Iterative Error Correction

Turbo, Low-Density Parity-Check and Repeat–Accumulate Codes

Iterative error correction codes have found widespread application in cellular communications, digital broadcasting, deep space communications, and wireless LANs. This self-contained treatment of iterative error correction presents all the key ideas needed to understand, design, implement, and analyze these powerful codes.

Turbo, low-density parity-check, and repeat–accumulate codes are given equal, detailed coverage, with precise presentations of encoding and decoding procedures. Worked examples are integrated into the text to illuminate each new idea and pseudo-code is included for important algorithms to facilitate the reader's development of the techniques described. For each subject, the treatment begins with the simplest case before generalizing. There is also coverage of advanced topics such as density-evolution and EXIT charts for those readers interested in gaining a deeper understanding of the field. This text is ideal for graduate students in electrical engineering and computer science departments, as well as practitioners in the communications industry.

Sarah J. Johnson is a Research Fellow in the School of Electrical Engineering and Computer Science at the University of Newcastle, Australia. She is a member of the IEEE Information Theory and Communications Societies.

Iterative Error Correction

Turbo, Low-Density Parity-Check and Repeat–Accumulate Codes

SARAH J. JOHNSON

University of Newcastle, New South Wales

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo, Delhi

Cambridge University Press
The Edinburgh Building, Cambridge CB2 8RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521871488

First published 2010

Printed in the United Kingdom at the University Press, Cambridge

A catalog record for this publication is available from the British Library

ISBN 978-0-521-87148-8 Hardback

To Ro and Clara

Contents

Preface *page* xi
Notation xv
Commonly used abbreviations xix

1 Channels, codes and capacity 1

1.1 Binary input memoryless channels 2
1.2 Entropy, mutual information and capacity 8
1.2.1 A measure of information 8
1.2.2 Entropy 10
1.2.3 Capacity 13
1.3 Codes and decoding 16
1.3.1 Decoding 18
1.3.2 Performance measures 24
1.4 Bibliographic notes 29
1.5 Exercises 30

2 Low-density parity-check codes 34

2.1 Introduction 34
2.2 Error correction using parity checks 34
2.2.1 Low-density parity-check codes 37
2.3 Encoding 40
2.3.1 (Almost) linear-time encoding for LDPC codes 44
2.3.2 Repeat–accumulate codes 48
2.4 Decoding 48
2.4.1 Message passing on the binary erasure channel 50
2.4.2 Bit-flipping decoding 54
2.4.3 Sum–product decoding 61
2.5 Bibliographic notes 71
2.6 Exercises 72

3 Low-density parity-check codes: properties and constructions 75

3.1 Introduction 75
3.2 LDPC properties 75
3.2.1 Choosing the degree distribution 76
3.2.2 Girth and expansion 82
3.2.3 Codeword and pseudo-codeword weights 84
3.2.4 Cyclic and quasi-cyclic codes 89
3.2.5 Implementation and complexity 95
3.3 LDPC constructions 97
3.3.1 Pseudo-random constructions 97
3.3.2 Structured LDPC codes 105
3.4 LDPC design rules 114
3.5 Bibliographic notes 117
3.6 Exercises 118

4 Convolutional codes 121

4.1 Introduction 121
4.2 Convolutional encoders 121
4.2.1 Memory order and impulse response 125
4.2.2 Recursive convolutional encoders 127
4.2.3 Puncturing 131
4.2.4 Convolutional code trellises 132
4.2.5 Minimum free distance 135
4.3 Decoding convolutional codes 136
4.3.1 Maximum a posteriori (BCJR) decoding 136
4.3.2 Log MAP decoding 148
4.3.3 Maximum likelihood (Viterbi) decoding 156
4.4 Bibliographic notes 160
4.5 Exercises 161

5 Turbo codes 165

5.1 Introduction 165
5.2 Turbo encoders 165
5.3 Iterative turbo decoding 169
5.4 Turbo code design 176
5.4.1 Interleaving gain 176
5.4.2 Choosing the component codes 178
5.4.3 Interleaver design 182
5.4.4 Design rules 188
5.5 Factor graphs and implementation 190
5.5.1 Implementation and complexity 191
5.5.2 Stopping criteria 194

5.6 Bibliographic notes 196
5.7 Exercises 198

6 Serial concatenation and RA codes 201

6.1 Serial concatenation 201
6.1.1 Serially concatenated turbo codes 203
6.1.2 Turbo decoding of SC codes 203
6.1.3 Code design 207
6.2 Repeat–accumulate codes 209
6.2.1 Encoding RA codes 211
6.2.2 Turbo decoding of RA codes 212
6.2.3 Sum–product decoding of RA codes 217
6.2.4 Irregular RA codes 220
6.2.5 Weight-3 IRA codes 224
6.2.6 Accumulate–repeat–accumulate codes 226
6.2.7 Code design 227
6.3 Bibliographic notes 232
6.4 Exercises 234

7 Density evolution and EXIT charts 237

7.1 Introduction 237
7.2 Density evolution 238
7.2.1 Density evolution on the BEC 239
7.2.2 Ensemble thresholds 243
7.2.3 Density evolution and repeat–accumulate codes 247
7.2.4 Density evolution on general binary input memoryless channels 249
7.2.5 Density evolution and turbo codes 254
7.2.6 Designing ensembles with good thresholds 256
7.2.7 Approximations to density evolution 260
7.3 EXIT charts 261
7.3.1 Mutual information 262
7.3.2 EXIT charts for turbo codes 264
7.3.3 EXIT charts for RA codes 273
7.3.4 EXIT charts for LDPC codes 276
7.3.5 Code design and analysis using EXIT charts 279
7.4 Bibliographic notes 279
7.5 Exercises 281

8 Error floor analysis 283

8.1 Introduction 283
8.2 Maximum likelihood analysis 283

8.2.1 Input–output weight enumerating functions for convolutional codes 285
8.2.2 Parallel concatenated code ensembles 290
8.2.3 Serially concatenated code ensembles 296
8.2.4 The error floor of irregular block codes 301
8.2.5 Designing iterative codes to improve the interleaving gain and error floor 305
8.2.6 Asymptotic (in the code length) performance of iterative ensembles 306
8.3 Finite-length analysis 309
8.4 Bibliographic notes 318
8.5 Exercises 320

References 322
Index 331

Preface

The field of error correction coding was launched with Shannon's revolutionary 1948 work showing – quite counter to intuition – that it is possible to transmit digital data with arbitrarily high reliability over noise-corrupted channels, provided that the rate of transmission is below the capacity of the channel. The mechanism for achieving this reliable communication is to encode a digital message with an error correction code prior to transmission and apply a decoding algorithm at the receiver.

Classical block and convolutional error correction codes were described soon afterwards and the first iterative codes were published by Gallager in his 1962 thesis; however they received little attention until the late 1990s. In the meantime, the highly structured algebraic codes introduced by Hamming, Elias, Reed, Muller, Solomon and Golay among others dominated the field. Despite the enormous practical success of these classical codes, their performance fell well short of the theoretically achievable performances set down by Shannon in his seminal 1948 paper. By the late 1980s, despite decades of attempts, researchers were largely resigned to this seemingly insurmountable theory–practice gap.

The relative quiescence of the coding field was utterly transformed by the introduction of "turbo codes", proposed by Berrou, Glavieux and Thitimajshima in 1993, wherein all the key ingredients of successful error correction codes were replaced: turbo codes involve very little algebra, employ iterative, distributed, algorithms and focus on average (rather than worst-case) performance. This was a ground-shifting paper, forcing coding theorists to revise strongly held beliefs. Overnight, the gap to the Shannon capacity limit was all but eliminated, using decoders with manageable complexity.

As researchers worked through the 1990s to understand just why turbo codes worked as well as they did, two researchers, McKay and Neal, introduced a new class of block codes designed to possess many features of the new turbo codes. It was soon recognized that these block codes were in fact a rediscovery of the low-density parity-check (LDPC) codes developed years earlier by Gallager. Generalizations of Gallager's LDPC codes by a number of researchers, including Luby, Mitzenmacher, Shokrollahi, Spielman, Richardson and Urbanke, to irregular LDPC codes actually achieve Shannon's capacity limits on some

channels and approach Shannon's capacity to within hundredths of a decibel on others.

A new class of iterative codes introduced by Divsalar, Jin and McEliece, called repeat–accumulate (RA) codes, combine aspects of both turbo and LDPC codes, arguably containing the best features of each. The combination of simplicity with excellent decoding performance has made RA codes a hot topic for the coding community. Indeed, modified RA codes have been shown to be the first capacity-achieving codes with bounded complexity per message bit for both encoding and decoding.

So rapid has progress been in this area that coding theory today is in many ways unrecognizable from its state less than two decades ago. In addition to the strong theoretical interest in iterative codes, such codes have already been adopted in deep space applications, satellite-based digital video broadcasting, long-haul optical communication standards, wireless local area networking and mobile telephony.

This book presents an introductory treatment of these extremely powerful error-correcting codes and their associated decoding algorithms. The focus of the book is almost exclusively on iterative error correction. Indeed, classical coding topics are only covered where they are necessary for the development of iterative codes, as in Chapters 1 and 4. The umbrella term *iterative decoding*, sometimes referred to as *message-passing decoding*, *probabilistic decoding*, decoding of *codes on graphs*, or simply the *turbo principle*, here encompasses the decoding algorithms associated with turbo codes, low-density parity-check codes and repeat–accumulate codes.

The aim of this book is to provide a gentle introduction to turbo, low-density parity-check and repeat–accumulate codes, with equal coverage given to all three. No prior knowledge of coding theory is required. An emphasis on examples is employed to illuminate each new idea and pseudo-code is presented to facilitate the reader's development of the techniques described. The focus of this book is on ideas and examples rather than theorems and proofs. Where results are mentioned but not proved, references are provided for the reader wishing to delve deeper. Readers interested in a thorough exposition of the theory, however, need go no further than the excellent text by Richardson and Urbanke [1].

None of the material in this text is new; all of it is available in the iterative decoding literature. However, we have decided not to attempt to attribute every idea to the location of its first publication, not least of all because many ideas in the field have been developed independently by a number of authors in different ways. Most importantly, though, we want to keep the focus of the text on explanations of ideas rather than as a historical record. We do use references, however, to point out a few main turning points in the field and to provide direction for the reader wishing to pursue topics we have not fully covered.

Chapters 2 and 3 together can be read as a stand-alone introduction to LDPC codes and Chapters 4 and 5 together provide a stand-alone introduction to turbo codes. Subsequently, however, the distinction is less sharp. The material on RA codes in Chapter 6 draws heavily on the previous material and Chapters 7 and 8 apply to all three code types.

Book outline

Chapter 1 provides a general overview of point-to-point communications systems and the role played by error correction. Firstly we define three channel models, each of which will play an important role in the remainder of the book: the binary symmetric channel (BSC), the binary erasure channel (BEC) and the binary input additive white Gaussian noise channel (BI-AWGNC). From these channel models we introduce the concept of the capacity of a communications channel, a number that precisely captures the minimum redundancy needed in a coded system to permit (essentially) zero errors following decoding. We then turn to error correction and briefly introduce classical error correction techniques and analysis.

Chapters 2 and 3 serve as a self-contained introduction to low-density parity-check codes and their iterative decoding. To begin, block error correction codes are described as a linear combination of parity-check equations and thus defined by their parity-check matrix representation. Iterative decoding algorithms are introduced using a hard decision (bit-flipping) algorithm so that the topic is first developed without reference to probability theory. Subsequently, the sum–product decoding algorithm is presented. Taken alone, Chapter 2 provides a working understanding of the encoding and decoding of LDPC codes without any material on their design and analysis. In Chapter 3 we then discuss the properties of an LDPC code which affect its iterative decoding performance and we present some common construction methods used to produce codes with desired properties.

In Chapters 4 and 5 we turn to turbo codes and their soft, iterative, decoding, which has sparked an intense level of interest in such codes over the past decade. Our starting point is convolutional codes and their trellis representation. This opens the way to the optimal decoding of trellis-based codes using the BCJR algorithm for the computation of maximum a posteriori (MAP) (symbol) probabilities. Having presented the principles of convolutional component codes in Chapter 4, we develop the turbo encoder and decoder in Chapter 5, using the convolutional encoder and MAP decoder as the primary building blocks, and discuss design principles for turbo codes.

In Chapter 6 we consider iterative codes created using serial concatenation. Having so far treated separately the two main classes of iteratively decodable

codes, turbo and LDPC codes, we now consider the class of repeat–accumulate codes, which draws from both and arguably combines the best features of each.

Rather than consider the behaviour of any particular code, our focus in Chapter 7 shifts to *ensembles* of codes, and the *expected* behaviour of an iterative decoding algorithm over this ensemble. Tracking the evolution of the probability density function through successive iterations of the decoding algorithm, a process known as *density evolution* allows us to compute the thresholds of iterative code ensembles and compare their performance with the channel capacity. The chapter concludes by showing how density evolution can be usefully approximated by extrinsic information transfer (EXIT) analysis techniques to understand better the performance of iterative decoding algorithms.

In Chapter 8 we also use the concept of code ensembles but now focus on low-noise channels and consider the error floor performance of iterative codes. Except for the special case of the binary erasure channel, our analysis considers the properties of the codes independently of their respective iterative decoding algorithms. In fact we assume maximum likelihood decoding, for which the performance of a code depends only on its codeword weight distribution.

Acknowledgements

I am very grateful to the many students who have provided useful feedback and suggestions on earlier versions of this text. I would also like to thank Adrian Wills, Vladimir Trajkovic, David Hayes, Björn Rüffer and Tristan Perez for their valuable assistance.

Phil Meyler and Sarah Matthews from Cambridge University Press were very helpful and patient through the many delays. Thanks also to Anna-Marie Lovett, Karen Sawyer and the typesetters from Aptara Inc. for their assistance post production and to Susan Parkinson, who performed wonders with her copy-editing in a very tight time-frame.

I am particularly grateful to Steven Weller. As well as being responsible for starting me on this endeavor he has provided invaluable help and guidance along the way.

Most of all I owe more than I can say to my family, from whom much of the time to write this book was taken.

Notation

The following is a list of notation used across more than one chapter.

$\alpha_t(S_i)$	Forward metric of the BCJR decoder at time t for state S_i
a	Combiner parameter of an RA code
a_d	Number of codewords of weight d
$a_{w,d}$	Number of messages of weight w that produce codewords of weight d
$\mathbf{A}$	Vector of a priori LLRs
$\mathbf{A}_l^{(j)}$	Vector of a priori LLRs into the jth decoder at the lth iteration of turbo decoding
A_w	Number of messages of weight w
$A_{w,p}$	Number of messages of weight w that produce parity bits of weight p
$A(D)$	Weight enumerating function (of a_d's)
$A(W, D)$	Input–output weight enumerating function (of $a_{w,d}$'s)
$A(W, P)$	Input–redundancy weight enumerating function (of $A_{w,p}$'s)
$\mathcal{A}(S_i)$	LLR of $\alpha_t(S_i)$
$\beta_t(S_i)$	Backward metric of the BCJR decoder at time t for state S_i
$\mathcal{B}(S_i)$	LLR of $\beta_t(S_i)$
$\mathbf{c}$	Binary vector containing the codeword bits
$c_t^{(i)}$	Value of the codeword bit output at time t from the ith output of a convolutional encoder
$c(x)$	Capacity of a channel with parameter x
C	An error correction code
d	Hamming weight of a binary codeword
$\mathbf{d}$	Binary vector containing the interleaved message bits of an RA code
$d_{\min}$	Minimum distance of an error correction code

d_{f}	Free distance of a convolutional code
d_{ef}	Effective free distance of a convolutional or concatenated code
$d_{\mathrm{f}}^{(\mathrm{I})}$	Free distance of the inner code in a serially concatenated code system
$d_{\mathrm{f}}^{(\mathrm{O})}$	Free distance of the outer code in a serially concatenated code system
ε	Erasure probability of the binary erasure channel
ϵ	Crossover probability of the binary symmetric channel
$\mathbf{E}$	Vector of extrinsic LLRs
$E_{j,i}$	Extrinsic message from the jth check node to the ith bit node
$\mathbf{E}_l^{(j)}$	Vector of extrinsic LLRs from the jth decoder at the lth iteration of turbo decoding
E_b/N_0	Signal-to-noise ratio of an additive white Gaussian noise channel
$\gamma_t(S_r, S_s)$	State transition metric into the BCJR decoder for states S_r to S_s at time t
$\Gamma_t(S_r, S_s)$	LLR of $\gamma_t(S_r, S_s)$
G	Code generator matrix
$\mathbf{h}$	Row (check node) degree distribution of an irregular LDPC parity-check matrix
h_i	Fraction of Tanner graph check nodes that are degree i
H	Code parity-check matrix
$I(X;Y)$	The mutual information between X and Y
I_x	Identity matrix of size x
$I_{\max}$	Maximum number of decoder iterations
k	Number of input bits to a convolutional encoder at each time instant
k_{O}	k for the outer code in a serially concatenated code system
k_{I}	k for the inner code in a serially concatenated code system
k_i	k for the ith code in a parallel concatenated code system
K	Code dimension (length of the messages)
K_{O}	K for the outer code in a serially concatenated code system
K_{I}	K for the inner code in a serially concatenated code system
K_i	K for the ith code in a parallel concatenated code system
$\lambda(x)$	The (edge perspective) bit node degree distribution an irregular LDPC code
λ_i	Fraction of Tanner graph edges connected to a degree-i bit node
l	Iteration number
$\mathbf{L}$	Vector containing the LLRs of the decoded bits
$L(x)$	Log likelihood ratio of x
log	Logarithm to the base e

$\log_2$	Logarithm to the base 2
m	Number of parity-check equations in a block code
$m_t(S_r, S_s)$	State transition metric calculated by the BCJR decoder for states S_r to S_s at time t
M_i	Message from the ith bit node
$M_{j,i}$	Message from the ith bit node to the jth check node
n	Number of output bits from a convolutional encoder at each time instant
n_{O}	n for the outer code in a serially concatenated code system
n_{I}	n for the inner code in a serially concatenated code system
n_i	n for the ith code in a parallel concatenated code system
N	Code length (length of the codewords)
N_O	N for the outer code in a serially concatenated code system
N_I	N for the inner code in a serially concatenated code system
N_i	N for the ith code in a parallel concatenated code system
$\oplus$	Modulo-2 addition
Π	Permutation sequence
π_i	ith element of permutation sequence
$\mathbf{p}$	Binary vector containing the parity bits
p	Weight of the parity bits in a binary codeword
$p(x)$	Probability of x
P	Matrix giving a puncturing pattern
q	Repetition parameter of an RA code
$Q(x)$	The error function
$\rho(x)$	The (edge perspective) check node degree distribution of an irregular LDPC code
ρ_i	Fraction of Tanner graph edges connected to a degree-i check node
r	Code rate
$\mathbf{R}$	Vector containing the LLRs of the received bits
σ^2	Variance of the additive white Gaussian noise
$s_t^{(i)}$	Value of the ith shift register element at time t in a convolutional encoder
$\mathbf{s}$	Binary vector containing the bits at the output of the RA code combiner
$\mathbf{s}$	Syndrome of a parity-check code
S	Convolutional encoder state
t	Time elapsed
T	Number of trellis segments
$u_t^{(i)}$	Value of the message bit input at time t to the ith input of the convolutional encoder

$\mathbf{u}$	Binary vector containing the message bits
$\hat{\mathbf{u}}$	Binary vector containing the hard decisions on the decoded message bits
v	Memory order of a convolutional encoder
v_i	Fraction of Tanner graph bit nodes which are degree i
$\mathbf{v}$	Binary vector containing the repeated message bits of an RA code
$\mathbf{v}$	Column (bit node) degree distribution of an irregular LDPC parity-check matrix
w	Weight of a binary message
w_c	Column weight (bit node degree) of a regular LDPC parity-check matrix
w_r	Row weight (check node degree) of a regular LDPC parity-check matrix
$\mathbf{x}$	Vector containing the transmitted bits $\in \{+1, -1\}$
$\mathbf{y}$	Vector containing the values received from the channel
$\mathbf{z}$	Vector containing the additive white Gaussian noise

Commonly used abbreviations

APP	a posteriori probability
ARA	accumulate–repeat–accumulate
BCJR	Bahl, Cock, Jelenik and Raviv (algorithm)
BER	bit error rate
BEC	binary erasure channel
DE	density evolution
EXIT	extrinsic information transfer
BI-AWGN	binary input additive white Gaussian noise
BSC	binary symmetric channel
FIR	finite impulse response
FL	finite length
IIR	infinite impulse response
IOWEF	input–output weight enumerating function
IRA	irregular repeat–accumulate
IRWEF	input–redundancy weight enumerating function
LDPC	low-density parity-check
LLR	log-likelihood ratio
MAP	maximum a posteriori probability
ML	maximum likelihood
RA	repeat–accumulate
SC	serially concatenated
SNR	signal-to-noise ratio
WER	word error rate
WEF	weight enumerating function