IEEE Transactions on Circuits and Systems—Part I: Fundamental Theory and Applications*, vol. 49, no. 9, pp. 1309–1316, Sep. 2002.

[88] H. Meyr, M. Moeneclaey, and S. A. Fechtel, *Digital Communication Receivers: Synchronization, Channel Estimation, and Signal Processing*. New York, NY, USA: John Wiley and Sons, 1998.

[89] H. Meyr, M. Oerder, and A. Polydoros, "On sampling rate, analog prefiltering, and sufficient statistics for digital receivers," *IEEE Transactions on Communications*, vol. 42, no. 12, pp. 3208–3214, Dec. 1994.

[90] T. P. Minka, "Old and new matrix algebra useful for statistics," Microsoft Research, Tech. Rep., 2000.

[91] T. P. Minka, "Expectation propagation for approximate Bayesian inference," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, Seattle, WA, USA, Aug. 2001.

[92] T. P. Minka, "A family of algorithms for approximate Bayesian inference," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, Jan. 2001.

[93] T. P. Minka, "Pathologies of orthodox statistics," Microsoft Research, Nov. 2001. http://research.microsoft.com/en-us/um/people/minka/papers/pathologies.html

[94] T. P. Minka, "Divergence measures and message passing," Microsoft Research, Tech. Rep. MSR-TR-2005-173, 2005.

[95] T. P. Minka and Y. Qi, "Tree-structured approximations by expectation propagation," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. K. Saul, and B. Schölkopf, Eds. Cambridge, MA, USA: MIT Press, 2004, pp. 193–200.

[96] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 47–60, Nov. 1996.

[97] K. P. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: An empirical study," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, Stockholm, Sweden, Jul. 1999.

[98] F. D. Neeser and J. L. Massey, "Proper complex random processes with applications to information theory," *IEEE Transactions on Information Theory*, vol. 39, no. 4, pp. 1293–1302, Jul. 1993.

[99] N. Noels, C. Herzet, A. Dejonghe, V. Lottici, H. Steendam, M. Moeneclaey, M. Luise, and L. Vandendorpe, "Turbo synchronization: An EM algorithm interpretation," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Anchorage, AK, USA, May 2003.

[100] N. Noels, V. Lottici, A. Dejonghe, H. Steendam, M. Moeneclaey, M. Luise, and L. Vandendorpe, "A theoretical framework for soft-information-based synchronization in iterative (turbo) receivers," *EURASIP Journal on Wireless Communications and Networking*, vol. 2005, no. 2, pp. 117–129, Apr. 2005.

[101] M. Oerder and H. Meyr, "Digital filter and square timing recovery," *IEEE Transactions on Communications*, vol. 36, no. 5, pp. 605–612, May 1988.

[102] E. Ollila, V. Koivunen, and H. V. Poor, "Complex-valued signal processing—essential models, tools and statistics," San Diego, CA, USA, Feb. 2011.

[103] D. R. Pauluzzi and N. C. Beaulieu, "A comparison of SNR estimation techniques for the AWGN channel," *IEEE Transactions on Communications*, vol. 48, no. 10, pp. 1681–1691, Oct. 2000.

[104] J. Pearl, "Fusion, propagation, and structuring in belief networks," *Artificial Intelligence*, vol. 29, pp. 241–288, 1986.

[105] J. Pearl, "Markov and Bayes networks: A comparison of two graphical representations of probabilistic knowledge," Cognitive Systems Laboratory, Tech. Rep. R-46-I, Oct. 1986.

[106] B. Picinbono, "On circularity," *IEEE Transactions on Signal Processing*, vol. 42, no. 12, pp. 3473–3482, Dec. 1994.

[107] B. Picinbono, "Second-order complex random vectors and normal distributions," *IEEE Transactions on Signal Processing*, vol. 44, no. 10, pp. 2637–2640, Oct. 1996.

[108] B. Picinbono and P. Bondon, "Second-order statistics of complex signals," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 411–420, Feb. 1997.

[109] B. Picinbono and P. Chevalier, "Widely linear estimation with complex data," *IEEE Transactions on Signal Processing*, vol. 43, no. 8, pp. 2030–2033, Aug. 1995.

[110] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge, UK: Cambridge University Press, 2008.

[111] E. Riegler, G. E. Kirkelund, C. N. Manchón, M.-A. Badiu, and B. H. Fleury, "Merging belief propagation and the mean field approximation: A free energy approach," *IEEE Transactions on Information Theory*, vol. 59, no. 1, pp. 588–602, Jan. 2013.

[112] M. Samuel, M. Barsoum, and M. P. Fitz, "On the suitability of Gray bit mappings to outer channel codes in iteratively decoded BICM," in *Proceedings of the Asilomar Conference on Signals, Systems and Computers (ACSSC)*, Pacific Grove, CA, USA, Nov. 2009.

[113] L. K. Saul and M. I. Jordan, "Exploiting tractable substructures in intractable networks," in *Advances in Neural Information Processing Systems 8*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds. Cambridge, MA, USA: MIT Press, 1995, pp. 486–492.

[114] L. Schmitt and H. Meyr, "A systematic framework for iterative maximum likelihood receiver design," *IEEE Transactions on Communications*, vol. 58, no. 7, pp. 2035–2045, Jul. 2010.

[115] P. J. Schreier and L. L. Scharf, *Statistical Signal Processing of Complex-Valued Data: The Theory of Improper and Noncircular Signals*. Cambridge, UK: Cambridge University Press, 2010.

[116] M. Senst and G. Ascheid, "Markov Chain Monte Carlo MIMO detection for systems with imperfect channel state information," in *Proceedings of the IEEE Vehicular Technology Conference (VTC)*, Taipei, Taiwan, May 2010.

[117] M. Senst and G. Ascheid, "A combined belief propagation and mean field algorithm for soft carrier phase estimation," in *Proceedings of the International Symposium on Wireless Communication Systems (ISWCS)*, Aachen, Germany, Nov. 2011.

[118] M. Senst and G. Ascheid, "How the framework of expectation propagation yields an iterative IC-LMMSE MIMO receiver," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Houston, TX, USA, Dec. 2011.

[119] M. Senst and G. Ascheid, "A Rao-Blackwellized Markov Chain Monte Carlo algorithm for efficient MIMO detection," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Kyoto, Japan, Jun. 2011.

[120] M. Senst and G. Ascheid, "A message passing approach to iterative Bayesian SNR estimation," in *Proceedings of the International Symposium on Signals, Systems and Electronics (ISSSE)*, Potsdam, Germany, Oct. 2012.

[121] M. Senst, L. Krzymien, L. Szczecinski, and F. Labeau, "Calculating LLRs via saddlepoint approximation in front-end MIMO receivers," *IEEE Transactions on Communications*, vol. 61, no. 6, pp. 2330–2338, Jun. 2013.

[122] M. Senst, H. Lüders, and G. Ascheid, "Performance evaluation of the Markov Chain Monte Carlo MIMO detector based on mutual information," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Cape Town, South Africa, May 2010.

[123] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, Oct. 1948.

[124] V. Simon, A. Senst, M. Speth, and H. Meyr, "Phase noise estimation via adapted interpolation," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, San Antonio, TX, USA, Nov. 2001.

[125] C. Studer, S. Fateh, and D. Seethaler, "ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 7, pp. 1754–1765, Jul. 2011.

[126] L. Szczecinski, "Correction of mismatched L-values in BICM receivers," *IEEE Transactions on Communications*, vol. 60, no. 11, pp. 3198–3208, Nov. 2012.

[127] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, Sep. 1981.

[128] I. E. Telatar, "Capacity of multi-antenna Gaussian channels," *European Transactions on Communications*, vol. 10, no. 6, pp. 585–595, Nov. 1999.

[129] S. ten Brink, "Convergence of iterative decoding," *IET Electronic Letters*, vol. 35, no. 10, pp. 806–808, May 1999.

[130] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Transactions on Communications*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[131] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge, UK: Cambridge University Press, 2005.

[132] G. Ungerböck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. 28, no. 1, pp. 55–67, Jan. 1982.

[133] G. Ungerböck, "Trellis-coded modulation with redundant signal sets—Part I: Introduction," *IEEE Communications Magazine*, vol. 25, no. 2, pp. 5–11, Feb. 1987.

[134] G. Ungerböck, "Trellis-coded modulation with redundant signal sets—Part II: State of the art," *IEEE Communications Magazine*, vol. 25, no. 2, pp. 12–21, Feb. 1987.

[135] G. Ungerböck and I. Csajka, "On improving data-link performance by increasing the channel alphabet and introducing sequence coding," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Ronneby, Sweden, Jun. 1976.

[136] A. van den Bos, "Complex gradient and Hessian," *IEE Proceedings—Vision, Image and Signal Processing*, vol. 141, no. 6, pp. 380–383, Dec. 1994.

[137] A. van den Bos, "The multivariate complex normal distribution—A generalization," *IEEE Transactions on Information Theory*, vol. 41, no. 2, pp. 537–539, Mar. 1995.

[138] A. J. Viterbi and A. M. Viterbi, "Nonlinear estimation of PSK-modulated carrier phase with application to burst digital transmission," *IEEE Transactions on Information Theory*, vol. 29, no. 4, pp. 543–551, Jul. 1983.

[139] A. J. Viterbi, J. K. Wolf, E. Zehavi, and R. Padovani, "A pragmatic approach to trellis-coded modulation," *IEEE Communications Magazine*, vol. 27, no. 7, pp. 11–19, Jul. 1989.

[140] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, Jan. 2008.

[141] L. Wang, D. Xu, and X. Zhang, "Recursive bit metric generation for PSK signals with Gray labeling," *IEEE Communications Letters*, vol. 16, no. 2, pp. 180–182, Feb. 2012.

[142] A. Wiesel, J. Goldberg, and H. Messer, "Non-data-aided signal-to-noise-ratio estimation," in *Proceedings of the IEEE International Conference on Communications (ICC)*, New York, NY, USA, Apr. 2002.

[143] Wikipedia, "Principle of maximum entropy," 2015, [Online; accessed 15-Feb-2015]. http://en.wikipedia.org/wiki/Principle_of_maximum_entropy

[144] J. Winn, "Variational message passing and its applications," Ph.D. dissertation, Cambridge University, 2003.

[145] J. Winn and C. M. Bishop, "Variational message passing," *Journal of Machine Learning Research*, vol. 6, no. 12, pp. 661–694, 12 2005. http://portal.acm.org/citation.cfm?id=1046920.1088695

[146] W. Wirtinger, "Zur formalen Theorie der Funktionen von mehr komplexen Veränderlichen," *Mathematische Annalen*, vol. 97, no. 1, pp. 357–375, 1927.

[147] M. Witzke, S. Bäro, and J. Hagenauer, "Iterative detection of generalized coded MIMO signals using a widely linear detector," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, San Francisco, CA, USA, Dec. 2003.

[148] M. Witzke, S. Bäro, F. Schreckenbach, and J. Hagenauer, "Iterative detection of MIMO signals with linear detectors," in *Proceedings of the Asilomar Conference on Signals, Systems and Computers (ACSSC)*, Pacific Grove, CA, USA, Nov. 2002.

[149] P. W. Wolniansky, G. J. Foschini, G. D. Golden, and R. A. Valenzuela, "V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel," in *Proceedings of the International Symposium on Signals, Systems and Electronics (ISSSE)*, Pisa, Italy, Sep. 1998.

[150] A. Worm, P. Hoeher, and N. Wehn, "Turbo-decoding without SNR estimation," *IEEE Communications Letters*, vol. 4, no. 6, pp. 193–195, Jun. 2000.

[151] A. P. Worthen and W. E. Stark, "Unified design of iterative receivers using factor graphs," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 843–849, Feb. 2001.

[152] N. Wu, H. Wang, and J.-M. Kuang, "Maximum likelihood signal-to-noise ratio estimation for coded linearly modulated signals," *IET Communications*, vol. 4, no. 3, pp. 265–271, Feb. 2010.

[153] Y. Xie and C. N. Georghiades, "Two EM-type channel estimation algorithms for OFDM with transmitter diversity," *IEEE Transactions on Communications*, vol. 51, no. 1, pp. 106–115, Jan. 2003.

[154] J. Ylioinas, J. Karjalainen, M. Juntti, and O. Piirainen, "Scheduling of the activations in iterative detection, decoding, and channel estimation for MIMO-OFDM," *IEEE Transactions on Communications*, vol. 61, no. 2, pp. 638–647, Feb. 2013.

[155] J. Ylioinas, M. R. Raghavendra, and M. Juntti, "Avoiding matrix inversion in DD SAGE channel estimation in MIMO-OFDM with M-QAM," in *Proceedings of the IEEE Vehicular Technology Conference (VTC)*, Anchorage, AK, USA, Sep. 2009.

[156] E. Zehavi, "8-PSK trellis codes for a Rayleigh channel," *IEEE Transactions on Communications*, vol. 40, no. 5, pp. 873–884, May 1992.

[157] D. Zhang, G. Wang, G. Ascheid, and H. Meyr, "Searching for optimal scheduling of MIMO doubly iterative receivers: An ant colony optimization-based method," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, Anaheim, CA, USA, Dec. 2012.

[158] L. Zheng and D. Tse, "Diversity and multiplexing: A fundamental tradeoff in multiple-antenna channels," *IEEE Transactions on Information Theory*, vol. 49, no. 5, pp. 1073–1096, May 2003.

Curriculum Vitae

Name	Martin Senst
Geburtsdatum	14. Mai 1979
Geburtsort	Duisburg
1985–1989	Tannenbergschule, Moers
1989–1998	Gymnasium Adolfinum, Moers
Juni 1998	Abitur
1998–1999	Zivildienst im Alexianer-Krankenhaus, Krefeld
September 1999	Immatrikulation an der RWTH Aachen im Fach Elektrotechnik und Informationstechnik
Oktober 2001	Vordiplom
Mai–Sep. 2004	Praktikum im *Western Australian Telecommunications Research Institute* (WATRI) in Perth, Australien
Feb.–Aug. 2005	Diplomarbeit zum Thema *Detektions- und Synchronisationsalgorithmen für ein UWB-OFDM-System*
August 2005	Diplom (mit Auszeichnung) an der RWTH Aachen
2006–2012	Wissenschaftlicher Angestellter am Lehrstuhl für Integrierte Systeme der Signalverarbeitung an der RWTH Aachen
Seit Juni 2013	Rohde & Schwarz, München