AN INTRODUCTION TO ERROR CORRECTING CODES WITH APPLICATIONS

by

Scott A. Vanstone

and

Paul C. van Oorschot

AN INTRODUCTION TO ERROR CORRECTING CODES WITH APPLICATIONS

3

by

Scott A. Vanstone
University of Waterloo

KLUWER ACADEMIC PUBLISHERS
Boston/Dordrecht/London

Distributors for North America:
Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, Massachusetts 02061 USA

Distributors for all other countries:
Kluwer Academic Publishers Group
Distribution Centre
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS

Library of Congress Cataloging-in-Publication Data

Vanstone, Scott A.
An introduction to error correcting codes with applications / by Scott A. Vanstone and Paul C. van Oorschot.
p. cm. — (The Kluwer international series in engineering and computer science. Communications and information theory)
Bibliography: p.
Includes index.
ISBN 0-7923-9017-2
1. Error-correcting codes (Information theory) 2. Data transmission systems. I. Van Oorschot, Paul C. II. Title.
III. Series.
TK5102.5.V32 1989
005.7'2—dc20 89-31032
CIP

Symbols and Notation

$[n,M]$-code	5	block code of block length n with M codewords
$\lfloor a \rfloor$	12	largest integer smaller than or equal to a
$\lvert X \rvert$	19	cardinality of set X
Z_n	22	integers modulo n
$a^{-1}\, mod\, n$	22	multiplicative interse of a, modulo n; $a \cdot a^{-1} \equiv 1 (mod\, n)$
$Z_p[x]$	24	polynomials in indetermine x, with coefficients from Z_p
$[g(x)]$	25,149	equivalence class containing polynomial $g(x)$
$Z_p[x]/(f(x))$	25,27,149	et of all equivalence cla:ses in $Z_p[x]$ under congruence modulo the polynomial $f(x)$; sometimes denoted $Z_p[x]/f(x)$
$GF(q)$	28	Galois field (finite field) with q elements; q is a prime or a power of a prime number
$ord(\alpha)$	30	order of a field element
F^*	33	$F \setminus \{0\}$, the set of non-zero elements of F
$m_\alpha(x)$	33	minimal polynomial of α; sometimes $m_i(x)$ is used to denote $m_{\alpha^i}(x)$ or $m_{\beta^i}(x)$ (with α or β implied by the context)
$(a_0a_1a_2)$	35	vector re.resentation of polynomial $a_0+a_1x+a_2x^2$
$z(i)$	39,277	Zech's logarithm of α^i; $\alpha^{z(i)}=1+\alpha^i$
α^∞	39	$\alpha^\infty=0$, by notational convention
$V_k(F)$	45	set of k-tuples over F
(n,k)-code	48	linear code of block length n and dimension k
(n,k,d)-code	48	linear code of block length n and dimension k, with distance d
$C^\perp$	54	orthogonal element of linear code C ("C perp")
$\det D$	64	determinant of matrix D
$e^{2\pi i/p}$	89	$cos(2\pi/p)+i\, sin(2\pi/p)$, where $i^2=-1$
$R(1,r)$	115	first-order Reed-Muller code of block length 2^r
$R(k,r)$	144	kth-order Reed-Muller code of block length 2^r
$h_R(x)$	163	reciprocal polynomial of $h(x)$
C_i	182	cyclotomic cost (of q modulo n) containing i
C	182	set of cyclotomic costs (of q modulo n)
gcd	187,269	greatest common divisor

lcm	207	least common multiple
$\sigma(z)$	216	error locator polynomial for BCH decoding
$w(z)$	216	error evaluator polynomial for BCH decoding
$S(z)$	217	syndrome polynomial for BCH decoding
(n,k)–*RS*	237	(n,k) Reed-Solomon code
$\hat{\delta}(z)$	244	error-erasure locator polynomial for BCH decoding
$\lambda(z)$	245	erasure locator polynomial for BCH decoding

Foreword

The field of Error Correcting Codes had its roots in Shannon's development of Information Theory in the 1940's. Research on coding was very active in the 50's and 60's and most of the major results known today were developed in that period. Most "practical engineers" of that era regarded coding as useless and impractical, while many of the researchers despaired of the field every becoming more than an academic playground.

Applications for error correcting codes started in the 70's in very expensive communications systems such as deep space probes. As hardware costs dropped and as development engineers saw the benefits of real coding systems, applications expanded quickly. Today very sophisticated coding is used in ordinary consumer compact disc players, in storage technology, and in a broad range of communication systems.

This text develops the topic of error correcting block codes at an undergraduate level. It manages to do this both by being rather selective in the set of topics covered and by having an unusually clear organization and style of presentation. There are some sophistcated mathematical topics in the text and the authors develop them honestly, but the arguments are made accessible by using insight rather than excessive formalism.

One might wonder about the merits of an undergraduate course on error correcting block codes given the intense pressure to include more and more material in the curriculum for undergraduates in engineering, computer science, and mathematics. Wouldn't it be preferable to have a course covering convolutional codes as well as block coding and also covering many other communication topics? The type of course represented by this text, however, has many important advantages over a survey type course, not least of which is that survey courses are invariably boring. The major advantage of studying error correcting codes is the beauty of this particular combination of mathematics and engineering. The student, even if never involved with the field later, gets an example of applied mathematics and engineering at their best.

Robert G. Gallager, Consulting Editor
Fujitsu Professor of Electrical Engineering
Massachusetts Institute of Technology

Preface

An Introduction to Error Correcting Codes with Applications is a text intended for an introductory undergraduate course in error-correcting codes. It should be of interest to a wide range of students, from mathematics majors to those in computer science and engineering. The book deals almost entirely with codes that are linear in nature, and the necessary algebraic tools are developed, where necessary, to make the presentation self-contained. It is intended for students who are familiar with the fundamental concepts of linear algebra (solution of systems of linear equations, vector spaces, etc.). Background in abstract algebra (groups, rings and fields) would be of use, but is not assumed. In general, interested students with reasonable mathematical maturity will find the material to be at a suitable level.

While the underlying mathematics is completely developed, the book emphasizes the practical considerations of efficient encoding and decoding schemes for various codes. Most books in the area either emphasize the theoretical aspects of the subject (to the exclusion of practical considerations), or go to the other extreme, presenting implementation details at the circuit level. It is the purpose of this book to fill the void between the two, considering codes of practical interest, but stopping short of hardware-level implementation details. Codes which cannot be efficiently implemented are given little emphasis, and the temptation to discuss non-practical codes with "cute" mathematical properties has been resisted.

The book covers the basic principles involved in error-coding, and presents a thorough introduction to linear codes. It introduces cyclic codes in general, and progresses to the more sophisticated BCH codes, and the very practical Reed-Solomon codes. Applications of error-correcting codes to deep-space probe communications and to the increasingly popular *compact disc* players are emphasized. The goal is to provide a basic understanding of practical error-correcting codes and the principles they rely on. The development of an appreciation for the elegant mathematical concepts making efficient implementations possible is a most welcome byproduct.

The majority of the material presented is widely available in standard references, in various forms. Many of these we have included in our list of references; a most extensive bibliography can be found in [MacWilliams 77], although this of course does not contain the more recent results. Some results which we have included are less widely available, and are perhaps worth singling out. These include the results by van Lint and Wilson on bounds for cyclic codes (§6.3); the material dealing with coding techniques used in commercial compact disc players (parts of Chapter 7); and some factoring techniques (§5.9, §6.5) due to Berlekamp, which can be found in [Berlekamp 68] but are not typically included in other coding theory books.

Because of the importance of finite fields to a serious introduction to algebraic codes, the text provides a basic but thorough introduction to finite fields, assuming no prior knowledge of field theory. With the exception of a few sections which may be omitted at the instructor's discretion, the material can be covered in a one-semester course (thirteen weeks of lectures). It has been taught at the University of Waterloo for the past eight years to junior and senior undergraduate mathematics, computer science and engineering students. To aid instructors, we have denoted with a dagger (†) those sections which we feel may be omitted safely, if time is short. Inclusion of these sections, together with some of the theory developed in the exercise sets (eg. further bounds for linear codes, higher-order Reed-Muller codes, shortened Reed-Solomon codes), makes the text suitable for an introductory course at the graduate level, or for advanced undergraduates.

We have made a conscious effort to limit the amount of material in this book to that which one can reasonably expect to grasp in an introductory course. Unfortunately, this has meant that some areas meriting discussion have been left out. Perhaps the most notable of these is an introduction to convolutional codes, for which we refer the reader to [Blahut 83] or [Lin 83].

The text contains over 300 exercises, which range from routine to very challenging. We feel this is an essential component of an introductory course. A separate solution manual giving complete and detailed solutions to all exercises is currently being written.

Acknowledgements

The authors would like to thank the many people who took the time to constructively criticize this book in its various stages of development, point out simpler proofs to several standard results, and provide us with further sources of information. Among these people are Robert Day, Dieter Jungnickel, Therese Lee, Stan Lipshitz, Peter Manson, Bill McCuaig, Haluk Oral, Alfred Menezes, Ken Rosen, Doug Stinson, and Dan Wevrick.

We would also like to thank the editorial board of Kluwer Academic Publishers for many helpful suggestions. In particular, we gratefully acknowledge comments by R. Gallagher and R. McEliece. We would be remiss not to acknowledge the encouragement and support of this project by R. Holland, the publisher.

Finally, without the skillful typesetting of Christy Gillin, Joan Selwood and Marg Ward, the book would not exist. We thank them for all of their help and give special thanks to Joan Selwood for a great deal of extra effort during the final hectic days of preparation.

Table of Contents

† This section may be omitted without loss of continuity.

Chapter 1

INTRODUCTION and FUNDAMENTALS

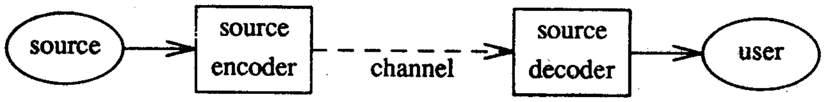
1.1 An Introduction to Coding Theory

The *theory of error detecting and correcting codes* is that branch of engineering and mathematics which deals with the reliable transmission and storage of data. Information media are not 100% reliable in practice, in the sense that *noise* (any form of interference) frequently causes data to be distorted. To deal with this undesirable but inevitable situation, some form of *redundancy* is incorporated in the original data. With this redundancy, even if errors are introduced (up to some tolerance level), the original information can be recovered, or at least the presence of errors can be detected. A small example serves to illustrate the concepts.

Suppose the information we are to transmit comes from the set of symbols $\{A,B,C,D\}$. For practical considerations we associate sequences of 0's and 1's with each of these symbols.

$$\begin{aligned} A &\rightarrow 00\\ B &\rightarrow 10\\ C &\rightarrow 01\\ D &\rightarrow 11 \end{aligned}$$

Hence, if we send a 00 sequence, the receiver is to interpret this as the message A. The proce can be represented by the following simple schematic diagram.

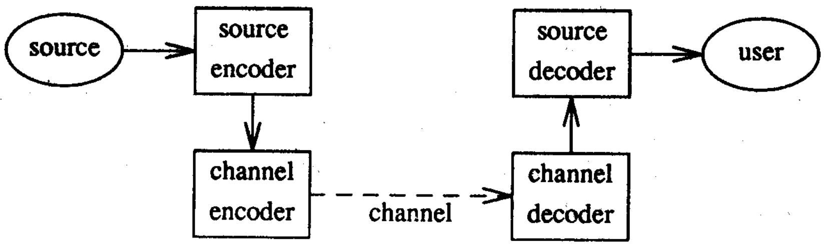


The source emits symbols from the information set which in our example is $\{A,B,C,D\}$. The *source encoder* associates each piece of information with a binary (0,1) sequence and then transmits it. The *channel* may be any information medium. For example, it may be a radio wave channel, a microwave channel, a cable, a digital integrated circuit or a storage disk. The *source decoder* receives the binary sequences from the channel, converts them back to the source alphabet and then passes this data to the user.

If the channel were 100% reliable, that is, if whatever the source encoder put on the channel was precisely what the source decoder received, then there would be no need for *error correction*. Unfortunately, most channels are not entirely reliable and there is a certain probability that what the decoder receives is not what was sent. If the source encoder sends 00 and the source decoder receives 01, then a single error has occurred. The decoder has no way of knowing this since 01 is a valid piece of information. The decoder has no choice but to pass the symbol C to the user. The challenge is to improve the reliability of message transmission. We do this by adding redundancy to each message. A major problem in coding theory is to determine how to add this redundancy to messages in order to detect and possibly correct channel errors. In our example, we might make the following associations.

$$\begin{array}{ccccc} A & \rightarrow & 00 & \rightarrow & 00000 \\ B & \rightarrow & 10 & \rightarrow & 10110 \\ C & \rightarrow & 01 & \rightarrow & 01011 \\ D & \rightarrow & 11 & \rightarrow & 11101 \end{array}$$

Now, if the receiver reads 01000 it knows that an error has occurred, since this sequence is not one which the encoder put on the channel. If we assume that errors are introduced randomly to the binary digits (*bits*) of the sequence, then it seems reasonable for the decoder to assume that the transmitted sequence was 00000, since the received sequence can be obtained from this by the introduction of a single error. At least two errors would have to occur to transform any one of the other three sequences to 01000. One can easily check that altering a single bit in any one of the above 5-bit sequences results in a unique sequence. Hence if (at most) a single bit is altered, the resulting sequence can be uniquely identified with one of the original sequences. Thus our simple example has the capability to correct a single error. The following diagram schematically illustrates the new coding process.



One of the problems to be resolved then is to determine how the channel encoder should add redundancy to the source encoder output. Another problem is to determine how the channel decoder should decide which sequence to decode to. We shall address these problems and others in later chapters. In §1.2, we indicate some specific applications of error-correcting codes, and in §1.3 we formalize the concepts just introduced.

The study of error-correcting codes is one branch of *coding theory*, a more general field of science dealing with the representation of data, including *data compression* and *cryptography*. These three areas are related in that they involve the transformation of data from one representation to an alternate representation and back again via appropriate *encoding* and *decoding* rules.

The objective of data compression is to transform data into a representation which is more compact yet maintains the information content (meaning) of the original data, to allow for more *efficient* use of storage and transmission media. This is possible by removing redundancy from the data. The traditional goal of cryptography has been to ensure *privacy* in communication by transforming data to render it unintelligible to all but the intended recipient. This is achieved through the use of an encoding scheme that relies on a *secret key* known only by the sender and intended recipient. As discussed above, the purpose of error-correcting codes is to increase the *reliability* of communication despite noise in the data medium, by adding redundancy in a uniform and efficient manner.

It is interesting to note that whereas cryptography strives to render data unintelligible to all but the intended recipient, error-correcting codes attempt to ensure data is decodable despite any disruption introduced by the medium. Data compression and error correction also contrast one another in that the former involves compaction and the latter data expansion. In this book we will be concerned mainly with error detecting and correcting codes.

If data compression and cryptographic encoding were to be used in conjunction with error-correcting codes, then the coding process as outlined in the above diagram would be modified slightly. A data compression stage might precede or replace the source encoding stage, and cryptographic encoding would be best performed just prior to channel encoding. Corresponding decoding stages would then be necessary at the appropriate points at the decoding end. Note that the removal of redundancy by data compression and subsequent addition of redundancy during channel encoding are not contradictory, as the redundancy that is removed by data compression serves no useful purpose, whereas that added by the channel encoder is of use for error control.

1.2 Applications of Error Correcting Codes

The increasing reliance on digital communication and the emergence of the digital computer as an essential tool in a technological society have placed error-correcting codes in a most prominent position. We cite here a few specific applications, in an attempt to indicate their practicality and importance.

The use of a *parity-bit* as an error-detecting mechanism is one of the simplest and most well-known schemes used in association with computers and computer communication. Data is partitioned into blocks of n bits, and then to each block, an additional bit is appended as a 0 or a 1, so as to make the number of bits which are 1 in the block including the appended bit, an even number (for *even parity*). If during transmission then, a single bit-error occurs within the block, the parity of the block (i.e. the number of 1's in it) becomes odd, and checking the parity thus allows for detection of single errors.

Many computers now have error-correcting capabilities built into their random access memories; it is less expensive to compensate for errors through the use of error-correcting codes than to build integrated circuits that are 100% reliable. The single error-correcting *Hamming codes,* and *linear codes* in general, are of use here. These will be considered in Chapter 3. Disk storage is another area of computing where error-coding is employed. Storage capacity has been greatly increased through the use of disks of higher and higher density. With this increase in density, error probability also increases, and therefore information is now stored on many disks using error-correcting codes.

In 1972, the *Mariner* space probe flew past Mars and transmitted pictures back to earth. The channel for such transmissions is space and the earth's atmosphere. Solar activity and atmospheric conditions can introduce errors into weak signals coming from the spacecraft. In order that most of the pictures sent could be correctly recovered here on earth, the following coding scheme was used. The source alphabet consisted of 64 shades of grey. The source encoder encoded each of these into binary 6-tuples and the channel encoder produced binary 32-tuples. The source decoder could correct up to 7 errors in any 32-tuple. We shall discuss this system in more detail when we consider *Reed-Muller codes* in Chapter 4.

In 1979, the *Voyager* probes began transmitting colour pictures of Jupiter. For colour pictures the source alphabet needed to be much larger and was chosen to have 4096 colour shades. The source encoder produced binary 12-tuples for each colour shade and the channel encoder produced 24-tuples. This code would correct up to 3 errors in any 24-tuple, and is the *Golay code* discussed in Chapter 4.

The increasing popularity of *digital audio* is due in part to the powerful error-correcting codes that the digitization process facilitates. Information is typically stored on a small aluminized disk as a series of microscopic pits and smooth areas, the pattern representing a sequence of 0's and 1's. A laser beam is used as the playback mechanism to retrieve stored data. Because the data is digital, error correction schemes can be easily incorporated into such a system. Given an error-coded digital recording, a digital audio system can on playback, correct errors introduced by fingerprints, scratches, or even imperfections originally present in the storage medium. The *compact disc* system pioneered by Philips Corporation of the Netherlands in cooperation with Sony Corporation of Japan, allowing playback of pre-recorded digital audio disks, is an excellent example of the application of error-correcting codes to digital communication; *cross-interleaved Reed-Solomon* codes are used for error correction in this system. These ideas are pursued in Chapter 7. *Digital audio tape* (DAT) systems have also been developed, allowing digital recording as well as playback.

1.3 Fundamental Concepts

In this book we shall be primarily interested in *block codes*. We begin with a few definitions, in order to develop a working vocabulary. Let A be an *alphabet* of q symbols. For example, $A = \{a,b,c,...,z\}$ is the standard lower case alphabet for the English language, and $A = \{0,1\}$ is the *binary alphabet* used in the example of §1.1.

Definition. A *block code* of length n containing M *codewords* over the alphabet A is a set of M n-tuples where each n-tuple takes its components from A. We refer to such a block code as an $[n,M]$-code over A.

In practice, we most frequently take A to be the binary alphabet. We often refer simply to an $[n,M]$-code, the alphabet A being understood. We reserve round brackets ( ) for special types of codes, the *linear codes*, to be introduced in Chapter 3. Given a code C of block length n over an alphabet A, those specific n-tuples over A which are in C are referred to as *codewords*. Note that while the channel encoder transmits codewords, the n-tuples received by the channel decoder may or may not be codewords, due to the possible occurrence of errors during transmission. We use the terms *vector* and *word* interchangeably for n-tuple.

In §1.1, we constructed a [5,4]-code over a binary alphabet. That is, we constructed a code with 4 codewords, each being a 5-tuple (block length 5), with each component of the 5-tuple being 0 or 1.