Coding Basics

and multiple users

2 Coding Basics

2.1 Increasing Dimensionality

2.1.1 Dimensionality-Increasing Examples

2.1.2 Formal Code definition and Redundancy

2.1.3 Asymptotics

2.1.3.1 Sphere Packing

2.1.4 Interleaving

2.2 Coded systems, performance, and decoding

2.2.1 Soft-Decoded Systems

2.2.1.1 Lattice and Block Codes

2.2.2 Hard-Decoded Systems

2.2.2.1 Binary Block Codes

2.2.2.2 Block Codes for the DMC

2.2.3 Erasure Decoding

2.2.4 Trellis and Convolutional Codes

2.2.4.1 Binary Convolutional Codes

2.3 Mutual Information and Channel Capacity

2.3.1 Entropy and Data Symbols' Distribution

2.3.2 Joint and Conditional Entropy, and Mutual Information

2.3.3 Asymptotic Equipartition

2.3.4 The Channel Capacity Theorem

2.3.4.1 Uniform Continuous Subsymbols relative to Gaussian

2.3.5 Simple Water-filling for the matrix AWGN

2.3.6 Decoding with Good-Code Gaussian inputs

2.4 The gap, and some simple channels and codes

2.4.1 The gap

2.4.2 Mutual Information and Capacity for DMCs

2.4.3 Capacity, Coding, and the Gap

2.4.4 Energy per bit and low-rate coding

2.4.5 Some Simple Design Examples with Coding

2.4.5.1 Designing with Reed Solomon/Cyclic Codes

2.4.5.2 Design with Simple Convolutional Code

2.4.5.3 Low Density Parity Check (LDPC) Codes

2.5 Parallel Channels and the Multi-tone AWGN Channel

2.5.1 Capacity Conversion for Memoryless Channels

2.5.2 Waveform Channels with Limited Bandwidth or Memory

2.5.3 Capacity of the infinite bandwidth channel

2.5.4 Example of Water-Filling Capacity Calculation

2.6 Multiuser Coding Basics
Chapter 2

Coding Basics

Coding improves data transmission through use of larger symbol dimensionality, which means larger $N$, $L_x$, and/or $L_y$ than in Chapter 1. On the AWGN, such coding can be viewed simply as “densely packing more evenly spaced points” into a given per-dimensional volume ($V_{x/N}$) and energy ($\bar{\mathcal{E}}_x$) as dimensionality increases. For any channel $p_y/x$, the AWGN’s packing generalizes to mutually exclusive input-message encoding/mapping to a certain asymptotic equal/uniform channel-distribution partitioning. Codes often use the same, or closely related, subsymbol constellation(s) repeatedly to form an $N$-dimensional codeword comprised of $\tilde{N}$-dimensional subsymbols. The codeword is a multi-dimensional symbol. The set of all such possible codewords (symbols) is the code. This text’s approach will find a certain minimum redundancy (a generalization of parity bits) will be necessary on subsymbol constellations with good codes. This chapter’s focus is both data-rate bounding and basic-design outlines, which will be used in the system designs of Chapters 3 - Chapter 7. Chapters 8 - 11 will more completely detail specific code designs and will also enumerate many popular codes designs, their parametrization, and their decoders. Chapter 8 focuses on sequential- and iterative-decoding methods. More complete code listings appear later in Chapters 9 (block and lattice codes), 10 (convolutional and trellis codes), and 11 (concatenated codes).

Section 2.1 provides some coding examples and basic encoder-generator and decoder definitions. These example codes are for the AWGN and for discrete memoryless channels. Section 2.1 introduces sphere-packing to help describe good codes, and to help build system-design insight. Section 2.1 also formally defines code redundancy for a code’s subsymbols. The subsymbols will be chosen from a constellation $C$ that may have extra subsymbol-value possibilities with possibly non-uniform subsymbol allowed-value distributions, even though there will be $M = 2^b$ equally likely codewords in the larger-dimensionality code. The extra subsymbol values characterize a code’s redundancy. Section 2.2 progresses to performance characterization and decoder basics, along with some basic concepts in code generators and parity and their relationship to redundancy. Section 2.2 also extends the sphere-packing insights to finite fields and “ball packing” concept in finite-field geometry. As well, Section 2.2 revisits and expands Chapter 1’s performance measures to coding. Section 2.3 introduces entropy that characterizes any probability distribution, but can provide useful interpretation of a code’s subsymbol-constellation values, and of the entire code’s possible codewords, when either are viewed as random vectors. Section 2.3 also introduces the channel’s mutual information. This mutual information’s maximum over possible input distributions is the channel capacity, which is the maximum possible bits/symbol that can be reliably transmitted. Section 2.3 will thereby introduce Shannon’s famous capacity result as a data-rate bound on best codes’ performance as dimensionality becomes infinite. AWGN channels’ minimum mean-square error estimation also relates closely to entropy and mutual information. Section 2.3’s Asymptotic Equal Partition (AEP) is the generalization of the Gaussian channel’s MMSE concept, allowing Gaussian channel insights to expand more generally. Section 2.3 also expands on Section 1.5’s vector coding parallel channels to introduce a best variable water-filling energy/dimension strategy. While this bound applies to a general channel model $p_y/x$, this chapter will mainly focus the capacity formula to the AWGN and binary symmetric channels. Section 2.4 will revisit Chapter 1’s gap concept in the context of measuring the strength or goodness of AWGN-channel codes relative to Shannon’s
capacity, as well as revisit a few simple DMCs’ capacities. Some simple design examples using coding
concepts then also appear in Section 2.4. Section 2.5 progresses to some simple codes as illustrations for
both a single AWGN and for coordinated coding of an otherwise independent set of such AWGNs that
is useful on filtered AWGN’s, expanding vector coding to its multi-tone limit.

Many channels have multiple users (multiple message senders) who share the channel resources (or
dimensions). This chapter’s later sections expand upon earlier sections’ concepts to introduce multiple-
user communication and coding basics. Section 2.6 introduces the 3 basic types of multi-user channels:
the multiple-access channel (MAC), the broadcast channel (BC), and the interference channel
(IC) that have different levels and locations of encoding or decoding collaboration between users. Section
2.6 also expands single-user capacity to a capacity region. Multi-user detectors also first appear in
Section 2.6, which leads to prioritization, or user order, and the related concepts of reliably decodable
users and not reliably decodable other users in capacity-region construction. Section 2.6 also uses these
concepts to describe the general multi-user capacity region that may involve various intersection and/or
union (convex hull) operations over a variety of prioritization orders and/or input probability density
possibilities.

Section 2.7 addresses the MAC, with particular emphasis on the Gaussian MAC and its capacity
region, for both a set of individual-user energy constraints and for a single total energy-sum constraint.
In the latter energy-sum-constraint case, the Gaussian MAC will have at least one primary user and
possibly secondary users and/or other primary users that are helpful to general Gaussian capacity-
region construction’s explanation. Successive-decoding methods appear and assist understanding and
implementation. Section 2.7 also generalizes Section 2.5’s multi-tone continuous-time channel concepts to
the MAC. The multi-user simultaneous-water-filling energy allocation also appears to characterize
the MAC maximum rate sum. Section 2.8 progresses to the Gaussian BC, essentially finding a
dual approach to the MAC where again primary and secondary users appear. The concept of worst-
case noise also appears to simplify the BC capacity-region construction. Section 2.8 introduces scalar
duality, while deferring more complex vector duality to Chapter 5. Section 2.8 also provides a saddle-
point converging algorithm of iterative steps of worst-case-noise determination and water-filling energy
allocation that provides the BC’s maximum rate sum. Section 2.8 then generalizes to the non-
Gaussian BC case briefly, again following Section 2.3’s AEP concepts effective generalization of MMSE
theory.

Section 2.9 proceeds to the Gaussian IC, finding its capacity region through simplifications using the
MAC-set concept or equivalently a dual BC set. Section 2.10 generalizes to stage-less channels that are
combinations of MAC, BC, IC, and/or single-user channels and strategies for simplifying capacity-region
calculation for the Gaussian case. Section 2.10 finds similarly the capacity region for the related multi-
stage relay channel through the an intersection use of Section 2.9’s MAC-set and BC-set concepts
for the most common single-stage relay channel, then adding an IC-set concept when there are multiple
relay stages.

An experienced reader may note that some multi-user material within would be novel and not yet
found elsewhere. Numerous examples appear that validate the concepts, but the author acknowledges
that the general communication/information-theory expert community may find issue (or mistakes) with
any new concepts, and certainly those inputs are welcome and encouraged on any oversights made.
2.1 Increasing Dimensionality

This section begins with Subsection 2.1.1’s simple code examples that increase symbol dimensionality (codeword length) to motivate coding theory. Subsection 2.1.2 then progresses to formal definitions. Subsection 2.1.2 also introduces and defines redundancy. Subsection 2.1.3 introduces the asymptotic analysis concept of the law of large numbers and proceeds to sphere packing to build coding intuition. Subsection 2.1.4 introduces some basic interleaving concepts that are useful to map codes to constellations.

2.1.1 Dimensionality-Increasing Examples

This subsection illustrates increased-dimensionality’s improved possible coding gain through simple examples. From Chapter 1, a sender transmits one of \( M = 2^b \) messages with index \( i = 0, ..., M - 1 \), and with a corresponding symbol vectors \( x_i, i = 0, ..., M - 1 \), over \( N \) dimensions. When the channel input allows real symbol inputs, then \( x \in \mathbb{R}^N \), and the average energy is \( E_x \) with \( \bar{E}_x = E_x / N \) as the average energy per dimension, and similarly the bits per dimension is \( \bar{b} = b / N \).

Coding’s formal definition expansion – beyond Chapter 1’s definitions – appears later in Subsection 2.1.2’s Definition 2.1.2. This subsection provides code examples with a code as a symbol set, just as in Chapter 1. All members of this symbol/code set \( \{x_i\}_{i=0}^{M-1} \) remain \( N \) dimensional, and these symbols are then also codewords. The code thus has \( M = 2^b \) codewords. In this text and the imminent examples, these codewords concatenate \( N \) subsymbols \( \tilde{x}_n, n = 1, ..., N \) to form an \( N \)-dimensional symbol-vector codeword

\[
x = \begin{bmatrix}
\tilde{x}_1 \\
\vdots \\
\tilde{x}_N
\end{bmatrix}
\]

The subsymbols each have \( \bar{N} \)-dimensions, and thus \( N = \bar{N} \cdot \tilde{N} \). The code uses subsymbols from an \( \tilde{N} \)-dimensional subsymbol constellation \( C \) that has \( |C| \) possible subsymbol-vector possibilities. Simple constellations may contain \( \tilde{x} \in C \), so two real dimensions as in Chapter 1’s QAM. In this case, \( x \in \mathbb{C}^\mathbb{N} \) is equivalent to \( x \in \mathbb{R}^N \) with \( N = 2\bar{N} \). Larger dimensional constellations are also possible, and have an even number of real dimensions \( \tilde{N} \in 2\mathbb{Z}^+ \) if built upon multiple complex dimensions. While Chapter 1’s constellations had \( M = |C| \) subsymbols, Chapter 1 stated that more generally \( |C| \geq M \), and that begins to occur with this subsection’s Example 2.1.3 after two simpler examples first have \( M = |C| \). However, the statement \( M = |C| \) applies when \( M \) and \( C \) correspond to symbols and subsymbols that have the same number of dimensions. This statement’s generalization going forward is \( |C| \geq M^{1/N} \). The number of bits/subsymbol will be \( \tilde{b} \triangleq b / \bar{N} \) and the energy per subsymbol will be \( \tilde{E}_x \triangleq \bar{E}_x / N \). Clearly \( \tilde{E}_x = \bar{E}_x / \bar{N} \) and \( \bar{b} = \tilde{b} / \tilde{N} \). This text’s coding viewpoint builds upon this repeated use of a smaller subsymbol constellation to develop all codes in this chapter, and in particular in Chapters 9 - 11.

EXAMPLE 2.1.1 (4PAM simple one-dimensional code) Figure 2.1 provides a trivial code example for \( N = 1 \) where constellation symbols are equally spaced within a given energy constraint, which is Chapter 1’s 4PAM constellation. The 4 symbols shown are \( \pm 1 \pm 3 \), and

![4PAM Constellation](image-url)
the minimum distance is \( d_{\text{min}} = 2 \). For the given energy of \( \tilde{E}_x = 5 \), 4PAM is the optimum single-dimension 2-bit code for the AWGN channel.

Single-dimension codes are trivial, but the next example illustrates a small gain from use of two dimensions.

**EXAMPLE 2.1.2 (SQ versus HEX constellations in two dimensions)**

Figure 2.2 progresses Example 2.1.1’s trivial 4PAM example to compare two different two-dimensional (2D) constellations. Figure 2.2(a)’s code repeats Example 2.1’s two 4PAM constellations to generate 16 symbol vectors in two dimensions, or 16SQ. However, 16SQ is inferior to Figure 2.2(b)’s 16HEX code that is shown as a hexagonal symbol-vector array. For a given energy per dimension of 5 units, 16HEX is 0.49 dB (see Problem 2.1) better than 16SQ because 16HEX packs symbols more tightly for the given minimum distance of \( d_{\text{min}} = 2 \). The number of nearest neighbors increases from 16SQ’s maximum of 4 to 16HEX’s maximum of 6, but 16HEX’s distance increase will be larger than the nearest-neighbor-increase effect in terms of impact on \( P_e \). 16HEX’s hexagonal symbol array for any given energy is best when \( N = 2 \). 16HEX’s positive coding gain (up to .625 dB for HEX over SQ constellations as in Chapter 1 notes) illustrates that two-dimensional packing can be more efficient than in 1 dimension.

![Diagram of 16SQ and 16HEX constellations](image)

Similarly in 3 dimensions, there are two equally dense packings: the first is the “face-centered cubic” (FCC), or also called “cubic close packed” (CCP) lattice, and the second is the “hexagonal closed-packed” (HCP or “A4”) lattice. These pack 3D symbol vectors as close as possible. Both have 12-sided Voronoi (decision) regions that tessellate all space (but these are not regular polyhedrons in that some planar faces have 3 sides and some have 4) and do not so well approximate a sphere. For any \( M \), code design would use one of these lattices and pick the \( M \) lowest energy symbols, before subtracting the

---

1. This distance-domination at least holds for SNR’s that exceed roughly 4.5 dB, which can be estimated by looking at this text’s mid-SNR Q-function plots in Appendix A. However, the nearest-neighbor union bound tends to over-emphasize nearest neighbors when these neighbors are not all on mutually orthogonal dimensions. Later more complete “sphere packing” and “asymptotic equipartion” arguments will assure that closer packing dominates the associated nearest-neighbor increase as long as data transmission is feasible.

2. More precisely, the M-HEX constellation should choose \( M \) symbols within the smallest circular boundary from a hexagonal array and then translate that array by subtracting its centroid, or mean two-dimensional average, see Problem 2.1.
centroid $\bar{x} = \frac{1}{M} \sum_{i=0}^{M-1} x_i$ from all symbols to get the best code in 3 dimensions. The fundamental gain for constellations based on such 3D lattices is 1.0 dB, as explored further in Problem 2.2. The best 4D structure (at least in terms of those based on lattices) occurs in Example 2.1.3. 4 dimensions, and more generally even numbers of dimensions, are more appealing to designers because modulation is often based on systems with multiple uses of an two-dimensional “subsymbol” (i.e. often generated by Chapter 1’s QAM), which Subsection 2.1.2 more generally defines.

The redundancy measures in bits a code constellation’s amount of extra subsymbol vectors.

**Definition 2.1.1 (Redundancy)** The code redundancy depends upon the code constellation $C$ and the number of bits/symbol $b$:

$$\rho \triangleq N \cdot \log_2(|C|) - b .$$

(2.1)

The redundancy per subsymbol consequently is

$$\tilde{\rho} = \log_2(|C|) - \tilde{b} ,$$

(2.2)

and the redundancy per-dimension is

$$\bar{\rho} = \tilde{\rho}/\tilde{N} .$$

(2.3)

Nonzero redundancy forces constellation subsymbol vectors closer together, so a good code’s concatenation of constellation uses creates subsymbol sequences (or codewords) that more than compensate for this otherwise apparent subsymbol-distance reduction. The next example further illustrates this effect.

**EXAMPLE 2.1.3 (Four Dimensional D4 Lattice with 24CR constellation)** Figure 2.3’s upper portion illustrates two successive 24CR-constellation subsymbols over a 4 dimensions (with minimum distance 2 between closest subsymbols in the 24CR subsymbol). As the trellis diagram below the constellations indicates, allowed codewords (symbols) can have only even (blue) 24CR subsymbols follow even subsymbols, while only odd (red) subsymbols can follow odd subsymbols. Further, allowed codewords can use the “outer” subsymbols outside the center 16SQ subsymbols maximally once within the two 2-dimensional subsymbols’ concatenated codeword.

The number of subsymbols that can appear for codewords formed along the upper blue trellis-branch pair is thus

$$M_{\text{Upper path}} = \frac{8}{\text{inner}} \times \frac{8}{\text{inner}} + \frac{4}{\text{outer}} \times \frac{8}{\text{inner}} + \frac{8}{\text{inner}} \times \frac{4}{\text{outer}} = 128$$

(2.4)

so there are 7 bits transmitted by 4-dimensional upper-path codewords. Similarly 128 possibilities exist for codewords formed along the red lower path. Thus, there are in total of $128 + 128 = 256$ possibilities in 4 dimensions for a total of 8 bits transmitted. All these 4D symbols are equally likely to occur.

However, in two dimensions, the outer subsymbols occur less frequently, in fact with probability

$$P_{24CR\text{-outer}} = \frac{32}{256} + \frac{32}{256} = \frac{1}{4} \text{ or } \frac{1}{32} \text{ each,}$$

(2.5)

while the inner subsymbols occur more frequently at probability $P_{24CR\text{-inner}} = \frac{3}{8}$, or $\frac{3}{64}$ each. 4D codewords/symbols with very large energy that might have occurred with two
successive outer subsymbols have thus been avoided in this D4 Lattice code. Such aversion of large-energy symbols improves Chapter 1’s shaping gain $\gamma_s$. The $d_{min}$ between closest codewords occurs when a 4D codeword with two even subsymbols in a row (trellis upper path) has closest odd subsymbols twice in a row (trellis lower path) in a second codeword. That minimum distance is $d_{min} = 2\sqrt{2}$. Closest symbols for that minimum distance of $2\sqrt{2}$ also can occur within the even subsymbols, or within the odd subsymbols, and is also at this same distance $2\sqrt{2}$, when the other subsymbol value in the compared codewords is the same.

By contrast, sending subsymbols from 16SQ with minimum distance 2, followed by subsymbols from 16SQ (with minimum distance 2) would be a form of “uncoded” transmission. It also conveys 8 bits. This uncoded system would use $E_x = 10 + 10 = 20$ units of total energy for the average 4-dimensional energy with minimum distance 2. The coded 24CR system instead uses $E_x = 2 \cdot \left[ 2^{8/24} \cdot 10 + 2^{8/2} \cdot 26 \right] = 28$ with minimum distance $2\sqrt{2}$ so the coding gain is $\gamma = (8/28)/(4/20) = 1.43$ (or 1.55 dB). Thus D4 is better yet than the 1D, 2D, and 3D cases. With larger subsymbol-constellation sizes, $|C|$, this D4 lattice-code fundamental gain can be as high as 1.5 dB - in this example there is a slight shaping gain also included in the coding gain.

Example 2.1.3’s 4D code uses the best possible lattice-based structure in four dimensions, and cor-

![Good 4-dimensional code based on the D4 lattice structure.](image-url)
responds to what is called a “Schlafli” or “D4” lattice in mathematics. This code’s γ = 1.55 dB gain illustrates again how dimensionality can be used to improve coding and shaping, even though the simple 2D constituent subsymbols have been maintained (so the transmitted signal looks like 24CR, where indeed |C| = 24 ≥ M^{1/2} = 16). With some effort, the maximum number of nearest D4 neighbors is computed as N_e = 24 = (4 × 4 + 4 + 4), but again the coding gain on the AWGN is larger than the loss from increased nearest neighbors. As |C| → ∞ and thus also M → ∞, then γ → γ_f = √2 or 1.5 dB. The pattern is clear, gain over simple uncoded transmission increases Chapter 1’s fundamental gain to γ_f = .625 dB in 2D, to γ_f = 1 dB in 3D, and to γ_f = 1.5 dB in 4D, which motivates the next subsection.

### 2.1.2 Formal Code definition and Redundancy

A formal code definition develops further the concept of dimensional use:

**Definition 2.1.2 (Code)** A code is any set of \( M = 2^b \), \( N \)-dimensional codewords

\[
C_x = \{ x_i \}_{i=0,\ldots,M-1}
\]

where the \( N \)-dimensional codewords have \( \bar{N} \), \( \bar{N} \)-dimensional subsymbols selected from an \( \bar{N} \)-dimensional subsymbol constellation \( C \) with |\( C \)| subsymbols. The subscript \( x \) on \( C_x \) distinguishes \( C_x \) from the subsymbol constellation \( C \). Thus,

\[
N = \bar{N} \cdot \frac{\bar{N}}{\# \text{ of subsymbols}}.
\]

A codeword generalizes Chapter 1’s symbol, and indeed also is a “symbol,” just usually a more complex structured symbol. \( \bar{N} \) is the blocklength in subsymbols, while \( N \) remains the symbol’s number of real dimensions when the constellation has real vector subsymbols \( x \in \mathbb{R} \). The difference between a subsymbol and Chapter 1’s symbol is simply the implication that the code may concatenate more than one successive subsymbol selected from the same \( \bar{N} \)-dimensional constellation to form a codeword. The choice of values within that constellation for that particular code may be limited to a subset of the constellation’s |\( C \)| subsymbol values. The number of bits per subsymbol is

\[
b = \frac{b}{\bar{N}} = \bar{b} \cdot \bar{N} = b \cdot \frac{\bar{N}}{N}.
\]

\( \bar{N} < \infty \) is always finite. (Often \( \bar{N} = 2 \).) The code’s minimum distance for AWGN use is

\[
d_{\text{min}}(C_x) \triangleq d_{\text{min}} = \min_{x_i \neq x_j} \| x_i - x_j \|,
\]

while for the BSC will remain, with \( d_H \) being the Hamming distance corresponding to the number of bit positions in which two bit vectors differ, this free distance is

\[
d_{\text{free}} = \min_{v_i \neq v_j} d_H(v_i - v_j).
\]

---

Figure 2.4 illustrates codeword construction from \( b \) input bits.

---

3The best lattice-based packings are known for all dimensions up to \( N = 8 \), as well as some higher dimensionalities as well, and can be found on-line but beyond this text’s scope.
The encoder’s generator possibly adds $\rho \geq 0$ redundant bits before Figure 2.4’s vector modulator uses those $b + \rho$ bits to select a vector/codeword from a code of $M = 2^b$, $N$-dimensional codeword symbols. Because there are only $M = 2^b$ codeword choices, the subsymbol’s redundant bits may appear useless. However, the redundant bits enlarge the set of subsymbol choices to a set of $2^{b+\rho}$ constellation vectors. Definition 2.1.1’s redundancy allows variation of possible subsymbol choice within a codeword. If $N = 1$, the redundancy necessarily is zero in practice. A longer nontrivial code’s codewords decompose into $N > 1$, $N$-dimensional subsymbols $\tilde{x}_{n=1,\ldots,N}$ as Figure 2.4 shows. The simple subsymbol modulator and the simple subsymbol demodulator are the same as in Chapter 1. The ML detector selects the code’s closest codeword $\hat{x} = \arg\max_{x \in \mathcal{C}} \{py|x, \} \to$ the channel output $y$ (with closest meaning euclidean distance for the AWGN channel and Hamming distance for the BSC). The ML-detector implementation may be complex in general, but often the $N$-dimensional codeword’s construction from $N$-dimensional subsymbols’ allows the ML detector’s simplification.

Following Subsection 2.1.1’s heuristic introduction, the code formally uses multiple dimensions to create a good code with greater minimum distance than could be obtained with simple constellations alone, improving upon those in Chapter 1. There are many ways to encode or generate codewords from bits that can be linear, non-linear, and/or involve memory from previous codewords and/or symbols. Again, $\tilde{N}$ is always finite. When $N < \infty$, and thus $N < \infty$, are finite, the code is called a block code (see Chapter 10). When $\tilde{N} \to \infty$, and thus $N \to \infty$, are infinite but the encoder bases their generation on a finite number of states that characterize the encoder’s history, the code is a trellis or sliding block code (see Chapter 9). Codes of codes may also occur (just as the code uses subsymbols that are simple codes, the subsymbols themselves can become sophisticated codes) in concatenated codes, as in Chapters 8, 9, and more elaborately Chapter 11.

There can be different views of the same code. As a simple example, Chapter 1’s 32CR constellation has two interpretations: From a 2D subsymbol viewpoint, $N = \tilde{N} = 2$, $M = 32$, $b = 5$, and thus $\tilde{N} = 1$ with $d_{min} = 2$ and $|C| = 32$. Also, $\tilde{b} = 2.5$ and $\tilde{\rho} = 5$. However, from another 1D subsymbol viewpoint 32CR has $N = 1$, $\tilde{N} = 32$, $b = 5$, $N = 2$, and thus $\tilde{N} = 2$ with $d_{min} = 2$ and $|C| = 6$. The second viewpoint suggests the definition of redundancy in Definition 2.1.1. $M$ and $\tilde{b} = \log_2(M)$ are the same from all viewpoints. $|C|$ and $\tilde{b}$ can vary with the choice of $N$ and $\tilde{N}$ or equivalently with $\tilde{N} = N/\tilde{N}$.

Reasonable codes must have non-negative redundancy, or else the code would not have enough symbols for a unique mapping of input messages to codewords. The second 1D interpretation of 32CR above would have a redundancy of $\rho = 2 \cdot \log_2(6) - 5 = .17$ bits/codeword and $\tilde{\rho} = .085$ bits/dimension. The first 2D interpretation has redundancy 0 bits/subsymbol. From the 1D subsymbol viewpoint, the successive 1D constellations might maximally have $6 \times 6 = 36 > 32$ symbols in two dimensions, implying that there may be extra information or redundancy in those 36 possibilities because the code uses only 32 of them. Clearly, no more than $2^b$ values are needed for any particular subsymbol, but the redundancy $\tilde{\rho}$ allows the choices to vary over the code’s blocklength. For the 32CR, the non-zero redundancy essentially implies that by reducing the probability of the largest outer (±5) one-dimensional subsymbol’s probability, as 32CR does, that some gain is possible (and indeed 32CR has a small coding
gain over Problem 1.14’s 32SQ for instance). However, from a 2D perspective, no further gain is possible when limited to the 32 symbols because all 32 symbols need to be equally likely to carry 5 bits of information. This suggests the concept of measuring information via the probability of constellation subsymbols, which Subsection 2.3.1’s entropy addresses.

Example 2.1.3’s D4 lattice code had, from a 2D viewpoint, $\tilde{N} = 2$, $M = 256$, $b = 8$, $N = 2$ and thus $N = 4$; also $d_{\text{min}} = 2\sqrt{2}$. Further $\tilde{b} = 4$ and $\bar{b} = 2$, but $|C| = 24$. The redundancy is thus $\rho = 2 \cdot \log_2(24) - 8 = 1.17$ bits/codeword, or $\bar{\rho} = 0.58$ bits/subsymbol. The redundancy for the compared “uncoded” 16SQ x 16SQ system would be 0. The D4 code uses redundancy to improve coding gain. A basic operation in Example 2.1.3 was concatenation of multiple instances (two) of a smaller code (24CR) into a 4D code, a subset of the D4 lattice. This is a form of code concatenation. Codes built on other lower-dimensional codes can increase the gain up to certain fundamental limits discussed in Section 2.3.

The above discussion and examples permit formal characterization of coded versus uncoded:

**Definition 2.1.3 (Uncoded and Coded)** An uncoded data transmission will have subsymbol constellation $C$ with zero redundancy $\bar{\rho} = 0$. Necessarily then uncoded transmission also has $\rho = 0$ and $\bar{\rho} = 0$. Usually in uncoded transmission, the codeword and the subsymbol are trivially the same. If the redundancy is greater than zero, $\rho > 0$, then data transmission is coded.

Examples 2.1.2 and 2.1.3 also illustrate two effects: The first effect is the good code’s packing of symbols more densely in a given spatial volume. This packing can improve as $N$ increases. The measure of this packing is Chapter 1’s fundamental coding gain, $\gamma_f$. The second effect is trying to delete or reduce the probability of the constellation’s larger outer subsymbols so that the constellation’s shape is more (hyper) spherical. The measure of this constellation shaping is Chapter 1’s shaping gain, $\gamma_s$.

### 2.1.3 Asymptotics

To pursue coding improvements, an important statistical result is the law of large numbers (LLN, from Appendix A). To understand this result, the sample average over $N$ observations of a random variable is:

$$\hat{x} \triangleq \frac{1}{N} \sum_{n=1}^{N} x_n .$$

(2.11)

Since functions of random variables are also random variables, the sample average also applies directly to the function $f(\bullet)$ applied to each and every observation so

$$\hat{f}(x) \triangleq \frac{1}{N} \sum_{n=1}^{N} f(x_n) ,$$

(2.12)

as long as the function is well behaved and stationary (invariant over the observations taken). With observations’ selection from a stationary probability distribution, the law of large numbers essentially states the obvious about the sample average and the true mean as $N \to \infty$.

**Theorem 2.1.1 (Law of Large Numbers (LLN))** The LLN observes that a stationary random variable $z$’s sample average over its observations $\{z_n\}_{n=1,...,N}$ converges to its mean with large $N$ such that

$$\lim_{N \to \infty} \Pr \left\{ \left| \frac{1}{N} \sum_{n=1}^{N} z_n - \mathbb{E}[z] \right| > \epsilon \right\} \to 0 \quad \text{weak form}$$

(2.13)

$$\lim_{N \to \infty} \Pr \left\{ \frac{1}{N} \sum_{n=1}^{N} z_n = \mathbb{E}[z] \right\} = 1 \quad \text{strong form} .$$

(2.14)
The LLN applies equally well to a function of a random variable and that function’s sample average. One LLN use profits from the random variable function being the logarithm of the sampled probability distribution itself, so \( z = \log_2(p_x) \). This leads to Subsection 2.3.3’s asymptotic equipartition. The LLN is also useful in extending dimensionality to where a codewords’ subsymbols correspond to independent multiple selections from the same distribution. In such a situation, the code is effectively chosen at random, which with enough attempts must find several good codes if transmission is feasible, as in the sphere packing to follow.

2.1.3.1 Sphere Packing

From the intuitive perspective of Subsections 2.1.1 and 2.1.2 so far, and as \( N \to \infty \), a good code could select codewords (symbols) in a hypersphere with those codewords uniformly distributed throughout that hypersphere’s volume at maximum density and with good inter-symbol spacing. The smallest volume to surround a point in space (for a given energy) is a hyper-sphere. Thus ideally, \( M \) smaller hyper-spheres can be packed into that larger hyper-sphere volume. The finite-dimensional small spheres don’t perfectly cover all the space and have gaps or “holes” such as those discussed in the 3D lattice above, but as \( N \to \infty \) the smaller hyper-spheres will cover the larger hyper-sphere with infinitesimally small error. Thus, at least one optimum or best code is thus heuristically described (although its implementation complexity may be very high) as the set of all center points of the mutual-uniformly positioned hyperspheres.

To follow further, those smaller hyper-spheres’ size/radius decreases as \( M \) increases (for fixed energy/dimension). The LLN relates that when the function is \( f(x) = p_2(x) \), then the sampled averaged probability distribution itself becomes a constant, namely the mean \( E[p_2(x)] \), implying a uniform distribution. Further, since the LLN applied with \( f(x) = \|x\|^2 \) says that all codewords have the same sample-average energy. That is, not only are the codewords approaching a uniform distribution in probability, but with probability approaching 1, they all lie on the hypersphere’s surface. (This is equivalent to the well-known geometry result that all the volume of a hypersphere is at its surface.) This requires that the codeword subsymbols were chosen effectively independently from the Gaussian distribution. Similarly then white Gaussian noise’s projection in each dimension can also be viewed as being sampled from a uniform distribution over its own hyper-spherical volume with the noise hyper-sphere’s radius \( \sigma = \sqrt{\frac{N_0}{2}} \). As the noise is usually smaller than the signal, the noise can be viewed as a small hypersphere and the signal as a larger hypersphere, and indeed the received signal (since it too is white Gaussian) as a slightly larger yet hypersphere.

Figure 2.5 depicts a single great arc on a great 2D circle of the hypersphere’s surface. Since all the codewords with significant probability must be on the outer surface, these align uniformly on any such great arc and circle as Figure 2.5 shows. For complete reliability (error probability approaching zero) and using the LLN, the probability of the noise’s asymptotically constant (probability 1) radius must be smaller than half the distance between the codewords. For the codeword vectors, the great circle’s circumference is \( 2\pi \sqrt{\frac{E}{y}} \) for any signal and noise. The signal’s great circle and the noise’s great possible circle positions both appear in Figure 2.5.

Another viewpoint uses the Central Limit Theorem and recognizes that any dimension’s component of any random infinite-dimensional uniformly distributed random signal can be viewed as a sum of all noise components onto an infinite-dimensional matched-filter basis. At least one such basis would have many contributions from all the other dimensions, and the Central Limit Theorem would say that component is Gaussian in that and any corresponding dimension.
Fitting the uniform noise circles around the the great circle uniformly and recognizing that the noise projection in the same (or any) circumferential single dimension would be $2\pi\sigma$, the number of codewords' subsymbols (with probability one by LLN) on any such circle is then

$$M^{1/N} = \frac{2\pi \sqrt{\bar{E}_y}}{2\pi \sigma} = \frac{\sqrt{\bar{E}_x + \sigma}}{\sigma} = \sqrt{1 + SNR}, \quad (2.15)$$

or

$$\bar{b} = \frac{1}{2} \cdot \log_2 (1 + SNR). \quad (2.16)$$

This is the AWGN's well-known capacity, simply shown through intuition of coding and some assistance from the LLN. When the noise circumferential projection on average exceeds half the circumferential distance, then with probability 1 the opposite is true, and performance will rapidly degrade. Thus, at least this one sphere-packing code exists that also has vanishingly small error probability when
the codeword spheres fit into the energy. Section 2.3 formalizes this radial-counting argument with the concept of asymptotic equipartition. The constellation essentially has infinite redundancy because subsymbols can be any selected from a Gaussian distribution. Clearly the large redundancy is useful in this case to attain maximum data rate reliably.

2.1.4 Interleaving

Figure 2.6 illustrates basic interleaving. Interleaving is a 1-to-1 reordering of information units (bits, subsymbols, codewords) in the transmitter with corresponding receiver’s de-interleaving (restoration of the original order). The interleaver typically is positioned between two concatenated codes, the first or outer code and the second or inner code. If the outer encode produces bits, the interleaver reorders those bits. Additionally, when the inner code is a simple modulator (like QAM or PAM, what is called here “uncoded”), the system is known as **Bit Interleaved Coded Modulation (BICM)**.

![Figure 2.6: Interleaving concept.](image)

Chapter 11 investigates many forms of interleaving and the theory behind it. Problem 2.8 provides a simple example of BICM and the gains achieved from such a simple interleaving system. BICM indeed is a means for transferring the high gains of most good well-known binary input/output codes for the BSC to the AWGN if the assignment of interleaved bits to the subsymbols in the consequent implied code uses Chapter 1’s gray mapping, see also Problem 2.8 for a simple example of gray coding, where adjacent 2D symbols differ only in one bit.
2.2 Coded systems, performance, and decoding

While Section 2.1’s concept of uniformly packed non-overlapping hyper-spheres of equal (one-dimensional) diameter ($\approx d_{\text{min}}$) inside a larger sphere of (one-dimensional) radius $\sqrt{\bar{E}_x + \sigma^2}$ is intuitively, and indeed actually, optimum in terms of the maximum number of messages that can be reliably transmitted on the AWGN, it avoids the actual code-design details, as well as encoder and decoder implementation. This subsection addresses these additional subjects. Code designers have identified various codes that approximate a good distribution of codeword subsymbols. These codes fall into a few broad categories that this subsection characterizes. Subsections 2.2.1 and 2.2.2 will respectively introduce “soft” and “hard” decoding concepts and implications for code performance and characterization. Subsection 2.2.3 looks at the intermediate (to soft/hard) erasure channels and associated performance and concepts. These first 3 subsections all address finite-length codes, while Subsection 2.2.4 extends these concepts to trellis/convolutional codes that use memory to approximate infinite code length with finite real-time complexity.

2.2.1 Soft-Decoded Systems

Soft-decoded systems apply directly to the AWGN channel, which has continuously distributed channel output for all subsymbols. The ML detector for any AWGN-based code finds the input codeword (symbol) that minimizes the sum of squared differences between channel output and that codeword, so

$$\hat{x} = \arg\left\{ \min_x \sum_{n=1}^{N} |y_n - \hat{x}_n|^2 \right\},$$

(2.17)

where $n$ is the subsymbol dimensional index, and

$$\hat{x} = \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \vdots \\ \hat{x}_N \end{bmatrix}.$$

(2.18)

Chapter 1’s ML decoders were comparatively simple in that the sum in (2.17) was one term. Equation (2.17)’s direct calculation will grow exponentially with $N$ as $|C|^N$ squared-distance computations. Thus, while simple in concept, the implementation may be overwhelming for reasonable $N$ values. Most code designs have structure that enable decoder implementations with acceptable complexity. The minimum distance calculation may also search over every possible pair of codewords, which also can grow in complexity proportional to $|C|^N$. Regardless of complexity, the ML detectors symbol-error probability can again be approximated with the NNUB (with $d_0 \Delta d_{\text{min}}$ and $d_{i \geq 1}$ as the distances in increasing size, while correspondingly $N_0 = N_e$ and $N_i$ are the number of neighbors at distance index $i$ ) as

$$P_e \leq \sum_{i \geq 0} N_i \cdot Q\left( \frac{d_i}{2\sigma} \right),$$

(2.19)

$$\approx N_e \cdot Q\left( \frac{d_{\text{min}}}{2\sigma} \right).$$

(2.20)

The nearest neighbor calculation ($N_e$) can rapidly become complex with large $N$. Also, as $N$ increases, some next-to-nearest neighbors - denoted $N_1$, or next-to-next-to-nearest neighbors – denoted $N_2$, and so on – can provide larger contributions to the $P_e$ bound than the $N_e$ term if $N_i \gg N_e$. Usually good codes also have well understood distributions of $d_{\text{min}}$, $d_1$, $d_2$, ..., and the corresponding numbers of nearest neighbors $N_e$, $N_1$, $N_2$ ... at these distances respectively.
Lattice codes have finite dimension. The codewords are \( N \) dimensional vectors and the notation \( \chi \in C_x \) means that subsymbol \( \chi_n \) is punctured (removed) from all codewords. Then, minimization of subsymbol \( n \)'s error probability would maximize the conditional probability\(^5\)

\[
p_{x_n/y}(\chi_n, v) \propto p_{x_n/y}(\chi_n, v) = \sum_{\chi \in \{\chi \in C_x \}} p_{x_n/y}(\chi \setminus \chi_n, v),
\]

which as indicated is proportional to (by a function only of the fixed specific channel output value \( y \) and not the input codewords over which the maximum is chosen) the marginal probability distribution, \( p_{x/y} \). The marginal distribution's calculation can be complex, but the concept is straightforward. Chapter 8 derives a number of iterative ways to compute all the marginal distributions for a codeword's subsymbols with reduced complexity. Similarly the probability of the input-bit vector \( u = [u_1 ... u_b] \), taking on all binary values (0, 1)\(^b\) corresponding to each codeword could be also maximized over its individual marginal distributions for each bit, allowing a receiver/detector that minimizes each and every bit's error probability through maximization of

\[
p_{u_i/y}(u, v) \propto p_{u_i/y}(u, v) = \sum_{u \in \{0,1\}^b \setminus u_i} p_{u_i, y}(u, v) \quad \forall \quad i = 1, ..., b.
\]

### Log-Likelihood Ratios:
As in Section 1.1.6, the bit-wise ML decoder may instead compute the log likelihood ratio (LLR) as a function of the channel output \( y \). Problem 2.3 provides a simple example of this calculation with SPAM. This LLR simplifies computation by exploiting first each bit's value limitations and second that products of many probability distributions simplify to sums under logarithmic calculation through

\[
\text{LLR}_i(y) \triangleq \ln \left( \frac{Pr\{u_i = 1; y\}}{Pr\{u_i = 0; y\}} \right).
\]

Positive values of the LLR for any bit indicate that a 1 is more likely, while negative values indicate a 0 is more likely.

### 2.2.1.1 Lattice and Block Codes

Lattice codes have finite \( \tilde{N} \) (and thus finite \( N \)), and often find use on the soft-decoded AWGN. Lattice codes also often use two-dimensional subsymbols, so then \( \tilde{N} = 2 \). A formal lattice definition appears in Appendix E, but succinctly a lattice \( \Lambda \) is a group of \( N \)-dimensional vectors \( x \) closed under addition. For real-number (\( x \in \mathbb{R}^N \)) or complex-number (\( x \in \mathbb{C}^N \)) addition, or equivalently the real or baseband/analytic complex AWGN respectively, the lattice necessarily has a countably infinite number of symbols. Such lattices have all elements generated by \( N \) linearly independent generator vectors \( g_n \) so that a lattice element is generated by \( x = \sum_{n=1}^{N} z_n \cdot g_n \) where \( z_n \in \mathbb{Z} \) are integers and \( g_n \) are the real (\( \tilde{N} = 1 \) and \( g_n \in \mathbb{R}^N = \mathbb{R}^N \)), or complex (\( \tilde{N} = 2 \) and \( g_n \in \mathbb{C}^N \), equivalently \( \mathbb{R}^{2N} \)) generator (column)\(^6\) generator matrix

---

\(^5\)The sum here is actually \( \tilde{N} - 1 \) summations, excluding input \( s \) index \( n \). \( \chi_1, \chi_2, ..., \chi_N \) to compute the desired marginal conditional distribution.

\(^6\)This text writes real/complex lattice generators with lattice name as subscript to distinguish them from finite-field codeword generators that will have no such subscript. While the concepts are very similar, they are not identical. Further coding theorists almost universally write finite-field codewords as row vectors to reflect time occurring horizontally. In those cases where \( v \) and \( u \) are finite-field codeword outputs and inputs respectively, then \( v = u \cdot G \) so then effectively the generator is transposed notionally with respect to lattice and real/complex modulating symbol generation.

---
$G_\Lambda$ describes the lattice as all vectors formed by $G_\Lambda \cdot z$ where $z \in \mathbb{Z}^N$ is a vector of $N$ integers and

$$ G_\Lambda = \left[ g_{\Lambda,1} \cdots g_{\Lambda,N} \right]. $$

The linear code’s minimum distance is the smallest Euclidean distance between any of the $N$ generator vectors and the all-zeros vector, or equivalently

$$ d_{\text{min}} = \min_i \| g_i \|. \quad (2.24) $$

The Voronoi Region $V(\Lambda)$ has fundamental volume $V(\Lambda) = |V(\Lambda)|$, which is thus the determinant of the square nonsingular generator matrix, $V(\Lambda) = |G_\Lambda|$, since $V(\mathbb{Z}^N) = 1$. Usually in lattice descriptions, the generator is real and so complex-baseband related lattices subsymbols are 2-dimensional real vectors for encoding. However, the generator can be complex (with half as many dimensions, but each complex) with complex integer (Gaussian integer) inputs $z$. Problem 2.4 explores further generators and their structure with lattices. Chapters 9 and 10 further pursue lattice codes.

**Voronoi Regions and Spheres:** Lattice codes for the AWGN channel typically pick the $M = 2^b$ lowest-energy elements as symbols from a lattice, and then translate that set to zero its centroid\(^8\) and thereby form the code. The lattice’s linear structure can simplify encoder and decoder designs. As in Chapter 1, each lattice element has a Voronoi region $V(\Lambda)$ that is like the hexagon of Example 2.1.2’s $A_2$-lattice-based constellation. Example 2.1.3’s lattice is yet better and has a 24-sized Voronoi region. The Voronoi region will always have the number of sides equal to any symbol’s maximum possible number of nearest neighbors (maximum $N_e$). Voronoi regions in very high dimensions can increasingly approximate hyperspheres for some lattices. Such lattices help design good codes, both in terms of symbol spacing (larger $d_{\text{min}}$ and better $\gamma_f$) as if each region $V(\Lambda)$ around a subsymbol is like a small hypersphere at a micro level, and secondly at a macro level the union of all symbols’ Voronoi regions approximates a larger hypersphere for the constellation’s boundary (smaller $\bar{E}_x$ and better $\gamma_s$) into which all $V(\Lambda)$ are packed, as further investigated in Chapters 8 and 10. There are also other non-Voronoi methods for code design.

**Iterative Decoding:** For lattice and block codes, marginal distributions can be computed so that individual bits (or subsymbols) can each/all have their own error probability minimized at the expense of additional complexity. Some such lattice/block-coded systems will be able to achieve coding gap close to 0 dB. Figure 2.7 illustrates the decoder’s $LLR$ computation for one bit position in such a block.

---

\(^8\)This translate is often called a *coset* of the lattice, meaning here that the $M$ lowest-energy elements from the centroid-centered coset $x_i \rightarrow x_i - \frac{1}{M} \sum_{j=0}^{M-1} x_j$ of the lattice are used.
code. Each bit’s LLR update calculation will use information from other code bits’ similar structures to compute LLR estimates iteratively. The other \( j \neq i \) bits’ contributions enter through the averaging implicit in the margin-distribution calculation. Each bit’s contribution to other bits’ decoders changes on each iteration, with hopefully the entire set converging to a stable set of \( LLR_i \)'s. A large impulsive (non-Gaussian) intermittent noise usually does not impact an LLR as much as it would impact ML’s squared-distance metric with AWGN. When noise is purely Gaussian, decoders’ use of LLR’s or squared distance will perform the same\(^9\), so both are optimum for the AWGN. Thus the LLR-based decoder can have additional merit when impulsive non-stationary noise adds to stationary Gaussian noise. An example of codes that perform well in such of decoder LLR's occurs with “low-density parity check” (LDPC) codes, which appear in Chapters 8 - 10.

### 2.2.2 Hard-Decoded Systems

Figure 2.8 repeats Chapter 1’s Binary Symmetric Channel (BSC). The BSC has an error probability of the discrete binary output not equaling the input. That error probability \( p \), is the BSC’s sole characterizing parameter. This BSC is used in hard decoding. Figure 2.8 describes the probability that a certain output is received, given any particular input.

![Figure 2.8: The Binary Symmetric Channel, BSC.](image)

In hard-decoded designs, intermediate decisions are made on each subsymbol to simplify decoding, despite that (possibly) being sub-optimum. For instance, there may be a bit-error probability \( \bar{P_b} \) for a subsymbol-by-subsymbol decoder as in Chapter 1. A BSC models this intermediate decision with \( p = \bar{P_b} \), as in Figure 2.9’s upper diagram. The hard decoder’s calculations differ from soft decoding’s optimum closest sum-squared distance as in Figure 2.9’s lower diagram. Such hard decoders might use different codes than those used with soft decoding. Hard decoding can help situations with intermittent very high-amplitude (non-Gaussian) “impulsive” noise events because a single large noise sample will not affect hard decoders as much as it typically affects soft decoders.

\(^9\)The \( LLR \) processing of \( y \) is 1-to-1 to the input bit, thus tacitly satisfying Chapter 1’s Reversibility Theorem.
Figure 2.9: Illustration of hard and soft decoding.

2.2.2.1 Binary Block Codes

Binary block codes use binary arithmetic or equivalently linear modulo-2 addition over the field $\mathbb{F}_2$, see Appendix E. A code rate $r = \frac{k}{n} = \bar{b}$ characterizes a block code. The lower-case $k$ and $n$ notation is in heavy use in the binary-coding literature. The binary code will have $|C| = 2$ with $\bar{N} = 1$ and so then $k = \bar{b} \leq n = N = \bar{N}$ and thus $r = \bar{b} \leq 1$. The situation with $r = \bar{b} = 1$ is uncoded. When $k < n$, the extra redundant bits are known as the $p$ parity bits\(^\text{10}\) where $p = n - k = \rho$ is the redundancy. The lower-case $p$ is also in common use and equal to $\rho$ with binary block codes. Common examples of binary block codes are low-density parity-check (LDPC) codes and polar codes. Also, $r + \bar{\rho} = 1$ for binary block codes. Binary block codes can also have each dimension mapped with $0, 1 \rightarrow -1, +1$ for use on the AWGN channel for either hard or soft decoding. For binary block codes, the aforementioned Hamming distance $d_H$ is simply the number of positions in which two codeword/symbol binary vectors $x_1$ and $x_2$ differ, and extends to define the free distance of a code as:

**Definition 2.2.1 (Free Distance)**

$$d_{f\text{ree}} \triangleq \min_{x_1 \neq x_2} d_H(x_1, x_2) . \ (2.25)$$

\(^{10}\)This $p$ for redundancy should not be confused with the $p$ used to characterize the BSC.
The free distance immediately provides two performance measures: The symbol/codeword error probability is zero if the number of bit errors over the BSC is less than or equal to \( \lfloor \frac{d_{\text{free}} - 1}{2} \rfloor \); and the probability of detecting an error (but not correcting it) is 1 if the number of bits in error over the BSC is less than or equal to \( d_{\text{free}} - 1 \). Clearly, the free distance is then an important parameter. The code designer may want to know for a given code rate \( r = \bar{b} = \frac{k}{n} \) (or equivalently redundancy \( \bar{\rho} = 1 - r \)) just how large a code’s \( d_{\text{free}} \) might be. For a length-\( n \) binary code, there are \( 2^n \) possible binary vectors available for use in the binary-code design. That design would pick \( 2^k \) of these. Such a code’s given free distance \( d_{\text{free}} \) would then imply that any set of \( d_{\text{free}} - 1 \) bits could be deleted from this code and still maintain distinct vectors. The number of possible codewords is then bounded as

\[
M(n, d_{\text{free}}) \leq 2^n - \left( d_{\text{free}} - 1 \right) + 1, \tag{2.26}
\]

which is known as the **Singleton Bound**\(^{11}\), formally:

**Lemma 2.2.1 (Singleton Bound)** If a designer knows the blocklength \( n \) and the \( d_{\text{free}} \) necessary for performance, then a binary block code’s rate must be less than

\[
k = \log_2(M) \leq n - d_{\text{free}} + 1 \quad \text{and} \quad r \leq 1 - \frac{d_{\text{free}} - 1}{n}. \tag{2.27, 2.28}
\]

**Proof:** See Equation (2.26). QED.

Larger \( d_{\text{free}} \) at fixed block length \( n \) imposes a lower code rate when good binary codes are used. Codes that meet the Singleton Bound are called Maximum Distance Separable (MDS) codes basically because for a given distance, no more codewords with greater separation can be added to the code. Sphere packing’s equivalent for the BSC is sometimes called ball packing and corresponds to finding symmetric regions around any particular codeword where no other codeword’s region overlaps. A lower bound (on code rate or equivalent the number of codewords) follows from the logic that the entire set of \( 2^n \) possible codewords is the union of a set of balls of “size” \( d_{\text{free}} - 1 \) around symbols\(^{12}\). The size (number of points included) of such a ball is \( \sum_{j=0}^{d_{\text{free}} - 1} \binom{n}{j} \), leading to\(^{13}\)

**Lemma 2.2.2 (Gilbert-Varshamov Bound)** If a designer knows the blocklength \( n \) and the \( d_{\text{free}} \) necessary of an existing binary block code, then

\[
\sum_{j=0}^{d_{\text{free}} - 1} \binom{n}{j} \leq M(n, d_{\text{free}}), \tag{2.29}
\]

which is known as the Gilbert-Varshamov Bound (GVB). **Proof:** See the preceding paragraph. QED.

So, a good code should have \( M(n, d_{\text{free}}) \) somewhere between these two bounds, hopefully as large as possible in this range. The GVB is easily less than the upper bound as the binomial-theorem\(^{14}\) relates that \( 2^n = \sum_{j=1}^{n} \binom{n}{j} \), or specifically \( 2^{d_{\text{free}} - 1} = \sum_{j=1}^{d_{\text{free}} - 1} \binom{d_{\text{free}} - 1}{j} \). Since \( d_{\text{free}} \leq n \), then the Gilbert-Varshamov Bound’s denominator is always greater than or equal to \( 2^{d_{\text{free}} - 1} \) because (2.29)-denominator’s sum clearly increases with the \( n \) inside the sum.

---

12This envisions \( d_{\text{free}} / 2 \) on each “side” of the ball-centered codeword.
14The Binomial Theorem states

\[
(a + b)^n = \sum_{j=0}^{n} \binom{n}{j} \cdot a^{n-j} \cdot b^j;
\]

here used with \( a = b = 1 \) and shortly to be used with \( a = 1 \) and \( b = q - 1 \).
Singleton Bound Tightness: For example, the SB states that a code that corrects up to 2 bit errors \((d_{\text{free}} \geq 5)\) with block-length \(n = 20\) needs \(r \leq 1 - \frac{4}{20} = 0.8\). At \(n = 20\), the GVB is \(\sum_{i=0}^{20} \binom{i}{j} = 169\) or \(b = 7.4\) bits. So, then \(r \geq 7.4/20 = 0.37\), then rate is somewhere between 0.37 and 0.8. As the block length grows, the required rate approaches unity, but that larger length also implies a larger codeword (so more corresponding bits) might be in error and thus larger retransmission volume if a symbol/codeword error occurs. While the Singleton Bound’s construction implies the existence of a code with \(d_{\text{free}}\), as distance for any \(n\), such a code may not exist and so there is no guarantee of tightness nor usefulness of this bound. However, there are some binary MDS codes that achieve \(2 \times 2\)’s Singleton Bound, which are for instance the rate 1 code that uses all symbols, and the all-repeat code of only 2 codewords (all ones and all zeros) where \(d_{\text{free}} = n\). Codes with a single parity bit that is the sum of all the information bits are also MDS and achieve the Singleton Bound. These are known as **trivial codes**. There are no other non-trivial **BINARY** codes that achieve the Singleton Bound. Problem 2.6 shows, the Singleton Bound is often not very tight for binary codes when \(d_{\text{free}} \neq 1, 2, \text{ or } n\).

Linear Binary Block Codes: Linear binary block codes simplify analysis greatly in that the codewords (of \(n = N\) bits) can be generated from the \(k = b\) input bits through a \(k \times n\) binary generator matrix (necessarily non-invertible if \(k < n\), unlike the invertible \(G_{\Lambda}\) for the lattice code)

\[
v = u \cdot G,
\]

where \(v\) is a \(1 \times n\) output-bit row vector and \(u\) is a \(1 \times k\) input-bit row vector\(^{15}\). The use of \(G\) here implies “generator” in a finite field\(^{16}\). The generator \(G\) has only 1’s and 0’s as entries, and the matrix multiplication uses binary addition and multiplication in \(GF(2)\). The BSC’s ML decoding rule with binary block code simply selects the codeword of least Hamming distance from the received BSC channel output. The ML decoder rule (for the linear case) then leads to a probability of codeword/symbol error of

\[
P_{\bar{e}} \leq \sum_{i > [d_{\text{free}}/2]} N_{i - [d_{\text{free}}/2] + 1} \cdot p^i \cdot (1 - p)^{N - i} \leq \sum_{i > [d_{\text{free}}/2]} \binom{N}{d_{\text{free}} + i} \cdot p^i \cdot (1 - p)^{N - i}.
\]

Again, nearest neighbors \(N_{e} = N_{0}\) are those codewords at distance \(d_{\text{free}}\) from any given codeword. The number of neighbors at distance \(d_{\text{free}} + i\) is \(N_{i}\) and cannot exceed \(\binom{N}{d_{\text{free}} + i}\) because this would mean the codewords have distance separation of \(d_{\text{free}} + i\) in all possible positions of \(d_{\text{free}} + i\) bits. The **average number of subsymbol errors per codeword** is

\[
\bar{P}_{e,ss} \leq \sum_{i > [d_{\text{free}}/2]} i \cdot N_{i} \cdot p^i \cdot (1 - p)^{N - i},
\]

while the average number of bit errors per subsymbol has upper bound

\[
\bar{P}_{b} \leq \frac{\bar{P}_{e,ss}}{b}.
\]

The free distance \(d_{\text{free}}\) is the least Hamming weight (number of 1’s) of any linear combination \(G^r\)’s rows (since the all-zeros symbol is a codeword and an input vector with \(d_{\text{free}}\) 1’s in it in the right places would be the linear combination necessary to create that all-zeros codeword as a combination of the generator rows).

MDS Code Properties: For a binary MDS code meeting the Singleton Bound, the number of nearest neighbors at distances \(d_{\text{free}} + i, i \geq 0\), (again with \(N_{0} \triangleq N_{e}\) while \(N_{i}\) is the \(i^{th}\) nearest neighbor) follows

\(^{15}\)Note the reversal of column vectors to row vectors that occurs throughout the binary-code literature.

\(^{16}\)Later sections use the notation \(G\) to represent a canonical-channel factor later that has similar function to a code generator in appropriate context - the context should be clear in each situational use of the common notation \(G\).
a form that generalizes later from this equation:

\[
N_i = \max \left\{ 0, \binom{N}{d_{free} + i} \cdot \sum_{j=0}^{i} (-1)^j \cdot \binom{d_{free} + i}{j} \cdot \left[ \binom{2i+1-j-1}{i+1-j} \right] \right\} \tag{2.34}
\]

Equation (2.34)’s first left term (before the summation symbol) clearly chooses the number of bit positions that can differ from \(N\) total bits. Then each sum term counts \(j\) bits’ selection from the exact \(d_{free} + i\) of interest for \(N_i\) but each term over counts through the last term those situations corresponding to a larger \(j + 1\) term, so they are reversed by the alternating sign in the sum. This formula is a simple form of some known as MacWilliams Identities.

**MDS Nearest Neighbors:** For instance, for binary codes, there are only a few MDS codes that achieve the singleton bound. For \(d_{free} = 1\), then and the nearest neighbor counts \(N_i\) are from Equation 2.34:

\[
N_0 = N_e = N \cdot 1 \cdot 1 \cdot 1 = N = \binom{N}{1} \tag{2.35}
\]

\[
N_1 = \binom{N}{2} \cdot [1 \cdot 1 \cdot 3 - 2 \cdot 1] = \binom{N}{2} \tag{2.36}
\]

\[
N_2 = \binom{N}{3} \cdot [1 \cdot 1 \cdot 7 - 3 \cdot 3 + 3 \cdot 1] = \binom{N}{3} \tag{2.37}
\]

\[
N_3 = \binom{N}{4} \cdot [1 \cdot 1 \cdot 15 - 4 \cdot 7 + 6 \cdot 3 - 4] = \binom{N}{4} \tag{2.38}
\]

\[
\vdots \tag{2.39}
\]

\[
N_i = \binom{N}{i+1} \tag{2.40}
\]

Equation (2.40) can be proved by induction. For the other two trivial binary MDS codes,

- if \(d_{free} = N\)

  \[
  N_e = N \tag{2.41}
  \]

  \[
  N_{i>0} = 0 \tag{2.42}
  \]

- if \(d_{free} = 2\)

  \[
  N_e = N \tag{2.43}
  \]

  \[
  N_{i>0} = 0 \tag{2.44}
  \]

\[
\text{MDS Generator/Parity: } d_{free} \text{ will thus be maximum if the rank of the generator matrix } G \text{ is } k, \text{ or equivalently the generator has largest number of linearly independent rows/columns. Otherwise, a linear combination of rows could replace a row with Hamming weight } d_{free} - 1 \text{ and a } k^{th} \text{ new row to increase to full rank and then necessarily must have at least 1 more 1 in this new row that could not be the combination with weight } d_{free} - 1 \text{ generator just formed. The generator’s null space forms the \textbf{“dual” code’s} generator matrix (often called } H \text{ or the } \textbf{parity matrix}). \text{ So } G \text{ spans the code space with rank } P_G, \text{ and } H \text{ spans the null space with rank } P_H. \text{ For these vectors of length } n,}
\]

\[
P_G + P_H = n. \tag{2.48}
\]

\textsuperscript{17}This \(H\) repeats notation for the filter AWGN \(H\), but both are in wide use in communication theory for these two very different meanings. The reader should be able to infer which is the subject from the context.
When $G$ is full rank ($\mathcal{P}_G = k$), it can be premultiplied by any rank-$k$ $k \times k$ scrambling matrix\(^{18}\) $A$ without changing the set of all linear combinations of the rows of $G$, i.e., the code (and thus all its codewords) stay(s) the same. Again only when $G$ is full rank, there is at least one such $A$ that will form a linear combination of codewords that has (exactly the minimum) $d_{\text{free}}$ ones in a row because there is a codeword that is at that minimum distance from the all zeros codeword in all linear codes, let us call that codeword $v'$. It is clear then that

$$v'H^* = 0$$ (2.49)

because all codewords will be orthogonal to the null space. So, the minimum number of columns that may be linearly dependent is $d_{\text{free}} - 1 \leq \rho_H$. Further, since $\mathcal{P}_H = n - \mathcal{P}_G = n - k$ (from Equation (2.48)) only for the full rank generator case, then

$$d_{\text{free}} \leq \mathcal{P}_H + 1 = n - k + 1$$ (2.50)

which is the Singleton Bound. When $d_{\text{free}} - 1 = \rho_H$, the bound is achieved and the code is MDS.

**Error-Probability Analysis:** For linear binary codes, the codeword-error probability can be well approximated\(^{19}\) through the BSC’s error-probability bound as

$$P_e \approx N_e \cdot \left( \frac{d_{\text{free}}}{[d_{\text{free}}/2]} \right) \cdot p^{d_{\text{free}}/2} \cdot (1 - p)^{n-d_{\text{free}}/2}$$ (2.51)

$$< N_e \cdot [4 \cdot p \cdot (1 - p)]^{\lceil \frac{d_{\text{free}}}{2} \rceil}.$$ (2.52)

Correspondingly, the average number of bit-errors per codeword, based on the ML detector, is (with $N_b$ as Chapter 1’s (Section 1.3.2.4) average total bit errors per error event)

$$\bar{P}_b \approx \frac{N_b}{b} \cdot [4 \cdot p \cdot (1 - p)]^{\lceil \frac{d_{\text{free}}}{2} \rceil}.$$ (2.53)

For the BSC, like the AWGN, It is possible to design an ML decoder for subsymbol or individual bit errors. The corresponding marginal distributions can be computed and used. Chapter 8 illustrates in detail these marginal-distribution calculations. Binary codes find use with 2PAM (or BPSK) directly on the AWGN channel with $1 \rightarrow +1$ and $0 \rightarrow -1$, and then the minimum distance for the binary-coded AWGN is

$$d_{\text{min,BSC}}^2 = 4 \cdot d_{\text{free}}.$$ (2.54)

Using (2.54) when $P_e < 10^{-3}$, hard decoding is about 3 dB worse than soft decoding by comparing the two error bounds in (2.52) and Chapter 1’s NNUB $P_e < N_e \cdot Q \left( \frac{d_{\text{min}}}{2b} \right)$. This 3dB approximation arises by comparing the 2-PAM AWGN NNUB error expression as

$$P_e = N_e \cdot Q(\sqrt{d_{\text{free}} \cdot SNR}) \propto e^{-d_{\text{free}} \cdot \frac{SNR}{2}}$$ (2.55)

with the hard-coded

$$p^{d_{\text{free}}/2} = \left[ Q(\sqrt{SNR}) \right]^{d_{\text{free}}/2} \propto e^{-d_{\text{free}} \cdot \frac{SNR}{2}},$$ (2.56)

so a factor of 2 worse or 3 dB.

From the ball-packing perspective but more precisely and returning to an upper bound, the Hamming Bound\(^{20}\) ensures sufficient ball space around each symbol for a given $n$ and radius $d_{\text{free}}/2$ and provides an alternative upper bound:

---

\(^{18}\) Scrambling here means 1-to-1 nonsingular matrix transformation.

\(^{19}\) Realizing that $n - d_{\text{free}}/2 \geq d_{\text{free}}/2$ or equivalently the free distance cannot exceed the codeword length, and that $\left( \frac{d_{\text{free}}}{2b} \right) \leq 4^{\left[ \frac{d_{\text{free}}}{2} \right]}$.

Lemma 2.2.3 (The Hamming Bound) With \( t \triangleq \left\lfloor \frac{d_{\text{free}}}{2} - 1 \right\rfloor \), the number of code-words is upper bounded as
\[
M(n, d_{\text{free}}) \leq \frac{2^n}{\sum_{j=0}^{t} \binom{n}{j}} = \frac{\# \text{ of total binary vectors}}{\# \text{ of binary vectors in ball of radius } t}.
\]
(2.57)

Proof: The proof is in the rightmost expression in (2.57). QED.

Problem 2.6 explores the Hamming bound’s tightness with respect to the Singleton Bound for binary codes. Codes achieving Hamming’s “ball-packing” bound are known as perfect codes. Hamming found a family of such Hamming Codes characterized by integer \( i \) such that \( n = 2^i, \) \( k = n - i + 1 \) and \( d_{\text{free}} = 3. \) Some other perfect codes are also known with larger free distances, see for instance Golay Codes.21

2.2.2.2 Block Codes for the DMC

Block-code design sometimes presumes inner hard decoding that results in a DMC, of which the simplest example is the BSC. For the BSC, these binary block codes are applicable with \( N = 1, \) \( n \triangleq N = N, \) and \( k \triangleq b. \) Some block codes will be based on a larger constellation size \(|C| > 2\), which presumes a DMC of that same large size (so the DMC has \(|C|\) inputs and \(|C|\) outputs). Often, \(|C| = 256\) (that is a byte-wise, or octet-wise, code). The arithmetic for corresponding encoders and decoders is most often modulo the size of the constellation \(|C|\), using Galois Fields as in Appendix E and Chapter 10, and often linear in the field \( \mathbb{GF}(C) \). For these codes also, \( N = 1 \) and \( N = N. \) Instead of a lower-case \( k, \) these codes use upper-case \( K \) to enumerate the number of information-bearing subsymbols in a codeword of \( N = N \) subsymbols. The number of parity subsymbols becomes \( P = N - K = \rho \cdot \log_2(|C|) \) and \( R \triangleq r = K / N \leq 1. \) With linear arithmetic over a finite field equal to the constellation size, the symbols in the input constellation will have uniform distribution in each dimension, so the sphere-packing concept with Gaussian distributions is lost with inner-channel hard decoding.23 Well-designed codes also will be able to correct \( d_{\text{free}}/2 \) subsymbols that have errors.

Block code can be characterized when not binary by \( q \) where \(|C| = q. \) Often the field of arithmetic is \( \mathbb{GF}(q) \). The base \( q \) is often a power of two, but need not be and could be any integer for which a finite field exists (which means a positive-integer power of a prime integer or products of such numbers). The free distance remains the number of minimum subsymbol positions in which the closest two codewords differ, but the subsymbol difference has values \( 0, ..., (q - 1); \) \( t \) remains \( t = \left\lfloor \frac{d_{\text{free}}}{2} \right\rfloor. \)

Discrete Memoryless Channel (DMC): Figure 2.10 shows a symmetric discrete memoryless channel where \( p(< 1/2) \) is the probability of an individual subsymbol error. Figure 2.10 also provides values for \( p_{y/x}. \) An ML decoder for such a channel will, similar to the BSC, select that codeword that has minimum Hamming distance from the received \( q \)-ary output vector.24 Equation (2.31) is again the corresponding minimized codeword-error probability, which has approximation (2.52).

---

22Discrete Memoryless Channel that first appears in Chapter 1, Section 4.
23Instead a uniform distribution on \( \mathbb{GF}(q) \) holds with \( N \rightarrow \infty. \)
24This is most easily seen by viewing the ML selection as maximizing the probability of being correct, which follows the BSC and the probability of being in error is also the same as it does not matter which erroneous value of the subsymbol emanated from the channel, an error is still an error.
The bit-error probability again follows the same expressions with perhaps a more complicated mapping of bit errors on average to the symbol values that will correspond on average typically to $\frac{K}{N} \cdot \log_2(q)$ bits/subsymbol. A direct mapping of $q$-ary subsymbols to a real/complex constellation is possible, but there is not a simple relationship of euclidian minimum distance on the AWGN to the Hamming distance when $q > 2$. Subsection 2.1.4 described BICM as a method to map good interleaved binary codes to multi-level constellations, which often works well. Block codes with $q > 2$ find instead good use in transmission designs where a lower level detector creates an inner channel that produces bits, bytes or other bit-groupings to an outer system. In effect, the inner subcode creates a DMC and incurs losses that may not be recoverable before passing its decoder results to an outer code’s decoder. Nonetheless, the outer block code can still significantly improve performance through the use of non-zero parity, which in turn means the inner code’s rate $r_{in}$ reduces by the outer code’s rate $r_{out}$ so $r = r_{in} \cdot r_{out}$. Such information reduction may be acceptable in exchange for further improvement in symbol and/or subsymbol error probability.

**General Bound:** For the DMC, the earlier GV Bound, Hamming, and Singleton bounds then generalize as in Lemma 2.2.4

**Lemma 2.2.4 (Bound Extension to $q \geq 2$)** The GV, Hamming, and Singleton bounds generalize for arbitrary $q$, and again $t = \left\lfloor \frac{d_{free} - 1}{2} \right\rfloor$, to

$$\sum_{j=0}^{d_{free} - 1} \binom{N}{j} \cdot (q - 1)^j \leq M_q(N, d_{free}) \leq \min \left\{ \begin{array}{l} q^N \sum_{j=0}^{t} \binom{N}{j} \cdot (q - 1)^j \\ \text{Gilbert-Varshamov} \\ \text{Hamming} \\ \text{Singleton} \end{array} \right\}. \quad (2.58)$$

In the case where $q > 2$, then the Hamming Bound may not be tightest of the two upper bounds. Non-trival MDS Codes exist that achieve the Singleton Bound, but not the Hamming Bound. Linear binary block codes exist for $q > 2$ that meet the Singleton Bound and are non-trivial MDS codes. MDS codes’ generator matrices have any (all) sets of $K = N - d_{free} + 1$ linearly independent (over the $q$-ary
field \( \mathbb{GF}^q \) rows (and conversely if all sets of \( K \times K \) sub matrices are non-singular, the code is MDS). Again, for the binary case, only the all zeros and all ones codewords for \( M = 2 \), the rate-1 uncoded, and the simple parity-check code at any \( n \) are MDS. Examples of MDS codes in \( \mathbb{GF}^q \) include Reed Solomon, BCH, and others. For MDS codes, the nearest neighbor counts in (2.34) at all distances generalize to (again with \( N_0 \triangleq N_e \) while \( N_i \) is the \( i \)th nearest neighbor) as

\[
N_i = \binom{N}{d_{\text{free}} + i} \cdot \sum_{j=0}^{i} (-1)^j \cdot \binom{d_{\text{free}} + i}{j} \cdot [q^{i+j-1}-1].
\]  

(2.59)

Negative values in (2.34) or (2.59) for \( N_i \) mean that an MDS code does not exist. There are a few codes that achieve the Hamming Bound and are known as perfect codes (binary or not) in addition to the Hamming Codes and Golay Codes for certain special values of \( d_{\text{free}}, k \) (or \( K \)), and \( n \) (or \( N \)). Problem 2.6 further explores these bounds. Problem 2.7 investigates nearest neighbor counts and provides a simple matlab program to compute Equation (2.59) when the code exists.

### 2.2.3 Erasure Decoding

Figure 2.11’s Binary Erasure Channel (BEC) has a 3-level output, with the extra output corresponding to “indeterminate” decisions by an inner channel. This is a crude form of “soft information” that is intermediate to hard decoding and full soft decoding. The new output is an erasure that means the channel output is uncertain. The BEC’s erasures positions are known. The BEC’s decoder treats the non-erased bits as correct (with probability one, or effectively very close to one). Clearly a decoder can decide without error the correct input if the number of erasures \( n_e \) is less than \( d_{\text{free}} \), which in the linear-code case means the number of erasures is less than the minimum weight of any non-zero codeword. The error probability is then

\[
P_e \approx N_e \cdot p^{d_{\text{free}}} \cdot (1-p)^{n-d_{\text{free}}}. \]

(2.60)

This error probability is intermediate to those of full soft decoding (AWGN) and hard decoding (BSC). Comparison of (2.60) to (2.53) shows this directly. In terms of hard decoding’s 3 dB loss with respect to soft decoding, some is recovered if the \( d_{\text{free}} \) is odd while all recovered if \( d_{\text{free}} \) is even. This calculation is optimistic since the BEC does not admit any possibility of \( 1 \rightarrow 0 \) or vice versa, and that might be a questionable assumption in practice. For instance, with \( d_{\text{free}} = 5 \), then the BEC system recovers 2 dB, or is only 1 dB worse than soft decoding. DMC’s with certain types of codes (known as circulant or the most famous version called a “Reed Solomon” code) can provide also erasure indications.
rather than just detect inputs. These cyclic codes can also correct up to $d_{free}$ “erased” subsymbol positions in error if the channel is a BEC and subsymbols containing erased bits are marked as erasures. Decoders may use detected codeword errors (so that $y$ is not equal to a possible $x$) to mark erasures. This operation is a cyclic-redundancy check or CRC\textsuperscript{25}, which is typically used with higher-level protocols (like the internet’s transmission-control protocol, TCP) to both mark and then retransmit packets that contain detected bit errors.

2.2.4 Trellis and Convolutional Codes

Trellis (or “sliding block”) codes have infinite $\tilde{N}$ and thus also have infinite $N$, but $\tilde{N} = \frac{N}{2}$ is always finite. The reason for the name “trellis” becomes apparent in Example 2.2.1, where a time-indexed description of a state-transition diagram known as a trellis diagram appears. This diagram, while similar to diagrams for the BSC and DMC conditional probabilities has different interpretation that becomes evident shortly. In many cases, infinite $\tilde{N}$ may just mean arbitrarily large corresponding to the codewords’ extension to the end of transmission or to the last subsymbol prior to transmission cessation. Codewords for trellis codes are thus possibly infinite-length subsymbol sequences.

Figure 2.12 depicts state-dependent subsymbol generation in a sequential encoder, where the encoder’s next subsymbol output depends on a state (history) of prior message transmissions. Thus, the current or next $\tilde{N}$-dimensional subsymbol depends not only on the current $\tilde{b}$ input bits but also on a state representing all previous subsymbol transmissions’ history. Trellis codes attempt to gain the benefit of very large block length $N$ while maintaining a finite (per unit time) complexity through the sequential encoder’s use. $\tilde{b}$ bits enter the sequential encoder at each subsymbol instant and combine with $\nu$ state-characterizing bits that enumerate one of $2^\nu$ historical states in which the sequential encoder can be at subsymbol time/dimension $k$. These $\tilde{b} + \nu$ bits determine the transmitted subsymbol value. The $\nu$ bits are not redundant bits, but together with the $\tilde{b} + \tilde{\rho}$ output bits may characterize the $\tilde{N}$-dimensional subsymbols $\tilde{x}_k$. The $\tilde{N}$-dimensional output subsymbol depends on both the $\tilde{b}$ input bits and the $\nu$ state bits through the function

$$\tilde{x}_k = f(m_k, s_k) \ . \quad (2.61)$$

The function $f$ can include any mapping to real (or complex) symbols and can also include interleaving between a binary code and a $\tilde{b} + \tilde{\rho}$ bit-to-subsymbol mapping table, such as in BICM, to a constellation $C$. The next state also depends on those same input bits and current state through the function

$$s_{k+1} = g(m_k, s_k) \ . \quad (2.62)$$

While the consequent decoder complexity grows with time, it is finite per subsymbol and can be recursively computed with a finite number of operations for each and every subsymbol period in a receiver without loss of decoder optimality (see Viterbi Decoding in Chapter 8).

For the AWGN channel, Chapter 10 reviews several popular trellis (or Ungerboeck\textsuperscript{26}) and coset (or Forney’s coset\textsuperscript{27}) codes that are based on a subsymbol lattice’s partitioning into congruent subsets with good inter-subsymbol distances. For the trellis code, $b$ is also infinite, while $\tilde{b}$ and $\tilde{\rho}$ are finite. The constellation size is also finite $|C|$ (although the constellation may be large and approximate samples from a Gaussian distribution).

\textsuperscript{25}CRC is a simple MDS block code that detects if the channel output is a codeword or not.

\textsuperscript{26}After Austrian/ Swiss Gottfried Ungerboeck, 1940 –, who developed the codes at IBM Research-Zurich.

\textsuperscript{27}After American G. David Forney, Jr., 1940 –, who developed these codes while at Motorola-Codex.
2.2.4.1 Binary Convolutional Codes

Convolutional codes are a special, usually binary only, case of trellis codes and again designed for hard-decoded AWGN’s with $|C| = 2^{\tilde{b} + \tilde{\rho}}$, but can also find use with soft decoding. The convolutional-code constellation is a concatenation of $\tilde{N}$ binary two-level constellations corresponding to $\tilde{N}$ successive BSC uses. In the convolutional-code case, $k = \tilde{b}$ and $n = \tilde{N}$ (which is different than binary block codes, one of the reasons this text tries to use consistent notation, avoiding $k$, $n$, and $K$ largely). The redundancy/subsymbol is $\tilde{\rho} = n - k$ bits. The overall redundancy is infinite and usually not discussed. The rate $r = k/n = \tilde{b} = \frac{\tilde{b}}{\tilde{N}} \leq 1$. The convolutional code’s redundancy and rate add to one

$$\frac{\tilde{\rho}}{n} + \frac{r}{\tilde{N}} = 1 .$$  (2.63)

Convolutional codes can use higher-order addition in $\mathbb{GF}^q$, but this is rare in transmission designs. More commonly, 2PAM (or BPSK) may be used with the convolutional code and soft-decoding where ML’s overall squared-distance metric is actually the sum over all $\tilde{N} \to \infty$ instances of $\tilde{N}$ summed-square scalar distances for each dimension within a subsymbol. Thus, convolutional codes can be hard or soft decoded when used in this way (as can also be binary block codes if they are similarly mapped into 2PAM or BPSK constellations).
EXAMPLE 2.2.1 (4-state convolutional code) Figure 2.13 illustrates a simple convolutional code with one input bit and two output bits for every subsymbol. There are 4 states corresponding to the 4 possible combinations of the most recent 2 input bits, or the state. There is one redundant bit per subsymbol, or \( \tilde{\rho} = 1 \), and per dimension if each output bit is viewed as a dimension, then \( \tilde{b} = \frac{1}{2} \) and \( r + \tilde{\rho} = 1 \). The trellis appears on Figure 2.13’s right.

Subsection 2.2.1’s intermediate (to hard/soft decoding) LLRs can also be used to decode convolutional codes. Iterative decoding methods also apply to these systems as per Chapter 8, which goes beyond the simple BICM concept of Subsection 2.1.4. Two convolutional codes’ encoder-output bits can be separated by interleaving. The inner or first decoder has not interleaving itself and simply decodes and creates \( LLR_i \) for every input bit’s values through calculation of the á posteriori probability distribution for the given channel outputs and then taking their ratio’s logarithm. However, rather than make an immediate bit decision based on each \( LLR_i \)’s polarity, these \( LLR_i \)’s are subsequently de-interleaved and create á priori distributions for the outer decoder, which in turn computes its á posteriori distribution for bits. Rather than even then make a decision, the new resulting \( LLR_i \)’s can be re-interleaved as used as á priori distributions for the inner code. This iterative decoding when used in such a form is often called turbo coding (more appropriately turbo decoding perhaps) in analogy with a turbo-charged engine that profits from its own exhaust. To avoid biasing any particular bit, only influence from other bits within it’s same codeword are used in the á priori distribution in each instance of decoding. Such systems can approach maximum performance with relatively simple inner and outer codes. Chapter 11 discusses further turbo codes as interleaved convolutional codes. Convolutional codes often terminate at the end of a packet or large symbol/codeword, and thus essentially become block codes but are designed with the sequential encoder and corresponding decoder in mind.

Figure 2.14 decomposes in a simple way the basic types of codes. Many codes use concatenations of these code types in various ways as in Chapters 8 - 11.
Various concatenations possible also

Figure 2.14: Simple code-type decomposition based on $N < \infty$ and $N \to \infty$ as well as decoder and type.
2.3 Mutual Information and Channel Capacity

This section formalizes Subsection 2.1.3 asymptotic results and Section 2.2’s ML decoders/channels to find the upper asymptotic bounds for coding performance, particularly the mutual information for any channel and input probability distributions. The mutual information, $I$, will bound reliably (arbitrarily small $P_e$) achievable bits/symbol. The mutual information’s maximized-input-distribution form is the channel capacity, $C = \max_p[I]$. Subsection 2.3.1 introduces the concept of entropy that measures the probability distribution’s information content and also helps characterize code redundancy when applied to the distribution of a randomly selected code’s subsymbols. Subsection 2.3.2 introduces mutual information by building on Subsection 2.3.1’s entropy concept through conditional entropy and also notes its direct tie to minimum-mean-square-error (MMSE) estimation (See Appendix D) for AWGN channels. Subsection 2.3.3 builds on Subsection 2.1.3’s LLN to introduce Asymptotic Equipartition (AEP), which formalizes mutual-information and capacity bounds to apply to any channel (not just the AWGN, from which these are obtained easily from MMSE theory). Indeed, the AEP generalizes the concept of sphere-packing and mean-square-error minimization with Gaussian distributions to any distribution. Subsection 2.3.4 formalizes the channel capacity, while Subsection 2.3.5 introduces the water-filling energy distribution that is often important to capacity. Subsection 2.3.6 makes a few more random-code-design remarks.

2.3.1 Entropy and Data Symbols’ Distribution

A random data symbol’s (or message’s) entropy $H_x$ is determined by the symbols’ (codewords’) probability distribution $p_x$. Entropy generalizes the concept of constellation size when the constellation’s subsymbols need not be equally probable.

**Definition 2.3.1 [Entropy]** The entropy of a set of random symbol vectors $x \in \{x_0, ..., x_{M-1}\}$ with stationary probability distribution $p_x(i) i = 0, ..., M - 1$ is:

$$H_x \triangleq - \sum_{i=0}^{M-1} p_x(i) \cdot \log_2[p_x(i)] \text{ (bits/symbol - } x)$$

$$= E\{\log_2[1/p_x]\} \geq 0 \text{ . } (2.64)$$

If the distribution is over a subsymbol’s $\tilde{x}$ constellation, then the entropy is for that subsymbol, written $H_{\tilde{x}}$. If the distribution is over the codewords (or symbols) $x$, then the entropy is over the codeword-probability distribution as in Definition 2.3.1. Often the codeword distribution is uniform, but it need not be, particularly in concatenated systems. Entropy generalizes the concept of constellation size $|C|$ to the situation where the symbol values are not necessarily equally likely. The entropy can be particularly interesting for codeword subsymbols $\tilde{x}$. Whether subsymbol, symbol, or code, the entropy can be normalized to a per-dimensional basis by dividing by the corresponding number of real dimensions. It is not necessarily true that the entropy of a single marginal distribution of $p_x$, $H_x$, is equal to $H_{\tilde{x}}$, and the difference is later shown to be the code’s redundancy. The discrete-distribution-based received symbol’s entropy $H_y$ has similar definition, $E\{\log_2[1/p_y]\}$. A discrete uniform distribution has the largest entropy, or information, in comparison to all other discrete distributions.

**Lemma 2.3.1 [Maximum Entropy for Discrete Distributions]**

The uniform probability distribution $p_x = \frac{1}{M}$ maximizes the entropy.
Proof: The Lagrangian for maximizing the (concave) Equation (2.65)'s entropy subject to the probability distribution’s unit-sum constraint is
\[ L = - \sum_{i=1}^{M} p_i \cdot \log_2(p_i) + \lambda \cdot \left[ \sum_{i=1}^{M} p_i - 1 \right], \] (2.66)
which has derivative for \( p_i \) equal to
\[ -\frac{\ln(p_i)}{\ln 2} - \frac{1}{\ln 2} + \lambda. \] (2.67)
Zeroing this derivative and solving provides \( p_i = \text{constant} \forall i \) and thus \( p_i = \frac{1}{M} \) to meet the unit-sum constraint. The maximum entropy is correspondingly then equal to the bits per symbol or
\[ \max_{p_x} H_x = \log_2(M) = b. \] (2.68)
QED.

Maximum Entropy: Lemma 2.3.1 says that equally likely messages carry the largest information if there is a fixed number of possible messages. By contrast, a deterministic quantity \( (p_x(i) = 1 \text{ for one value of } i \text{ and } p_x(j) = 0 \forall j \neq i) \) has no information, and thus zero entropy, \( H_x = 0 \). A communication system for such a known message would carry essentially no information because sending the same message repeatedly, with no chance of ever changing, makes the communication system unnecessary. With instead multiple messages possible, for instance a uniform distribution on 4 discrete values, the entropy is largest at \( H_x = 2 \) bits/symbol. This is the same as \( b \) for 4-level PAM in 1D or 4SQ in 2D with uniform input distributions. In Chapter 1, the message sets always had uniform distributions, so that the entropy was \( b \), the base-2 log of the number of messages. In general, \( H_x \leq \log_2(|C|) \), where \(|C|\) is the number of subsymbol values in the discrete distribution for the subsymbol’s constellation \( C \).

Entropy is tacitly a function of the dimensionality, particularly when constellations are designed for the AWGN. A 32 CR constellation with equally likely symbols has \( H_x = 5 = b \) bits/two-dimensional-symbol. However, in one dimension, this constellation has the one-dimensional subsymbol values \( \pm 5 \) each occurring with probability \( \frac{1}{8} \) while the subsymbols \( \pm 1 \) or \( \pm 3 \) each occur with probability \( \frac{3}{16} \), leading to a one-dimensional entropy
\[ H_x = \frac{2}{8} \cdot \log_2(8) + \frac{6}{8} \cdot \log_2(\frac{16}{3}) = 2.56 > b = 2.5 = \bar{b} \text{ bits/dim}. \] (2.69)
The one-dimensional entropy \( H_x \) is less than the maximum entropy of \( \log_2(6) = 2.585 \) bits/dimension for 6 equally likely symbols in one dimension (which would correspond in two dimensions to a 36-point square constellation). The two-dimensional maximum entropy for 32CR has \( H_x=2.5 \) bits/dimension, which is also less than the maximum entropy in one dimension for this constellation, \( \log_2(6) \). Essentially, the 32CR constellation when viewed in one dimension exhibits redundancy (or coding) if all 32 two-dimensional symbols are equally likely and 5 information bits are transmitted. The redundancy per dimension was earlier computed as \( \bar{\rho} = 0.085 \), also here noted to be equal to \( \max H_x - \bar{b} \) since \( \max \bar{H}_x = \log_2 |C| \). Problem 2.2 explores entropy for the D4 lattice of Example 2.3 for its 4, 2, and 1 dimensional entropies.

Entropy and Redundancy: Lemma 2.3.1 bounds the subsymbol entropy \( \bar{H}_x \) as
\[ \bar{H}_x \leq \log_2 |C| = \bar{b} + \bar{\rho}, \] (2.70)
with equality to \( \bar{b} \) if and only if \( p_x \) is uniform. Equivalently, an uncoded system has zero redundancy and entropy thus equal to \( \bar{b} \).

A good \( N \)-dimensional code with symbol/codeword \( x \) over the full \( N \) dimensions will thus typically have \( H_x = \log_2(M) = b \) and a uniform distribution of symbols, or codewords. For example for the
AWGN, this uniform distribution is roughly within Subsection 2.1.3’s “hypersphere” (which was the circle in two dimensions that 32CR better approximates than 32SQ). As the dimensionality grows, typically the number of possible values for a good code’s subsymbols increases, and these values within the subsymbol constellations can have unequal likelihood (probability); but the code usually has equally likely \( N \)-dimensional symbols. Large or tending-to-infinite constellations with many points suggests entropy’s generalization to continuous distributions, often called the “differential entropy”:

\[
\mathcal{H}_x \triangleq -\int_{-\infty}^{\infty} p_x(u) \cdot \log_2 [p_x(u)] \cdot du .
\]  

(2.71)

**Definition 2.3.2** [Differential Entropy] The continuous random variable’s distribution \( p_x(u) \) has differential entropy defined as

Distinguishing Entropy and Differential Entropy: This text adds the qualifying adjective “differential,” and uses a script \( \mathcal{H} \) notation, because the entropy \( \mathcal{H}_x \)’s limiting-summation value in (2.64) that corresponds to infinitesimally small probability-distribution partitioning will be infinite\(^{28}\). But, \( \mathcal{H} < \infty \); differential entropy is different than entropy. For instance, the differential entropy for a random uniform \( x \) on any interval of length \( d < 1 \) produces \( \mathcal{H} < 0 \), while a discrete uniform distribution with large \( |C| \) approximating a continuous uniform would have very large entropy, \( \mathcal{H}_x \rightarrow \infty \). Further, a scaled version of a continuously distributed random variable \( a \cdot x \) has differential entropy \( \mathcal{H}_x = \mathcal{H}_x + \log_2 |a| \), which generalizes to \( \mathcal{H}_{A\cdot x} = \mathcal{H}_x + \log_2 |A| \) when \( A \) is a square nonsingular matrix; but for a discrete distribution \( \mathcal{H}_{ax} = \mathcal{H}_x \) and \( \mathcal{H}_{A\cdot x} = \mathcal{H}_x \). Even more generally if \( x \) is mapped into some \( x' \), when \( f \) is invertible and one-to-one, such that \( x' = f(x) \), then \( \mathcal{H}_{x'} = \mathcal{H}_x + \int p(x) \log_2 \left( |\partial f / \partial x| \right) \cdot dx \) where \( |\partial f / \partial x| \) is the Jacobian determinant, again showing a difference for the continuous \( \mathcal{H} \) case where instead such an invertible functional mapping would not change entropy \( \mathcal{H} \). This text’s developments will progress shortly (and elsewhere where entropy \( \mathcal{H} \) or differential entropy \( \mathcal{H}_x \) appear) to quantities that correspond to the difference between between \( \mathcal{H}_x \) and another entropy that will however have the same constant offset. The common offset in the \( \mathcal{H} \)’s then disappears when the 2 quantities are subtracted, whether or not the distribution is discrete or continuous. In that differential-based context, somewhat exclusive to data transmission, the interchangeable use of entropy and differential entropy is acceptable. Indeed, many textbooks use \( H \) and \( \mathcal{H} \) interchangeably without this explanation, but ultimately only have the same differential-based quantities used further in these textbooks’ ensuing results.

Large constellations on AWGN channels have an average-energy constraint in practice. The constraint on average energy over \( N \) dimensions is analogous to the interior of hypersphere - that is \( E \left( \int |x|^2 dx \right) \leq \mathcal{E}_x \), and codewords roughly lie within a sphere. As \( N \rightarrow \infty \), the LLN reveals codewords on the hypersphere’s surface have probability \( 1 \). A maximum entropy over these \( N \) dimensions would then have a uniform codeword distribution within such a hypersphere. The limiting one-dimensional (or finite \( N \)-dimensional) distribution of such a uniform spherical distribution is thus Gaussian in any subsymbol:

\[
\text{Lemma 2.3.2} \quad \text{[Maximum Entropy for Continuous Distributions]} \quad \text{The Gaussian probability distribution} \quad p_x(u) = \frac{1}{\sqrt{2\pi x}} \cdot e^{-\frac{u^2}{2x}} \quad \text{maximizes the subsymbol differential entropy} \quad \text{(and thus per real dimension)} \quad \text{for any given} \quad \mathcal{E}_x \geq 0.
\]

\(^{28}\)The correct limiting value essentially would have a \( p_x \cdot dx \) ALSO inside the \( \log_2(\cdot) \) term in order to retain its proper probability-mass-function properties, and differential entropy omits the extra \( dx \) term.
Proof: Let \( g_x(u) \) denote the Gaussian distribution, then
\[
\log_2 g_x(u) = -\log_2 \left( \sqrt{2\pi e} \right) - \left( \frac{u}{\sqrt{2\sigma^2}} \right)^2 \cdot (\ln(2))^{-1} .
\] (2.72)

For any other distribution \( p_x(u) \) with mean zero and the same given variance \( \overline{E}_n \)
\[
-\int_{-\infty}^{\infty} p_x(u) \cdot \log_2 (g_x(u)) \cdot du = \log_2 \left( \sqrt{2\pi e} \right) + \frac{1}{2\ln(2)}
\] (2.73)
\[
= \mathcal{H}(g_x)
\] (2.74)
which depends only on \( \overline{E}_n \). Then, letting the distribution for \( x \) be an argument for the entropy,
\[
\mathcal{H}_x(g_x) - \mathcal{H}_x(p_x) = -\int_{-\infty}^{\infty} g_x(u) \cdot \log_2(g_x(u)) \cdot du + \int_{-\infty}^{\infty} p_x(u) \cdot \log_2(p_x(u)) \cdot du
\]
now using (2.73) to obtain
\[
= -\int_{-\infty}^{\infty} p_x(u) \cdot \log_2(g_x(u)) \cdot du + \int_{-\infty}^{\infty} p_x(u) \cdot \log_2(p_x(u)) \cdot du
\] (2.75)
\[
= -\int_{-\infty}^{\infty} p_x(u) \cdot \log_2 \left( \frac{g_x(u)}{p_x(u)} \right) \cdot du
\] (2.76)
\[
\geq \frac{1}{\ln 2} \int_{-\infty}^{\infty} p_x(u) \cdot \left( 1 - \frac{g_x(u)}{p_x(u)} \right) \cdot du
\] (2.77)
\[
\geq \frac{1}{\ln 2} (1-1) = 0 ,
\] (2.78)

or\(^{29}\)
\[
\mathcal{H}_x(g_x) \geq \mathcal{H}_x(p_x) .
\] (2.79)

This same proof can be repeated for a complex Gaussian (or two-dimensional Gaussian with equal and uncorrelated real and imaginary parts) for \( \tilde{x} \in \mathbb{C} \), and thereby extends to any \( \tilde{N} < \infty . \) QED.

Thus, the marginal distributions for dimensions (or subsymbols) of an average-energy-limited uniform distribution over the hyper-sphere tends to a Gaussian in any and all dimensions as \( N \to \infty . \) This result holds for complex Gaussian (or two-dimensional Gaussian, which is the same) in two dimensions when \( \tilde{N} = 2 \) and for any finite \( \tilde{N} \) as an \( \tilde{N} \)-dimensional Gaussian subsymbol distribution. Good codes on the AWGN essentially attempt to approximate this uniform codeword distribution over an infinite-dimensional hypersphere, or equivalently have random subsymbols with sample values that appear to be selected from Gaussian distributions.

The differential entropy of a (real) Gaussian random variable with variance \( \sigma^2 \) is
\[
\mathcal{H}_x = \frac{1}{2} \log_2 \left( 2\pi e\sigma^2 \right) \ \text{bits/dimension} .
\] (2.80)

A complex Gaussian variable with variance \( \sigma^2 \) has differential entropy
\[
\mathcal{H}_x = \log_2 \left( \pi e\sigma^2 \right) \ \text{bits/subsymbol} .
\] (2.81)
Careful examination justifiably reveals that these two quantities in Equations (2.80) and (2.81) are the same, and differ only semantically.
2.3.2 Joint and Conditional Entropy, and Mutual Information

The joint entropy (or differential joint entropy) simply follows by writing some elements of the vector random variables as one sub-vector and the rest as the remaining sub-vector, so

$$\mathcal{H}_{x,y} = -E_{x,y} \left[ \log_2 (p_{x,y}) \right]$$  \hspace{1cm} (2.82)

$$\mathcal{H}_{x,y} = -E_{x,y} \left[ \log_2 (p_{x,y}) \right] ,$$  \hspace{1cm} (2.83)

where again differential entropy follows with integrals replacing summations on whatever components have continuous distributions. (This again creates potential mixed offsets when $x$ discrete and $y$ is continuous, but again those offsets disappear when differences between like mixtures are the only quantities of interest, as is again the case here.)

The conditional entropy of one random variable given another is

$$\mathcal{H}_{x|y} \triangleq \sum_{v} \sum_{i=0}^{M-1} p_x(i) \cdot p_{y|x}(v, i) \cdot \log_2 \left( \frac{1}{p_{y|x}(v, i)} \right) = E_{x,y} \left\{ \log_2 \left[ 1/p_{x|y} \right] \right\}$$  \hspace{1cm} (2.84)

$$H_{y|x} \triangleq \sum_{v} \sum_{i=0}^{M-1} p_x(i) \cdot p_{y|x}(v, i) \cdot \log_2 \left( \frac{1}{p_{y|x}(v, i)} \right) = E_{x,y} \left\{ \log_2 \left[ 1/p_{y|x} \right] \right\} ,$$  \hspace{1cm} (2.85)

with integrals replacing summations when random variables/vectors have continuous distributions, and $\mathcal{H} \rightarrow \mathcal{H}$. The conditional-entropy definition averages over all joint possibilities in some given input and channel-output distributions. This conditional entropy is thus a function of both $p_{y|x}$ and $p_x$. More generally, conditional entropy measures a random variable’s average residual information given the value of another random variable.

The chain rule of probability has a direct translation into an entropy chain rule (also for differential entropy) as

$$\mathcal{H}_{x,y} = \mathcal{H}_x + \mathcal{H}_{y|x} = \mathcal{H}_y + \mathcal{H}_{x|y}$$  \hspace{1cm} (2.86)

$$\mathcal{H}_{x,y} = \mathcal{H}_x + \mathcal{H}_{y|x} = \mathcal{H}_y + \mathcal{H}_{x|y} ,$$  \hspace{1cm} (2.87)

which also notes the symmetry in interchangeability of order in the chain rule. The joint entropy does not depend on the random-vector components’ order. From (2.86), $\mathcal{H}_{x|y} \leq \mathcal{H}_x$ with equality only when $x$ and $y$ are independent. This same chain rule applies equally to differential entropy.

For a communication channel characterized by $p_{y|x}$, the conditional entropy, $\mathcal{H}_{y|x}$, is basically the noise’s information/symbol. If the conditional distribution is Gaussian, as is the case with the AWGN, the conditional differential entropy of a Gaussian scalar $x$ in bits/symbol becomes

$$\mathcal{H}_{x|y} = \left\{ \begin{array}{ll}
\frac{1}{2} \log_2 \left( 2\pi e \sigma_{mmse}^2 \right) & x \in \mathbb{R} \\
\log_2 \left( \pi e \sigma_{mmse}^2 \right) & x \in \mathbb{C} 
\end{array} \right.$$

Equation 2.88’s “MMSE” arises from the minimum-mean-square-error (MMSE) estimation of $x$, given $y$ (See Appendix D). Thus, the conditional entropy measures the information remaining after the effect of $y$ has been removed through MMSE estimation with Gaussian processes. The conditional entropy thus measures information that a receiver might not be able to estimate about $x$.

The following extends Lemma 2.3.2 to any autocorrelation matrix between two scalars or vectors\(^30\)

$$R_{[xy]} \triangleq E [x^* y] = \begin{bmatrix} R_{xx} & R_{xy} \\ R_{xy} & R_{yy} \end{bmatrix} .$$  \hspace{1cm} (2.89)

The vector MMSE is the determinant of this matrix

$$|R_{ee}| = |R_{xx} - R_{xy} R_{yy}^{-1} R_{yx^*}| ,$$  \hspace{1cm} (2.90)

as also in Appendix D.

\(^30\)Note the brackets in the subscript of the defined matrix $R_{[xy]}$ to distinguish it from one of its components the cross-correlation between $x$ and $y$ as $R_{xy}$.
Lemma 2.3.3 [Maximum Entropy for Multiple Vector Continuous Distributions] The Gaussian probability distribution

\[ p_{xy}(u, v) = \frac{1}{\sqrt{\pi |R_{xy}|^{1/N}}} e^{-\left\{ (u^* v^*) - R_{xy}^{-1} \right\}} \]  

(2.91)

maximizes the subsymbol differential entropy for any given positive semi-definite \( R_{xy} \).

Proof: Let \( g_{xy}(u, v) \) denote the complex-vector Gaussian distribution, then

\[ \log_2 g_{xy}(u, v) = -\log_2 \left( \sqrt{\pi |R_{xy}|^{1/N}} \right) - \left( (u^* v^*) - R_{xy}^{-1} \right) (\ln(2))^{-1} \]

For any other distribution \( p_{xy}(u, v) \) with mean zero and the same given autocorrelation \( R_{xy} \)

\[ -\int_{-\infty}^{\infty} p_{xy}(u, v) \cdot \log_2 (g_{xy}(u, v)) \cdot d[uv] = \log_2 \left( \sqrt{\pi |R_{xy}|^{1/N}} \right) + \frac{1}{\ln(2)} \]

(2.92)

which depends only on \( R_{xy} \). Then, letting the joint distribution for \( [xy] \) be an argument for the entropy,

\[ \mathcal{H}_{xy}(g_{xy}) - \mathcal{H}_{xy}(p_{xy}) = -\int_{-\infty}^{\infty} g_{xy}(u, v) \cdot \log_2 (g_{xy}(u, v)) \cdot d[uv] \]

\[ + \int_{-\infty}^{\infty} p_{xy}(u, v) \cdot \log_2 (g_{xy}(u, v)) \cdot d[uv] \]

now using (2.92) to obtain

\[ = -\int_{-\infty}^{\infty} p_{xy}(u, v) \cdot \log_2 (g_{xy}(u, v)) \cdot d[uv] \]

\[ + \int_{-\infty}^{\infty} p_{xy}(u, v) \cdot \log_2 (p_{xy}(u, v)) \cdot du \]

\[ = -\int_{-\infty}^{\infty} p_{xy}(u, v) \cdot \log_2 \left( \frac{g_{xy}(u, v)}{p_{xy}(u, v)} \right) \cdot d[uv] \]

\[ \geq \frac{1}{\ln 2} \int_{-\infty}^{\infty} p_{xy}(u, v) \cdot \left( 1 - \frac{g_{xy}(u, v)}{p_{xy}(u, v)} \right) \cdot d[uv] \]

\[ \geq \frac{1}{\ln 2} (1 - 1) = 0 \]

or \(^{31}\)

\[ \mathcal{H}_{x}(g_{xy}(u, v)) \geq \mathcal{H}_{x}(p_{xy}(u, v)) \]  

(2.95)

with equality only when \( p_{xy}(u, v) = g_{xy}(u, v) \). This joint-entropy proof is very useful in showing that all linear multi-user systems with additive Gaussian noise have best performance with Gaussian input distributions in Section 2.6. QED.

While the source entropy meaningfully measures data transmitted, the channel-output entropy has an extra constituent component that is caused by the randomness of noise (or other distorting effects) that is not helpful for transmission reliability. Given that a receiver observes only the output \( y \), this extra
noise information detracts from the possible input data rate. Thus, the entropy difference $\mathcal{H}_y - \mathcal{H}_{y|x}$ measures the recoverable-data-rate component of $x$ in the channel output. Equivalently, if $\mathcal{H}_x$ is the input entropy, then $\mathcal{H}_{x|y}$ measures the distortion in $x$ that is not estimable from $y$. The same statements apply equally well to differential entropy. This information is called the mutual information.

**Definition 2.3.3 [Mutual Information]** The mutual information for any $N$-dimensional signal set with probability distribution $p_x(i) i = 0,...,M - 1$, and a corresponding channel description $p_{y|x}(v,i)$, is:

$$I_{x,y} \triangleq \mathcal{H}_x - \mathcal{H}_{x|y} = E \left\{ \log_2 \left[ \frac{p_{x,y}(v,i)}{p_x(v) \cdot p_y} \right] \right\} \geq 0 , \quad (2.96)$$

and remains valid with differential entropy $\mathcal{H}$ replacing $\mathcal{H}$ when $x$ and/or $y$ has continuous distribution. Mutual information’s non-negativity follows trivially from the chain rule and $\mathcal{H}_{x|y} \leq \mathcal{H}_x$.

Mutual information is the quantity where the difference between two differential entropies $\mathcal{H}_x$ and $\mathcal{H}_{x|y}$ causes their equal-but-infinite-difference from entropies $\mathcal{H}_x$ and $\mathcal{H}_{x|y}$ to cancel. Thus, mutual information works in both cases and is the same quantity, whether based on entropy and/or differential entropy. The identity,

$$I_{x,y} = \mathcal{H}_y - \mathcal{H}_{y|x} = \mathcal{H}_x - \mathcal{H}_{x|y} , \quad (2.97)$$

easily follows from transposing the “symmetric” $x$ and $y$ as follows:

$$I_{x,y} \triangleq \sum_{v} \sum_{i=0}^{M-1} p_x(i) \cdot p_{y|x}(v,i) \cdot \log_2 \left[ \frac{p_{y|x}(v,i)}{\sum_{m=0}^{M-1} p_{y|x}(v,m) \cdot p_x(m)} \right] \text{ bits/symbol} \quad (2.98)$$

$$= E \log_2 \left[ \frac{p_{y|x}}{p_y} \right] \quad (2.99)$$

$$= E \log_2 \left[ \frac{p_{y,x}}{p_x \cdot p_y} \right] \quad (2.100)$$

$$= E \log_2 \left[ \frac{p_{x,y}}{p_x} \right] , \quad (2.101)$$

In the case of a continuous distribution on $y$ and/or $x$, the summation(s) is (are) replaced by the appropriate integral(s).

Mutual information also has a Chain Rule that follows direction from entropy’s chain rule:

**Lemma 2.3.4 [Chain Rule]** The mutual information satisfies (for any order):

$$I(x;y) = \sum_{i=1}^{U} I(x_i; y|x_{i-1}) . \quad (2.102)$$

**Proof:** From entropy chain rule:

$$\mathcal{H}(x) = \sum_{i=1}^{U} \mathcal{H}(x_i; x_{i-1}) \text{ and } \quad (2.103)$$

$$\mathcal{H}(x/y) = \sum_{i=1}^{U} \mathcal{H}(x_i/y, x_{i-1}) . \quad (2.104)$$

Subtraction of (2.104) from (2.103) completes the proof, which could be repeated with $\mathcal{H} \rightarrow \mathcal{H}$. QED.
The chain rule would also apply to any subset of \( x \)'s dimensions, which has implications in Section 2.6’s multi-user analysis. The following lemma also has wide use throughout this text, so while effectively trivial, it’s statement is for completeness. It is similar to the reversibility theorem.

\[
I(u; v) = I(x; y) \quad (2.105)
\]

**Proof:** Both entropy (including differential) and mutual information are functions only of their own (joint) probability distributions evaluated and averaged. An invertible transformation \( u = f(x) \ni x = f^{-1}(u) \) and \( v = g(y) \ni x = g^{-1}(v) \) for a discrete distribution will simply have the same probabilities at each of the evaluated points as well the logarithms of those points. This applies also to the joint distribution and conditional distributions. Thus, the lemma follows trivially because the sums evaluated are the same. For a continuous distribution, the integrals are evaluated over \( df = du \) and \( dv \) respectively but reduce to the original integrals under change of variable substitution. QED.

### Lemma 2.3.5 [Preservation of Information under invertible transformation]

If either or both of \( x \) and \( y \) undergo invertible transformations to \( u \) and \( v \) respectively, the mutual information (and the corresponding entropies) do not change, equivalently

\[
I(u; v) = I(x; y)
\]

2.3.3 Asymptotic Equipartition

Asymptotic Equipartition (AEP) uses Subsection 2.1.3’s law of large numbers (LLN) concept to measure – on average – codes that are designed by randomly picking subsymbols from the same distribution to construct codewords. Such “random coding” concepts were used by Shannon in his original capacity development because he did not know the actual codes, but rather determined that such codes exist because random selection of codes, on average, would produce at least one code that achieves his fundamental capacity limit. Subsection 2.1’s AWGN development found sphere-packing codes to be one such capacity-achieving code, which would be among the set found by picking codewords from a uniform distribution with constant average energy constraint as \( N \to \infty \). AEP generalizes this random-coding concept distribution, and of course maximization over all allowed input distribution choices then produces a more general capacity result.

**Random Code Design:** Random code design computes a codeword-distribution’s entropy per sub-symbol through random and independent selections from the same stationary subsymbol distribution \( p_{\tilde{x}} \) \( N \) times for each codeword in the code. In effect, random-code design views these randomly selected codewords’ subsymbols themselves as a message source with probability distribution \( p_{\tilde{x}} \). First, using \( \log_2(p_{\tilde{x}}) = \log_2 \left( \prod_{n=1}^{N} p_{\tilde{x}_n} \right) \), the randomly constructed vector-codeword symbols has per-symbol entropy

\[
\tilde{H}_{\tilde{x}} = -\frac{1}{N} \cdot E \left[ \log_2(p_{\tilde{x}}) \right] \quad (2.106)
\]

\[
= -\frac{1}{N} \sum_{n=1}^{N} E \left[ \log_2(p_{\tilde{x}_n}) \right] \quad (2.107)
\]

which is also \( E \left[ \log_2(p_{\tilde{x}_n}^{-1}) \right] = \tilde{H}_{\tilde{x}} \) for a stationary distribution \( p_{\tilde{x}_n} = p_{\tilde{x}} \forall n = 1, ..., N \). In random coding, \( \rho_{\tilde{x}} = 0 \), but \( \rho_{\tilde{x}} > 0 \). When \( p_{\tilde{x}} \) is continuous, then \( \tilde{H} \) replaces \( \tilde{H} \) in (2.106) and (2.107), as well as throughout all AEP discussion.
Entropy as a Sample Average: The entropy \( \hat{H}_x \) is the average number of bits/dimension for such a randomly constructed code (equivalently that represents a source with this same distribution) \( x \); or stated equivalently, the random-code design process selects, on average, a number of codewords \( x \in C_x \) that is \( 2^{N \cdot \hat{H}_x} \). The probability distribution \( p_{x_n} \) is also a function of the random variable \( \tilde{x}_n \) itself for application of the LLN (See Theorem 2.1.1), and thus \( p_{\tilde{x}_n} \to p(\tilde{x}_n) \) is a random variable also in this view: Equation (2.107) then also has interpretation as the log-inverse-probability distribution’s sample-average value for random variable \( 1/p(\tilde{x}_n) \) over each and every codeword in the random code-design process:

\[
\hat{H}_x = -\frac{1}{N} \sum_{n=1}^{N} \log_2 [p(\tilde{x}_n)] = -\frac{1}{N} \log_2 [p(x)] .
\] (2.108)

(This same statement applies also to joint entropy and conditional entropy with sample averages over both \( \tilde{x} \) and \( \tilde{y} \), and thus also to their difference, the mutual information.) The LLN then states that the asymptotic \( (N \to \infty) \) sample-average value \( \hat{H}_x \) will almost surely have a value equal to \( \hat{H}_x \) for such a codeword, and any other codeword that might not have value equal to \( \hat{H}_x \) will be asymptotically negligible with arbitrarily small probability. Thus from this LLN viewpoint for a code \( x \in C_x \) with randomly selected subsymbols \( \tilde{x} \) from \( p_{\tilde{x}} \):

\[
\hat{H}_x \to \tilde{H}_x \quad \quad (2.109)
\]

\[
H_{\tilde{x}/\tilde{y}} \to H_{x/y} \quad \quad (2.110)
\]

\[
I(\tilde{x};\tilde{y}) \to I(x;y) \quad \quad (2.111)
\]

Because \( \log_2(p_x) \) is constant over all codewords asymptotically, it thus has uniform distribution asymptotically.

Typical Sets: There will be a set of sample codeword values that have the entropy value \( \hat{H}_x \) with probability one (strong form of LLN), which is called the typical set as below in Definition 2.3.4. The sample-average-codeword values outside this set are atypical and have infinitesimally small probability of occurring (weak form of LLN). Again, these statements all presume \( N \to \infty \). This concept generalizes Section 2.1’s sampling from a uniform distribution with a hyper-sphere boundary that lead to a Gaussian marginal distribution in each dimension. This AEP generalization does not require a energy constraint and works for any stationary distribution over the codeword length. More subtly, and the crucial point in AEP, through the subsymbols’ symmetry in sampling from the same distribution repeatedly, is that each of these increasingly large codewords must asymptotically have the same uniform-distribution probability \( 2^{-N \cdot \Pi_x} \), or equivalently the size of the typical set approaches \( 2^N \cdot \Pi_x \) symbols with uniform distribution.\(^{32}\) More formally, with \( N \)-dimensional subsymbols:

**Definition 2.3.4** [typical set] A typical set of length-\( N \)-dimensional codewords for a stationary (over dimensional index \( n \)) code, with subsymbol samples \( \tilde{x}_n \), \( n = 1, ..., N \) and entropy \( \hat{H}_x \) or with \( \hat{H}_x \) replacing \( \hat{H}_x \) for continuously distributed \( x \), is

\[
A^c_x(\epsilon) \triangleq \left\{ x = [\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_N] \bigg| 2^{-N \cdot \hat{H}_x - \epsilon} \leq p(\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_N) \leq 2^{-N \cdot \hat{H}_x + \epsilon} \right\} \quad (2.112)
\]

for any \( \epsilon > 0 \).

The AEP Lemma follows from the above discussion:

\(^{32}\)Any subsymbols that might be more/less likely than uniform would under dimensional rearrangement of the stationary/repeated distribution would look like other subsymbols, thus imposingly heuristically the uniformity of the asymptotic distribution on the typical set.
Lemma 2.3.6 [AEP Lemma] For a typical set with $N \to \infty$, the following are true:

- $\Pr\{A_N^c(x)\} \to 1$
- for any codeword $x \in A_N^c$, $\Pr\{x\} \to 2^{-N \cdot \mathcal{H}_x}$

**Proof:** These statements are equivalent to Theorem 2.1.1’s LLN with $z = \log_2(p_x)$, and the distribution’s stationarity, along with the independence of each subsymbol’s sample selection therefrom, which over all constitutes the proof. QED.

Relation to Redundancy and Subsymbol Constellation: The form of the AEP is such that $\mathcal{H}_x$ (or $\mathcal{H}_x$) takes the place of $\log_2 |C|$ in specific codes. In effect, for either continuous or discrete input distribution,

$$\mathcal{H}_x = \hat{b} + \hat{\rho} \quad .$$

(2.113)

The information content of a subsymbol exceeds the message information by the code’s redundancy/subsymbol under AEP. The AEP typical set $A_N^c(x)$ implies a maximum codeword entropy for equally likely codewords. The designer can choose the $\hat{b} \leq \mathcal{H}_x$, and thus $b = N \cdot \hat{b}$, which then determines the redundancy of a randomly constructed code (asymptotically with very long blocklength).

Maximum Rate of Random Codes: The AEP generalizes the AWGN concept of “energy-constrained uniform distribution in many dimensions is Gaussian in a finite or single-dimensional slice.” The conditional probability $p_y|x$ characterizes the general channel, and the input distribution $p_x$ need no longer have an input-energy constraint (just more generally any input constraints that restrict $p_x$). A code selects codewords through $M$ random, length-$\hat{N}$, independent $\hat{x}$ sample sets; each codeword has $\hat{N}$ selections from the subsymbol channel-input distribution $p_{\hat{x}}$. Further many such random codes exist, and the entire set (with enough selections) of codes contains some good ones with probability approaching 1. Indeed any random code as $N \to \infty$ has probability approaching 1 of being good. The AEP ensures that no matter the stationary input distribution, such a set of codes all have their codewords dominated in uniform probability by the typical set. The AEP does not find the good codes, but guarantees they exist. Further since $I(x;y) = \mathcal{H}_x - \mathcal{H}_x/y$, then

$$\frac{|A_N^c(x)|}{|A_N^c(x)/y|} \to 2^{I(x;y)} \quad ,$$

(2.114)

suggestion that $2^I$ essentially counts $A_N^c(x/y)$ subsets in $A_N^c(x)$ and so may limit generally the rate of selection of the number of codewords asymptotically.
Venn Diagram Decision Regions: Figure 2.15 illustrates the AEP typical-set concept with a Venn Diagram where the rectangle represents the space of all possible random-code codewords. The circle within the Venn diagram represents the typical set. This typical set dominates the probability as the sequences get longer and longer, and each sequence within this set becomes equally likely. Each codeword has asymptotically an equal probability of $2^{-N \cdot H_x}$. Figure 2.16 further illustrates that on average the conditional-typical-set size for $x$ given $y$ is smaller (because the entropy is less and in a codeword subset). Subsymbols within the subset can be viewed as indistinguishable from one another given $y$. Thus a MAP detector likely errs if this $y$-specific subset contains two possible codewords, because it resolves such situation by randomly selecting one codeword from this subset. The AEP-Venn-diagram-illustrated MAP-detector decision region generalizes the AWGN ML detector’s picking the closest symbol that asymptotically would lie in a hypersphere surrounding the symbol. Good code design avoids more than one codeword in any independent set of size $2^{-N \cdot H_x}$. This good code design generalizes the AWGN good-code concept of trying to pack as many spheres into a larger sphere while keeping them sufficiently separated. Since there are $2^{-N \cdot H_x}$ codewords as $N$ gets large, a good code design selects codewords from this larger set so that there is only one in each possible conditional-typical subset of size $2^{-N \cdot H_x}$. Thus with this good code, the largest number of distinguishable codewords is

$$M = \frac{2^{-N \cdot H_x}}{2^{-N \cdot H(x|y)}} = 2^{N \cdot I(x;y)} \quad (2.115)$$

With the random code-selection exercise, there will be, on average, such a code within the set (of codes analyzed on average) as long as $M$ is not larger than (2.115). There may be more than 1 such good code; indeed with $N \to \infty$, it is intuitively evident there are an infinite number of such good codes that correspond to different orderings of, and/or mappings to, the conditional-typical sets. The MAP decoder has codeword-error probability that tends to zero as long as no two codewords are from the same set, $A_N^x(x/y)$, meaning reliable decoding is possible. Further, even a slightly higher data rate ensures more than 1 typical-set codeword in at least one conditional-typical set occurs on average with probability one, so that $P_e$ would deteriorate rapidly when data rate exceeds the mutual information. The AEP design process essentially creates the (infinite-length) decision regions on the channel output $y$ as the mapping of each $y$ to the single $x \in A_N^x(x/y)$. Thus, as $N \to \infty$, $I(x;y)$ represents the maximum number of bits per subsymbol that can be reliably transmitted over the communication channel. That
is,
\[ b \leq I(x; y) \quad , \quad (2.116) \]

or on a per-subsymbol basis
\[ \tilde{b} < I(x; y) \quad ; . \quad (2.117) \]

### 2.3.4 The Channel Capacity Theorem

The channel capacity measures the maximum data rate that can be reliably transmitted over any given channel. The mutual information essentially measures this data rate (in bits/symbol) for any given input distribution, presuming that good code design ensures that the typical sets \( A_{\epsilon}^N(x) \) each contain one and only one codeword \( x \). Generally, \( I(x; y) \leq \mathcal{H}(x) \) and less by at least the minimum redundancy \( \rho \). Different input distributions can lead to different mutual information. The best input then has \( p_x \) that maximizes the mutual information, a concept first introduced by Shannon in his 1948 paper:

**Definition 2.3.5 (Channel Capacity)** The channel capacity in bits/subsymbol for a channel described by \( p_y|x \) is defined by
\[
\tilde{C} \triangleq \max_{p_x} I(x; y) \quad \text{bits/subsymbol} . \quad (2.118)
\]

It is sometimes convenient to normalize \( \tilde{C} \) to one dimension, by dividing \( \tilde{C} \) by \( \tilde{N} \),
\[
\bar{C} \triangleq \frac{\tilde{C}}{\tilde{N}} = \frac{C}{N} , \quad (2.119)
\]

which is in bits/dimension. The capacity is measured in bits/dimension as \( C \), bits/subsymbol \( \tilde{C} \), or bits per codeword/symbol as \( \bar{C} \). The calculation of (2.118) can be difficult in some cases, and may
require numerical techniques to approximate the $C$ value. Any input constraints in the input $x$ vectors’ choice affects the capacity value. Given the mutual information definition, Shannon’s [1948 BSTJ] famed channel coding theorem is:

**Theorem 2.3.1 (The Channel Capacity Theorem)** Given a channel with capacity $C = \max_{p(x)} I(x; y)$, then there exists a code with $b < C$ such that $P_e \leq \delta$ for any $\delta > 0$. Further, if $b > C$, then $P_e \geq$ positive constant, which is typically large even for $b$ slightly greater than $C$.

**proof:** See the discussion surrounding Equations (2.115) and (2.116) and the AEP discussion preceding this theorem.

This theorem’s desired interpretation is that reliable transmission can only be achieved when $b < C$, or equivalently $b < C$. The capacity for the complex AWGN is probably the best known and most studied. It is determined from

\[
\tilde{C}_{awgn} = I(x; y) = H_y - H_{y|x} = H_y - \log_2(\pi e \sigma_y^2) \tag{2.120}
\]

which, because $\sigma_y^2$ is constant, is maximum when $H_{y|x}$ corresponds to Gaussian $y$ as per Lemma 2.3.2. This implies that $x$ also has a Gaussian distribution, because $x = y - n$ and linear combinations of Gaussian random processes are also Gaussian. Further then because input $x$ and noise $n$ are independent, the well-known AWGN capacity formula arises

\[
\tilde{C}_{awgn} = I(x; y) = H_y + n - \log_2(\pi e \sigma_y^2) = \log_2 \left( \frac{\tilde{\mathcal{E}}_x + \sigma^2_x}{\sigma^2_y} \right) = \frac{1}{2} \log_2(1 + \text{SNR}) \tag{2.121}
\]

**Some Gaussian Entropy Basics:** Given two complex jointly Gaussian random-vector subsymbols, $\tilde{x}$ ($L_x \times 1$) and $\tilde{y}$ ($L_y \times 1$), the conditional probability density $p_{\tilde{x}|\tilde{y}}$ also has a Gaussian probability density with mean $E[\tilde{x}|\tilde{y}] = R_{\tilde{x}\tilde{y}} \cdot R_{\tilde{y}^{-1}} \cdot \tilde{y}$ equal to the MMSE estimate of $\tilde{x}$ given $\tilde{y}$ as in Appendix D. (This result is easily proved as an exercise by simply taking the ratio of $p_{\tilde{x}|\tilde{y}}$ to $p_{\tilde{y}}$, which are both Gaussian with the general $(L_x, L_y)$-complex-dimensional $(2\cdot(L_x, L_y)$ real dimensions) form. The (complex) Gaussian vector distribution is $1/(\pi e^{L_x} |R_{\tilde{x}\tilde{x}}|) \cdot e^{-(x-u_x)^* R_{\tilde{x}\tilde{x}}^{-1} (x-u_x^*)}$, where $R_{\tilde{x}\tilde{x}}$ is the subsymbol autocovariance matrix and $u_x$ is the mean.) The differential entropy of a complex Gaussian subsymbol is

\[
H_{\tilde{x}|\tilde{y}} = \log_2((\pi e)^{L_x} |R_{\tilde{x}\tilde{x}}|) \tag{2.122}
\]

where $R_{\tilde{x}\tilde{x}}$ is the autocorrelation matrix of zero-mean subsymbol $\tilde{x}$, and $|R_{\tilde{x}\tilde{x}}|$ is its determinant (singular random-vector processes have $H = 0$). Thus, the conditional entropy of $\tilde{x}$ given $\tilde{y}$ is

\[
H_{\tilde{x}|\tilde{y}} = \log_2((\pi e)^{L_x} |R_{\tilde{x}\tilde{x}}|) \tag{2.123}
\]

where $R_{\tilde{x}|\tilde{y}} = R_{\tilde{x}\tilde{y}} - R_{\tilde{x}\tilde{y}} \cdot R_{\tilde{y}\tilde{y}}^{-1} \cdot R_{\tilde{y}\tilde{x}}$ is the autocorrelation matrix of the vector MMSE estimating $\tilde{x}$ from $\tilde{y}$, because $p_{\tilde{x}|\tilde{y}}$ also has a Gaussian distribution. For a scalar white Gaussian $x$, $H_{x} = \frac{1}{2} \cdot \log_2(2\pi e \mathcal{E}_x)$ whether $x$ is real or complex. Further, $\mathcal{E}_{\tilde{x}} = \frac{1}{2} \log_2((\pi e)^{L_x} |R_{\tilde{x}\tilde{x}}|^{1/L_x})$. If $x = x$, i.e., a scalar, then $R_{xx} = 2\mathcal{E}_{x}$ in the entropy formula with $L_x = 1$ and all per-dimensional results are consistent. 33

33For the interested in alternative expressions (that provide the same entropy): If $\tilde{x}$ is real, then $H_{\tilde{x}} = \frac{1}{2} \log_2 \left( (2\pi e)^{L_x} |R_{\tilde{x}\tilde{x}}| \right)$ or

\[
\mathcal{E}_{\tilde{x}} = \frac{1}{2L_x} \log_2 \left( (2\pi e)^{L_x} |R_{\tilde{x}\tilde{x}}| \right) \tag{2.124}
\]
For the scalar complex Gaussian case with \( L_x = 1 \), and \( \text{SNR}_{\text{mmse}} \triangleq \frac{\tilde{e}_x}{\sigma^2_{\text{mmse}}} \),

\[
\tilde{C}_{\text{awgn}} = \log_2(1 + \text{SNR}) = \max (\mathcal{H}_y - \mathcal{H}_{y/x})
\]

\[
= \mathcal{H}_x - \mathcal{H}_{x/y}
\]

\[
= \log_2(\pi \sigma^2_e) - \log_2(\pi \sigma^2_{\text{mmse}})
\]

\[
= \log_2(1 + \text{SNR}_{\text{unbiased}})
\]

\[
= \log_2(1 + \text{SNR})
\]  

(2.131)

The term \( \text{SNR}_{\text{unbiased}} \triangleq \text{SNR}_{\text{mmse}} - 1 \) is addressed more completely in Chapter 3. Following also from the vector AWGN form

\[
y = Hx + n
\]

(2.134)

the mutual information is maximum when \( x \) is Gaussian; because if \( \mathcal{H}_x \) is Gaussian, then \( y \) is Gaussian following the expression in (2.120). Also, the minimum MSE error vector \( \hat{e} = \tilde{x} - R_{\tilde{x}\tilde{y}} \cdot R_{\tilde{y}\tilde{y}}^{-1} \cdot y \) is then also Gaussian (linear combinations of Gaussians are Gaussian). The mutual-information formula is then (noting also the symmetry that \( I = \mathcal{H}_y - \mathcal{H}_{y/x} = \mathcal{H}_x - \mathcal{H}_{x/y} \))

\[
\tilde{I}_{\text{vector awgn}} = \frac{1}{N} \cdot \log_2 \left( \frac{|R_{\tilde{y}\tilde{y}}|}{|R_{\tilde{n}\tilde{n}}|} \right) = \frac{1}{L_x} \cdot \log_2 \left( \frac{|R_{\tilde{n}\tilde{n}} + H R_{\tilde{x}\tilde{x}} H^*|}{|R_{\tilde{n}\tilde{n}}|} \right) = \frac{1}{L_x} \cdot \log_2 \left( \frac{|R_{\tilde{x}\tilde{x}}|}{|R_{\tilde{e}\tilde{e}}|} \right)
\]

(2.135)

The symbol \( \tilde{I}_{\text{vector awgn}} \) later replaces \( \tilde{I}_{\text{vector awgn}} \) in Subsection 2.3.5 that finds the best Gaussian distribution’s autocorrelation matrix. This leads to formal statement that Gaussian inputs are optimum on all Gaussian channels.

**Theorem 2.3.2 (Gaussian Inputs Maximize AWGN Channel Mutual Information)**

*For any linear AWGN, as in (2.134), with specific autocorrelation \( R_{\tilde{x}\tilde{x}} \), a joint Gaussian distribution on all \( \tilde{x} \)’s elements maximizes the mutual information \( I(\tilde{x}; \tilde{y}) \).*

**Proof:** See the preceding paragraph. QED.

This theorem has broad application in multi-user channels in Sections 2.6 - 2.9.

### 2.3.4.1 Uniform Continuous Subsymbols relative to Gaussian

For the uniform distribution \( p_{\tilde{x}} \) in two real dimensions,

\[
\mathcal{H}_{\tilde{x}} = \log_2(6 \cdot \tilde{e}_x) \quad \text{uniform distribution}
\]

(2.136)

in comparison to the larger value of \( \mathcal{H}_{\tilde{x}} = \log_2(\pi \cdot e \cdot \tilde{e}_x) \) from Equation (2.88). For reasonable SNR, the MMSE estimate of \( \tilde{x} \) given \( y \) will be largely determined by the noise (which is Gaussian and independent

\[
= \frac{1}{2} \log_2 \left[ (2L_x \pi e) \cdot |R_{\tilde{x}\tilde{x}}|^{1/L_x} \right]
\]

(2.125)

which checks with one-dimensional formula. If \( \tilde{x} \) is complex, then \( \mathcal{H}_{\tilde{x}} = \log_2 \left( |\pi e| L_x \cdot |R_{\tilde{x}\tilde{x}}| \right) \) or

\[
\mathcal{H}_{\tilde{x}} = \frac{1}{2L_x} \log_2 \left( (\pi e)^{L_x} \cdot (2L_x) |R_{\tilde{x}\tilde{x}}| \right)
\]

(2.126)

\[
= \frac{1}{2L_x} \log_2 \left( (\pi e)^{L_x} \cdot (2L_x)^{L_x} |R_{\tilde{x}\tilde{x}}| \right)
\]

(2.127)

\[
= \frac{1}{2} \log_2 \left( (2L_x \pi e) \cdot |R_{\tilde{x}\tilde{x}}|^{1/L_x} \right)
\]

(2.128)

which also checks with the one dimensional formulae. When a complex vector is modeled as a doubly-dimensional real vector, one can see the two formulae for normalized entropy are the same as they should be.

\[34\] The zero-mean Gaussian depends only on its autocorrelation matrix.

243
of the input distribution. Thus, $\mathcal{H}_{X,Y}$ is largely then independent of the input distribution. In this case, the loss in mutual information that accrues to using square constellations (with random coding from a uniform square constellation instead of the differential-entropy maximizing Gaussian) is

$$\text{loss SQ constellation} = \log_2(\pi \cdot e) - \log_2(6) \approx 0.5 \text{ bit/subsymbol} .$$  

(2.137)

This loss is equivalent to the maximum shaping gain over a rectangular constellation, which is known from Chapter 1 to be $\gamma_{s,max} = \pi \cdot e/6$ (1.5 dB). So

$$\mathcal{I}_{awgn,SQ} = C_{awgn} - 0.5\text{ bits/subsymbol} .$$  

(2.138)

A more exact bound and calculation has been derived by Forney [4] for any subsymbol region defined by a lattice Voronoi region, and reduces at reasonable SNR of practical interest to (2.138). As most wireless and wireline systems today for practical reasons use square constellations, this expression and loss of 0.5 bit/subsymbol is a useful rule to the practicing engineer.

### 2.3.5 Simple Water-filling for the matrix AWGN

![Figure 2.17: Simple Vector Coding “Parallelization” of a Filter AWGN with SVD.](image)

The tilde notation on sub symbols is dropped in this subsection. Optimization over input distribution $p_x$ can consider different $R_{xx}$ for (2.134)’s channel, subject to constant input energy $\text{trace}\{R_{xx}\} \leq \mathcal{E}_x$. The matrix $R_{nn}^{-1/2} \cdot H$ replaces $H$ so that the noise-whitened channel equivalent has unit-variance noise in all (complex) dimensions. The $L_y \times L_x$ matrix $H$ then has a singular value decomposition

$$H = F \cdot \Lambda \cdot M^* ,$$  

(2.139)

where the $L_y \times L_y$ matrix $F$ is unitary, $FF^* = F^*F = I$, as is also the $L_x \times L_x$ matrix $M$, $MM^* = M^*M = I$ and $\Lambda$ is a real (even if $H$ is complex) diagonal-plus-zeros matrix. Figure 2.17 illustrates a vector coded approach to the filter AWGN that uses the lossless energy-preserving invertible $M^*$ matrix as a transmit filter and the white-noise-preserving invertible $F$ matrix as a receiver with no loss in mutual information. The zero rows are below a diagonal $\Lambda$ when $L_y > L_x$ and to the right of diagonal $\Lambda$ when $L_x > L_y$. The nonzero entries of $\Lambda$ are the singular values. Transform of the input to $x' = M \cdot x$ does not change the energy, while transformation of the output to $y' = F y$ leaves $L^* = \min(L_x, L_y)$ nontrivial set of parallel independent “subchannel”

$$y'_\ell = \lambda_\ell \cdot x'_\ell + n'_\ell \quad \ell = 1, \ldots, L^* ,$$  

(2.140)

where the transformed noise is also white and unit variance on all dimensions so statistically equivalent to the original noise. This channel has the same mutual information as the original channel (because of

---

35See Matlab “svd” command.
the 1-to-1 transformations) and also will have the same performance as the original channel with an ML
detector under the reversibility theorem. Then

\[ I(x; y) = \sum_{\ell=1}^{L^*} I(x_\ell; y_\ell) = \sum_{\ell=1}^{L^*} \log_2(1 + \mathcal{E}_\ell \cdot \lambda^2_\ell) \quad . \]  

(2.141)

Maximization of this subject to the energy constraint yields

\[ \mathcal{E}_\ell + \frac{1}{\lambda^2_\ell} = K \quad \wedge \quad \mathcal{E}_\ell \geq 0 \quad , \]

(2.142)

where \( K \) is a constant independent of \( n \) that is determined by the energy constraint (negative energies
are discarded and the \( L^* \) value reduces by the number of such zeroed-energy subchannels). Equation
(2.142) is the water-filling energy distribution (studied in depth in Chapter 4). \( K \) derives from the
total energy constraint \( E = \text{trace}\{R_{xx}\} = \sum_{n=1}^{L^*} \mathcal{E}_n \), and then each energy is found from (2.142) before
inserting the energies into (2.141) to compute the water-filling capacity.

### 2.3.6 Decoding with Good-Code Gaussian inputs

The ML decoder for the Gaussian channel has subtleties that this section addresses. For any code
\( C_x \), the ML decoder will select the sequence of subsymbols, equivalently the codeword, that minimizes
\( \|y - Hx\|^2 \) when the noise has \( R_{nn} = I \) finds and selects the codeword such that \( Hx \) is closest to \( y \) - that
is the sequence of subsymbols. While the number of bits/subsymbol is finite, \( b < \infty \), a particular code’s
constellation also is finite \( |C| < \infty \), but the average over the ensemble of codes has an infinite number of
subsymbol possibilities selected from a continuous Gaussian distribution. Thus the ML detector selects
the codeword \( x \) over all its subsymbols that minimizes

\[ \min_{\{x_k\}} \sum_{k=-\infty}^{\infty} ||\hat{y}_k - H \cdot \hat{x}_k||^2 \neq \sum_{k} ||\hat{n}_k||^2 \quad . \]  

(2.143)

(2.143)’s inequality follows because with an effectively continuous distribution for \( \hat{x}_k \), it is possible to
trade some noise reduction for subsymbol miss in the limiting sum. Via the AEP, the infinite-dimensional
codewords have a uniform distribution with probability 1. Thus, the ML and the MAP detector provide
the same result \( x \) and minimize codeword-error probability; and indeed this best codeword has individual
subsymbols \( \hat{x}_k \) over all time \( k \).

**Theorem 2.3.3 [AWGN Channel Convergence Theorem]** The asymptotic optimum
MAP/ML detector for the matrix (any \( H \)) AWGN also minimizes mean square error. **Proof:**
For any codeword \( x \), the sample MMSE estimate

\[ \hat{x}_{\text{mmse}} = W \cdot \hat{y} = E[\hat{x}/\hat{y}] \quad . \]

(2.146)

But since \( \hat{x} \) and \( \hat{y} \) are on average with random coding jointly (as well as individually)
Gaussian, the estimate \( \hat{x}_{\text{mmse}} = E[\hat{x}/\hat{y}] \) also must maximize the à posteriori distribution
\( p(\hat{x}/\hat{y}) \) for any given channel output \( \hat{y} \)’s sequence of subsymbol values \( \hat{y} \). Thus, on average,
the MMSE estimate (sequence of subsymbol estimates) is MAP (and consequently ML) for
the codeword \( x \). QED.
The convergence theorem holds for any input autocorrelation matrix $R_{\tilde{x}\tilde{x}}$, noise correlation matrix $R_{\tilde{n}\tilde{n}}$ and channel $H$. It means that on average over all codes, the MMSE receiver for each subsymbol produces a sequence that is the MAP sequence with probability 1. It does not mean the sequence decoder search can be avoided for any particular code over which random coding arguments apply. However, it does explain the last inequality in (2.143).

For the channel, there are two subsymbol-based MMSE estimation problems:

$$\tilde{x} \rightarrow \tilde{y} \ (H) \quad (2.147)$$
$$\tilde{y} \rightarrow \tilde{x} \ (W) \ , \quad (2.148)$$

which have close relationship, and indeed both have the same mutual information

$$I(\tilde{x};\tilde{y}) = \mathcal{H}_{\tilde{y}} - \mathcal{H}_{\tilde{y}/\tilde{x}}$$
$$= \log_2 \left( \frac{|R_{\tilde{y}\tilde{y}}|}{|R_{\tilde{n}\tilde{n}}|} \right) \text{ bits/subsymbol} \quad (2.150)$$
$$= \mathcal{H}_{\tilde{x}} - \mathcal{H}_{\tilde{x}/\tilde{y}}$$
$$= \log_2 \left( \frac{|R_{\tilde{x}\tilde{x}}|}{|R_{ee}|} \right) \text{ bits/subsymbol} \quad (2.152)$$
$$= \log_2 |I - W \cdot H| \quad (2.153)$$
$$= \log_2 |I - H \cdot W| \ . \quad (2.154)$$

The ML detector corresponds to the second problem in (2.148) that accepts $\tilde{y}$ and produces $\tilde{x}$. The first problem in (2.147) has MMSE $\tilde{n}$ but is not the ML detector for $\tilde{x}$.

**The Zero-Forcing Solution:** The first problem in (2.147) does produce a zero-forcing estimate when written as its inverse ($H^+$ is the pseudoinverse, see Appendix C):

$$\hat{x}_{ZF} = H^+ \cdot \tilde{y} \ . \quad (2.155)$$

While this $\hat{x}_{ZF}$ estimates the channel input, it is not ML (and consequently neither MAP nor MMSe). The difference is that the sum of squared errors can be reduced if the amplitude of $\tilde{x}$ reduces just enough that the benefit of reducing signal power to reduction of the squared error terms (using LLN and AEP that with probability 1 the sequence of squared errors converges to its average). That reduction in signal strength is good and improves the ML codeword selection. Indeed, the second problem has a MMSE solution that is asymptotically the ML solution: While the mean-square error is minimum, the estimate has bias in that

$$E \left[ \hat{x}/\tilde{x} \right] \leq \tilde{x} \ . \quad (2.156)$$

Removal of this bias improves the detector so that it remains the asymptotic ML detector (see comment Ignoring Bias below) but also becomes unbiased for all codes’ use. One method of bias removal (there are many methods) recognizes that the mutual information has the relationship

$$2^{I(\tilde{x};\tilde{y})} = \left| \frac{R_{\tilde{y}\tilde{y}}}{R_{\tilde{n}\tilde{n}}} \right| = \left| \frac{R_{\tilde{x}\tilde{x}}}{R_{ee}} \right| \ . \quad (2.157)$$

Just as vector coding diagonalizes a channel $y = Hx + n$ so can it also diagonalize a channel $x = Wy + e$. Applying vectoring coding to the MMSE estimate of $x$, given $y$, creates a set of parallel channels whose mutual information remains $I$. Each such channel has an SNR given by

$$SNR_{mmse,k} = \frac{E_k}{\sigma^2_{mmse,k}} = SNR_{unbias,k} + 1 \quad (2.158)$$
Appendix D and Chapter 3 both prove this formally. Bias removal on each dimension requires increasing the received signal on that dimension by the ratio

\[
y'' = \frac{SNR_{mmse,k}}{SNR_{mmse,k} - 1} \cdot y' = \frac{SNR_{unbias,k} + 1}{SNR_{unbias,k}} \cdot y'.
\]  

(2.159)

Simple scaling of each dimension clearly does not change the mutual information. The linear MMSE solution does not depend on Gaussian inputs, and so can be used with codes that have \( \Gamma > 0 \text{ dB} \). The bias in this case requires removal for decision regions based on the constellation \( C \) (with \( |C| < \infty \)) and removal becomes more important.

**Ignoring Bias** Reference [4] notes that bias removal for a true Gaussian code is superfluous because all codewords with probability 1 are on the hypersphere’s surface. Decision regions are thus “hypercones” from the origin to each codeword on the surface. Thus a true infinite-delay ML decoder would be independent of scaling and bias. Nonetheless, individual subsymbols have bias and can affect any finite-complexity approximations to ML decoders. Usually \( SNR_{mmse} > 10 \) for many practical channels so the bias is small and removal may be inconsequential. Nonetheless, removal of bias does no harm and can always improve performance in practice.

**Forward-Channel Bias** Curiously, it is also true that \( E \left[ \frac{\hat{y}/\bar{y}}{\hat{y}} \right] \leq \hat{y} \) also has bias that can be removed by the same scale factor on each dimension in vectoring coding. However, the data-transmission problem of interest is the ML detection/estimation of the channel input, given its output. Scalar systems have ZF and MMSE the same when \( L_x = L_y = 1 \) or if \( H \) is square diagonal, the ZF and unbiased MMSE estimates are the same. However, it is otherwise not the case and the unbiased MMSE is ML and the better estimate in general. Mutual information’s symmetry in channel input and output does not mean that the reversed detector is ML for the opposite-direction problem. Multi-user systems later in this chapter have the first user decision unbiased with ZF and MMSE the same, but all other primary users decoders need bias removal.
2.4 The gap, and some simple channels and codes

This section develops design insight for code use as a system component. In particular, the concept of a coding gap, originally introduced by Forney and described earlier in Subsection 1.3.4.2.4, can simplify such designs. Specifically, the gap allows the code choice independent of other modulation parameters.

2.4.1 The gap

An AWGN channel with input energy per dimension $\bar{E}_x$ and noise power spectral density $\sigma^2$ has SNR = $\bar{E}_x / \sigma^2$. Dividing (2.133) by 2 real dimensions per subsymbol, this AWGN channel has maximum data rate or capacity

$$\bar{C} = \frac{1}{2} \log_2(1 + \text{SNR}) \text{ bits/dimension.} \quad (2.160)$$

As in Subsection 1.3.4.2.4, many codes (simple codes like PAM and QAM, but many others too) can be characterized by a single parameter called the gap $\Gamma$ that is tacitly a function of both the code and of the desired $P_e$. For such codes, the achievable $\bar{b}$ at that $P_e$ is

$$\bar{b} = \frac{1}{2} \cdot \log_2(1 + \frac{\text{SNR}}{\Gamma}) \text{ bits/dimension.} \quad (2.161)$$

This gap approximation is the capacity formula with the SNR reduced by the gap. For these codes usually, the approximation\(^{36}\) is accurate for all $\bar{b} \geq .5$.

Any reliable and implementable system must transmit at $\bar{b}$ at least slightly below capacity. The gap helps analyze and design these systems that transmit with $\bar{b} < \bar{C}$. For gap-characterized codes, the gap is a constant function of the SNR and known bits/dimension for the given $P_e$ and code class $C_x$ as

$$\Gamma(C_x, P_e) \Delta= \frac{2^{2\bar{E}_x} - 1}{2^{2\bar{b}} - 1} = \frac{\text{SNR}}{2^{2\bar{b}} - 1}. \quad (2.162)$$

Table 2.1 lists achievable $\bar{b}$ for the uncoded QAM schemes which has constant $\Gamma = 8.8$ dB at $P_e = 10^{-6}$ using square constellations $N$ versus AWGN SNR.\(^{37}\)

<table>
<thead>
<tr>
<th>$P_e$ = $10^{-6}$</th>
<th>$C_x$ = SQ QAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bar{b}$</td>
<td>.5</td>
</tr>
<tr>
<td>SNR for $P_e = 10^{-6}$ (dB)</td>
<td>8.8</td>
</tr>
<tr>
<td>$2^{2\bar{b}} - 1$ (dB)</td>
<td>0</td>
</tr>
<tr>
<td>$\Gamma$ (dB)</td>
<td>8.8</td>
</tr>
</tbody>
</table>

Table 2.1: Table of AWGN SNR Gaps for $P_e = 10^{-6}$.

Table 2.1 shows that uncoded SQ QAM, or equivalently with PAM, at $P_e = 10^{-6}$, has constant SNR gap $\Gamma = 8.8$ dB. With $C_x$ as uncoded QAM or PAM and $P_e = 10^{-7}$, the gap is instead a constant 9.5 dB for $\bar{b} < .5$.\(^{36}\) For $\bar{b} < .5$, systems of codes can be constructed and viewed as one code that exhibit the same constant gap as for $\bar{b} \geq .5$.

\(^{37}\)Thus, Table 2.1 assumes for $\bar{b} = .5$ that instead of simple BPSK transmission of $[\pm \sqrt{2} \ 0]$ for an energy/symbol of 2, $\bar{E}_x = 1$, and $d_{\text{min}}^2 = 8$ that instead the system transmits one of the following 4 symbol values for two successive QAM channel sub-symbols $2/\sqrt{3} \cdot [1 \ 1 \ 0 \ 0], 2/\sqrt{3} \cdot [-1 \ -1 \ 1 \ 0], 2/\sqrt{3} \cdot [1 \ -1 \ -1 \ 0], \text{or } 2/\sqrt{3} \cdot [-1 \ 1 \ -1 \ 0]$ for also an energy per dimension of $\bar{E}_x = 1$ and $\bar{b} = .5$, but with a minimum distance squared of $d_{\text{min}}^2 = 32/3$. This improvement is $32/24 = 1.3$ dB. Using a few more dimensions like this can provide another .4 dB fairly easily, up to 1.7 dB. So while pure BPSK would have a gap of 10.5 dB at $P_e = 10^{-6}$, this mildly coded system has a gap of 10.5-1.7=8.8 dB, rendering constant the QAM gap of 8.8 dB for all $\bar{b} \geq .5$ as shown in Table 2.1. Similar coded designs, also at $P_e = 10^{-6}$ with lower gaps (than $\Gamma \leq 8.8$ dB) when $\bar{b} = .5$, constructed with more effort to maintain constant gap to as low as 0.25 bits/dimension although not shown here.
dB. The use of codes, say trellis, turbo coding, BICM, and/or forward error correction (see Chapters 9 - 11) reduces the gap. Often these codes will have gaps that do not vary for $\bar{b} \geq .5$. A very well-coded system may have a gap as low as .5 dB at $P_e \leq 10^{-6}$. Figure 2.18 plots $\bar{b}$ versus SNR for various gaps. Smaller gap indicates stronger coding. The curve for $\Gamma = 9$ dB approximates uncoded PAM or QAM transmission at symbol-error probability $P_e \approx 10^{-6}$ (really 8.8 dB). A gap of 0 dB means the theoretical maximum bit rate has been achieved, and of course is not strictly possible but can be closely approached.

![Achievable bit rate for various gaps](image)

Figure 2.18: Illustration of bit rates versus SNR for various gaps.

For a given coding scheme, practical transmission designs often mandate a specific value for $\bar{b}$, or equivalently a fixed data rate. In this case, the design is not for $\bar{b}_{max} = \frac{1}{2} \cdot \log_2(1 + \text{SNR}/\Gamma)$, but rather for $\bar{b}$. The **margin** measures the excess SNR for that given bit rate.

**Definition 2.4.1 (Margin, repeated from Section 1.3.5)** The **margin**, $\gamma_m$, for transmission on an AWGN (sub)channel with a given SNR, for a given number of bits per dimension $\bar{b}$, and for a given $C_x$ and $P_e$ with gap $\Gamma$ is the amount by which the SNR can be reduced (must be increased for negative margin in dB) and still maintain an error probability at or below the target $P_e$.

Margin is accurately approximated through the use of the gap formula by

$$\gamma_m = \frac{2^{2\bar{b}_{max}} - 1}{2^{\bar{b}} - 1} = \frac{\text{SNR}/\Gamma}{2^{2\bar{b}} - 1}.$$  

(2.163)

The margin is the amount by which the SNR can lower before performance degrades to an error probability greater than the target $P_e$ that defines the gap. A negative margin in dB means that the SNR
must improve by the margin’s magnitude to achieve the $P_e$. The margin relation has alternative form written as

$$b = 0.5 \cdot \log_2 \left( 1 + \frac{\text{SNR}}{\Gamma \cdot \gamma_m} \right), \quad (2.164)$$

where the margin and gap are somewhat interchangeable. For instance, gap $\Gamma = 3$ dB with $\gamma_m = 6$ dB margin might be replaced by $\Gamma = 6$ dB and margin $\gamma_m = 3$ dB. The gap measures the code; the margin measures excess SNR.

**EXAMPLE 2.4.1 (AWGN with SNR = 20.5 dB)** An AWGN has SNR of 20.5 dB. The capacity of this channel is then

$$c = 0.5 \cdot \log_2 (1 + \text{SNR}) = 3.5 \text{ bits/dim} . \quad (2.165)$$

With a $P_e = 10^{-6}$ and 2B1Q (4 PAM),

$$\bar{b} = 0.5 \cdot \log_2 \left( 1 + \frac{\text{SNR}}{10^{-38}} \right) = 2 \text{ bits/dim} . \quad (2.166)$$

With concatenated trellis and forward error correction (or with “turbo codes” – see Chapters 10 and 11), there is a coding gain of 7 dB at $P_e = 10^{-6}$, which implies a gap of $\Gamma = 8.8 - 7 = 1.7$ dB, then the achievable data rate is

$$\bar{b} = 0.5 \cdot \log_2 \left( 1 + \frac{\text{SNR}}{10^{-18}} \right) = 3 \text{ bits/dim} . \quad (2.167)$$

Suppose a transmission application requires $\bar{b} = 2.5$ bits/dimension, then the margin for the coded system is

$$\gamma_m = \frac{2^{2.3} - 1}{2^{2.5} - 1} = \frac{63}{31} \approx 3 \text{ dB} . \quad (2.168)$$

This means the noise power can increase (or the transmit power reduce) by up to 3 dB before the target error probability of $10^{-6}$ and $\bar{b} = 2.5$ bits/dimension is no longer met. Alternatively, suppose a design transmits 4-QAM over this channel and no code is used, then the margin is

$$\gamma_m = \frac{2^{2.2} - 1}{2^{2.1} - 1} = \frac{15}{3} \approx 7 \text{ dB} . \quad (2.169)$$

### 2.4.2 Mutual Information and Capacity for DMCs

Subsection 2.2.2 and Chapter 1 introduced the discrete memoryless channel; both the DMC’s inputs and outputs are elements of discrete finite sets. There are $M$ inputs, $x_0, \ldots, x_{M-1}$ and $J$ outputs $y_0, \ldots, y_{J-1}$. The term “memoryless” means that the outputs on the next use of the channel are independent of any previous inputs.

**BSC Capacity:** Figure 2.8’s BSC is probably the most widely cited DMC, and might be characterized for an underlying hard-coded AWGN by

$$p \triangleq P_e = \frac{N_b}{b} \cdot Q \left( \frac{d_{\text{min}}}{2\sigma} \right) . \quad (2.170)$$

The capacity of the BSC can be computed in a straightforward manner by substitution into the mutual information formula:

$$I(x; y) = \sum_{m=0}^{1} \sum_{j=0}^{1} p_x(m) \cdot p_y / x(j, m) \cdot \log_2 \left( \frac{p_y(x(j, m))}{p_y(j)} \right) \quad (2.171)$$
\begin{align*}
&= \ p_x(0) \cdot (1 - p) \cdot \log_2 \left( \frac{1 - p}{p_x(0) \cdot (1 - p) + p_x(1) \cdot p} \right) \quad (2.172) \\
&+ \ p_x(0) \cdot (p) \cdot \log_2 \left( \frac{p}{p_x(0) \cdot p + p_x(1) \cdot (1 - p)} \right) \quad (2.173) \\
&+ \ p_x(1) \cdot (p) \cdot \log_2 \left( \frac{p}{p_x(0) \cdot (1 - p) + p_x(1) \cdot p} \right) \quad (2.174) \\
&+ \ p_x(1) \cdot (1 - p) \cdot \log_2 \left( \frac{1 - p}{p_x(0) \cdot p + p_x(1) \cdot (1 - p)} \right) \quad (2.175)
\end{align*}

The input probabilities \( p_x(0) \) and \( p_x(1) \) are interchangeable in the above expression. Thus, the maximum must occur when they are equal: \( p_x(0) = p_x(1) = .5 \). Then,

\begin{align*}
\bar{C} &= (1 - p) \cdot \log_2 [2(1 - p)] + p \cdot \log_2(2p) \\
&= 1 - H(p) \quad (2.176)
\end{align*}

where

\[ H(p) \triangleq -p \cdot \log_2(p) - (1 - p) \cdot \log_2(1 - p) \quad (2.178) \]

the entropy of a binary distribution with probabilities \( p \) and \( 1 - p \). As \( p \to 0 \), there are no errors made and \( \bar{C} \to 1 \) bit/symbol (or bit/dimension), otherwise \( C \leq 1 \) for the BSC.

**BEC’s Capacity:** A second commonly encountered channel is the binary erasure channel (BEC) of Figure 2.11. The channel is again symmetric in \( p_x \), so that the maximizing input distribution for the mutual information is \( p_x(0) = p_x(1) = .5 \). The capacity is then

\begin{align*}
\bar{C} &= \left[ \frac{1}{2} (1 - p) \log_2 \frac{1 - p}{2(1 - p)} \right] + \left[ \frac{1}{2} p \log_2 \frac{p}{2p^2} \right] 2 \\
&= 1 - p \quad (2.179)
\end{align*}

Again, as \( p \to 0 \), there are no errors made and \( \bar{C} \to 1 \) bit/symbol, otherwise \( \bar{C} \leq 1 \) for the BEC. When \( p \leq 0.5 \), then \( \bar{C}_{BEC} \geq \bar{C}_{BSC} \), which Problem 2.4 explores further. An uncoded BEC clearly has \( \bar{b} = 1 - p \) because 0 and 1 pass without error, but \( \bar{P}_e \to 0 \) with good code use.

More generally, the symmetric DMC may have an \( M \times J \) matrix of transition probabilities \( (p_{\text{output/input}} \text{ without change over row index and input change over column index}) \) such that every row is just a permutation of the first row and every column is a permutation of the first column. For instance, the BSC has

\[ \begin{bmatrix} 1 - p & p \\ p & 1 - p \end{bmatrix} \quad (2.181) \]

The symmetric DMC’s capacity/maximizing distribution is again uniform by symmetry. A special case of interest is Figure 2.10’s **Universal Discrete Symmetric Channel (UDSC)** has \( 2^b \) discrete inputs and the same set of \( 2^b \) outputs. The probability of the output being the same as the input is \( (1 - p_s) \) while the probability of any other possible value is \( p_s/(2^b - 1) \). Because of the symmetry, the maximizing input distribution is again uniform among the \( 2^b \) possible discrete messages. The capacity is

\[ C = b - p_s \cdot \log_2 \frac{2^b - 1}{p_s} + (1 - p_s) \cdot \log_2 (1 - p_s) \leq b \text{ bits.} \quad (2.182) \]

A typical use of this channel is when \( b = 8 \) or the transmission system is organized to carry an integer number of information bytes. If the UDSC is constructed from 8 successive uses of the BSC, then

\[ p_s = 1 - (1 - p)^8 \approx b \cdot p = 8p \text{ for small } p \quad . \quad (2.183) \]

Outer codes may be then organized in terms of byte symbols or modulo-256 arithmetic (or more generally modulo-\( 2^b \) arithmetic). In particular, MDS codes like Reed Solomon can for small additional rate reduction improve \( P_e \) essentially to 0.  

251
2.4.3 Capacity, Coding, and the Gap

The AWGN capacity has been previously computed as \( \bar{C} = 0.5 \cdot \log_2(1 + \text{SNR}) \) and appears in Figure 2.19 with expanded SNR on the horizontal axis. Figure 2.19’s semi-log plot shows that as SNR becomes reasonably large (say 20 dB or more), that increase of capacity by 1 bit per dimension requires an additional 6 dB of SNR. Since QAM finds heavy use in practice, a bit per dimension of QAM corresponds to two bits per symbol or an often quoted rule of “3 dB per bit”. At low SNR (below 10 dB), this rule no longer applies and for very low SNR (below 0 dB), capacity essentially scales linearly with SNR (instead of logarithmically). This is evident in that

\[
\lim_{\text{SNR} \to 0} \bar{C} = \frac{0.5}{\ln 2} \cdot \text{SNR} .
\]  

(2.184)

since \((\log(1 + x) \approx x)\) for small \(x\).

Codes that are constructed from PAM- and QAM-like sub-symbols can recover this lost 9 dB of uncoded transmission. The price will not be a higher error rate (and indeed the error probability can be driven to zero according to the capacity theorem), nor transmit energy increase, but rather a significantly more complex encoder, and especially, a more complex decoder. The use of \(10^{-6}\) and the corresponding gap of 8.8 dB may seem somewhat arbitrary – one could argue why not \(10^{-7}\) or smaller, where the corresponding larger gaps would suggest yet even higher than 9 dB improvement in SNR is possible. Chapter 11 will illustrate that once the error probability is less than \(10^{-6}\), an outer concatenated code working on the presumption that the inner AWGN has been well handled and converted to a BSC or DMC with probability \(p = 10^{-6}\), can easily drive overall error probability (bit, subsymbol, or codeword) close to zero with little data rate loss, so \(10^{-6}\) is an often used design figure for the inner channel and the first decoder.
2.4.4 Energy per bit and low-rate coding

The gap concept is inapplicable below SNR = 10 dB. In this low-SNR range, typical transmission is at $\bar{b} \leq 0.5$, and codes such as binary convolutional or block codes with $|C| = 2$ find use. There is essentially a varying coding-gain limit at lower SNR with any given type of code, so the gap does not apply.

The AWGN capacity formula also provides the minimum energy per bit for reliable data transmission. This arises by writing the capacity result as

$$\bar{b} < \bar{C} = \frac{1}{2} \cdot \log_2 \left( 1 + \frac{\bar{E}_x}{\sigma^2} \right)$$  \hspace{1cm} (2.185)

$$= \frac{1}{2} \log_2 \left( 1 + \frac{\bar{E}_x}{N\sigma^2} \right)$$  \hspace{1cm} (2.186)

$$= \frac{1}{2} \log_2 \left( 1 + \frac{\bar{b}\bar{E}_x}{\sigma^2} \right)$$  \hspace{1cm} (2.187)

$$= \frac{1}{2} \log_2 \left( 1 + \frac{\bar{b}\bar{E}_b}{\sigma^2} \right)$$  \hspace{1cm} (2.188)

Solving for $\frac{\bar{E}_b}{\sigma^2}$ in (2.188) yields

$$\frac{\bar{E}_b}{\sigma^2} = 2^{\frac{2\bar{b}}{b}} - 1$$  \hspace{1cm} (2.189)

Equation (2.189) essentially tells us the minimum required $\bar{E}_b/\sigma^2$ for any given code rate $\bar{b}$ on the AWGN. Of fundamental interest is the case where $\bar{b} \to 0$ (that is large redundancy in the code). Then (2.189) reduces to

$$\frac{\bar{E}_b}{\sigma^2} = 2 \cdot \ln 2 \ (1.4 \text{ dB})$$  \hspace{1cm} (2.190)

meaning that the energy/bit must be above this finite value even if a code uses infinite redundancy (or infinite dimensionality or bandwidth). This result is sometimes phrased in terms of the quantity $\bar{E}_b/N_0=.5(\bar{E}_b/\sigma^2)$, which is equivalent to the statement that the minimum required $\bar{E}_b/N_0$ is -1.6dB, an often-cited result.

2.4.5 Some Simple Design Examples with Coding

This subsection provides some code-use examples that improve the data-transmission system. More complete code-design address appears in Chapters 8 - 11.

2.4.5.1 Designing with Reed Solomon/Cyclic Codes

Cyclic block codes over the field $GF(q^i)$ for integer $i$ have all codewords as circular shifts of one another. Reed Solomon\(^{38}\) codes are linear and also MDS and therefore achieve the Singleton Bound performance, namely $d_{\text{free}} = N - K + 1$. For the binary case of $q^i = 2$, only trivial codes are MDS and then have $d_{\text{free}} = n - k + 1$, but when $q > 2$, nontrivial codes exist and Reed Solomon (RS) are an example. A RS code with $GF^{256}$ therefore works with bytes (octets) of 8 bits each, and Figure 2.10’s Symmetric DMC can be a potential byte-in-byte-out channel with $q - 1 = 255$. Such RS codes allow $1 < N < 256$ with $\bar{N} = 1$ byte. The number of parity subsymbols allowed within the codeword is 0 $\leq P \leq N - 1$ with correspondingly is $d_{\text{free}} = P - 1$. Usually the design selects $P$ even so that $d_{\text{free}}$ is odd. Thus the RS code for any $q$ can correct $P/2$ subsymbols in error anywhere in the received codeword and detect up to $P$ subsymbols in error correctly. The RS code has a special feature that if the channel accurately provides the location of erased subsymbols (so all the remaining subsymbols are almost certainly correct), the ML decoder can correct up to $P$ of such marked/erased subsymbols within a codeword. RS decoders can correct combinations of up to $P'$ erased known byte locations and $(P - P')/2 \geq 0$ unknown erred byte locations.

Equation (2.31) provides the codeword error probability, while the average number of erred subsymbols per codeword, and similarly average number of erred bits per codeword are in (2.32) and (2.33) respectively.

A channel might have inner decoded bit-error probability $p = 1.25 \times 10^{-4}$ and thus a byte-error probability of approximately $10^{-3}$. By using the first term of (2.32), with $N_e = N_{d_{free}} / K$, the code must choose $N$ and $K$ to satisfy

$$N_e = \left( \frac{N}{d_{free}} \right) / (K(p)^{\frac{d_{free} - 1}{2}}) < 10^{-7}$$

(2.191)

to have a subsymbol (byte) error probability less than $10^{-7}$. Some trial values for $N$ and $K$ then suggest that a code with $N = 80$ and $P = 16$ will produce a byte-error probability of roughly $2.7 \times 10^{-8} < 10^{-7}$. There is a loss in data rate of 20% to gain this improvement. On an AWGN (prior to hard decoding), a 20% increase in symbol rate to offset results in 1 dB loss, which may be an acceptable trade for the improved reliability.

### 2.4.5.2 Design with Simple Convolutional Code

The convolutional code in Example 2.2.1 has $d_{free} = 5$ and $r = 1/2$. It can be used on any hard-decoded (to BSC) channel with bit-error probability $p$ to reduce that probability to $p^{\left\lceil \frac{d_{free} - 1}{2} \right\rceil} = p^2$ at the expense of more complicated decoding and $r = \frac{1}{2}$. (A simple repeat-once code would not be able to achieve such a reduction, or really any error-probability reduction.)

For this same code with soft decoding, the improvement in minimum distance is from $d_{free,uncoded} = d^2$ to $d_{free} = 5d^2$, an improvement of 7 dB, but at loss of 50% data rate. If the AWGN subsymbol rate can be increased by a factor of 2 to offset the data loss, this would be an SNR loss of 3 dB. Thus, when the bandwidth can be doubled, this code gains 4 dB over simple 2PAM or QPSK. Its use improves error probability from $10^{-3}$ to $10^{-6}$.

### 2.4.5.3 Low Density Parity Check (LDPC) Codes

Low Density Parity Check codes are theoretically designed by random parity-matrix construction of a linear binary block code. These apply directly to a binary AWGN (or to a BSC). Among the set of such codes constructed at random will be some that have one codeword in each typical set for $x$ given $y$ if the dimensions of the LDPC parity matrix are such that $r < C$ asymptotically as $n \to \infty$. Such codes typically have long block lengths in practical use, and iterative decoders that use LLR’s for each bit (so ML or MAP decoder for each bit instead of the entire codeword) can be constructed with high degree of parallelism and reasonable complexity. They are much more complex than the simple convolutional code of Example 2.2.1, but have gains that essentially achieve capacity on channels where $C < 1$. For AWGN channels with larger capacities (larger QAM/PAM constellations’ $|C|$), these codes can be mapped with a Gray code (and interleaving as in the BICM case) and achieve also highest performance; except for shaping gain that must be independently implemented.
2.5 Parallel Channels and the Multi-tone AWGN Channel

Sections 2.3 and 2.4 (for $\bar{C}$) illustrate the most essential capacity concepts. In practice, engineers usually require in data rates in units of bits/second. This section investigates the conversion of the previous results to units of bits/second. This section assumes throughout that $L_x = L_y = 1$, but easily applies independently to each independent spatial dimension in a MIMO system.

2.5.1 Capacity Conversion for Memoryless Channels

Both the DMC’s discussed previously and the AWGN are memoryless channels in that the current channel output depends only on the current channel input. Memoryless AWGN channels can directly use modulation methods such as PAM, QAM, PSK, and others with $2W$ dimensions/second, where $W$ denotes the system’s (positive-frequency-only) “bandwidth.” As Chapter 3 shows a basis function like $\varphi(t) = \frac{1}{\sqrt{T}} \cdot \text{sinc}(t/T)$ allows successive transmitted waveforms to be independently sampled at times $kT$ at the channel output without interference into one another, which is also intuitively obvious from the nature of the sinc($t$) function’s zero crossings and that sinc($t$) $\ast$ sinc($t$) = sinc($t$). This basis function requires a bandwidth of $\frac{1}{T} = 2\left(\frac{1}{2T}\right) = 2W$ dimensions per second. The sinc waveform has strict non-zero-energy bandwidth $[-1/2T, 1/2T]$. The term “bandwidth” is approximate in practice. More generally, this text presumes for coding purposes that the system engineer has designed the modulation system for a memoryless (or close to memoryless) channel such that the number of time dimensions per second transmitted over the channel, or the subsymbol rate, is $1/\bar{T}$, where $\bar{T} \triangleq T/N$.

Thus, the AWGN channel with transmit functions of bandwidth $W = 1/(2\bar{T})$ has $N/T$ output dimensions per second, so the continuous-time capacity is then

$$C = 2W \cdot \bar{C} \text{ bits/sec}$$

(2.192)

$$= W \cdot \log_2(1 + \text{SNR}) \text{ bps}$$

(2.193)

a well-known result.

2.5.2 Waveform Channels with Limited Bandwidth or Memory

Most practical channels are not memoryless, so of interest also is the channel capacity for a filtered Gaussian noise channel with impulse response $h(t)$ and additive Gaussian noise with known power spectral density $S_n(f)$. Figure 2.20 illustrates multi-tone modulation where frequency is decomposed into a parallel set of $N$ non-overlapping channels of equal bandwidth. These channels will use the noise-equivalent channel of Section 1.3.7 where the channel transfer function $H(f)$ is divided by the the canonical noise-spectrum factorization factor to provide $H_{eq}(f)$, which is a filtered AWGN. It is this channel that is then viewed for capacity purposes as a parallel set of frequency-indexed channels.

In the limiting case, the symbol interval $T = \bar{N} \cdot \bar{T}$ becomes arbitrarily large as $\bar{N} \to \infty$, and analysis considers the channel as “one-shot” with infinite complexity and infinite decoding delay. The corresponding filtered-channel capacity explicitly includes the symbol energy $E_s = P_x \cdot T$ and the symbol period $T$ as arguments, $\bar{C} \to \bar{C}(E_s, T)$. Then, the capacity, in bits/second, for a filtered Gaussian-noise channel is

$$C = \lim_{T \to \infty} \left(\frac{1}{T}\right) \cdot \frac{\bar{C}(P_x T, T)}{\text{bits/dim}}$$

(2.194)

$$= \lim_{T \to \infty} \frac{\bar{C}(P_x T, T)}{T}.$$  

(2.195)

$\bar{C}(P_x T, T)$ is still the quantity defined earlier, but the notation now emphasizes the dependence upon the symbol period $T$.  

255
Because $H_{eq}(f)$ is one-to-one\textsuperscript{39}, the sub channels will each have their own capacity. Each channel will have a gain equal to the center frequency of its Fourier transform, which is the Fourier Transform value as $N \to \infty$.

The capacity of the overall set of channels simply adds the capacity of each:

\begin{equation}
C(P_x, T) = \sum_{i=1}^{\infty} \max\left[0, \frac{1}{2} \log_2 \left(1 + \frac{E_i}{\sigma_i^2}\right)\right]
\end{equation}

with maximizing transmit-energy allocation\textsuperscript{40}:

\begin{equation}
P_x \cdot T = \sum_{i=1}^{\infty} \max\left[0, \lambda' - \sigma_i^2\right]
\end{equation}

Dividing both sides by $T$ and taking limits as $T \to \infty$ produces

\begin{equation}
\mathcal{C} = \lim_{T \to \infty} \frac{1}{T} \sum_{i=1}^{\infty} \max\left[0, \frac{1}{2} \log_2 \left(\frac{\lambda'}{\sigma_i^2}\right)\right]
\end{equation}

and

\begin{equation}
P_x = \lim_{T \to \infty} \frac{1}{T} \sum_{i=1}^{\infty} \max\left[0, \lambda' - \sigma_i^2\right]
\end{equation}

\textsuperscript{39}Unless $H_{eq}(f) = 0$ at some frequencies, in which case no energy is transmitted at those frequencies.

\textsuperscript{40}This is derived by differentiating the mutual information with respect to the energies subject to a lagrange multiplier on the side constraint of constant energy sum of all subsymbol energies.
Both sums above are nonzero over the same range for \( f \), which this text calls \( \mathcal{F}_e \). In the limit,

\[
\sigma_i^2 \rightarrow \frac{S_n(f)}{|H(f)|^2}
\]

and

\[
\frac{1}{T} \rightarrow df \ , \quad (2.201)
\]

leaving Shannon’s famous “water-filling” scheme for the waveform channel’s capacity calculation:

\[
\mathcal{C} = \frac{1}{2} \int_{\mathcal{F}_{opt}} \log_2 \left( \frac{\lambda \cdot |H(f)|^2}{S_n(f)} \right) df \quad (2.202)
\]

and

\[
P_x = \int_{\mathcal{F}_{opt}} \left( \lambda' - \frac{S_n(f)}{|H(f)|^2} \right) df \ , \quad (2.203)
\]

where the transmit spectrum is chosen to satisfy

\[
\lambda' = \frac{S_n(f)}{|H(f)|^2} + S_x(f) \ , \quad (2.204)
\]

which results in the equivalent capacity expression

\[
\mathcal{C} = \frac{1}{2} \int_{\mathcal{F}_{opt}} \log_2 \left( 1 + \frac{S_x(f) \cdot |H(f)|^2}{S_n(f)} \right) df = \frac{1}{2} \int_{\mathcal{F}_{opt}} \log_2 \left( \lambda' \frac{|H(f)|^2}{S_n(f)} \right) \cdot df \ . \quad (2.205)
\]
Figure 2.21 illustrates the continuous water filling concept. The optimum transmit spectra computes as energied “poured” into the inverted channel (multiplied by any noise power spectral density) until no energy remains, which determines both \( \lambda' \) and \( F_{opt} \). Then \( C \) is computed through (2.202) or (2.205).

### 2.5.3 Capacity of the infinite bandwidth channel

An interesting interpretation of the AWGN capacity result presumes infinite bandwidth on the part of the transmitter and a channel that ideally passes all frequencies with equal gain and no phase distortion. In this case, \( W \to \infty \) in \( C = W \cdot \log_2 (1 + \text{SNR}) \), or

\[
C_\infty = \lim_{{W \to \infty}} W \cdot \frac{1}{\ln 2} \ln \left( 1 + \frac{P_x}{2W\sigma^2} \right) = \frac{1}{\ln 2} \cdot \frac{P_x}{2\sigma^2} .
\]

(2.206)

This result shows that even with infinite bandwidth that have a finite-power constraint imposes a finite data rate.

### 2.5.4 Example of Water-Filling Capacity Calculation

An example of the continuous water filling is the flat AWGN channel with \( H(f) = 1 \) and \( \mathcal{S}_n(f) = \frac{N_0}{2} \). Then, one orthonormal set of basis functions is \( \frac{1}{\sqrt{T}} \cdot \text{sinc} \left( \frac{t - iT}{T} \right) \) \( \forall i \); the noise samples have constant \( \sigma_i^2 = \frac{N_0}{2} \) \( \forall i \). Thus,

\[
P_x = (\lambda' - \frac{N_0}{2}) \cdot 2W
\]

(2.207)

where \( W = 1/2T \). Then,

\[
\lambda' = \frac{P_x}{2W} + \frac{N_0}{2}
\]

(2.208)

and

\[
C = \left( \frac{1}{2} \cdot \log_2 \left[ \frac{P_x}{2W} + \frac{N_0}{2} \right] \right) 2W = W \cdot \log_2 (1 + \text{SNR})
\]

(2.209)

the same result as obtained earlier in Equation (2.193).

**EXAMPLE 2.5.1** \((1 + .9 \cdot e^{-j\pi f} \text{ Channel Capacity})\) A second example is the lowpass channel with impulse response \( h(t) = \text{sinc}(t) + .9 \cdot \text{sinc}(t - 1) \) and AWGN with \( \frac{N_0}{2} = .181 \). Then

\[
P_x = \int_{-W}^{W} \left( \lambda' - \frac{.181}{1.81 + 1.8 \cdot \cos(2\pi f)} \right) df
\]

(2.210)

where \( W \) is implicitly in Hz for this example. If \( P_x = 1 \) with an SNR of 10 dB, the integral in (2.210) simplifies to\(^{41}\)

\[
\frac{1}{2} = \int_{0}^{W} \left( \lambda' - \frac{.181}{1.81 + 1.8 \cdot \cos(2\pi f)} \right) df
\]

(2.211)

\[
= \lambda'W - .181 \left\{ \frac{1}{\pi \cdot \sqrt{1.81^2 - 1.8^2}} \arctan \left[ \frac{1.81 - 1.8}{1.81 + 1.8} \cdot \tan(\pi W) \right] \right\}
\]

(2.212)

At the bandedge \( W \),

\[
\lambda' = \frac{.181}{1.81 + 1.8 \cdot \cos(2\pi W)} .
\]

(2.213)

\(^{41}\)From a table of integrals with \( a^2 > b^2 \):

\[
\int \frac{df}{a + b \cdot \cos(2\pi f)} = \frac{1}{\pi \cdot \sqrt{a^2 - b^2}} \cdot \arctan \left[ \sqrt{\frac{a - b}{a + b}} \cdot \tan(\pi f) \right] + \text{constant}
\]

258
leaving the following transcendental equation to solve by trial and error:

\[
\frac{1}{2} = \frac{0.181W}{1.81 + 1.8 \cdot \cos(2\pi W)} - 0.3032 \cdot \arctan(0.0526 \cdot \tan(\pi W))
\] (2.214)

\(W = 0.44\) approximately solves (2.214) and leaves \(\lambda' = 1.33\).

The capacity is then

\[
C = \int_0^{0.44} \log_2 \left( \frac{1.33}{1.81} \left( 1.81 + 1.8 \cos 2\pi f \right) \right) \cdot df
\] (2.215)

\[
= \int_0^{0.44} \log_2 7.35 \cdot df + \frac{1}{2\pi} \int_0^{0.44} \log_2 (1.81 + 1.8 \cos 2\pi f) \cdot df
\] (2.216)

\[
= 1.266 + 0.284
\] (2.217)

\[
\approx 1.55 \text{ bits/second}
\] (2.218)

Chapters 3, 4, and 5 will provide a more complete development of this example and band-limited channels’ capacities.
2.6 Multiuser Coding Basics

Transmission channels often accommodate more than one user, as in Figure 2.22’s two independent users’ messages $m_1$ and $m_2$. Each user has a transmitter/encoder and a corresponding receiver/detector. The most general channel model remains the conditional probability distribution $p_y/x$, where the channel input has 2 users’ components $x = \{x_1, x_2\}$, as does the channel output $y = \{y_1, y_2\}$. These “multi-user” channels42 have three commonly encountered types that Subsection 2.6.1’s Figures 2.25, 2.26, and 2.27 later illustrate respectively: the multi-user multiple-access channel (MAC), broadcast channel (BC), and interference channel (IC) for $U$ users, $u = 1, ..., U$. The user set is $U = \{1, 2, ..., U\}$, and thus $|U| = U$.

These three multi-user-channel types respectively permit coordinated signal processing at the only the receiver/detector (MAC), only the transmitter/encoder (BC), or neither (IC). Other multi-user channels can be represented as some combination of these 3 basic types as in Section 2.10. The term “multi-user” means all users’ transmit symbols (and messages) are independent of any other user’s corresponding transmit symbols (messages). Specifically then Section 2.10’s “relay” or “mesh” channels, where an intermediate-user chain successively passes the same message to an ultimate intended message recipient, do not directly fall under this multi-user definition. Relay channels are instead a special case of Section 2.10’s expanded-user channels that circumvent inter-user dependencies with dependent stages of independent expanded users.

---

42 This text uses a hyphen in “multi-user” instead of writing the compound “multiuser” often found elsewhere as the author could find no evidence that such a word exists.

260
The independent-user-input restriction allows Figure 2.23’s capacity-region $C(b)$ characterization of the multi-user channel. Subsection 2.6.2 further expands Section 2.3’s single-user mutual-information and capacity concepts, before formally introducing rate regions. More simply, the capacity region $C(b)$ expands “data-rate” to a two-dimensional data-rate vector $R = [R_1, R_2]^*$, or similarly the bits/symbol vector is

$$b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = R \cdot T = \begin{bmatrix} R_1 \cdot T \\ R_2 \cdot T \end{bmatrix},$$

(2.219)

where $T$ is a symbol rate common to all users$^{43}$. Subsection 2.6.2 more formally defines the capacity region. Figure 2.22’s 2-user channel may cause the 2 independent users’ rates to be mutually dependent, so an achievable rate region better describes the trade-off, as in Figure 2.23. When Figure 2.23’s shaded area is a rectangle, the users are independent and thus single-user design and analysis applies to both users individually. Subsection 2.6.3 generalizes detection methods for multi-user channels. Subsection 2.6.2’s various concepts and Subsection 2.6.3’s detection foundation lay a foundation for Subsection 2.6.4’s general multi-user achievable- and capacity-region specifications.

---

**Sub-users:** Figure 2.24 illustrates sub-users. Sub-users are independent messages components of a single user. A user’s data rate is the sum of its sub-users’ data rates, e.g. $b_u = b_{u,a} + b_{u,b}$. The sub-user

---

$^{43}$Such synchronization between users make take many forms that involve anything from global-positioning-satellite-based synchronization to symbol system framing with various network clocks to just making packets long enough in various approaches. Chapter 6 addresses synchronization methods.
decomposition may be important because only certain components of transmitted signal (or instance the signal from only one antenna of a MIMO transmitter) reaches a particular receiver. That particular receiver may then possible decode that sub-user if it is independent of the other sub-user component that is not sensed. Prior detection of other (sub-) users allows their distortion’s removal from the desired user’s detection. Figure 2.24 illustrates that receiver 3 may actually decode (and consequently remove) sub-user 1b and user 2 before decoding user 3’s two components. Similarly receiver may decode (and remove) 1b first, then decode 2, and so on. All sub-users are independent, so $p_x$ factors accordingly. In such cases, it is convenient to increase $U$ to the number of sub-users, perhaps $L_u$ of them, and this text does so. Then, the large dimensional rate region can collapse to the original users through the sums, $u = 1...U$ ($U$ is original number of users) trivially.

$$b_u = \sum_{\ell=1}^{L_u} b_{u,\ell}. \tag{2.220}$$

Sub-users are equivalent to the terminology “time-sharing” or more generally “dimensional sharing.” In effect, sharing a dimension means subdividing its use into two (or more) codes/users that each apply a fraction of the time (dimensions). Each fraction/code is effectively a sub-user. This type of sub-user indexing can be particularly useful also in MIMO systems with many antennas per user. Often this text uses $U'$, instead of $U$, when sub-users’ specific enumeration helps analysis.

**Macro Users:** In a similar but reverse manner, a *macro user* occurs when 2 or more independent users are identical in their impact; formally interchange of their positions in the overall $x$ does not change $pxy$. These users’ aggregation into a macro user can sometimes simplify $C(b)$ construction. For instance the channel $y = x_1 + x_2 + n$ with $px_1 = p_{x_2}$ and $px = px_1 = px_2$ is essentially a single user channel for which the capacity would be $b = C = \log_2(1 + \frac{1}{\sigma^2})$ if each user had unit energy. Any split of this rate into non-negative bits/subsymbol $b_1 + b_2 = b$ is trivially possible by reversing the macro user into the two original users (sub-users from the macro-user perspective).

**Some basic mutual-information bounds:** Two mutual-information types are of interest in a multi-user channel: the mutual information corresponding to the maximum of all users’ sum bits/symbol $b \leq I(x; y)$ and the individual-user mutual information corresponding to each user $I(x_u; y), u = 1,...,U$.\(^44\) The word maximum implies good code here, as in those asymptotically considered in Section 2.3. If each user has per-user bits per symbol $b_u$, $u = 1,...,U$, then

$$b = \sum_{u=1}^{U} b_u \leq I(x; y), \tag{2.221}$$

with the inequality constraint imposed because inputs, outputs, or both may not be coordinated, even with the best codes.\(^45\) Also, a bound for the best average number of bits is

$$I(x_u; y) \leq I(x; y), \tag{2.222}$$

where $I(x_u; y)$’s calculation averages the other $U - 1$ users are averaged over their distributions in calculation of $I(x_u; y)$. Equation (2.222) uses distributions with $\chi$ as the integration variable for the random vector $x$ and $v$ as the integration variable for the random vector $y$, are\(^46\)

$$px_u(x_u) = \int_{\chi \in \mathbb{X}\setminus x_u} px_\chi \cdot d\chi. \tag{2.223}$$

\(^44\)The individual mutual information rates for a particular output $I(x_u; y_u)$ are also of interest in the interference channel.

\(^45\)There is a presumption of a common symbol period in multi-user communication that essentially assumes a synchronization that may not be present in practice. However, a sufficiently long symbol interval may always be defined so that essentially all users conform to it. Introduction of multiple symbol rates or actual data rates obfuscates basic principles and adds little to the discussion, but such a constraint would need consideration in practice.

\(^46\)Readers should recall that mutual information and entropy use the distribution in two ways: (1) as a probability density for a functional average and (2) as the function itself.
and

\[ p_{x_u, y}(x_u, v) = \int_{x \in \{x \setminus x_u\}} p_{x, y}(x, v) \cdot d\chi. \]  

(2.224)

The set-removal operation indicated by \( \setminus \) removes an element (or subset) from a larger set. The mutual information \( I(x_u; y) \) has a conditional form when given other users \( u \subseteq \{U \setminus u\} \) to obtain the conditional mutual information per user \( I(x_u; y / x_u) \). The conditional mutual information effectively presumes previous decisions on other users’ \( (x_u)'s \) symbols, which essentially removes those other signals degrading effect upon subsequent user \( u \)'s symbol detection. Thus, \( I(x_u; y) \) is not necessarily an upper bound on \( b_u \) in the multi-user channel.

### 2.6.1 Multi-User Channel Types

**Multiple-Access Channel (MAC):** Figure 2.25’s single receiver characterizes the multiple-access channel (MAC), which accepts independent messages \( m_u, u = 1, ..., U \) from \( U \) physically separated encoders, each with its own transmit symbol vector \( x_u, u = 1, ..., U \). The single detector accepts the single channel-output vector \( y \) and provides the detected outputs \( \hat{m}_u, u = 1, ..., U \). There are thus multiple users that access the channel with a single output, whence the name “multiple-access” channel. The single output vector \( y \) (as well as each input vector) may have \( L_x > 1 \) and/or \( N \geq 1 \) such that it creates a \textit{vector multiple access} channel. All users have the same number of dimensions – when this is not the case physically, MAC analysis may add dummy unused dimensions to the input and output vector as necessary to ensure a consistent number of any user’s output dimensions. The MAC’s single output allows a single “coordinated” receiver to detect jointly all the transmitted user messages. The single detector’s use also simplifies MAC analysis. The MAC inputs may share encoding strategy or policy, but each encoder \( u \) knows only its input message \( m_u \). If a single user decomposes into two or more transmit-cooperating users (sub-) for some \( p_x \), then the channel ceases to be strictly a MAC. If such a single user decomposes into two or more separated transmitters, then the channel remains a MAC but with a larger value for \( U \).

Multiple-access channels abound in communications, often occurring where several subscribers communicate to a single central service provider: For instance, a wireless channel may have several user transmitters that each share a common frequency band to transmit “uplink” to an “access point” or “base station” (cell tower). Such architectures occur in cellular and most Wi-Fi systems, and are vector

---

47This is a sum instead of an integral if \( \mathbf{x} \) is a discrete random vector.

48Usually the case where \( N > 1 \) occurs in multi-tone designs where each tone can be separately viewed as a multi-user channel by itself, so this chapter views \( N \to \infty \) in the good-code/AEP sense; Chapters 4 and 5 address finite time-frequency dimensional channels and how they still decompose into largely independent multi-user channels, so \( L_x \) and \( L_y \), indicating space dimensionality is more useful here.
MACs when the base station (or access point) has multiple antennas. The upstream\textsuperscript{49} direction of a cable or passive-optical network is also a scalar-output MAC example with various residential customers all sharing a common frequency band for transmissions to a central “hub” receiver. Upstream DSL systems form an interesting vector multiple-access channel when a common cable of several homes’ twisted pairs combine (and crosstalk into one another). Yet another example is a disk drive where a single receive (read) head (or an array of such read heads) may sense the signals (simultaneously) of several previously written adjacent tracks.

Of interest in multiple-access channels are the maximum achievable data rates for each user (which may be a function of the other users’ selected data rates, complicating design and analysis). Section 2.7 provides some MAC-specific transmitter/receiver designs and analysis. Section 2.7 also includes a simplified description and construction of the MAC capacity rate region. Chapter 5 studies the vector MAC in more detail.

\textbf{Broadcast Channel (BC):} The MAC’s dual is Figure 2.26’s broadcast channel (BC) in which a single BC encoder generates all independent users’ common channel input symbol $x$. The BC also has $U$ physically separated channel outputs $y_u$. When the single input is a vector, the BC is vector broadcast. Each BC message $m_u, u = 1, \ldots, U$ is within the common single transmitter’s channel-input (“broadcast”) symbol $x$, and so there is transmit-signal “coordination” – i.e., co-encoding of all users’ message signals occurs. The BC order reverses (so user 1 at top or left, while user $U$ at bottom/right) for reasons that become apparent in Section 2.8. The $U$ BC signals’ coordinated reception is not possible. Section 2.8 provides some bounds and BC transmitter/receiver designs, while Chapter 5 further investigates the vector BC. If some user associated with a receiver location can be decomposed into more than one sub-users with cooperating receivers, then the consequent channel is no longer strictly a BC for the enlarged user set. If such a user decomposes into two isolated receivers, then the channel remains BC with an increased value for $U$.

BC examples include MAC’s opposite-direction transmissions from a service provider to a customer: For instance a “downlink” in wireless or the “downstream directions” in cable, fiber, or DSL networks. BCs also occur for television and radio where no reverse multiple-access-like channel occurs (usually).

\textsuperscript{49}Upstream is the term used for customers’ transmissions to a central site in cable and DSL, while “uplink” is the term used in wireless even though it is not a word (upstream is a word).
Interference Channel (IC): The third channel type is Figure 2.27’s interference channel (IC). Coordination of neither inputs nor outputs is possible. Each of $U$ possible receivers can attempt independent detection of one or of all the $U$ transmitted messages. In the vector interference channel, some of the inputs and/or outputs, are vectors. Interference channels occur when multiple transmitters and receivers share a frequency band (wireless) or medium (wire-line). Some examples include some types of home networks that share unregulated bands and establish links on an ad-hoc basis. Military and espionage channels may also then fall into the IC category. The IC may expand to $U^2$ users since any of the $U$ transmit locations might more generally transmit non-zero information content to any receiver. If any user decomposes into two or more sub-users that coordinate at either or both of the transmitter and receiver, then the channel is no longer strictly an IC. If such a decomposition retains independent transmit and receive locations for each component sub-user, then the overall channel remains an IC with yet a larger $U$ value (maybe in excess of the original $U^2$) for the number of sub-users.

Linear additive-noise multi-user channel models: The linear additive noise channel (particularly with Gaussian noise) merits some additional consideration. This chapter’s earlier capacity-achieving codes require asymptotically infinite block lengths, so effectively $N \to \infty$ and also $\overline{N} \to \infty$. However, there can remain spatially\textsuperscript{50} $L_x$ transmit dimensions and $L_y$ receive dimensions. The presumed asymptotically Gaussian codes ($\Gamma = 0$ dB) will thus have $N$ real dimensions or $\overline{N}$ complex dimensions (with $\overline{N} = 2$) with presumed $\overline{N} \to \infty$ (but not shown explicitly).

\begin{center}
\textbf{Strictly speaking, in the infinite dimensional context of implied codewords, Sections 2.1 - 2.5’s notation to this point would call the subsymbol channel input $\underline{\mathbf{x}}$. This creates unnecessary superfluous notation. Instead from this point forward in this chapter (and also Chapters 3, 4, and 5 where there is implied use of capacity-achieving codes), the subsymbol notation will relax to simply $\mathbf{x}$.}
\end{center}

Table 2.2 lists the possible input and output dimension sizes for each simple multi-user Gaussian channel. Simple means that $L_x = L_y = 1$ The quantity $\overline{N}$ applies also to enumerate situations with frequency-indexed dimensions (like Section 2.5.4’s multi-tone) where multi-user concepts can apply independently on each such indexed AWGN channel/tone (eliminating the need to carry the extra notation and index). Effectively then, there will be $\overline{N}$ channels of the type in Table 2.2. The matrix AWGN remains:

$$y = H\mathbf{x} + \mathbf{n}.$$  \hfill (2.225)

\footnote{Or otherwise equivalently separate user transmitters or receiver locations}
Table 2.2: Table of dimensionality for the multi-user Gaussian channel $y = Hx + n$. (When all users have the same dimensionality)

<table>
<thead>
<tr>
<th>Type</th>
<th>$x$ Number of inputs</th>
<th>$y$ Number of outputs</th>
<th>$H$</th>
</tr>
</thead>
<tbody>
<tr>
<td>multiple access</td>
<td>$U \cdot L_x$</td>
<td>$L_y$</td>
<td>$[H_U \ldots H_2 H_1]$</td>
</tr>
<tr>
<td>broadcast</td>
<td>$L_x$</td>
<td>$U \cdot L_y$</td>
<td>$\begin{bmatrix} H_1 \ \vdots \ H_{U-1} \ H_U \end{bmatrix}$</td>
</tr>
<tr>
<td>interference</td>
<td>$U \cdot L_x$</td>
<td>$U \cdot L_y$</td>
<td>$\begin{bmatrix} H_{UU} &amp; \ldots &amp; H_{U1} \ \vdots &amp; \ddots &amp; \vdots \ H_{2U} &amp; \ldots &amp; H_{21} \ H_{1U} &amp; \ldots &amp; H_{11} \end{bmatrix}$</td>
</tr>
</tbody>
</table>

For the MAC, any of Table 2.2’s $H_u$ has singular value decomposition (SVD) $H_u = F_u \cdot \Lambda_u \cdot M_u^*$ and that user’s transmitter could pre-multiply $x_u$ by $M_u$ and/or that user’s receiver could post-multiply $y$ by $F_u^*$ without loss for that user $u$. This pre- and post-processing then leaves $\varrho_{H_u}$ non-trivial dimensions to carry data optimally, where $\varrho_{H_u}$ is the rank of $H_u$. When $L > 1$, then user decomposition into $\varrho_{H_u}$ sub-users each often simplifies capacity-region construction because it leads to a single best user-decoding order instead of a larger set of order possibilities. (The BC can similarly use SVD on its users’ $H_u$ matrices.) It is also possible that each MAC input has a user-variable $L_{x,u}$ and each BC output has a user-variable $L_{y,u}$, and indeed the IC could have both. In these cases, multi-user analysis can add dummy dimensions so that the $L = \max_u (\varrho_{H_u})$ is the largest over all users. However, the dummy dimensions will carry no data and of course have no final implementation. Sections 2.7 - 2.9 directly address usable dimensions, and use sub-users for each such usable dimension. The following assumes that such dimensional adjustment occurs to the corresponding $H_u$ (or the IC’s $H_{uu'}$) matrices.

Figure 2.22 characterizes a multi-user transmission channel by the conditional probability distribution $p_{y|x}$, consistent with single-user channels. The input probability distribution also consistently is $p_x$. All other probability distributions consequently derive from $p_{xy} = p_{y|x} \cdot p_x$. Because the user inputs are independent, the input distribution also must factor as:

$$p_x = \prod_{u=1}^{U} p_{x_u}.$$  \hspace{1cm} (2.226)

The MAC and IC input probability distribution’s factorization holds for any channel model (and not just additive noise). In some BC situations, the channel input distribution may also factor, but this is not necessarily always true, although an equivalent input $p_{\nu} = \prod_{u=1}^{U} p_{\nu_u}$ will instead represent independent user inputs with $\nu_u$ in 1-to-1 relationship with non-zero-probability elements of the $x_u$, the latter viewed as a Hilbert Space.

### 2.6.2 Order, Data Rates, and Rate Regions

This subsection begins with user order, where the concept of earlier users at a particular receiver $u$’s chosen order means prior users decode first with later users as noise. This leads to a set of mutual information (or maximum data rates for every prior user to $u$, as well as $u$ with later users at receiver
u treated as noise and for whom their mutual-information maximum data rates are not of consequence. Within the aggregate of receivers (if there are more than 1, as in BC and IC or more general multi-user channel types), the minimum such mutual information of any user that necessarily decodes at any receiver limits that user’s data rate for the set of users, or that user is “decodable” at that data rate.

**Multi-User AEP Observation:** The AEP’s random construction of good codes only depends on the input distribution \( p_{x_u} \) and the bits/subsymbol \( b_u \). The code is therefore asymptotically good for any \( p_{y/x} \) channel as long as \( b_u \) is below the corresponding channel’s (with input distribution’s) mutual information.

**User Order:** At first glance, a designer might associate a single specific transmitter and receiver with each user. However, it may be possible that one or more of the (up to) \( U \) receivers can reliably detect, with vanishingly small error probability, any or all of up to \( U \) user transmitters’ messages. However, perhaps not all receivers can decode them reliably. Any receiver’s signal correspondingly decomposes into detectable and undetectable (sub-) users’ messages.

The concept that a (sub-) user is either reliably detectable (or not) is fundamental to this text’s capacity regions, and certainly consistent with the single-user capacity concept. This concept follows directly from the mutual-information chain rule, where the mutual information between any channel output and all channel inputs decomposes into a chain of conditional mutual-information quantities. A receiver reliably first detects all such conditional random-variable components (think users) in each such mutual-information term, leaving only the unconditional components to average in contribution to the specific mutual-information chain-rule term. There are many chain-rule decompositions that depend upon an order of the user/components. Any mutual information across multiple channel inputs has chain-rule decomposition(s), where each term corresponds to a maximum bit rate given the previous users in the given order. If a particular user in the given order has a \( b_u \) exceeding the chain-rule term, it is not decodable in that order. If that other user, at that data rate \( b_{i \neq u} \), is not reliably decodable in any order where \( i \) precedes \( u \), then \( b_u \) is not achievable for the given probability distribution \( p_{xy} \).

The possible orders become fundamental to multi-user transmission’s optima. Within an order, an un-detectable other-user’s signal simply becomes “noise” (the decoder design sums/integrates, effectively averages that user’s average contribution to the channel conditional probability distribution) for the receivers that are unable to detect it. Within an order, the detectable users are best set as the “given” users/components. A search over all users’ possible orders \( \Pi \) helps specify fundamental limits. Formally order defines as:

**Definition 2.6.1 [User Order and Prior-User Set]** Each receiver \( u \) can attempt to process all users by decoding all those prior to user \( u \) in a \( U \times 1 \) order vector \( \pi_u \). Receiver \( u \) decodes the user indices before \( u \) in any order \( \pi_u \) that are in the positions below \( u \)’s position in \( \pi_u \). The order can also be expressed as the discrete input/output function \( j = \pi_u(i) \) which is the position of user \( i \) counting from the bottom up in \( \pi_u \) for receiver \( u \). The vector \( \pi_u \) has specific form

\[
\pi_u = \begin{bmatrix}
\pi_u(U) \\
\vdots \\
\pi_u(1)
\end{bmatrix},
\tag{2.227}
\]

Correspondingly, \( \pi_u^{-1}(j) = i \) is the user decoded at position \( j \), with inverse order vector trivially (but for completeness) is

\[
\pi_u^{-1} = \begin{bmatrix}
U \\
\vdots \\
1
\end{bmatrix},
\tag{2.228}
\]
Those users above user u’s position in \( \pi_u \) are averaged (treated as noise) in the marginal probability distribution that the detector design uses. Receiver u removes those before. The matrix \( \Pi = [\pi_U \ldots \pi_1] \) describes all receivers’ orders in any given multi-user design.

The prior-user set \( \mathbb{P}_u(\pi) \) is the set of users \( \{j \mid \pi^{-1}(j) < \pi^{-1}(u)\} \), all which need reliable decoding prior to decoding of the specific receiver’s desired message u. \( \mathbb{P}_u(\pi) \)'s argument \( \pi \) can be any column order vector from the set \( \{\pi_1, \ldots, \pi_U\} \).

For an example, suppose \( \pi_2 = [321]^\ast \), then receiver 2 attempts to decode first user 1 before decoding user 2; receiver 2 treats user 3 as noise for all detection. By contrast, if \( \pi_2 = [231] \), then receiver 2 attempts to decode user 1, then user 3, before finally decoding user 2. Clearly each receiver can have such an order, and there are maximally \( U! \) orders for each receiver. Then also, \( \Pi \) can characterize up to \( (U!)^U \) different multi-user order possibilities for decoder design. (Fortunately, as this chapter progresses, considerable elimination of such order possibilities often trivially occurs.). For any order \( \pi \), the set \( \mathbb{P}_u(\pi) \) may or may not be reliably decodable, which depends on the vector \( b \) corresponding to the \( U \) users’ code choices/data rates.

Table 2.3 further illustrates an order table (on the left) for a 4-user channel. This is one of \( (4!)^4 \) possible orders. The prior-user sets for each receiver also appear in the bottom row of this table.

\[
\begin{array}{cccc}
\text{rcvr/user} & \pi_1^{-1} & \pi_2^{-1} & \pi_3^{-1} \\
\text{top} & 3 & 3 & 4 & 3 \\
& 4 & 2 & 3 & 2 \\
& 1 & 4 & 2 & 1 \\
\text{bottom} & 2 & 1 & 1 & 4 \\
\mathbb{P}_u(\pi_u) & \{1, 2\} & \{2, 4, 1\} & \{1\} & \{4\} \\
\end{array}
\]

\[
\Pi = [\pi_4 \ \pi_3 \ \pi_2 \ \pi_1] = \begin{bmatrix}
3 & 2 & 4 & 1 \\
4 & 4 & 3 & 4 \\
1 & 3 & 2 & 3 \\
2 & 1 & 1 & 2 \\
\end{bmatrix}
\]

\[
I_{\min} = \begin{bmatrix}
4 & 20 \\
1 & 2 \\
\end{bmatrix}
\]

Table 2.3: Order Example.

A mutual-information-like vector exists for each user’s decoding at receiver u and for that receiver’s order \( \pi_u \):

\[
\mathcal{I}_u(\Pi; p_{xy}) = \begin{bmatrix}
\mathcal{I}_u \left( x_{\pi_u^{-1}(U)}; y_u/\mathbb{P}_{\pi_u^{-1}(U)}(\pi_u) \right) \\
\vdots \\
\mathcal{I}_u \left( x_{\pi_u^{-1}(i)}; y_u/\mathbb{P}_{\pi_u^{-1}(i)}(\pi_u) \right) \\
\vdots \\
\mathcal{I}_u \left( x_{\pi_u^{-1}(1)}; y_u/\mathbb{P}_{\pi_u^{-1}(1)}(\pi_u) \right)
\end{bmatrix},
\]

where by definition (with the order \( \Pi \) and the joint distribution \( p_{xy} \) are implied but not shown explicitly
to avoid burdensome notation). The mutual-information-like quantity $I_u$ is

$$I_u \left( x_{\pi_u^{-1}(i)}; y_u / P_{\pi_u^{-1}(i)}(\pi_u) \right) \Delta \left\{ \begin{array}{ll}
\infty & \pi_u^{-1}(i) > \pi_u^{-1}(u) \\
I_u \left( x_{\pi_u^{-1}(i)}; y_u / P_{\pi_u^{-1}(i)}(\pi_u) \right) & \pi_u^{-1}(i) \leq \pi_u^{-1}(u) \end{array} \right..$$  \hspace{1cm} (2.230)

The $\infty$ mutual-information values do not affect receiver $u$ that simply treats higher-order indexed users as noise and does not decode them. The lower-order indexed users have maximum rate given by the $I_u$ value. Table 2.3 provides finite example values in red, while listing $\infty$ for the positions of no receiver concern. Table 2.3 also shows a minimum mutual-information vector for each user that considers that user must be decodable also at any receiver for which that user precedes the receiver’s desired user in the given order.

**Decodable Set:** For a given order $\Pi$ and joint distribution $p_{xy}$, receiver $u$ will be able to decode reliably users in its **decodable set**

**Definition 2.6.2 [Decodable Set and Minimum Mutual Information Vector]**

For a given $\Pi$, $p_{xy}$, and $b$, each receiver $u$ will be able to detect reliably (on average in the AEP sense) other ($i \neq u$) users in the set

$$i \in D_u(\Pi, p_{xy}, b)$$  \hspace{1cm} (2.231)

with $P_e \rightarrow 0$, When receiver $u$ can detect no other users reliably, $D_u(\Pi, p_{xy}, b) = \emptyset$, with this order.

Every multi-user channel has a minimum mutual-information vector with components

$$I_{\min,u}(\Pi, p_{xy}) = \min_{i \in \{P_{\pi_u^{-1}(u)}, u\}} \left\{ I_i \left( x_{\pi_u^{-1}(i)}; y_i / P_{\pi_u^{-1}(\pi_i)} \right) \right\},$$

and thus

$$I_{\min}(\Pi, p_{xy}) = \begin{bmatrix}
I_{\min,u}(\Pi, p_{xy}) \\
\vdots \\
I_{\min,1}(\Pi, p_{xy})
\end{bmatrix}.$$  \hspace{1cm} (2.232)

This definition helps then determine the best decodable set:

**Lemma 2.6.1 [Best Decodable Set]** When good codes (with $\Gamma = 0$ dB), given $\Pi$ and $p_{xy}$, and with

$$b \leq I_{\min}(\Pi, p_{xy}),$$

then

$$P_u(\pi_u) \subseteq D_u(\Pi, p_{xy}, b)$$  \hspace{1cm} (2.233)

and receiver $u$ reliably achieves the data rate $b = I_u(\pi_u; y_u / P_u(\pi_u))$ with order $\pi_u$.

**Proof:** When (2.234) is met, then for for user $u$ to be decodable at receiver $u$, a search over other users’ decoders $i \neq u$, and users $j \in P_u(\pi_u)$, can find if the condition

$$I_u \left( x_{\pi_u^{-1}(i)}; y_u / x_{P_{\pi_u^{-1}(i)}(\pi_u)} \right) \geq I_i \left( x_{\pi_u^{-1}(j)}; y_i / x_{P_{\pi_u^{-1}(j)}(\pi_i)} \right) \forall \{i \neq u \land j \in P_u(\pi_u)\}$$

\hspace{1cm} (2.235)
is met. The search in (2.236) may be best understood by looking at Table 2.3’s example). If (2.234) holds, then receiver \( u \) can reliably decode all the prior users that any other receiver (including \( u \) at receiver \( u \)) for the given order \( \Pi \). This is because \( \mathcal{I}_{\min} \) has the lowest rate in the necessary set of decodable users, and also because the random-coding (AEP sense) process depends for all channels at the given \( \tilde{b}_u \) only on the same input distribution \( p_{x_u} \) as in the above AEP observation. If (2.234) does not hold, then this decoding order does not support user \( u \)’s reliable detection with any codes chosen at \( b \).

QED.

Figure 2.28: User Transmitters and possible precoded other user’ set \( \mathcal{D}_u \) (BC) or as decoded other users (same set, BC, MAC, IC).

Figure 2.28 illustrates the decodable set concept. Each user desires reliable decoding of its own user; however, other users may also be decodable (and consequently cancellable) at receiver \( u \). The decodable set is a function of both the design order \( \Pi \) for all receivers and the corresponding capacity-achieving (single-user randomly designed asymptotically) codes (for the corresponding probability distribution and order) used by each user. The dependency on other receivers’ orders (so \( \Pi \) and not just \( \pi_u \)) arises because other users’ \( i \neq u \) will have bits/subsymbol \( b_i \) that may depend on other users’ receivers simultaneous ability to decode user \( i \) that depends on its position in those receivers’ orders. So while the dependency may not be on all other receivers, it is possible to depend on some other receivers.
These imply an associated data rate (bits/subsymbol) $b_u$ in the code design process for each user. Figure 2.28 also illustrates “precoders” at each transmitter that can facilitate user $u$’s decoding at all receivers; specifically Section 2.8’s broadcast and Section 2.9’s mixed multi-user channels may use precoders effectively.

**Cross-User Set:** Correspondingly, there is also a set of receivers that can reliably detect user $u$:

**Definition 2.6.3 [Cross-User Set]** The cross-user set of receiver indices at which $u$ can be reliably decoded (on average in the AEP sense) is

$$S_u(\Pi, p_{xy}, b)$$

(2.237)

with $P_e \to 0$. When no receiver $i$ can detect user $u$, including receiver $u$, then $S_u = \emptyset$.

The cross-user sets for the users $i \neq u$ will determine the possible bits/subsymbol values that a random single-user design (asymptotically under AEP) selects. A minimum of these codes’ any-relevant-receiver-associated maxima ensures that all receivers that must decode a user can do so reliably. This concept later returns in Equation (2.256). The sets $\mathcal{D}_u(\Pi, p_{xy}, b)$ and $S_u(\Pi, p_{xy}, b)$ depend on $\Pi$, the joint probability distribution (equivalently, the input distribution and the channel distribution), and $b$. These sets have thus many choices. A key to this text’s approach to capacity-region construction is search over these dependencies, especially over the set $S_u(\Pi, p_{xy}, b)$. The probability-distribution search tacitly implies an AEP-like capacity-achieving random code design for the user with other users cancelled or treated as noise.

**Mutual Information of Interest:** All three mutual-information quantities, $I(x; y)$, $I(x_u; y)$, $I(x_u; y/x_{u\setminus u})$, and $I(x_u; y/\mathcal{D}_u(\Pi, p_{xy}, b))$ (implies averaging of $S_u(\Pi, p_{xy}, b)$ within the same average as $I$) may be of interest for a particular user $u$’s analysis. The mutual information $I(x_u; y)$ is of interest to the $u^{th}$ particular user, but averages the joint probability distribution over all users. It is an average data rate bound, as mutual information, that treats all other users as “noise.” Furthermore, $I(x_u; y)$ does not account for the potential decoding of other users. So it is possible for $b_u$ to exceed $I(x_u; y)$, but never to exceed $I(x; y)$, the maximum user-bit/symbol sum. $I(x_u; y/x_{u\setminus u})$ attempts to characterize a decoder’s prior removal of other users, or specifically $u = \mathcal{D}_u(\Pi, p_{xy}, b)$. There are $2^{|U|-1}$ possible choices for the subset $u \in \{U \setminus u\}$ along with a check to see if those other users are decodable. The bound of best average conditional bits/symbol is (using Lemma refivecmin)

$$b_u \leq \max_{u \subseteq \mathcal{D}_u(\Pi, p_{xy}, b)} I(x_u; y/x_{u\setminus u}) = I_{min,u}(\Pi, p_{xy})$$

(2.238)

where the maximization is over the possible subsets $u \subseteq \{\mathcal{D}_u(\Pi, p_{xy}) \setminus u\} \subseteq U$ that contain only all those other ($u' \neq u$) users that the detector can reliably first decode. Thus, the best data-rate calculation is more complex than in the single-user case. Equation (2.238) is key, again depending on Definition 2.6.2, and allows the general capacity region specifications to be explicitly described in Subsection 2.6.4.

More formally, the capacity rate region $C(b)$ is a plot of all possible data-rate vectors ($U$-tuples) that receivers can reliably achieve. Some $C(b)$ outer boundaries represent best multiple-user code designs. Interior points represent systems using codes that have data rates (bits/symbol $b_u$) below best. Rate regions vary with the multi-user channel type, and may be complex to construct. Subsection 2.6.4’s end provides a general capacity rate region that applies to all $U \times 1$ multiple-access, $1 \times U$ broadcast, and $U \times U$ interference channels, as well as any (non-relay) multi-user channel. This chapter’s later sections simplify $C(b)$ construction for various special cases of the MAC (Section 2.7, the BC (Section 2.8), and the IC (Section 2.9).
Definition 2.6.4 [Capacity Rate Region] The capacity region \( C(b) \) of any multiuser channel is defined as the largest union (or convex hull) of all \( U \)-dimensional bits/symbol vectors \( b = [b_1, b_2, \ldots, b_U] \) such that (any) of the receiver(s) may decode its user(s) of interest with arbitrarily small error probability.

Furthermore, any point outside this region results in at least one of the receivers having error probability bounded away from 0 no matter what codes that users employ.

The capacity region \( C(b) \) is less specific than Section 2.3’s capacity \( C \) theorem for single-user channels. In this form, it is a definition; Subsection 2.6.4 still state it more directly as a theorem. The following lemma is useful and perhaps immediately obvious:

Lemma 2.6.2 [Convexity of Capacity Rate Region] The capacity rate region is convex, meaning that any convex combination of two rate tuples within the region produces another rate tuple within the region. Mathematically, if \( b_\alpha \in C(b) \), \( b_\beta \in C(b) \), and \( \alpha + \beta = 1 \), then \( \alpha \cdot b_\alpha + \beta \cdot b_\beta \in C(b) \).

Proof: The basic proof follows from the definition: The convex combination of two rate tuples corresponds to dimension (“time”) sharing of the two codes (or sub-users), and the corresponding receivers, for the same fraction of dimensions (“time slots”) as used in the convex combination. Such a system is an asymptotically allowable implementation, and thus corresponds to an achievable point within the capacity region.

A simple example helps illustrate the noise averaging or removal concept:

EXAMPLE 2.6.1 [sum of 3 Gaussian message signals and noise] Three (AEP sense) Gaussian signals are summed with noise to form

\[
y = x_1 + x_2 + x_3 + n.
\]

For the decoding orders, the following table summaries the data rate for decoding of each user.

<table>
<thead>
<tr>
<th>Order ( II )</th>
<th>( b_1 )</th>
<th>( b_2 )</th>
<th>( b_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1 \ 2 \ 3]^*</td>
<td>( \log_2 \left( \frac{1 + \frac{E_1}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_2}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_3}{\sigma^2}}{2} \right) )</td>
</tr>
<tr>
<td>[1 \ 3 \ 2]^*</td>
<td>( \log_2 \left( \frac{1 + \frac{E_1}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_2}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_3}{\sigma^2}}{2} \right) )</td>
</tr>
<tr>
<td>[3 \ 1 \ 2]^*</td>
<td>( \log_2 \left( \frac{1 + \frac{E_1}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_2}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_3}{\sigma^2}}{2} \right) )</td>
</tr>
<tr>
<td>[2 \ 3 \ 1]^*</td>
<td>( \log_2 \left( \frac{1 + \frac{E_1}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_2}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_3}{\sigma^2}}{2} \right) )</td>
</tr>
<tr>
<td>[3 \ 1 \ 2]^*</td>
<td>( \log_2 \left( \frac{1 + \frac{E_1}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_2}{\sigma^2}}{2} \right) )</td>
<td>( \log_2 \left( \frac{1 + \frac{E_3}{\sigma^2}}{2} \right) )</td>
</tr>
</tbody>
</table>

If energies are \( \sigma_n^2 = .001 \), \( E_1 = 3.072 \), \( E_2 = 1.008 \), and \( E_3 = .015 \), then with Gaussian codes (\( \mu \) Gaussian) the order \([123]^*\) corresponds to \( b_1 = 1 \), \( b_2 = 3 \), and \( b_3 = 2 \).

Technically, the concept of “time-sharing” or dimensional-sharing of two different single-user message components requires a sub-division of that user into two (or more) users. These shared codes may be different, but the combination is also a single code; this text takes that latter view.
2.6.3 Optimum multi-user detection

The optimum multi-user detector generalizes Chapter 1’s optimum single-user detector. The set of all possible coded multi-user channel inputs remains $C_D$, and contains $M = |C_D|$ possible distinct symbols, which may be a large number for the multiple-user channel. $C_D$ is thus a code for the set of all users that represents the aggregation of the individual users’ codes, which Section 2.3’s AEP arguments most generally describe by the probability density/distribution $p_D$ from which designers select good codes. The different user codes are independent and each has $\Gamma = 0$ dB, designed by AEP random-coding principles for a specified $b_u$ from the distribution $p_{x_u}$.

**Theorem 2.6.1** [Optimum multi-user detection - maximum à posteriori] The multi-user-symbol-error probability is minimum when the detector selects $\hat{x} \in C_D$ to maximize $p_{x/y}$ and is the maximum à posteriori multi-user detector. When all possible multi-user input symbol values are equally likely, this optimum detector simplifies to maximization of the conditional probability $p_{y/x}$ over the choice for $\hat{x} \in C_D$, and is the maximum likelihood multi-user detector.

**Proof:** See Chapter 1. QED.

Even though the $U$ codes are independent, the channel outputs may not (and usually are not) independent. Then, for the BC and IC, each physically distinct output $y_u$ replaces the general $y$ in the above Theorem 2.6.1.

à posteriori and à priori densities with Detection Order: The individual user’s à posteriori conditional probability distribution is conditioned on that user receiver’s decodable other users. The optimum detector then follows directly from the overall conditional distribution with known (at design time) $\mathcal{D}_u(\Pi, p_{xy}, b)$, which is here abbreviated by $\mathcal{D}_u(\Pi)$ because it is known for each input distribution and bit vector, with compliment $\overline{\mathcal{D}_u(\Pi)} \triangleq \{ U \setminus \mathcal{D}_u(\Pi) \}$,

$$p_{x_u|x \in \mathcal{D}_u(n)(y)}(x_u, x \in \mathcal{D}_u(n), v) = \int_{x \in x \in \mathcal{D}_u(n) \setminus u} p_{x/y}(x, x \in \mathcal{D}_u(n), y) \cdot d\chi \quad (2.240)$$

This integral averages $x \in \mathcal{D}_u(n) \setminus u$, for which variables no longer appear after the integration., effectively treating $\mathcal{D}_u(\Pi)$ users as “noise.” The à posteriori conditional distribution inside (2.240)’s integral derives from the two known (channel) à priori distributions (using the independence of the users’ messages) as

$$p_{x/y}(x, x \in \mathcal{D}_u(n)) = \frac{p_{y/x}(x \in \mathcal{D}_u(n), x \in \mathcal{D}_u(n), v) \cdot p_x(x \in \mathcal{D}_u(n))}{p_{x/y}(x, x \in \mathcal{D}_u(n))} \quad (2.241)$$

Equivalently, the conditional individual ML detector for $x_u$ maximizes the integral of (2.241)’s left numerator term (assuming all $x_u$ possible values are equally likely)

$$\hat{x}_u = \arg \max_{x_u \in C_D} \int_{x \in x | \mathcal{D}_u(n)} p_{y/x}(x \in \mathcal{D}_u(n), x \in \mathcal{D}_u(n), v) \cdot p_x(x \in \mathcal{D}_u(n)) \cdot d\chi \quad (2.242)$$

Similarly the MAP maximizes (2.240). When the receivers do not coordinate, $y \rightarrow y_i$ in the above for receiver $i \in \{ U \}$. The expressions in Equations (2.243) and/or (??) then permit calculation of individual user error probabilities later in this subsection for any order $\Pi$ and of course also for a specific $p_{xy}$.

**Overall Error Probability:** The overall error probability for detecting all users is (as always)

$$P_e = 1 - P_e = 1 - \sum_{i=0}^{\lfloor C_D \rfloor-1} P_{e/i} \cdot p_{x(i)} \quad (2.243)$$

51 From Chapter 1, see Subsections 1.1.4, 1.1.6, and 1.3.2 specifically: $P_{e/i} = \sum_{v \in \mathcal{D}_i} p_{y/m(v, i)}$ where the $i$ indexes the potential messages for a specific user in this case. The present development avoids this further detail to avoid confusion of indexing for specific messages with indexing for specific users, the latter of which each have a set of specific messages.
This single error probability has meaning for all multi-user channels, but in particular may be of most interest in the MAC where a single receiver detects all users. However, this $P_e$ is not necessarily attainable unless there is only one (coordinated) receiver.

**Individual-User MAP:** The individual user’s MAP detector depends on the order matrix $\Pi$:

$$\hat{x}_u = \arg \left\{ \max_{x_u, n_u} p_{x_u|y_u, n_u} \right\},$$

which is (as always, even in single-user case) an implied function of $b$. The multi-user case admits the

$$p_{c,u}(\Pi_{MAP}) = \max_{\Pi} \left\{ \sum_{i=0}^{\text{C}_x} P_{c,u,i}(\Pi) \cdot p_{u,i} \right\}.$$  \hspace{1cm} (2.245)

Then receiver implements a MAP (or ML because all long-length codewords are equally probable for these types of codes) by using $p_{x_u|y_u, n_{MAP,u}}$. The best order $\Pi_{MAP,u}$ varies with the receiver $u$. Thus, a MAP detector for one user may best use an optimum order that makes another user’s performance degrade. Such user trade-off thus is evident in $P_{e,u}$ just as it is in $C(b)$. A solution is to use a weighted $P_e$ so minimize over the $\Pi$ the sum

$$P_e = \sum_{u=1}^{U} w_u \cdot P_{e,u},$$

and use this order for all receivers. Again, once the best single order is known, the remaining receiver designs follow directly for the applicable probability distribution. For the MAC, this weighting is not necessary.

For the BC and IC, the best detectors for $y_u$ also detects first all other users that are decodable (over any order), so effectively $D_u(p_{y_u|\Pi_{MAP}, p_{xy}, b})$. Thus, each receiver optimally detects each of the inputs that it can reliably, even though only user $u$ may be of interest at receiver $y_u$ in terms of message delivery. The error probability then becomes a function of $u$.

The individual decoders error probabilities on an AWGN channel’s still follows the NNUB approximates as

$$P_e \leq N_e \cdot Q \left( \frac{d_{\text{min}}}{2\sigma} \right),$$

where the number of nearest neighbors, $N_e$, is the same, but the $\sigma^2$ will include users treated as noise (later in the optimum applicable order).

The situation of independent channels greatly simplifies detectors:

\begin{definition}[Crosstalk-Free Channel] A crosstalk-free multi-user channel (CFC) has a conditional probability distribution that satisfies

$$p_{y|x} = \prod_{u=1}^{U} p_{y_u|x_u}.$$ \hspace{1cm} (2.248)

That is, the channel probability distribution factors into independent terms for each of the users. When the channel is not CFC, it has crosstalk.
\end{definition}

A receiver for a CFC trivializes into an independent set of single-user receivers:
**Lemma 2.6.3 [Independent Detection]** The ML decoder for the CFC is equivalent to a set of independent optimum decoders for each individual user.

The proof follows trivially from inspection of $p_y/x$, which factors into $U$ independent terms that separate ML decoders independently maximize when $(2.226)$ holds.

Independent detection has a separate receiver for each user, which can enormously simplify detector implementation. Crosstalk’s absence renders the channel a set of independent single-user channels. Such a receiver is analogous to Chapter 3’s symbol-by-symbol detector, except now “user by user.” The CFC assumption need not always occur though especially when design restrictions do not allow dimensionality increase (that typically requires larger bandwidth, longer delay, and/or more antennas).

The CFC channel’s overall (all users) error probability is$^{52}$

$$P_e = 1 - \prod_{u=1}^{U} P_{e,u},$$

and the overall error probability can never be less than any single user’s error probability

$$P_e \geq P_{e,u} \forall u \in \{1, ..., U\}.$$  \hspace{1cm} (2.250)

### 2.6.3.1 MAC Detection Characterization for the linear multi-user AWGN

The linear multi-user AWGN is

$$y = Hx + n.$$  \hspace{1cm} (2.251)

For MAC detection of only user $u$, it may be that the overall minimum distance for all users is too small. That is, a single fixed value for $x_u$ may correspond to the two multi-user codewords that determine the overall $d_{\text{min}}$. So, there is a **user-specific minimum distance**

$$d_{\text{min},u} = \min_{x_u \neq x'_u} \| H(x - x') \|.$$  \hspace{1cm} (2.252)

Trivially,

$$d_{\text{min},u} \geq d_{\text{min}}$$  \hspace{1cm} (2.253)

with equality if and only if any codewords in $C_x$ corresponding to the overall $d_{\text{min}}$ also correspond to different values for the $u^{th}$ user’s symbol contribution. Also,

$$\min_u d_{\text{min},u} = d_{\text{min}}.$$  \hspace{1cm} (2.254)

This more clearly illustrates how it is possible for a detector extracting a single user to have better performance than one that extracts all users. It is possible for $d_{\text{min},u}$’s modification to $d_{\text{min},u}/u \setminus u$ if users in $u \setminus u$ are given or removed, as per the following example.

---

$^{52}$ A receiver index $l$ is not shown but would normally be used in a BC or IC, except it becomes superfluous in the CFC case.
EXAMPLE 2.6.2 [2 users in 2 dimensions] Two MAC users both use 4-QAM with identical symbol periods, but different energy levels. The combined constellation when these two signals are added is shown in Figure 2.29. The users’ symbol contributions are not orthogonal, and both occupy both dimensions. The first signal has the smaller \(d_{\text{min},1}\), which is equal to the overall \(d_{\text{min}}\). The second signal has a larger \(d_{\text{min},2}\) and whose values correspond to the distance between the 4 colors in the figure.

A detector for both signals simply selects the closest of the 16 points to a received two-dimensional value \(y\) and then outputs the corresponding two messages. The error probability for the first user and for the overall detector are approximately the same. However a detector for only the second user (the different colors) clearly would perform significantly better if noise well below either \(d_{\text{min}}\). In this simple example, a detector that assumed user 1 was Gaussian noise has the same decision regions as the optimum detector for user 2, but such simplification is usually not the case. In this case \(d_{\text{min},2>/1 > d_{\text{min},2} \) but \(d_{\text{min},1/2 = d_{\text{min},1}}.\)

2.6.4 A General Multi-User Capacity Region

The general multi-user channel’s capacity-region specification can require search over as many as \((U!)^U\) possible user orders \(\Pi\). Such a search may be very complex\(^{53}\). Some channels’ searches need less than \(U!\) order possibilities, as will be the case for the MAC (Section 2.7), BC (Section 2.8), and the IC (Section 2.9).

However, the general capacity-region specification proceeds in 3 separate (possibly very complex) optimizations/searches:

**Search 1: Finding the order- and input-distribution-dependent mutual-information vector**

For a general multi-user channel with fixed \(py/x\), the information-like quantity is

\[
\mathcal{I}_u(x_u; y_i/\pi_u) \Delta \left\{ \begin{array}{ll}
\infty \\
I_u(x_u; y_i/\pi_u) \\
\pi_i^{-1}(u) > \pi_i^{-1}(i) \\
\pi_i^{-1}(u) \leq \pi_i^{-1}(i)
\end{array} \right. \]

\(2.255\)

\(^{53}\)The value of \(U\) may itself vary with the particular user in channels with physically separated receivers, so the maximum such \(U\) value is implied here in searches, which avoids some burdensome notation.
and again tacitly a function of $\Pi$ and $p_{xy}$ to avoid burdensome notation. This is the same information-like quantity as in (2.230) with reversed $u$ and $i$.

Similarly, the minimum mutual-formation quantity arrives through the minimization process (through a different path because of the index reversal but nonetheless at the same quantity as (2.232)) as

$$I_{\min,u}(\Pi, p_{xy}) = \min_{i \in \{p_{u-1,i}, u\}} \{ J_i(x_u; y_i/p_u(\pi_i)) \},$$

and thus as in (2.233) - (2.233):

$$I_{\min}(\Pi, p_{xy}) = \begin{bmatrix} I_{\min,u}(\Pi, p_{xy}) \\ \vdots \\ I_{\min}(\Pi, p_{xy}) \end{bmatrix}.$$  

Any $b_u \geq I_{\min,u}(\Pi, p_{xy})$ cannot be reliably decoded, as needed for successive decoding, by at least one receiver, by the single-user capacity theorem. Thus again

$$b \preceq I_{\min}(\Pi, p_{xy}).$$

All elements of this vector can be achieved at all corresponding receivers in $S_u(\Pi, p_{xy})$ that decode its corresponding elements for the given order matrix and input distribution. While the number of possible $\Pi$ is less than $(U!)^U$, roughly $\left( \frac{U^U}{2} \right)^U$ according to (2.256)-minimization’s order restriction, further search reduction is possible to just $U$ orders when the next step in (2.259) refines results. For Sections 2.7 and 2.8 on MAC and BC respectively, (2.256)’s minimization step is not necessary. Section 2.9’s IC will need it, and the proof that only $U$ order needs consideration appears there.

**Search 2: The order optimization search to achievable region** There are as many as $(U!)^U$ orders and various ways to discretize, enumerate, and search over $p_{xy}$ possibilities, and so there are correspondingly many points $b(\Pi, p_{xy})$. An achievable rate-region construction permits dimensional(time)-sharing corresponding $b_{\min}(\Pi, p_{xy})$ values for the possible order matrices $\{\Pi\}$: equivalently the convex hull of the region formed by the set of points over all orders for any given allowed $p_{xy}$ is achievable as the order-and-distribution-dependent achievable rate region:

$$A(b, p_{xy}) = \bigcup_{\Pi} \mathcal{A}(b, p_{xy}).$$

Any point outside this convex hull has at least one user’s data rate, that for the given $p_{xy}$ and channel $p_y/x$, corresponds to at least one receiver that cannot decode that one user, no matter the order. Such a point then violates a single-user capacity limit for all orders and the given input $p_{xy}$. This convex hull step on orders tacitly permits removal from consideration many superfluous orders in the search in (2.256); and indeed as developed more fully in Section 2.9, only $U$ possible $\Pi$ values are necessary. This follows because the order at each receiver where its desired user is last (at top) is also always the largest $b_u$ of interest. The remaining indices in column order $\pi_u$ are chosen from $\Pi$ such that all necessarily distinct columns create $b_u(i)$ that can be dimension-shared with $b_u$ in any convex-hull proportion. To enumerate all contributions, the process can reduce orders searched to those where the lower $U-1$ rows of $\Pi$ have columns that are any single distinct set of mutually exclusive permutations.

**Search 3: Finding the best input probability distribution** Finally then,

$$C_{\text{general}}(b) = \bigcup_{\text{allowed}(p_{xy})} \mathcal{A}(b, p_{xy}).$$

---

54This $(\frac{U^U}{2})^U$ is a coarse approximation that recognizes that roughly half the other users occur after any given user in any given receiver and order and these treat-as-noise users’ subset order does not change results.
where the convex hull over all possible input probability distributions allows input distributions for each independent user that each must satisfy the particular user’s input constraints.

These 3 above steps are the following theorem’s proof:

**Theorem 2.6.2 [The Multi-User Capacity Region Theorem]** The multi-user capacity region for any multi-user channel with $p_Y/x$ is in Equation (2.260).

**Best Rate Sum:** The maximum sum $b_{max}$ occurs when the plane $\sum_{n=1}^{U} b_n = K$ is tangent to $C(b)$’s boundary, which necessarily must occur for at least one $b$ since the region is convex.

The search over probability distribution depends upon the allowed probability distributions. Clearly with a single or small number of allowed distributions, the search proceeds readily by trial of each. The continuously parametrized continuous probability density function may complicate search. The designer will need to quantize the range of parameters. A $N_p \times 1$ vector of such parameters is $r_x$, each with range parameter having

$$r_{x,\min,n} \leq r_{x,n} \leq r_{x,\max,n} \quad (2.261)$$

A reasonable quantization step $\Delta_{x,n}$ is chosen by partitioning the range with judgement to cover the range of each parameters’ extent. An outer loop for the evaluation of Equation (2.260) then becomes

$$\bigcup_b \left\{ \sum_{r_{x,\max,1}} \sum_{r_{x,\max,2}} \cdots \sum_{r_{x,\max,N_p}} A(b, p_x(rx)) \right\} \quad (2.262)$$

While the search may be tedious, it is otherwise a straightforward “outer loop” to the internal calculation of the achievable region for each given $p_x$ that corresponds to indices in the sum cascade of (2.262). The search collapses to a single sum for the BC.

**Energy Enumeration Approximation:** The Gaussian channel probability distribution search reduces to a search of the possible $R_{xx}$ (zero mean because a non-zero transmit average value adds no information to data transmission). Whether for individual user inputs (MAC and IC, and more general channels) or a single aggregate input as for the BC, the search can be simplified to searching over energy parameter ranges. When each user’s input is a scalar, this simply corresponds to $\Delta_{x,n} = E_u$, $n = 1, ..., U$. When the input has $L_x > 1$, the search becomes a search over the sub energies $\sum_{l_{x,u}} E_{u,l} \leq E_u$ for each $u$ (again in the BC case, this is a single search over the dimensions for the BC’s single input). The $R_{xx}(u)$ in these cases is formed as

$$M_u \cdot \text{Diag} \{ E \} \cdot M_u^*$$

where $M_u$ arise from the singular value decomposition of $H_u$ for the MAC, of $H$ for the BC, and of $H_{iu}$ $i = 1, ..., U$ for the IC. This result follows from vector coding’s maximum-rate choice of channel decomposition in Section 2.3. In the IC case, a full search presumably needs further search over every possible cartesian product of the $M_{iu}$ choices for each user’s energy. Some simplification though can occur for various order choices as per Section 2.9. The more general Gaussian MU channel’s search will parallel the search described for the IC.

2.6.4.1 Admissibility

While $C(b)$’s construction provides design guidance, an actual system often desires only the knowledge that if a specific rate vector $\beta \in C(b)$. That is, if the specific vector $\beta$ has a design that is admissible. The test of $\beta$’s admissibility is often much simpler than construction of the entire $C(b)$.

This leads to Chapter 5’s Gaussian-channel alternative designs that attempt find for any $\beta$, the minimum energy sum that satisfies all users’ energy constraints. Equivalently, maximization of a weighted (for each user) rate sum for a given energy constraint is often used to trace the Gaussian Channel capacity region over different sets of user weights. The existence of such a set of weights exists (without finding them all), then confirms the $bvec$ is feasible (for reliable decoding).
2.6.5 The Matrix AWGN Gaussian Channel

The matrix AWGN (with some intrasymbol matrix filtering $H$) remains

$$y = Hx + n.$$  \hspace{1cm} (2.264)

**Lemma 2.6.4** [Gaussian Inputs’ Optimality on Multi-User AWGN Channels]

For any linear AWGN $y = Hx + n$ where $n$ is Gaussian with autocorrelation $R_{nn}$ and given input autocorrelation $R_{xx}$, the optimum input distribution for each and every user $x_u \in U$ is Gaussian, and thus achievement of $C_{awgn}(b)$ boundary points also has all users with Gaussian input distributions.

**Proof:** Theorem 2.3.2 already established the Gaussian input’s single-user matrix AWGN optimality for the mutual information $I(x;y)$. Lemma 2.3.4’s Chain Rule applies to any order $\pi(u)$ of users. The last user $U$ in such an order corresponds to a single-user AWGN for any given specific values of $x_{U-1}, ..., x_1$, so $p_y|x(x_U, [x_{U-1}, ..., x_1], v)$, which has maximum value over $p_x$ distributions that is for Gaussian $p_{x_U}$. Since $x_U$ is then Gaussian, then an induction to users $U - 1...1$ applies in success so that they all are Gaussian. This holds for any $R_{xx}$, including a given specified autocorrelation.

Equation (2.256) is a mutual information and thus has maximum when $p_x$ is jointly Gaussian over all users (and infinite values are trivially not of interest). At this point with all Gaussian distributions, Equation (2.256)’s minimization is over the possible receiver points for a mutual information but does not change the fact that $I_{min,u}(\Pi, p_{xy})$ remains a mutual information, and thus is a highest data rate for any $R_{xx}$ that satisfies the other decodability restrictions this section just addressed. Clearly Equation (2.258)’s stacking of user bits into a bit vector is just notational and does not compromise Gaussian optimality.

Equation (2.259)’s convex-hull union operation, over all orders, introduces dimension sharing. The dimensional sharing can be viewed as decomposing the vectors $\beta_1$ and $\beta_2$ each into two vectors, doubling dimensionality, so that

$$\beta_1 \rightarrow \begin{bmatrix} \beta_{1,\theta} \\ \beta_{1,1-\theta} \end{bmatrix}$$  \hspace{1cm} (2.265)

$$\beta_2 \rightarrow \begin{bmatrix} \beta_{2,\theta} \\ \beta_{2,1-\theta} \end{bmatrix}.$$  \hspace{1cm} (2.266)

Lemma 2.3.3 however, with expansion of $\beta_1$ and $\beta_2$, as in Equations (2.265) and (2.266), proved that for any $\theta$ and the consequent $R_{xx}$, the entropy maximizing distributions remain Gaussian, and thus so does the consequent mutual information remain maximum. Thus, Gaussian $p_x$ continues to maximize the boundary/extreme points in (2.259)’s $A(b, p_x)$. The capacity region $C(b)$ in (2.260) then maximizes over all possible (allowed by $\Pi$ earlier and as expanded by dimension sharing) auto-correlation matrices $R_{xx}$ such that any applicable energy constraints are also satisfied, but all remain Gaussian in that optimization of autocorrelation possibilities.

The general search is complex with much bookkeeping on $\Pi$, allowed $p_{xy}$, dimension-sharing, etc., but often simplifies as subsequent sections show. This text expands and slightly simplifies the definition of a degraded channel with respect to use elsewhere for the specific Gaussian-channel case that will be of interest to the next 3 sections. It is equivalent to more general “Markov-Chain” degraded definitions with the matrix AWGN.

279
Definition 2.6.6 [(Subsymbol) Degraded Multi-User Gaussian Channel] With $L_{x,u} = L_{y,u} = N = L = 1$, a (subsymbol)-degraded AWGN multi-user channel has matrix ranks for $H$ and/or $R_{xx}$ that are $\varrho_H$ and $\varrho_{R_{xx}}$ respectively, such that

$$\min \{ \varrho (R_{xx}), \varrho (H) \} < U . \tag{2.267}$$

Otherwise, the channel is non-degraded. The literature often omits the word “subsymbol,” but it is tacit in definitions of degraded channels.

This degraded multi-user channel basically forces users to share subsymbol dimensions, or equivalently the system does not have sufficient dimensions for to orthogonally multiplex different users’ symbols each on to their own set of dimensions. Thus some users consequently must be viewed as noise for the decoding of other users, within a subsymbol, and thus cause the “degradation.” Dimensional-sharing over many subsymbols (if allowed) can make any channel non-degraded, which is why the adjective use of “subsymbol” applies.

2.6.6 Multi-User Channel Control

The multi-user channel designs so far tacitly imply a form of external coordination or control. For the single-user case, presumably a common design understanding (or standard) exists so that the transmitter and receiver jointly agree on the code, modulation, times of use and other necessary assumptions. The multi-user case also implies a common design assumption, but goes further to imply, even in the IC or more general multi-user configurations, that there is some common agreement on the codes, data rates ($b$ elements), input probability distributions (or perhaps more meaningfully expressed as spectra use and time/space dimensions assigned). While input data may not be shared, or outputs not co-processed, in some multi-user channels, assignments of codes/dimensions may need coordination. There are basically four approaches to this

**Random Access** Random access systems make design assumptions based on dimensional availability probabilities and collision detection and retransmission.

**Central Management** These systems have a policy controller that informs each transmitter what dimensions and codes to use (and when).

**Consensus Management** These systems (often also called “blockchain”) derive a consensus of all users through overhead messages passed between users. There may be an elected leader that may change with time through the consensus process. This is often called RAFT or resource aggregation fault tolerance.

**Distributed Management** These systems have policies implemented by each transmitter that are functions only of each user’s measurements of the channel and its own performance.

2.6.6.1 Random Access
to be added

2.6.6.2 Central Management or Network Slicing
to be added

2.6.6.3 Consensus Management or Block Chain
to be added
2.6.6.4 Distributed Management

to be added
2.7 Multiple-Access Channel Capacity Rate Regions

This section investigates further the MAC’s capacity rate region. The general capacity region in Section 2.6.3’s Equations (2.256) - (2.260) simplifies considerably through the use of bits/subsymbol sums for the MAC with any particular input and channel probability distributions, \( p_x \) and \( p_y | x \). Often the literature and this text abbreviate the term “bits/subsymbol sum” with the less accurate “rate sum” because the two differ only in scaling by the symbol rate. This rate-sum formulation exploits the MAC’s single receiver to reduce (2.258)’s complexity. Subsection 2.7.1 uses rate sums to address the MAC capacity region \( C(b) \)'s simplified calculation and specification, while Subsection 2.7.2 specializes to the Gaussian MAC. Section 2.7.4 also extends Section 2.5’s water-filling to compute directly the filtered Gaussian MAC’s maximum rate sum. Subsection 2.7.5 expands to continuous time-frequency MACs.

2.7.1 The General Multiple-access Rate Region

Simplest MAC analysis focuses on rate sums

\[
    b_u = \sum_{u \in \{ u \}} b_u . \tag{2.268}
\]

**notation:** The notation \( \{ u \} \), or sometimes more simply \( u \) when understood in context, refers to a set of users so that \( \{ u \} \subseteq \{ U \} \). Thus (2.268) adds users data rates from users \( u \in \{ u \} \) to compute the rate sum for the set of users in \( \{ u \} \). Equation (2.268) applies to any subset \( u \subseteq U \). The full rate sum has simple notation \( b = \sum_{u=1}^{U} b_u \). The notation \( \{ x \} \) also refers to the set containing the subsymbol vector \( x \)'s elements. Again, sometimes simply \( x \) instead of \( \{ x \} \) appears when the context is clear. The matrix multiplication expression \( y = H x \) clearly uses vectors, while indices on sums, or limits on integrals, often refer to the sets of elements contained in \( x \). The overall rate sum has many equivalent notational specifications such as

\[
    b = \sum_{u=1}^{U} b_u = \sum_{u \in \{ U \}} b_u = \sum_{u \in \{ U \}} b \pi(u) = 1^* b = \sum_{u=1}^{\vert U \vert} b_u . \tag{2.269}
\]

A similar expression for any \( b_u \leq b \) also follows directly for any user \( u \in \{ u \} \). There is also the bits/subsymbol vector \( b = [b_U, b_{U-1}, ..., b_1]^* \), and there is a subset of bits/subsymbol/user values \( \{ b_u \} \). The highest indexed user is at the top (or left) of the vector. Different order functions \( \pi(u) \) will lead to different user stackings within vectors \( x \) and \( b \), but the same rate sums for the same user subsets. The MAC’s matrix order \( \Pi \) simplifies to a single column vector

\[
    \Pi = \pi = \begin{bmatrix} \pi(U) \\ \vdots \\ \pi(1) \end{bmatrix} , \tag{2.270}
\]

or equivalently \( p_{i_u}(i) = \pi(i) \forall u = 1, ..., U \). The MAC detector treats all individual users’ inputs as a single input vector

\[
    x = \begin{bmatrix} x_U \\ x_{U-1} \\ \vdots \\ x_1 \end{bmatrix} . \tag{2.271}
\]

The multi-user channel’s mutual information is the same as channel’s single-user mutual information from all users in \( x \) to \( y \) when allowed input distributions \( p_x \) are the same. The MAC inputs are independent.

\[
    p_x = \prod_{u=1}^{U} p_{x_u} . \tag{2.272}
\]
This need not be true for the single-user matrix AWGN, which makes the MAC a special more restricted case. Clearly then, the rate-sum $b \leq \mathcal{I}(x; y)$, which is the single-user maximum with no $p_x$ restriction, but also the MAC maximum when the $p_x$ search space narrows to satisfy (2.272).

**Partial rate-sums and MAC macro users:** As in (2.222) and (2.238), an individual $b_u$ may exceed $\mathcal{I}(x_u; y)$ when the detector decodes other users first, an important observation for constructing MAC capacity regions. A best case is that the MAC detector knows all other users' input symbols. Then

$$b_u \leq \mathcal{I}(x_u; y/x_i \in \{U \setminus u\}) \quad .$$

Subsets $\{u\} \subseteq \{U\}$ have mutual information with MAC channel output $y$ that is

$$\mathcal{I}(x_u; y) \forall u \subseteq U \quad .$$

These subsets effectively become Section 2.6's macro users. There are $2^U$ possible such macro users or subsets, including the null set $\emptyset$, which of course has $\mathcal{I} = 0$. Equation (2.274)'s mutual information represents the partial maximum rate sums over subset of user indices in $u$ with unknown remaining users $i \in \{U \setminus u\}$, so $\mathcal{I}(x_u; y)$'s calculation averages over this latter user set.

These (macro-user) partial rate sums are maximums over the excluded users' $i$ averaged statistics, so the remaining users' effect is as if they are “noise.” A higher partial rate sum can be achieved for any (macro-) user group $u$ by conditioning the mutual information on all the remaining users, $\{U \setminus u\}$. This conditioning essentially means the detector treats all the other remaining users as known, effectively eliminated or canceled in their effect on $u$'s detection. Knowledge of the set $\{U \setminus u\}'s$ specific actual values can only help an optimal detector and thus increases the rate sum bound. $\mathcal{I}(x_u; y) \leq \mathcal{I}(x_u; y/\{U \setminus u\})$ because $\mathcal{I}(x_u; y/\{U \setminus u\})$ cancels the other-users' influence, with equality if and only if users $u$ and $y$ are jointly independent of the $\{U \setminus u\}$. Since both (2.273) and (2.274) are mutual information quantities, they each represent maximum (sum) data rates for users in $u$ under their respective conditions. Then the following bound holds for all $2^U$ possible macro users $u$ and any given input probability distribution and channel:

$$b_u \overset{\Delta}{=} \left\{ \sum_{u \in u \subseteq U} b_u \right\} \leq \mathcal{I}(x_u; y/\{U \setminus u\}) \quad .$$

Equations (2.273), (2.274), and all the instances of (2.275) over user subsets, simplify the achievable MAC multi-user rate-region specification for any given input distribution $p_x$. This process averts use of Section 2.6's sets $S_u(\Pi, p_{xy}, b), \mathcal{R}(\Pi, p_{xy}, b)$, and $P_1(\Pi)$.

**Chain-Rule Use and Implications:** The proof that (2.275)’s achievable region is the bound on $b$ for any given probability distribution uses single-user mutual information bounds from Lemma 2.3.4 that updates here in the multi-user context to the multi-user chain rule of information theory:

**Theorem 2.7.1 (Chain Rule)** The multi-user mutual-information chain rule is

$$\mathcal{I}(x; y) = \mathcal{I}(x_1; y) + \mathcal{I}(x_2; y/x_1) + \ldots + \mathcal{I}(x_U; y/x_1, x_2, \ldots, x_{U-1}) \quad (2.276)$$

$$= \sum_{u=1}^{U} \mathcal{I}(x_u; y/x_1, x_2, \ldots, x_{u-1}) \quad (2.277)$$

The proof appears in Lemma 2.3.4, but here with each dimension now as a user.

The chain rule applies for all user orders $\pi$ with the same rate-sum bound $b \leq \mathcal{I}(x; y)$. The chain rule also applies within any user subset $u$ and their partial (macro-user) rate sum bounds are the same, with remaining users given, $\mathcal{I}(x_u; y/x_{U\setminus u})$. Thus, a simplification of Section 2.6.3’s numerous-situational enumeration instead considers only (2.275)’s possible rate sums for different macro-user subsets $u \subseteq U$. 283
**MAC Successive Decoding:** Variable π re-indexes users so that, without loss of generality, the MAC analysis can assume that \( u = \{ u + 1, ... , U \} \) while \( U \setminus u = \{ 1, 2, ... , u \} \). The chain rule suggests a successive decoding strategy\(^{55}\) that detects users within \( \{ U \setminus u \} \) prior to user \( u \) and then removes their distorting effect on \( w \)'s detection. This successive-decoder detector averages over users \( i \in \{ u \setminus u \} \), so experiences them as noise, but not the prior-decoded users. Thus, the MAC detector for a specific channel output \( y = v \) progresses from detecting user 1 first as if all others are noise with marginal \( p_y / x_1 (x_{1}, v) \), then proceeds to user 2 with higher indexed users \( i > 2 \) as noise, but user 1’s detected symbol \( \hat{x}_1 \) specifically inserted into marginal \( p_y / (x_1, x_2) ([\hat{x}_1, \chi_2], v) \), and so forth sequentially to user \( U \)'s detection with (non-marginal) \( p_y / x((\{ x_1, ..., \hat{x}_{U-1}, x_U \}, v) \).

Code design with successive decoding also treats higher-indexed users as “noise,” i.e. the corresponding decoder averages the marginal distributions for those users \( i > u \). This code design treats lower-index \( i < u \) users as if they are not present. User 1 would likely have a lower data rate than user \( U \), for which user \( U \)'s design only considers the non-user “noise.” Re-ordering with various \( \pi \) choices allows the designer to trade users’ possible \( b_u \) values. When the single receiver correctly decodes user 1 (with \( P_{e,1} \to 0 \)), \( \hat{x}_1 \to x_1 \) with a good capacity-achieving code. Thus in theory, successive decoding experiences no previous-user-decoded-incorrectly error propagation. Successive decoding also achieves the rate sum \( I(\chi; y) \). There are \( U! \) orderings, and thus \( U! \) data-rate \( U \)-tuples that achieve this same \( I(\chi; y) \). Achievable rate-sum calculation computes \( I(\chi; y) \) only once for all these orders. However, finding an acceptable trade between individual data rates requires full rate-region construction, where order variation helps construct \( C(b) \). A set of \( 2^U - 1 \) nontrivial corresponding mutual-information quantities bound these macro-user partial rate-sums (bits/subsymbol-sum) through the hyperplanes \( \sum_{j=1}^{U} b_j = I(x_{u \subseteq U}; y) \).

**Dimensional sharing and sub-users:** When \( u = U \), the macro-user becomes equivalently a single user. In effect macro user \( U \) (all users) partitions into the original users, now from this perspective as sub-users. All these users have the same sum-rate (independent of \( \pi \)). This order-independent rate-sum also occurs within each proper subset \( u \subset U \). For \( u = U \), the original \( i = 1, ..., U! \) orders in the set \( \{ \pi \} \) each have a rate tuple \( b_i \) that sums to the same \( I(\chi; y) \), for which time/dimensional sharing creates a \( b \) as

\[
\mathbf{b} = \sum_{i=1}^{U!} \theta_i \cdot \mathbf{b}_i
\]  

(2.278)

where

\[
\sum_{i=1}^{U!} \theta_i = 1
\]

(2.279)

\[
\theta_i \geq 0 \quad \forall \, i = 0, ..., U!
\]

(2.280)

is also achievable through simple use of the corresponding codes in the proportions \( \theta_i \).

**Rate Regions:** This corresponds to partitioning some/all original users into sub-users over multiple subsymbols with use fractions \( \theta_i \). Each sub-user set is separately decoded for each such macro-to-sub-user set’s corresponding MAC usage on \( \theta \times 100 \) percent of the subsymbols. The remaining rate-region calculation involves finding the \( 2^{U-1} \) proper subsets \( \{ u \} \subset \{ U \} \) corresponding to all maximum partial bits/subsymbol sums and hyperplanes (equivalently an order search within each macro group), and finding the interior of their intersection, using also the hyperplanes defined by \( b_u = 0 \) since \( b_u \geq 0 \). This process recursively observes that the chain rule similarly applies to all the user subsets \( u \). There are thus \( \sum_{u=1}^{U} (U) = 2^U - 1 < U! \) (when \( U \geq 2 \)) partial bits/subsymbol sums to compute, which may be a substantial reduction in calculations with respect to Subsection 2.6.3’s all-\( \Pi \)-matrices search procedure as \( U \) becomes large.

First a MAC rate region for any input distribution is formally:

---

55 This also is called successive detection.
**Theorem 2.7.2 (Multiple-Access Channel Rate Region)** The general multiple-access rate region $\mathcal{A}(b, p_x)$ for a given user input distribution $p_x$ is the set of all $U$-dimensional rate vectors $b$ that satisfy:

$$\mathcal{A}(b, p_x) = \left\{ b \mid b \in \bigcup_{u \in U} \text{conv} \left\{ u \subseteq U \right\}, 0 \leq \sum_{u \in \{U \setminus u\}} b_u \leq I(x_u; y|x_{i \in \{U \setminus u\}}) \right\}. \quad (2.282)$$

**Proof:** The successive-decoding discussion leading to this theorem describes achievement of any $b' \in \mathcal{A}(b, p_x)$ point with independent single-user capacity-achieving codes independently. Through that same discussion, any point outside $\mathcal{A}(b, p_x)$ that has a rate sum greater than the boundary then violates a single-user mutual-information bound for at least one user corresponding to that boundary, with other users in $\{U \setminus u\}$ given. QED.

The hyperplane’s intersection occurs through the convex hull in Equation (2.282) that essentially runs through all orders, but recognizes for the MAC that the “users’ order” used in computing their partial rate sum does not change the rate sum, nor does the “order” within the given users $\{U \setminus u\}$ excluded for the rate-sum’s computation, thus simplifying that computation to one calculation per different macro-user group. The subsequent capacity region simply optimizes over all possible multi-user input distributions $p_x$:

**Lemma 2.7.1 (Capacity Region for the Multiple Access Channel)** The MAC capacity region $C(b)$ is the maximum, or precisely the union’s convex hull of the achievable regions $\mathcal{A}(b, p_x)$, over all the possible joint input probability distributions $p_x$ that the MAC permits.

$$C(b) = \left\{ b \mid b \in \bigcup_{p_x} \text{conv} \mathcal{A}(b, p_x) \right\}, \quad (2.283)$$

where the convex hull (special union operation in (2.283)) appears in (2.278) - (2.281).

**Proof:** Any possible input distribution leads to a valid rate region according to Theorem 2.7.2. Such regions’ union is thus also achievable for at least one of the distributions. This union’s convex hull corresponds to points that may time-share (or dimension-share sub-user decompositions) some of the different possible probability distributions – assuming such a sub-user partitioning is within the allowed input probability-distribution set.

By contrast, any point outside the region $C(b)$ violates at least one single-user capacity theorem and so is not achievable with arbitrarily small error probability. QED.

**Simple MAC Achievable Region Illustration:** Figure 2.30 provides a 2-user achievable rate-region example for a particular $p_x$ that uses both Subsection 2.6.3’s method in Equations (2.259) - (2.260) and this subsection’s rate-sum method in Equation (2.277).
Figure 2.30: Achievable Rate Region for two-user multiple-access channel.

Figure 2.30's 2-user achievable region is the pentagon with the thick colored boundary lines and its interior through the convex-hull operation. This rate region’s computation requires $2^U - 1 = 3$ rate sums' computation to bound the region along with the two axes for $b_1 \geq 0$ and $b_2 \geq 0$, tracing a pentagon for any particular input distribution $p_x$. The approach of Section 2.6.3 is also fairly easy to compute for any given $p_x$. Points in the left rectangle are those for which user 1 is decoded first and then used to assist the decoding of user 2. The lower rectangle corresponds to decoding first user 2 and then user 1. The lower left corner corresponding to the two rectangles’ overlap admits either order of decoding. The upper right triangular “time-share” region more precisely corresponds to dimension-sharing (convex hull of union) the two sub-users’ codes and decoding orders that corresponds to sub-users that follow either A or B. Each particular $p_x$ again corresponds to a pentagon after the dimension-sharing, for which the 4 non-zero vertices may change. The convex hull of all such pentagons is $C(b)$.

For the MAC with $U$ users, the rate region is a polytope in the first orthant with

$$2^U - 1 + U$$

faces

$$\sum_{k=0}^{U} \binom{U}{k} \cdot k!$$

vertices.

(2.284)'s faces follow from $\sum_{k=1}^{U} \binom{U}{k} = 2^U - 1$ possible choices of $k$ users from $U$ to energize, plus the $U$ planes that bound the all-positive orthant. (2.285)'s vertices follow from $k!$ different orders among the $k$-user macro groups times $\binom{U}{k}$ possible choices of marco-group $k$-user sets from $U$ users. There are $U!$ orders that correspond to all $U$ users active (with nonzero energy).

Figure 2.31 depicts the achievable region for $U = 3$. There are $2^U - 1 = 7$ numbered (non-trivial) faces plus the $U = 3$ (un-numbered) for the positive octant. The vertices are the $U! = 3! = 6$ shown as rose-colored points for all $\binom{U}{k} = 1$ users active, $\binom{U}{2} \cdot 2! = 3 \cdot 2 = 6$ blue dots for any of the pairs of only 2 users energized, $3 = \binom{3}{1} \cdot 1!$ green dots for only one user active, and the single black dot for no users active. This sums$^{56}$ to 16, as per Equation (2.285).

---

$^{56}$But the general formula is not necessarily $2^{U+1}$, as evident already for $U = 2$ where there are 5 vertices ($= 1+2\cdot1!+1\cdot2!$)
2.7.2 The Gaussian multiple-access channel.

Figure 2.33’s Gaussian MAC rate region $A(b, p_x)$ simplifies through use of familiar signal-to-noise-ratio and MMSE measures as in Subsection 2.3.6, which this present subsection addresses. The number of user dimensions now generalizes from $L_x$ to $L_{x,u}$ for user $u$; thus each MAC user may have different dimensionality.
Each MAC user has its own channel matrix processing, $H_u$, before addition of other users and of a common Gaussian noise $n$ to construct $y$. When MAC inputs are Gaussian, best design achieves points on a rate-region boundary. Gaussian inputs' sufficiency and necessity to achieve boundary points also follow from the last-decoded MAC user: The joint distribution $p_{y/x}(x,v)$ simplifies to a Gaussian distribution with fixed sample values for all the other $x_i \neq u$, and is equivalent to a single-user AWGN $y = H_U \cdot x_U + n$, so then $p_{y/x} \rightarrow p_{n}(v - H_U \cdot x_U)$. By the single-user capacity theorem, then $b_U = I(x_U; y/x \mid x_U)$ is largest rate and the corresponding largest-rate-achieving distribution $p_{x_U}$ is Gaussian. In turn, because this last user $x_U$ has Gaussian distribution, the noise plus crosstalk seen by the 2nd to last decoded user, user $U - 1$, also has Gaussian distribution $p_{n + H_{U-1} \cdot x_{U-1}}(v - H_{U-1} \cdot x_{U-1})$. Continuing, user 1 and all lower-indexed users must also be Gaussian by induction.

2.7.2.1 The scalar Gaussian multiple-access channel.

The scalar Gaussian MAC is very special, but often encountered, case with $L_y = L_{x,u} = 1 \forall u = 1, ..., U$. With Gaussian MAC inputs, each and every rate sum corresponds to a macro-user. For the largest macro-user rate, the macro-user distribution must be vector Gaussian. Because this is true for each and every macro-user, each individual (sub-) user’s marginal distribution is then also Gaussian. The mutual information quantities of the scalar ($N = 1$) MAC capacity region rate-sum descriptions become log-one-plus-signal-to-noise-ratio quantities, where any user’s signal energy is the product of the input energy $\mathcal{E}_u$ and the channel gain $|H_{1u}|^2$, and the noise is the sum of the Gaussian noise and any uncanceled other users. Mathematically, for any $\{u\} \subseteq \{U\}$:

$$I(x_u; y/x_i \in U \setminus u) = \frac{1}{2} \log_2 \left( 1 + \frac{\mathcal{E}_u \cdot |H_{1u}|^2}{\sigma^2 + \sum_{i \in U \setminus u} \mathcal{E}_i \cdot |H_{iu}|^2} \right).$$

(2.286)

For a scalar AWGN MAC (so $L_y = 1$) example if $U = 2$ and the $2 \times 1$ channel is $H = [h_1 \ h_2]$, then

$$y = h_1 \cdot x_1 + h_2 \cdot x_2 + n.$$

(2.287)

Each rate sum is associated with an SNR:

$$\text{SNR}_1 = \frac{\mathcal{E}_1 \cdot |h_1|^2}{\sigma^2} \quad (2.288)$$

$$\text{SNR}_2 = \frac{\mathcal{E}_2 \cdot |h_2|^2}{\sigma^2} \quad (2.289)$$

$$\text{SNR} = \frac{\mathcal{E}_1 \cdot |h_1|^2 + \mathcal{E}_2 \cdot |h_2|^2}{\sigma^2} \quad (2.290)$$

288
and

\[ \bar{b}_1 \leq \frac{1}{2} \log_2 (1 + \text{SNR}_1) \]  
\[ \bar{b}_2 \leq \frac{1}{2} \log_2 (1 + \text{SNR}_2) \]  
\[ \bar{b}_1 + \bar{b}_2 \leq \frac{1}{2} \log_2 (1 + \text{SNR}) . \]

The SNR’s and thus rate sums for all possible Gaussian input distributions would all be less than those shown because trivially each SNR either stays the same or reduces for any Gaussian distributions other than those using the maximum energy. Thus, the 3 equations (2.291) - (2.293) also specify the capacity region, along with \( b_1 \geq 0 \) and \( b_2 \geq 0 \).

In general with \( U > 2 \) scalar Gaussian MACs sum users, and the pentagon generalizes to an \( U \)-dimensional region with \( 2^U + U - 1 \) sides in \( U \)-space (\( \mathcal{C}^U \) for complex channels). Each axis has a maximum single-user data-rate hyperplane orthogonal to that axis and forms a boundary for the \( U \)-dimensional capacity region at \( \bar{C}_u = .5 \cdot \log_2 (1 + \text{SNR}_u) \) where \( \text{SNR}_u = \mathcal{E}_u \cdot |H_u|^2 / \sigma^2 = \mathcal{E}_x \cdot g_u \). There are hyperplanes for all possible subsets \( u \), each as \( \sum_{u \in u} b_u \leq \bar{C}_u = 1/2 \cdot \log_2 (1 + \text{SNR}_u) \) where \( \text{SNR}_u = \sum_{u \in u} \mathcal{E}_u \cdot |H_u|^2 / \sigma^2 \).

**Energy-Sum MAC:** The MAC may have \( \sum_{u=1}^U \mathcal{E}_u = \sum_{u=1}^U \text{trace} \{ R_{xx}(u) \} \leq \mathcal{E}_x \) in special cases of allowed inputs. When this happens in the scalar MAC, clearly then

\[ b_{\text{max}} = \log_2 (1 + \mathcal{E}_x \cdot g_u) \text{ bits/complex-subsymbol} \]  

because \( \max_u g_u = g_U \) by definition. In this case, the maximum rate sum depends on order and in fact must have user \( U \) at the top of \( \pi_U \) with all energy allocated to this “primary” user.

**Example 2.7.1 (Scalar AWGN channel)** Figure 2.34 illustrates a simple 2-user AWGN channel. This channel’s sum-rate capacity follows easily by recognizing \( z \) is a single-user (macro-user) AWGN channel input. This single-user capacity is \( \mathcal{I}(x; y) = \mathcal{I}(z; y) = .5 \cdot \log_2 (1 + 10^4) = 6.64 \text{ bits/dimension} \). However, consideration of either user’s signal as Gaussian noise, which distorts the other users, leads to marginal capacities of

\[ \mathcal{I}(x_1; y) = \frac{1}{2} \log_2 \left( 1 + \frac{.36 \cdot 1}{.0001 + .64} \right) = .32 \text{ bits/dimension} \]  
\[ \mathcal{I}(x_2; y) = \frac{1}{2} \log_2 \left( 1 + \frac{.64 \cdot 1}{.0001 + .36} \right) = .74 \text{ bits/dimension} \]  
\[ \text{total} = 1.06 \text{ bits/dimension} \]
which is well below the maximum rate sum.

Decoding $x_1$ first (point A in Figure 2.35) leads to

$$ I(x_2; y/x_1) = \frac{1}{2} \log_2 (1 + \text{SNR}_2) = \frac{1}{2} \log_2 \left( 1 + \frac{.64 \cdot 1}{.0001} \right) = 6.32 \text{ bits/dimension}, \quad (2.298) $$

or decoding $x_2$ first (point B) leads to

$$ I(x_1; y/x_2) = \frac{1}{2} \log_2 (1 + \text{SNR}_1) = \frac{1}{2} \log_2 \left( 1 + \frac{.36 \cdot 1}{.0001} \right) = 5.90 \text{ bits/dimension} \quad (2.299) $$

In either case, the sum is the same as $0.32 + 6.32 = 0.74 + 5.90 = 6.64 \text{ bits/dimension}$. Figure 2.35 illustrates the capacity rate region and the two successive-decoding points, $A$ and $B$. Figure 2.36 illustrates the two successive-decoding receivers for this channel. Other points along the slanted line imply sub-users with the two receivers and two independent codes. In the Gaussian case, time-sharing of two different distributions always corresponds to a single vector Gaussian distribution for each user. For instance, a time-sharing of $3/4$ point B and $1/4$ point A yields point C with

$$ E_{x_2} = \frac{1}{4}(1) + \frac{3}{4}(1) = 1 \quad (2.300) $$
$$ E_{x_1} = \frac{1}{4}(1) + \frac{3}{4}(1) = 1 \quad (2.301) $$
$$ b_2 = \frac{1}{4}(5.90) + \frac{3}{4}(6.32) = 1.72 \quad (2.302) $$
$$ b_1 = \frac{1}{4}(0.74) + \frac{3}{4}(6.32) = 4.92 \quad (2.303) $$
$$ b_1 + b_2 = 6.64 \quad (2.304) $$

This sub-user multi-subsymbol view is no longer strictly a MAC because dimensional sharing within the subsymbol would require sub-user transmitter coordination at a signal level. From a codeword view, it is possible to assign the point A and point B solutions to their respective fractions of subsymbols in a way that does not require original-users’ transmit coordination. This sub-user multi-subsymbol view is no longer strictly a MAC because dimensional sharing within the subsymbol would require sub-user transmitter coordination at a signal level. From a codeword view, it is possible to assign the point A and point B solutions to their respective fractions of subsymbols in a way that does not require original-users’ transmit coordination. Thus, from a an infinite-length codeword view, any dimension-sharing design remains a MAC as long as the subsymbol dimensions themselves remain uncoordinated within their respective fractions of use. With dimension-sharing for point C, all previous-single-dimension subsymbols become 4-dimensional, 3 uses each for solution B and 1 use each for solution A.

![Figure 2.35: 2-user AWGN rate region.](image)
Figure 2.36: Example successive decoders.

Figure 2.37 continues the $C(b)$-construction if this MAC relaxes input energy constraints to an energy-sum MAC, so

$$\mathcal{E}_1 + \mathcal{E}_2 = \mathcal{E}_x,$$

which might characterize total energy for a system of users that have a common energy source (even if in different locations). In this case, clearly all energy on user 2 provides the largest data rate of $b_2 = b_{\text{max}} = 6.82$ bits/subsymbol. All energy on user 1 leads to $b_1 = 6.4 < b_{\text{max}}$ bits/subsymbol. The region is curved with slope -1 (corresponding to the line $b_1 + b_2 = b_{\text{max}}$) tangent at point A. The triangle’s line slope has greater magnitude and thus slopes downward faster. The curved region arises from different users’ energy assignments, each generating a pentagon, and then taking the union (convex hull) of all such pentagons, with the red and green dashed pentagons so illustrating in Figure 2.37.

Figure 2.37: For the energy-sum constraint.

**Scalar MAC primary and secondary user intuition:** Example 2.7.1 illustrates that one scalar MAC user is “primary” in that it’s energy-sum-MAC maximum data rate computes as if there is no crosstalk (for this example). The other user is secondary in that its signal shares the primary user’s common output dimension but experiences that primary user as crosstalk noise energy that reduces its reliably achievable data rate. The energy-sum MAC’s primary and secondary users, once identified,
retain their characterization even if there is an energy vector. Example 2.7.1’s sum-energy MAC situation becomes of interest later (in duality with BC’s) as $E^*1 \leq E_x$ (=2 in Example 2.7.1).

For the scalar MAC with $L_u = 1$ and $U > 2$ users, one user (largest channel gain $g_u$ after ordering) is primary and the others are secondary. When two users have the same gain, they are fully symmetric with respect to any-other/all users and thus are a single macro user. When these two users both have the largest channel gain, the macro user is primary. By scalar-MAC convention in this text’s single total energy constraint, the scalar-MAC users order with again $g_u = \frac{|b_u|^2}{\sigma_u^2}$ such that $g_u \geq g_{u-1} \geq ... \geq g_1$.

When there is a different and separate non-zero energy constraint for each user, then the order generalizes to $E_u \cdot g_u \geq E_{u-1} \cdot g_{u-1} \geq ... \geq E_1 \cdot g_1$. The tangent line/plane at the maximum sum rate generalizes to

$$1^*b = b_{max}.$$ (2.306)

### 2.7.2.2 The Vector Gaussian MAC

With the vector MAC’s larger output dimensionality $L_y > 1$, this subsection shows that is possible for more than 1 MAC user to be primary and thus associate simultaneously with the maximum rate-sum when the MAC has only a energy-sum constraint. Such primary users are useful later in BC multi-user MAC duals. These primary users have $E_u > 0$ and thus $b_u > 0$ when $1^*b = b_{max}$, with formal definition:

**Definition 2.7.1 [Primary (Sub-) User for an Energy-Sum MAC]** An energy-sum Gaussian MAC primary user in primary-user set $u^o \subseteq U$ transmits non-zero energy when the energy-sum MAC achieves maximum rate sum, $b_{max}$, for any energy-vector constraint $E^*1 \leq E_x$. Users that must transmit zero-energy when the energy-sum MAC achieves this maximum rate sum are secondary users and belong to the set $u^s = U \setminus u^o$. Further, $U^o \triangleq \{u^o\}$ and $U^s \triangleq \{u^s\}$. When $U^o < U$, the energy-sum MAC is degraded.

When 2 or more users have the same channel gain $g_u = g_i$, these become one macro user for the purposes of primary/secondary classification, and the total number of users reduces by for each user beyond the first $g_u$.

Clearly from Definition 2.7.1, $U^o + U^s = U$. When $L_y = 1$ (the scalar Gaussian MAC), there is only one primary user. When $L_y > 1$, then there can more than 1 primary user. The number of primary (sub-) users depends on the channel rank through the following theorem:

**Theorem 2.7.3 [Primary User Determination]** The number of primary (sub-) users $U^o$ is $\prod_{u=1}^{L_y-1/2} \lambda_u$ and they correspond to the largest sub-user channel gains, $g_u \geq g_{u-1} \geq ... \geq g_{U^o+1}$.

**Proof:** Each user’s noise-whitened matrix channel is $\tilde{H}_u = R_{nn}^{-1/2} \cdot H_u$, with singular value decomposition

$$\tilde{H}_u = F_u \cdot \Lambda_u \cdot M_u^*,$$ (2.307)

which has non-zero singular values $\lambda_{u,1}, ..., \lambda_{u,L_u}$, $H_u$ of $H_u$. Simple receiver multiplication for user $u$’s detection by $F_u^*$ and pre-multiplication in transmitter $u$ by $M_u$ diagonalizes user $u$’s noise-whitened component channel. The corresponding channel gains (with all other users given) are $g_{u,\ell} = \lambda_{u,\ell} \cdot \lambda_u$, $\ell = 1, ..., L_u$. The product of the squared singular values for user $u$ also corresponding to $g_u = |\tilde{H}_u^* \tilde{H}_u| = \prod_{\ell=1}^{L_u} \lambda_{u,\ell}$ (with any zero singular values removed from the product). With an overall energy-sum constraint only, the maximum rate sum would become a water-filling problem for the $\sum_{u=1}^{U} \theta_{H_u}$ dimensions, but only
a maximum of \( q_H \leq \sum_{u=1}^{U} q_{H_u} \) can be used without crosstalk among them (which could only reduce rate sum). At some energy \( E_x \) sufficiently large, water-filling will use the largest \( q_H \) dimensions, each carrying one user (really sub-user)’s energy and achieving maximum rate sum. The remaining dimensions are all zero and any energization thereof would only reduce rate sum. QED.

In effect, Theorem 2.7.3 shows that the secondary users are weaker and absorb energy that has better use (in a rate-sum sense) on the primary users’ dimensions. With only a sum-energy constraint on users, the primary users have constant rate sum that is order independent, and is the maximum rate sum over all users. This energy-value ensemble solves the water-filling over the primary (sub-) users’ corresponding ensemble, see Subsection 2.7.4. Unlike the scalar case, maximum rate sum need not occur with all users \( i \in U \setminus u \) zeroed. Instead, all primary users share the MAC’s nonsingular (pass-space) dimensions to achieve the maximum rate sum. As secondary users use non-zero energy, their data rates increase but the sum data rate decreases from the all-primary-user maximum, regardless of a fixed sum-energy’s distribution to users. The best rate sum when energized secondary users are present depends upon the secondary users’ order. The primary users’ (maximum) rate sum is order-independent for any fixed \( E \) or \( E^*1 = E_x \), with no secondary users energized. Example 2.7.1 trivially exhibits this in that the maximum rate sum is the primary user’s (user 2)’s rate; however this fixed order-independent maximum occurs more generally for the primary user group \( u^o \). While defined in terms of an energy-sum MAC, these primary users remain identified for any MAC once found and may have additional interpretational uses.

Also when \( L_y > 1 \), the successive decoding approach consistently follows the mutual-information chain rule, but successive decoding does not simply correspond to subtraction of \( H_u \cdot \hat{x}_u \) from \( y \) to detect users \( i > u \). Subsection 2.3.6 shows that mutual information requires a MMSE estimate, which only simplifies to \( h_x \cdot \hat{x}_x \) in the scalar case with \( L_y = 1 \). Subsection 2.3.6 shows the MMSE estimate to be the ML decoder decision, which only equals the ZF solution when there is only 1 primary user. There are some later examples that help illustrate these MAC variations and primary/secondary users.

Figure 2.33’s Gaussian MAC has matrix\(^{37} H \) that is \( L_y \times L_x \), where

\[
L_x \Delta = \sum_{y=1}^{U} L_{x,u} . \tag{2.308}
\]

The MAC situations where \( L_y > 1 \) and/or \( L_{x,u} > 1 \) need further examination for primary and secondary users.

---

\(^{37} n = 1, ..., N \) is a frequency/time index that is not included here, but Chapters 4 and 5 show that the Gaussian MAC decompositions into \( N \) separate MACs, indexed by \( n \). Thus the methods of this chapter apply to each tone/dimension of such a system.
Vector Gaussian MAC Normalization with known block diagonal $R_{xx}$: The following prepares the vector MAC for common analysis with a fixed given set of user autocorrelation matrices $\{R_{xx}(u)\}_{u=1,\ldots,U}$.

1. Step 1: Vector MAC transformation

   The MAC-user sub-matrices $H_u$ normalize through definition of the $L_y \times L_{x,u}$ matrices (for $u = 1, \ldots, U$)

   $$
   \bar{H}_u \triangleq R_{nn}^{-1/2} \cdot H_u \cdot R_{xx}^{1/2}(u) .
   $$

   User $u$ partitions into $q_{x,u} \triangleq q_{R_{xx}(u)}$ sub-users, one for each non-zero probability component of the random Hilbert vector process $x_u$. Figure 2.38 illustrates this normalization that creates a new “white input” with

   $$
   R_{\nu \nu} = \begin{bmatrix}
   R_{\nu \nu}(U) & \cdots & 0 \\
   \vdots & \ddots & \vdots \\
   0 & \cdots & R_{\nu \nu}(1)
   \end{bmatrix} = I ,
   $$

   (2.310)

   The individual unit-energy sub-user scalar (possibly complex) components are $\nu_{u,\ell}, \ell = 1, \ldots, q_{x,u}$ and

   $\nu_u = \begin{bmatrix}
   \nu_{u,q_{x,u}} \\
   \vdots \\
   \nu_{u,1}
   \end{bmatrix} .
   $ (2.311)

   Transmitter $u$ retains the correct given $R_{xx}(u)$ on the actual channel input $x_u$ through the matrix filter $R_{xx}^{1/2}(u)$. $\bar{H}_u$’s transformation in (2.309) also whitens the noise. The noise whitening matrix always exists with non-singular noise (always in practice). The matrix $R_{xx}^{1/2}(u)$ in (2.309) is any square root for which it’s pseudoinverse will produce $q_{x,u}$ independent unit-energy inputs $\nu_u$, $u = 1, \ldots, q_{x,u}$. A particular choice follows from $R_{xx}(u)$’s eigen-decomposition

   $$
   R_{xx}(u) = \sum_{\ell=1}^{L_{x,u}} \xi_{u,\ell} \cdot q_{u,\ell} \cdot q_{u,\ell}^* = Q \cdot \begin{bmatrix}
   \Xi_u & 0 \\
   0 & 0
   \end{bmatrix} \cdot Q^* ,
   $$

   (2.312)

   where the diagonal $\Xi_u$ has $q_{x,u}$ positive-real values. Then with (pseudo) square-root choice

   $$
   R_{xx}^{1/2}(u) = Q \cdot \begin{bmatrix}
   \Xi_u^{1/2} & 0 \\
   0 & 0
   \end{bmatrix} \cdot Q^* ,
   $$

   (2.313)

   $R_{xx}^{1/2}(u) \cdot R_{xx}^{1/2}(u) = R_{xx}(u)$. The desired square-root pseudoinverse is then

   $$
   R_{xx}^{-1/2}(u) = Q \cdot \begin{bmatrix}
   \Xi_u^{-1/2} & 0 \\
   0 & 0
   \end{bmatrix} \cdot Q^* .
   $$

   (2.314)

   So then the set of distinct codewords subsymbols formed from $q_{x,u}$ values of average unit-energy $\nu_{u,\ell}$ per subsymbol map uniquely (and invertible with probability 1) to codewords $x_u$ that have autocorrelation $R_{xx}(u)$. The mapping is unique and invertible because any distinct vector $x_u$ that exists (with nonzero probability) on the eigenvectors $q_{u,\ell} \forall \ell = 1, \ldots, q_{R_{xx}(u)}$, so that $x_u = \sum_{\ell=1}^{q_{x,u}} \chi_{u,\ell} \cdot q_{u,\ell}$ has a linear invertible mapping between $x_u$ and a corresponding $\nu_u$. Any inversion ambiguity disappears by choosing $\nu_u$ to have the pseudoinverse’s unique $\|\nu_u\|^2 = \|x_u\|^2$ choice. By Lemma 2.3.5, there is no information loss. Clearly $q_{x,u} \leq L_{x,u}$, or effectively the input $\nu_u$ has autocorrelation $\begin{bmatrix}
   I & 0 \\
   0 & 0
   \end{bmatrix}$ if $x_u$ has $L_{x,u} > q_{x,u}$ dimensions. Singular inputs can be useful generally in that they avoid energy transmission on poor channel modes (e.g., water-filling).
2. Step 2: Vector MAC reduction

This step removes (sub-) users that produce zero output energy and $U'$ correspondingly reduces. This step differs from an earlier step in the proof and enumeration of primary users in that the matrix filter $R^{1/2}_{yy}(u)$ now is part of $\tilde{H}_u$. This absorption of $R^{1/2}_{yy}(u)$ facilitates the upcoming designs because it leaves a unit-energy input per (used) dimension in the upcoming structures. Thus, it may find different execution as $C(b)$ construction considers different input autocorrelations that satisfy the applicable energy constraint(s). To determine order for primary-user identification, $\tilde{H}_u$’s singular value decomposition yields

$$\tilde{H}_u = [F_u \quad f_u] \begin{bmatrix} \Lambda_u & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} M_u^* \\ \tilde{m}_u^* \end{bmatrix} \quad (2.315)$$

where $\Lambda_u$ is a $\varrho_{\tilde{H}_u} \times \varrho_{\tilde{H}_u}$ matrix; and $F_u$ is $L_y \times \varrho_{\tilde{H}_u}$ while $f_u$ is $L_y \times (L_y - \varrho_{\tilde{H}_u})$ and similarly $M_u^*$ is $\varrho_{\tilde{H}_u} \times L_{x,u}$ while $\tilde{m}_u^*$ is $(L_{x,u} - \varrho_{\tilde{H}_u}) \times L_{x,u}$. The user channel of interest in terms of information and energy then becomes

$$\tilde{H}_u = F_u \cdot \Lambda_u \cdot M_u^* , \quad (2.316)$$

which has non-singular $\Lambda_u$.

The order ranking over all (sub-) users is

$$\lambda_{U'} \geq \lambda_{U-1} \geq \ldots \geq \lambda_1 > 0 \quad . \quad (2.317)$$

Subsequent Vector MAC analysis simplifies by relabeling (or setting equal) user $u$’s channel by a new $L_y \times \varrho_{x,u}$ white-noise equivalent that presumes this Step 1’s and 2’s execution.

It will be convenient to read the following passage as if $L_{x,u} = 1$ and $\varrho_{\tilde{H}_u} = L_y = U = U'$, then comments at the end suggest rereading to envision dimensional adjustment, expanding to user-dependent indexed dimensionality. Thus related parenthetical comments may best await second reading. Further, the sequel of this subsection then treats the primary users’ order as $u = 1, \ldots, U^0$ to simplify notation from $u = U^0 + 1, \ldots, U'$, essentially tacitly subtracting $U^0$ from all user indices.

**Canonical Forward/Backward MAC:**

The general Gaussian MAC, as modified in Step 1 begins with a receiver matched-matrix (filter) that (as always) retains all information:

$$y' = \tilde{H}^* \cdot \tilde{H} \cdot y + \tilde{H}^* \cdot n' \quad , \quad (2.318)$$

which however produces noise $n'$ with autocorrelation $R_{y' n'} = R_f$. The matrix $R_f$ is the MAC’s **canonical forward-channel matrix**. The diagonal (block) components of $R_f$ are $R_f(u)$ and their singular values are the same as those in (2.317). It may be tempting to simply Cholesky factor $R_f$ find its (monic) Cholesky factor’s inverse and then (after scaling each user) do successive decoding, which is a **zero-forcing** solution. While zero-forcing is feasible, the ML detection of each user is not optimum successive decoding (even if the previous users’ decisions are absolutely correct, as per Subsection 2.3.6).

The best successive decoding instead uses the MAC’s **backward canonical matrix** $R_b$ (actually its inverse)

$$R_b^{-1} \equiv R_f + I = G^* \cdot S_0 \cdot G \quad , \quad (2.319)$$

where $^58$ $G$ is a $\mathcal{L}_y \times \mathcal{L}_x$ monic$^59$ upper-triangular matrix found from Cholesky factorization of $R_b^{-1}$, and $S_0$ is a matrix of positive real Cholesky factors. (Appendix D describes Cholesky Factorization.) The relation

$$2^{2T(x; y)} = |R_b^{-1}| = |S_0| \quad , \quad (2.320)$$

follows immediately.

---

$^58$ This $G$ is not the linear finite-field generator, although the two have analogous functions and whence the common notation.

$^59$ Monic means 1’s on the diagonal.
**MMSE MAC:** Appendix D’s MMSE estimator of $\mathbf{x}$ (where the relabeled $\mathbf{x}$ arose from Step 1’s absorption of $R_{xx}^{-1/2}(u)$ into $\tilde{H}_u$ that left $R_{xx} = I$), given $\mathbf{y}'$ is

$$R_{xy'} \cdot R_{y'y}^{-1} = R_f [R_f \cdot R_f + R_f]^{-1} = [R_f + I]^{-1} = R_b ,$$

(2.321)

so $R_b$ is this MMSE estimator for $\nu$. Also, $R_b$ is always non-singular with nonsingular $R_{nn}$ and has Cholesky factorization:

$$R_b = G^{-1} \cdot S_0^{-1} \cdot G^{-*} .$$

(2.322)

$R_b$ is thus well conditioned and has inverse and Cholesky factorization, and thus $G$ and $S_0$ are also invertible. Then, the receiver forms (recalling that the MMSE estimate of a linear combination is the linear combination of the estimates)

$$\mathbf{y}'' = S_0^{-1} \cdot G^{-*} \mathbf{y}'$$

(2.323)

$$= S_0^{-1} \cdot G^{-*} \cdot [R_f \cdot \nu + \mathbf{n'}]$$

(2.324)

$$= S_0^{-1} \cdot G^{-*} \cdot [R_b^{-1} - I] \cdot \nu + S_0^{-1} \cdot G^{-*} \cdot \mathbf{n'} ,$$

(2.325)

Further

$$\nu = R_b \cdot \mathbf{y}' + \mathbf{e} ,$$

(2.326)

so

$$\mathbf{y}' = R_b^{-1} \cdot \nu - R_b^{-1} \cdot \mathbf{e}$$

(2.327)

$$S_0^{-1} \cdot G^{-*} \mathbf{y}' = G \cdot \nu - G \cdot \mathbf{e}$$

(2.328)

$$\mathbf{y}'' = G \cdot \nu - \mathbf{e}' ,$$

(2.329)

where the MMSE error vector $\mathbf{e}' \overset{\Delta}{=} G \cdot \mathbf{e}$ has diagonal autocorrelation

$$Re'e' = S_0^{-1} .$$

(2.330)

The cascade of receiver matrix multiplies is (See Figure 2.39.)

$$\mathbf{y}'' = \underbrace{S_0^{-1} \cdot G^{-*} \cdot \tilde{H}^*} \cdot \mathbf{y} .$$

(2.331)
Correct Successive Decoding: The monic upper triangular matrix $G$ has structure

$$
G = \begin{bmatrix}
1 & g_{U',U'-1} & \cdots & g_{U',2} & g_{U',1} \\
0 & 1 & \cdots & g_{U'-1,2} & g_{U'-1,1} \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & 1 & g_{2.1} \\
0 & 0 & \cdots & 0 & 1
\end{bmatrix}
$$

(2.332)

The receiver processes a subsymbol $y''$ by its non-degraded (primary) user components successively from user 1 up through user $U$:

$$
\hat{\nu}_1 = \text{decision} (y''_1) \\
\hat{\nu}_2 = \text{decision} (y''_2 - g_{2.1} \cdot \hat{\nu}_1) \\
\vdots \quad \vdots \quad \vdots
$$

The operation of $\hat{\nu}_u$ corresponds to a decision that finds the subsymbol value for user $u$ from $y_u$ directly - in actual implementation, receivers might approximate this with a decision at the subsymbol time or might delay to process several subsymbol periods for the same user, depending on code specifics - the ideal ML good-code decoder would need to wait an infinite delay and then the “hat” notation corresponds to the correct subsymbol in user $u$’s codeword; then user 2’s values could be found for each subsymbol $\nu_1$ already known from the infinite delay.
\[ \hat{\nu}_U' = \text{decision} \left( y''_U - \sum_{i=1}^{U-1} g_{U,i} \cdot \hat{\nu}_i \right). \] (2.333)

This successive decoding, or back-substitution, process implements \( G^{-1} \) so that user 1 decodes perfectly with Gaussian code so \( P_{e,1} \to 0 \), and then the resultant \( \hat{\nu}_1 = \nu_1 \). This “decision” operation may have decoder delay to process a long symbol before the subsymbols become available for use. The first primary user channel experiences no crosstalk from any user because the receiver (in Step 1) already subtracts secondary users. This \( \nu_1 \) now multiplies the red \( g_{2,1} \) component in Equation (2.332)’s second row up from bottom and disappears through the operation \( y''_2 - g_{2,1} \cdot \nu_1 = \nu_2 + e' \), another simple single-user AWGN. The back-substitution process continues inductively. Successive decoding using the matrix \( G \) leads to independent user decisions, each with a \((L_y)\) signal-to-noise ratio(s) from (2.329) that is (are diagonal elements of) \( S_0(u) \) (recalling \( R_{\nu\nu} \) has been normalized to the identity \( R_{\nu\nu} = I \) in the original channel definition’s Step 1 \( \tilde{H} \) formation).

**SNR’s and User bits/subsymbol:** Each \( S_0(u) \) is a diagonal \( \mathcal{L}_x \times \mathcal{L}_z \) matrix with diagonal SNR elements

\[ SNR_{\text{mmse},u,\ell} = S_0(u, \ell). \] (2.334)

Step 1’s processing preserves the mutual information, which is then

\[ I(x; y) = \log_2 \left( \frac{\vert \tilde{H}^* \tilde{H} + I \vert}{R_{e^{-1}}} \right) \] (2.335)

\[ = \log_2 \left\{ \prod_{u=1}^{U} \prod_{\ell=1}^{L_y} \text{SNR}_{\text{mmse},u,\ell} \right\} \text{bits / complex symbol}. \] (2.336)

Thus, this processing creates a set of individual decoders whose \( SNR_{\text{mmse},i,\ell} \)’s and corresponding individual subsymbol mutual information quantities \( I_{u,\ell} \triangleq \log_2(SNR_{\text{mmse},u,\ell}) \) sum to the mutual information. As per Section 2.3.6 and Appendix D, the individual residual outputs at each decoding stage of \( y'''_u = y''_u - \sum_{i=1}^{u-1} g_{u,i} \cdot \hat{\nu}_i \) have residual bias and so

\[ E[y'''_u/\nu_u] = \{ I_{L_y} - S_0^{-1}(u) \} \cdot \nu_u. \] (2.338)

The reduction in energy accrues to the MMSE-receiver-processing signal reduction, along with the Gaussian noise \( n \), to create a smaller error \( e \). Thus \( SNR_{\text{mmse},i} = \vert S_0(i) \vert \) is slightly higher than the unbiased (scaled) channel SNR and indeed each dimension actually has \( SNR_{u,\ell} = SNR_{\text{bias},u,\ell} - 1 \). This bias removes according to (See Figure 2.39)

\[ y''' = (S_0 - I)^{-1} \cdot G^{-*} \cdot H^* \cdot R_{nn}^{-1} y, \] (2.339)

or equivalently

\[ W_{\text{unb}} = (S_0 - I)^{-1} \cdot S_0 \cdot W, \] (2.340)

can be of consequence with codes that have \( \Gamma > 0 \) dB. Nonetheless, the overall mutual information remains the same as

\[ I(x; y) = \log_2 \left( \frac{\vert \tilde{H}^* \tilde{H} + I \vert}{R_{e^{-1}}} \right) = \log_2 \left\{ \prod_{i=1}^{L_y} \prod_{\ell=1}^{L_z} (1 + SNR_{i,\ell}) \right\} \text{bits / complex symbol}. \] (2.341)
The MMSE/mutual-information-chain-rule solution has residual components of all other primary users when $L_y > 1$, but this is good because the MMSE criterion trades some crosstalk for noise amplitude reduction in a way that improves SNR. The lower-index crosstalk components remove in the successive-decoding/back-substitution process. The higher-index components are part of the error $e$. The receiver processing produces a signal component (ignoring noise $n$ components)

\[
\text{signal part of } y'' = S_0 \cdot [S_0^{-1} \cdot G^{-*} \cdot R_f \cdot \nu] \\
= S_0^{-1} \cdot G^{-*} \cdot [R_0^{-1} - \bar{I}] \nu \\
= G \cdot \nu - S_0^{-1} \cdot G^{-*} \cdot \nu .
\]

(2.342) (2.343) (2.344)

Bias removal in the back-substitution process based on $\nu$ produces an unbiased successive decoder that uses

\[
G_{\text{unbiased}} = I + [S_0 - I]^{-1} \cdot S_0 \cdot [G - I] .
\]

(2.345)

The processing also produces a lower-triangular user-error component that is

\[
-S_0^{-1} \cdot G^{-*} \cdot \nu .
\]

(2.346)

Thus, unlike a zero-forcing solution, the MMSE will have some crosstalk from all users in every error, with those of lower index removed exactly but those of higher index being residual error. When $S_0^{-1}(u)$ is small (large SNR), this residual error is small and the forward processing from channel input to output (prior to back substitution) will increasingly appear upper triangular. However, if $S_0^{-1}(u)$ is large (small SNR), the deviation from triangular form can be large, as in Example 2.7.2.

**Gaussian’s Optimality:** Further, the last decoded primary user has all other users given, and so has Gaussian conditional distribution that is the Gaussian noise distribution. Thus this last-decoded user sees a single-user channel. This last-decoded user’s best input distribution is therefore Gaussian. The second-last decoded primary user subsequently has all other users except the last given. Since the last is already Gaussian, then this second-last decoding is also a single-user channel with Gaussian noise, and thus the best distribution for the second-last user is Gaussian. Inductively, even with bias, all users remain Gaussian, consistent with earlier findings. The (pre-Step 1) Gaussian MAC input autocorrelation matrix $R_{xx}$ is always block diagonal. Step 1 forces this to $I$, but energy constraints apply as originally given on $\text{trace}\{R_{xx}(u)\}$ and correspond to $\varrho_{x,u}$ on the normalized inputs $\nu_u$, $\text{trace}\{R \nu \nu(u)\} = \varrho_{x,u}$.

There are two consequent maximum rate-sum situations of interest:

**Energy Vector Constraint** The individual energy-constraints $\text{trace}\{R_{xx}(u)\} \leq \mathcal{E}_u$ will have the same rate sum for any particular $R_{xx}$ and order $\pi$, and this rate sum is $I(x; y) = \log_2 |\bar{H} \cdot \bar{H}^* + I|$ bits/subsymbol.

**Energy-Sum MAC** The total energy-constraint of $\text{trace}\{R_{xx}\} \leq \mathcal{E}_x$ will have the same maximum rate sum for all $U$ primary orders, as long as the secondary users carry no energy. The rate sum is not maximum if any secondary user carries energy, and further the rate sum then depends on the order within the secondary users (but not within the primary users).

With only primary users, the sum rate is largest for all $R_{xx}$ such that $\text{trace}\{R_{xx}\} \leq \mathcal{E}_x$ with zero-energy secondary components, and this maximum rate sum is independent of order (another argument for all user inputs being Gaussian in best case). When the energy-sum MAC has no energized secondary users, it is non-degraded. Degraded energy-sum MACs have non-zero secondary user energy, and this energy reduces the sum rate of the primary users with respect to this maximum. The primary-user rate sum $b_{\text{max}} = \sum_{u=U+1}^U b_u$ will remain the same for the sum-energy MAC and given $R_{xx}^*$ for any $R_{xx}$ on secondary users since the receiver first removes them. However, for a fixed energy-sum MAC with $\mathcal{E}_x$ over all users, primary and secondary $U = u^* \cup u^*$, any energy on the secondary users so that effectively $u^* \not\subset \emptyset$ reduces the primary users’ maximum rate sum, which implies also then reduction of all users’ maximum rate sum.
EXAMPLE 2.7.2 (Two primary users) A $2 \times 2$ MAC with $U = U^o = 2$ has channel matrix

$$\tilde{H} = \begin{bmatrix} 5 & 2 \\ 3 & 1 \end{bmatrix}. \quad (2.347)$$

The following sequence of matlab commands investigates these two primary users’ processing by determining $G$ and $S_0$:

```matlab
>> Rf=H'*H =
   34   13
   13   5
>> Rbinv=Rf+eye(2) =
   35   13
   13   6
>> Gbar=chol(Rbinv) =
   5.9161  2.1974
    0    1.0823
>> S0=diag(diag(Gbar))*diag(diag(Gbar)) =
   35.0000   0
    0  1.1714
>> G=inv(diag(diag(Gbar)))*Gbar =
   1.0000  0.3714
    0  1.0000
>> b=0.5*log2(diag(S0)) =
   2.5646
    0.1141
>> sum(b) = 2.6788
```

The bit vector shows larger user 2 and smaller user 1. The matlab results below illustrate several of Figure 2.39’s receiver-filtering Gaussian-MMSE MAC concepts. The post-matched filter processing, MMSE and with biased removed (shown as $W$ and $W_{unb}$ respectively) are triangular as expected. The unbiased filter/matched-filter combination is MSWMFu and $2 \times 2$ (non triangular). Application of this to the channel ($MSWMFu*H$) should match the upper triangular unbiased feedback ($Gunb$), and it does. However, it is not triangular - MMSE solutions will not be exactly triangular because there is a residual error traded for noise reduction. The lower left component 2.1667 is this match, and appears large. However, this example intentionally has low SNR to emphasize MMSE effects. The bias removal of $1.17/(1.17 -1)$ is large when the SNR is small.

```matlab
>> W=inv(S0)*inv(G') =
    0.0286   0
   -0.3171   0.8537
>> Wunb=S0*inv(S0-eye(2))*W =
    0.0294   0
  -2.1667   5.8333
>> MSWMFu=Wunb*H' =
   0.1471   0.0882
   0.8333  -0.6667
>> Gunb=eye(2)+S0*inv(S0-eye(2))*(G-eye(2)) =
   1.0000   0.3824
    0  1.0000
>> MSWMFu*H =
   1.0000   0.3824
  2.1667  1.0000
```

300
The zero-forcing solution is also of interest to contrast processing. It can be implemented by Cholesky factorization of $R_f$ or also directly QR factorization of $H$, as per Figure 2.40. Figure 2.40 shows the 1-to-1 transformations (which do not change mutual information) that result in a channel that has diagonal noise and triangular channel $G_{ZF}$, which can be inverted by a back-substitution process that is identical to the $G^{-1}$ process in (2.333). This is the straightforward implementation of successive decoding that many confuse many because it is not ML in 2 dimensions. Indeed the 2nd dimension has a lower SNR and bit rate than the correct MMSE implementation displayed above. While the 1-to-1 processing produces a set of parallel independent channels, this set of channels have lower SNR than is possible and indeed do not minimize the ML sum in (2.143). Figure 2.40’s receiver processing $S_{ZF}^{-1} \cdot Q^*$ does exactly produce $G_{ZF}$, and is thus upper triangular also, but the performance is indeed worse. Simple successive decoding is not optimum ML in the multi-dimensional zero-forcing case.

Figure 2.40: Zero-Forcing MAC Receiver.

\[
\tilde{H}_{ZF} = Q \cdot S_{ZF} \cdot G_{ZF}
\]

\[
\begin{align*}
&\begin{bmatrix}
Q & R
\end{bmatrix} = \text{qr}(H) \\
&Q = \\
&\begin{bmatrix}
-0.8575 & -0.5145 \\
-0.5145 & 0.8575
\end{bmatrix} \\
&R = \\
&\begin{bmatrix}
-5.8310 & -2.2295 \\
0 & -0.1715
\end{bmatrix} \\
&Q \cdot R = \\
&\begin{bmatrix}
5.0000 & 2.0000 \\
3.0000 & 1.0000
\end{bmatrix} \text{ (checks)}
\end{align*}
\]

\[
\begin{align*}
&\begin{bmatrix}
S_{ZF} = \text{diag}(\text{diag}(R)) \\
&\begin{bmatrix}
-5.8310 & 0 \\
0 & -0.1715
\end{bmatrix}
\end{bmatrix} \\
&G_{ZF} = (S_{ZF})^{-1} \cdot R = \\
&\begin{bmatrix}
1.0000 & 0.3824 \\
0 & 1.0000
\end{bmatrix} \\
&\text{det}(S_{ZF} \cdot S_{ZF}) = 1.0000 \\
&W = (S_{ZF})^{-1} \cdot Q^* = \\
&\begin{bmatrix}
0.1471 & 0.0882 \\
3.0000 & -5.0000
\end{bmatrix} \\
&W \cdot H = \\
&\begin{bmatrix}
1.0000 & 0.3824 \\
0 & 1.0000
\end{bmatrix} \text{ (checks = Gzf)}
\end{align*}
\]

\[
0.5 \cdot \log_2(\text{eye}(2) + S_{ZF} \cdot S_{ZF}) = \\
2.5646 \\
0.020
\]

\[
\begin{align*}
&b = \\
&\begin{bmatrix}
2.5646 \\
0.1141
\end{bmatrix}
\end{align*}
\]
$$\text{sum}(b) = 2.6787$$

$$b - 0.5\times\log(\text{diag}(\text{eye}(2)+Szf*Szf)) =$$

$$0.0932$$

(MMSE and ZF same in user 2, but not user 1.)

$$\text{Wzf} = \text{inv}(Szf)\times Q' =$$

$$\begin{bmatrix}
0.1471 & 0.0882 \\
3.0000 & -5.0000
\end{bmatrix}$$

$$\text{Wzf}\times H =$$

$$\begin{bmatrix}
1.0000 & 0.3824 \\
0 & 1.0000
\end{bmatrix}$$

In this zero-forcing example, the highest priority primary user has the same (optimum) data rate, but the second primary user has reduced data rate and SNR with respect to MMSE. The use of a single primary user to illustrate the Gaussian MAC thus misses some important issues because the ZF and MMSE are the same only in this special case with $L_y = L_{x,u} = 1$. This MAC's expansion to a $2 \times 3$ expansion where user 3 may have also (now extra) energy $E_3 = 1$ will have the same data rates on users 1 and 2, as per the example below. However if total energy is constrained so that all users have only $E_\mathcal{P} = 2$, then the data rates of the first two users decline as does the overall rate sum with respect to just 2 users.

$$\begin{bmatrix}
5 & 2 & 1 \\
3 & 1 & 1
\end{bmatrix};$$

$$\text{Rf} = H'\times H =$$

$$\begin{bmatrix}
34 & 13 & 8 \\
13 & 5 & 3 \\
8 & 3 & 2
\end{bmatrix}$$

$$\text{Rbinv} = \text{Rf} + \text{eye}(3) =$$

$$\begin{bmatrix}
35 & 13 & 8 \\
13 & 6 & 3 \\
8 & 3 & 3
\end{bmatrix}$$

$$\text{Gbar} = \text{chol}(\text{Rbinv}) =$$

$$\begin{bmatrix}
5.9161 & 2.1974 & 1.3522 \\
0 & 1.0823 & 0.0264 \\
0 & 0 & 1.0820
\end{bmatrix}$$

$$\text{G} = \text{inv}(\text{diag}(\text{diag}(\text{Gbar})))\times\text{Gbar} =$$

$$\begin{bmatrix}
1.0000 & 0.3714 & 0.2286 \\
0 & 1.0000 & 0.0244 \\
0 & 0 & 1.0000
\end{bmatrix}$$

$$\text{S0} = \text{diag}(\text{diag}(\text{Gbar}))\times\text{diag}(\text{diag}(\text{Gbar})) =$$

$$\begin{bmatrix}
35.0000 & 0 & 0 \\
0 & 1.1714 & 0 \\
0 & 0 & 1.1707
\end{bmatrix}$$

$$b = 0.5\times\log(\text{diag}(\text{S0})) =$$

$$\begin{bmatrix}
2.5646 \\
0.1141 \\
0.1137
\end{bmatrix}$$

More data rate because extra energy unit (total of 3)

$$\text{H} = \sqrt{2/3}\times H =$$

$$\begin{bmatrix}
4.0825 & 1.6330 & 0.8165 \\
2.4495 & 0.8165 & 0.8165
\end{bmatrix}$$

$$\text{Rf} = H'\times H =$$

$$\begin{bmatrix}
22.6667 & 8.6667 & 5.3333
\end{bmatrix}$$
8.6667  3.3333  2.0000  
5.3333  2.0000  1.3333  

>> Rinv=Rf+eye(3) =
23.6667  8.6667  5.3333  
8.6667  4.3333  2.0000  
5.3333  2.0000  2.3333  

>> Gbar=chol(Rinv) =
4.8648  1.7815  1.0963  
0  1.0769  0.0436  
0  0  1.0628  

>> G=inv(diag(diag(Gbar)))*Gbar =
1.0000  0.3662  0.2254  
0  1.0000  0.0405  
0  0  1.0000  

>> S0=diag(diag(Gbar))*diag(diag(Gbar)) =
23.6667  0  0  
0  1.1596  0  
0  0  1.1296  

>> b=0.5*log2(diag(S0)) =
2.2824  0.1068  0.0879  

>> sum(b) = 2.4771  
>> b(1)+b(2) = 2.3892 (< 2.6788)  

Further, the maximum (MMSE-based) sum rate of 2.6788 occurs only if $E_3 = 0$, and $E_1+E_2 = 2$.  

2.7.2.3 Non-Zero Gaps  
The rate regions so far presume $\Gamma = 0$ dB. Strictly speaking, the chain rule does not exactly hold when the gap is nonzero. For instance, if Example 2.7.1 instead has 3 dB gap, $I(x_2; y) \rightarrow 0.1788$ and $I(x_2; y/x_1) \rightarrow 0.4587$ while $I(x_2; y/x_1) \rightarrow 5.8222$ and $I(x_1; y/x_2) \rightarrow 5.403$. The two rate sums are no longer equal, .01788 + 5.8222 = 5.8301 ≤ 0.4587 + 5.403 = 5.866, and there is a higher rate sum when decoding user 2 first. Non-zero gap reinforces the intuition of best rate sum’s energy placement on only the primary user(s): Basically a non-zero-gap code’s placement of energy on secondary users magnifies the code’s inefficiency. For Example 2.7.1, 5.866 < 6.144 = .5 log₂(1 + 10,000/$\Gamma$), where the latter “overall gap-reduced capacity” really has no direct bearing for non-zero gaps. Any point on the line between [.01788 , 5.8222] and [.4587 , 5.403] could be achieved by time/dimension sharing with sub-users, but the line does not have slope −1, and is not even necessarily the boundary to the “gap-reduced” rate region. Generally, some caution should be exercised with rate regions with non-zero gaps. Jagannathan [6] investigates the construction of non-zero-gap exact achievable rate regions (p. 164-173 of his 2008 Stanford dissertation). His findings show that the rate-sum-bounding-plane method no longer applies. As the gap grows, so does the deviation. His approach basically follows the more general approach of Equations (2.256) to (2.260) in Subsection 2.6.3 and replaces each mutual information by the corresponding $\frac{1}{2} \cdot \log(1+SNR/$\Gamma$) style term and the convex-hull-union process becomes more involved but still produces an achievable rate region. Even this approach assumes that other users can be decoded without error (which may not be true for non-zero code gaps) and that furthermore the users are all Gaussian codes (which of course they may not be so with non-zero gaps). With sophisticated systems, the price of using a powerful code (with close to zero gap) may be only a small fraction of the multi-user design costs, and presumption of good-code use guides practice. Later chapters also address more adaptive approaches to finding best energy allocation for any given $b \in \mathcal{C}(b)$, which may involve non-zero gaps and various artificially
intelligent algorithms to find the “best” multi-user designs for any given situation, but this section’s MAC fundamentals indeed will guide such use. $C_{\gamma>0}(b)$ is always inside $C(b)$, so the latter helps guide design in any case.

2.7.3 Maximum rate-sum calculation for the matrix AWGN

The matrix-AWGN MAC rate sum is always

$$\bar{b} = \log_2 \frac{|H \cdot R_{xx} \cdot H^* + R_{nn}|}{|R_{nn}|}.$$  \hspace{1cm} (2.348)

The maximum over input $R_{xx}$ corresponds to maximizing the numerator inside the log or equivalently

$$\max_{R_{xx}(u)} |H \cdot R_{xx}(u) \cdot H^* + \sum_{i \neq u} H_i \cdot R_{xx}(i) \cdot H_i^* + R_{nn}|$$  \hspace{1cm} (2.349)

the problem separates into $U$ optimizations where each user views all the rest as “noise,” as indicated by $R_{noise}(u)$ in (2.349), specifically

$$R_{noise}(u) \triangleq \sum_{i \neq u} H_i \cdot R_{xx}(i) \cdot H_i^* + R_{nn}.$$  \hspace{1cm} (2.350)

Each optimal $R_{xx}(u)$ satisfies the water-filling criteria for the vector-coded channel $R_{-1/2}(u) \cdot H_u$. The overall rate-maximization problem is convex in the $U$ autocorrelation matrices and thus has a solution that can be found by various “descent” or gradient methods. One such method, “iterative water-filling,” has a strong intuitive appeal and appears shortly. Then, formally:

**Theorem 2.7.4 [Optimality of SWF for the matrix-AWGN MAC]** The autocorrelation matrices $R_{xx}(u)$ that simultaneously water-fill with respect to all other users as noise, maximize the rate sum for the matrix-AWGN MAC.

**proof:** See the preceding paragraph and note that the optimization problem is separable for each of the $R_{xx}(u)$ as a well-known water-filling problem. QED.

More precisely, maximum sum-rate calculation finds these SWF energies through the singular-value-decomposition of the $u^{th}$ user’s noise-whitened channel equivalent

$$R_{noise}(u)^{-1/2} \cdot H_u = F_u \cdot \Lambda_u \cdot M_u^*.$$  \hspace{1cm} (2.351)

Then the energies are assigned according to (where $g_{u,\ell} = \lambda_{u,\ell}^2$)

$$\mathcal{E}_{u,\ell} + \frac{1}{g_{u,\ell}} = \text{constant} \quad \forall \ u,$$  \hspace{1cm} (2.352)

such that

$$\sum_{\ell=1}^{L_u} \mathcal{E}_{u,\ell} = \mathcal{E}_u,$$  \hspace{1cm} (2.353)

and

$$\mathcal{E}_{u,\ell} \geq 0.$$  \hspace{1cm} (2.354)

The $u^{th}$ user’s input is constructed as

$$x_u = M_u \cdot X_u,$$  \hspace{1cm} (2.355)

where the elements of $X_u$ are independent and each has energy $\mathcal{E}_{u,\ell}$.  

304
**iterative water-filling:** A sequence of water-filling steps that cycle one-by-one through all users, and where each step’s user-energy allocation treats all other users as noise, will converge. Figure 2.41 illustrates the basic 2-user capacity region for two users and also illustrates iterative water-filling (IW).

![Diagram](image)

**Figure 2.41:** Illustration that SWF solution achieves MA channel maximum rate sum.

**IW Convergence:** The two users $x_1$ and $x_2$ contribute to the common MAC output $y$. Then, by the chain rule,

$$I(x; y) = I(x_1; y) + I(x_2; y/x_1) = I(x_2; y) + I(x_1; y/x_2)$$

and there are thus two ways to achieve the same rate sum $I(x; y)$. Since either order can be used, the receiver can consider $x_1$ first, and water-fill for user $x_1$ first with user $x_2$ as noise. Reversing the order of decoding maintains the rate sum, but sends the implementation to point B on the same (red) pentagon. But, with the opposite order and maintaining user 1’s $R_{xx}(1)$, user 2 now water fills. Since the receiver decodes user 1 first and does not alter $R_{xx}(1)$, this process maintains user 1’s rate $b_1$. But user 2 must increase its rate (because of water-filling) and so $b_2$ increases on the blue pentagon. Again maintaining the rate sum, the decoder’s order reversal leads to the upper corner point on the colorblueblue pentagon. Since user 2 now maintains the same $R_{xx}(2)$, then user 2’s rate $b_2$ does not change. However, now again reversing order, user 1’s rate increases to point C through another iteration of water filling. Clearly the pentagons must keep moving up and to the right until the rate-sum $b_{max} = b_1 + b_2$ line is SWF everywhere. That SWF everywhere could correspond to a single point (shown in green), or in cases where multiple SWF may exist a 45° line that bounds the capacity region $C(b)$. Thus, the result is established for $U = 2$. By setting a single user $b$ to correspond to the rate sum $b_1 + b_2$ and introducing a 3rd user $x_3$, the concept can be repeated with $b$ replacing $x_1$ and $x_3$ replacing $x_2$. Thus by induction, the iteration of water-filling must converge to a rate-sum-maximizing point and thus SWF.

### 2.7.4 Continuous-Time/Frequency Extension

From Section 2.5 for each user, a single capacity (whether the other users are “noise” or zeroed) follows from water-filling. This subsection generalizes this water-filling to **simultaneous water-filling** for the multi-user Gaussian MAC maximum sum rate.

---

61Since the SWF condition corresponds to a convex optimization, and descent algorithm can be used and will thus converge, the objective here is a more heuristic explanation.
2.7.4.1 Capacity Regions for the Scalar Filtered Gaussian MAC

Subsection 2.7.1’s MAC bounds each user’s energy at $\mathcal{E}_u$. More generally, this bound could be for linear time-invariant channels with pulse responses $h_u(t)$ and Fourier Transforms $H_u(f)$, and then translates $\mathcal{E}_u$ to $P_u$, the individual power constraint that applies to user $u$’s integrated power spectral density. Such channels are limiting cases of Section 2.5’s multi-tone (MT) system for every user (with common symbol boundary for all users). For this subsection’s Gaussian MAC $L_y = L_x = 1$. There may be more than one power spectral density that meets any particular user’s energy constraint. The set of such spectra, a vector power spectral density, over all $U$ users is the $\times 1 S_x(f)$. Any of the corresponding users’ spectra could be used with any of the possible spectra for other users, leading to a large range of possibilities for construction of $A(b, S_x(f))$ and $C(b)$. Each and every different power spectral density $S_x(f)$ creates a possible Gaussian input distribution that contributes to $C(b)$’s convex-hull construction.

Figure 2.42 illustrates $C(b)$’s construction via such a convex-hull union of pentagon regions for $U = 2$. For a specific choice power spectral density choice $S_x(f)$, $A(b, S_x(f))$ is just one pentagon. Different $S_x(f)$ lead to possibly different pentagons. These pentagons’ convex-hull union traces a smooth convex region (the convex hull which is the same as the union for the Gaussian noise case). Any $C(b)$ boundary or interior point may correspond to a different $S_x(f)$. Any such point is reliably achievable with capacity-achieving codes. Interior points may be achievable with $\Gamma > 0$ dB. Dimension-sharing is tacitly present through the continuous frequency $f$, because the designer can always find a pentagon for which one of the corner points corresponds to the selected point on the capacity region boundary. For two flat channels ($h(t) = h \cdot \delta(t)$ or $H(f) = H$), the designer could decide to divide the spectrum according to rate (and user-channel SNR) such that simple frequency-division-multiplex suffices.

The green line with slope -1 is the sum rate line and is tangent to $C(b)$ at the maximum rate

---

62 Even on two flat channels ($h(t) = h \cdot \delta(t)$ or $H(f) = H$), the designer could decide to divide the spectrum according to rate (and user-channel SNR) such that simple frequency-division-multiplex suffices.
While there are infinite dimensions and time, each frequency corresponds to a scalar MAC channel. At each such frequency, there is a primary user and a secondary user that is determined by the larger of the two filtered spectra $|H_1(f)|^2 \cdot S_{x_1}(f)$ and $|H_2(f)|^2 \cdot S_{x_2}(f)$ (presuming AWGN with constant power-spectral density $\frac{N_0}{2}$). This could be a different user as largest at different frequencies. Effectively, each frequency becomes a sub-user for both users. The larger sub-user is primary. There is a rate sum in the absence of individual energy constraints so that only a total energy constraint $\int S_{x_1}(f) df + \int S_{x_2}(f) \leq \mathcal{E}_1 + \mathcal{E}_2 = \mathcal{E}_x$ applies, which then would place all energy on the primary sub-user as long as there was sufficient energy to do so. This is a frequency-division multiplexed (only one sub-user at each frequency) that will achieve maximum rate sum. Such a solution is easily simultaneous water-filling. This type of frequency-by-frequency multi-user channel decomposition finds wide use in MIMO wireline and wireless systems in practice, as Chapters 4 and 5 further address.

Calculation of any channel’s rate region (for any particular choice of input energies and power-spectral densities) can calculate the maximum rate sums for all subsets of users, as earlier in this section. Then if all power spectral densities were somehow enumerated, the convex hull of all $2^U + U – 1$-sided polytopes describes the rate region. The power spectral densities’ enumeration for a Gaussian channel is equivalent to enumerating all input-probability densities $p_x$. Enumeration of all power spectral densities requires forming a discretization of frequency and energy in a double loop that proceeds through all such that the power constraints are satisfied for each choice of user-set energies and corresponding allocations to frequencies for each.

\[
\bar{b} = \sum_{u=1}^{U} b_u \leq \mathcal{I}(x; y) = \int_{-\infty}^{\infty} \frac{1}{2} \cdot \log_2 \left[ 1 + \sum_{u=1}^{U} \frac{S_{x,u}(f) \cdot |H(f)|^2}{S_n(f)} \right] df , \quad (2.357)
\]

where the bits/subsymbol sum becomes an integral over all the infinitesimal small tones of Section 2.5’s MT system. Each user’s individual energy constraint causes the rate-sum maximization problem to differ slightly from straightforward water-filling. Similar to the matrix-AWGN case, forming the Lagrangian for each user’s power-spectral-density integrand with respect to input user spectra and the side constraint of non-negative power spectral density provides

\[
S_{x,u}(f) + \frac{\sigma^2 + \sum_{i \neq u} S_{x,i}(f) \cdot |H_i(f)|^2}{|H_u(f)|^2} = \lambda_u . \quad (2.358)
\]

Since the original function’s integrand was convex and all the constraints are convex (linear inequality), then a solution exists. Each instance of (2.358) for $u = 1, ..., U$ is water-filling with all the other users viewed as “noise.” This leads to the following SWF Lemma extension:

**Lemma 2.7.2 (MT Simultaneous Water-Filling Optimum)** The maximum rate sum $b = \sum_u b_u \cdot u$ for the scalar $(L_u = L_x = 1, y = y, x_u = x_u)$ Gaussian MAC with $u^{th}$ user’s input power spectral density $S_{x,u}(f)$ satisfies

\[
S_{x,u}(f) + \frac{\sigma^2(f)}{|H_u(f)|^2} = \lambda_u \quad (2.359)
\]
where
\[
\sigma^2_u(f) = \sigma^2 + \sum_{i \neq u} S_{x,i}(f) \cdot |H_i(f)|^2
\] (2.360)
with
\[
S_{x,u}(f) \geq 0 ,
\] (2.361)
and
\[
\int_{-\infty}^{\infty} S_{x,u}(f) \cdot df = P_u .
\] (2.362)

\textit{H}_u(f) \text{ is the Fourier transform of the } u^{th} \text{ channel impulse response, and } P_u \text{ is the } U^{th} \text{ user’s power constraint. proof: See above paragraph.}

With a vector energy constraint, order is not important for the maximum rate sum; Gaussian MAC order is only important with a sum-energy constraint. Generalization of this result to vector MACs occurs in Chapter 5.

Figure 2.43 illustrates the more general 2-user AWGN with user-channel filters \( H_1(f) \) and \( H_2(f) \). A special case arises when the two channels have equal filters \( H_1(f) = H_2(f) \): In this case, the rate region remains a pentagon as Figure 2.44 shows with \( C_1, C_2, \) and \( C_{total} \) all being water-filling capacities for the multiple-access channel. Three simultaneous water-filling spectra situations appear in Figure 2.44. Any division of the water-filling spectra that achieves \( C_{total} \) and also satisfies each individual power constraint corresponds to a valid point on the pentagon’s slanted boundary.

Figure 2.44: 2-user AWGN capacity region for \( P_1(D) = P_2(D) \) case.
or FDM) if the channel output is one-dimensional \((L_y = 1)\). This FDM point clearly satisfies SWF individually as each user is zero in the others band. At this point, the sum of the individual user rates treating all others as noise is equal to the maximum rate sum (which is not often true for other non FDM points). An FDM point is CFC and has an independent set of optimum receivers. The infinite number of frequencies renders further “time-sharing” useless; this is a reason this text uses “dimension-sharing” instead of “time-sharing” in nomenclature. Any time sharing can be made equivalent to a frequency sharing because the infinite-length Fourier Transform is indeed a unitary (energy and volume preserving). Chapter 5 investigates more deeply this important concept of simultaneous water-filling.

Simultaneous Water-Filling Examples

**EXAMPLE 2.7.3 ((.8,.6) AWGN revisited)** Figure refcapagain revisits the capacity region for an AWGN MA channel with \(L = N = 1\) and \(U = 2\) with \(P_1 = .8, P_2 = .6\) and \(\sigma^2 = .0001\) is shown in Figure 2.45.

To extend the analysis of this channel, the designer could let the frequency index \(N \to \infty\) while keeping \(L_x = L_y = 1\). With an infinite \(N\), there are an infinite number of simultaneous water-filling solutions possible, some of which are shown in Figure 2.46. Solution (a) corresponds to both users flat over the entire band and use of a GDFE at one of the corner points. Solution (b) corresponds to an FDMA solution that is also SWF and has 47.6% of the bandwidth allocated to User 1 and remaining 52.4% of the bandwidth allocated to User 2. Both are flat over their ranges, and clearly both satisfy SWF criteria. The decoder is 2 independent decoders (no feedback or successive decoding necessary) in this FDM case. Solution (c) corresponds to a mixture of the two that is part FDMA and part sharing of bandwidth – thus a mixture of solution a and solution b. Because the boundary of this capacity region is a non-zero length flat line with slope -1, several SWF solutions are possible for this example, each corresponding to a point on this line. The representative decoder structure is also shown below each choice of spectra.
**EXAMPLE 2.7.4** \(2 \times 2\) **Simultaneous Water-Filling** This example provides a \(2 \times 2\) non-degraded Gaussian MAC. The initial energy constraint is 1 unit on each independent user. A first pass performance analysis in matlab provides the following:

\[
\begin{align*}
H &= \begin{bmatrix} 10 & 8 \\ 3 & 5 \end{bmatrix}; \\
R_f &= H' \cdot H = \\
&= \begin{bmatrix} 109 & 95 \\ 95 & 89 \end{bmatrix}; \\
R_{binv} &= R_f + \text{eye}(2) = \\
&= \begin{bmatrix} 110 & 95 \\ 95 & 90 \end{bmatrix}; \\
G_{bar} &= \text{chol}(R_{binv}) = \\
&= \begin{bmatrix} 10.4881 & 9.0579 \\ 0 & 2.8204 \end{bmatrix}; \\
G &= \text{inv(diag(diag(G_{bar})))*G_{bar}} = \\
&= \begin{bmatrix} 1.0000 & 0.8636 \\ 0 & 1.0000 \end{bmatrix}; \\
S_0 &= \text{diag(diag(G_{bar}))*diag(diag(G_{bar}))} = \\
&= \begin{bmatrix} 110.0000 & 0 \\ 0 & 7.9545 \end{bmatrix}; \\
b_{bar} &= 0.5 * \log_2(\text{diag}(S_0)) = \\
&= \begin{bmatrix} 3.3907 \\ 1.4959 \end{bmatrix}; \\
\text{sum(b_{bar}}) &= 4.8866; \\
W &= \text{inv(S_0)*inv(G')} = \\
&= \begin{bmatrix} 0.0091 & 0.0000 \\ -0.1279 & 0.1257 \end{bmatrix}; \\
W_{unb} &= \text{inv(S_0-eye(2))*S_0*W} = \\
&= \begin{bmatrix} 0.0092 & 0 \\ 0.0090 & 0.1438 \end{bmatrix}; \\
\text{MSWMF} &= W_{unb}*H' = \\
&= \begin{bmatrix} 0.0917 & 0.0275 \\ 1.2401 & 0.7459 \end{bmatrix};
\end{align*}
\]
For this $2 \times 2$ 2-user MAC with necessarily diagonal $R_{xx}$ and energy-vector constraint $\mathbf{E} = [1 1]^*$, the maximum data rate is the sum $4.8866$. This solution is trivially simultaneously water-filling, and indeed this property holds with any fixed energy vector. With the relaxed total energy constraint that $\mathbf{E}_2 + \mathbf{E}_1 = 2$, the solution is also simultaneously water-filling for any specific energy combination, but of interest would be the best energy combination. Such an energy combination maximizes

$$ |S_0| = |H \cdot R_{xx}H^* + I|$$

$$= |H \begin{bmatrix} \mathbf{E}_2 & 0 \\ 0 & \mathbf{E}_1 \end{bmatrix} H^* + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}|$$

$$= -676 \cdot \mathbf{E}_2^2 + 1372 \cdot \mathbf{E}_2 + 179 \text{ uses } \mathbf{E}_1 = 2 - \mathbf{E}_2,$$  (2.365)

which has maximum at

$$\mathbf{E}_2 = \frac{1372}{1352} = 1.0148, \quad \mathbf{E}_1 = .9852. \quad \text{(2.366)}$$

This value corresponds to

$$> E_2 = 1372/1352 = 1.0148$$

$$> E_1=2-E_2 = 0.9852$$

$$> \det(H\cdot\text{diag}([E_2 E_1])\cdot H^*\cdot\text{eye}(2)) = 875.1479$$

$$> \text{bsum} = .5*\log2(\text{ans}) = 4.8867.$$
Clearly all energy on users 3 and 2 (the same as users 2 and 1 earlier) has a higher sum rate of 4.8867, because user 1 is secondary, and thus reduces rate sum when energized.
2.8 Rate Regions for the Broadcast Channel

Section 2.6 describes the general capacity region in Equations (2.258) - (2.260). The single Gaussian BC capacity region also follows more directly from this section’s algorithms. A few basic concepts in Subsections 2.8.1 illustrate BC rate regions, including noncausal or lossless precoders and Forney’s Crypto Lemma. Section 2.8.2 then follows with some direct calculations, while Subsection 2.8.3 describes the vector Gaussian BC’s design (precoder and detectors). Subsection 2.8.3 also introduces the concept of worst-case noise that minimizes a vector channel’s mutual information, while diagonalizing MMSE receiver processing when combined with Subsection 2.8.1’s lossless precoding. The energy-sum MAC concept of primary and secondary users applies there also to the vector BC channel in a dual way, where the (sub-) user partitioning corresponds to BC output user definition instead of the MAC’s input. Section 2.8.4 then introduces scalar duality (a concept first appearing in this text’s earlier versions) to follow the observed symmetric reflection between the MAC and the BC.

This book typically follows a vector- and matrix-element naming convention where the highest index is at the top/left and lowest index is at the bottom/right. Subsection 2.8.4’s broadcast duality simplifies and better follows intuition if the BC’s user 1 is at the top/left and BC user U is at the bottom/right. This is because the BC channel’s user 1 is the dual of a MAC’s user U, so this section’s BC analysis intentionally reverses this indexing convention. Subsection 2.8.5 specializes to the BC’s maximum rate that can be found in an iterative procedure that alternates between water-filling and worst-case-noise steps. Subsection 2.8.6 address continuous-time extension of the Gaussian BC, while Subsection 2.8.7 generalizes the results to the non-Gaussian case.

2.8.1 Basic concepts for Gaussian Broadcast Capacity

This subsection reviews some transmission basics that simplify Gaussian BC capacity rate-region description.

2.8.1.1 Decomposition of Gaussian process into user message components

Lemma 2.6.4 establishes that the AWGN BC’s best single input \( x \), and consequently all its marginal components \( \{x_u \} \}, have Gaussian distributions. BC AWGN analysis and design, as with the AWGN single-user channel and the AWGN MAC, thus best encodes all users with Gaussian-code inputs and \( \Gamma \rightarrow 0 \) dB. The Gaussian BC input \( x \) is also consequently some linear combination of its marginal (also Gaussian) components in vector

\[
x' = \begin{bmatrix} x_1 \\ \vdots \\ x_U \end{bmatrix},
\]

(2.367)

A simple combination for the scalar Gaussian BC is \( x' = \sum_{u=1}^{U} x_u = 1^* x' \), but that is not the only linear combination for the scalar channel, nor for any other channel with \( L_x > 1 \). Following Subsection 2.3.2’s entropy chain rule, each conditional entropy chain that sums to \( H(x') \) corresponds to MMSE estimates of \( x_u \) from \( x_{i>u} \), defining

\[
\nu_u = x_u - \hat x_u/\{x_{u+1}...x_U\}.
\]

(2.368)

These MMSE estimates \( \nu_u \) depend tacitly on the order \( \Pi \). All Gaussian MMSE estimates are always linear, so effectively

\[
\nu = \begin{bmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_U \end{bmatrix} = \begin{bmatrix} 1 & g_{1,2} & \cdots & g_{1,U} \\ 0 & 1 & \cdots & g_{2,U} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_U \end{bmatrix} = G_x^{-1} \cdot x',
\]

(2.369)

equivalently \( x' = G_x \cdot \nu \), where both \( G_x \) and \( G_x^{-1} \) are upper triangular and monic (1’s on the diagonal). Clearly \( |G_x| = 1 \) so transformation of a region by \( G_x \) preserves the region’s volume, and \( G_x \) is invertible. There can be a different \( G_x \) for each possible user-order function \( \pi(u) \). Nonetheless for any order,
\( R_{\nu\nu} \triangleq S_x \) is a diagonal matrix since each successive estimation error \( \nu_u \) must (by Appendix D’s Orthogonality Principle) be independent of all \( x_{i>u} \) and thus then also \( \nu_{i>u} \). The random vectors \( \nu_u \) each have differential entropies \( H_{\nu_u} \) and via the chain rule then

\[
H_x = H_{x'} = \sum_{u=1}^{U} H_{\nu_u} = H_{\nu} \tag{2.370}
\]

for any and all orders. Theorem 2.3.5 and (2.369)’s invertible relationship then permit design to view \( \{\nu_1, \ldots, \nu_U\} \) as independent BC encoder inputs to an encoder generator matrix \( G_x \). This \( G_x \) produces a channel input \( x \) that has autocorrelation matrix \( R_{xx} \).

The sequence \( \nu_{u=1, \ldots, U} \) is sometimes also called the **innovations** of the vector \( x' \), see [3]. The Gaussian BC’s single input \( x \) or equivalently \( \nu \), admits Equation (2.142)’s water-filling energy use over the set of all user dimensions in some special situations later in Subsection 2.8.5. In that situation and some others known as Subsection 2.8.3’s worst-case noise, the MMSE receiver-side matrix filtering essentially becomes a diagonal matrix and thus applicable to the BC.

### 2.8.1.2 The Lossless non-causal “dirty paper” precoder

BC precoding uses a modulo operation that needs definition with respect to an \( L_x \)-dimensional lattice \( \Lambda \):

**Definition 2.8.1 (Modulo Operation)** The modulo operation \( (\nu)_\Lambda \) produces the vector signal in the Lattice \( \Lambda \)’s origin-centered Voronoi Region \( V(\Lambda) \) for the vector input \( \nu \) that is equal to the difference between \( \nu \) and the closest lattice point \( \lambda \in \Lambda \). Further the notation \( \nu \oplus_\Lambda \mu \) means \( (\nu + \mu)_\Lambda \).

Thus for any complex vector \( \nu \), then \( (\nu)_\Lambda = \nu - \lambda_{\nu} \) where \( \lambda_{\nu} \in \Lambda \) is Definition 2.8.1’s closest vector. This is effectively the same operation as ML decoding on the AWGN. The following Lemma is very useful in Gaussian-channel precoding:

**Lemma 2.8.1 (distribution of modulo addition)** Modulo addition distributes as

\[
(\mu + \nu)_\Lambda = (\mu)_\Lambda \oplus_\Lambda (\mu)_\Lambda \tag{2.371}
\]

**Proof:**

\[
(\mu + \nu)_\Lambda = \nu + \mu - \lambda_{(\nu+\mu)} \tag{2.372}
\]

\[
= (\nu)_\Lambda + \lambda_{\nu} + (\mu)_\Lambda + \lambda_{\mu} - \lambda_{(\nu+\mu)} \tag{2.373}
\]

\[
\lambda \triangleq \lambda_{\nu} + \lambda_{\mu} - \lambda_{(\nu+\mu)} \tag{2.374}
\]

\[
(\nu + \mu)_\Lambda = (\nu)_\Lambda + (\mu)_\Lambda + \lambda \tag{2.375}
\]

\[
= (\mu)_\Lambda \oplus_\Lambda (\mu)_\Lambda \ , \tag{2.376}
\]

where the Lattice \( \Lambda \)’s closure under addition means that \( \lambda \in \Lambda \) and thus \( \lambda \) in (2.375) must be the lattice point at minimum distance from the complex sum \( \nu + \mu \). The operation \( \oplus_\Lambda \) by definition removes this lambda so then (2.371) holds directly. QED.
Figure 2.47 illustrates lossless non-causal precoding, which is sometimes also known as “dirty paper” precoding and uses the modulo operation. Non-causal precoding observes that the channel’s addition of a transmitter-known sequence $s$ does not change channel capacity. In 1983, M. Costa first observed this result ([2], references within) for Gaussian channels, and Forney [4] extended it in 2003 through a use of the so-called “crypto-lemma” that corresponds to the blue dashed-outline box and precoder on Figure 2.47’s left.

**Lemma 2.8.2 (Forney’s Crypto Lemma)** Given the additive-noise channel in Figure 2.47 with both $\nu$ and $s$ uniform in distribution over some Voronoi region $\Lambda$, then $x = \text{mod}_\Lambda(x')$ has energy $\mathcal{H}_x = \mathcal{H}_{\nu}$, and $x$ is independent of $s$ and $\nu$.

**proof:** Addition modulo $\Lambda$ is a group operation so that any two points added together will produce another point within the group. Thus by Lemma 2.8.1, $x = (x')_{\Lambda} = (\nu - s)_\Lambda = \nu \oplus s$ since both $\nu \in \mathcal{V}(\Lambda)$ and $s = x - \nu$. Let $\nu$ and $s$ first be random and uniform over $\mathcal{V}_\Lambda$ with $s$ independent of $\nu$, then $p_{x|\nu}(\nu \oplus x) = p_{s}(\nu \oplus x) \equiv$ constant (since $s$ has uniform distribution) for all $x$ and any, and all, particular $\nu$. Thus $x$ has uniform distribution over $\mathcal{V}_\Lambda$ and is independent of $\nu$. Further, since the distributions are the same, $\mathcal{H}_{x'} = \mathcal{H}_x$. QED.

Continuing if $s$ is deterministic and not random nor uniform over $\mathcal{V}(\Lambda)$, then a precomputed uniform (over $\mathcal{V}(\Lambda)$) random dither sequence $d$ may be added to $s$ at the transmitter and subtracted at the receiver without loss. The sum $s \oplus d$ is uniform and the theorem then still applies. (Dither is not necessary in realization, but it helps prove the uniformity and independence.)

The corresponding decoder: Figure 2.47’s right half shows the ML decoder’s form for the lossless non-causal precoder. Figure 2.47 shows neither the good-code’s $\Gamma = 0$ dB encoder nor decoder appears explicitly because they both span infinite time over many subsymbols $\nu$. Indeed for such a code, $\Lambda$ becomes a hypersphere and $x$ is a Gaussian code. When $y = y$ is a scalar, the green shaded boxes’ function reduces to passing $y$ through. In this simple scalar case with $H = h = 1$, then the modulo device processes $y$. This leads to

\[
(y)_\Lambda = (x + s + n)_\Lambda \\
= ((\nu - s)_\Lambda + s + n)_\Lambda \\
= (\nu)_\Lambda \oplus_{\Lambda} s \oplus (n)_\Lambda \\
= \nu \oplus_{\Lambda} (n)_\Lambda.
\]

The channel output is basically the channel input $\nu$ plus noise. Indeed, even if the code is not Gaussian and simply has continuous uniform distribution (or is a code selected at random from such a distribution uniform on the lattice’s Voronoi region), this same result holds for any lattice $\Lambda$. Generalizing slightly,
such a code will achieve reliably a bits/symbol equaling the mutual information $I_{\lambda}$ from a view of $s$ as a "key" ($y$ proceeds to find the codeword closest to the sequence of outputs $\nu$). For coding applications, this implication for the AWGN and Gaussian capacity-achieving codes becomes evident later.

When $H = h \neq 1$, the MMSE (from Appendix D) is $\sigma^2_{\text{mmse}} = \bar{\mathcal{E}}_x - (|h|^2 \cdot \bar{\mathcal{E}}_x)/\sigma^2$. If $S_0 \triangleq \bar{\mathcal{E}}_x/\sigma^2_{\text{mmse}}$, then the first MMSE receiver box inherently biases or scales the subsymbol by $(S_0 - 1)/S_0$ and the second box inverts this scaling. Indeed this happens independently for all dimensions of any diagonal $H$ also.

However, when $H$ is a non-diagonal matrix, then the MMSE estimation is non-trivial and will maximize the ratio $\bar{\mathcal{E}}_x/\sigma^2_{\text{mmse}}$, since $\sigma^2_{\text{mmse}} = \text{trace}(\text{Re}e\{\text{Re}e\})$ is, by definition minimum. As with the MAC and single-user MMSE cases, if the MMSE output is $x$, then $E[L_2(x)] = (1 - 1/S_0) \cdot x$, then the subsymbol amplitude needs to increase by $S_0/(S_0 - 1)$ so the decoder has the right (unbiased) scaling on all subsymbol dimensions $i = 1, ..., L_x = L_y$ and is ML. Without the MMSE and the bias scaling removal, there will be performance loss (often small, but nonzero) in any case other than the trivial scalar cases where there is no inter-dimensional interference (crosstalk). This error often occurs in the preceding and successive-decoding literature where a zero-forcing precorder is used incorrectly, so the reader might best be vigilant in alternative review. Part of the literature confusion on this arises from failure to distinguish the subsymbol bias from the infinite-length codeword bias. The latter codeword bias does not change the infinite-length ML decoder decision regions prior to the outer AWGN ML decoding. However, it does affect decoders that use subsymbols where receiver should remove the bias.

The MMSE estimator on the receiver input also automatically addresses any dimensional differences when $L_y \neq L_x$ and will produce an $L_x$-dimensional output consistent with $x$ and $\nu$ that are both $L_x$ dimensional.

**More precoder observations** The name “crypto” arises from the independence of $\nu$ from $x$, which basically means the input $\nu$ is not recoverable if $s$ is an encryption key. The name “dirty paper” arises from a view of $s$ as existing writing on paper, and then $x$ is new writing on the same paper that contains a “key” ($-s$ hidden within the new writing) such that the receiver can “cleanse” the paper through modulo application. Figure 2.47’s output dimensionality is trivially $L_x$. More sophisticated channels require encoder mappings that reconcile various numbers of dimensions at different modulo locations (as becomes evident later).

**Implication for the AWGN and Gaussian capacity-achieving codes** For coding applications, the crypto lemma applies to any lattice $\Lambda$ of $L_x$ dimensions from which a good code effectively selects the subsymbols $\nu$ (or equivalently $x$) from a continuous uniform distribution over $\mathcal{V}(\Lambda)$. By the AEP, such a code will achieve reliably a bits/symbol equaling the mutual information $I(\nu; y)$ for that uniform distribution. The ML detector begins with Figure 2.47’s modulo (and unbiased MMSE) and then proceeds to find the codeword closest to the sequence of outputs $y$ without influence of $s$. When the input has an average energy constraint $\mathcal{E}_x$ then

$$x \in \{x \mid E[\|x\|^2] \leq \mathcal{E}_x\},$$

(2.381)

and $\mathcal{V}(\Lambda)$ and the energy constraint are not quite consistent; unless the number of subsymbols in a codeword becomes infinite $N \to \infty$. Then essentially the subsymbol selected from $\Lambda$ tends towards selection from a Gaussian distribution where $\Lambda \to \mathcal{S}_N$, the on-average $N$-dimensional complex hypersphere’s interior where

$$\mathcal{S}_N \triangleq \{x \in \mathbb{C}^N \mid E[\|x\|^2] \leq \mathcal{E}_x\}.$$  

(2.382)

The subsymbol’s finite-dimensional ($L_x = \bar{N} < \infty$) marginal distribution becomes Gaussian while the overall symbol/codeword distribution is uniform within the infinite-dimensional hypersphere $\mathcal{S}_\infty$. The latter uniform distribution over the ball of $\mathcal{S}_\infty$ meets directly the crypto lemma requirement. However,  

---

63 These can be envisioned as “hypercones” extending to encompass each one unique typical-set codeword that with probability 1 is on the hypersphere’s (and thus hypercone’s) surface.

64 If an eavesdropper does also know $s$.

65 A ball means the hypersphere’s interior.
the crypto lemma’s dither-argument proof for non-uniform \( s \) also applies to the subsymbol’s non uniform (that is, Gaussian) distribution. While encoding of \( \nu \) may require infinite delay, good codes may well approximate this subsymbol Gaussian marginal distribution and may be known at each subsymbol period. Thus the modulo operation might well reflect any \( \Lambda \approx \mathcal{S}_N \), with some shaping loss that reduces as \( N \) becomes large. The AEP suggests more generally that any codeword distribution limited to being uniform in its subsymbols’ marginals is optimum and meets \( b \rightarrow I(x;y) \); when those subsymbols have themselves large finite-dimensional constellations \( |C| \rightarrow \infty \). With an average-energy constraint (not a uniform subsymbol distribution), then all finite-dimensional marginal distributions become Gaussian, and the infinite-dimensional distribution is uniform over \( \mathcal{S}_\infty \), and \( b \rightarrow C \). Clearly \( I \leq C \).

**Implication for the Gaussian BC**. The crypto lemma has important use in Gaussian BCs: This use is asymptotic as \( N \rightarrow \infty \). Here, the side information \( s \) becomes other previously encoded users. Once the encoder knows the subsymbols from other users’ encoders, implying again a user order \( \pi(u) \), these “equivalent index” users can be part of \( s \). Then the current \( \nu_u \), or equivalently \( x_u \), passes to its ML decoder through Figure 2.47’s initial decoder processing as if those other users are not present. Of course those other users may reduce the amount of \( \mathcal{E}_x \) that is available for user \( u \), but they create no crosstalk. Technically, the term “non-causal” is correct, but in practice good codes will indeed know other users’ subsymbols with finite delay, often at the subsymbol period encoding, and so the order \( \pi(u) \) also applies to the subsymbols in implementation practice. The precoder designs that follow assume this ordered-other-user à priori knowledge within each subsymbol (possibly with some delay).

To avoid specific lattice choice \( \Lambda \), the modulo operation here generalizes from modulo \( \mathcal{V}_\Lambda \) to \( \mathcal{V}_\mathcal{E}_u \); this operation preserves user \( u \)’s energy \( \mathcal{E}_u \), and views the Voronoi region \( \mathcal{V}_{\mathcal{E}_u} \) as the interior of origin-centered hypersphere as in Section 2.1’s Gaussian-code sphere packing with marginal distributions appropriately approaching Gaussian. The total energy will be \( \mathcal{E}_x = \sum_{u=1}^U \mathcal{E}_u \) for the addition of all the independent users’ energy at the channel input. The last encoded user’s subsymbol experiences no crosstalk from earlier-encoded users’ subsymbols, while the first encoded user must consider all others as crosstalk. The multi-user BC channel input thus adds \( s = -\sum_{i=u+1}^U g_{u,i} \cdot x_i \), where \( g_{u,i} \) has been trivially \( g_{u,i} = 1 \) when \( H = I \), and will be specifically determined in other cases. The precoder’s pre-modulo subtraction of this same “side information as other users” ensures that the receiver’s second modulo operation restores \( \nu \). The encoder re-adds the signal, \( s \) to \( x \) prior to channel entry. The ideal precoder modules in Figures 2.47 and 2.52 retain exact energy and Gaussian distribution for an infinite-length code that reuses the channel for many successive subsymbol transmissions because they satisfy the limiting case of the Crypto Lemma. Successive decoding alternately can be used independently at each of \( U - 1 \) receivers as in Figures 2.52 and 2.50, as Subsection 2.8.2 further discusses.

2.8.2 Scalar Gaussian BC Channel Capacity regions

Figure 2.48 illustrates three equivalent versions of the two-user scalar AWGN BC with \( L_x = L_y,1 = L_y,2 = 1 \). Figure 2.48a shows the 2-user BC as is. Figure 2.48b shows the channel with presumed receiver noise-whitening/normalization. Figure 2.48c illustrates the same channel, but with user 1’s signal as noise for user 2’s receiver and with again presumed receiver noise-whitening normalization. This simple case loses no generality with equal noise variances on the two channels and order such that \( |h_1| \geq |h_2| \).

---

*66* Again, as with the MAC, this statement qualifies as “no harmful” crosstalk.
More generally, the $U$-user scalar\footnote{when either or both of $L_x > 1$ or $L_y > 1$, then the $g_u$ definition is $g_u = |H_u \cdot R_{nn}^{-1}(u) \cdot H_u|$ and then also orders the users.} Gaussian BC has given channel gains

\[ g_u = \frac{|h_u|^2}{\sigma_u^2}, \]  

(2.383)

with a chosen order such that

\[ g_1 \geq g_2 \geq \ldots \geq g_U, \]  

(2.384)

or equivalently $|h_u| = \sigma_u \cdot \sqrt{g_u}$. Effectively, the noise variances are set equal because any difference can be accommodated in redefining the channel transfer factors $h_1$ and $h_2$. The scalar BC energy partitions between the two users who share the common, scalar, channel-input signal $x$ so that

\[
\begin{align*}
\mathcal{E}_1 &= \alpha \cdot \mathcal{E}_x, \\
\mathcal{E}_2 &= (1 - \alpha) \cdot \mathcal{E}_x \\
\mathcal{E}_1 + \mathcal{E}_2 &= \mathcal{E}_x,
\end{align*}
\]  

(2.385) (2.386) (2.387)

where $0 \leq \alpha \leq 1$. The dimensionality of $\mathcal{E}_1$ and $\mathcal{E}_2$ is the same as $\mathcal{E}_x$ and the noise in any SNRs. Thus, if all are 1 real dimension, then the noise energy is the AWGN power-spectral density level $\sigma^2 = \frac{N_0}{2}$.

### 2.8.2.1 Scalar successive decoding

Before specific further BC-precoder study, successive decoding alternatively applies to (non-precoded) BCs. Figure 2.49 shows the successive decoders that apply to Figure 2.48b and 2.48c channels. (Figure 2.49 removes the bars from $y$’s to avoid superfluous notation.) Receiver 1 first decodes and removes scalar signal $x_2$ from $y_1$ before decoding scalar $x_1$. Receiver 2 treats user 1’s signal $x_1$ as Gaussian noise. Then the data rates have bounds:

\[
\begin{align*}
\bar{b}_1 &\leq I(x_1; y_1/x_2) = \frac{1}{2} \log_2 \left( 1 + \alpha \cdot \hat{\mathcal{E}}_x \cdot \tilde{g}_1 \right) \\
\bar{b}_2 &\leq I(x_2; y_2) = \frac{1}{2} \log_2 \left( 1 + \frac{(1 - \alpha) \cdot \hat{\mathcal{E}}_x \cdot \tilde{g}_2}{1 + \alpha \cdot \hat{\mathcal{E}}_x \cdot \tilde{g}_2} \right)
\end{align*}
\]  

(2.388) (2.389)
The bound (2.389) presumes that user 2’s receiver considers any $x_1$ component as (Gaussian) noise; however this is optimum as per Lemma 2.6.4.

Reverse order has lower bounds: With reverse order, receiver 2 attempts to decode first user 1 (treating necessarily then user 2 as Gaussian noise). However, receiver 1’s $b_1$ (also with user 2 as noise) must then reduce data rate to the level of receiver 2’s $b_1$ with user 2 as noise because $g_1 > g_2$. This receiver-1 rate reduction holds for any specific two-user non-zero energy allocation. Further, both receivers’ subsequent user-2 decoding must also decrease at receiver 1 because $g_1 > g_2$. Since both users’ bits/symbol must be lowered at receiver 1, the original order of decoding first user 2 and removing it at receiver 1 provides a better rate pair. This concept easily extends to $U > 2$ by induction: Users 1 and 2 the become a single macro-user 1, and user 3 then effectively renames to user 2 and the same argument inductively applies.

General scalar BC successive decoding: Figure 2.50 generalizes the 2-user scalar Gaussian BC to $U$ users. Figure 2.50 tacitly repeats for receiver’s implementation at each receiver location. Both the single BC transmitter and a corresponding receiver $u$ appear. There are $U$ such receivers, but only 1 transmitter for all users. Thus, successive decoding appears more complex than precoding for BC implementation because of the repeated successive-decoding implementations. Figure 2.50 also uses the individual $h_u$ directly rather than the scalar $g_u$ values that Figures 2.48 and 2.49 use. This is because the noise whitening is unimportant for the BC scalar case, and the total noise is the white channel noise plus the other later-decoded users. Figure 2.49’s dashed green boxes can be omitted if the ML decoders target $h_u \cdot x_{i \geq u}$ directly. The receivers first scale by $h_u^{-1}$, then decode user $U$, remove it after multiplication by $h_U$, then user $U - 1$ and remove it, and so on. This scalar case generalizes to a vector case later, and then the simple division by $h_u$ generalizes to Figure 2.47’s MMSE estimate and bias removal.
2.8.2.2 Scalar precoding

Figure 2.51 details a simple 2-user Gaussian BC scalar precoder and corresponding detectors. Users 1 and 2 both use Gaussian codes, which means that $\Gamma \approx 0$ dB. Further, $g_1 > g_2$. The encoder processes the information bearing signals $\nu_1$ and $\nu_2$ respectively into codewords $x_1$ and $x_2$ that are ideally infinity long (Gaussian) subsymbol sequences, with $N \rightarrow \infty$. The modulo device presumes this infinite-length encoding creates Gaussian-distributed subsymbols $x_u$, and consequently the modulo would need infinite delay to perfectly reflect a signal inside the origin-centered hypersphere $S_{\infty}$. Specifically, the ideal non-causal modulo device thus translates any codeword sequence $\nu_1$ into signal sequence $x_1 = \left(\sqrt{\alpha \cdot E_x} \cdot \nu_1 - x_2\right)_{E_1}$ (where $E_1 \triangleq \alpha \cdot E_x$). This $\text{mod}(\bullet)_{E_{\text{G}}} E_{\text{G}}$ operation reflects sequence values outside the hypersphere of radius $\sqrt{E_1}$ into an equivalent point inside that hypersphere. Specifically, the transmitted signal $x_1$ equals the difference between $\nu_1 - x_2$ and the closest hypersphere center point. Thus, the modulo output has energy $E_1$, no matter what the input energy. User 2’s signal $x_2 = \sqrt{(1-\alpha) \cdot E_x}$ has energy $E_2 = (1-\alpha) \cdot E_x$. As Figure 2.51 shows, $E_{\text{G}} = E_1 + E_2$ on the BC input.
Understanding Figure 2.51: Users 1 and 2 share the same single dimension on this “degraded” Gaussian BC with $\varrho_H = 1 < U = 2$. This means the precoder’s single transmit dimension/subsymbol shares energy so that user 1 consumes energy fraction $\alpha$, and thus user 2 consumes fraction $1 - \alpha$. The precoder subtracts user 2’s signal from user 1 prior to the transmitter modulo, so the modulo-input energy increases because the two users’ signals are independent; however the modulo resets path 1’s energy to $E_1$. Users 1 and 2 consequently have two components in each of

$$y_1 = \sqrt{g_1} \cdot x + n_1$$  \hspace{1cm} (2.390)

$$= \sqrt{g_1} \cdot (x_1 + x_2) + n_1$$  \hspace{1cm} (2.391)

$$= \sqrt{g_1} \cdot (\sqrt{E_1} \cdot \nu_1 \oplus x_2) + n_1$$  \hspace{1cm} (2.392)

$$\left( \frac{y_1}{\sqrt{g_1}} \right) = \sqrt{E_1} \cdot \nu_1 \oplus \frac{n_1}{\sqrt{g_1}}$$  \hspace{1cm} (2.393)

$$\approx \sqrt{\alpha \cdot E_x} \cdot \nu_1$$  \hspace{1cm} (2.394)

and

$$y_2 = \sqrt{g_2} \cdot (x_2 + x_1) + n_2$$  \hspace{1cm} (2.395)

$$= \sqrt{g_2} \cdot \left[ x_2 + (\nu_1 \oplus x_2) \nu_1 \right] + n_2$$  \hspace{1cm} (2.396)

$$\frac{y_2}{\sqrt{g_2}} = \sqrt{(1 - \alpha) \cdot E_x} \cdot \nu_2 + \left( \sqrt{\alpha \cdot E_x} \cdot \nu + \frac{n_2}{\sqrt{g_2}} \right).$$  \hspace{1cm} (2.397)

Equations (2.393), (2.397), and Figure 2.51 shows decoder noises in the color red: User 1 has only (scaled) noise in its non-user-1 component; but user 2 has both components of user 1 and noise in its non-user-2 component. Figure 2.51’s receiver signal $\nu$ is random (Gaussian) with unit energy and is the signal at the precoder modulo output that is independent of user 2. The component $\nu$ thus has the same (Gaussian) distribution as both $\nu_1$ and/or $\nu_2$. User 1’s receiver modulo element removes both the user 2 components because they are equal and opposite in sign, modulo $E_1$, leaving only the noise $g_1^{1/2} \cdot n_1$. Effectively, user 2’s receiver sees $\nu$ equivalent to $x_1$ with energy $g_2 \cdot E_1 = \alpha \cdot g_2 \cdot E_x$. The crosstalk noise $\sqrt{g_2} \cdot \alpha \cdot E_x \cdot \nu$ adds to channel noise $n_2$. This total receiver 2 noise is not such that receiver 2 can reliably decode user 1. The single-user capacity theorem with user 2 as noise to user 1’s detection at receiver 2 ensures that receiver 2 cannot reliably decode user 1. The precoder is equivalent to the set of successive decoders, although arguably simpler, being implemented only once in the BC transmitter.

More scalar BC users: For $U > 2$, the transmitter uses a sequence of lossless precoders. The user order is always $U, U - 1, ..., 1$ in the scalar BC because, by induction from the case of 2 users, all other rate points for other orders will lie within this region (as long as energies are such that their sum is the maximum sum energy allowed). To trace the $C(b)$ boundary, then the designer needs to compute rate tuples for all possible combinations of $U$-way energy assignments that sum to the total energy. This requires in general $U - 1$ energy factors $\alpha_u$ such that

$$E_u = \alpha_u \cdot E \quad \forall \ u = 1, ..., U$$  \hspace{1cm} (2.398)

where $\alpha_U = 1 - \sum_{i=1}^{U-1} \alpha_i$ and

$$0 \leq \alpha_u \leq 1.$$  \hspace{1cm} (2.399)

The $U$-user scalar precoder essentially implements the crypto lemma as in Figure 2.52.
Figure 2.52 later generalizes to Subsection 2.8.3’s vector case, where many basic functions expand.

**A degraded-channel example**  A 2-user scalar Gaussian BC helps illustrate various effects:

**EXAMPLE 2.8.1 (Simple Broadcast Channel)** This example returns to Figure 2.48 and sets $h_1 = .8$ and $h_2 = .5$ and $\sigma^2 = .0001$. The single-user mutual information upper bound from (2.135), which may require receiver coordination and thus not be attainable, is

$$I(x; y) = \frac{1}{2} \log_2 \left( \frac{|R_{yy}|}{|R_{nn}|} \right) = \frac{1}{2} \log_2 \left( \frac{(.6401) \cdot (.2501) - .4^2}{.0001^2} \right) = 6.56 \text{ bits/dimension.}$$  

(2.400)

The sum of the data rates thus cannot exceed 6.56 bits/dimension for the BC. Using the exact formulas in (2.388) and (2.389) for various $\alpha$ produces the following table:

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$\bar{b}_1$</th>
<th>$\bar{b}_2$</th>
<th>$\bar{b} = \bar{b}_1 + \bar{b}_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>6.32</td>
<td>0</td>
<td>6.32</td>
</tr>
<tr>
<td>.75</td>
<td>6.12</td>
<td>.20</td>
<td>6.32</td>
</tr>
<tr>
<td>.50</td>
<td>5.82</td>
<td>.50</td>
<td>6.32</td>
</tr>
<tr>
<td>.25</td>
<td>5.32</td>
<td>1.0</td>
<td>6.32</td>
</tr>
<tr>
<td>.10</td>
<td>4.66</td>
<td>1.66</td>
<td>6.32</td>
</tr>
<tr>
<td>.05</td>
<td>4.16</td>
<td>2.15</td>
<td>6.31</td>
</tr>
<tr>
<td>.01</td>
<td>3.01</td>
<td>3.29</td>
<td>6.30</td>
</tr>
<tr>
<td>.001</td>
<td>1.44</td>
<td>4.74</td>
<td>6.18</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>5.64</td>
<td>5.64</td>
</tr>
</tbody>
</table>

Table 2.4: Example 2.8.1’s BC energy choices and corresponding data rates.

The rate sum simplifies with algebra for $\alpha \gg \mathcal{E}_x/g_2 = 1/2500 = .0004$ to

$$\bar{b} = \frac{1}{2} \log_2 \left( \frac{(1 + \mathcal{E}_x \cdot g_2) \cdot (1 + \alpha \cdot \mathcal{E}_x \cdot g_1)}{1 + \alpha \cdot \mathcal{E}_x \cdot g_2} \right) = \frac{1}{2} \log_2 (\mathcal{E}_x \cdot g_1) = 0.5 \cdot \log_2 (80^2) = 6.32$$  

(2.401)

The corresponding rate region appears in Figure 2.53.
Slope = -1

\[ \alpha = 0.05 \]
\[ \alpha = 0.01 \]
\[ \alpha = 0.001 \]

4.74
5.64
3.29
2.15

\[ b_1 \leq I(x_1; y_1/x_2, x_3, \ldots, x_U) = \frac{1}{2} \log_2 \left( 1 + \frac{\alpha_1 \cdot \mathcal{E}_x \cdot g_1}{\Gamma \cdot \sigma^2} \right) \] (2.408)

Non-Zero Gap; The rate regions in (2.388) and (2.389) can approximate the situation for nonzero gaps.

\[ b_1 \leq I(x_1; y_1/x_2) = \frac{1}{2} \log_2 \left( 1 + \frac{\alpha \cdot \mathcal{E}_x |h_1|^2}{\Gamma \cdot \sigma^2} \right) \] (2.409)
As with the MA case, these regions are approximate because the gap approximation is not exact. Non-zero-gap achievable regions expand to more than two users in the obvious way. Further, as with the MAC, the results dilute the gap concept as $\Gamma$ increases. An investigation$^{68}$ awaits the efforts of a motivated communication expert. Best BC designs might well use $\Gamma = 0$ dB to avoid the complexity induced by non-zero gap design, or simply accept accuracy and loss that grow with $\Gamma$ (unlike the single-user case). The $\Gamma > 0$ dB region is then a subregion $\mathcal{C}_\Gamma(b) \subset \mathcal{C}(b)$ with boundary separation $\{\mathcal{C}(b) \setminus \mathcal{C}_\Gamma(b)\} \ni \emptyset$ increasing with $\Gamma$.

### 2.8.3 The vector Gaussian BC design

The vector Gaussian BC case extrapolates from the $2 \times 1$ scalar case into primary and secondary (sub-) user sets. This subsection defines that vector Gaussian BC. Subsection 2.8.3.1 generalizes the precoder and finds the BC’s primary dimension set that best supports primary users and that energized secondary users must share. This expands upon Section 2.7’s energy-sum MAC’s primary and secondary users. Subsection 2.8.3.2 introduces worst-case mutual-information-minimizing overall noise that helps develop the correct BC precoders and receivers. Subsection 2.8.3.3 then describes channel normalization that best occurs prior to Subsection 2.8.3.4’s design steps that proceed from a given input autocorrelation $R_{xx}$, its worst-case noise, channel normalization thereof, and finally the noiseless precoder and a optimum diagonalized MMSE optimum multi-user receiver that uses worst-case noise in its design.

---

$^{68}$Such as done by S. Jagannathan in Problem 2.21, which suggests a “true” margin only against channel noise and not other users. This concept can lead to variation of user-energy strategy for large gaps, and suggests a machine-learning solution for a future researcher.
simplifies analysis. A good number-of-sub-users choice is

\[ U' = \sum_{u=1}^{U} \min(\varrho_u, \varrho_{H,u}) \leq \varrho_H, \]

(2.410)

with equality at least when there is no \( R_{xx} \) constraint beyond the energy-sum constraint \( \text{trace}\{R_{xx}\} \leq \mathcal{E}_x \). The latter energy-sum condition permits excitation of every possible nonzero energy/information transfer to any channel output.

### 2.8.3.1 Finding BC Primary and Secondary User Components

Like the energy-sum MAC, secondary user components must zero for the max-rate sum to occur. Figure 2.54’s \( U' \) (sub-) user components map into the primary (sub-) users’ encoder input signals \( \nu_{U+1}',...,\nu_U' \) and the secondary (sub-) users’ encoder input signals \( \nu_{U+1}',...,\nu_U' \). Again, \( U' = U^o + U^s \). The secondary components will use dimensions that otherwise find best use carrying only primary components. The number of channel outputs is

\[ \mathcal{L}_y = \sum_{u=1}^{U} L_{y,u} \geq U' \]

(2.411)

where the bound by \( U' \) follows from (2.410) and \( L_{y,u} \geq \varrho_{H,u} \). \( U' \) need not include zero-energy dimensions when there is a specified \( R_{xx} \). Both \( y \) and \( n \) are \( \mathbb{Z}_y \times 1 \) vectors. The nonsingular noise components all have standard inverse square-root noise whitening to produce Figure 2.54’s \( y_u', u = 1, ..., U \) and \( n'_u \) where \( R_{uu'}(u) = I_{L_{y,u}} \). This subsection briefly returns to the scalar Gaussian BC for guidance and then extends to the vector case.

**Scalar Gaussian BC primary and secondary users:** The scalar Gaussian BC is always degraded for \( U > 1 \) because \( L_x = 1 \) and thus \( \varrho_H = 1 < U \), recalling that degraded Gaussian AWGN matrix channels have \( \varrho_H < U \) (or \( \varrho_H < U' \) with sub-users). In the scalar Gaussian BC, user 1 fully uses the energy allocated to it as if user 2 is not present. User 1 is thus a primary user. User 2 treats user 1 as noise as is thus a secondary user. This concept is similar to Subsection 2.7.2.2’s energy-sum MAC except that the mixing of secondary user’s dimensions into primary users’ dimensions occurs at the BC input. The MAC’s dimensional energy superposition, by contrast, occurs within the channel. User 1’s successive-decoding receiver decodes all users’ signals that are present at its input. Equivalently user 1 precodes last with full knowledge of all users \( u = 2, ..., U \) as side information. This generalizes as per Figures 2.50 through 2.52 to user 1 as primary and users 2, ..., \( U \) as secondary. The scalar Gaussian BC’s secondary users share the primary users’ dimensions and reduce rate sum if any have nonzero energy. Any dimension- (time-)sharing’s expansion indirectly increases \( L_{x,u} \), and thus effectively increases. Such expansion effectively increases channel rank \( \varrho_H \) so that more (or all) users become primary (sub-) users, but dimension sharing may not always be practical. Secondary (sub-) users have smaller contribution to rate sum at the same energy level as a primary user.

Designs that consider other users as noise, i.e. those that have at least one secondary user, sometimes have the name **Non-Orthogonal Multiple Access (NOMA).** The set of primary users is \( \{u^o\} \subseteq \{U\} \). There is always at least 1 primary user. The set of secondary users is therefore \( u^s = U \setminus u^o \). The primary set size is \( U^o \Delta = |u^o| \), while the secondary set size is \( U^s = |u^s| \). As with the energy-sum MAC, \( U^s = U - U^o \), and \( U \rightarrow U' \), and as well \( U \rightarrow U' \) if there are sub-users.

---

69Indirectly because this type of sharing implies splitting a dimension into two dimensions, typically through higher-dimensional symbols that result from concatenation of several subsymbols that may correspond each to one of the time-dimensionally shared solutions. See footnote 70

70Dimension sharing increases subsymbol dimensionality (usually in time) through subsymbol packets that use 2 or more codes for respective fractions (of time). Thus \( \varrho_H \) increase and all-primary (sub) users may become possible through the enlarged subsymbol packets.

71For instance, time-sharing increases transmitter and receiver implementation delay, which may be unacceptable.

72The “multiple-access” in NOMA refers to the users, not to the channel type, so specifically MA is not necessarily the same MA as in MAC.
Vector Gaussian BC primary and secondary users: The vector BC case uses (2.410) and then compares \(U'\) with \(\varrho_H\). If \(U' \leq \varrho_H\), the channel is non-degraded and there are no secondary (sub-) users. When \(U' > \varrho_H\), there are secondary (sub-) users (if energized, these reduce the rate sum). The original-user matrices individually “noise-whiten” at each receiver to

\[
\tilde{H}_u \triangleq R_{\text{nn}}^{-1/2}(u) \cdot H_u .
\]  \hspace{1cm} (2.412)

\(\tilde{H}_u\) is a function only of \(R_{\text{nn}}(u)\), not the full \(R_{\text{nn}}\) so dependent only on the (block) diagonal elements of \(R_{\text{nn}}\), which will shortly not be true of worst-case noise that also has the same block diagonal elements and a specific correlation between them. A related matrix is

\[
\tilde{H}_{BC} \triangleq \begin{bmatrix}
\tilde{H}_1 \\
\tilde{H}_2 \\
\vdots \\
\tilde{H}_U
\end{bmatrix} .
\]  \hspace{1cm} (2.413)

The rows of \(\tilde{H}_{BC}\) essentially create a potential receiver sub-user set as large as \(U' \leq \mathcal{L}_y\). When \(L_{y,u} = 1\), then each row of \(\tilde{H}\) associates with a user/receiver, which assumption helps to understand the following process initially (so then \(U' = U\) initially). With this assumption, there will be \(\varrho_H = U^o\) primary users. Each such receiver has output

\[
y_u =  \hat{h}_{u,1} \cdot x_1 + ... +  \hat{h}_{u,U} \cdot x_U .
\]  \hspace{1cm} (2.414)

\(\tilde{H}_{BC}\)’s row indices will be \(i \in I_{BC}\) and the column indices \(j \in J_{BC}\). Since any of the \(x_u\) may linearly combine \(v_{u=1,...,U}\) inputs, the largest element

\[
\hat{h}_{\text{max}} = \max_{i,j} \left| \hat{h}_{i,j} \right| \quad \forall i \in I_{BC} \land j \in J_{BC}
\]  \hspace{1cm} (2.415)

determines the first user with index \(i_1\) in first position, which is then assigned to index \(i\) in a new order. That user \(i_1\) now becomes noise to any subsequent user, while decoding without crosstalk interference other users. Any second primary user (so \(U^o \geq 2\)) now examines

\[
\hat{h}_{\text{max}} = \max_{i,j} \left| \hat{h}_{i,j} \right|^2 / (\left| \hat{h}_{i,i_1} \right|^2 + 1) \quad \forall i \in \{I_{BC} \setminus i_1\} \land j \in \{J_{BC} \setminus \{i_1\}\}
\]  \hspace{1cm} (2.416)

which determines the second user as \(i_2\). For yet larger \(U^o\), the search continues now with both \(i_1\) and \(i_2\) (generally \(i_m, m = 1,...,U^o\)) by finding possible channel gains with the earlier users as noise. The maximum determination becomes increasingly complex as the number of primary users increases. An exception occurs in this process if the reduced channel \(H^o\) that includes only primary users becomes singular. Any addition of a user to \(u^o\) that reduces rank (so then \(\varrho_{H^o} < \varrho_H\) voids that user’s addition to the primary-user set, and the next best user that does not introduce singularity instead joins the primary-user set.

When the BC outputs individually have multiple dimensions with \(L_{y,u} > 1\), the MMSE equivalents must be used in the search in Equation (2.416). This can be complicated, so the next subsection’s worst-case noise finds a simpler way to determine the two sets of users.

Energized primary users thus have bigger impact on any associated mutual information \(I(x; y)\) that has \(\{R_{\text{nn}}(u)\}\), and a water-fill energy distribution can optimally allocate energy thereto. If there are equal singular values at the boundary of the groups, the designer then picks the sub-user from the user group according to secondary criteria. Placement into the primary group is advantageous. This tie-splitting issue disappears when different \(R_{xx}\) possibilities are searched later, because the search process essentially attempts all possible equal-user assignments in constructing a full rate region, which depends upon a specified \(R_{\text{nn}}\).

2.8.3.2 Worst-Case Noise Autocorrelation and the Gaussian BC

For any linear Gaussian noise channel\(^{73}\)

\[
y = Hx + n
\]  \hspace{1cm} (2.417)

\(^{73}\)The notation \([U \times U] \cdot L_y\) means there are \(U^o\) matrix elements are \(L_y \times L_y\).
with given input autocorrelation matrix $R_{xx}$, there is a worst-case-noise autocorrelation for a given per-user noise energy set $\{\sigma_u^2\}_{u=1,\ldots,U}$, which generalize to a $L_y \times L_y$ block-diagonal matrix $R_{nn}$ (where each $R_{wcn}(u)$ is a $L_y \times L_y$ block-diagonal matrix)

$$\text{Diag}_{\text{block}} \{R_{nn}\} = \begin{bmatrix} R_{wcn}(1) & 0 & \ldots & 0 \\ 0 & R_{wcn}(2) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & R_{wcn}(U) \end{bmatrix}. \quad (2.418)$$

This $R_{wcn}$ (to be determined below) minimizes mutual information. $R_{wcn}(u)$, $u = 1,\ldots,U$ are the given BC noise autocorrelation matrices at each corresponding receiver and will be diagonal entries of $R_{nn}$, which need not be diagonal by itself. (Worst-case noise applies generally without sub-user decomposition, and indeed can help identify primary and secondary sub-users.)

**Theorem 2.8.1 (Worst-Case Noise)** For a given linear AWGN with channel matrix $H$ and noise energies specified by (2.418), the mutual information $\mathcal{I}(x;y)$ has minimum when $R_{nn}$ satisfies

$$R_{nn}^{-1} - [H \cdot R_{xx} \cdot H^* + R_{nn}]^{-1} = S_{wcn} \quad (2.419)$$

where $S_{wcn}$ is a $L_y \times L_y$ block-diagonal matrix. This worst case noise has abbreviated name $R_{wcn}$. Further, $\max_{R_{nn}} S_{wcn} = U^o$ and any secondary users BC-output indices correspond to the zeroed rows of $R_{wcn}^{-1} \cdot H$ or equivalently to the zeroed rows of $S_{wcn}$.

**Proof:** The mutual information needs to be differentiated with respect to the $L_y \times L_y$ elements of $R_{nn}$ where diagonal sub-blocks have side constraints. The mutual information plus side constraints have the Lagrangian to minimize with $\sum_{u=1}^{U} L_{y,u}^2$ Lagrange multipliers $S_{wcn,i,j}(u)$ where $i = 1,\ldots,L_y,u$ and $j = 1,\ldots,L_y,u$ and for each $u$:

$$\min_{R_{nn}} \mathcal{L}(R_{nn}, \{S_{wcn,i,j}(u)\}) \quad (2.420)$$

where, in bits per complex subsymbol, for all $(i,j) \in \{1,\ldots,L_y,u\} \otimes \{1,\ldots,L_y,u\}$

$$\mathcal{L}(R_{nn}, \{\lambda_u\}) = \log_2 |H \cdot R_{xx} \cdot H^* + R_{nn}| \cdot \frac{1}{x(y)} - \log_2 |R_{nn}|$$

$$+ \sum_{u=1}^{U} S_{wcn,i,j}(u) \cdot |R_{nn,i,j}(u) - R_{wcn,i,j}(u)| \quad (2.421)$$

The chain-rule of differentiation can be applied by first taking the derivative of $\mathcal{L}$ with respect to $|R_{nn}|$, and then multiplying by the derivative of $|R_{nn}|$ with respect to each and every element of $|R_{nn}|$.

$$\frac{d\mathcal{I}(x;y)}{d|R_{nn}|} = |H \cdot R_{xx} \cdot H^* + R_{nn}|^{-1} - |R_{nn}|^{-1} \quad (2.422)$$

The derivative of the $|R_{nn}|$ with respect to each of its elements $R_{nn}(i,j)$ is the determinant of the co-factor matrix $R_{nn}^C_{(i,j)}$ of that element. The co-factor is the submatrix formed by deleting row $i$ and column $j$ from $R_{nn}$ multiplied by $(-1)^{i+j}$. Thus

$$\frac{d\mathcal{I}(x;y)}{d|R_{nn}|} \cdot \frac{d|R_{nn}|}{dR_{nn}(i,j)} \quad (2.423)$$

---

74 Each diagonal element is an $L_y \times L_y$ positive-definite submatrix - this development can proceed identically even if each such block element has different dimension, so then $L_y \times L_y$ that will correspond to different BC channel user receivers having different numbers of dimensions (think antennas), which certainly can occur in practice.

75 BC channel indexing reverses order top/left to bottom/right on highest index.
can be grouped as the matrix of co-factors of $R_{nn}(i,j)$ divided by the determinant. Further, because the matrix is Hermitian, then this derivatice-chain-rule product is the matrix’ inverse. The same applies to $[H \cdot R_{xx} \cdot H^* + R_{nn}]$’s derivative. For the remaining side-constraint terms in $L$, these are only active for (block) diagonal elements where the derivative is $S_{wcn,i,j}(u)$, and essentially then 0 for off diagonal terms. Thus, the worst-case noise satisfies

$$R_{wcn}^{-1} - [H \cdot R_{xx} \cdot H^* + R_{wcn}]^{-1} = S_{wcn}$$

(2.424)

Solution finds the Lagrange multipliers $S_{wcn}$ through the side constraints. Since the mutual information and the constraint are convex, the solution is a unique minimum (as long as $R_{nn}$ is nonsingular).

Since the rate-sum is a special minimum mutual information with worst-case noise, which is associated with a particular input $R_{xx}$. The non-zero Lagrange multipliers ($S_{wcn}$’s nonzero diagonal entries) represent the different BC noise dimension’s infinitesimal change affect on the rate sum. The zero diagonal entries reflect noises that cannot affect the worst-case noise for the given $R_{xx}$. With an energy-sum-only constraint that defines primary and secondary users, as $R_{xx}$ varies it will certainly include situations where all primary users are energized and thus these primary users’ sum data rate would be affected by noises on the corresponding $S_{wcn}$ non-zero diagonal entries, but not those on the zeroed entries. Thus $\max R_{xx} S_{wcn} = U^\circ$. While the mutual information is minimized over noise autocorrelations, it remains a maximum rate for the given input autocorrelation. Since it is possible for any $R_{xx}$ to have energy from secondary users generally (on the BC because it is a single input), then the noise-whitening plus its matched filter is $R_{wcn} \cdot H$ and any energy from a secondary user would need to have been zeroed or its data rate at its receiver would be non-zero.

Alternatively, the $S_{wcn}$ are Lagrange multipliers and represent the infinitesimal change of the minimum mutual information with respect to the BC output noises. That mutual information is also a maximum rate sum for the given $R_{xx}$. If a specific diagonal element of $S_{wcn}(u,u) = 0$, then this noise does not affect this maximum mutual information if it changes. Correspondingly, this noise’s will influence the rate sum if the corresponding user for that output transmits nonzero energy. Thus, the corresponding user must be secondary and the worst-case noise presumes it is not present in the $R_{xx}$ that is instead entirely determined by the primary users. (Correspondingly, the primary users have nonzero diagonal elements of $S_{wcn}$. QED.

The worst-case noise autocorrelation $R_{wcn}$ is not necessarily (block) diagonal, but $\text{Diag}_{\text{block}} \{R_{wcn}\} = \text{Diag}_{\text{block}} \{R_{nn}\}$. Worst-case noise has useful implications for the Gaussian BC. Indeed, worst-case noise renders coordination among user receivers useless, as later in Subsection 2.8.3.3 Thus any optimum detector that experiences worst-case noise consequently would have independent optimum receivers. The Gaussian BC then basically can perform no better than if the noise were worst case. A matlab program written by former student Dr. Mehdi Molseni appears here and computes the worst-case noise through an iterative optimization algorithm for any supplied noise-whitened $R_{nn}^{-1/2} \cdot H$ for whatever the real noise $R_{nn}$ might be and given input autocorrelation matrix $R_{xx}$.

In general, there are 3 cases that may arise:

### 3 Primary/Secondary cases:

**Perfect MIMO:** $U^\circ = U = U'$. This case is non-degraded since $U = U' = \varrho_H$; there are no secondary users (or sub-users). Necessarily $\varnothing_{z,u} = 1 \forall u = 1,\ldots, U = U'$. This case has each user with 1 dimension (think antenna) at the transmitter and correspondingly one dimension at the receiver. Perfect MIMO users each have a path largely (MMSE sense) free of other users’ crosstalk, as becomes evident shortly. Essentially, all the (sub-) users get their own dimension.
Degraded (NOMA): $U^o < U = U'$. In this case $\varrho_H = \varrho_x = U^o$ and $\varrho_{x,u} = 1 \forall u = 1, \ldots, U^o = U$ and secondary users have no dimensions (antennas) to themselves, $\varrho_{x,u} = 0 \forall u = U^o+1, \ldots, U'$. In this case, energy-sharing (or dimension-sharing) of sub-users is necessary if the sub-users transmit non-zero information.

Enlarged MIMO: $U^o > U$. This case corresponds to at least one individual user’s receiver having $L_{y,u} > 1$; there are multiple receiver dimensions (antennas) per user. There are two sub-cases:

1. $U < U^o < U'$ (degraded enlarged MIMO). In this case some sub-users may share dimensions to receivers, and there are secondary sub-users.

2. $U < U^o = U'$ (non-degraded enlarged MIMO). In this case, each sub-user has at least one dimension that is largely free (MMSE sense) of crosstalk from other sub-users.

When $U^o >> U$, this enlarged MIMO often has the name “Massive MIMO.”

Mohseni’s Worst-Case Noise Program  This (revised by this text’s author) program can initialize the dual_gap (which is the duality gap) to 1e-6, as in the original wcnoise, while similarly nerr is Newton’s method acceptable error, set to 1e-4 in the original wcnoise. The author has found relaxation of either or both can allow convergence in situations where the iterative algorithm within may otherwise have difficulty converging. This program’s $b_{max}$ is for a real channel and thus needs to be doubled if the channel is complex baseband.

% the channel matrix is K*N by m, m is the number of transmitter’s antennas,  
% K is the number of users and N is the number od receiver’s antennas for  
% each users  
% Rx is the m by m input covariance matrix  
% Rz is the K*N by K*N worst case noise minimizing  
% 0.5 * log(det(H*Rx*H'+Rz)/det(Rz)), Rz has its N by N diagonal sub-blocks  
% equal to identity matrix  
% dual_gap is the duality gap, set to 1e-6 in wcnoise  
% nerr is Newton’s method acceptable error, set to 1e-4 in wcnoise  

function [Rz, sumRate] = wcnoise(Rx, H, N, dual_gap, nerr)  
% Adjusted by J. Cioffi in 2020 to allow dual_gap and nerr to be inputs  
[n,m] = size(H);  
K = n / N;

A = zeros(n,n,n*(n+1)/2);  
for i = 1:n  
    A(i,i,i) = 1;  
end

count = n+1;  
for i = 2:n  
    for j = 1:i - 1  
        A(i,j,count) = 1;  
        A(j,i,count) = 1;  
        count = count+1;  
    end
end
map = zeros(n,n);
for i = 1:K
    map((i-1) * N + 1:i * N,(i-1) * N + 1:i * N) = ones(N,N);
end

NT_max_it = 1000; % Maximum number of Newton's method iterations
%dual_gap = 1e-6; % step size for t
mu = 10; % back tracking line search parameters
alpha = 0.001;
beta = 0.5;

count = 1;
%nerr = 1e-4; % acceptable error for inner loop Newton's method

v_0 = zeros(n*(n+1)/2,1); % Strictly feasible point;
v_0(1:n) = 0.5 * ones(n,1);
v = v_0;
t = 1;
l_v = 1; % lambda(v) for newton's method termination

while (1+n)/t > dual_gap
    t = t * mu;
l_v = 1;
count = 1;
while l_v > nerr & count < NT_max_it
    f_val = 0; % calculating function value
    Rz = zeros(n,n);
    Rzprime = zeros(n,n);
    for i = 1:n*(n+1)/2 % computing Rz
        Rz = Rz + v(i) * A(:,:,i);
    end
    for i = 1:K
        Rzprime((i-1) * N + 1:i * N,(i-1) * N + 1:i * N) = Rz((i-1) * N + 1:i * N,(i-1) * N + 1:i * N);
    end
    f_val = t * log(det(H * Rx * H' + Rz)) - (t + 1) * log(det(Rz)) - log(det(eye(n) - Rzprime));
    S = inv(H * Rx * H' + Rz);
    Q = inv(eye(n) - Rzprime);
    Rz_inv = inv(Rz);
    g = zeros(n*(n+1)/2,1);
    h = zeros(n*(n+1)/2,n*(n+1)/2);
for i = 1:n*(n+1)/2
    g(i) = t * trace(A(:,:,i) * S) - (t + 1) * trace(A(:,:,i) * Rz_inv)...
    + (sum(sum(A(:,:,i) .* map)) ~= 0) * trace(A(:,:,i) * Q); % gradient
end

for i = 1:n*(n+1)/2
    for j = 1:n*(n+1)/2
        h(i,j) = -t * trace(A(:,:,i) * S * A(:,:,j) * S) + (t + 1) * trace(A(:,:,i) * ...
        Rz_inv * A(:,:,j) * Rz_inv)...+
        +(sum(sum(A(:,:,i) .* map)) ~= 0) * (sum(sum(A(:,:,j) .* map)) ~= 0) * ...
        trace(A(:,:,i) * Q * A(:,:,j) * Q); % hessian
    end
end

dv = -h\g; % search direction

s = 1; % checking v = v+s*dx feasible
% and also back tracking algorithm

v_new = v + s * dv;
f_new = 0;
Rz_new = zeros(n,n);

for i = 1:n*(n+1)/2
    Rz_new = Rz_new + v_new(i) * A(:,:,i);
end

for i = 1:K
    Rzprime((i-1) * N + 1:i * N,(i-1) * N + 1:i * N) = Rz_new((i-1) * N + 1:i * N,...
    N,(i-1) * N + 1:i * N);
end

f_new = t * log(det(H * Rx * H' + Rz_new)) - (t + 1) * log(det(Rz_new)) - ...
log(det(eye(n) - Rzprime));

feas_check = 1;
if real(eig(Rz_new)) > zeros(n,1)
    feas_check = 1;
else
    feas_check = 0;
end

if real(eig(eye(n) - Rzprime)) > zeros(n,1)
    feas_check = 1;
else
    feas_check = 0;
end

331
end

feas_check = feas_check * (f_new < f_val + alpha * s * g' * dv);

while feas_check ~= 1
    s = s * beta;
    v_new = v + s * dv;
    f_new = 0;
    Rz_new = zeros(n,n);
    for i = 1:n*(n+1)/2
        Rz_new = Rz_new + v_new(i) * A(:,:,i);
    end
    for i = 1:K
        Rzprime((i-1) * N + 1:i * N, (i-1) * N + 1:i * N) = Rz_new((i-1) * N + 1:i * N, (i-1) * N + 1:i * N);
    end
    f_new = t * log(det(H * Rx * H' + Rz_new)) - (t + 1) * log(det(Rz_new)) - ...
    log(det(eye(n) - Rzprime));
    feas_check = 1;
    if real(eig(Rz_new)) > zeros(n,1)
        feas_check = 1;
    else
        feas_check = 0;
    end
    if real(eig(eye(n) - Rzprime)) > zeros(n,1)
        feas_check = 1;
    else
        feas_check = 0;
    end
    feas_check = feas_check * (f_new < f_val + alpha * s * g' * dv);
end

v = v + s * dv; % update v
l_v = -g'*dv; % lambda(v)^2 for Newton’s method
count = count + 1; % number of Newtons method iterations
end

end

Rz = zeros(n,n);
for i = 1:n*(n+1)/2
    Rz = Rz + v(i) * A(:,:,i);
end

for i = 1:K
    Rz((i-1) * N + 1:i * N,(i-1) * N + 1:i * N) = eye(N);
end

sumRate = 0.5 * log2(det(H * Rx * H' + Rz)/det(Rz));

Return to the earlier simple Example 2.8.1 illustrates worst-case noise:

**EXAMPLE 2.8.2 (Simple Broadcast Channel Continued with Worst-Case Noise)**

Example 2.8.1 has $h_1 = .8$ and $h_2 = .5$ and $\sigma^2 = .0001$. Again, $L_x = L_y = 1$. Figure 2.53 shows the corresponding rate region. For this channel $R_{xx} = 1$. The third argument in the wcnoise program is 1 here. This channel has a worst-case noise and the steps to this BC’s characterization are:

```matlab
>> H=[80
50] =
80
50
(noise-whitened/normalized channel)

>> [Rwcn,b]=wcnoiseplus(1,H,1,1e-6,1e-4)
Rwcn =
    1.0000    0.6250
    0.6250    1.0000

b = 6.3220.
```

This example shows that $R_{wcn}$ is not diagonal, but its diagonal has the correct values. Continuing:

```matlab
>> Htilde=inv(Rwcn)*H =
    80.0000
    0.0000

>> Swcn=inv(Rwcn)-inv(H*H' + Rwcn). =
    0.9998    0.0000
    0.0000    0.0000

>> Ryy=H*H'*Rwcn = 1.0e+03 *
    6.4010    4.0006
    4.0006    2.5010

>> SNRp1=det(Ryy)/det(Rwcn) = 6.4010e+03
>> 0.5*log2(SNRp1) = 6.3220 (checks!)
```

The program’s rate-sum output $b = 6.3220$ is the same rate sum that occurs for $R_{xx} = 1$ from previous work on this same example. Note that the program, nor the worst-case noise concept, specify to which users the energy maps. From previous examination, this maximum occurred when $E_2 = 0$. Correspondingly, the receiver matched noise-whitening filter zeros user 2. This receiver filtering is thus another way to identify secondary users (the ones that are multiplied by zeroed rows of $R_{wcn}^{-1} H$).

Receiver 1 receives total energy 6401, and since it can decode both users, it essentially obtains the maximum rate sum. Nonetheless, receiver 2 can detect any user 2’s component reliably as can user 1, but energy on user 2 will reduce the sum rate and user 1’s data rate, so would be below 6.322 as per previous invocations of this example. Worst-case noise simply implies correctly that receiver 2 is not necessary for the best sum rate $I_{wcr}(x; y)$, and indeed user 2 should carry no information nor energy to obtain the largest rate sum on this example scalar BC, which also happens to be the matrix AWGN channel’s minimum-over-noise mutual information.

333
2.8.3.3 Precoder Generalization

This subsection generalizes the precoder (and corresponding decoder)’s architecture. The precoder will use a square root to take a element-normalized vector of inputs \( \mathbf{v} \) at the output of the precoder and transform it into the correct \( R_{XX} \). Square roots are not unique, and indeed this one can be non-square or singular, but will satisfy

\[
R_{XX} = A \cdot A^* ,
\]

where is a special square root that depends on both \( R_{XX} \) and \( R_{VV} \), or equivalently on \( R_{XX} \) and the BC \( H \). The determination of this special \( A \) is part of the design process to come. For now, it is sufficient to know that it always exists.

**Precoders for the Vector BC**: Figure 2.55 generalizes the earlier Figure 2.51 to the vector BC case. The primary and secondary user precoders\(^7\) both appear as generating identity autocorrelation matrices with respective fractions of primary- and secondary-user energy, so \( R_{VV} = I \), while \( R_{VV}^o = \alpha \cdot I \) and \( R_{VV}^s = (1 - \alpha) \cdot I \).

![Figure 2.55: Vector Gaussian BC Precoders.](image)

Both precoders process inputs recursively with decreasing index from \( u = U^o + 1 \) in the secondary precoder \( G^{-s} \) and from \( U^o \) to 1 in the primary precoder. Figure 2.55’s transmitter’s square-root matrix then converts \( R_{VV} = I \) to \( R_{XX} \) on the channel input, per requirement. Individual primary sub-user energy components are for primary (sub-) users \( \mathcal{E}_u \) \( \forall u \in U^o \) and

\[
1 = \mathcal{E}_u + \sum_{i=U^o+1}^{U^o} \mathcal{E}_{u,i} \forall u = 1, ..., U^o ,
\]

where \( \mathcal{E}_{u,i} \) is the energy from secondary users \( i = U^o + 1, ..., U^o \) that ads to primary users’ dimensions. In the scalar Gaussian BC case, this was a single parameter \( \alpha_u \) where \( \sum_{u=1}^{U^o} \alpha_u = 1 \), but there is a possibility of distributing the energy from any secondary user into any combination of primary users that will not lead to crosstalk but will reduce the primary users’ available energy according to the choice of \( \mathcal{E}_{u,i} \). This simply becomes a multiplier of the primary user’s \( SNR \rightarrow 1 + \mathcal{E}_u \cdot (S_{0,u} - 1) \) being reduced

\(^7\) Again, determined with noise-whitening but only pure energy constraint for any given set of (block) diagonal elements \( \{R_{nn}(u)\} \).

334
by the corresponding energy reduction. MMSE processing continues to target energy of one unit, but some of that energy will be removed by the receiver modulo processing for each primary user’s receiver. Correspondingly, the secondary data rates will also be affected by the $E_{u,i}$ distribution. Essentially each primary user’s energy becomes noise for every secondary user (and indeed some secondary users for one another also). The set of possibilities’ union completes the capacity region. The search simply increments $E_{u,i}$ by some $\Delta E$ from zero on each user in a nested loop of energy attempts such that (2.426) remains satisfied. Each BC secondary user’s data rate will compute directly from its $H_u$ entry with all lower-index users as noise.

The secondary precoder generalizes user 2 from the simpler BC scalar case, as Figure 2.55 also shows. Starting always with (sub-) user $U'$ that treats all other users as noise and generates $\nu'_{u}$, user $U'-1$ then precodes user $U'$ by subtracting it prior to a modulo for $(\cdot)E_{u,-1}$ and then adding it afterward (see Figure 2.55’s dashed (black) line to the square-root matrix-filter input) to dimension/sub-user $U'-1$ to get $\nu_{U'-1}$. This pre-subtract and add process continues for any set of primary energies $\{E_u\}$ and secondary user energies $\{E_{u,i}\}$ that satisfy (2.426). For each primary-user precoder output, these $U$ secondary components all add with their respective energy fractions to each primary (sub-) user’s dimension as they successively stack into the set of $\nu$’s dimensions that then proceed (after precoding has processed all users) to generate $x$ with proper autocorrelation, $R_{xx}$ for the BC input, through $A$.

The primary precoder continues this similar process for $u = U', \ldots, 1$, but with non-unity precoder coefficients $G_{u,i}$ that may not have unit values for $i \leq U'$ and do not add again to each subsequent primary-user dimension at the precoder output. Figure 2.55 also shows the primary precoder’s action for each primary-user $u'$ where the action of the square-root matrix and the noise-whitened users channels ensemble will re-add the other primary users crosstalk in the right gain and phase to cancel those primary-user components added in the precoder. $G'$ is Subsection 2.8.3’s $G$ to come.

The primary-group members’ rate sum $b_o$ will be the same for all orders within the primary group if the secondary group has zero (or equivalently the BC is non-degraded). Order within the primary (sub-) user group affects each primary (sub-) user’s individual rate $\{b_u \mid u \in u^o\}$, but the sum $b^o = \sum_{u=1}^{U^o} b_u$ remains constant for the given $R_{xx}$. The secondary (sub-) user group’s individual rates $\{b_u \mid u \in u^s\}$ and rate sum $b^s = \sum_{u=U^o+1}^{U^s} b_u$ both vary with secondary group energy apportionment from the energy-possibility set

$$\{E_x^s\} = \left\{ \text{trace} \{R_{xx}(s,u)\} \mid u \in u^s \land \left[R_{xx}^s = \sum_{u=1}^{U^s} R_{xx}(s,u) \right] \land \left[R_{xx} = R_{xx}^s + R_{xx}^o \right] \right\}.$$  (2.427)

There is one overall order as per (??), in terms of best rate sum for any input energy assignment (i.e., the $\theta$’s and $\alpha$ choices)
### 2.8.3.4 Worst-Case Noise and Diagonal Receivers’ Optimality

This subsection shows that worst-case noise has diagonal primary users’ optimum receiver processing with a transmit lossless precoder. Diagonal receiver processing thus applies to the BC. First, a reminder is that the vector precoder recursively computes outputs through the recursion

\[
v_u = \left( v_u - \sum_{i>u}^{U^o} g_{u,i} \cdot v_i - \sum_{j=U^o+1}^{U'} v_j \right) \varepsilon_u , \quad \forall \ u = U', ..., 1 ,
\]

with \( v_u = \nu_U \). The resultant \( U^o \times 1 \) vector \( v = v^o + v^s \) is input to the \( R_{xx}^{1/2} \) matrix filter. The vector \( v^s \) is simply a \( U^o \)-times repeat of the secondary-user precoder output for each dimension (its variation occurring through variation of the energy factors \( \theta_u \) within the secondary group and the factor \( (1 - \alpha) \) relative to the primary group energy. User \( u \)'s receiver modulo operation reproduces (after all receiver processing at each output) the quantity \( \hat{v}_u \).

The design process first deletes rows/columns corresponding to \( S_{wc}(u, \ell_0) \) secondary (sub-) users from (2.424). These secondary users instead are handled by the secondary precoders in Figures 2.55 and 2.56. Substitution shortly continues into the remaining \( U^o \times U^o \) worst case-noise Equation (2.424) using the \( L_x \times U^o \) A matrix square-root such that \( R_{xx} = A \cdot A^* \), with

\[
A = Q \cdot \Phi .
\]

The precoder retains the statistical independence of Figure 2.56’s dimensional outputs \( v_u^o, \ u = 1, ..., U^o \) (since the lossless-precoder output’s \( R_{uu} = I \)). Then (2.424) becomes

\[
R_{wc}^{-1} - [H \cdot A \cdot A^* \cdot H^* + R_{wc}]^{-1} = S_{wc}.
\]
There are $U$ block-diagonal (positive definite) entries in $S_{\text{wcn}}(u)$ in $S_{\text{wcn}}$, each of which is now nonsingular and the $U^o \times U^o$ $S_{\text{wcn}}$ has an (all-real-positive-eigenvalue) eigen-decomposition
\[
Q^*_\text{wcn} \cdot S^\prime_{\text{wcn}} \cdot Q_{\text{wcn}} = (2.431)
\]
where $S^\prime_{\text{wcn}}$ is positive-definite diagonal. Indeed, $Q_{\text{wcn}}$ is not only unitary ($Q^*_{\text{wcn}} Q_{\text{wcn}} = I$, but also block diagonal with each block also unitary so that $Q^*_\text{wcn}(u) \cdot Q_{\text{wcn}}(u) = I$.

The $L_x \times U^o$ matrix $A$, and the $L_x \times L_x$ autocorrelation matrix $R_{xx}$ have the same rank $R_A = R_x = U^o \leq L_x$. As per Figure 2.56, all secondary users’ energies add to the primary users’ dimensions, but retain $R_x$ as the transmission rank. These inverted matrices in (2.430) are nonsingular, allowing all following matrix inversions to exist. The Matrix Inversion Lemma’s\textsuperscript{77} use on (2.430)’s second term (on the left), and flipping equality sides, leads to
\[
S_{\text{wcn}} = R_{\text{wcn}}^{-1} - \left[ R_{\text{wcn}}^{-1} - R_{\text{wcn}}^{-1} \cdot H \cdot A (I + A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} \cdot H \cdot A)^{-1} A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} \right]
\]

\[
Q^*_{\text{wcn}} \cdot S^\prime_{\text{wcn}} \cdot Q_{\text{wcn}} = R_{\text{wcn}}^{-1} \cdot H \cdot A (I + A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} \cdot H \cdot A)^{-1} A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} (2.432)
\]

\[
S^\prime_{\text{wcn}} = Q_{\text{wcn}} \cdot R_{\text{wcn}}^{-1} \cdot H \cdot A \cdot R_0 \cdot A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} \cdot Q^*_{\text{wcn}} (2.433)
\]

where $G$ is a $U^o \times U^o$ upper triangular monic matrix and $S_0$ is a $U^o \times U^o$ positive-definite diagonal matrix, and $R_0^{-1} = G \cdot S_0 \cdot G^*$ is therefore a $U^o \times U^o$ positive definite matrix with Cholesky\textsuperscript{78} factor $G$. Section 2.7 found $R_0$ is the MMSE matrix for the backward-channel estimation problem of estimating $\nu$ from $y$, but this present worst-case-noise development does not need that interpretation. Continuing (2.433)

\[
S^\prime_{\text{wcn}} = Q_{\text{wcn}} \cdot R_{\text{wcn}}^{-1} \cdot H \cdot A \cdot G^{-1} \cdot S_0^{-1} \cdot G^{-*} \cdot A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} \cdot Q^*_{\text{wcn}} (2.434)
\]

Again, all inverted matrices above are nonsingular if (as is always the case in practice) the noise is nonsingular. The specific $R_{xx}^{1/2}$ choice of the matrix $A$ remains ambiguous at this point, and thus so do consequently $G$ and $S_0$.

**Two steps to the MMSE BC equivalent:** Determination of $G$ and thus $A$ follows two steps that “triangularize” individually the worst-case-noise-equivalent channel and the input autocorrelation matrix. The second step essentially finds $A$, but depends on completion of the first step.

**The channel triangular component** The $L_y \times L_x$ matrix $R_{\text{wcn}}^{-1} \cdot H$ has generalized “QR” factorization\textsuperscript{79}

\[
Q_{\text{wcn}} \cdot R_{\text{wcn}}^{-1} \cdot H = \begin{bmatrix} 0 & R_{\text{wcn}}^{-1} \cdot H \cdot A (I + A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} \cdot H \cdot A)^{-1} A^* \cdot H^* \cdot R_{\text{wcn}}^{-1} \\ 0 & 0 \end{bmatrix} = R \cdot Q^* (2.435)
\]

**The input triangular component** The $U^o \times U^o$ square nonsingular upper-triangular matrix $\Phi$ satisfies the Cholesky square-root factorization:

\[
\Phi \cdot \Phi^* = Q^* \cdot R_{xx} \cdot Q (2.436)
\]

Then, the desired specific square root is

\[
R_{xx}^{1/2} = A = Q \cdot \Phi (2.437)
\]

which now can unambiguously and uniquely be computed from $Q$ and $\Phi$ that are also now known. When $U^o < L_x$, then $x$ Hilbert-space ($R_{xx}$) components in the channel null space\textsuperscript{80} do not pass to the channel output. Thus, strictly $A \cdot A^* \neq R_{xx}$ only when $U^o < L_x$ occurs. This is ok, and $A \cdot A^*$ represents the only information content of interest.

\textsuperscript{77}The matrix inversion lemma is

\[
[A + BCD]^{-1} = A^{-1} - A^{-1} B [C^{-1} + DA^{-1} B]^{-1} DA^{-1} .
\]

\textsuperscript{78}See Matlab’s “chol” command, or for more general block Cholesky see the Appendix.

\textsuperscript{79}see matlab qr command.

\textsuperscript{80}The null space will be all vectors $\nu$ such that $Q_{\text{wcn}} \cdot R_{\text{wcn}}^{-1} \cdot H \cdot \nu = 0$. 

337
Finding The Overall MMSE BC Triangular Channel: A \( U^o \times U^o \) diagonal matrix \( D_A \) has definition
\[
D_A \triangleq \text{Diag}\{R \cdot \Phi\}.
\] (2.438)
The upper triangular matrix \( G \) and the diagonal factor \( S_0 \) respectively can satisfy:
\[
G = D_A^{-1} \cdot R \cdot \Phi
\] (2.439)
\[
S_0 = D_A \cdot (S')^{-1}_{wcn} \cdot D_A
\] (2.440)
which checks through
\[
G^{-1} \cdot S_0^{-1} \cdot G^{-*} = (\Phi^{-1} \cdot R^{-1} \cdot D_A) \cdot (D_A^{-1} \cdot S_{wcn} \cdot D_A^{-1}) \cdot (D_A \cdot R^{-*} \cdot G^{-*})
\] (2.441)
\[
= \Phi^{-1} \cdot R^{-1} \cdot S_{wcn} \cdot R^{-*} \cdot \Phi^{-*}
\] (2.442)
\[
= \Phi^{-1} \cdot R^{-1} \cdot [Q_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot A] \cdot R_b \cdot A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot Q_{wcn}^* \cdot R^{-*} \cdot \Phi^{-*}
\] (2.443)
\[
= \Phi^{-1} \cdot R^{-1} \cdot R \cdot Q^* \cdot Q \cdot \Phi \cdot R_b \cdot \Phi^* \cdot R^* \cdot R^{-*} \cdot A \cdot Q^* \cdot \Phi^{-*}
\] (2.444)
\[
= R_b.
\] (2.444)
The relations
\[
Q_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot A = \Phi \cdot R = D_A \cdot G
\] (2.445)
also follow immediately.

The MMSE User Channels Are Canonical: Also of interest is the worst-case-noise mutual information
\[
\mathcal{I}_{wcn}(x;y) = \frac{|H \cdot R_{xw} \cdot H^* + R_{wcn}|}{|R_{wcn}|}
\] (2.446)
\[
= \frac{|R_{wcn}^{-1/2} \cdot H \cdot R_{xw} \cdot H^* \cdot R_{wcn}^{-*}/2 + I|}{|R_{wcn}|}
\] (2.447)
\[
= \frac{|R_{wcn}^{-1/2} \cdot H \cdot A \cdot R_b \cdot R_{wcn}^{-*}/2 + I|}{|R_{wcn}|}
\] (2.448)
\[
= \frac{|A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot H \cdot A + I|}{|R_{wcn}^{-1}|}
\] (2.449)
follows from SVD of \( R_{wcn}^{-1/2} \cdot H \cdot A \)
\[
= |R_{wcn}^{-1}|
\] (2.450)
\[
= |S_0|
\] (2.451)
\[
\mathcal{I}_{wcn}(x;y) = \log_2 (|S_0|) \text{ bits/complex subsymbol.}
\] (2.452)
Equation (2.449) also relates that a valid set of individual channel SNR’s are given through single-user vector-coding’s singular value decomposition (SVD) of the channel path \( R_{wcn}^{-1/2} \cdot H \cdot A \) to be the squared singular values plus 1, so vector-coded SNR’s would be \( SNR_{vc,wcn}(u) = \frac{\lambda^2_{vc,wcn,u-1}}{\sigma^2} \) and thus
\[
|S_0| = \prod_{u=1}^{U^o} S_{b,u} = \prod_{u=1}^{U^o} (SNR_{vc,wcn}(u) + 1),
\] (2.453)
confirming that
\[
\mathcal{I}_{wcn}(x;y) = \sum_{u=1}^{U^o} \log_2 (1 + SNR_{vc,wcn}(u)) \text{ bits/complex subsymbol,}
\] (2.454)
recalling that Equations (2.452) and (2.454) can be divided by 2 to be interpreted in terms of sums over real dimensions.

338
Cannonical User Channel Set Has Diagonal Receiver: An optimum single-user receiver that sees all users’ components of $y$ would begin with the 1-to-1 mapping cascade with noise-whitening matched-matrix filter, recalling also Lemma 2.3.5,

$$W = \left( S^{-1}_0 \cdot G^{-\ast} \cdot \left( A^* \cdot H^* \cdot R^{-1}_{\text{wcn}} \right) \cdot Q^*_{\text{wcn}} \cdot Q_{\text{wcn}} \right)$$

where $\Phi^* \cdot Q^* \cdot Q \cdot R^* \cdot Q_{\text{wcn}}$

$$W_{\text{noise-white-match}}$$

$$W = S^{-1}_0 \cdot G^{-\ast} \cdot \Phi^* \cdot Q^* \cdot Q \cdot R^* \cdot Q_{\text{wcn}}$$

$$W = S^{-1}_0 \cdot G^{-\ast} \cdot G^{-1} \cdot D_A \cdot Q_{\text{wcn}}$$

which is (block, because of the $Q_{\text{wcn}}$ block diagonal unitary matrix) diagonal and so needs no coordination among the $U^o$ primary users. This also follows directly from Equation (2.445), but without the intermediate decomposition into a noise-whitening-match and triangular components. Thus, $W$ can be used on the Gaussian BC primary users’ dimensions. ML detectors then follow on each user at the corresponding receiver. The $Q_{\text{wcn}}$ matrix allows local noise correlation at a single user’s receiver to be exploited, within the overall user-to-user worst-case noise structure.

Further, the data path from input innovations/data vector $\nu$ to the detector input (with the precoder equivalent to $G^{-1}$ if the coefficients of the monic upper-triangular matrix $G$ characterize the primary-user precoder) is

$$W \cdot (y - n) = (S^{-1}_0 \cdot D_A \cdot Q_{\text{wcn}}) \cdot (H \cdot A \cdot G^{-1}) \cdot \nu$$

$$= S^{-1}_0 \cdot D_A \cdot Q_{\text{wcn},u} \cdot R_{\text{wcn}} \cdot R^{-1}_{\text{wcn}} \cdot H \cdot Q \cdot \Phi \cdot G^{-1} \cdot \nu$$

$$= S^{-1}_0 \cdot D_A \cdot Q_{\text{wcn}} \cdot R_{\text{wcn}} \cdot R \cdot Q^* \cdot Q \cdot \Phi \cdot G^{-1} \cdot \nu$$

$$= S^{-1}_0 \cdot D_A \cdot \Phi \cdot G^{-1} \cdot \nu$$

$$= S^{-1}_0 \cdot D_A \cdot Q_{\text{wcn}} \cdot R_{\text{wcn}} \cdot R \cdot \Phi \cdot \Phi^{-1} \cdot R^{-1} \cdot D_A \nu$$

MMSE Bias Removal: The BC outputs of interest to the (uncoordinated BC primary users’) receivers are

$$z_u = \left( S^{-1}_{0,u} \cdot D_{A,u} \right) \cdot \sum_{i=1}^{U^o} Q_{\text{wcn}}(u) \cdot R_{\text{wcn}}(u, i) \cdot D_{A,i} \cdot \nu_i$$

Specifically, $z_u$ is not simply a scaled and rotated version of the correspondingly diagonal element of $R_{\text{wcn}}$, or $R_{nn}(u)$. This means, the transmitter preprocessing causes output signal energy from other inputs $\nu_{i\neq u}$ to contribute to the output, or

$$E[z_u/\nu_u] \neq \nu_u$$

which is bias in the detector. This receiver has $I_{\text{wcn}}$ because the processing in Equation (2.460) is 1-to-1 (invertible over all complex vectors), which by Lemma 2.3.5 retains all information $I_{\text{wcn}}$ at this invertible transformation’s output. Further, this is also then in the Gaussian case a MMSE estimate with SNR product $|R_{xx}| / |R_{ee}|$ because

$$E[\nu y^\ast] \cdot E[y y^\ast] \cdot E[\nu^\ast y^\ast]$$

$$= E[\nu \nu^\ast] [A^* H^* [H \cdot A \cdot E[\nu \nu^\ast] \cdot A^* - H + \nu \nu^\ast]]^{-1}$$

$$= G \cdot A^* \cdot H^* \cdot [H \cdot A \cdot I \cdot A^* - H + R_{\text{wcn}}]^{-1}$$

$$= G \cdot R_{\text{wcn}} \cdot A^* \cdot Q \cdot R^* \cdot Q_{\text{wcn}}$$

$$= G \cdot G^{-1} \cdot S^{-1}_0 \cdot G^{-\ast} \cdot \Phi^* \cdot R^* \cdot Q_{\text{wcn}}$$

$$= S^{-1}_0 \cdot G^{-\ast} \cdot G^* \cdot D_A \cdot Q_{\text{wcn}}$$

$$= W$$
confirming that the diagonal BC receiver processing \( W \) is MMSE for the Gaussian BC with worst-case noise.

Equivalently, the signal power of interest (the component that arises from \( \nu_u \)) to the detector is slightly smaller than the full signal power of \( z_u \). This noise path to the same output (since \( W \) is diagonal) is

\[
S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot n,
\]

with noise autocorrelation

\[
R_{\nu'\nu} = S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot R_{wcn} \cdot Q_{wcn}^* \cdot D_A \cdot S_0^{-1}.
\]

(2.475)

Individual MMSE User Channel Set Is Canonical: The ratio of signal energy to noise energy on the diagonals is indeed \( S_{0,u} \), and this is a MMSE biased SNR, but an ML detector will maximize \( p_z/\nu \), which has the subsymbol bias (non-zero conditional mean) in Equation (2.467). Subsection 2.3.6 finds the exact relationship in situations with minimum mean-square-error to be with

\[
SNR_{bc,wcn,u} \triangleq S_{0,u} - 1,
\]

(2.477)

and then

\[
E[z_u/\nu_u] = (1 - \frac{1}{S_{0,u}}) \cdot \nu_u
\]

(2.478)

so the unbiased SNR is not \( S_{0,u} \) but \( SNR_{BC,wcn,u} = S_{0,u} - 1 \) when the subsymbol decision regions are correctly set to remove (2.467)’s bias. While

\[
| S_0 | = \prod_{u=1}^{U^o} S_{0,u} = \prod_{u=1}^{U^n} (SNR_{BC,wcn}(u) + 1) = \prod_{u=1}^{U^n} (SNR_{VC,wcn}(u) + 1),
\]

(2.479)

and \( I_{wcn}(x;y) \) remain the same, it is not necessarily true that the individual SNRs for single-user vector-coding and multi-user BC, even when both have the same worst-case noise, are equal. Thus, in general \( SNR_{vc,wcn,u} \neq SNR_{BC,wcn,u} \) but their products (+1) are equal as in (2.479). \( W \) is an MMSE estimator (see Appendix), which follows from (2.477) - (2.479).

The diagonalized-receiver’s primary users rate sum \( b^o \) is independent of the order with \( \{u^o\} \), but \( b = b^o + b^* \) reduces along with \( b^o = b_{u \leq U^o} \) as \( b^* = b_{u > U^o} \) increases. By contrast, MAC primary-user order can change the rate sum, but only because the individual users have energy constraints - with an MAC-energy-sum constraint, then primary users’ rate sum also becomes order-independent.

Returning again to Example 2.48:

**EXAMPLE 2.8.3 (Simple Broadcast Continued)** Example 2.48 earlier continued with worst-case noise. For this example, \( L_x = 1, L_y = 1 (l_x = 1) \), and \( U = 2 \). the channel has rank \( R = 1 \) and so is trivially degraded NOMA. There is one primary user (user 1) and one secondary user (user 2). In this case, the \( R_b \) matrix is a scalar, so \( G = 1 \), and thus \( S_0 = R_b \). \( S_0 \) can also be computed directly from the match-filtered channel output determinant, which is a simple scalar in this case. The following matlab commands continue the previous example.

\[
G=1;
\]

\[
S0=H'*inv(Rwcn)*H + 1
\]

\[
0.5*log2(S0)=6.3220
\]

\[
W=(1/S0)*H'*inv(Rwcn) =
0.0125 0.0000
\]

\[
Wunb=(1-1/S0)*W =
0.0125 0.0000
\]

The feed forward filter is diagonal (trivially a scalar when \( L_x = 1 \) on the primary users and of course then shows zero value for secondary user (which is user 2). This zeroing of secondary users simply means that the primary dimensions carry any nonzero user-2 energy
at the expense of reducing primary user 1’s energy $\mathcal{E}_1$, and consequently bits per dimension $b_1$. The data rate $b_2$ for user 2 is reliably decodable at user 1’s receiver, so worst-case noise analysis is not per-se multi-user, and simply states that the maximum (over energy allocations) rate sum is both users are reliably decodable at user 1’s receiver for the worst-case noise. Receiver 2 simply would treat any user 1 component as noise for the reliable detection, also there, of user 2.

A more interesting case is when $L_x = 2$ with 2 users in the following example:

**EXAMPLE 2.8.4 (Simple $2 \times 2$ non-degraded BC)** A perfect-MIMO BC has white Gaussian noise with $2 \times 2$ autocorrelation $R_{nn} = .0001 \cdot I$ and channel matrix

$$H = \begin{bmatrix} .8 & .7 \\ .5 & .6 \end{bmatrix},$$

(2.480)

with rank $\rho_H \approx 2 = U = U'$. The channel is non-degraded and both user 1 and user 2 are primary. There is no secondary user. Thus, $R_b$ is $2 \times 2$, as is $G$. $L_y = 1$, while $L_x = 2$ (and $l_x = 1$).

```matlab
>> H = [ 80 70 \\
60 50 ];
>> Rxx=[1 .8 \\
.8 1];
```

The energy is $\mathcal{E}_x = trace\{R_{xx}\} = 2$: for the BC $R_{xx}(1) = R_{xx}(2) = 1$ does not imply equal user treatment. There are many $R_{xx}$ choices for which $trace\{R_{xx}\} = 2$ that correspond to different rate sums and user rates. This differs from the $2 \times 2$ MAC earlier, for which there was only 1 rate sum in Example 2.7.1, when an energy-vector constraint applies.

```matlab
>> [Rwcn,b]=wcnoise(Rxx,H,1)
Rwcn =
    1.0000    0.0232
    0.0232    1.0000
b = 9.6430
>> Htilde=inv(Rwcn)*H =
    78.8817    68.6440
    48.1687    58.4064
>> Swcn = inv(Rwcn)-inv(H*Rxx*H'+Rwcn) =
    0.9835    0.0000
    0.0000    0.9688
>> J2=hanke1([0 1]);
>> [Q,R]=qr(J2*Htilde'*J2)
Q =
    -0.7715   -0.6363
    -0.6363   -0.7715
R =
    -75.7068  -103.1463
         0   17.1808
>> R=(J2*R*J2)' =
   17.1808  -103.1463
         0   -75.7068
>> Q=(J2*Q'*J2) =
    0.7715   -0.6363
    -0.6363   -0.7715
```

$R * Q' = 341$
>> Rxxrot=Q'*Rxx*Q =
0.2146  -0.1523
-0.1523  1.7854
>> Phi=(J2*chol(J2*Rxxrot*J2)*J2)' =
0.4490  -0.1140
0   1.3362
>> DA=diag(diag(R*Phi)) =
7.7149   0
0  -101.1578
>> G=inv(DA)*R*Phi =
1.0000 -18.1182
0        1.0000
>> A=Q*inv(R)*DA*G =
0.3464 -0.9381
-0.2857 -0.9583
>> S0=DA*inv(Swcn)*DA =
1.0e+04 *
0.0061  0.0000
0.0000  1.0562
>> Wunb=(inv(S0)-eye(2))*DA =
-7.5874   0.0000
-0.0000  101.1482
>> Gunb=eye(2)+S0*inv(S0-eye(2))*(G-eye(2)) =
1.0000 -18.4226
0        1.0000
>> b=0.5*log2(diag(S0))' =
2.9597   6.6833
>> sum(b) = 9.6430 (checks)

------------------- repeat with cross correlation -.8 ---------------

>> Rxx=[1 -.8
   -.8 1];
>> [Rwcn,b]=wcnoise(Rxx,H,1) =
   1.0000   0.0025
   0.0025   1.0000
b = 9.6116
>> Htilde=inv(Rwcn)*H =
  79.8731   69.8475
  49.7964   59.8220
>> Swcn = inv(Rwcn)-inv(H*Rxx*H'+Rwcn) =
   0.9979   0.0000
   0.0000   0.9962
>> [Q,R]=qr(J2*Htilde'*J2)
Q =
  -0.7686  -0.6398
  -0.6398   0.7686
R =
  -77.8354 -104.7828
   0    16.7020
>> Q=J2*Q*J2 =
   0.7686   -0.6398
\[
R = (J_2 * R * J_2)' =
\begin{bmatrix}
16.7020 & -104.7828 \\
0 & -77.8354
\end{bmatrix}
\]

\[
R * Q' =
\begin{bmatrix}
79.8731 & 69.8475 \\
49.7964 & 59.8220
\end{bmatrix}
\]

\[
R_{xxrot} = Q' * R_{xx} * Q =
\begin{bmatrix}
1.7867 & 0.1451 \\
0.1451 & 0.2133
\end{bmatrix}
\]

\[
\Phi = (J_2 * \text{chol}(J_2 * R_{xxrot} * J_2) * J_2)' =
\begin{bmatrix}
1.2992 & 0.3142 \\
0 & 0.4618
\end{bmatrix}
\]

\[
DA = \text{diag}(\text{diag}(R * \Phi)) =
\begin{bmatrix}
21.6997 & 0 \\
0 & -35.9455
\end{bmatrix}
\]

\[
G = \text{inv}(DA) * R * \Phi =
\begin{bmatrix}
1.0000 & -1.9881 \\
0 & 1.0000
\end{bmatrix}
\]

\[
A = Q * \text{inv}(R) * DA * G =
\begin{bmatrix}
0.9985 & -0.0539 \\
-0.8312 & -0.5560
\end{bmatrix}
\]

\[
S_0 = DA * \text{inv}(Swcn) * DA =
\begin{bmatrix}
1.0e+03 & * \\
0.4719 & 0.0000 \\
0.0000 & 1.2970
\end{bmatrix}
\]

\[
W_{unb} = \text{inv}(S_0 - \text{eye}(2)) * DA =
\begin{bmatrix}
0.0461 & 0.0000 \\
-0.0000 & -0.0277
\end{bmatrix}
\]

\[
G_{unb} = \text{eye}(2) + S_0 * \text{inv}(S_0 - \text{eye}(2)) * (G - \text{eye}(2)) =
\begin{bmatrix}
1.0000 & -1.9924 \\
0.0000 & 1.0000
\end{bmatrix}
\]

\[
b = 0.5 * \text{log2}(\text{diag}(S_0))'
\]

\[
b =
\begin{bmatrix}
4.4411 \\
5.1705
\end{bmatrix}
\]

\[
\text{sum}(b) = 9.6116
\]

\[
---------- \text{ with cross correlation .2 } \text{ ---------}
\]

\[
R_{xx} = [1.2 \\
.2 1] ;
\]

\[
[R_{wcn}, b] = \text{wcnoise}(R_{xx}, H, 1) =
\begin{bmatrix}
1.0000 & 0.0060 \\
0.0060 & 1.0000
\end{bmatrix}
\]

\[
b =
\begin{bmatrix}
10.3240
\end{bmatrix}
\]

\[
---------- \text{ identity input -----}
\]

\[
R_{xx} = \text{eye}(2) ;
\]

\[
[R_{wcn}, b] = \text{wcnoise}(R_{xx}, H, 1) =
\begin{bmatrix}
1.0000 & 0.0048 \\
0.0048 & 1.0000
\end{bmatrix}
\]

\[
b =
\begin{bmatrix}
10.3517
\end{bmatrix}
\]

\[
---------- \text{ may need to change dual gap to 1e-5 } \text{ ---------}
\]

\[
R_{xx} = [1.5 .8 \\
.8 .5] ;
\]

\[
[R_{wcn}, b] = \text{wcnoise}(R_{xx}, H, 1) =
\begin{bmatrix}
1.0000 & 0.0682 \\
0.0682 & 1.0000
\end{bmatrix}
\]

343
The diagonal receiver matrix is $W$ (this examples matlab code calls it WUNB). The $G$ matrix is nontrivial and defines the channel’s input precoder. There is no crosstalk between the two users, and each has full use of its single dimension. The sum data rate is as high as it can be for the BC and the given $R_{xx}$, which leads to the diagonal receiver for the situation of the noise having worst-case characteristics (and this is the maximum (over inputs) sum data rate for the situation of no receiver coordination and this $R_{xx}$). If there were a 3rd user, then that user would be secondary, the primary users’ dimensions would carry it with some $\bar{b}_1$ and/or $\bar{b}_2$ loss and sum-rate loss. The example also illustrates the variation of sum rate with $R_{xx}$ that Subsection 2.8.5 investigates further.

A third example with a $3 \times 3$ singular channel is more potent in illustrating primary and secondary users and the overall handling:

**EXAMPLE 2.8.5 (Singular $3 \times 3$ degraded BC)** A real-baseband degraded-NOMA BC
has white Gaussian noise with $2 \times 2$ autocorrelation $R_{nn} = .0001 \cdot I$ and channel matrix

$$H = \begin{bmatrix}
.8 & .6 & .4 \\
.6 & .45 & .3 \\
.2 & .2 & .2 
\end{bmatrix}, \quad (2.481)$$

with rank $\rho_H = 2 = U^o < U$. The channel is degraded. The input autocorrelation matrix is given as

$$R_{xx} = \begin{bmatrix}
3 & 0 & 0 \\
0 & 4 & 0 \\
0 & 0 & 2 
\end{bmatrix}. \quad (2.482)$$

The objective is a precoded design and the corresponding data rates. This begins by finding the secondary user:

```matlab
>> H=[80 60 40
 60 45 30
20 20 20];
>> rank(H) = 2
>> Rxx=diag([3 4 2]);
>> [Rwcn, b]=wcnoiseplus(Rxx, H, 1, 1e-5 , 1e-4);
>> Rwcn
1.0000 0.7500 0.0016
0.7500 1.0000 0.0012
0.0016 0.0012 1.0000
>> b = 11.3777
>> Swcn=inv(Rwcn)-inv(H*Rxx*H'*Rwcn) =
 0.9995 -0.0000 -0.0000
-0.0000 0.9948 -0.0000
-0.0000 0.0000 0.9948
```

so user 2 is secondary and users 1 and 3 are primary. The sum rate if all energy is on users 1 and 3 would then be 11.4 bits/dimension. BC receiver 2 is temporarily removed because it will simply treat the other two users as noise on whatever emanates from the channel’s two dimensions. The remaining matrix for users 1 and 3 is then

```matlab
>> H1=[H(1,1:3)
H(3,1:3)] =
80 60 40
20 20 20
>> [Rwcn, b]=wcnoiseplus(Rxx, H1, 1, 1e-5 , 1e-4);
>> Rwcn =
1.0000 0.0016
0.0016 1.0000
>> b = 11.3777
>> Swcn=inv(Rwcn)-inv(H1*Rxx*H1'*Rwcn) =
 0.9995 0.0000
-0.0000 0.9948
```

The special square root is next:

```matlab
>> J2=hankel([0 1]);
>> J3=hankel([0 0 1]);
>> [Q,R]=qr(J3*H1'*J2);
>> Q
```

345
-0.5774    0.7071    0.4082
-0.5774    -0.0000   -0.8165
-0.5774    -0.7071    0.4082

>> R =
-34.6410  -103.9230
0   -28.2843
0        0

>> R=(J3*R*J2)' =
0   -28.2843  -103.9230
0        0   -34.6410

>> Q=(J3*Q'*J3)' =
0.4082    -0.7071   -0.5774
-0.8165    -0.0000   -0.5774
0.4082    0.7071   -0.5774

>> R*Q' =
80.0000  60.0000  40.0000
20.0000  20.0000  20.0000 (checks)

>> R1=R(1:2,2:3) =
-28.2843  -103.9230
0   -34.6410

>> Q1=Q(1:3,2:3) =
-0.7071    -0.5774
-0.0000   -0.5774
0.7071    -0.5774

>> R1*Q1' =
80.0000  60.0000  40.0000
20.0000  20.0000  20.0000 (checks again)

>> Rxxrot=Q1'*Rxx*Q1 =
2.5000    0.4082
0.4082    3.0000

>> Phi=(J2*chol(J2*Rxxrot*J2)*J2)' =
1.5635    0.2357
0    1.7321

>> DA=diag(diag(R1*Phi)) =
-44.2217        0
0   -60.0000

>> G=inv(DA)*R1*Phi =
1.0000    4.2212
0    1.0000

>> A=Q1*inv(R1)*DA*G =
-1.1055   -1.1667
-0.0000   -1.0000
1.1055   -0.8333

>> A*A' =
2.5833    1.1667   -0.2500
1.1667    1.0000    0.8333
-0.2500    0.8333    1.9167

The product \( A \cdot A^* \neq R_{xx} \) only because \( R_{xx} \) places energy into the BC null space. The part of the energy that will pass is \( A \cdot A^* \) and the remaining difference \( R_{xx} - A \cdot A^* \) is not relevant and carries no useful information, although it does represent energy waste. Thus, the \( R_{xx} \) is not a very good design. Other choices would be better that have no such waste. The design now completes for primary users with

>> S0=DA*inv(Swcn)*DA =

346
The data rates with no secondary user are given above.

The transmitter input to the $A$ matrix has unit energy on each of the two primary user dimensions. The channel outputs thus have matrix description $y - n = H v$

$$H * A =
\begin{bmatrix}
-44.2217 & -186.6667 \\
-33.1662 & -140.0000 \\
-0.0000 & -60.0000 \\
\end{bmatrix}$$

Thus

$$y_3 = -60 \cdot v_2$$

It is not always true that secondary users depend only on 1 primary channel input dimension as in this example. Problem 2.30 is similar to this example, but has a secondary output that depends on both primary input dimensional components. Further

$$v_1 = \sqrt{E_1} \cdot v_1^0 + \sqrt{E_{1,2}} \cdot v_2$$

$$v_3 = \sqrt{E_3} \cdot v_3^0 + \sqrt{E_{3,2}} \cdot v_2$$

These two dimensions then enter matrix $A$ so

$$x = \begin{bmatrix}
-1.1055 & -1.1667 \\
-0.0000 & -1.0000 \\
1.1055 & -0.8333 \\
\end{bmatrix} \begin{bmatrix}
v_1 \\
v_2 \\
\end{bmatrix}$$

Energy distribution to user 2 reduces the rate sum, but user 2 may need to carry data. The original energy choices might provide guidance (Chapter 5 addresses user priority weighting) in that both user 1 and user 3 in the earlier (noted already poorly designed because it wastes energy in the BC null space) had 3 and 2 units of energy respectively, while user 2 had 4 units of energy. A choice might then be

$$E_1 = \frac{3}{7} \quad (2.483)$$

$$E_2 = \frac{2}{6} \quad (2.484)$$

and correspondingly then

$$E_{1,2} = \frac{4}{7} \quad (2.485)$$

$$E_{3,2} = \frac{4}{6} \quad (2.486)$$

The primary data rates must account for the secondary user so
\[ b = 0.5 \log_2(\text{diag}([3/7 \ 2/6]) \ast \text{diag}(S0)) = \\
4.8559 \\
5.1182 \]

User 2 remains with signal energy component
\[(60)^2 \cdot 4/6 = 2400 \ , \]
with noise components from only user 2 (because of the zero in \(\bar{H} \cdot A\)'s 3rd row) of
\[(60)^2 \cdot 2/6 = 1200 \ , \]

\[ b_3 = \frac{1}{2} \cdot \log_2 \left(1 + \frac{2400}{1200}\right) \]
\[ = 0.8 \text{ bits/subsymbol}. \]

The rate sum is now 4.85 + 5.12 + .80 = 10.77 < 11.33. A reduction in rate sum occurs for any nonzero energy allocation to user 2. Figure 2.57 summarizes the design, with the choices of energy illustrated specifically and generically.

\[ 4.22 \times \mathcal{E} = \mathcal{E}_x \]
\[ \mathcal{E}_u = 0 \times \mathcal{E} \]
\[ \mathcal{E}_u = & \times \mathcal{E}_x \]
\[ -1.11 -1.17 -1 -0.83 \]
\[ \times \]
\[ 44.2 \]
\[ \mathcal{E}_u \]
\[ 60 \]
\[ \mathcal{E}_x \]
\[ -1.11 \]
\[ 0 \]
\[ 1 \]
\[ 0 \]
\[ All \ Switches \ move \ once \ in \ direction \ of \ arrow \ simultaneously \]

Figure 2.57: Illustration of Example 2.8.5

2.8.4 Scalar duality for Gaussian MAC/BC Channels

Duality relates the MAC and BC when \(H_{MAC} = J \cdot H_{BC}^*\) in multi-user transmission (and first appeared before publication elsewhere in this text’s earlier on-line turn-century editions), where \(J\) is a reversal matrix (1’s on antidiagonal). The interchange of successive decoding and non-causal precoding allows movement of multi-user Gaussian channel’s “coordinated end” from transmitter to receiver and vice-versa. BC receiver \(u\)’s scalar \(h_u\) multiplies “other-user” noise for all other users; while for the common MAC receiver, the “other-user” noise from any user \((i \neq u)\)’s component has scaling \(h_i\) in user \(u\)’s detection. The BC has a single common energy constraint
\[ \sum_{u=1}^{U} \epsilon_{u}^{BC} \leq \mathcal{E}_x \ ; \quad (2.487) \]
while the MAC has additional restrictions of energy per user \(\epsilon_u\) individually bounded or an energy vector \(\mathcal{E}\) such that
\[
\begin{bmatrix}
\epsilon_{MAC}^1 \\
\vdots \\
\epsilon_{MAC}^U
\end{bmatrix} \leq \mathcal{E}_x
\quad (2.488)
\]
where \( \preceq \) denotes component-wise \( \leq \). Duality envisions a set of energy-sum MAC’s, each such MAC with its own \( \mathcal{E}_x \) such that any and all sum-of-components are less than \( \mathcal{E}_x \) or

\[
\{ \mathcal{E} | 1^* \mathcal{E} \leq \mathcal{E}_x \}. \tag{2.489}
\]

There is also an order reversal in duality that simplifies the energy-sum constraint equality between the MAC and BC energies (as is shown momentarily). The gap is 0 dB, and thus all users employ Gaussian capacity-achieving codes.

Figure 2.58 illustrates the basics.

Table 2.5 lists the \( \tilde{b}_u \) for the MAC and the BC with the order reversal in most preferred position evident (The most preferred BC user is 1 while the most preferred MAC user is \( U \). The reason for the earlier order reversal is now apparent because it simplifies the duality table entries.) The table’s bits per user subsymbol are equal, thus imposing a constraint on the selected energies\(^\text{81}\)

---

\(^81\)While the bit rates could always be set equal and the consequent energies of MAC derived from those of BC, the reversed-order of preference allows the sum of the energies to be the same.
\[ \bar{b}_1 = \frac{1}{2} \log_2 \left( 1 + \frac{e^{MAC} g_1}{1 + e^{MAC} g_2 + \ldots + e_U^{MAC} g_U} \right) \]
\[ \bar{b}_2 = \frac{1}{2} \log_2 \left( 1 + \frac{e^{MAC} g_2}{1 + e^{MAC} g_3 + \ldots + e_U^{MAC} g_U} \right) \]
\[ \vdots \]
\[ \bar{b}_U = \frac{1}{2} \log_2 \left( 1 + \frac{e^{MAC} g_U}{1 + e^{MAC} g_1 + \ldots + e_U^{MAC} g_U} \right) \]

Table 2.5: Scalar Duality bits/user

For equality of bit rates, these equations follow

\[ e^{BC}_1 = e^{MAC}_1 \cdot \frac{1}{1 + e^{MAC}_2 \cdot g_2 + \ldots + e_U^{MAC} \cdot g_U} \] (2.490)
\[ e^{BC}_2 = e^{MAC}_2 \cdot \frac{1}{1 + e^{MAC}_3 \cdot g_3 + \ldots + e_U^{MAC} \cdot g_U} \] (2.491)
\[ \vdots \] (2.492)
\[ e^{BC}_U = e^{MAC}_U \cdot (1 + [e^{BC}_1 + \ldots + e^{BC}_{U-1}] \cdot g_U) \] (2.493)

**Theorem 2.8.2 (Equal Sum Energy for Duality)** Scalar duality has the same MAC and BC energy sums when the bit rates are set equal, namely

\[ \sum_{u=1}^{U} e^{BC}_u = \sum_{u=1}^{U} e^{MAC}_u \] (2.495)

**Proof:** The sum of the equations:

\[ e^{BC}_1 \cdot [1 + e^{MAC}_2 \cdot g_2 + \ldots + e_U^{MAC} \cdot g_U] = e^{MAC}_1 \] (2.496)
\[ e^{BC}_2 \cdot [1 + e^{MAC}_3 \cdot g_3 + \ldots + e_U^{MAC} \cdot g_U] = e^{MAC}_2 \cdot [1 + e^{BC}_2 g_2] \] (2.497)
\[ \vdots \] (2.498)
\[ e^{BC}_U = e^{MAC}_U \cdot (1 + [e^{BC}_1 + \ldots + e^{BC}_{U-1}] \cdot g_U) \] (2.499)

equals

\[ \sum_{i=1}^{U} e^{BC}_i + \sum_{i=1}^{U} e^{BC}_i \cdot \sum_{k=i+1}^{U} g_k \cdot e^{MAC}_k = \sum_{i=1}^{U} e^{MAC}_i + \sum_{i=2}^{U} e^{MAC}_i \cdot g_i \cdot \sum_{k=1}^{i-1} e^{BC}_k \] (2.499)

Inspection of the 2nd term on the left and on the right in (2.499) when \( U = 3 \) provides insight:

\[ e^{BC}_1 (g_2 \cdot e^{MAC}_2 + g_3 \cdot e^{MAC}_3) + e^{BC}_2 \cdot g_3 \cdot e^{MAC}_3 \]
\[ = g_2 \cdot e^{BC}_1 \cdot e^{MAC}_2 + g_3 \cdot (e^{BC}_1 \cdot e^{MAC}_3 + e^{BC}_2 \cdot e^{MAC}_3) \] (2.500)
The remaining terms on the left and right of (2.499) are also equal. In general, the 2nd term on the left then can be rewritten

\[ \sum_{i=2}^{U} \xi_i^{MAC} \cdot g_i \cdot \sum_{k=1}^{i-1} \xi_k^{BC} \]

and thus again is equal to the 2nd term on the right, leaving the energy sums equal for the BC and the MAC. QED.

Chapter 5 provides the generalized Gaussian vector BC and MAC duality forms.

**EXAMPLE 2.8.6 (Example 2.8.1 revisited with duality)** Returning again to Examples 2.8.2 and 2.8.1 and Figure 2.48, again let \( h_1 = .8 \) and \( h_2 = .5 \) and \( \sigma^2 = .0001 \). As an example point for duality, let \( \xi_1^{BC} = \xi_2^{BC} = .5 \), or equivalently \( \alpha = .5 \). Then, \( g_1 = 80^2 \), \( g_2 = 50^2 \) and

\[ \xi_2^{MAC} = \frac{\xi_2^{BC}}{1 + \xi_1^{MAC}} \cdot g_2 = \frac{1}{2} \left( \frac{1}{1 + 2500(0.5)} \right) = \frac{1}{2502}, \]  

while then

\[ \xi_1^{MAC} = \xi_1^{BC} \cdot (1 + g_2 \cdot \xi_2^{MAC}) = (0.5) \cdot \left( 1 + \frac{2500}{2502} \right) = \frac{2501}{2502} = 1 - \xi_2^{MAC}. \]  

The users’ energy sum is again 1 as it should be. Then the two bits/subsymbol for the dual MAC channel are

\[ b_1 = \frac{1}{2} \log_2 \left( 1 + \frac{\xi_1^{MAC} \cdot g_1}{1 + \xi_2^{MAC} \cdot g_2} \right) = 5.82 \]  

\[ b_2 = \frac{1}{2} \log_2 \left( 1 + \frac{\xi_2^{MAC} \cdot g_2}{1} \right) = .50, \]

which are the same as for this energy point in Example 2.8.1. Thus, the rate region could be traced for set of dual MAC channels for which the sum of the energy constraints are 1. Duality avoids the need to find worst-case noise with the scalar BC. It also uses a suboptimal order for the MAC. With care to reverse MAC order with respect to its best order in duality, Subsection 2.7.2.2’s MMSE MAC follows for this dual channel as:

\>
H=[50 80];
Rxx=diag([1/2502 2501/2502]) =
    0.0004  0
    0  0.9996
Ht=H*sqrtm(Rxx) =
    0.9996  79.9840
Rf=Ht'*Ht = 1.00e+03 *
    0.0010  0.0800
    0.0800  6.3974
Rbinv=Rf+eye(2) = 1.00e+03 *
    0.0020  0.0800
    0.0800  6.3984
Gbar=chol(Rbinv);
G=inv(diag(Gbar))*Gbar;
Gunb=eye(2)+(S0*inv(S0-eye(2))*(G-eye(2))) =
    1.0000  80.0160
    0  1.0000
Wunb=inv(S0-eye(2))*inv(G')*Ht' =
    1.0004
\>
The two users’ bits/subsymbol match as expected both the BC and the dual-MAC. For receiver implementation, the large feedback tap (80) arises through user 2’s small energy and the MMSE-MAC’s message-input normalization to unit energy per dimension in Subsection 2.7.2.2 This is (with round-off error) the channel coefficient of $\sqrt{g_1}$ = 80. The feedforward filter (after bias removal) multiplies channel output 1 by .0125 = 1/80. Thus, while with suboptimal MAC order, the dual MMSE-MAC describes the original BC’s data rates, transforms to their equal BC-input energies, and provides the feedback coefficient for the receiver 1’s successive-decoder BC implementation (which is trivially $\sqrt{g_1}$ = 80 in this scalar case; however, the precoder coefficient is 1 because user 2 is secondary.

Duality uses an existing MAC capacity-region generator (for instance an easy pentagon generator for $U = 2$ and the Gaussian scalar MAC) and then forms the union of such regions for all possible energies that sum to the total allowed. The MAC’S energy-constraint vector is

$$\mathcal{E} = \begin{bmatrix} E_{1,\text{max}} \\ \vdots \\ E_{U,\text{max}} \end{bmatrix}. \quad (2.506)$$

The capacity rate region of the linear Gaussian BC would then be traced by the union of all the rate regions its dual MAC:

1. Initialize capacity set to $\{c(b)\} = \emptyset$.
2. for all $\mathcal{E}$ such that $1^*\mathcal{E} = \mathcal{E}_x$ (Using the discretization process to make computation finite as in (2.263)
   - Compute $a(b)$ for the MAC defined by $g_1, ..., g_U$ with $\mathcal{E}$ as energy-vector constraint.
   - Form $\{c(b)\} = \bigcup \{c(b), a(b)\}$.

Chapter 5 generalizes duality to the vector case, where the basics remain the same in that the BC and MAC rate sums and energy sums remain the same, but the precoders or successive decoders, respectively, simply follow more elaborate calculations to ensure the equalities. Duality then helps the next subsection’s discussion of the BC maximum rate sum’s calculation. The ability to generate a Gaussian MAC capacity rate region is sufficient to generate its dual Gaussian BC capacity region, which also helps express Section 2.10’s relay-channel capacity.

### 2.8.5 Maximum BC sum rate

The Gaussian BC channels so far have a specified $R_{xx}$. Optimization of $I_{wcn}(x; y)$ over $R_{xx}$ for the corresponding $R_{wcn}$ is indeed possible. The $R_{wcn}$ for the corresponding primary users of a dual MAC and BC is the same (same mutual information is minimized at the same $R_{xx}$). Using this fact, this process follows Figure 2.59’s iterative process:
Iterative Calculation of the BC Maximum Rate Sum:

1. Optimize the dual MAC for primary-user (block) diagonal $R_{xx}$’s rate-sum maximum with energy-sum constraint only. This step executes waterfall on the independent primary-users’ dimensions and SNR’s found from a MAC $S_{0,u}$ set, which were determined initially for $R_{xx} = \frac{\mathcal{E}_U}{\mathcal{E}}$. The water-fill solution provides a (block) diagonal MAC autocorrelation set with energies $\{E_u\}_{u=1}^{\infty}$ and correspondingly diagonal $R_{xx} = \text{Diag} \{E_u\}$, for which a $I_{wcn,bc}$ and corresponding $I_{wcn,bc}(x; y)$ can be found. This is Figure 2.59’s black arrow marked 1.

2. Step 1’ worst-case noise determines a dual MAC with the same $I_{temp} = I_{wcn,bc}(x; y)$. The MAC receiver for this $R_{wcn}$ has corresponding MSE’s $S_{0,u}$ for which a new water-filling set of energies can be found as $\{E'_u\}_{u=1}^{\infty}$. This $I_{wcn,mac} \leq I_{w,f,mac}$ because both the MAC water-fill always must allocate user energies to different dimensions when there are only primary users, and because the earlier set’s ($\{S_{0,u}\}$) receiver-only MAC processing’s optimization certainly considered the possibility of the newer set $\frac{S_{0,u}}{\mathcal{E}_U}$ in its $I_{w,mac}(x; y)$ original optimization. This is the blue arrow marked 2.

3. The process returns to the first step, now with the new water-filling energy set $\{E'_u\}_{u=1}^{\infty}$, which is Figure 2.59’s green arrow marked 1’. Figure 2.59’s red arrow marked 2’ is the repeat of the second step. When the change in mutual information is negligible, this process stops.

The above process must converge because the worst-case noise is concave in noise and the mutual information is convex in water-filling energy. The maximum rate sum will reduce if secondary users have non-zero energy, so they always carry zero energy for the maximum rate-sum calculation.

**EXAMPLE 2.8.7** (Example 2.8.4 continued) Example 2.8.4 continues here to follow the above iterative algorithm to compute the maximum bits/subsymbol.

```matlab
H = [ 80 70 ; 50 60 ];
>> Rxx=[1 .8 ; .8 1];
>> [Rwcn,b]=wcnoise(Rxx,H,1) =
1.0000 0.0232
0.0232 1.0000
b = 9.6430
>> Htilde=inv(Rwcn)*H =
78.8817 68.6440
48.1687 58.4064
>> Swcn = inv(Rwcn)-inv(H*Rxx*H’+Rwcn) =
0.9835 0.0000
```

353
\[
\begin{align*}
0.0000 & \quad 0.9688 \\
J2 &= \text{hankel([0 1])}; \\
\{Q,R\} &= \text{qr}(J2*Htilde'*J2); \\
S0 &= DA*\text{inv}(Swcn)*DA = 1.0e+04 * \\
& \quad 0.0061 \quad 0.0000 \\
& \quad 0.0000 \quad 1.0562 \\
\text{b} &= 0.5*\log2(\text{diag}(S0))' = 2.9597 \quad 6.6833 \\
\text{sum}(b) &= 9.6430 \\
\text{This is a rate sum, but not necessarily the maximum, but water-filling can follow for the two user channels created:} \\
iS0 &= \text{inv}(S0 - \text{eye}(2)) = \\
& \quad 0.0168 \quad -0.0000 \\
& \quad -0.0000 \quad 0.0001 \\
K &= 1+0.5*\text{sum}(\text{diag}(iS0)) = 1.0084 \\
E1 &= K-\text{iS0}(1,1) = 0.9916 \\
E2 &= K-\text{iS0}(2,2) = 1.0084 \\
[V,D] &= \text{eig}(Rxx) \\
V &= \\
& \quad -0.7071 \quad 0.7071 \\
& \quad 0.7071 \quad 0.7071 \\
D &= \\
& \quad 0.2000 \quad 0 \\
& \quad 0 \quad 1.8000 \\
Rxx1 &= V*\text{diag([E1,E2])}*V' = 1.0000 \quad 0.0084 \\
& \quad 0.0084 \quad 1.0000 \\
[Rwcn,b] &= \text{wcnoise}(Rxx1,H,1) = 1.0000 \quad 0.0048 \\
& \quad 0.0048 \quad 1.0000 \\
b &= 10.3517 \\
Htilde &= \text{inv}(Rwcn)*H; \\
Swcn &= \text{inv}(Rwcn)-\text{inv}(H*Rxx1*H'*Rwcn); \\
S0 &= DA*\text{inv}(Swcn)*DA = 1.0e+03 * \\
& \quad 0.2794 \quad 0.0000 \\
& \quad 0.0000 \quad 6.1112 \\
\text{b} &= 0.5*\log2(\text{diag}(S0))' = 4.0631 \quad 6.2886 \\
\text{sum}(b) &= 10.3517 \\
\text{Clearly the water-filling step increased the rate sum. With the new set of subchannels created, another iteration of water-filling can now occur:} \\
iS0 &= \text{inv}(S0 - \text{eye}(2)); \\
K &= 1+0.5*\text{sum}(\text{diag}(iS0));
\end{align*}
\]
\[
E_1 = K \cdot \mathbf{I} \cdot \mathbf{S}_0(1,1);
E_2 = K \cdot \mathbf{I} \cdot \mathbf{S}_0(2,2);
\]

\[
\begin{bmatrix}
1.0000 & 0.0017 \\
0.0017 & 1.0000
\end{bmatrix}
\]

\[
[R_wcn, b] = \text{wcnoise}(R_{xx2}, H, 1)
\]

\[
R_wcn =
\begin{bmatrix}
1.0000 & 0.0048 \\
0.0048 & 1.0000
\end{bmatrix}
\]

\[
b = 10.3517
\]

\[
[R_wcn, b] = \text{wcnoise}(R_{xx1}, H, 1)
\]

\[
R_wcn =
\begin{bmatrix}
1.0000 & 0.0048 \\
0.0048 & 1.0000
\end{bmatrix}
\]

\[
b = 10.3517
\]

There is no more improvement possible.

### 2.8.6 The continuous time Gaussian BC

The continuous-time Gaussian BC material will be added at a later date. However, it follows from scalar duality with the dimensional gains becoming transfer functions and energies becoming power spectral densities, both functions of frequency \( f \). Then the MAC continuous-time results can be applied directly to the continuous-time scalar Gaussian BC.

Figure 2.60 illustrates the basics.
2.8.7 Non-Gaussian BC rate regions

To be added.
2.9 The Gaussian Interference Channel

IC analysis uses Equation (2.258)’s minimum mutual-information vector, $\mathcal{I}_{\min}(\Pi, p_{xy})$. Capacity region construction for the single-receiver MAC and BC did not need this $\mathcal{I}_{\min}$, as per the simplified constructions in the preceding Sections 2.7 and 2.8. The Gaussian IC $\mathcal{C}_{IC}(b)$ (for the channel $y = H \cdot x + n$) however uses this step, which however often reduces with respect to the general solution’s complexity in Section 2.6.

The IC has (at least) two possible interpretations:

- **MAC set**: The Gaussian IC is a set of $U$ Gaussian MACs with a common (block) diagonal $R_{xx}$.

- **BC set**: The Gaussian IC is a set of $U$ dual Gaussian BC’s to the MACs in the MAC-set approach.

There is also an IC dual that corresponds to $\tilde{H}^*$ that can translate the MAC-set approach to/from the BC-set approach.

2.9.1 The MAC-set approach

For the Gaussian IC, (2.232)’s minimization occurs for a given order $\Pi$ and a given (necessarily [block] diagonal) input autocorrelation $R_{xx}$. The order search simplifies through use of the MAC (and BC) primary/secondary-user concepts. Some two-user examples first provide insight, with a scalar Gaussian IC first in Subsection 2.9.1.1 and then a vector IC with all-primary users in Subsection 2.9.2. The first example will have 2 users, one primary and one secondary for the IC’s $U = 2$ MAC receivers. Only $U = 2$ orders need consideration, not $(2!)^2 = 4$ orders. In the second example $L_y = 2$ for both receivers (with $L_x = 1$) and needs only 1 order. Subsection 2.9.2’s examples allow partition of a more general situation into sets of primary and secondary users that will be like the 1st and 2nd users in the examples, and allow significant reduction in the complex order-search step.

2.9.1.1 The Scalar Gaussian IC

![Interference-channel for Example 2.9.1](image)

**EXAMPLE 2.9.1** [Simple $2 \times 2$ scalar IC] Figure 2.61 illustrates a $2 \times 2$ scalar IC, within which are two scalar MACs to each physically separated output. The input $R_{xx}$ is diagonal
for the IC, and this example’s noise is also diagonal as in this example. The primary user
for each of the two MACs is user 2, leaving user 1 as the secondary user on each MAC (even
though receiver 1’s primary function is to decode user 1). Some calculations are necessary
to compute various data rates with different possible orders and input energies, so Table 2.6
simply computes and enumerates them before proceeding further.

Table 2.6: Some useful calculations for the upcoming example

<table>
<thead>
<tr>
<th>$\Pi$</th>
<th>$u$</th>
<th>$E_u$</th>
<th>$P_u(\pi_u)$</th>
<th>$I_u(x_u; y_u) / P_u(\pi_u)$</th>
<th>$I_u(x_u; y_u) / P_u(\pi_u)$</th>
<th>$I_{min, u}(\Pi, E)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 2</td>
<td>2 1</td>
<td>{1}</td>
<td>6.322</td>
<td>$\infty$</td>
<td>6.322</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1</td>
<td>$\emptyset$</td>
<td>.3973</td>
<td>.2378</td>
<td>.2378</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1</td>
<td>{2}</td>
<td>5.9071</td>
<td>$\infty$</td>
<td>5.9071</td>
<td></td>
</tr>
<tr>
<td>2 2</td>
<td>2 1</td>
<td>$\emptyset$</td>
<td>.9157</td>
<td>.6196</td>
<td>.6196</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1</td>
<td>$\emptyset$</td>
<td>.3973</td>
<td>$\infty$</td>
<td>.3973</td>
<td></td>
</tr>
<tr>
<td>2 1</td>
<td>2 1</td>
<td>{1}</td>
<td>6.322</td>
<td>.2378</td>
<td>.2378</td>
<td></td>
</tr>
<tr>
<td>1 2</td>
<td>1 1</td>
<td>{2}</td>
<td>5.9071</td>
<td>.6196</td>
<td>.6196</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.7: Evaluation of $I_{min}$ for different orders.

<table>
<thead>
<tr>
<th>$\Pi$</th>
<th>$u$</th>
<th>$E_u$</th>
<th>$P_u(\pi_u)$</th>
<th>$I_u(x_u; y_u) / P_u(\pi_u)$</th>
<th>$I_u(x_u; y_u) / P_u(\pi_u)$</th>
<th>$I_{min, u}(\Pi, E)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 2</td>
<td>2 1</td>
<td>{1}</td>
<td>6.322</td>
<td>$\infty$</td>
<td>6.322</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 0.5</td>
<td>{1}</td>
<td>2.257</td>
<td>.1287</td>
<td>.1287</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1</td>
<td>$\emptyset$</td>
<td>1.3063</td>
<td>.9478</td>
<td>.9478</td>
<td></td>
</tr>
<tr>
<td>2 2</td>
<td>1 0.5</td>
<td>{2}</td>
<td>5.4073</td>
<td>$\infty$</td>
<td>5.4073</td>
<td></td>
</tr>
<tr>
<td>2 2</td>
<td>2 0.5</td>
<td>{1}</td>
<td>5.822</td>
<td>$\infty$</td>
<td>5.822</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1</td>
<td>$\emptyset$</td>
<td>.6519</td>
<td>.4163</td>
<td>.4163</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>2 0.5</td>
<td>{1}</td>
<td>5.822</td>
<td>.3744</td>
<td>.3744</td>
<td></td>
</tr>
<tr>
<td>2 2</td>
<td>1 1</td>
<td>{1}</td>
<td>5.9071</td>
<td>$\infty$</td>
<td>5.9071</td>
<td></td>
</tr>
<tr>
<td>2 2</td>
<td>2 1</td>
<td>{1, 2}</td>
<td>6.322</td>
<td>$\infty$</td>
<td>6.322</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 0.95</td>
<td>{1}</td>
<td>.3818</td>
<td>.2276</td>
<td>.2276</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>2 1</td>
<td>{2}</td>
<td>.9425</td>
<td>.6412</td>
<td>.6412</td>
<td></td>
</tr>
<tr>
<td>2 2</td>
<td>1 0.95</td>
<td>{1, 2}</td>
<td>5.8701</td>
<td>$\infty$</td>
<td>5.8701</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.8: More example points with the best orders
Some of these points also appear in Figure 2.62(a). The dimension-sharing line has formula

\[ b_2 = \frac{6.322 - .6196}{5.9071 - .2378} \cdot b_1 + (6.322 + 1.006 \cdot .2378) = -1.006 \cdot b_1 + 6.5612 \quad (2.507) \]

Figure 2.62(a)’s dimension-shared pentagon is for these IC specific values and is \( \mathcal{C}(b)’s \) outer boundary via the following argument: Calculation of the two derivatives at the upper vertex elements with respect to \( \mathcal{E}_2 \) finds:

\[
\ln(2) \cdot \frac{db_2}{d\mathcal{E}_2} = \frac{3200}{6400 \cdot \mathcal{E}_2^2 + 1} = 0.4999 \quad (2.508)
\]

\[
\ln(2) \cdot \frac{db_1}{d\mathcal{E}_2} = -\frac{3200 \cdot 2500 \cdot \mathcal{E}_1}{6400 \cdot (\mathcal{E}_2^2 + 2500 \cdot \mathcal{E}_1 + 1) \cdot (6400 \cdot \mathcal{E}_2^2 + 1)} = - .1404 \quad (2.509)
\]

The ratio of these two for a constant small value of \( d\mathcal{E}_2 \) is the derivative \( db_2/db_1 \) at the upper vertex point (where \( \mathcal{E}_2 = 1 \) and \( \mathcal{E} = 1 \)). This ratio clearly has larger line-slope magnitude of -3.56 (slopes downward more quickly than rightward) than the dimension-sharing line’s slope of -1.006. Thus, the curvature is within the pentagon shown (this need not occur in general and depends on \( H \)). Figure 2.62(a) shows that points evaluated for either user’s reduced energy therefore fall in the interior also.

An enlarged capacity region also appears in Figure 2.62(b) for the energy-sum IC case where the energy constraint relaxes to \( \mathcal{E}_u = \mathcal{E}_1 + \mathcal{E}_2 \), allowing for larger single-user vertices that touch the axes, but otherwise follow the same format. Again, only two orders are of interest. Further the same comparison of derivative magnitudes holds at this point and tends to infinity when \( \mathcal{E}_1 \to 0 \) at the upper point, so dimension-sharing again dominates for this \( H \). Indeed any relaxed energy constraint will ensure that curvature would be below the dimension-sharing line.

**Scalar IC Order Generalization:** Example 2.9.1 provide insight that generalizes. Clearly the order matrix \( \Pi \) with user \( u \) simultaneously at the top (most preferred position) for all users in the MAC-set approach is important, particularly to determine vertices. There are only \( U = 2 \) such orders for the
2-user IC. In both these orders, the top user has maximum possible \( b_u \) value (for its given energy \( \mathcal{E}_u \)) at receiver \( u \), and the values for this same user at other receivers is irrelevant (set at \( \infty \)) because those receivers do not desire to know user \( u \)'s messages. Further those other receivers \( i \neq u \) do not use user \( u \) in any successive decoding for this order. When both \( \mathcal{E}_u > 0 \), there is a residual other-user data rate that has no effect on user \( u \)'s data rate at receiver \( u \) – this residual amount zeroes if there is only a energy-sum constraint and not individual non-zero energy constraints.

The convex combination of these two vertex points creates the line shown, which is not of slope -1 (so not a maximum rate-sum line) unless the channels are symmetric in the vertices. Example 2.9.1’s line slope of -1.006 has magnitude greater than -1 with tangent means that the maximum sum point \( b_{\text{max}} \) (in Example 2.9.1, this is the point \([6.322, 2378]\)) is the vertical axis’ variable \((b_2)\). If the slope had magnitude less than 1, then the other vertex is the max rate-sum point and has slope -1 there.

The final convex combination over input probability densities (simplifying to energy in this scalar Gaussian case) is independent of the convex hull over vertices (orders or dimension-sharing) and completes \( \mathcal{C}_{G-IC}(b) \)'s construction.

A similar argument holds for \( \pi^{-1}(U-1) \), when given any specific \( \pi^{-1}(U) \). There are \( U \) choices for \( \pi^{-1}(U) \). Given that specific choice user \( \pi^{-1}(U) \), the best order position for user \( \pi^{-1}(U-1) \) is position \( U-1 \) at all receivers. There are then \( U-1 \) such choices. This argument continues to \( \pi^{-1}(1) \), creating \( U! \) vertices. Since each vertex represents a maximum \( b_u \) for at least the one user in highest position. and that same vertex’ best other-user positions given that maximum-\( b_u \) choice and their respective conditional status for better-position users, any convex combination of other orders’ (unsearched) vertices would produce a points that necessary are within the convex hull of the searched vertices:

\[
b \in \bigcup_{\ell \in U! \text{ vertices}} b_{\ell} = \text{convhull}
\]

If \( U = 3 \), \( \mathcal{C}_{G-IC}(b) \)'s construction searches the \( 3! = 6 \) orders (with dummy variable \( x \in \{U \setminus \{u\} \} \) and \( y \in \{U \setminus \{u, x\} \} \) )

\[
\begin{bmatrix}
3 & 3 & 3 \\
x & x & x \\
y & y & y
\end{bmatrix}, \begin{bmatrix}
2 & 2 & 2 \\
x & x & x \\
y & y & y
\end{bmatrix}, \text{ and } \begin{bmatrix}
1 & 1 & 1 \\
x & x & x \\
y & y & y
\end{bmatrix}.
\]

For each particular order’s possible dummy-variable \((x,y)\) choices, the \(3\)-way) convex combinations with the other vertices create a 3-dimensional planes where the tangent plane \( \mathbf{1} \cdot \mathbf{b} = b_{\text{max}} \) touches the point (or points if symmetric) that has overall maximum rate sum. The rate region curves below the tangent plane at the rate-sum maximum point to the other vertices. For 4 users, there are \( 4! = 24 \ll (4!)^4 = 20,736 \) orders. In general, the search has

\[
|\mathbf{1}\text{search} = U! \leq (U!)^U.
\]

### 2.9.2 Scalar IC Generalizations on curvature and convex-hull:

The \( 2 \times 2 \) scalar Gaussian IC generalizes from Example 2.8.4 with (but now as an IC)

\[
\begin{align*}
6400 & \rightarrow g_{22} \quad 4900 \rightarrow g_{21} \\
2500 & \rightarrow g_{12} \quad 3600 \rightarrow g_{11}
\end{align*}
\]

The two general mutual-information derivative expressions then are

\[
\begin{align*}
\ln(2) \cdot \frac{db_2}{d\mathcal{E}_2} & = \frac{1/2}{\mathcal{E}_2 + 1} \quad \text{(2.514)} \\
\ln(2) \cdot \frac{db_1}{d\mathcal{E}_2} & = -\frac{1/2 \cdot g_{12} \cdot \mathcal{E}_1}{(\mathcal{E}_2 + g_{12} \cdot \mathcal{E}_1 + 1) \cdot (g_{22} \cdot \mathcal{E}_2 + 1)} \quad \text{OR} \\
\ln(2) \cdot \frac{db_1}{d\mathcal{E}_2} & = -\frac{1/2 \cdot g_{11} \cdot \mathcal{E}_1}{(\mathcal{E}_2 + g_{11} \cdot \mathcal{E}_1 + 1) \cdot (g_{21} \cdot \mathcal{E}_2 + 1)} \quad \text{(2.516)}
\end{align*}
\]
where the “or” corresponds to the two potential vertex choices for $b_1$. It is possible through the choices of the $g_{ij} > 0$ to cause the derivative to have less magnitude than the line slope in Example 2.9.1 and thus have a curved region instead of a pentagon, and here the convex hull (union) of energy combinations. Further, the pentagon could be come a rectangle for very small $g_{21}$ and/or $g_{12}$. Figure 2.63 shows the 3 possibilities.

Figure 2.63: IC capacity-region shape types.

Some authors, whom use a different more situationally dependent characterization of posed capacity regions (here they are the capacity regions), basically enumerate different magnitude of the $g_{ij}$ for the $2 \times 2$ case (see for instance [5], See p. 140, Figure 6.3). The successive steps of first $I_{\text{min}}$ and convex combinations of its possibilities over the necessary searched orders, and then variation of energy, with each step possibly involving convex hull and dimension sharing provides the general IC capacity region without need of special cases. The various shapes thereof elate to the relative derivative magnitudes. In general, the derivative generalizes to finding each vertex’ possible derivative sets

$$\{db_u/dE_u\}.$$  \hspace{1cm} (2.517)

There is a dimension-sharing plane of all the $b_{i\neq u, \text{max}}$ with $b_u$ dimension-sharing values that will intersect at this vertex. When the derivative magnitude with respect to another user $i \neq u$ has its component above the plane (larger change in $b_u$ than in $b_i$), then $C(b)$ will curve above the plane to that other point $b_{i\neq u}$. When the derivative’s magnitudes reverse, the dimension-sharing plane bounds the region. This derivative test, while not trivial, is perceptively easier than characterizing into strong, weak, very strong, etc, particularly when $U > 2$ and that characterization becomes overly complex. The basic derivative-set test(s) at each vertex is (are) sufficient.

2.9.2.1 The Vector Gaussian IC

A second example illustrates that the vector-IC channel simplifies for situations where all users are primary ($U^s = 0$ on all MACs that comprise an IC).

EXAMPLE 2.9.2 [4 x 2 IC with 2 users.] A Gaussian IC has a $4 \times 2$ channel matrix

$$\frac{y}{4 \times 1} = \begin{bmatrix} y_2 \\ y_1 \end{bmatrix} = \begin{bmatrix} H_2 \\ H_1 \end{bmatrix} \begin{bmatrix} x_2 \\ x_1 \end{bmatrix} + \begin{bmatrix} n_2 \\ n_1 \end{bmatrix},$$  \hspace{1cm} (2.518)
where the constituent MACs are

\[
H_2 = \begin{bmatrix}
h_{22} & h_{21}
\end{bmatrix} = \begin{bmatrix}
.9 & .3 \\
.3 & .8
\end{bmatrix}
\]

(2.519)

and

\[
H_1 = \begin{bmatrix}
h_{12} & h_{11}
\end{bmatrix} = \begin{bmatrix}
.8 & .7 \\
.6 & .5
\end{bmatrix}
\]

(2.520)

and \( R_{uu} = 0.01 \cdot I_1 \). Each receiver’s two dimensions, with two users (\( U = 2 \)), allows each constituent MAC to be primary (since \( \varrho_{H_2} = \varrho_{H_1} = 2 \)). Each of receiver 2 and receiver 1 can separately implement a MAC design for the given

\[
R_{xx} = \begin{bmatrix}
\mathcal{E}_2 & 0 \\
0 & \mathcal{E}_1
\end{bmatrix}
\]

(2.521)

This 4 × 2 example follows Example 2.9.1’s 2 × 2 MAC with \( U = 2 \), but each user is primary at both receivers. This again requires search of the two orders, each with the same user in upper priority position at both receivers, if the separate energy constraints of (2.521) apply individually. However, with a sum-energy constraint, there is only 1 order.

Receiver 2’s order places user 2 in the upper position where there is no crosstalk while receiver 1’s order places user 1 in that same position. This position, uniquely, has equal zero-forcing and unbiased MMSE, as in Section 2.7. The other user (which the respective receivers decoded first) must have a bits/subsymbol that equals the other receiver’s minimum value, relative to its maximum value on its receiver.

\[
\begin{align*}
&\text{>> } H_2 = \begin{bmatrix} 9 & 3 \\
3 & 8 \end{bmatrix}; \\
&\text{>> } R_{b2}\text{inv}=H_2^*H_2+\text{diag}([1 \ 1]) = \\
&\begin{array}{cc}
91 & 51 \\
51 & 74
\end{array} \\
&\text{>> } G_{\text{bar}2}=\text{chol}(R_{b2}\text{inv}) = \\
&\begin{array}{cc}
9.5394 & 5.3463 \\
0 & 6.7393
\end{array} \\
&\text{>> } G_2=\text{inv(diag(diag(G_{\text{bar}2})))*G_{\text{bar}2}} = \\
&\begin{array}{cc}
1.0000 & 0.5604 \\
0 & 1.0000
\end{array} \\
&\text{>> } S_{02}=\text{diag(diag(G_{\text{bar}2}))*diag(diag(G_{\text{bar}2}))} = \\
&\begin{array}{cc}
91.0000 & 0 \\
0 & 45.4176
\end{array} \\
&\text{>> } 0.5*\log_2(diag(S_{02})) = \\
&b_2 = 3.2539 \\
&b_1 = 2.7526 \\
&\text{>> } H_1 = \begin{bmatrix} 8 & 7 \\
6 & 5 \end{bmatrix}; \\
&\text{>> } R_{b1}\text{inv}=H_1^*H_1+\text{diag}([1 \ 1]) = \\
&\begin{array}{cc}
101 & 86 \\
86 & 75
\end{array} \\
&\text{>> } G_{\text{bar}1}=\text{chol}(R_{b1}\text{inv}); \\
&\text{>> } S_{01}=\text{diag(diag(G_{\text{bar}1}))*diag(diag(G_{\text{bar}1}))}; \\
&\text{>> } 0.5*\log_2(diag(S_{01})) = \\
&b_2 = 3.3291 \\
&b_1 = 0.4128 \\
&\text{>> } J_2=\text{hankel([0 \ 1]);}
\end{align*}
\]
\[
\begin{align*}
\text{Rb2inv} &= J_2 H_2' H_2 J_2 + \text{diag([1 1])} = \\
&= \begin{bmatrix} 74 & 51 \\
51 & 91 \\
\end{bmatrix} \\
\text{Gbar2} &= \text{chol(Rb2inv)}; \\
\text{S02} &= \text{diag(diag(Gbar2))} \times \text{diag(diag(Gbar2))}; \\
0.5 \log_2 (\text{diag(S02)}) &= \\
b_1 &= 3.1047 \\
b_2 &= 2.9018 \\
\text{Rb1inv} &= J_2 H_1' H_1 J_2 + \text{diag([1 1])}; \\
\text{Gbar1} &= \text{chol(Rb1inv)}; \\
\text{S01} &= \text{diag(diag(Gbar1))} \times \text{diag(diag(Gbar1))}; \\
0.5 \log_2 (\text{diag(S01)}) &= \\
b_1 &= 3.1144 \\
b_2 &= 0.6275 \\
\text{For } \mathcal{E}_2 = 0.5: \\
\text{Rb2inv} &= \text{diag([1/sqrt(2) 1])} \times H_2' H_2 \text{diag([1/sqrt(2) 1])} + \text{diag([1 1])} = \\
&= \begin{bmatrix} 46.0000 & 36.0624 \\
36.0624 & 74.0000 \\
\end{bmatrix} \\
\text{Gbar2} &= \text{chol(Rb2inv)}; \\
\text{S02} &= \text{diag(diag(Gbar2))} \times \text{diag(diag(Gbar2))}; \\
0.5 \log_2 (\text{diag(S02)}) &= \\
b_2 &= 2.7618 \\
b_1 &= 2.7575 \\
\text{Rb1inv} &= \text{diag([1/sqrt(2) 1])} \times H_1' H_1 \text{diag([1/sqrt(2) 1])} + \text{diag([1 1])} = \\
&= \begin{bmatrix} 51.0000 & 60.8112 \\
60.8112 & 75.0000 \\
\end{bmatrix} \\
\text{Gbar1} &= \text{chol(Rb1inv)}; \\
\text{S01} &= \text{diag(diag(Gbar1))} \times \text{diag(diag(Gbar1))}; \\
0.5 \log_2 (\text{diag(S01)}) &= \\
b_2 &= 2.8362 \\
b_1 &= 0.6581 \\
\text{Rb2inv} &= \text{diag([1 1/sqrt(2)])} \times J_2 H_2' H_2 J_2 \text{diag([1 1/sqrt(2)])} + \text{diag([1 1])} = \\
&= \begin{bmatrix} 74.0000 & 36.0624 \\
36.0624 & 46.0000 \\
\end{bmatrix} \\
\text{Gbar2} &= \text{chol(Rb2inv)}; \\
\text{S02} &= \text{diag(diag(Gbar2))} \times \text{diag(diag(Gbar2))}; \\
0.5 \log_2 (\text{diag(S02)}) &= \\
b_1 &= 3.1047 \\
b_2 &= 2.4146 \\
\text{Rb1inv} &= \text{diag([1 1/sqrt(2)])} \times J_2 H_1' H_1 J_2 \text{diag([1 1/sqrt(2)])} + \text{diag([1 1])} = \\
&= \begin{bmatrix} 74.0000 & 36.0624 \\
36.0624 & 46.0000 \\
\end{bmatrix} \\
\text{Gbar1} &= \text{chol(Rb1inv)}; \\
\text{S01} &= \text{diag(diag(Gbar1))} \times \text{diag(diag(Gbar1))}; \\
0.5 \log_2 (\text{diag(S01)}) &= \\
b_1 &= 3.1144 \\
b_2 &= 0.3799
\end{align*}
\]
For $E_1 = 0.5$:

```matlab
>> Rb2inv=diag([1 1/sqrt(2)])*H2'*H2*diag([1 1/sqrt(2)])+diag([1 1]) =
   91.0000  36.0624
   36.0624  37.5000
>> Gbar2=chol(Rb2inv);
>> S02=diag(diag(Gbar2))*diag(diag(Gbar2));
>> 0.5*log2(diag(S02)) =
   b2 =  3.2539
   b1 =  2.2683

>> Rb1inv=diag([1 1/sqrt(2)])*H1'*H1*diag([1 1/sqrt(2)])+diag([1 1]) =
   101.0000  60.8112
   60.8112  38.0000
>> Gbar1=chol(Rb1inv);
>> S01=diag(diag(Gbar1))*diag(diag(Gbar1));
>> 0.5*log2(diag(S01)) =
   b2 =  3.3291
   b1 =  0.2355

>> Rb2inv=diag([1/sqrt(2) 1])*J2*H2'*H2*J2*diag([1/sqrt(2) 1]) + diag([1 1]) =
   37.5000  36.0624
   36.0624  91.0000
>> Gbar2=chol(Rb2inv);
>> S02=diag(diag(Gbar2))*diag(diag(Gbar2));
>> 0.5*log2(diag(S02)) =
   b1 =  2.6144
   b2 =  2.9078

>> Rb1inv=diag([1/sqrt(2) 1])*J2*H1'*H1*J2*diag([1/sqrt(2) 1]) + diag([1 1]) =
   38.0000  60.8112
   60.8112  101.0000
>> Gbar1=chol(Rb1inv);
>> S01=diag(diag(Gbar1))*diag(diag(Gbar1));
>> 0.5*log2(diag(S01)) =
   b1 =  2.6240
   b2 =  0.9407

>> Rb2inv=diag([1 .95])*H2'*H2*diag([1 .95]) + diag([1 1]);
>> Gbar2=chol(Rb2inv);
>> S02=diag(diag(Gbar2))*diag(diag(Gbar2));
>> 0.5*log2(diag(S02)) =
   b2 =  3.2539
   b1 =  2.6803

>> Rb1inv=diag([1 .95])*H1'*H1*diag([1 .95]) + diag([1 1]);
>> Gbar1=chol(Rb1inv);
>> S01=diag(diag(Gbar1))*diag(diag(Gbar1));
>> 0.5*log2(diag(S01)) =
   b2 =  3.3291
   b1 =  0.3815
```

364
Example 2.9.1 searches 2 orders, but Example 2.9.2 searches only one order. The examples somewhat merge the step of finding $\mathcal{I}_{\min}$ over all orders with the subsequent step of then finding convex hull over all orders. That subsequent step is in general

$$\mathcal{A}_{GIC}(b, R_{xx}) = \bigcup_{\Pi} \mathcal{I}_{\min}(\Pi, R_{xx}) \,.$$

Any point outside this convex-hull/union will violate the single-user capacity theorem for at least one user that must be decoded by at least one MAC-set receiver. The convex hull operation corresponds to the dimension-sharing. This dimension sharing essentially reduces the order search because for the primary users, the only $\Pi$ that matters is the one where each user is at the top of its column. This is the maximum data rate for that user at its receiver. Convex combinations of the different maxima (and associated other primary user rates) subsumes all the other orders. When the set $u^* \neq \emptyset$, then there is a possibility of $(U^* + 1)!$ different orders of the primary user set with all the possibly secondary user-set orders, which may have a variety of options for the secondary users’ best data rates. Thus, in general the order search is over $(U^* + 1)! \leq (U!)^U$ possible orders (often significantly less).

The convex hull of all permitted $R_{xx}$ that conform to the energy limit $\mathcal{E}$ then completes the capacity region as

$$\mathcal{C}_{GIC}(b) = \bigcup_{\text{trace}(R_{xx}) \leq \mathcal{E}} \mathcal{A}(b, R_{xx}) \,.$$

The examples illustrated a portion of this step by allowing some additional energy choices (namely total sum energy), but not all choices. In the simple $L_x = 1$ examples, this is exhaustive, but it is not in general where different $R_{xx}$ must be enumerated (and this can be tedious). Specifically, the pentagons for the simple MAC yield to more complicated regions for each $R_{xx}$’s corresponding intersection. However, these more complicated sets remain convex because of the convex-hull operation over orders/dimension-sharing.

### 2.9.3 The BC-set approach

The BC-set approach uses duality. Each of the MAC’s in the MAC approach can be viewed as the dual of a corresponding BC with $H^*$ as the dual IC channel. In effect, this approach simply finds the MAC
set. The IC’s limitation of independent energy for each user at each transmitter is effectively the $b_{\text{min}}$ step where the minima of different possibilities correspond to the exclusion of energy from the precoders. This BC-set approach is useful in relay/mesh channels’ $C(b)$ characterization.
2.10 Generalized Multi-User Channels

Many multi-user channels are combinations of the 3 basic types: MAC, BC, and IC. These combinations allow the 3 previous sections' simplified rate-region constructions’s use for the combination channel’s capacity region. Figure 2.65 provides an example mixture that Subsection 2.10.1 investigates in more detail. Subsection 2.10.1 also introduces revisits the sub-user concept. This chapter’s results and capacity regions then apply to an expanded sub-user set. These capacity regions have larger dimensionality. Subsection 2.10.1 computes the desired capacity region from expanded achievable regions. Expanded multi-user channels leverage the concept of order and successive decoding (or precoding). Subsection 2.10.2 addresses relay channels (or mesh networks).

![Figure 2.65: Mixed Gassian Channel.](image)

2.10.1 User Expansion and the Capacity Region

Figure 2.65 has three 2-user MACs ([5, 1], [3, 2], and [6, 4]) with possibly interdependent energies and one single-user channel (7) that shares transmit energy with two of the MACs. Alternately, this same multi-user channel is a combination of 3 BC’s ([1, 2], [3, 4], and [5, 6, 7]), but there is coordinated reception among users in different BC’s. These embedded mixed channels can each individually expand their user set to all \( U = 7 \) users over all these combinations.

Generalized MAC Set: An expanded multi-user channel has a number of sub-users \( U \) that exceeds the number of transmit (MAC, IC, or general multi-user) or receive (BC, IC, or general multi-user) locations in the channel description. For instance, the Figure 2.65’s 2-user MAC [5, 1] may expand its user set to all 7 users, call this 7-dimensional bit vector then \( b_u \). The input autocorrelation matrix \( R_{xx} \) spans all 7 users. For this expanded-user MAC, for any \( 0 \leq b \neq 1,5 < \infty \) and the given \( R_{xx} \), the achievable region \( A(b, R_{xx}) \) is the same pentagon for \( b_1 \) and \( b_5 \). Indeed, if Figure 2.65’s lines represent 1 dimension each, this 7-user region expands in 5 positive half spaces to \( \infty \) on the users not received, essentially viewing these users as zero-energy noises. The concept generalizes conceptually if the users have more than 1 dimension.

For the same \( R_{xx} \), the other two 2-user MACs also both expand similarly to 7-user MACs, each with its own set of not-received users’ half spaces with the same pentagon for the 2 received users. The single-channel expands to 7 users with 6 semi-infinite half spaces, again for the specific \( R_{xx}(7) \) that is within \( R_{xx} \). Repeating from (2.232):

\[
I_{\text{min},u}(\Pi, p_{xy}) = \min_{i \in \{P_{\pi^{-1}(u)}\}} \left\{ \mathcal{J}_i \left( x_{\pi^{-1}(u)}; y_i / P_{\pi^{-1}(\pi_i)} \right) \right\}.
\]

\[ (2.524) \]

82 “larger” means that \( U \) is enlarged so that the rate-region plot has \( |b| > U \) dimensions.
The achievable region then follows as
\[ A_{\text{GMUC}}(b, R_{xx}) = \bigcup_{\Pi} \{ A_{5.1}(b_{e,5.1}, \pi_{5.1}, R_{xx}); A_{3.2}(b_{e,3.2}, \pi_{3.2}, R_{xx}); \} \times \ \cdots \times \ \cdots \ \times \ A_{6.4}(b_{e,6.4}, R_{xx}); A_{7}(b_{e,7}, R_{xx}) \} \]  
\[ \text{(2.525)} \]

is compact for each \( R_{xx} \). Then, the capacity region searches over all \( R_{xx} \) that satisfy the energy constraints \( R_{xx} \in \{ R_{\text{allowed}} \} \):
\[ C_{\text{GMUC}}(b) = \bigcup_{R_{xx} \in \{ R_{\text{allowed}} \}} A(b, R_{xx}) \]  
\[ \text{(2.526)} \]

Typically, the BC-set approach is more complex than the MAC-set approach. However, the final capacity construction then completes searches allowed autocorrelations, repeating then (2.526). The BC-set approach finds use in Subsection 2.10.2’s relay-channel capacity-region construction.

### 2.10.2 Relay Channel Capacity Region

An alternative extended-user channel example is Figure 2.66’s relay channel. Relay channels have intermediate users that decode and then forward messages from some users to different users. There are \( k_r \) intermediate relays, and Figure 2.66’s first (left-side) stage extends users to a maximum of \( k_r \cdot U \). The corresponding first-stage capacity region description has \( k_r \cdot U \) users with corresponding \( k_r \times U \) bit matrix \( B \), where each column is a \( U \times 1 \) bit vector \( b_k \) with \( k = 1, \ldots, k_r \) elements that represents users’ bits/subsymbols to relay \( k \). Each relay has a first-stage-side achievable region \( A_{k}^{(1)} \left( B^{(1)}, R_{xx}^{(1)} \right) \), paralleling the IC approach but retaining the extended users through the subscript \( k = 1, \ldots, k_r \) so \( U \cdot k_r \) total users. The first (left) stage thus has a \( k_r \times U \) bit matrix:
\[ \left[ b^{(1)}_{k,1} \ldots b^{(1)}_{k,r} \right] \]  
\[ \text{(2.531)} \]

which has meaning for any bit vector, but is of interest for each column’s minimum elements. These minimum bit vectors at the intermediate relay stage for each relay \( k = 1, \ldots, k_r \) are
\[ b_{k,\text{min}}^{(1)}(\pi^{(1)}_k, R_{xx}^{(1)}) \triangleq \begin{bmatrix} \min_{u \in S_{U}(\Pi^{(1)}_k, R_{xx}^{(1)})} \left( b^{(1)}_{\pi^{(1)}_u k} \right) \left( k, R_{xx}^{(1)} \right) \vdots \vdots \min_{u \in S_{U}(\Pi^{(1)}_k, R_{xx}^{(1)})} \left( b^{(1)}_{\pi^{(1)}_u k} \right) \left( k, R_{xx}^{(1)} \right) \end{bmatrix} \]  
\[ \text{(2.532)} \]
The $U \cdot k_r$-dimensional first-stage extended-user achievable region is then the convex-hull combination

$$A^{(1)}(B^{(1)}, R^{(1)}_{xx}) = \bigcup_{\Pi^{(1)}}^{\text{conv}} \left\{ B^{(1)}_{\min} \left( \Pi^{(1)}, R^{(1)}_{xx} \right) \right\} .$$  \hfill (2.533)

The $k_r$ achievable first-stage rate vector possibilities are then all the vectors $\beta_k^{(1)}$ that satisfy

$$B_k^{(1)}(R^{(1)}_{xx}) = \left\{ \beta_k^{(1)} \mid \beta_k^{(1)}(R^{(1)}_{xx}) \in A^{(1)}(B^{(1)}, R^{(1)}_{xx}) \right\} .$$  \hfill (2.534)

The $\beta_k^{(1)}$ vectors have individual vector elements $\beta_k^{(1)}(u, R^{(1)}_{xx})$ for the users $u = 1, ..., U$ that index each column-vector’s elements. The second stage similarly has a $U \times k_r$ bit matrix for a BC-set approach

$$B^{(2)} \triangleq \left[ b_2^{(2)}, ..., b_U^{(2)} \right] .$$  \hfill (2.535)

which has meaning for any bit vector, but of interest for the minimum as for each column. These minimum bit vectors at the relay-channel outputs $u = 1, ..., U$ are

$$b^{(2)}_{u, \min}(k, \pi_u^{(2)}, R^{(2)}_{xx}) = \left[ \begin{array}{c}
\min_{k \in S_k^{(2)}}(\Pi^{(2)}, R^{(2)}_{xx}) \left\{ b^{(2)}_{[\pi^{(2)}]^{-1}(k)}(u, R^{(2)}_{xx}) \right\} \\
\vdots \\
\min_{u \in S_u^{(2)}}(\Pi^{(2)}, R^{(2)}_{xx}) \left\{ b^{(2)}_{[\pi^{(2)}]^{-1}(k)}(u, R^{(2)}_{xx}) \right\}
\end{array} \right] .$$  \hfill (2.536)

The convex-hull-over-orders step becomes the $k_r \cdot U$-dimensional region

$$A^{(2)}(B^{(2)}, R^{(2)}_{xx}) = \bigcup_{\Pi^{(2)}}^{\text{conv}} \left\{ B^{(2)}_{\min} \left( \Pi^{(2)}, R^{(2)}_{xx} \right) \right\} .$$  \hfill (2.537)

This second region essentially expands the $U$ second-stage BCs to $k_r \cdot U$ users. The $U$ achievable second-stage rate vector possibilities are then all the vectors $\beta_k^{(2)}$ that satisfy

$$B_u^{(2)}(R^{(2)}_{xx}) = \left\{ \beta_u^{(2)} \mid \beta_u^{(2)}(R^{(2)}_{xx}) \in A^{(2)}(B^{(2)}, R^{(2)}_{xx}) \right\} ,$$  \hfill (2.538)

Figure 2.66: Simple Two-Stage relay channel.
with individual elements $\beta^{(2)}_u(k, R^{(2)}_{xx})$.

A stage-combination step contracts the number of users to the original $U$ and computes for each possible autocorrelation-stage pair $[R^{(1)}_{xx}, R^{(2)}_{xx}]$ the corresponding achievable region’s entries through

$$b_u(R^{(1)}_{xx}, R^{(2)}_{xx}) = \left\{ b_u \mid b_u \in \sum_{k=1}^{k_r} \min_{\beta_k^{(1)} \in \mathcal{B}_k^{(1)}, \beta_u \in \mathcal{B}_u^{(2)}} \left[ \beta^{(1)}_k (u, R^{(1)}_{xx}), \beta^{(2)}_u (k, R^{(2)}_{xx}) \right] \right\}.$$  \hspace{1cm} (2.539)

The combined two-stage relay channel has $U$ users and bits/subsymbol vector $b$. The extended users' data rates may be implied, but hidden, in a final capacity region of $U$ users for the relay channel. Again, the convex-hull operation over each and every allowed $\Pi$ in each stage arises to compute the achievable regions $\mathcal{A}^{(1)}(B, R^{(1)}_{xx})$ for the first relay stage and $\mathcal{A}^{(2)}(B, R^{(2)}_{xx})$ for the second relay stage. Essentially, then the enlarged channel is searched over all possible enlarged rate tuples for the maximum possible transfers over all possible multi-hop paths. While the concept is fairly straightforward, the actual search can be complex. These bits/sub-symbol-user entries in (2.539) then form a $U \times 1$ vector $b$ and the capacity region is

$$C_{RC2}(b) = \bigcup_{[R^{(1)}_{xx}, R^{(2)}_{xx}]} \left\{ \left[ b_1(R^{(1)}_{xx}, R^{(2)}_{xx}) \right] \right\},$$  \hspace{1cm} (2.540)

which is the two-stage relay-channel capacity region.

### 2.10.3 Multi-Stage Relay Capacity Regions

The $S > 1$ stage relay follows the $S = 1$ single-stage relay of Subsection 2.10.2, but becomes somewhat tedious. The first $S - 1$ stages cascade in to a expanded set of

$$U^e = k_r^{(S-1)} \cdot k_r^{(S-2)} \cdot k_r^{(1)} \cdot U$$  \hspace{1cm} (2.541)

users. $B$ becomes then an $S$-dimensional bit tensor. Each MAC stage expands the user set by a multiplicative factor of $k_r^s$ where $s = 1, ..., S - 1$. The order becomes the Cartesian-Product tensor

$$\Pi = \otimes_{s=1}^{S-1} \Pi^s.$$  \hspace{1cm} (2.542)

The last stage has order matrix $\Pi^S$ also with $U^c$ users that contract to the original $U$ users. The sum in (2.539) becomes an $S - 1$ stage sum and the minimum is overall all the stages in the sum of the form:

$$b_u(R^{(1)}_{xx}, R^{(2)}_{xx}, ..., R^{(S)}_{xx}) = \sum_{k=1}^{k_r^{(S-1)}} \sum_{k=1}^{k_r^{(S-2)}} \min_{\beta_k^{(1)} \in \mathcal{B}_k^{(1)}, \beta_k^{(2)} \in \mathcal{B}_k^{(2)}, \beta^{(S)}_u \in \mathcal{B}_u^{(S)}} \left[ \beta^{(1)}_k (u, R^{(1)}_{xx}), \beta^{(2)}_k (k^{(1)}_r, R^{(2)}_{xx}), \beta^{(3)}_k (k^{(2)}_r, R^{(3)}_{xx}), ..., \beta^{(S)}_u (k^{(S-1)}_r, R^{(S)}_{xx}) \right].$$  \hspace{1cm} (2.543)

### 2.10.4 Mesh Networks

Mesh networks slightly adjust multi-stage relay networks in that users from earlier stages have active usable channels not only to the next stages’ relays but also to subsequent stages’ relays. Again, user expansion addresses this situation. Such a mesh user with channels directly to two or more stage relays (or the final stage receivers) subdivides into sub users corresponding to the different stages’ end point for that user. All searches and computations treat the sub-users as separate users until the step corresponding to Equation (2.543). The minimum operation in (2.543) must consider any such recombination of sub users at any intermediate stage (or at the final stage) as the sum of $b_u(R_{xx})$ values to that relay or final stages through the previous stages, instead of just the minimum of the individual channels between stages. While simple in concept, this can lead to very high complexity in step (2.543).
Exercises - Chapter 2

2.1 16HEX Coding Gain, Subsection 2.1.1 - 12 pts
This problem studies Figure 2.2(b)’s 16HEX constellation with each subsymbol being equally likely in comparison to 16QAM. The study begins with a translated 16HEX constellation that has a subsymbol vector at the origin, and then offsets the entire constellation by a constant to have zero mean value.

a. Find the translated 16 HEX’s average energy that uses 16 points with the lowest energy and with one point at the origin with $d_{\text{min}} = 2$. (2 pts)

b. Find this translated 16 HEX’ mean value. (2 pts)

c. By using the fact that the mean-square is the sum of the variance (about the mean) and the squared mean, compute the energy of the minimum-energy translate of 16HEX. [hint: look at Figure 2.2(b)]’s placement of axes. (2 pts)

d. Find the coding gain of 16HEX over 16QAM. (2 pts)

e. By reviewing Chapter 1, Section 1.3.4, interpret the answer to Part d as the number of points in the HEX constellation becomes very large and interpret the difference in terms of shaping gain and fundamental gain [hint: consider the constellations 7HEX and 19HEX and what might be special about them.] (2 pts)

f. Following Part e, suppose 14 input bits form a block, how many 7HEX symbols would be needed to cover the possibilities. How many extra message possibilities would be left over? Why did this question suggest 14 bits here as an input block size? (2 pts)

2.2 3D Lattice-based codes and coding gain Subsections 2.1.1 and 2.2.1 - 14 pts
The face-centered-cubic lattice has generator vectors

$$g_1 = \frac{d}{\sqrt{2}} \cdot [1 \ 1 \ 0] \quad (2.544)$$

$$g_2 = \frac{d}{\sqrt{2}} \cdot [0 \ 1 \ 1] \quad (2.545)$$

$$g_3 = \frac{d}{\sqrt{2}} \cdot [1 \ 0 \ 1] \quad (2.546)$$

a. Compute the minimum distance in terms of $d$. (1 pt)

b. Draw a picture of the the lowest energy 13 points if one lattice point is centered at the origin and comment on the name of this lattice. (2 pts)

c. Compute the number of nearest neighbors. (1 pts)

d. Find the fundamental volume for a decision or Voronoi region of this lattice. (2 pts)

e. Find the fundamental coding gain of this FCC lattice. (2 pts)

f. Compare this FCC lattice to the two-dimensional best lattice in terms of fundamental gain. If better, how many dB better? (2 pts)

g. The $A_3$ lattice builds upon two dimensions of the $A_2$ lattice. $A_3$ then centers adjacent layers above and below this 2D lattice in the “holes” that would appear if circles were “penny-packed” in 2D. Find a set of generator vectors for the $A_3$ lattice and find its fundamental coding gain. (4 pts)
2.3 Log Likelihood Computation for PAM - Subsections 2.1.4 and 2.2.1 - 17 pts

Figure 2.67 shows an 8PAM constellation with “Gray” bit encoding\textsuperscript{83} for transmission over an AWGN with noise $\sigma^2$. This problem investigates the direct minimization of each bit’s error probability rather than the overall symbol-error probability. The equally likely input bits will be labelled $u_1$, $u_2$, and $u_3$. The MAP bit error probability is $p_{u_i|y}$ for bit $i = 1, 2, 3$ and channel output $y$, which is proportional to $p_{u_i,y}$ for any given channel output $y$, and thus the MAP detector minimizes $p_{u_i,y}$ over the two input bit choices $u + i = 0$ or $1$.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{8PAM.png}
\caption{8 PAM for bit MAP decoding.}
\end{figure}

a. Show that (with $x(u_j)$ being the 8-PAM value corresponding to $u_j$ having a particular value, 1 or 0)

$$p_{u_i,y} \propto \sum_{u_j \neq i} e^{-\frac{(y-x(u_j))^2}{2\sigma^2}} \quad (2 \text{ pts}).$$

b. For any given $y$, how many values can your answer in Part a take? (1 pt)

c. How many terms are there in Part a’s sum for each $u_i$ value? (1 pt)

d. Compute the values for all 3 bits’s possibilities when $y = 5.1$ and $\sigma^2 = .25$. (3 pts)

e. Find the corresponding 3 LLR values for the situation in Part d. (3 pts)

f. With $x^*(u_i = 1)$ being the closest 8-PAM point to $y$ that has a $u_i = 1$ value and correspondingly $x^*(u_i = 0)$ being the closest 8-PAM point to $y$ that has a $u_i = 0$, show that the LLR is approximated for reasonable noises by

$$LLR_i \approx [y - x^*(i = 0)]^2 - [y - x^*(i = 1)]^2 \quad . \quad (2 \text{ pts})$$

g. Compare the answers in Parts e and f for the values given in Part e. (3 pts)

h. Would the answers above change if a symbol-error-ML detector were used on the final decisions? Would this always be true if the code were more complex? If not, what would it depend on? (2 pts)

2.4 Lattice structure in encoding/decoding (D4 example) - Section 2.2.1 - 22 pts

This problem develops some interesting properties for encoding and decoding with (linear) lattice codes. The set of all integer ordered pairs will be called $Z^2$, which is trivially a lattice under closure of integer addition. The $D_2$ set of subsymbol vectors will correspond to taking every other ordered pair in a checkerboard fashion from $Z^2$, partitioning it into the even and odd points as in Figure 2.3. The matrix

$$G_{D_2} \triangleq \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

formally defines $D_2$ as $D_2 = G_{D_2} \cdot Z^2$, which means the matrix operates on any ordered pair vector of integer inputs to generate corresponding outputs.

a. Show that $D_2$ is a lattice (that is closed under addition of any two points). What are two generator vectors of $D_2$. (2 pts)

---

\textsuperscript{83}After Frank Gray, who is credited for designing constellations in which adjacent points differ in only one bit in the bit-level encoding.
b. A coset adds the constant vector \([0 1]\) to each point of \(D_2\) to form \(D_2 + [0 1]\). Identify the union \(D_2 \cup D_2 + [0 1]\). (1 pt).

c. Following Part b, \(D_2\) partitions \(Z^2\) into how many (co)sets? What is the fundamental volume \(V(D_2)\)? (2 pts)

It may be helpful to recall that any two-dimensional region’s volume scales by the determinant of a linear matrix that operates on vectors in that first region to create a second region.

d. What is the determinant of \(G_{D_2}\)? What is \(V(Z^2)\)? Validate your answers by comparing with Part c (2 pts)

e. Suppose instead of viewing \(Z^2\) as a two-dimensional real-integer vector space that the generation of lattice (constellation) points is viewed as a single complex symbol. How might the scalar complex scaled rotation implied by \(G_{D_2}\) or its equivalent action if the inputs are all possible complex integers (sometimes called Gaussian integers)? (2 pts)

A cartesian (or “cross”) product forms an ordered pair set of items where the first pair item is selected from a first set and paired with any item selected from the second set, and is written with the symbol \(\otimes\). For instance \(Z^4 \triangleq Z^2 \otimes Z^2\) to form 4-dimensional integer points (and \(Z^2 = Z \otimes Z\)). The diagram in Figure 2.3 shows a trellis description of the 4-dimensional code in its lower portion. A lattice generalization would allow the constellation points to extend to infinity in the pattern shown (so ignoring the constellation boundaries in the upper portion of the figure).

f. Using the notation \(\otimes\) for Cartesian product, \(\cup\) for union, and + for addition, write a single expression for the \(D_4\) lattice as the union of two 2-dimensional Cartesian products. (2 pts)

g. Find a set of 4 generator vectors for \(D_4\) and thus the matrix \(G_{D_4}\). (2 pts)

h. How many (co)sets of \(D_4\) when combined, will create \(Z^4\)? What is \(V(D_4)\)? (2 pts)

i. Find the determinant \(|G_{D_4}|\). (1 pt)

j. Find \(d_{\text{min}}\) and \(\gamma_f\) for \(D_4\) based on your generator. (2 pts)

k. Returning to the 24CR constellation of Figure 2.3, there were 256 possible codewords. Given a received vector \(y\), how many real calculations would the straightforward exhaustive ML decoder have to perform? A real calculation would be an addition, or subtraction/compare. Truncations don’t count. (1 pt)

l. Suppose instead the received 4-dimensional vector were broken into two 2D sub-symbol outputs and each were decoded separately for the closest points in first 2D sets \(D_2\) and \(D_2 + [0 1]\) and then again in the same way for the second 2D sets. Note these operations are basically simple 2D truncations plus a final single addition. What final operations would be needed to do a full ML decode? (This is a very simple example of what is often called sequential or Viterbi decoding, 2 pts).

m. For the 24CR situation, how might similar a simpler encoder be based on 2D encoding and some selection operations be performed? It would help in thought to divide the 24 point constellation’s even (or odd) points into 3 equal-size groups of 4 points each. (A sketch would be simplest way to respond.) (3 pts)

2.5 Design - Mapping 16 QAM into DMC’s – Subsection 2.2.2 - 19 pts

This problem explores the mapping of an AWGN channel into DMC’s from a code-design standpoint. An AWGN has SNR = 17.7 dB. For uncoded transmission, the system uses an 8SQ QAM constellation as a reference. Coded systems here will be restricted to use 16QAM with the same transmit energy per symbol.
a. Determine $P_e$ for each QAM system, 8SQ or $b = 3$ and 16SQ or $b = 4$. (2 pts)

This symbol error probability $P_e$ for the 16QAM constellation will be assumed to be the probability of bit error for first a BSC. A coded system uses one bit of redundancy per QAM subsymbol that is the sum of the other 3 bits, mapping then the extra 8 redundant points into a 2nd 8SQ that in union with the original 8SQ would be 16 QAM. The QAM constellation symbol rate is 1 MHz.

b. What are $N$, $\tilde{N}$ and $\overline{N}$ for the 16QAM system? (2 pts)

c. What is the redundancy for this QAM system per-dimension, per sub-symbol, and per codeword? (3 pts)

d. What is $d_{free}$ for the coded system? (1 pt)

e. Which system is better, coded or uncoded original 8SQ original? What is the data rate of the uncoded system. (1 pt)

Two successive QAM symbols now are viewed as one large symbol for which a hard decision is made on all 8 bits as byte or 256-possible-valued sub-symbol. An MDS code is applied. This MDS code (a byte-wide Reed Solomon for the expert reader) has $\overline{N} \leq 255$ total bytes per codeword. There are up to $P \leq 32$ parity bytes allowed. This problem will use only even-number values for parity and block length.

f. $\tilde{N}$ and $\overline{N}$ for this coded system? (2 pts)

g. What is the probability that either of the two sub-symbols contributing to one byte are in error (before the MDS decoder is applied) or equivalently that there is a byte error? (1 pt)

h. In order for the data rate to exceed or equal the original uncoded 8SQ system’s data rate, what is the maximum value of the ratio $P/N$? (1 pt)

The following enhanced version of matlab’s nchoosek command (which allows a vector in the lower entry, which matlab does not) may be helpful:

```matlab
function [outvec] = choosevec(N,vec);

% Computes a vector values of size vec (1 x something)
% N = top of choose
% vec = vector of values for choose
% created by J. Cioffi, 12/26/18
%-------------------------------------------------------------------
size = length(vec);
outvec = [];
for in=1:size
    outvec = [outvec, nchoosek(N,vec(in))];
end

Error messages of 8-digit accuracy limit on the choose commands can be ignored because precise integer accuracy is not necessary for the next part of this problem.

i. What is the minimum number of parity bytes that can be used to ensure that the coded system has better data rate and probability of error than the uncoded system? The answer may approximate probability of error by the first term in the probability of error expressions. Also find the corresponding $N$ value. (3 pts)

j. What is the maximum data rate that could be achieved with this code (MDS $q = 256$)? (2 pts)
2.6 Bounds’ Tightness for the DMC - Subsection 2.2.2 - 17 pts
This problem explores the Singleton and Hamming bounds in terms of their bounding of code rate as a function of free distance and codeword length.

a. Compute the Singleton Bound for a (binary) codeword length of $n = 10$ bits and a free distance of $d_{\text{free}} = 5$. What would be the code rate of such a bound if were achieved? Can the bound be achieved in this case? (2 pts).

b. Compute and compare the Hamming Bound for the same codeword and free distance as in Part a. (2 pts)

c. Comment on the tightness of the Singleton Bound in Part a in view of the (correct) answer for Part b? (1 pt)

d. Repeat Parts a - c for $n = 12$ with both $d_{\text{free}} = 1$ and then again for $d_{\text{free}} = 12$ and for 2. (2 pts).

e. A well-known binary Hamming code has $n = 7$, $d_{\text{free}} = 3$ and $k = 4$. What are the corresponding Singleton and Hamming Bounds? (2 pts)

f. What happens to the rate ($R = K/N$ or $r = k/n$) as inferred from the two bounds at fixed $d_{\text{free}}$ as $n$ becomes very large (for any $q$)? (1 pt)

These 4 parts are for $q = 16$, $K = 10$, $N = 12$.

g. What is the $d_{\text{free}}$ for an MDS code? (1 pt).

h. How many erred symbols can be corrected? (1 pt)

i. Compute the Singleton Bound for this code. (1 pt)

j. Compute the Hamming Bound for this code and compare to the Singleton Bound of Part i. (1 pt)

Now, this problem generalizes for any $q > 2$.

k. For any finite $d_{\text{free}}$ (and thus number of errors that can be corrected), what might be a strategy for coding on the DMC to ensure that a data rate arbitrarily close to $R = K/N \to 1$ can be achieved with arbitrarily small probability of symbol error? What might be the drawbacks of such an approach? (3 pts)

2.7 MDS Code – Subsection 2.2.2 and Subsubsection 2.4.5.1 - 11 pts
This problem explores an MDS code in terms of more than just free distance. Implementation of the formula in Equation (2.59) is provided here for copy and use by the student.

% --------------------------------------------------------------
% [Ni] = nearest(N,dh,idx,q);
% %
% Computes nearest neighbors for MDS codes
% An MDS code needs to exist for the program to work, but this existence is
% checked by negative codeword count and reset to zero with error message
% inputs:
%  N  = block length (presumed scalar)
%  dh  = free distance
%  idx = index of nearest neighbor (0 is Ne, 1 is N1, ...) - can be vector
%  q   = arithmetic base for the MDS code
% outputs
%  Ni  = Nearest neighbor count
% created by J. Cioffi, 12/25/18
% --------------------------------------------------------------
function [Ni] = nearest(N,dh,idx,q);
Suppose an MDS code with \( q = 4 \), \( N = 10 \) and \( P = 2 \) is used on a channel for which the inner probability of a byte error is 0.01. What is \( d_{\text{free}} \)? Is this a trivial code? (2 pt)

b. Compute \( N_i \) for \( i = 0, 1, 2, 3, 4 \). (1 pt)

c. Compute the first 5 terms in the probability of codeword error equation. (2 pts)

d. How do these terms compare, and does a single term well represent the overall probability of error if \( p = 10^{-5} \)? (1 pt)

e. A code with \( q = 16 \), \( N = 14 \), \( P = 4 \) is instead used. Now find the 5 largest contributions and comment similarly thereupon.

f. As \( q \) grows to say 256 (so bytes), unfortunately the matlab program provided starts to experience numerical-range problems. More careful analysis would try to match powers of the single-position error \( p_i \) to portions of the choose command to compute the correct \( P_e \) contributions without taking the product of a very large and a very small number, which is beyond the scope of this text. However, quantitatively predict what you would expect as \( N \) grows and as \( P \) grows for any particular \( N \). (3 pts).

2.8 Bandwidth Expansion, Hamming Codes, and Bit-Interleaved Coded Modulation - Subsection 2.2.2

This problem first investigates the exploitation of the AWGN’s essentially infinite bandwidth by noting that power \( P = \frac{E}{T} \) decreases linearly with increased symbol rate at fixed symbol energy \( E \), but power increases exponentially with integer numbers of bits \( b \in \mathbb{Z} \) at fixed symbol rate \( 1/T \). A 128SQ QAM constellation (See Section 1.3.4 and Problem 1.14) is initially used for uncoded QAM transmission on an ideal AWGN channel with symbol error probability \( P_e \). The ideal AWGN has no bandwidth (symbol rate) limitation, so the designer may decide any symbol rate that meets the performance objectives.
a. Is 128SQ a good design in terms of use of fixing any two quantities of the set \( \{ \text{power } P, \text{ data rate } R, \text{ error probability } P_e \} \) and comparing the other? Hint: The design could use another different smaller uncoded constellation with a different symbol rate. (1 pt)

b. What might be the power savings with fixed \( R \) and \( P_e \) with respect to the original 128SQ QAM? (1 pt)

c. By what factor would be the data-rate increase for the same \( P_e \) and \( P \) as the original 128SQ QAM? (1 pt)

d. Is a linear increase in bandwidth or power better in your answers above? Is this consistent with the capacity formula for the AWGN? (1 pt)

The examination of using wider bandwidth on the AWGN now continues at \( b < 1 \) with the use of codes. A Hamming binary code has generator

\[
G = \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1
\end{bmatrix}
\]

with \( \mathbf{v} = \mathbf{u}G \). This will imply a new sample clock for encoder output bits \( 1/T_{out} \) that is larger than the sample clock for input bits \( 1/T_{in} \). The channel will use BPSK on the encoder output bits. The symbol rate is \( 1/(k \cdot T_{in}) \).

e. Find \( n, k, r \) for this code (1 pt)

f. Find \( d_{free} \) (1 pt).

This code will be used with BPSK signaling on an AWGN and compared with uncoded transmission at the same data rate.

g. How much faster than the input bit rate does the output bit rate \( 1/T_{out} > 1/T_{in} \) need to be for the data rate of the code to remain the same as uncoded? (1 pt).

h. If the power \( \tilde{E}_x/T \) remains the same for the coded system, then by what amount does the new SNR decrease for the coded system at the same data rate as per Part b, presuming the AWGN power spectral density \( N_0/2 \) remains the same? (1 pt)

i. What is the coding gain of the coded system, including the effect of the \( d_{free} \) from Part f? (1 pt)

j. Under what condition does your answer in Parts d and continue to hold with coding and yet wider bandwidth exploitation of the AWGN? (1 pt)

k. What happens asymptotically to the SNR as the output sample clock for very good codes becomes very large? Also calculate the energy per bit \( E/b \). Hint: use the capacity formula for this. (2 pts)

Now suppose bandwidth is limited and the output bits \( \mathbf{v} \) from the encoder/generator are mapped with gray code on to the 128SQ constellation ideally so that closest points have only 1 bit difference, next closest points have 2 bit differences, with each \( \sqrt{2} \) factor increase in Euclidean distance corresponding to an additional bit difference, so that squared distance is linearly proportional to free distance. This is called Bit Interleaved Coded Modulation (BICM) if applied to a sequence of such 128SQ constellations with appropriate bit interleaving between the symbols.

l. Does the code work any better than uncoded on a single instance of a 128SQ constellation (no interleaving used)? (1 pt)
n. “Depth 3” successive 128SQ symbols correspond to an interleaving of encoder output bits from 3 successive uses of the encoder (so 3k input bits and correspondingly 3n output bits), where output bits are taken 1 at a time from the first symbol’s encoder, then 1 bit from the second symbol’s encoder, then from the 3rd, and then taking the 2nd bit from the first symbol’s encoder, and so. This repeats until 21 bits from 3 successive encoder uses are obtained and then applied to the 3 symbols’ constellations. The bits are correspondingly de-interleaved after ML decoding of the entire set of 3 symbols as 1 larger symbol. The bit mapping of the interleaved bits to the 128SQ uses an ideal Gray coding where closest constellation points differ in 1 bit, next closest in 2 bits, and next-to-next closest in 3 bit positions, so that input bits to the constellation mapping that differ in 3 places then would have 4 times the squared distance of those differing in only 1 place. What is the improvement in distance (the coding gain) over your answer in Part l? (3 pts)

n. Continuing to increase the depth to very large values, what would be the maximum gain? How does this compare to the coding gain found in Part j (2 pts)

This is a fairly general result with good binary codes and random interleaving of successive codewords - the gain of the binary code roughly passes to coded modulation. Effectively the concatenation of the constellation and good code corresponds to a good for the given repeated constellation. This effect is heavily used in wireless transmission systems.

2.9 Design with Erasures - Subsection 2.2.3 - 14 pts

For this problem, refer to Figures 2.8 and 2.11, and assume the parameter 0 < p < \frac{1}{2} in each is the same. The binary-input messages are equally likely.

a. Compute and compare the error probability for one-shot use of the BSC and the BEC. (1 pt)

b. If two successive channel uses repeat the same message to create a simple repeat code, now find the new BSC and BEC symbol-error probabilities. The BSC’s ML decoder “flips a coin” if two successive BSC outputs are different; the BEC’s ML decoder will flip a coin if two successive BEC erasures occur. (2 pts)

c. Is there a practical flaw (even slightly) in the answer to Part b that exaggerates the performance difference? Say what it is. (1 pt)

d. Compute the \textit{LLR}(y) for the one message bit in the one-shot use of Part a for each of the 3 BEC possible outputs. (1 pt)

e. Repeat Part d with one message bit, and the repeat code of Part b, but for the BSC (1 pt)

f. Based, on this problem so far, in what types of situations would the BEC be interesting as a model with respect to the BSC. In effect, what does an “erasure” tell the decoder to do? (1 pt)

The presence a BEC output erasure anywhere in a block of n BEC outputs corresponding to n independent BEC input bits could be viewed as “erasing the entire block,” particularly when there is no code creating a relationship between the successively transmitted bits, or possibly a very simple code that reliably detects only that an error was made somewhere within the block. Such reliable detection of an error, but not correction, is the subject of \textit{cyclic redundancy checks (CRCs)} in Chapter 10. In the context of this exercise, the knowledge that the message is correct is highly reliable as is also the opposite indication that the message may have contained erred bits. In this context, the BEC can be viewed as a higher-level channel that has two possible decisions made on its outputs: (1) correct receipt of message, that corresponds to the BEC’s zero or a one outputs, or (2) an error possibly occurred, the BEC’s erasure.

This concept is fundamental to what are known as \textit{retransmission methods} where an erasure causes the receiver to alert the transmitter through a feedback channel that the block had an error. The feedback channel may be slower or more reliable by design. The transmitter then resends that erred block. The receiver may assume responsibility for delaying, or buffering, correctly received packets (in which there was no erasure). The correctly received blocks’ information will contain the block’s sequence
number. The incorrect blocks will leave gaps in the sequence numbers, and those corresponding blocks need retransmission. Both the transmitter and the receiver will need retain (thus delay) blocks’ release until they know all sequence-indexed blocks to a certain time have been correctly received.

This problem continues by looking at some very basic retransmission concepts, also called Automatic Repeat reQuest (ARQ) systems, associated with this alternate type of coding that uses acknowledgements of correctly received blocks (called ACKS) and requests to retransmit (called NAKS), effectively introducing redundancy \( r < 1 \) and delay. The consequent reduced data rate is often called the throughput, although it is the same as this text’s information rate \( b \) in a block code context, because the designer might desire to call the bit-clock a “data rate” and thus the actual data rate is the “throughput.” Usually designers who do this are working at a level above the physical transmission layer that essentially supplies to that higher layer a “bit rate.” The higher-level designer will then use a code on this bit-rate as if it were the bit-clock rate \( 1/T' \), with a (possibly new) level of redundancy/coding applied.

g. If the BEC has probability \( p \), determine the error detect probability that there is at least 1 bit error in the block of \( n \) bits. Then, approximate this new block-error rate \( p' \) when \( p < 10^{-2} \). (Hint: the approximation should make good intuitive sense.) (1 pt)

h. What is the probability \( P_1 \) that the first transmission of a block is correct? What is the probability \( P_2 \) of exactly two attempts? Of exactly \( i \) attempts? Determine the average rate loss, and thereby derive the throughput \( R_{\text{tp}} \). (2 pts)

i. A real system will have a finite transmit buffer size of \( \Delta < \infty \) blocks, so the oldest messages corresponding to NAKs will be discarded in buffer overflow when the number of blocks to be transmitted exceeds \( \Delta \). Determine the minimum \( \Delta \) necessary so that overflow probability is less than \((p')^3\). (2 pts)

j. What is the total delay (in units of \( T' \) corresponding to your answer in Part i, including both transmitter and receiver delays. The answer can assume the transmission delay of both the channel and the feedback channel is negligible with respect to the block size. (2 pts)

2.10 D4 Entropy - Subsection 2.3.1 - 10 pts

Consider the D4 constellation of Example 2.1.3 with each of the 4-dimensional points being equally likely.

a. How many possible messages can be sent for one symbol? (1 pt)

b. What is the probability of each of these messages? (1 pt)

c. What is the 4-dimensional entropy \( H_x \) of this constellation? Also, then what are \( \tilde{H}_x \) and \( \bar{H}_x \) (1 pt)

d. What is the 2-dimensional entropy \( H_{\tilde{x}} \) of this constellation? Also, then what is \( \overline{H}_{\tilde{x}} \) (2 pts)

e. What is the 1-dimensional entropy \( H_x \) of this constellation (2 pts)

f. Compare the answers in Parts c to e to the corresponding maximum entropy in those dimensions. (3 pts)

2.11 Differential Entropy for Continuous Distributions of Interest - Subsection 2.3.1 - 9 pts

Find the differential entropy in bits of the random variable with the following probability density functions:

a. A continuous random variable \( x \) has uniform probability density function \( p_x(u) = \frac{1}{d} \) over the interval \( u \in [-d/2, d/2] \) (and zero elsewhere). What happens if \( d < 1 \)? Comment on differential entropy and continuous entropy difference. (1 pt)
b. A continuous random variable \( g \) with the exponential density \( p_g(u) = \frac{e^{-u/E_g}}{E_g} \forall g \geq 0 \). What happens as \( E_g \) gets small? (2 pts)

c. A continuous random variable \( a \) with Rayleigh distribution \( p_a(u) = \frac{u}{\overline{E_a}} \cdot e^{-u^2/2\overline{E_a}} \forall a \geq 0 \). Hint: Use that the definite integral \( \int_0^\infty e^{-x} \cdot \ln(x)dx = -\gamma = -0.577216 \) (The Euler Macheroni constant). (3 pts)

d. A continuous random variable \( x \) with lognormal distribution \( p_x(u) = \frac{1}{\sigma \sqrt{2\pi} \cdot e^{-\frac{(\ln(u)-\mu_x)^2}{2\sigma^2}}} \forall x \) real. (2 pts)

e. Can you think of a reason why these distributions’ entropy might be of interest? (1 pt)

2.12 Shaping via Encryption with Dither/random coding - Subsection 2.3.3 - 11 pts

Figure 2.68 has an AWGN channel for which a shaping lattice \( \Lambda_s \) is used with a good fundamental-gain code on its codewords with \( \tilde{N} \)-dimensional subsymbols \( \tilde{x} \). The \( x \) code has codewords with good separation and high fundamental gain so its performance without regard to shaping would be at or near the AWGN maximum. It’s subsymbols \( \tilde{x} \in C \) where \( C \) is a discrete constellation with \( |C| \) points selected from a lattice \( \Lambda \). This problem focuses on an approach to shaping codes that uses the shaping lattice \( \Lambda_s \)’s Voronoi Region \( \mathcal{V}(\Lambda_s) \) as the subsymbol constellation \( C \)’s outer Voronoi boundary \( \mathcal{V}_x \). This means that the union of the \( |C| \) constellation points’ Voronoi Regions have Voronoi Boundary equal to the Voronoi region of \( \Lambda_s \)

\[
\bigcup_{i=0}^{\lfloor |C| \rfloor} \mathcal{V}_i(\Lambda) = \mathcal{V}(\Lambda_s).
\]

(For this to happen, a sufficient condition is that \( \Lambda_s \) be a sub-lattice of \( \Lambda \).)

The “dither” sequence \( \tilde{\delta} \) is fixed and known to transmitter and receiver, but in any given subsymbol \( \tilde{x} \) it was generated by randomly selecting the subsymbol from a continuous distribution with the Voronoi Boundary region \( \mathcal{V}_x = \mathcal{V}(\Lambda_s) \). Such a sequence is sometimes known in cryptography as a key; this sequence’s generation may be complicated depending on the region; \( \tilde{\delta} \)’s generation is beyond this text’s scope (and is done only once before transmission begins and shared with transmitter and receiver). The Voronoi Boundary’s volume is given as \( V(\Lambda_s) \) with minimum energy \( \mathcal{E}_{\Lambda_s} \) corresponding to zero mean. Both volume and energy are a function of the lattice chosen. When \( \Lambda_s = Z^{\tilde{N}} \), then \( V(Z^{\tilde{N}}) = 1 \) and \( \mathcal{E}_{\Lambda_s} = 1/12 \).

![Figure 2.68: Dither Shaper.](image)

As in Figure 2.68’s lower portion, the modulo-\( \Lambda_s \) device treats its continuously distributed input as a kind of “received signal input” to a decoder (even though it’s located in the transmitter). That ML
decoder finds the closest point in the lattice $\Lambda_s$ to that dithered signal $X$ and then finds the difference $\tilde{X}$ between the ML-detector’s choice and the input. The modulo-$\Lambda_s$ (sometimes called a Voronoi Shaping Code Encoder. Clearly $\tilde{X} \in V(\Lambda_s)$. $\tilde{X}$ has mean value equal to the origin (all zeros). The modulo of any point $\tilde{z}$ is abbreviated $(\tilde{z})_{\Lambda_s}$. Application of modulo to the sum of two values (real or complex) is abbreviated $\mod_{\Lambda_s}(z_1 + z_2) = z_1 \oplus_{\Lambda_s} z_2$.

a. Show that $z_1 \oplus_{\Lambda_s} z_2 = (z_1)_{\Lambda_s} \oplus_{\Lambda_s} (z_2)_{\Lambda_s}$. (2 pts)

b. Show that the transmitted sequence $\tilde{X}$ and the input subsymbol sequence $\tilde{x}$ are independent. (This independence essentially renders the message unrecognizable by any observer of the link, this securing transmission from eavesdropping\textsuperscript{84}.) (1 pt)

c. Show that the transmitted sequence $\tilde{X}$ is uniform continuous in the Voronoi Region $V(\Lambda_s)$. (1 pt)

d. Compute this code’s shaping gain as a function of the volume of $V(\Lambda_s)$ and $\mathcal{E}_{\Lambda_s}$ using a hypercube using the continuous approximation. (1 pt)

The receiver adds the negative of the known dither sequence and then that sequence undergoes a second identical modulo-$\Lambda_s$ device. Such a system is also known as writing on dirty paper in information theory because the sequence $\tilde{x}$ is written on top of “dirt” or existing writing that is the dither sequence.

e. If the noise has $\sigma^2 = 0$, show that $z = \tilde{x}$ exactly. (2 pts)

f. Find a upper bound on the mutual information $I$ for this system (presuming the code has highest possible fundamental performance and only shaping is of interest) by showing that (3 pts)

$$I_{x,y} \leq \bar{C} - \log_2 \left( 2\pi e \frac{\mathcal{E}_x}{|V^2/N(\Lambda_s)|} \right).$$

g. Evaluate your bound for an AWGN with known SNR when $\Lambda_s = Z_N^2$ if $x$ is a very good code. (1 pt).

\textbf{2.13 AEP Code Revelations for the Symmetric DMC - Sections 2.2 and 2.3 - 11 pts}

The AEP generalizes the AWGN-channel’s sphere-packing capacity arguments where a typical set replaces the hypersphere and set size replaces hypersphere volume. This problem explores what this means for a finite-field channel, particularly looking at the AEP’s interpretation in terms of the Hamming and Singleton Bounds.

a. Use differential entropy $\mathcal{H}_{\tilde{X}}$ to relate an AWGN’s hypersphere squared radius (average 2D energy) $\mathcal{E}_{\tilde{X}}$ to the size of the size of the typical set $|A_{\tilde{X}}|$ for large $n$. (1 pt)

b. Using Part a’s finding, similarly interpret the size of the conditional typical set $|A_{\tilde{X}/y}|$. (1 pt)

c. Divide the set-size answer for part a by the answer to Part b and interpret in terms of number of codewords and capacity. (1 pt)

d. Where are most (nearly all) the points in the typical sets located for the Gaussian distribution? (1 pt)

Given the sphere arguments and findings, this problem progresses to view finite-field balls as spheres

e. Repeat Part a, now with entropy replacing differential entropy, for a channel input with $q$ subsymbol possibilities into a symmetric DMC that is $q \times q$ for each subsymbol of a code. (1 pt)

\textsuperscript{84}Of course no dither sequence or key will ever be perfect and so this method will not absolutely protect transmission in practice.
f. Review the Hamming Bound of 2.2.4 and now repeat Parts b and c and try to approximate the set-size and the mutual information codeword count. (1 pt)

g. Relate the answer in Part f to the capacity expression for the symmetric DMC. (1 pt)

h. Pose a reason why codes meeting the Hamming bound are called “perfect codes.” (1 pt)

i. MDS codes can be used to drive $P_e$ to zero with increasing block length (which implies larger $q$ also). Can the Singleton Bound be similarly used to characterize the AEP and capacity? Why or Why not? (2 pts)

j. As block length increases and the AEP results become increasingly accurate, what code type would be expected to be better and why? (1 pt)

2.14 Capacity in bits/sec - Subsection 2.3.4 - 7 pts
An AWGN channel has SNR=20 dB when the symbol rate is $1/T = 1$ MHz for PAM transmission.

a. What is the capacity in bits/dimension? (1 pt)

b. What is the capacity in bits/second? (1 pt)

c. What is the capacity at any symbol rate $a/T$ where $a > 0$? (2 pts)

d. Find the max bit rate in Mbps that can be transmitted on this channel (you may vary $a < 100$)? (2 pts)

e. Comment on your answer in Part d as to the practicality of a variable $a$? (1 pt)

2.15 Capacity Calculation - Section 2.3.4 - 10 pts

An AWGN has an input sequence $x$ with transmitted flat two-sided power spectral density $\tilde{E}_x = -60$ dBm/Hz and white-noise power spectral density $\frac{N_0}{2} = -140$ dBm/Hz. (dBm means $10\log_{10}(S)$ where $S$ is the random process' flat power spectral density level in mW/Hz and mW means 1 milli-Watt or $10^{-3}$ Watts.)

a. Show that the energy/real-dimension $\tilde{E}_x$ for PAM and QAM is equal to the two-sided power spectral density if that power spectral density is flat. Comment on the flat one-sided power spectral density for QAM and its relationship to energy/subsymbol. (2 pts)

The transmission channel is simple pure attenuation of all frequencies by 53 dB, that is $H = 10^{-5.3}$.

b. What is the SNR for PAM transmission on this channel? For QAM transmission? (2 pts)

c. What is the capacity in bits/(real)dimension? (1 pt)

d. What is the minimum amount of transmit power necessary to send 18 Mbps over this channel if a code that is based on QAM sub-symbols with subsymbol rate of $1/T = 2$ MHz? And, if based on PAM with subsymbol rate 4 MHz? (2 pts) (assume long codewords are used).

e. If the capacity is known to be 72 Mbps, and a code with $\Gamma = 5.7$ dB is used, What is the subsymbol rate for PAM at the $P_e$ of this gap? For QAM? What is the data rate? (2 pts)

f. (extra 1 pt) - Guess what type of system the QAM version might be?

2.16 Gap and Margin - 11 pts

a. TBD
2.17 Channel Capacity for the BSC and the BEC - 12 pts

The BSC and BEC are discussed in Subsection 2.3.2.

a. Graph the channel capacity of the binary symmetric channel and of the binary erasure channel as a function of the bit error probability $p$ over the range for $p$ of 0 to 1. (2 pts)

b. Explain intuitively why the capacity of the binary symmetric channel decreases monotonically from 1 to (for $p = 0$) to 0 (for $p = .5$) and then increases back to 1 for $p = 1$. (2 pts)

c. In contrast to the BSC, the capacity of the BEC decreases monotonically from 1 (for $p = 0$) to 0 (for $p = 1$). Explain why this is the case. (1 pts)

d. Find the capacity of an AWGN with SNR of 10 dB. (1 pt)

e. Find $P_b$ for binary PAM transmission on the channel of part (d). (2 pts)

f. Letting $p$ for a BSC be equal to the $P_b$ you found in part (e), find the capacity of this BSC. (2 pts)

g. Compare the capacity in part (f) with the capacity in part (d). Why are they different? (2 pts)

2.18 Universal DMC - 5 pts

Inputs to and outputs from a DMC are presumed to be any one of 256 possible messages and the channel probabilities are given by

$$p(i/j) = \begin{cases} \frac{p_s}{255} & \forall i \neq j \\ 1-p_s & i = j \end{cases}$$

(2.547)

a. Find the input distribution that achieves capacity for this channel. (1 pt)

b. Find the capacity. (2 pts)

c. What is the capacity as $p_s \to 0$? (1 pt)

d. Why might this channel be of interest? (1 pt)

2.19 new problem

2.20 MMSE and Entropy - 14 pts

A matrix AWGN has (real baseband)

$$H = \begin{bmatrix} 10 & 4 \\ 3 & 1 \end{bmatrix}.$$ 

(2.548)

The input vector of users is $x$ and the output vector is $y$, while the noise is $n$, with $R_{nn} = I$ and $R_{xx} = I$, with $L_y = 2$ and $L_x = 1$. The input and noise are stationary and Gaussian. The user inputs follow good codes with $\Gamma = 0$ dB that involves an infinite number of (on-average in AEP sense) independent channel uses.

a. How many input dimensions are there? (1 pt)

b. Find the MMSE estimator $W$ for $x$ given the channel output $y$. (1 pt)

c. Find the MMSE estimator for $y$ given the channel input $x$. (1 pt)

d. What type of decoder satisfies (1 pt)

$$\hat{x} = \arg \min_x \left\{ \sum_{n=-\infty}^{\infty} \|y - H \cdot \hat{x}\|^2 \right\}$$

383
e. What type of estimator satisfies (1pt)

\[ \hat{x} = \arg \min_{\hat{x}} \left\{ \sum_{n=-\infty}^{\infty} \| \hat{x} - W \cdot y \|^2 \right\} \]

In the asymptotic (law of large numbers or AEP) sense, what type of detector is this?

f. Compute \( \hat{H}_{x/y} \) and \( \hat{H}_{y/x} \). Compare them. (2 pts)

g. Suppose that Part b’s \( W \) multiplies Part c’s \( \hat{y} \), instead of \( y \). What does \( W \cdot \hat{y} \) estimate? Similarly, what does \( H \cdot \hat{x} \) estimate? What does this mean intuitively? (3 pts)

h. Write the SNR in two forms using only \( H \) and \( W \). (2 pts)

i. What causes ML and MAP estimates to be the same? Is that applicable in this problem? (2 pts)

2.21 Non-Zero Gaps to be added

2.22 MMSE Estimation (8 pts)

Two zero-mean complex Gaussian random variables have probability distributions \( p_x \) and \( p_y \) with joint probability distribution \( p_{x,y} \) and nonsingular autocorrelation matrix

\[ R_{x,y} = \begin{bmatrix} \mathcal{E}_x & R_{xy} \\ R_{xy}^* & \mathcal{E}_y \end{bmatrix} \]

The minimum mean-square estimate of \( x \) given \( y \) has variance \( \sigma_{x/y}^2 \). The orthogonality principle of Appendix D may be useful throughout.

a. (1 pt) Find \( \sigma_{x/y}^2 \) in terms of \( \mathcal{E}_x, R_{xy}, \) and \( \mathcal{E}_y \).

b. (1 pt) Relate the conditional entropy \( H_{x/y} \) to \( \sigma_{x/y}^2 \) and therefore to the MMSE estimate.

c. (2 pts) Interchange the roles of \( x \) and \( y \) in the results in Parts ?? and ?? and compare the SNRs for the two results.

d. (1 pt) Relate the mutual information to the SNR of Part c.

e. (3 pts) Suppose \( y \) becomes a complex vector \( y \) but \( x \) remains a scalar with \( y = \begin{bmatrix} 1 \\ 1 \end{bmatrix} x + n \) where

\( n \) has independent zero-mean Gaussian components all with variance \( \sigma^2 \). Does your answer to part d change?

2.23 Multiuser Channel Types - 8 pts

This problem provides several multi-user channels for identification of type (e.g., MAC, BC, ....)

a. Identify Figure 2.69's multi-user channel type in both “downstream” (towards customer, 1480 nm) and “upstream” (opposite 1310 nm) directions. What is \( U \) in either direction? (3 pts)

This network uses fiber and has the name “passive optical network” or PON because the yellow-colored triangular devices are passive and simply couple the fiber on the triangle vertex to both fibers on the side opposite the vertex with optical materials that split the wavelength in two downstream direction and combine them in the upstream direction. OLT is optical line terminal and typically at a network edge owned by the internet service provider, while the ONT is the optical network terminal and is at the edge owned by the consumer of internet service.
b. Identify Figure 2.70’s channel type when transmitting in one direction only, and when transmitting in two direction simultaneously. What is $U$ in each case? (3 pts)

This ethernet-cable channel has 8 wires used to transmit differentially (a connection uses two wires, one of the twisted pairs, to transmit a voltage between the two wires). There is a common transmitter and a common receiver that connect through the RJ45 connectors shown.

c. You and your neighbor’s Wi-Fi coverage overlap. You both use the same (unlicensed) channel to transmit only to your respective laptops via the Wi-Fi connection. You both listen and only transmit when the other is silent. What kind of multi-user channel is this? How many users are there? (2 pts).

### 2.24 Mutual-Information Vector - 6 pts

This problem determines the minimum information vector for a multi-user channel for the orders and information-like

<table>
<thead>
<tr>
<th>rcvr/ user</th>
<th>$\pi_5$</th>
<th>$\pi_4$</th>
<th>$\pi_3$</th>
<th>$\pi_2$</th>
<th>$\pi_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>top</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>bottom</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>$P_u(\pi_u)$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>rcvr/ User 3</th>
<th>$\pi_5$</th>
<th>$\pi_4$</th>
<th>$\pi_3$</th>
<th>$\pi_2$</th>
<th>$\pi_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>top</td>
<td>10</td>
<td>$\infty$</td>
<td>5</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>7</td>
<td>8</td>
<td>4</td>
<td>$\infty$</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>bottom</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

| $P_u(\pi_u)$ |        |        |        |        |        |

Table 2.9: Tables for Problem 2.24.

a. How many users are there? (1 pt)
b. Find the values of $P_u(\pi_u)$ for each of the users. (2.5 pts)
c. Find the minimum information vector for this channel. (2.5 pts).

2.25 Time-Division Multiplexed Multiuser Channel - 12 pts
A time-division multiplexed (TDM) channel has $U = 3$ users that use the same single dimension at different times. Only one user may be present at any time on this AWGN. A single receiver accepts transmission and decodes whatever user is present before forwarding it to the appropriate user’s destination. The maximum energy on that dimension is fixed at $E_x$, with noise power spectral density $\sigma^2$ of the same dimensional measure as the energy.

a. Is this channel an energy-sum MAC? (1 pt)
b. Write an expression for the maximum rate sum. (2 pts)
c. How many faces has $C(b)$? (2 pts).
d. How many vertices has $C(b)$? (1 pt).
e. Write an expression for the face that has largest rate sum. (2 pts)
f. What geometric shape is the capacity region? (1 pt)
g. Repeat Part f in general for $U > 3$. (1 pt)
h. Suppose the channel were such that all users transmit simultaneously, but each on its own carrier frequency and subsymbol rates equal for all users. What happens to the capacity region? (2 pts)

2.26 Vector Gaussian MAC - 20 pts
A real-subsymbol-only vector Gaussian MAC has

$$H = \begin{bmatrix} 5 & 2 & 1 \\ 3 & 1 & 1 \end{bmatrix}$$

with Gaussian noise autocorrelation

$$R_{nn} = \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix} \cdot \begin{bmatrix} \frac{9}{256} & 0 & 0 \\ 0 & \frac{1}{16} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \cdot \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}.$$  

Input energy follows from

$$R_{xx} = 2 \cdot I.$$  

Hint: Parts e, f, and h may solve easily using Cholesky factorization of $R_{-1}$ and using the diagonal elements of the corresponding $S_0$, rather than attempting to enumerate solutions by the Chain Rule.

a. Find the maximum number users $U$ there could be? Assume this is the number of users for the rest of this problem. (1 pt)
b. Find $R_{nn}$, its square root, and the noise-equivalent channel using the eigenvalue-based square root implied in the problem statement. (2 pts)
c. Determine if the channel is degraded and find $u^o$ and $u^s$ (assuming the conventional user indexing of highest number MAC user at top/left). Find possible user orders that would achieve maximum rate-sum on an energy-sum MAC version of this channel. (4 pts).
d. Find the largest rate sum for the given input $R_{xx}^{1/2}$. (1 pt)
e. Find the 3 two-dimensional pentagon regions that correspond to $E_3 = 0$, $E_2 = 0$, and $E_1 = 0$ respectively. (3 pts)
f. Find the 6 vertices when $E_u = 2$, $\forall u = 1, 2, 3$ that characterize the maximum rate-sum plane boundary of $A(b, R_{xx})$. (6 pts)

g. Is the point $b^* = [b_3 \ b_2 \ b_1] = [0.1 \ 2.4 \ 3.2]$ in the achievable region? (2 pts)

h. Relax the energy constraint to $\text{trace}(R_{xx}) \leq 6$ and find the maximum rate sum. (1 pt)

---

**2.27 Vestigially Symmetric Continuous MAC rate sum - 12 pts**

(See Section 2.5 end example.) Figure 2.71’s 2-user continuous-time Gaussian MAC has two channel responses

\[
\begin{align*}
x_2(t) &= \text{sinc}(t) + a \cdot \text{sinc}(t - 1) \\
x_1(t) &= \text{sinc}(t) - a \cdot \text{sinc}(t - 1)
\end{align*}
\]

\[n(t) \sim \sigma^2\]

Figure 2.71: Simultaneous Water Filling for Problem ??.

with AWGN with (two-sided) power spectral density $\sigma^2$ and $0 \leq a \leq 1$, with power $P_1 = P_2 = 1$. The integral

\[
\int \frac{df}{a + b \cdot \cos(2\pi f)} = \frac{1}{\pi \sqrt{a^2 - b^2}} \cdot \arctan \left[ \frac{a - b}{a + b} \cdot \tan(\pi f) \right] + \text{constant}
\]

may be useful.

a. Sketch the frequency responses of the two users’ channels $|H_2(f)|$ and $|H_1(f)|$ on the same graph. (2pts)

b. Describe the frequency-division SWF (simultaneous water fill) solution’s spectra in terms of a given water level $\lambda$ for any value of $a$ and $\sigma^2$. What happens as $a \to 0$? (2 pts)

c. Are there other SWF solutions? If so describe them. (2 pts)

d. Find the rate sum and two data rates for the FDM solution if $a = 0.9$ and $\sigma^2 = .181$. (Hint: save time by reviewing the example in Section 2.5 that will have similar integrals) (4 pts)

e. What would happen to the solution if the power was only a sum constraint adding to 2? (1 pt)

f. Suppose now that an FDM solution is to be maintained, but the power levels are $P_2 = 1.25$ and $P_1 = .75$. Describe roughly what you would expect to happen to the FDM SWF solution and the rate sum? (You need not compute the new rate sum, just say if it increases or decreases and what happens to the two users’ spectra.) (1 pt)

---

**2.28 Modulo Precoder - 12 pts**

The precoder modulo can be viewed as a ML decoder where the output is the error vector between the closest point and a lattice point. The scalar integer lattice is the set of integers $\mathbb{Z}$, while the set of integer ordered pairs is lattice $\Lambda = \mathbb{Z}^2$. The lattice of all even integer pairs is $2\mathbb{Z}^2$. 

387
a. Find the result of the following operations for \( (v)_{\mathbb{R}^2} \): (4 pts)

(i) \( v = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \)

(ii) \( v = \begin{bmatrix} 4.5 \\ 1 \end{bmatrix} \)

(iii) \( v = \begin{bmatrix} 4.2 \\ -4.2 \end{bmatrix} \)

(iv) \( v = \begin{bmatrix} -17 \\ -21 \end{bmatrix} \)

This is the **rectangular lattice** (scaled by 4).

b. Using the generators for the **hexagonal lattice** \( \Lambda = A_2 \), \( g_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \) and \( g_2 = \begin{bmatrix} \sqrt{3} \\ 1 \end{bmatrix} \) so that any \( \lambda \in A_2 \) can be written as \( \lambda = z_1 \cdot g_1 + z_2 \cdot g_2 \) where \( z_1 \in \mathbb{Z} \) and \( z_2 \in \mathbb{Z} \), find \( (v)_{A_2} \): (4 pts)

(i) \( v = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} \)

(ii) \( v = \begin{bmatrix} 2.1 \\ 0 \end{bmatrix} \)

(iii) \( v = \begin{bmatrix} 0 \\ -2.1 \end{bmatrix} \)

(iv) \( v = \begin{bmatrix} \sqrt{3}/2 + \epsilon \\ 1/2 + \epsilon \end{bmatrix} \) with \( 0 < \epsilon < 0.5 \).

c. The **face-centered cubic** \( D_3 \) lattice has \( \lambda \in \Lambda \) such that \( \lambda = z_1 \cdot g_1 + z_2 \cdot g_2 + z_3 g_3 = G \cdot z \), with \( z \in \mathbb{Z}^3 \) where: (4 pts)

\[
G = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ \end{bmatrix}
\]

(i) \( v = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \)

(ii) \( v = \begin{bmatrix} 2.1 \\ 1 \\ 1 \end{bmatrix} \)

(iii) \( v = \begin{bmatrix} -1 \\ -1.1 \\ -2.1 \end{bmatrix} \)

(iv) \( v = \begin{bmatrix} 5 \\ 4 \\ 5 \end{bmatrix} \) with .

---

**2.29 Scalar BC - 8 pts** A scalar real Gaussian BC has

\[
H = \begin{bmatrix} -2 \\ 1 \end{bmatrix}
\]

with noise per dimension \( \sigma^2 = .01 \), and \( \bar{\xi}_x = 2 = \bar{\xi}_1 + \bar{\xi}_2 \).

a. Show the two receivers for successive decoding. (1 pts)
2.30 Vector Gaussian BC - 14 pts

A vector baseband-complex Gaussian BC has

$$H = \begin{bmatrix} 8 & 2 & 6 \\ 4 & 2 & 3 \\ 2 & 2 & 1.5 \end{bmatrix}$$

with noise per dimension $\sigma^2 = .01$, and $\mathcal{E}_x = \mathcal{E}_1 + \mathcal{E}_2 + \mathcal{E}_3$.

a. Find the noise-whitened channel, and then find the primary and secondary user sets $u^o$ and $u^s$ respectively. (2 pts)

b. Find the worst-case noise for this channel when all input energies are equal, and confirm your answer in Part a. (2 pts)

c. Try any other autocorrelation matrix $R_{xx}$. Does this change $R_{wcn}$? Does it change the set of primary and secondary users? (2 pts)

The users should now be re-ordered optimally for the remainder of this problem.

d. Find $\mathcal{C}_{BC}(b)$ with removal of the secondary users, now number 3 ($\mathcal{E}_3 = 0$). [Hint, simpler solution may arise from use of the worst-case noise program], (2 pts)

e. Design a precoder (using the new order) for $R_{xx} = \text{Diag}(1 1 4)$. For user 3, allow its energy to be $4/5$ for each of the primary dimensions. Verify that the new rate sum is less than the maximum for this part’s $R_{xx}$. (5 pts)

f. Design the decoder for the solution in Part e. (1 pts)

2.31 Mixed-Dimensional IC - 11 pts

A vector baseband-complex Gaussian IC has

$$H = \begin{bmatrix} 8 & 2 & 6 \\ 4 & 2 & 3 \\ 2 & 1 & 1.5 \end{bmatrix}$$

User 1 (corresponding to the lower 2 rows and leftmost columns for this IC is two-dimensional complex (4 real dimensions), while user 2 is one-dimensional complex (two real dimensions). The noise per dimension $\sigma^2 = .01$ and independent in all dimensions, and $\mathcal{E}_x = \mathcal{E}_1 + \mathcal{E}_2$, and $\mathcal{E}_1 = 1$.

a. Find the possible orders to search (1 pt)

b. Reorganize the $H \rightarrow \sigma^{-1} \cdot H$ matrix so that user 2 is at the top as per convention. Also find representations for the two-dimensional user 2 in terms of all inputs, including signal part as a matrix, crosstalk term as noise where appropriate and the AWGN. Do this also for user 1. User 2 has a two-dimensional channel - can you simplify it to a single input-dimension of energy $\mathcal{E}_2$ without loss of any data rate? If so, do so. (4 pts)

c. What well-known principle would you use for a 2-dimensional IC that does not simplify as in Part b? (1 pt)

d. Find this IC’s two non-trivial vertices. (2 pts)

e. Sketch $\mathcal{C}_{IC}(b)$ by finding points. Show the max rate-sum point and slope of tangent at that point (3 pts)
2.32 Simple Relay Channel - 9 pts A bonded relay channel has one transmitter that transmits to both of two relay devices with gains upper $g_2 = 4$ and lower $g_1 = 3$ that decode and then forward the resultant messages over a second channel to a single receiver. All signals are complex baseband. The input has $E1$ as are also the transmit energy limits for each of the two relays. The noise on all channels is unit variance additive white Gaussian that is independent of other noises. The two relays are synchronized but otherwise cannot coordinate other than in overall strategy. The corresponding second-stage gains are upper 3 and lower 4 respectively. Each transmitter may transmit one unit of energy per subsymbol.

a. What kind of channel is the first stage? How many sub-users does it have? (1 pt)
b. What kind of channel is the second stage? How many does it have? (1 pt)
c. What is the best strategy for this system’s overall data rate? Find simple bounds that deviate by no more than 15% from best. (1 pt)
d. Design the first stage. (2 pts)
e. Design the second stage. (2 pts)
f. What is the maximum data rate? (1 pts)
g. Find a margin equivalent of your result in Part d that any extra data rate might be used for redundancy with real code against extra noise in the second stage. (1 pt)
Bibliography


Index

admissibility, 278

ball packing, 219
binomial theorem, 219
bit
  matrix, 368
bit vector, 261
bits per subsymbol, 204
bits/subsymbol
  vector, 282
bits/symbol
  per-user, 262
  vector, 261

canonical
  backward, 295
canical
  forward, 295
capacity, 241
  region, 203, 261
  theorem, 242
capacity region
  multiple-access channel, 285
  relay-channel, 370
chain rule
  multi-user, 283
  mutual information, 236
channel
  broadcast, 203, 264
    Gaussian, 313
crosstalk-free, 274
degraded, 280
interference, 203, 265
mesh, 260
multiple-access, 203, 263
relay, 260, 367, 368
  vector-broadcast, 264
code, 202, 204, 208
  block, 218
    linear, 220
  convolutional, 227
cyclic, 253
generator, 209
lattice, 215
LDPC, 254
  maximum-distance separable, 219
  rate, 218
  sliding-block, 226
trellis, 226
codeword, 202, 204, 208
coding, 202
  rancom, 237
correlation
  16HEX, 205
  16SQ, 205
  subsymbol, 204
crypto lemma, 315
cubic closed packed, 205
cyclic redundancy check, 226
data rate
  vector, 261
decision region
  Venn Diagram, 240
decodable set, 269
    best, 269
decoder
  successive, 284
detection
  successive, 284
duality
  BC, 348
dependent
  energy per bit, 253
dependent energy per subsymbol, 204
dependent entropy, 230
  conditional, 234, 242
differential, 232, 233
  maximum, 230, 232
  vector, 235
Equipartition
  Asymptotic, 237
erasure, 225
face-centered cubic, 205
Gilbert-Varshamov