Coding Basics
and multiple users

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 ($V_{\boldsymbol{x}}^{1/N}$) and energy ($\bar{\mathcal{E}}_{\boldsymbol{x}}$) as dimensionality increases. As such, code study here conveniently sits in the limiting sense of $N \to \infty$, so the codes aggregate temporal dimensions into symbols with $N > 1$. MIMO systems may also use such codes over spatial dimension(s) if $1 < L_x < \infty$. However, adding $L_x > 1$ MIMO to coding results complicates study with little benefit, so this chapter's single-user code description (Sections 2.1 - 2.5) sets $L_x = L_y = 1$.

For any channel $p_{\boldsymbol{y}/\boldsymbol{x}}$, an encoder maps mutually exclusive input messages to a certain asymptotic equally probable (uniform) channel-distribution-based decision regions. Codes often use the same, or closely related, **subsymbol** constellation(s) repeatedly to form an N-dimensional **codeword** comprised of $\tilde{N}$-dimensional subsymbols. This codeword is also a multi-dimensional symbol. The set of all such possible codewords (symbols) is the **code**. Good codes need a certain minimum redundancy[1] over "extra" subsymbol constellation points. This chapter focuses on both data-rate bounding and basic-design outlines, which find later use in the system designs of Chapters 3 - Chapter 7. Chapters 8 - 11 more completely detail specific code designs and also enumerate many popular codes designs, their parametrization, and their decoders. Chapter 8 focuses on ML/MAP sequential and iterative decoders. More complete code listings appear later in Chapters 9 (block and lattice codes), 10 (convolutional and trellis codes), and 11 (concatenated codes).

Single-User Coding Sections: This chapter's first half of this chapter addresses single-user 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. A codes subsymbol sequences map input messages to a sequence of constellation C points. Each subsymbol is a point within C. The constellation C 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 coded systems.

Section 2.3 introduces **entropy** that characterizes any probability distribution. Entropy 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 capac-

[1] Redundancy generalizes the well-known concept of parity bits.

ity, which is the maximum possible bits/symbol that can be reliably transmitted. Section 2.3 thereby introduces Shannon's famous capacity result as a data-rate bound on best codes' performance as dimensionality becomes infinite. Section 2.3 also relates AWGN channels' minimum mean-square error estimation to entropy and mutual information. Section 2.3's **Asymptotic Equal Partition (AEP)** generalizes 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-coded parallel channels to a best variable **water-filling** energy/dimension strategy. While capacity applies to a general channel model $p_{\boldsymbol{y}/\boldsymbol{x}}$, this chapter mainly focuses upon the AWGN and binary symmetric channels.

Section 2.4 revisits Chapter 1's gap concept to measure AWGN-channel code strength relative to Shannon's capacity, as well as revisits a few simple DMC capacities. Some simple design examples then also appear in Section 2.4.

Section 2.5 progresses to some simple codes for both a single AWGN and for coordinated coding of an otherwise independent set of such AWGNs. This parallel-channel set finds use on filtered AWGN's, expanding vector coding to its infinite-dimensional "multi-tone" limit.

Multiuser Codes: 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 multi-user channels: the **multiple-access channel (MAC)**, the **broadcast channel (BC)**, and the **interference channel (IC)** that each have different levels and locations of encoding or decoding collaboration between users. Section 2.6 also expands Section 2.3's single-user capacity to a **capacity region**. Multiuser detectors also first appear in Section 2.6, which leads to prioritization, or **user order**, and the related concept of reliably decodable 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 2.6 also expands Section 2.4's gap to multiuser channels.

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** component and possibly **secondary-user** components and/or other primary components 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 and finds the **multi-user simultaneous-water-filling** energy allocation that characterizes the **MAC's maximum rate sum**.

Section 2.8 progresses to the Gaussian BC, essentially finding a dual approach to the MAC where again primary- and secondary-user components appear. The concept of **worst-case noise** also appears to simplify the BC capacity-region construction. Section 2.8 alternately also introduces scalar duality to simplify BC capacity-region construction, while deferring more complex vector duality to Chapter 5. Section 2.8 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 Gaussian capacity-region calculation.

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. This section introduces the concept of multiuser channel nesting to describe more complex multiuser channels in basic-multiuser-channel-combination terms.

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 2.1.1 also defines redundancy. Subsection 2.1.2 then progresses to formal definitions. 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 $\boldsymbol{x}_i$, $i = 0, ..., M-1$, over N dimensions. When the channel input allows real symbol inputs, then $\boldsymbol{x} \in \mathbb{R}^N$, and the average energy is $\mathcal{E}_{\boldsymbol{x}}$ with $\bar{\mathcal{E}}_{\boldsymbol{x}} = \mathcal{E}_{\boldsymbol{x}}/N$ as the average energy per dimension, and similarly the bits per dimension is $\bar{b} = b/N$.

Codewords and Subsymbols: 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 that use multiple repeated instances of Chapter 1's symbol constellations. All members of this symbol/code set $\{\boldsymbol{x}_i\}_{i=0,...,M-1}$ 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 $\overline{N}$ **subsymbols** $\tilde{\boldsymbol{x}}_n$, $n = 1, ..., \overline{N}$ to form an $\bar{N}$-dimensional symbol-vector codeword

$$\boldsymbol{x} = \begin{bmatrix} \tilde{\boldsymbol{x}}_1 \\ \vdots \\ \tilde{\boldsymbol{x}}_{\bar{N}} \end{bmatrix} \quad .$$

The subsymbols each have $\widetilde{N}$-dimensions[2]. The symbol period is $T = \bar{N} \cdot \widetilde{T}$, where $\widetilde{T}$ is the **subsymbol period**, so $\widetilde{T} \leq T$, and the **subsymbol rate** is then $1/\widetilde{T} \geq 1/T$. $\bar{N}$ never includes any spatial dimensionality, and counts the number of subsymbols in a codeword, MIMO or not. The codewords select subsymbols from an $\widetilde{N}$-dimensional **subsymbol constellation** C that has $|C|$ possible subsymbol-vector possibilities. Simple constellations may contain $\tilde{\boldsymbol{x}} \in \mathbb{C}$, so two real dimensions as in Chapter 1's QAM. In this case, $\boldsymbol{x} \in \mathbb{C}^{\bar{N}}$ is equivalent to $\boldsymbol{x} \in \mathbb{R}^N$ with $N = 2\bar{N}$. Larger dimensional constellations are also possible, and have an even number of real dimensions $\widetilde{N} \in 2\mathbb{Z}^+$ 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 $M = |C|$. This statement's generalization going forward is $|C| \geq M^{1/\bar{N}}$. The number of **bits/subsymbol** will be $\tilde{b} \triangleq b/\overline{N}$ and the **energy per subsymbol** will be $\widetilde{\mathcal{E}}_x \triangleq \mathcal{E}_{\boldsymbol{x}}/\overline{N}$. Also, $\bar{\mathcal{E}}_{\boldsymbol{x}} = \widetilde{\mathcal{E}}_x/\widetilde{N}$ and $\bar{b} = \tilde{b}/\widetilde{N}$. The power remains

$$P_x = \frac{\mathcal{E}_{\boldsymbol{x}}}{T} = \frac{\widetilde{\mathcal{E}}_x}{\widetilde{T}} \quad . \tag{2.1}$$

Power typically is a given fixed parameter. The energy $\mathcal{E}_{\boldsymbol{x}} = P_x \cdot T$ often has units of power/Hz, for instance dBm/Hz is common. This energy is the same unit used for power spectral density, whether specifying energy/symbol or energy/subsymbol. Energy specification in communication design always implies, albeit tacitly, a time interval over which this energy finds use.

This text's coding viewpoint builds upon this repeated use of a smaller subsymbol constellation to develop all codes in this chapter, and later in Chapters 8 and 8.

[2]For the MIMO case, $\widetilde{N}$ will absorb any L_x factor for input dimensions, so that a subsymbol then multiplies any temporal subsymbol-dimension increase by the spatial dimensionality to obtain $\widetilde{N}$; when this happens $N \to L_x \cdot N$, and thus $N = \bar{N} \cdot \widetilde{N}$.

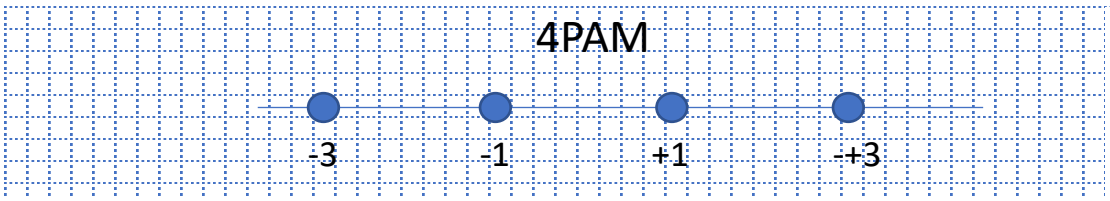


Figure 2.1: 4-PAM as the optimal one-dimensional $b = 2$ code.

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 ± 1 ± 3, and the minimum distance is $d_{min} = 2$. For the given energy of $\bar{\mathcal{E}}_{\boldsymbol{x}} = 5$, 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)

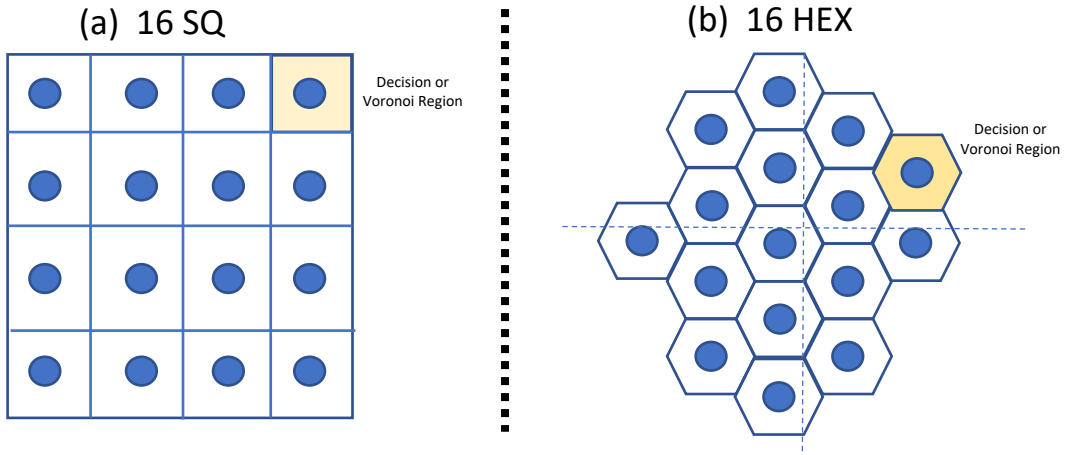


Figure 2.2: 16-SQ , or 2x 4PAM versus the optimal two-dimensional hexagonally packed code.

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 $d_{min} = 2$. 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[3] on P_e. 16HEX's hexagonal symbol array for any given energy[4] is best when $N = 2$. 16HEX's positive coding gain (up to .625 dB for HEX over SQ constellations as in Chapter 1) illustrates that two-dimensional packing can be more efficient than in 1 dimension.

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 "A_3") 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 centroid $\bar{\boldsymbol{x}} = \frac{1}{M}\sum_{i=0}^{M-1} \boldsymbol{x}_i$ 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 Problem 2.2 explores further. 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, $\bar{N}$, and the number of bits/symbol b:*

$$\rho \triangleq \overline{N} \cdot \log_2(|C|) - b \ . \tag{2.2}$$

The redundancy per subsymbol consequently is

$$\tilde{\rho} = \log_2(|C|) - \tilde{b} \ , \tag{2.3}$$

and the redundancy per-dimension is

$$\bar{\rho} = \tilde{\rho}/\widetilde{N} \ . \tag{2.4}$$

Nonzero redundancy forces constellation subsymbol vectors closer together within a given energy/volume, so a good code's concatenation of successive constellation subsymbols 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 subsymbols 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 only once the "outer" subsymbols outside the center 16SQ subsymbols within the two 2-dimensional subsymbols' concatenated codeword.

[3] 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.

[4] 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.

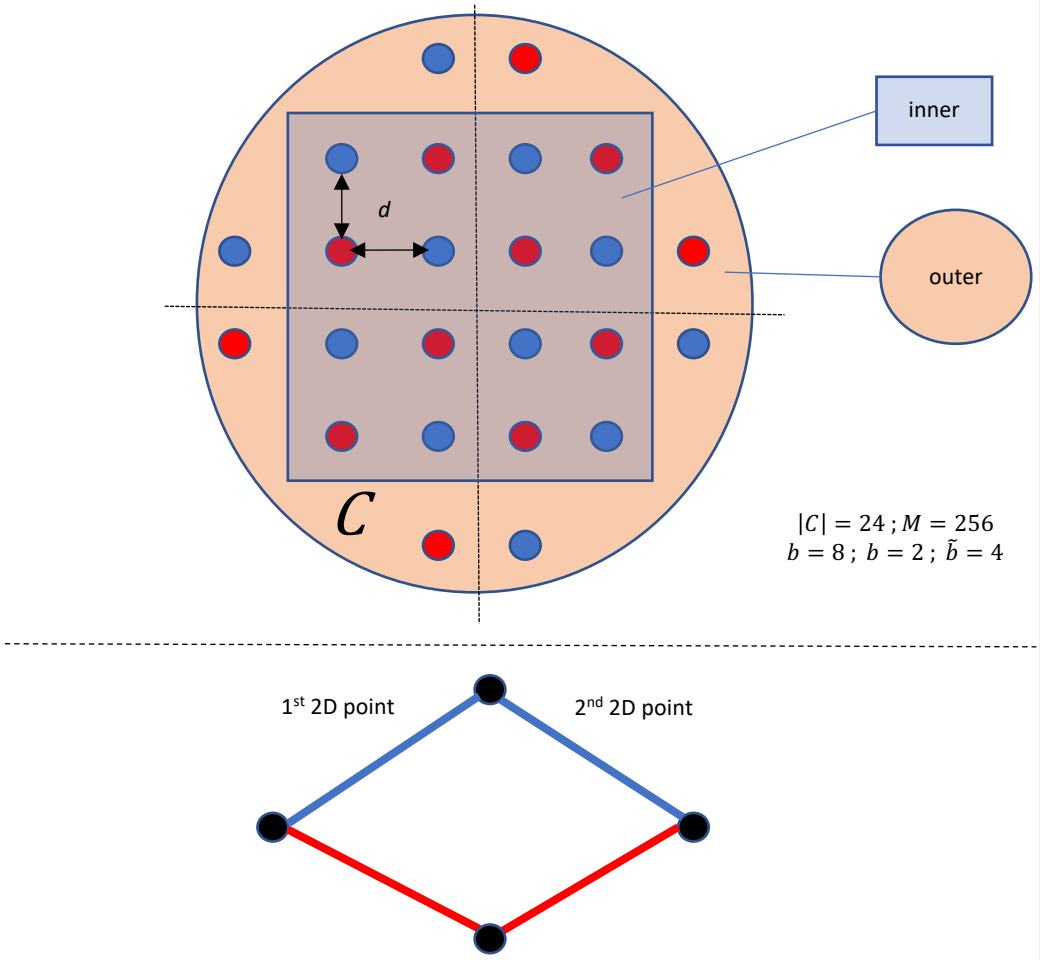


Figure 2.3: Good 4-dimensional code based on the D4 lattice structure.

The number of subsymbol values for codewords formed along the upper blue trellis-branch pair is thus

$$M_{\text{Upper path}} = \underbrace{8}_{\text{inner}} \times \underbrace{8}_{\text{inner}} + \underbrace{4}_{\text{outer}} \times \underbrace{8}_{\text{inner}} + \underbrace{8}_{\text{inner}} \times \underbrace{4}_{\text{outer}} = 128 \qquad (2.5)$$

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

$$P_{\text{24CR outer}} = \underbrace{\frac{32}{256}}_{\text{(in , out)}} + \underbrace{\frac{32}{256}}_{\text{(out , in)}} = \frac{1}{4} \text{ or } \frac{1}{32} \text{ each,} \qquad (2.6)$$

while the inner subysmbols occur more frequently at probability $P_{24CR,inner} = \frac{3}{4}$, 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** γ_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 $\mathcal{E}_{\boldsymbol{x}} = 10 + 10 = 20$ units of total energy for the average 4-dimensional energy with minimum distance 2. The coded 24CR system instead uses $\mathcal{E}_{\boldsymbol{x}} = 2 \cdot [\frac{3}{4} \cdot 10 + \frac{1}{4} \cdot 26] = 28$ 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, because of the lower-probability outer points.

Example 2.1.3's 4D code uses the best possible 4D-lattice-based structure, which corresponds to what is called a "Schlafi" or "D4" lattice in mathematics[5]. This code's $\gamma = 1.55$ dB gain illustrates again how dimensionality can be used to improve coding and shaping, even though the simple 2D constituent subsymbols have been maintained. The transmitted signal looks like 24CR, where indeed $|C| = 24 \geq M^{1/2} = 16$. With some effort, the maximum number of nearest D4 neighbors is computed as $N_e = 24 = (4 \times 4 + 4 + 4)$, but again the coding gain on the AWGN is larger than the loss from increased nearest neighbors. As $|C| \to \infty$ and thus also $M \to \infty$, then $\gamma \to \gamma_f = \sqrt{2}$ or 1.5 dB. These examples' pattern is clear, gain over simple uncoded transmission increases Chapter 1's fundamental gain to $\gamma_f = .625$ dB in 2D, to $\gamma_f = 1$ dB in 3D, and to $\gamma_f = 1.5$ dB in 4D, which motivates the next subsection.

2.1.2 Formal Code definition and Redundancy

Formal code definition develops further the concept of dimensional use (with again $L_x = 1$ presumed):

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

$$C_{\boldsymbol{x}} = \{\boldsymbol{x}_i\}_{i=0,\ldots,M-1} \tag{2.7}$$

where the N-dimensional codewords have $\overline{N}$, $\widetilde{N}$-dimensional **subsymbols** *selected from an $\widetilde{N}$-dimensional subsymbol constellation C with $|C|$ subsymbol values. The subscript $\boldsymbol{x}$ on $C_{\boldsymbol{x}}$ distinguishes $C_{\boldsymbol{x}}$ from the subsymbol constellation C . Thus,*

$$N = \underbrace{\widetilde{N}}_{\substack{\text{subsymbol} \\ \text{size}}} \cdot \underbrace{\overline{N}}_{\substack{\text{\# of} \\ \text{subsymbols}}} \quad . \tag{2.8}$$

A codeword generalizes Chapter 1's symbol, and indeed also is a "symbol," just usually a more complex structured symbol. $\overline{N}$ is the **blocklength** *in subsymbols, while N remains the symbol's number of real dimensions when the constellation has real vector subsymbols $\boldsymbol{x} \in \mathbb{R}$. 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 $\widetilde{N}$-dimensional constellation C to form a codeword. The choice of values within*

[5] The 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.

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*

$$\tilde{b} = \frac{b}{\widetilde{N}} = \bar{b} \cdot \widetilde{N} = b \cdot \frac{\widetilde{N}}{N} \quad . \tag{2.9}$$

$\widetilde{N} < \infty$ *is always finite. (Often* $\widetilde{N} = 2$*.). The code's minimum distance for AWGN use is*

$$d_{min}(C_{\boldsymbol{x}}) \triangleq d_{min} = \min_{\boldsymbol{x}_i \neq \boldsymbol{x}_j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \,, \tag{2.10}$$

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*

$$d_{free} = \min_{\boldsymbol{v}_i \neq \boldsymbol{v}_j} d_H(\boldsymbol{v}_i - \boldsymbol{v}_j) \quad , \tag{2.11}$$

where Figure 2.4 illustrates $\boldsymbol{v}$ *as encoder output bits.*

Figure 2.4 illustrates codeword construction from message $\boldsymbol{m}$'s b input bits to encoder output bits. Figure 2.4's discrete-transmitter **endcoder** 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 codewords (or larger 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 $2^{\tilde{b}+\tilde{\rho}}$ constellation vectors. Definition 2.1.1's redundancy allows variation of possible subsymbol choice within a codeword. If $\overline{N} = 1$, the redundancy necessarily is zero in practice. A longer nontrivial code's codewords decompose into $\overline{N} > 1$, $\widetilde{N}$-dimensional subsymbols $\tilde{\boldsymbol{x}}_{n=1,\ldots,\overline{N}}$ 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 $\hat{\boldsymbol{x}} = \arg \max_{\boldsymbol{x}_i \in C_{\boldsymbol{x}}} \{p_{\boldsymbol{y}/\boldsymbol{x}_i}\}$ to the channel output $\boldsymbol{y}$ (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 $\widetilde{N}$-dimensional subsymbols' allows the ML detector's simplification. The vector $\boldsymbol{v}$ sometimes also denotes a vector $\boldsymbol{v} \in \mathbb{C}^{\overline{N}}$ tacitly including a mapping of the bits into a complex vector such that $R_{\boldsymbol{v}\boldsymbol{v}} = I$. When the channel is AWGN with no filtering, then $\boldsymbol{x} = \boldsymbol{v}$; however when the channel is a matrix AWGN with $H \neq I$, the discrete modulator may completes the modulation into vector $\boldsymbol{x} \in \mathbb{C}^{\overline{N}}$ where it is possible that $R_{\boldsymbol{x}\boldsymbol{x}} \neq I$, but $trace\{R_{\boldsymbol{x}\boldsymbol{x}}\} = \mathcal{E}_{\boldsymbol{x}}$. The ML detector minimizes the squared distance between the channel output $\boldsymbol{y}$ and the possible corresponding message-indexed resultant vectors at the point just prior to the white-noise addition.

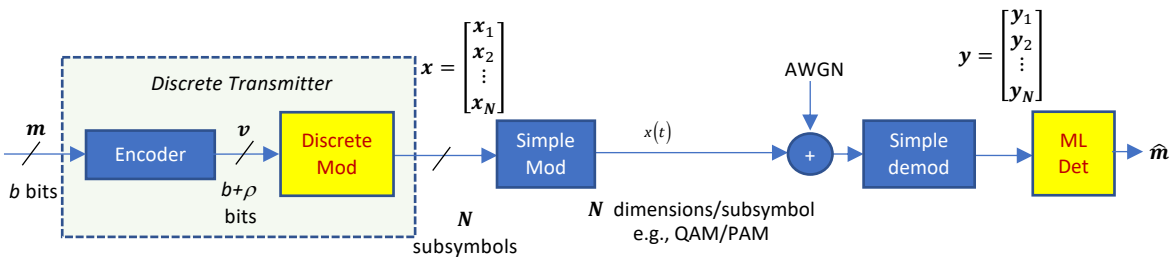


Figure 2.4: Construction of codewords with vector modulator and encoder.

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, $\widetilde{N}$ is always finite. When $\overline{N} < \infty$, and thus $N < \infty$, are finite, the code is called a **block code** (see Chapter 8). When $\overline{N} \to \infty$, 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 8). 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 Chapter 8. Appendix B also provides some alternative "trellis codes" that once found wide use, but now yielding to better codes.

Code Viewpoints: 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, $N = \widetilde{N} = 2$, $M = 32$, $b = 5$, and thus $\overline{N} = 1$ with $d_{min} = 2$ and $|C| = 32$. Also, $\bar{b} = 2.5$ and $\tilde{b} = 5$. However, from another 1D subsymbol viewpoint 32CR has $\widetilde{N} = 1$, $M = 32$, $b = 5$, $N = 2$, and thus $\overline{N} = 2$ with $d_{min} = 2$ and $|C| = 6$. The second viewpoint suggests the definition of redundancy in Definition 2.1.1, $\rho = 2 \cdot \log_2(6) - 5$. M and $b = \log_2(M)$ are the same from all viewpoints. $|C|$ and $\tilde{b}$ can vary with the choice of N and $\widetilde{N}$ or equivalently with $\overline{N} = N/\widetilde{N}$.

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 again has a redundancy of $\rho = 2 \cdot \log_2(6) - 5 = .17$ bits/codeword and $\bar{\rho} = .085$ bits/dimension. The first 2D interpretation has redundancy $\rho = 0$ 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 $2^{\tilde{b}}$ 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 measuring information via the constellation subsymbols' probability distribution, which Subsection 2.3.1's entropy addresses further.

Example 2.1.3's D4 lattice code had, from a 2D viewpoint, $\widetilde{N} = 2$, $M = 256$, $b = 8$, $\overline{N} = 2$ and thus $N = 4$; also $d_{min} = 2\sqrt{2}$. 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 $\tilde{\rho} = 0.58$ 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) Uncoded data transmission** *has subsymbol constellation C with zero redundancy $\tilde{\rho} = 0$. 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, $\tilde{\rho} > 0$, 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. This packing's measure is Chapter 1's **fundamental coding gain**, γ_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. This shaping's measure is Chapter 1's **shaping gain**, γ_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 LLN, the **sample average** over N observations of a random variable is:

$$\hat{\boldsymbol{x}} \triangleq \frac{1}{N} \cdot \sum_{n=1}^{N} \boldsymbol{x}_n \quad . \tag{2.12}$$

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

$$\hat{f}(\boldsymbol{x}) \triangleq \frac{1}{N} \cdot \sum_{n=1}^{N} f(\boldsymbol{x}_n) \quad , \tag{2.13}$$

as long as the function is well behaved and stationary (invariant probability density 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 $\{z_n\}_{n=1,\ldots,N}$ converges to its mean with large N such that*

$$\lim_{N\to\infty} \quad Pr\left\{\left|\left(\tfrac{1}{N}\textstyle\sum_{n=1}^{N} z_n\right) - \mathbb{E}[z]\right| > \epsilon\right\} \to 0 \quad \textbf{weak form} \tag{2.14}$$

$$\lim_{N\to\infty} \quad Pr\left\{\tfrac{1}{N}\textstyle\sum_{n=1}^{N} z_n = \mathbb{E}[z]\right\} = 1 \quad \textbf{strong form} \quad . \tag{2.15}$$

Proof: *See Appendix A.* **QED**.

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** that follows.

2.1.3.1 Sphere Packing

Subsections 2.1.1 and 2.1.2 's intuitive perspective with as $N \to \infty$ relates that a good AWGN-channel 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.

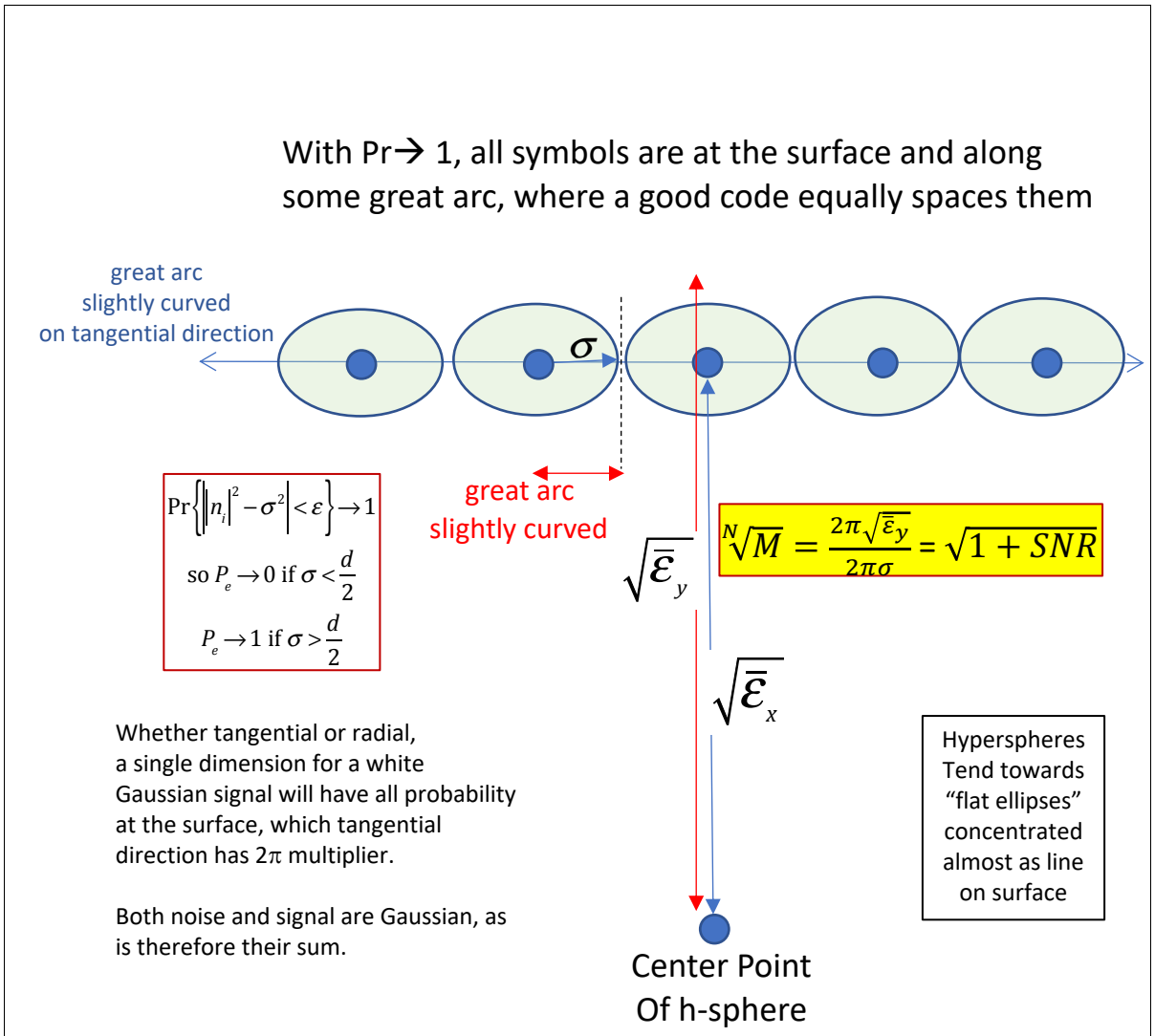


Figure 2.5: Ilustration of one dimension of noise-sphere packing into total-energy sphere.

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 $f(\boldsymbol{x}) = p_{\boldsymbol{x}}(\boldsymbol{x})$, then the sample-average probability distribution itself becomes a constant, namely the mean $E\left[p_{\boldsymbol{x}}(\boldsymbol{x})\right]$, implying a uniform distribution. Further, the LLN applied with $f(\boldsymbol{x}) = \|\boldsymbol{x}\|^2$ says that all codewords have the same sample-average energy. That is, not only is the codewords' subsymbol distribution approaching uniform, but with probability approaching 1, these codewords 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[6].

[6]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" sample 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 consequently, dimension.

Similarly, 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 $\sigma = \sqrt{\frac{\mathcal{N}_0}{2}}$. 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 $2\pi\sqrt{\bar{\mathcal{E}}_{\boldsymbol{y}}}$ for any signal and noise. The signal's great circle and the noise's great possible circle positions both appear in Figure 2.5. 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

$$M^{1/N} = \frac{2\pi\sqrt{\bar{\mathcal{E}}_{\boldsymbol{y}}}}{2\pi\sigma} = \frac{\sqrt{\bar{\mathcal{E}}_{\boldsymbol{x}} + \sigma}}{\sigma} = \sqrt{1 + SNR} \ , \tag{2.16}$$

or

$$\bar{b} = \frac{1}{2} \cdot \log_2 (1 + SNR) \ . \tag{2.17}$$

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)**.

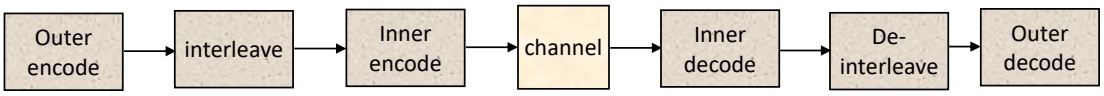


Figure 2.6: Interleaving concept.

Chapter 8, Sections 3 and 6, investigate the many forms of interleaving, its uses, and the theory behind it. Problem 2.8 provides a simple BICM example 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, known as a gray code, of interleaved bits to the subsymbols in the consequent implied code preserves the appearance of a memoryless channel like the AWGN, BSC, or BEC at the bit level, even with multilevel constellations (so $|C| > 2$).

Problem 2.8 also provides a simple example of gray coding, where adjacent 2D symbols differ only in one bit.

2.2 Coded systems, performance, and decoding

While Subsection 2.1.3's uniformly packed non-overlapping hyperspheres of equal diameter ($\approx d_{min}$) inside a larger sphere of (one-dimensional) radius $\sqrt{\bar{\mathcal{E}}_{\boldsymbol{x}} + \sigma^2}$ 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 encoder and decoder basics. Code designers have identified various codes that approximate good codeword-subsymbols' distribution. These codes fall into a few broad categories: Subsections 2.2.1 and 2.2.2 will respectively introduce "soft" and "hard" decoding concepts and their 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. Any AWGN-based code's ML detector for finds the input codeword (symbol) that minimizes the sum-squared difference between channel output and that codeword, so

$$\hat{\boldsymbol{x}} = \arg \left\{ \min_{\hat{\boldsymbol{x}}} \sum_{n=1}^{\bar{N}} \| \tilde{\boldsymbol{y}}_n - \hat{\tilde{\boldsymbol{x}}}_n \|^2 \right\} \quad , \tag{2.18}$$

where n is the subsymbol dimensional index, and

$$\hat{\boldsymbol{x}} = \begin{bmatrix} \hat{\tilde{\boldsymbol{x}}}_1 \\ \hat{\tilde{\boldsymbol{x}}}_2 \\ \vdots \\ \hat{\tilde{\boldsymbol{x}}}_{\bar{N}} \end{bmatrix} \quad . \tag{2.19}$$

Chapter 1's ML decoders are comparatively simple in that (2.18)'s sum has one term. Equation (2.18)'s direct calculation grows exponentially with $\bar{N}$ as $|C|^{\bar{N}}$ squared-distance computations. Thus, while simple in concept, the implementation may be overwhelming for reasonable $\bar{N}$ 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 $|C|^{\bar{N}}$. Regardless of complexity, the ML detectors symbol-error probability can again be approximated with the NNUB (with distance index d going from d_{min} upward as the distances in increasing size, while correspondingly $N_0 = N_e$ and N_d are the number of neighbors at distance index d) as

$$P_e \leq \sum_{d \geq d_{min}} N_d \cdot Q\left(\frac{d}{2\sigma}\right) \tag{2.20}$$

$$\approx N_e \cdot Q\left(\frac{d_{min}}{2\sigma}\right) \quad . \tag{2.21}$$

The nearest neighbor calculation (N_e) can rapidly become complex with large $\bar{N}$. Also, as $\bar{N}$ 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 $N_i >> N_e$. Usually good codes also have well understood distributions of $\{d_{min},\ d_{next},\ d_{next-to-next}\}$, ... and the corresponding numbers of nearest neighbors N_e, N_1, N_2 ... at these distances respectively. For systems with binary (or even finite-field) codes and upcoming Defintion 2.2.1, then $d_{min} = d_{free}$, $d_{next} = d_{free} + 1$, $d_{next-to-next} = d_{free} + 2$ so that the sum index simplifies to integers d_{free} ... ∞ with corresponding $N_d = 0$ if that particular distance does not appear in the linear code.

Marginal distributions: While ML detection minimizes the codeword-error probability (presuming the code has a uniform input distribution associated with the codewords), the error probability of individual codewords' subsymbols, or bits, may be of interest or may be desired to be minimized, as in Section 1.1.6. Minimizing the codeword error does not necessarily minimize the subsymbol error probability for any particular subsymbol; similarly nor does it minimize any particular input bit's error probability. The codewords are $\bar{N}$ dimensional vectors and the notation $\boldsymbol{\chi} \setminus \tilde{\boldsymbol{\chi}}_n \bigcap (\boldsymbol{\chi} \in C_{\boldsymbol{x}})$ here means that subsymbol $\tilde{\boldsymbol{\chi}}_n$ is punctured (removed) from all codewords. Then, minimization of subsymbol n's error probability would maximize the conditional probability[7]

$$p_{\tilde{\boldsymbol{x}}_n/\boldsymbol{y}}(\boldsymbol{\chi}_n, \boldsymbol{v}) \propto p_{\tilde{\boldsymbol{x}}_n,\boldsymbol{y}}(\boldsymbol{\chi}_n, \boldsymbol{v}) = \sum_{\boldsymbol{\chi} \in \{\boldsymbol{\chi} \setminus \tilde{\boldsymbol{\chi}}_n \bigcap \boldsymbol{\chi} \in C_{\boldsymbol{x}}\}} p_{[\tilde{\boldsymbol{x}}_n,\boldsymbol{y}]}(\boldsymbol{\chi} \setminus \boldsymbol{\chi}_n, \boldsymbol{v}) \quad , \tag{2.22}$$

which is proportional to the marginal probability distribution, $p_{\tilde{\boldsymbol{x}},\boldsymbol{y}}$ (by a function only of the fixed specific channel output value $\boldsymbol{y}$ and not the input codewords over which the maximum occurs). Equation (2.22)'s marginal distribution's calculation can be complex, but the concept is straightforward. Chapter 7 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 $\boldsymbol{u} = [u_1 \ ... \ u_b]$ 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

$$p_{u_i/\boldsymbol{y}}(u, \boldsymbol{v}) \propto p_{u_i,\boldsymbol{y}}(u, \boldsymbol{v}) = \sum_{\boldsymbol{u} \in (0,1)^b \setminus u_i} p_{[\boldsymbol{u},u_i],\boldsymbol{y}}(\boldsymbol{u}, \boldsymbol{v}) \quad \forall \ \ i = 1, ..., b \quad . \tag{2.23}$$

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 $\boldsymbol{y}$. Problem 2.3 provides a simple example of this calculation with 8PAM. 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

$$LLR_i(\boldsymbol{y}) \triangleq \ln \left(\frac{Pr\{u_i = 1; \boldsymbol{y}\}}{Pr\{u_i = 0; \boldsymbol{y}\}} \right) \quad . \tag{2.24}$$

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 $\bar{N}$ (and thus finite N), and often find use with the soft-decoded AWGN. Lattice codes also often use two-dimensional subsymbols, so then $\tilde{N} = 2$. A formal lattice definition appears in Appendix B, but succinctly a lattice Λ is a group of N-dimensional vectors $\boldsymbol{x}$ closed under addition. For real-number ($\boldsymbol{x} \in \mathbb{R}^N$) or complex-number ($\boldsymbol{x} \in \mathbb{C}^{\tilde{N}}$) addition, or equivalently the real or baseband/analytic complex AWGN respectively, the lattice necessarily has a countably infinite number of possible vector elements. Such lattices have all elements generated by N linearly independent **generator vectors** $\boldsymbol{g}_n$ so that a lattice element is generated by $\boldsymbol{x} = \sum_{n=1}^{N} z_n \cdot \boldsymbol{g}_n$ where $z_n \in \mathbb{Z}$ are integers and $\boldsymbol{g}_n$ are the real ($\tilde{N} = 1$ and $\boldsymbol{g}_n \in \mathbb{R}^N = R^{\tilde{N}}$), or complex ($\tilde{N} = 2$ and $\boldsymbol{g}_n \in \mathbb{C}^{\tilde{N}}$, equivalently $\mathbb{R}^{2\tilde{N}}$) generator (column[8]) vectors with real or complex entries respectively. The $\bar{N} \times \bar{N}$ nonsingular[9] **generator matrix**

[7] The sum here is actually $\bar{N} - 1$ summations, excluding input index n, $\sum_{\boldsymbol{\chi}_1} \sum_{\boldsymbol{\chi}_2} \cdots \sum_{\boldsymbol{\chi}_{\bar{N}}}$ to compute the desired marginal conditional distribution.

[8] 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 $\boldsymbol{v}$ and $\boldsymbol{u}$ are finite-field codeword outputs and inputs respectively, then $\boldsymbol{v} = \boldsymbol{u} \cdot G$ so then effectively the generator is transposed notationally with respect to lattice and real/complex modulating symbol generation.

[9] If the generator matrix were singular, it really corresponds to a smaller dimensional code and the extra dimensions should be eliminated.

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

$$G_\Lambda = \left[\boldsymbol{g}_{\Lambda,\bar{N}} \cdots \boldsymbol{g}_{\Lambda,1} \right] \quad .$$

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

$$d_{min} = \min_i \|\boldsymbol{g}_i\| \quad . \tag{2.25}$$

The Voronoi Region $\mathcal{V}(\Lambda)$ has fundamental volume $V(\Lambda) = |\mathcal{V}(\Lambda)|$, which is thus the determinant of the square nonsingular generator matrix, $V(\Lambda) = |G_\Lambda|$, since $V(\mathbb{Z}^{\bar{N}}) = 1$. 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 $\boldsymbol{z}$. Problem 2.4 explores further generators and their structure with lattices. Appendix B further pursues lattice codes.

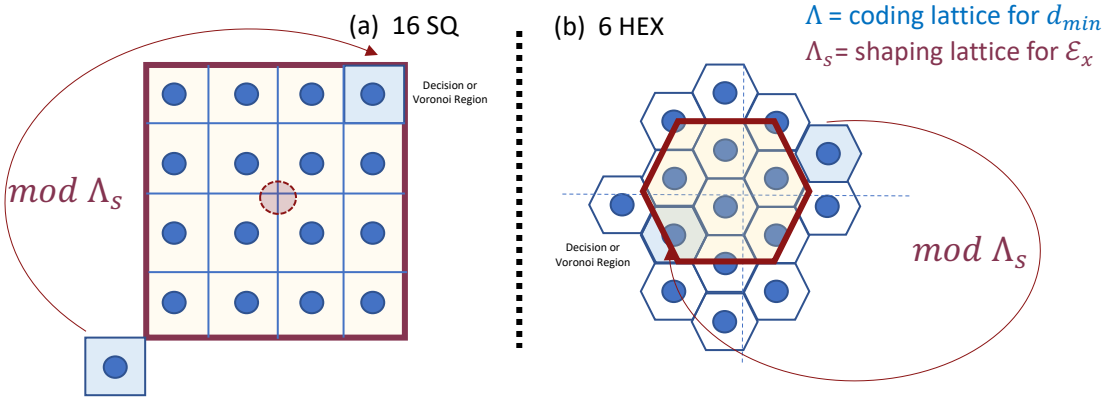


Figure 2.7: Revisit Figure 2.2.

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[10] 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 $V(\Lambda)$,for example 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_{min} and better γ_f) as if each region $\mathcal{V}(\Lambda)$ 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 $\bar{\mathcal{E}}_{\boldsymbol{x}}$ and better γ_s) into which all $\mathcal{V}(\Lambda)$ are packed, as further investigated in Appendix B. There are also other non-Voronoi methods for code design in Chapter 8.

Chapter 1, Section 1.3.4.3 on "More Constellation Performance Measures" introduced **coding gain** γ_c as the sum (in dB) of fundamental gain γ_f and shaping gain γ_s. Figure 2.7 revisits Figure 2.2 here

[10]This translate is often called a **coset** of the lattice, meaning here that the M lowest-energy elements from the centroid-centered coset $\boldsymbol{x}_i \rightarrow \boldsymbol{x}_i - 1/M \sum_{j=0}^{M-1} \boldsymbol{x}_j$ of the lattice are used

with now the shaping-lattice overlaying the coding lattice, visually illustrating the two contributions, which follow:

$$\gamma = \frac{\left(\frac{d^2_{min}(\boldsymbol{x})}{V^{2/N}(\Lambda)}\right)}{\left(\frac{d^2_{min}(\breve{\boldsymbol{x}})}{V^{2/N}(\breve{\Lambda}])}\right)} \cdot \frac{\left(\frac{V^{2/N}(\Lambda)}{\bar{\mathcal{E}}_{\boldsymbol{x}}}\right)}{\left(\frac{V^{2/N}(\breve{\Lambda})}{\bar{\mathcal{E}}_{\breve{\boldsymbol{x}}}}\right)} \tag{2.26}$$

$$= \gamma_f + \gamma_s \quad (dB) \tag{2.27}$$

While the shaping lattice Λ_s does not explicitly appear in Equation 2.27, this shaping lattice indirectly determines the energy $\mathcal{E}$ for the compared structure relative to a shaping lattice for another compared code. As earlier, typically the reference system for both fundamental and shaping is the integer lattice $\mathbb{Z}^N$. Also important is that code design separates essentially into two independent parts: A designer can design first a good code in terms of only fundamental gain. This will ensure the best density of codewords with a given distance distribution within any volume. Designers typically first address good fundamental-gain code design, essentially ignoring energy but looking at maximum distance between codewords within some volume, which is a function of packing in an N-dimensional space. Designers then proceed to shaping-code design (which offers at best only 1.53 dB additional gain when the boundary is a hypersphere as in Chapter 1). This design focuses on the boundary's proximity to a sphere in the N-dimensional space. The two effects are largely separable in practice. Technically, if a code with $\Gamma = 0$ dB is used, both components are optimized as $N \to \infty$.

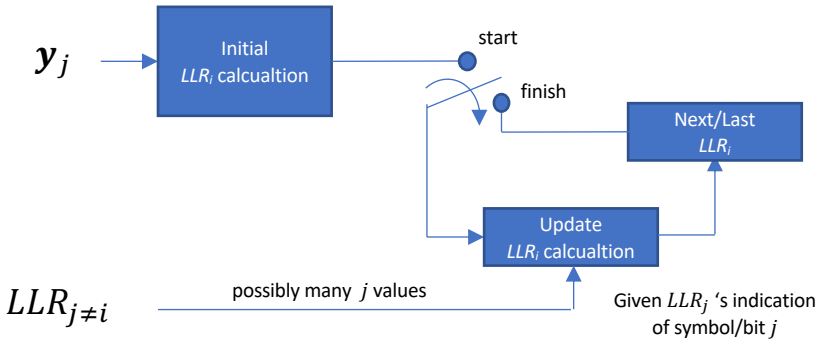


Figure 2.8: Iterative calculation of LLR_i.

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 2.8 illustrates the decoder's LLR computation for one bit position in such a block 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 the actual set of LLR_i's with lower complexity than computing each directly. 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[11], 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 Chapter 8.

[11]The LLR processing of $\boldsymbol{y}$ is 1-to-1 to the input bit, thus tacitly satisfying Chapter 1's Reversibility Theorem.

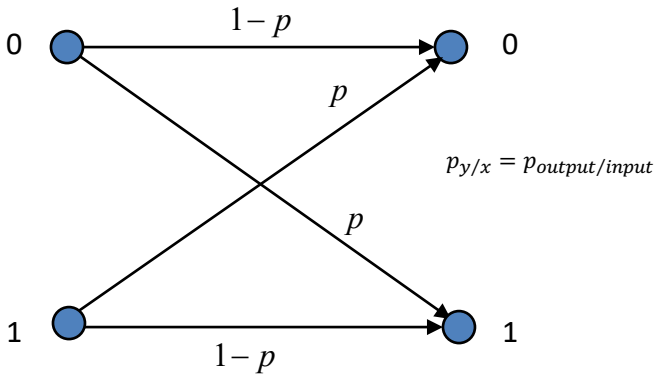


Figure 2.9: The Binary Symmetric Channel, BSC.

2.2.2 Hard-Decoded Systems

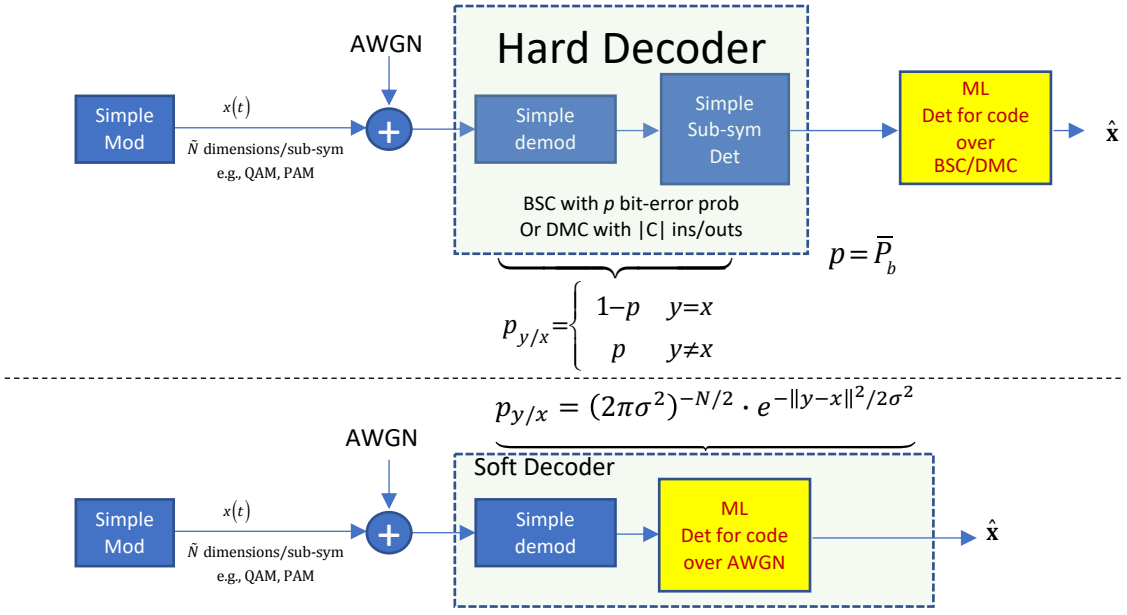


Figure 2.10: Illustration of hard and soft decoding.

Figure 2.9 repeats Chapter 1's Binary Symmetric Channel (BSC). The BSC has an error probability that measures the likelihood that the discrete binary output is not equal to the input. That error probability p, is the BSC's sole characterizing parameter. This BSC is used in hard decoding. Figure 2.9 describes the probability that a certain output is received, given any particular input. Hard-decoders make intermediate decisions 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 2.10's upper diagram.

The hard decoder's calculations differ from soft decoding's optimum closest sum-squared distance as in Figure 2.10'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.

2.2.2.1 Binary Block Codes

Binary block codes use binary arithmetic or equivalently linear modulo-2 addition over the field $\mathbb{GF}^2$, see Appendix B. 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 $\widetilde{N} = 1$ and so then $k = b \leq n = N = \bar{N}$ 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**[12] where $p \stackrel{\Delta}{=} n - k = \rho$ is the redundancy. The lower-case p is also in common use and equal to ρ 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 $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ differ, and extends to define the **free distance** of a code as:

Definition 2.2.1 (Free Distance)

$$d_{free} \stackrel{\Delta}{=} \min_{\substack{\boldsymbol{x}_1 \neq \boldsymbol{x}_2 \\ \in C_{\boldsymbol{x}}}} d_H(\boldsymbol{x}_1, \boldsymbol{x}_2) \quad . \tag{2.28}$$

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 $\lfloor \frac{d_{free}-1}{2} \rfloor$; 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 $d_{free} - 1$. 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_{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_{free} would then imply that any set of $d_{free} - 1$ bits could be deleted from this code and still maintain distinct vectors. The number of possible codewords is then bounded as

$$M(n, d_{free}) \leq 2^{n-(d_{free}-1)} = 2^{n-d_{free}+1} \quad , \tag{2.29}$$

which is known as the **Singleton Bound**[13], formally:

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

$$k = \log_2(M) \leq n - d_{free} + 1 \tag{2.30}$$

$$r \leq 1 - \frac{d_{free} - 1}{n} \quad . \tag{2.31}$$

Proof: *See Equation (2.29).* **QED.**

[12] This p for redundancy should not be confused with the p used to characterize the BSC.
[13] After former Stanford student Richard Colom Singleton, 1927-2007.

Larger d_{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" $d_{free}-1$ around symbols[14]. The size (number of points included) of such a ball is $\sum_{j=0}^{d_{free}-1}\binom{n}{j}$, leading to[15]

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

$$\frac{2^n}{\sum_{j=0}^{d_{free}-1}\binom{n}{j}} \leq M(n,d_{free}) \quad , \tag{2.32}$$

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

So, a good code should have $M(n,d_{free})$ 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[16] relates that $2^n=\sum_{j=1}^{n}\binom{n}{j}$, or specifically $2^{d_{free}-1}=\sum_{j=1}^{d_{free}-1}\binom{d_{free}-1}{j}$. Since $d_{free}\leq n$, then the Gibert-Varshamov Bound's denominator is always greater than or equal to $2^{d_{free}-1}$ because (2.32)-denominator's sum clearly increases with the n inside the sum.

Singleton Bound Tightness: For example, the SB states that a code that corrects up to 2 bit errors ($d_{free}\geq 5$) with block-length $n=20$ needs $r\leq 1-\frac{4}{20}=0.8$. At $n=20$, the GVB is $\frac{2^{20}}{\sum_{j=0}^{4}\binom{20}{j}}=169$ 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 need if a symbol/codeword error occurs. While the Singleton Bound's construction implies the existence of a code with d_{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 (2.29)'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 $d_{free}=n$. 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 $d_{free}\neq 1,2,$ nor n. For nonbinary codes, there can be nontrivial MDS codes.

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_Λ for the lattice code)

$$\boldsymbol{v}=\boldsymbol{u}\cdot G \quad , \tag{2.33}$$

[14] This envisions $d_{free}/2$ on each "side" of the ball-centered codeword

[15] After Edgar Gilbert, Bell Laboratories Scientist, 1923-2013 and Aremnian coding theorist Rom Varshamov, 1927-1999.

[16] The 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 $b=q-1$.

where $\boldsymbol{v}$ is a $1 \times n$ output-bit row vector and $\boldsymbol{u}$ is a $1 \times k$ input-bit row vector[17]. The use of G here implies "generator" in a finite field[18]. The generator G has only 1's and 0's as entries, and the matrix multiplication uses binary addition and multiplication in $\mathbb{GF}^2$. 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 codeword/symbol error probability of

$$P_e \leq \sum_{i=\lceil \frac{d_{free}}{2} \rceil}^{N} N_{i-\lceil d_{free}/2 \rceil} \cdot p^i \cdot (1-p)^{N-i} \leq \sum_{i=\lceil \frac{d_{free}}{2} \rceil}^{N} \binom{N}{d_{free}+i} \cdot p^i \cdot (1-p)^{N-i} \; . \tag{2.34}$$

Again, nearest neighbors $N_e = N_0$ are those codewords at distance d_{free} from any given codeword. The number of neighbors at distance $d_{free}+i$ is N_i and cannot exceed $\binom{N}{d_{free}+i}$. The **average number of subsymbol errors per codeword** is

$$\tilde{P}_{e,ss} \leq \sum_{d=\lceil d_{free}/2 \rceil}^{N} d \cdot N_{d-\lceil d_{free}/2 \rceil} \cdot p^d \cdot (1-p)^{N-d} \; . \tag{2.35}$$

The subsymbol error probability is

$$\bar{P}_{ss} = \frac{\tilde{P}_{e,ss}}{N} \quad , \tag{2.36}$$

while the average number of bit errors[19] per subsymbol has upper bound

$$\bar{P}_b \leq \frac{2^{\tilde{b}-1}}{2^{\tilde{b}}-1} \cdot \frac{\tilde{P}_{e,ss}}{N} \; . \tag{2.37}$$

(2.37)'s left fraction follows as at least one bit is in error as in the numerator overall the possibly subsymbol to bit mappings, deleting the correct bit combination, as in the denominator. This differs from the "softer" decoder situation where $N(b,d)$ characterizes number number of bit errors b at distance d and sums over b instead of a hard subsymbol decision, for which all subsymbol bit-error combinations are equally likely for all situations independent of the encoder. In this softer-decoder situation, then the number of bit errors per codeword error event

$$P_b \leq \sum_{d=\lceil d_{free}/2 \rceil}^{N} b \cdot N(b,d) \cdot p^d \cdot (1-p)^{N-d} \; . \tag{2.38}$$

with $\bar{P}_b = \frac{1}{b} \cdot P_b$, as in Chapter 1. The analysis applies (2.35) if the decoder makes a hard decision on a subsymbol and applies (2.38) if the hard decisions are made directly at bit level. Indeed when a subsymbol equals a bit, the two are the same.

Free distance and minimum codeword weight: The free distance d_{free} is the least Hamming weight (number of 1's) of any linear combination G's rows (since the all-zeros symbol is a codeword and an input vector with d_{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 $d_{free}+i$, $i \geq 0$, (again with $N_0 \triangleq N_e$ while N_d is the number of nearest neighbors at distance d) follows a form that generalizes later from this equation:

$$N_d = \max \left\{ 0, \binom{N}{d} \cdot \sum_{j=0}^{i} \underbrace{(-1)^j}_{\substack{\text{reverse over count} \\ \text{previous step}}} \cdot \underbrace{\binom{d_{free}+i}{j}}_{\text{position choices}} \cdot \underbrace{[2^{i+1-j}-1]}_{\text{ways to differ}} \right\} \; . \tag{2.39}$$

[17] Note the reversal of column vectors to row vectors that occurs throughout the binary-code literature.

[18] 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.

[19] Assuming all $2^{\tilde{b}}$ subsymbol mappings are equally likely.

Equation (2.39)'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 of interest for N_d 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[20] **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 Equation (2.39)'s nearest neighbor counts N_d are:

$$N_0 = N_e = N \cdot 1 \cdot 1 \cdot 1 = N = \binom{N}{1} \tag{2.40}$$

$$N_1 = \binom{N}{2} \cdot [1 \cdot 1 \cdot 3 - 2 \cdot 1] = \binom{N}{2} \tag{2.41}$$

$$N_2 = \binom{N}{3} \cdot [1 \cdot 1 \cdot 7 - 3 \cdot 3 + 3 \cdot 1] = \binom{N}{3} \tag{2.42}$$

$$N_3 = \binom{N}{4} \cdot [1 \cdot 1 \cdot 15 - 4 \cdot 7 + 6 \cdot 3 - 4] = \binom{N}{4} \tag{2.43}$$

$$\vdots = \vdots \tag{2.44}$$

$$N_d = \binom{N}{d+1} \tag{2.45}$$

Equation (2.45) can be proved by induction. For the other two trivial binary MDS codes,

$$\text{if } d_{free} = N \tag{2.46}$$

$$N_e = N \tag{2.47}$$

$$N_{d>d_{free}} = 0 \tag{2.48}$$

$$\tag{2.49}$$

$$\text{if } d_{free} = 2 \tag{2.50}$$

$$N_e = N \tag{2.51}$$

$$N_d = \begin{cases} \binom{N}{d+2} & \text{even } d \\ 0 & \text{odd } d \end{cases} \tag{2.52}$$

MDS Generator/Parity: d_{free} will thus be maximum if the rank of the generator matrix G is k, 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 $d_{free} - 1$ and a k^{th} 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 $d_{free} - 1$ generator just formed. The generator's null space forms the **"dual" code's** generator matrix (often called[21] H or the **parity matrix**). So G spans the code space with rank $\mathscr{P}_G$, and H spans the null space with rank $\mathscr{P}_H$. For these vectors of length n,

$$\mathscr{P}_G + \mathscr{P}_H = n \ . \tag{2.53}$$

When G is full rank ($\mathscr{P}_G = k$), it can be premultiplied by any rank-k $k \times k$ scrambling matrix[22] 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_{free} ones in a row because there is

[20] After (Florence) Jesse MacWilliams, 1917-1990, a British mathematician was an early woman pioneer in the area of finite fields and group theory.

[21] 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.

[22] Scrambling here means 1-to-1 nonsingular matrix transformation.

a codeword that is at that minimum distance from the all zeros codeword in all linear codes, let us call that codeword $\boldsymbol{v}'$. It is clear then that

$$\boldsymbol{v}' H^* = 0 \tag{2.54}$$

because all codewords will be orthogonal to the null space. So, the minimum number of its columns that may be linearly dependent is $d_{free} - 1 \leq \rho_H$. Further, since $\mathscr{P}_H = n - \mathscr{P}_G = n - k$ (from Equation (2.53)) **only for the full rank generator case**, then

$$d_{free} \leq \mathscr{P}_H + 1 = n - k + 1 \quad , \tag{2.55}$$

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

Error-Probability Analysis: For linear binary codes, the codeword-error probability can be well approximated[23] through the BSC's error-probability tighter-symmetric B-bound (See Section 1.1) as

$$P_e \approx N_e \cdot \binom{d_{free}}{\lceil d_{free}/2 \rceil} \cdot p^{d_{free}/2} \cdot (1-p)^{n - d_{free}/2} \tag{2.56}$$

$$< \frac{N_e}{2} \cdot [4 \cdot p \cdot (1-p)]^{\lceil \frac{d_{free}}{2} \rceil} \quad . \tag{2.57}$$

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)

$$\bar{P_b} \approx \frac{N_b}{b} \cdot [4 \cdot p \cdot (1-p)]^{\lceil \frac{d_{free}}{2} \rceil} \quad . \tag{2.58}$$

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

$$d^2_{min,BSC} = 4 \cdot d_{free} \quad . \tag{2.59}$$

Using (2.59) when $P_e < 10^{-3}$, hard decoding is about 3 dB worse than soft decoding by comparing the two error bounds in (2.57) and Chapter 1's NNUB $P_e < N_e \cdot Q\left(\frac{d_{min}}{2\sigma}\right)$. This 3dB approximation arises by comparing the 2-PAM AWGN NNUB error expression as

$$P_e = N_e \cdot Q(\sqrt{d_{free} \cdot SNR}) \propto e^{-d_{free} \cdot \frac{SNR}{2}} \tag{2.60}$$

with the hard-coded

$$p^{d_{free}/2} = \left[Q(\sqrt{SNR})\right]^{d_{free}/2} \propto e^{-\frac{d_{free}}{2} \cdot \frac{SNR}{2}}, \tag{2.61}$$

so a factor of 2 worse or 3 dB. The approximations in Equations (2.56) - (2.61) hold when errors largely arise from only the minimum-distance events; very powerful codes' analysis cannot ignore other larger distances for which the $N_{d>d_{free}}$ and corresponding $N_{b,d>d_{free}}$ contribute significantly to error probability. Chapter 7 provides the more complex analysis necessary for such codes.

From the ball-packing perspective but more precisely and returning to an upper bound, the **Hamming Bound**[24] ensures sufficient ball space around each symbol for a given n and radius $d_{free}/2$ and provides an alternative upper bound:

[23] Realizing that $n - d_{free}/2 \geq d_{free}/2$ or equivalently the free distance cannot exceed the codeword length, and that $\binom{d_{free}}{d_{free}/2} \leq 4^{\lceil d_{free}/2 \rceil}$.

[24] After Bell Laboratories scientist Richard W. Hamming, 1915-1998.

Lemma 2.2.3 (The Hamming Bound) *With* $t \triangleq \lfloor \frac{d_{free}-1}{2} \rfloor$, *the number of codewords is upper bounded as*

$$M(n, d_{free}) \leq \frac{2^n}{\sum_{j=0}^{t} \binom{n}{j}} = \frac{\#\ of\ total\ binary\ vectors}{\#\ of\ binary\ vectors\ in\ ball\ of\ radius\ t} \quad . \tag{2.62}$$

Proof: The proof is in the rightmost expression in (2.62).**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 $d_{free} = 3$. Some other perfect codes are also known with larger free distances, see for instance Golay Codes[25].

2.2.2.2 Block Codes for the DMC

Block-code design sometimes presumes inner hard decoding that results in a DMC[26], of which the simplest example is the BSC. For the BSC, these binary block codes are applicable with $\widetilde{N} = 1$, $n \triangleq N = \bar{N}$, 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 constellation's size $|C|$, using Galois Fields as in Appendix B and Chapter 8, and often linear in the field $\mathbb{GF}^{|C|}$. For these codes also, $\widetilde{N} = 1$ and $N = \bar{N}$. Instead of a lower-case k, these codes use upper-case K to enumerate the number of information-bearing subsymbols in a codeword of $N = \bar{N}$ subsymbols. The number of parity subsymbols becomes $P = N - K = \rho \cdot \log_2(|C|)$ and $R \triangleq r = \frac{K}{N} \leq 1$. 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[27]. Well-designed codes also will be able to correct $d_{free}/2$ subsymbols that have errors.

Block codes 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 $t = \lfloor \frac{dfree}{2} \rfloor$.

Discrete Memoryless Channel (DMC): Figure 2.11 shows a symmetric discrete memoryless channel where $(p < 1/2)$ is the probability of an individual subsymbol error. Figure 2.11 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[28]. Equation (2.34) is again the corresponding minimized codeword-error probability, which has error-probability approximation (2.57).

[25] For instance, at Wikepedia page https://en.wikipedia.org/wiki/Binary_Golay_code .
[26] Discrete Memoryless Channel that first appears in Chapter 1, Section 4.
[27] Instead a uniform distribution on $\mathbb{GF}^{|C|}$ holds with $\bar{N} \to \infty$.
[28] This 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.

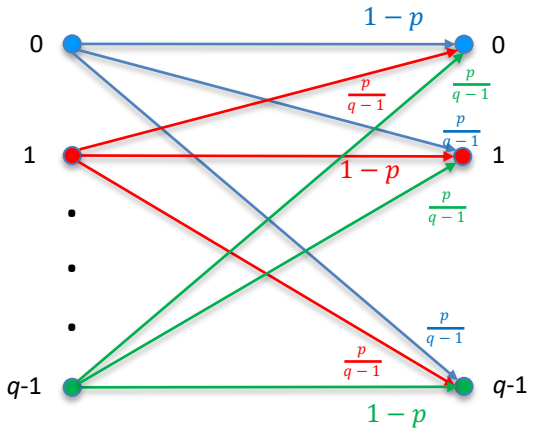


Figure 2.11: General Symmetric Discrete Memoryless Channel.

The bit-error probability again follows (2.58) 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 euclidean 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 and corresponds to good uniformly spread ball/sphere packing. 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 arbitary q, and again $t \triangleq \lfloor \frac{d_{free}-1}{2} \rfloor$, to*

$$\underbrace{\frac{q^N}{\sum_{j=0}^{d_{free}-1} \binom{N}{j} \cdot (q-1)^j}}_{Gilbert\text{-}Varshamov} \leq M_q(N, d_{free}) \leq \min \left\{ \underbrace{\frac{q^N}{\sum_{j=0}^{t} \binom{N}{j} \cdot (q-1)^j}}_{Hamming}, \underbrace{q^{N-d_{free}+1}}_{Singleton} \right\} . \tag{2.63}$$

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 q-ary 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 $q = 2$, 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[29], BCH[30], and others [7]. For MDS codes, the nearest neighbor counts in (2.39) at all distances approximate as (again with $N_0 \triangleq N_e$ while N_d is the $(i \triangleq d - d_{free})^{th}$ nearest neighbor) as

$$N_d \simeq \binom{N}{d} \cdot \sum_{j=0}^{i} (-1)^j \cdot \binom{d}{j} \cdot \left[q^{i+1-j} - 1\right] \quad . \tag{2.64}$$

Negative values in (2.39) or (2.64) for N_d 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_{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 (2.64) when the code exists.

2.2.3 Erasure Decoding

Figure 2.12'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_{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

$$P_e \approx N_e \cdot p^{d_{free}} \cdot (1-p)^{n-d_{free}} \quad . \tag{2.65}$$

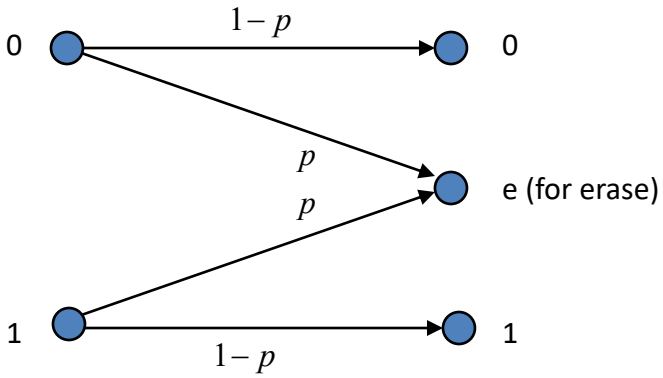


Figure 2.12: The Binary Erasure Channel, BEC. Note there are no errors, only erasures when the input is uncertain.

This error probability is intermediate to those of full soft decoding (AWGN) and hard decoding (BSC). Comparison of (2.65) to (2.58) shows this directly. In terms of hard decoding's 3 dB loss with respect to soft decoding, some is recovered if the d_{free} is odd while all is recovered if d_{free} is even. This

[29] After Irving S. Reed, 1923-2012, a USC Mathematician and Gustave Solomon, 1930-1996, a Cal Tech mathematician, both of whom made contributions to finite-field codes, but none more famous than the heavily used Reed Solomon codes.

[30] Bose–Chaudhuri–Hocquenghem codes, after mathemsticians Raj Bose, 1901-1987, D. K. Ray-Chaudhuri, 1933- , of India, and Alexis Hocquenghem, 1908-1990 of France

calculation is optimistic since the BEC does not admit any possibility of $1 \to 0$ or vice versa, and that might be a questionable assumption in practice. For instance, with $d_{free} = 5$, 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 cyclic 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 $d_{free} - 1$ "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**[31], 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 $\bar{N}$ and thus also have infinite N, but $\tilde{N} = \frac{N}{N} = \frac{\bar{N}}{\bar{N}}$ 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 $\bar{N}$ 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.

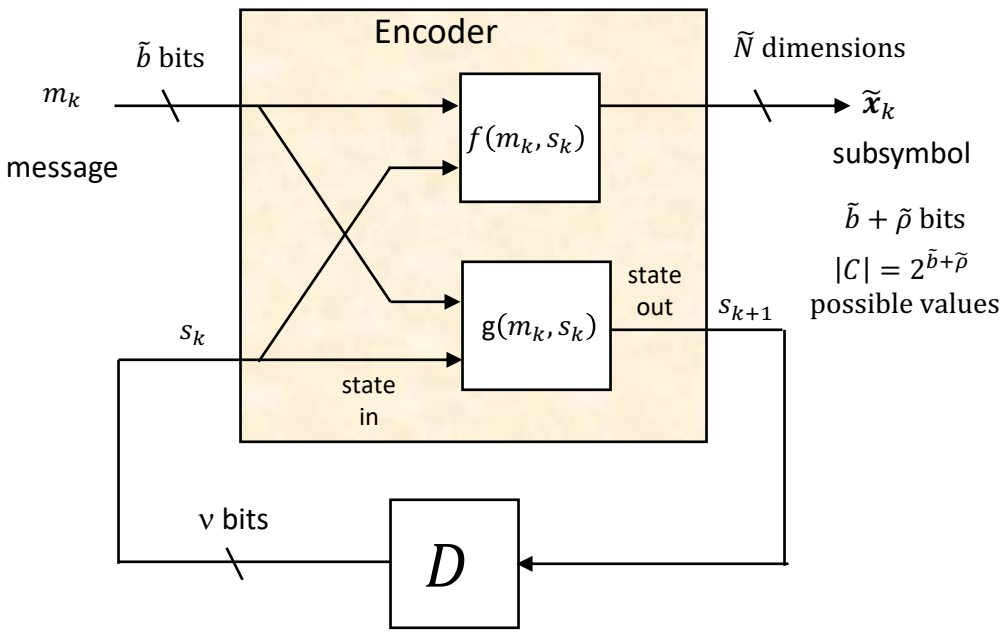


Figure 2.13: Sequential encoder for trellis codes.

Figure 2.13 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 $\bar{N}$ while maintaining a finite (per unit time) complexity through the

[31] CRC is a smple MDS block code that detects if the channel output is a codword or not.

sequential encoder's use. $\tilde{b}$ bits enter the sequential encoder at each subsymbol instant and combine with ν state-characterizing bits that enumerate one of 2^ν 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 ν bits are not redundant bits, but together with the $\tilde{b}+\tilde{\rho}$ output bits may characterize the $\widetilde{N}$-dimensional subsymbols $\tilde{\boldsymbol{x}}_k$. The $\widetilde{N}$-dimensional output subsymbol depends on both the $\tilde{b}$ input bits and the ν state bits through the function

$$\tilde{\boldsymbol{x}}_k = f(m_k, s_k) \quad . \tag{2.66}$$

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

$$s_{k+1} = g(m_k, s_k) \quad . \tag{2.67}$$

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 7).

For the AWGN channel, Appendix B reviews several popular trellis (or Ungerboeck[32]) and coset (or Forney's coset [33]) 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 $\bar{b}$ and $\tilde{b}$ are finite. The constellation size is also finite $|C|$ (although the constellation may be large and approximate samples from a Gaussian distribution).

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 $\widetilde{N}$ binary two-level constellations corresponding to $\widetilde{N}$ successive BSC uses. In the convolutional-code case, $k = \tilde{b}$ and $n = \widetilde{N}$ (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 $r = k/n = \bar{b} = \frac{\tilde{b}}{\widetilde{N}} \leq 1$. The convolutional code's redundancy and rate add to one

$$\underbrace{\bar{\rho}}_{\frac{n-k}{n}} + \underbrace{r}_{\frac{k}{n}} = 1 \quad . \tag{2.68}$$

Convolutional codes can use higher-order addition in $\mathbb{GF}^{q^j}$, 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 $\bar{N} \to \infty$ instances of $\widetilde{N}$ 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).

[32] After Austrian/Swiss Gottfried Ungerboeck, 1940 –, who developed the codes at IBM Research-Zurich.
[33] After American G. David Forney, Jr., 1940 – , who developed these codes while at Motorola-Codex.

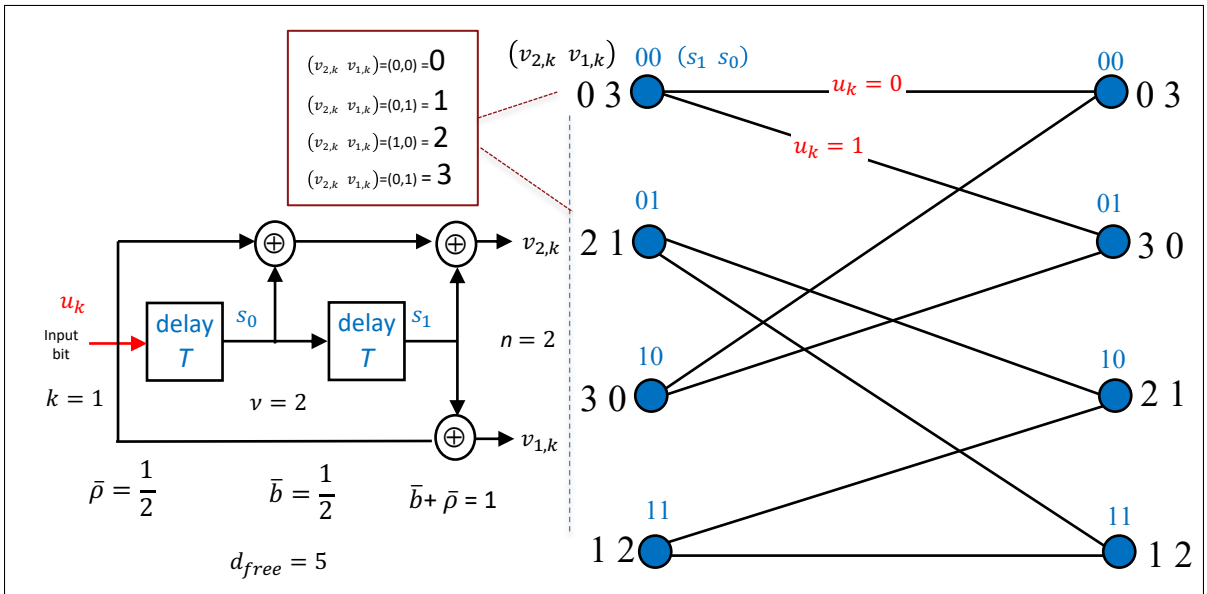


Figure 2.14: Simple 4 state convolutional encocder and trellis diagram.

EXAMPLE 2.2.1 (4-state convolutional code) Figure 2.14 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 $\bar{b} = \frac{1}{2}$ and $r + \bar{\rho} = 1$. The trellis appears on Figure 2.14'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 7, 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 8 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 2.15 decomposes in a simple way the basic types of codes. Many codes use concatenations of these code types in various ways as in Chapter 8.

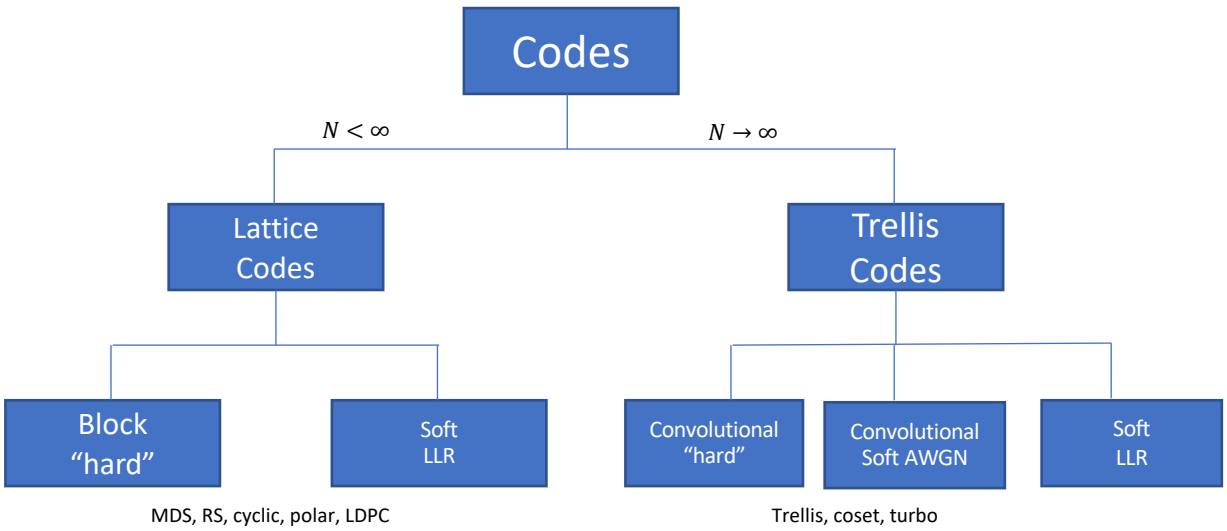


Figure 2.15: Simple code-type decomposition based on $N < \infty$ and $N \to \infty$as well as decoder and type.

2.3 Mutual Information and Channel Capacity

This section formalizes Subsection 2.1.3's 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**,

$$\mathcal{C} = \max_{p_{\tilde{\boldsymbol{x}}}}[I] \quad . \tag{2.69}$$

Subsection 2.3.1 introduces the concept of entropy that measures a 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 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 bounds follow 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 filtered AWGN-channel capacity. Subsection 2.3.6 makes a few more random-code-design observations.

2.3.1 Entropy and Data Symbols' Distribution

A random data symbol's (or message's) entropy $\mathcal{H}_{\boldsymbol{x}}$ is determined by the symbols' (codewords') probability distribution $p_{\boldsymbol{x}}$. 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* $\boldsymbol{x} \in \{\boldsymbol{x}_0, ..., \boldsymbol{x}_{M-1}\}$ *with stationary probability distribution* $p_{\boldsymbol{x}}(i)$ $i = 0, ..., M-1$ *is:*

$$\mathcal{H}_{\boldsymbol{x}} \triangleq -\sum_{i=0}^{M-1} p_{\boldsymbol{x}}(i) \cdot \log_2 [p_{\boldsymbol{x}}(i)] \quad (\text{ bits/symbol} - \boldsymbol{x}) \tag{2.70}$$

$$= \mathbb{E}\{\log_2 [1/p_{\boldsymbol{x}}]\} \geq 0 \quad . \tag{2.71}$$

If the distribution is over a subsymbol's $\tilde{\boldsymbol{x}}$ constellation, then the entropy is for that subsymbol, written $\mathcal{H}_{\tilde{\boldsymbol{x}}}$. If the distribution is over the codewords (or symbols) $\boldsymbol{x}$, 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 $\tilde{\boldsymbol{x}}$. 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 $p_{\boldsymbol{x}}$, $\mathcal{H}_x$, is equal to $\overline{\mathcal{H}}_{\boldsymbol{x}}$, and the difference is later shown to be the code's redundancy. The discrete-distribution-based received symbol's entropy $\mathcal{H}_{\boldsymbol{y}}$ has similar definition, $E\{\log_2 [1/p_{\boldsymbol{y}}]\}$. 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* $p_{\boldsymbol{x}} = \frac{1}{M}$ *maximizes the entropy.*

Proof: The Lagrangian for maximizing the (concave) Equation (2.71)'s entropy subject to the probability distribution's unit-sum constraint is

$$\mathcal{L} = -\sum_{i=1}^{M} p_i \cdot \log_2(p_i) + \lambda \cdot \left[\sum_{i=1}^{M} p_i - 1\right] \quad , \tag{2.72}$$

which has derivative for p_i equal to

$$-\frac{\ln(p_i)}{\ln 2} - \frac{1}{\ln 2} + \lambda \quad . \tag{2.73}$$

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

$$\max_{p_{\boldsymbol{x}}} \mathcal{H}_{\boldsymbol{x}} = \log_2(M) = b \quad . \tag{2.74}$$

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 ($p_{\boldsymbol{x}}(i) = 1$ for one value of i and $p_{\boldsymbol{x}}(j) = 0\ \forall\ j \neq i$) has no information, and thus zero entropy, $\mathcal{H}_{\boldsymbol{x}} = 0$. 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 $\mathcal{H}_{\boldsymbol{x}} = 2$ 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, $\mathcal{H}_{\tilde{\boldsymbol{x}}} \leq \log_2(|C|)$, 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 $\mathcal{H}_{\boldsymbol{x}} = 5 = b$ bits/two-dimensional-symbol. However, in one dimension, this constellation has the one-dimensional subsymbol values ± 5 each occurring with probability $\frac{1}{8}$ while the subsymbols ± 1 or ± 3 each occur with probability $\frac{3}{16}$, leading to a one-dimensional entropy

$$\mathcal{H}_x = \frac{2}{8} \cdot \log_2(8) + \frac{6}{8} \cdot \log_2(\frac{16}{3}) = 2.56 > \bar{b} = 2.5 = \overline{\mathcal{H}}_{\boldsymbol{x}} \text{ bits/dim} \quad . \tag{2.75}$$

The one-dimensional entropy $\mathcal{H}_x$ 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 $\bar{\mathcal{H}}_{\boldsymbol{x}}$=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 $\max \mathcal{H}_x - \bar{b}$ since $\max \bar{\mathcal{H}}_{\boldsymbol{x}} = \log_2 |C|$. 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 $\mathcal{H}_{\tilde{\boldsymbol{x}}}$ as

$$\mathcal{H}_{\tilde{\boldsymbol{x}}} \leq \log_2 |C| = \tilde{b} + \tilde{\rho} \quad , \tag{2.76}$$

with equality to $\tilde{b}$ if and only if $p_{\tilde{\boldsymbol{x}}}$ is uniform. Equivalently, an uncoded system has zero redundancy and entropy thus equal to $\tilde{b}$.

A good N-dimensional code with symbol/codeword $\boldsymbol{x}$ over the full N dimensions will thus typically have $\mathcal{H}_{\boldsymbol{x}} = \log_2(M) = b$ 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":

Definition 2.3.2 [Differential Entropy] *The continuous random variable's distribution $p_{\boldsymbol{x}}(u)$ has differential entropy defined as*

$$\mathscr{H}_{\boldsymbol{x}} \stackrel{\Delta}{=} -\int_{-\infty}^{\infty} p_{\boldsymbol{x}}(u) \cdot \log_2 \left[p_{\boldsymbol{x}}(u)\right] \cdot du \quad . \tag{2.77}$$

Distinguishing Entropy and Differential Entropy: This text adds the qualifying adjective "differential," and uses a script $\mathscr{H}$ notation, because the entropy $\mathcal{H}_{\boldsymbol{x}}$'s limiting-summation value in (2.70) that corresponds to infinitesimally small probability-distribution partitioning will be infinite[34]. But, $\mathscr{H} < \infty$; differential entropy is different than entropy. For instance, the differential entropy for a random uniform x on any interval of length $d < 1$ produces $\mathscr{H} < 0$, while a discrete uniform distribution with large $|C|$ approximating a continuous uniform would have very large entropy, $\mathcal{H}_{\boldsymbol{x}} \to \infty$. Further, a scaled version of a continuously distributed random variable $a \cdot x$ has differential entropy $\mathscr{H}_{ax} = \mathscr{H}_x + \log_2 |a|$, which generalizes to $\mathscr{H}_{A\boldsymbol{x}} = \mathscr{H}_{\boldsymbol{x}} + \log_2 |A|$ when A is a square nonsingular matrix; but for a discrete distribution $\mathcal{H}_{ax} = \mathcal{H}_x$ and $\mathcal{H}_{A\boldsymbol{x}} = \mathcal{H}_{\boldsymbol{x}}$. Even more generally if $\boldsymbol{x}$ is mapped into some $\boldsymbol{x}'$, when f is invertible and one-to-one, such that $\boldsymbol{x}' = f(\boldsymbol{x})$, then $\mathscr{H}_{\boldsymbol{x}'} = \mathscr{H}_{\boldsymbol{x}} + \int p(\boldsymbol{x}) \log_2 \{|\partial f / \partial \boldsymbol{x}|\} \cdot d\boldsymbol{x}$ where $|\partial f / \partial \boldsymbol{x}|$ is the Jacobian determinant, again showing a difference for the continuous $\mathscr{H}$ 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 $\mathscr{H}$ appear) to quantities that correspond to the difference between between $\mathcal{H}_{\boldsymbol{x}}$ and an upcoming conditional entropy that will however have the same constant offset. The common offset in the $\mathscr{H}$'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 $\mathscr{H}$ 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 $\mathbb{E}\left[\int |x|^2 dx\right] \leq \mathcal{E}_{\boldsymbol{x}}$, and codewords roughly lie within a sphere. As $N \to \infty$, 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 $\widetilde{N}$-dimensional) marginal distribution of such a uniform spherical distribution is thus Gaussian in any subsymbol:

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

[34] The correct limiting value essentially would have a $p_{\boldsymbol{x}} \cdot d\boldsymbol{x}$ ALSO inside the $\log_2(\cdot)$ term in order to retain its proper probabilty-mass-function properties, and differential entropy omits the extra $d\boldsymbol{x}$ term.

Proof: Let $g_x(u)$ denote the Gaussian distribution, then

$$\log_2 g_x(u) = -\log_2\left(\sqrt{2\pi\bar{\mathcal{E}}_{\boldsymbol{x}}}\right) - \left(\frac{u}{\sqrt{2\bar{\mathcal{E}}_{\boldsymbol{x}}}}\right)^2 \cdot (\ln(2))^{-1} \quad . \tag{2.78}$$

For any other distribution $p_x(u)$ with mean zero and the same given variance $\bar{\mathcal{E}}_n$

$$-\int_{-\infty}^{\infty} p_x(u) \cdot \log_2\left(g_x(u)\right) \cdot du = \log_2\left(\sqrt{2\pi\bar{\mathcal{E}}_{\boldsymbol{x}}}\right) + \frac{1}{2\ln(2)} \tag{2.79}$$

$$= \mathscr{H}(g_x) \quad , \tag{2.80}$$

which depends only on $\bar{\mathcal{E}}_{\boldsymbol{x}}$. Then, letting the distribution for x be an argument for the entropy,

$$\begin{aligned}
\mathscr{H}_x(g_x) - \mathscr{H}_x(p_x) &= -\int_{-\infty}^{\infty} g_x(u) \cdot \log_2(g_x(u)) \cdot du + \int_{-\infty}^{\infty} p_x(u) \cdot \log_2(p_x(u)) \cdot du \\
&\quad \text{now using (2.79) to obtain} \qquad (2.81) \\
&= -\int_{-\infty}^{\infty} p_x(u) \cdot \log_2(g_x(u)) \cdot du + \int_{-\infty}^{\infty} p_x(u) \cdot \log_2(p_x(u)) \cdot du \\
&= -\int_{-\infty}^{\infty} p_x(u) \cdot \log_2\left(\frac{g_x(u)}{p_x(u)}\right) \cdot du \qquad (2.82) \\
&\geq \frac{1}{\ln 2}\int_{-\infty}^{\infty} p_x(u) \cdot \left(1 - \frac{g_x(u)}{p_x(u)}\right) \cdot du \qquad (2.83) \\
&\geq \frac{1}{\ln 2}(1-1) = 0 \quad , \qquad (2.84)
\end{aligned}$$

or[35]

$$\mathscr{H}_x(g_x) \geq \mathscr{H}_x(p_x) \quad . \tag{2.85}$$

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

Thus, the marginal distributions for dimensions (or subsymbols) of an average-energy-limited uniform distribution over the hypersphere tends to a Gaussian in any and all dimensions as $\bar{N} \to \infty$. This result holds for complex Gaussian (or two-dimensional Gaussian, which is the same) in two dimensions when $\widetilde{N} = 2$ and for any finite $\widetilde{N}$ as an $\widetilde{N}$-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 $\bar{\sigma}^2$ is

$$\mathscr{H}_x = \frac{1}{2}\log_2\left(2\pi e\bar{\sigma}^2\right) \quad \text{bits/dimension.} \tag{2.86}$$

A complex Gaussian variable with variance σ^2 has differential entropy

$$\mathscr{H}_{\boldsymbol{x}} = \log_2\left(\pi e\sigma^2\right) \quad \text{bits/subsymbol.} \tag{2.87}$$

Careful examination justifiably reveals that these two quantities in Equations (2.86) and (2.87) 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

$$\mathcal{H}_{\boldsymbol{x},\boldsymbol{y}} = -E_{\boldsymbol{x},\boldsymbol{y}}\left[\log_2\left(p_{\boldsymbol{x},\boldsymbol{y}}\right)\right] \tag{2.88}$$

$$\mathscr{H}_{\boldsymbol{x},\boldsymbol{y}} = -E_{\boldsymbol{x},\boldsymbol{y}}\left[\log_2\left(p_{\boldsymbol{x},\boldsymbol{y}}\right)\right] \quad , \tag{2.89}$$

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

The **conditional entropy** of one random variable given another is

$$\mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}} \triangleq \sum_{\boldsymbol{v}}\sum_{i=0}^{M-1} p_{\boldsymbol{x}}(i)\cdot p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{v},i)\cdot\log_2\frac{1}{\mathrm{p}_{\boldsymbol{x}/\boldsymbol{y}}(\boldsymbol{v},i)} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\left\{\log_2\left[1/p_{\boldsymbol{x}/\boldsymbol{y}}\right]\right\} \tag{2.90}$$

$$\mathcal{H}_{\boldsymbol{y}/\boldsymbol{x}} \triangleq \sum_{\boldsymbol{v}}\sum_{i=0}^{M-1} p_{\boldsymbol{x}}(i)\cdot p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{v},i)\cdot\log_2\frac{1}{p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{v},i)} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}\left\{\log_2\left[1/p_{\boldsymbol{y}/\boldsymbol{x}}\right]\right\} \quad , \tag{2.91}$$

with integrals replacing summations when random variables/vectors have continuous distributions, and $\mathcal{H}\rightarrow\mathscr{H}$. 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 $p_{\boldsymbol{y}/\boldsymbol{x}}$ and $p_{\boldsymbol{x}}$. More generally, conditional entropy measures a random variable's average residual information given the value of another random variable, just as a MMSE measures such randomness for two jointly Gaussian vectors.

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

$$\mathcal{H}_{\boldsymbol{x},\boldsymbol{y}} = \mathcal{H}_{\boldsymbol{x}} + \mathcal{H}_{\boldsymbol{y}/\boldsymbol{x}} = \mathcal{H}_{\boldsymbol{y}} + \mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}} \tag{2.92}$$

$$\mathscr{H}_{\boldsymbol{x},\boldsymbol{y}} = \mathscr{H}_{\boldsymbol{x}} + \mathscr{H}_{\boldsymbol{y}/\boldsymbol{x}} = \mathscr{H}_{\boldsymbol{y}} + \mathscr{H}_{\boldsymbol{x}/\boldsymbol{y}} \quad , \tag{2.93}$$

which also notes the symmetry in interchangeability of chain-rule order. The joint entropy does not depend on the random-vector components' order. From (2.92), $\mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}}\leq\mathcal{H}_{\boldsymbol{x}}$ with equality only when $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent. This same chain rule applies equally to differential entropy.

For a communication channel characterized by $p_{\boldsymbol{y}/\boldsymbol{x}}$, the conditional entropy, $\mathcal{H}_{\boldsymbol{y}/\boldsymbol{x}}$, basically generalizes 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

$$\mathscr{H}_{x/\boldsymbol{y}} = \begin{cases} \frac{1}{2}\log_2\left(2\pi\cdot e\cdot\sigma^2_{mmse}\right) & x\in\mathbb{R} \\ \log_2\left(\pi\cdot e\cdot\sigma^2_{mmse}\right) & x\in\mathbb{C} \end{cases} \tag{2.94}$$

Equation 2.94's "MMSE" arises from the minimum-mean-square-error (MMSE) estimation of x, given $\boldsymbol{y}$ (See Appendix D). Thus, the conditional entropy measures the information remaining after the effect of $\boldsymbol{y}$ has been removed through MMSE estimation with Gaussian processes. The conditional entropy thus measures information that a receiver cannot estimate about x.

The following extends Lemma 2.3.2 to any autocorrelation matrix between two scalars or vectors[36]

$$R_{[\boldsymbol{x}\boldsymbol{y}]} \triangleq E\left[\boldsymbol{x}^*\boldsymbol{y}^*\right]\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix} = \begin{bmatrix} R_{\boldsymbol{x}\boldsymbol{x}} & R^*_{\boldsymbol{x}\boldsymbol{y}} \\ R_{\boldsymbol{x}\boldsymbol{y}} & R_{\boldsymbol{y}\boldsymbol{y}}\end{bmatrix} \quad . \tag{2.95}$$

The vector MMSE is the determinant of this matrix

$$\mid R_{\boldsymbol{e}\boldsymbol{e}}\mid = \mid R_{\boldsymbol{x}\boldsymbol{x}} - R_{\boldsymbol{x}\boldsymbol{y}}\cdot R^{-1}_{\boldsymbol{y}\boldsymbol{y}}R^*_{\boldsymbol{x}\boldsymbol{y}}\mid \quad , \tag{2.96}$$

as also in Appendix D.

[36] Note the brackets in the subscript of the defined matrix $R_{[\boldsymbol{x}\boldsymbol{y}]}$ to distinguish it from one of its components the cross-correlation between $\boldsymbol{x}$ and $\boldsymbol{y}$ as $R_{\boldsymbol{x}\boldsymbol{y}}$.

Lemma 2.3.3 [Maximum Entropy for Multiple Vector Continuous Distributions] *The Gaussian probability distribution*

$$p_{[\boldsymbol{x},\boldsymbol{y}]}(\boldsymbol{u},\boldsymbol{v})) = \frac{1}{\sqrt{\pi |R_{[\boldsymbol{xy}]}|^{1/(\widetilde{N}_x+\widetilde{N}_y)\cdot}}} \cdot e^{-\left\{ [\boldsymbol{u}^*\boldsymbol{v}^*]\cdot R^{-1}_{[\boldsymbol{xy}]}\cdot \left[\begin{array}{c} \boldsymbol{u} \\ \boldsymbol{v} \end{array} \right] \right\}} \tag{2.97}$$

maximizes the subsymbol differential entropy for any given positive semi-definite $R_{[\boldsymbol{xy}]}$.

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

$$\log_2 g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) = -\log_2 \left(\sqrt{\pi |R_{[\boldsymbol{xy}]}|^{1/1/(\widetilde{N}_x+\widetilde{N}_y)}} \right) - \left([\boldsymbol{u}^*\boldsymbol{v}^*] \cdot R^{-1}_{[\boldsymbol{xy}]} \cdot \left[\begin{array}{c} \boldsymbol{u} \\ \boldsymbol{v} \end{array} \right] \right) \cdot (\ln(2))^{-1} \quad .$$

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

$$\begin{aligned} & -\int_{-\infty}^{\infty} p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \cdot \log_2 \left(g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \right) \cdot d[\boldsymbol{uv}] = \log_2 \left(\sqrt{\pi \mid R_{[\boldsymbol{xy}]} \mid^{1/1/(\widetilde{N}_x+\widetilde{N}_y)}} \right) + \frac{1}{\ln(2)} \\ = \quad & \mathscr{H}(g_{[\boldsymbol{xy}]}) \quad , \end{aligned} \tag{2.98}$$

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

$$\begin{aligned} \mathscr{H}_{[\boldsymbol{xy}]}(g_{[\boldsymbol{xy}]}) - \mathscr{H}_{[\boldsymbol{xy}]}(p_{[\boldsymbol{xy}]}) &= -\int_{-\infty}^{\infty} g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \cdot \log_2(g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})) \cdot d[\boldsymbol{uv}] \quad (2.99) \\ &\quad + \int_{-\infty}^{\infty} p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \cdot \log_2(g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})) \cdot d[\boldsymbol{uv}] \\ &\quad \text{now using (2.98) to obtain} \quad (2.100) \\ &= -\int_{-\infty}^{\infty} p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \cdot \log_2(g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})) \cdot d[\boldsymbol{uv}] \\ &\quad + \int_{-\infty}^{\infty} p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \cdot \log_2(p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})) \cdot du \\ &= -\int_{-\infty}^{\infty} p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \cdot \log_2 \left(\frac{g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})}{p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})} \right) \cdot d[\boldsymbol{uv}] \\ &\geq \frac{1}{\ln 2} \int_{-\infty}^{\infty} p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) \cdot \left(1 - \frac{g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})}{p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})} \right) \cdot d[\boldsymbol{uv}] \\ &\geq \frac{1}{\ln 2}(1-1) = 0 \quad , \end{aligned}$$

or[37]

$$\mathscr{H}_x(g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})) \geq \mathscr{H}_x(p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})) \quad , \tag{2.101}$$

with equality only when $p_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v}) = g_{[\boldsymbol{xy}]}(\boldsymbol{u},\boldsymbol{v})$. 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 $\boldsymbol{y}$, this extra

noise information detracts from the possible input data rate. Thus, the entropy difference $\mathcal{H}_{\boldsymbol{y}} - \mathcal{H}_{\boldsymbol{y}/\boldsymbol{x}}$ measures the recoverable-data-rate component of $\boldsymbol{x}$ in the channel output. Equivalently, if $\mathcal{H}_{\boldsymbol{x}}$ is the input entropy, then $\mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}}$ measures the distortion in $\boldsymbol{x}$ that is not estimable from $\boldsymbol{y}$. 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 $p_{\boldsymbol{x}}(i)$ $i = 0, ..., M-1$, and a corresponding channel description $p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{v}, i)$, is:*

$$\mathcal{I}_{\boldsymbol{x},\boldsymbol{y}} \triangleq \mathcal{H}_{\boldsymbol{x}} - \mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}} = E\left\{\log_2\left[\frac{p_{\boldsymbol{x},\boldsymbol{y}}}{p_{\boldsymbol{x}} \cdot p_{\boldsymbol{y}}}\right]\right\} \geq 0 \quad , \tag{2.102}$$

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

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

$$\mathcal{I}_{\boldsymbol{x},\boldsymbol{y}} = \mathcal{H}_{\boldsymbol{y}} - \mathcal{H}_{\boldsymbol{y}/\boldsymbol{x}} = \mathcal{H}_{\boldsymbol{x}} - \mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}} \quad , \tag{2.103}$$

easily follows from transposing the "symmetric" $\boldsymbol{x}$ and $\boldsymbol{y}$ as follows:

$$\mathcal{I}_{\boldsymbol{x},\boldsymbol{y}} \triangleq \sum_{\boldsymbol{v}} \sum_{i=0}^{M-1} p_{\boldsymbol{x}}(i) \cdot p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{v}, i) \cdot \log_2\left[\frac{p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{v}, i)}{\sum_{m=0}^{M-1} p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{v}, m) \cdot p_{\boldsymbol{x}}(m)}\right] \quad \text{bits/symbol} \tag{2.104}$$

$$= E \log_2\left[\frac{p_{\boldsymbol{y}/\boldsymbol{x}}}{\mathrm{p}_{\boldsymbol{y}}}\right] \tag{2.105}$$

$$= E \log_2\left[\frac{p_{\boldsymbol{y},\boldsymbol{x}}}{p_{\boldsymbol{x}} \cdot p_{\boldsymbol{y}}}\right] \tag{2.106}$$

$$= E \log_2\left[\frac{p_{\boldsymbol{x}/\boldsymbol{y}}}{p_{\boldsymbol{x}}}\right] \quad , \tag{2.107}$$

In the case of a continuous distribution on $\boldsymbol{y}$ and/or $\boldsymbol{x}$, appropriate integrals replace 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):*

$$\mathcal{I}(\boldsymbol{x}; \boldsymbol{y}) = \sum_{i=1}^{U} I\left(\boldsymbol{x}_i; \boldsymbol{y}/[\boldsymbol{x}_1 ... \boldsymbol{x}_{i-1}]\right) \quad . \tag{2.108}$$

Proof: *From entropy chain rule:*

$$\mathscr{H}(\boldsymbol{x}) = \sum_{i=1}^{U} \mathscr{H}\left(\boldsymbol{x}_i; [\boldsymbol{x}_1 ... \boldsymbol{x}_{i-1}]\right) \text{ and} \tag{2.109}$$

$$\mathscr{H}(\boldsymbol{x}/\boldsymbol{y}) = \sum_{i=1}^{U} \mathscr{H}\left(\boldsymbol{x}_i/\boldsymbol{y}, [\boldsymbol{x}_1 ... \boldsymbol{x}_{i-1}]\right) \quad . \tag{2.110}$$

Subtraction of (2.110) from (2.109) completes the proof, which could be repeated with $\mathcal{H} \to \mathscr{H}$. **QED.**

The chain rule would also apply to any subset of $\boldsymbol{x}$'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.

Lemma 2.3.5 [Preservation of Information under invertible transformation] *If either or both of $\boldsymbol{x}$ and $\boldsymbol{y}$ undergo invertible transformations to $\boldsymbol{u}$ and $\boldsymbol{v}$ respectively, the mutual information (and the corresponding entropies) do not change, equivalently*

$$\mathcal{I}(\boldsymbol{u};\boldsymbol{v}) = \mathcal{I}(\boldsymbol{x};\boldsymbol{y}) \quad . \tag{2.111}$$

Proof: Both entropy (including differential) and mutual information are functions only of their own (joint) probability distributions evaluated and averaged. An invertible transformation $\boldsymbol{u} = f(\boldsymbol{x}) \ni \boldsymbol{x} = f^{-1}(\boldsymbol{u})$ and $\boldsymbol{v} = g(\boldsymbol{y}) \ni \boldsymbol{x} = g^{-1}(\boldsymbol{v})$ 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 $df = d\boldsymbol{u}$ and $dg = d\boldsymbol{v}$ respectively but reduce to the original integrals under change of variable substitution. **QED.**

2.3.2.1 MMSE and Gaussian Random Processes

Appendix D describes MMSE estimation's strong linear-AWGN-channel connection with Gaussian random variables, and Section D.3 extends readily to stationary Gaussian random processes for any point in time. The overall (Gaussian case) MMSE estimate $\hat{\boldsymbol{x}} = \mathbb{E}[\boldsymbol{x}/\boldsymbol{y}]$ is linear, so $\hat{\boldsymbol{x}} = \boldsymbol{w} \cdot \boldsymbol{y}$ where $\boldsymbol{w}$ is a row vector/matrix with estimator coefficients for each and every dimension of $\boldsymbol{y}$. Because $\boldsymbol{x}$ in particular is stationary, Section D.3 extends MMSE estimates to include past, or more generally other, sample values as the estimator input, so $\hat{\boldsymbol{x}}_k = \mathbb{E}[\boldsymbol{x}_k/\boldsymbol{x}_{i\neq k}]$, which for Gaussian processes is again linear in the other samples $i \neq k$, when $i = k-1, k-2, \ldots$. The entropy of Gaussian random process is solely an invertible function of its linear-prediction MMSE. Section D.3 shows that the MMSE linear-prediction estimate, under a Paley Weiner Criterion (PWC) existence criterion, is a causal, causally invertible, monic function of past values, and that in finite-length cases this linear prediction corresponds to a monic upper-triangular Cholesky factor (G_x) of the autocorrelation matrix $R_{\boldsymbol{x}\boldsymbol{x}} = G_x \cdot S_x \cdot G_x^*$. A "finite PWC" is essentially that $R_{\boldsymbol{x}\boldsymbol{x}}$ be nonsingular so any zero-energy input dimensions are removed from its description without change of information/entry content.

Mutual information's chain rule for $\mathcal{I}(\boldsymbol{x};\boldsymbol{y})$ has both $\boldsymbol{y}$ for the MMSE estimate SNR of $\boldsymbol{x}$ given $\boldsymbol{y}$, but also decomposes into chain-rule component terms that have $\boldsymbol{x}_{i<k}$. Thus the MMSE estimate essentially becomes a set of data rates corresponding to estimates of the form $\hat{\boldsymbol{x}}_k = \boldsymbol{w} \cdot \boldsymbol{y} + \boldsymbol{g} \cdot \boldsymbol{x}_{k-1,\ldots,k-L}$. Indeed, this chain-rule sum effectively corresponds to a series of independent AWGNs, each with its own SNR_k. Indeed this works for any finite-dimensional $R_{\boldsymbol{x}\boldsymbol{x}}$ and yields to a need for for only nonsingular $R_{\boldsymbol{n}\boldsymbol{n}}$ on a matrix AWGN. As long as subsymbols $\boldsymbol{x}_k$ have constant (block-stationary) $R_{\boldsymbol{x}\boldsymbol{x}}$, the dimensions within a finite block can be reordered in any possible sequence and the same mutual information is achieved with different sets of independent subchannels because by the chain rule they all add to the same mutual information. This text in Sections 2.6 - 2.10 as well as in Chapters 3 - 5 make heavy use of this chain-rule, mutual-information, and MMSE connection for Gaussian channels. But first a more general relationship develops for any channel conditional (joint with input) probability distribution in the upcoming Subsection 2.3.3.

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. The LLN's stationarity assumption views a codeword's subsymbols as temporal

for AEP development, but the AEP will independently apply to stationary vector sequences of a MIMO system with input distribution $p_{\boldsymbol{x}}$ representing whatever the input subsymbol distribution may be. 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 $\overline{N} \to \infty$. AEP generalizes this random-coding distribution, and of course maximization over all allowed input distribution choices then produces a more general capacity result. AEP basically generalizes the previous subsection's rich MMSE-mutual-information connection for matrix AWGNs to any channel, which allows general address of many communication design issues in theory but potentially large complexity.

Random Code Design: Random-code design computes a codeword-distribution's entropy per subsymbol through random and **independent** selections from the same stationary subsymbol distribution $p_{\tilde{\boldsymbol{x}}}$, $\overline{N}$ 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 $p_{\tilde{\boldsymbol{x}}}$. First, using $\log_2(p_{\boldsymbol{x}}) = \log_2\left(\prod_{n=1}^{\overline{N}} p_{\tilde{\boldsymbol{x}}_n}\right)$, the randomly constructed vector-codeword symbols has per-symbol entropy

$$\widetilde{\mathcal{H}}_{\boldsymbol{x}} = \frac{-1}{\overline{N}} \cdot \mathbb{E}\left[\log_2(p_{\boldsymbol{x}})\right] \tag{2.112}$$

$$= \frac{-1}{\overline{N}} \sum_{n=1}^{\overline{N}} \mathbb{E}\left[\log_2(p_{\tilde{\boldsymbol{x}}_n})\right] \quad , \tag{2.113}$$

which is also $\mathbb{E}\left[\log_2(p_{\tilde{\boldsymbol{x}}_n}^{-1})\right] = \mathcal{H}_{\tilde{\boldsymbol{x}}}$ for a stationary distribution $p_{\tilde{\boldsymbol{x}}_n} = p_{\tilde{\boldsymbol{x}}}\ \forall\ n = 1, ..., \bar{N}$. In random coding, $\rho_{\boldsymbol{x}} = 0$, but $\rho_{\tilde{\boldsymbol{x}}} > 0$. When $p_{\tilde{\boldsymbol{x}}}$ is continuous, then $\widetilde{\mathscr{H}}$ replaces $\widetilde{\mathcal{H}}$ in (2.112) and (2.113), as well as throughout all AEP discussion.

Entropy as a Sample Average: The entropy $\widetilde{\mathcal{H}}_{\boldsymbol{x}}$ is the average number of bits/dimension for such a randomly constructed code (equivalently that represents a source with this same distribution) $\boldsymbol{x}$; or stated equivalently, the random-code design process selects, on average, a number of codewords $\boldsymbol{x} \in C_{\boldsymbol{x}}$ that is $2^{\overline{N} \cdot \widetilde{\mathcal{H}}_{\boldsymbol{x}}}$. The probability distribution $p_{\tilde{\boldsymbol{x}}_n}$ is also a function of the random variable $\tilde{\boldsymbol{x}}_n$ itself for application of the LLN (See Theorem 2.1.1), and thus $p_{\tilde{\boldsymbol{x}}_n} \to p(\tilde{\boldsymbol{x}}_n)$ is a random variable also in this view: Equation (2.113) then also has interpretation as the log-inverse-probability distribution's sample-average value for random variable $1/p(\tilde{\boldsymbol{x}}_n)$ over each and every codeword in the random code-design process:

$$\begin{aligned}
\hat{\widetilde{\mathcal{H}}}_{\tilde{\boldsymbol{x}}} &= \frac{-1}{\overline{N}} \cdot \sum_{n=1}^{\overline{N}} \log_2\left[p(\tilde{\boldsymbol{x}}_n)\right] \\
&= \frac{-1}{\overline{N}} \cdot \log_2\left[\prod_{n=1}^{\overline{N}} p(\tilde{\boldsymbol{x}}_n)\right] \\
&= \frac{-1}{\overline{N}} \cdot \log_2\left[\, p_{\boldsymbol{x}}\,\right] \quad .
\end{aligned} \tag{2.114}$$

(This same statement applies also to joint entropy and conditional entropy with sample averages over both $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$, and thus also to their difference, the mutual information.) The LLN then states that the asymptotic ($\overline{N} \to \infty$) sample-average value $\hat{\widetilde{\mathcal{H}}}_{\tilde{\boldsymbol{x}}}$ will almost surely have a value equal to $\widetilde{\mathcal{H}}_{\boldsymbol{x}}$ in (2.113) for such a codeword, and any other codeword that might not have value equal to $\widetilde{\mathcal{H}}_{\boldsymbol{x}}$ will be

asymptotically negligible with arbitrarily small probabability. Thus from this LLN viewpoint for a code $\boldsymbol{x} \in C_{\boldsymbol{x}}$ with randomly selected subsymbols $\tilde{\boldsymbol{x}}$ from $p_{\tilde{\boldsymbol{x}}}$:

$$\hat{\widetilde{\mathcal{H}}}_{\boldsymbol{x}} \quad \to \widetilde{\mathcal{H}}_{\boldsymbol{x}} \quad \to \mathcal{H}_{\tilde{\boldsymbol{x}}} \tag{2.115}$$

$$\hat{\widetilde{\mathcal{H}}}_{\boldsymbol{x}/\boldsymbol{y}} \quad \to \widetilde{\mathcal{H}}_{\boldsymbol{x}/\boldsymbol{y}} \quad \to \mathcal{H}_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}} \tag{2.116}$$

$$\hat{\widetilde{\mathcal{I}}}\left(\boldsymbol{x};\boldsymbol{y}\right) \quad \to \widetilde{\mathcal{I}}\left(\boldsymbol{x};\boldsymbol{y}\right) \quad \to \mathcal{I}\left(\tilde{\boldsymbol{x}};\tilde{\boldsymbol{y}}\right) \tag{2.117}$$

Then, because $\log_2(p_{\boldsymbol{x}})$ 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 $\widetilde{\mathcal{H}}_{\boldsymbol{x}}$ 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 $\overline{N} \to \infty$. 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 independent sampling from the same distribution repeatedly, is that each of these increasingly large codewords must asymptotically have the same uniform-distribution probability $2^{-N\cdot\overline{\mathcal{H}}\boldsymbol{x}}$, or equivalently the size of the typical set approaches $2^{N\cdot\overline{\mathcal{H}}\boldsymbol{x}}$ symbols with uniform distribution.[38]. More formally, with $\widetilde{N}$-dimensional subsymbols:

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

$$A_{\bar{N}}^{\epsilon}(\boldsymbol{x}) \triangleq \left\{ \boldsymbol{x} = [\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_{\bar{N}}] \;\middle|\; 2^{-\bar{N}\cdot\mathcal{H}_{\tilde{\boldsymbol{x}}}-\epsilon} \leq p(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_{\bar{N}}) \leq 2^{-\bar{N}\cdot\mathcal{H}_{\tilde{\boldsymbol{x}}}+\epsilon} \right\} \tag{2.118}$$

for any $\epsilon > 0$.

The AEP Lemma follows from the above discussion:

Lemma 2.3.6 [AEP Lemma] *For a typical set with $\bar{N} \to \infty$, the following are true:*

- $Pr\{A_{\bar{N}}^{\epsilon}(\boldsymbol{x})\} \to 1$
- *for any codeword $\boldsymbol{x} \in A_{\bar{N}}^{\epsilon}$, $Pr\{\boldsymbol{x}\} \to 2^{-\bar{N}\cdot\mathcal{H}_{\tilde{\boldsymbol{x}}}}$*

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

[38] 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.

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

$$\mathcal{H}_{\tilde{\boldsymbol{x}}} = \tilde{b} + \tilde{\rho} \quad . \tag{2.119}$$

The information content of a subsymbol exceeds the message information by the code's redundancy/subsymbol under AEP. The AEP typical set $A_N^{\epsilon}(\boldsymbol{x})$ implies a maximum codeword entropy for equally likely codewords. The designer can choose the $\tilde{b} \leq \mathcal{H}_{\tilde{\boldsymbol{x}}}$, and thus $b = \overline{N} \cdot \tilde{b}$, 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 $p_{\boldsymbol{y}/\boldsymbol{x}}$ characterizes the general channel, and the input distribution $p_{\boldsymbol{x}}$ need no longer have an input-energy constraint (just more generally any input constraints that restrict $p_{\boldsymbol{x}}$). A code selects codewords through M random, length-$\overline{N}$, independent $\tilde{\boldsymbol{x}}$ sample sets; each codeword has $\overline{N}$ selections from the subsymbol channel-input distribution $p_{\tilde{\boldsymbol{x}}}$. 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 $\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = \mathcal{H}_{\boldsymbol{x}} - \mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}}$, then

$$\frac{|A_N^{\epsilon}(\boldsymbol{x})|}{|A_N^{\epsilon}(\boldsymbol{x}/\boldsymbol{y})|} \to 2^{\mathcal{I}(\boldsymbol{x};\boldsymbol{y})} \quad , \tag{2.120}$$

suggestion that $2^{\mathcal{I}}$ essentially counts $A_{\overline{N}}^{\epsilon}(\boldsymbol{x}/\boldsymbol{y})$ subsets in $A_{\overline{N}}^{\epsilon}(\boldsymbol{x})$ and so may limit generally the rate of selection of the number of codewords asymptotically.

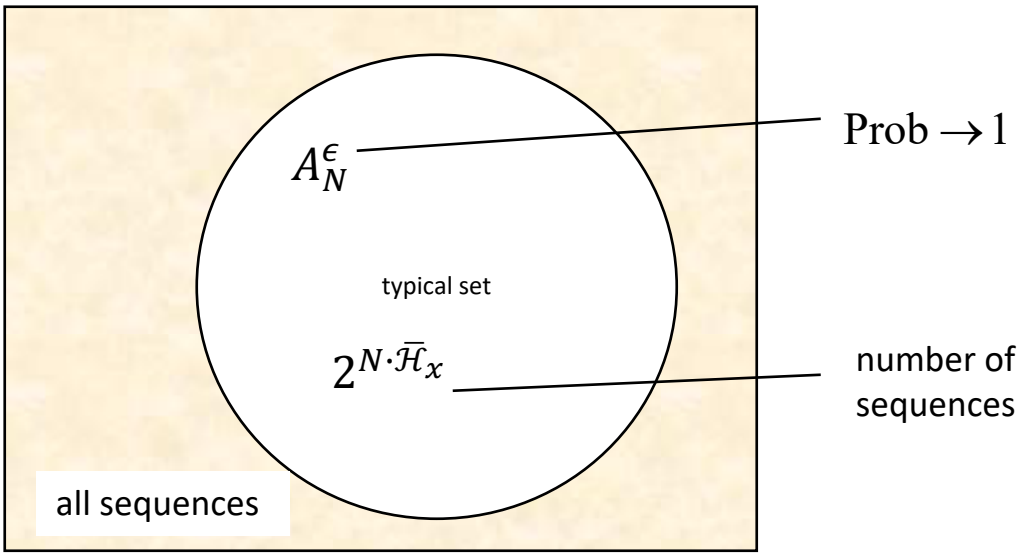


Figure 2.16: Illustration of sequence space and set of equal probability.

Venn Diagram Decision Regions: Figure 2.16 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 $2^{-\overline{N}\cdot\widetilde{\mathcal{H}}_{\tilde{\boldsymbol{x}}}}$. Figure 2.17 further illustrates that on average the conditional-typical-set size for $\boldsymbol{x}$ given $\boldsymbol{y}$ 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 $\boldsymbol{y}$. Thus a MAP detector likely errs if this $\boldsymbol{y}$-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 $2^{N\cdot\widetilde{\mathcal{H}}_{\boldsymbol{x}/\boldsymbol{y}}}$. 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 $2^{\overline{N}\cdot\widetilde{\mathcal{H}}_{\boldsymbol{x}}}$ codewords as $\overline{N}$ 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 $2^{\overline{N}\cdot\widetilde{\mathcal{H}}_{\boldsymbol{x}/\boldsymbol{y}}}$. Thus with this good code, the largest number of distinguishable codewords is

$$M = \frac{2^{\overline{N}\cdot\widetilde{\mathcal{H}}_{\boldsymbol{x}}}}{2^{\overline{N}\cdot\widetilde{\mathcal{H}}_{\boldsymbol{x}/\boldsymbol{y}}}} = 2^{\overline{N}\cdot\widetilde{\mathcal{I}}(\boldsymbol{x};\boldsymbol{y})} \quad . \tag{2.121}$$

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 (2.121). There may be more than 1 such good code; indeed with $\overline{N} \to \infty$, 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, $A_{\overline{N}}^{\epsilon}(\boldsymbol{x}/\boldsymbol{y})$, 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 $\boldsymbol{y}$ as the mapping of each $\boldsymbol{y}$ to the single $\boldsymbol{x} \in A_{\overline{N}}^{\epsilon}(\boldsymbol{x}/\boldsymbol{y})$. Thus, as $\overline{N} \to \infty$, $I(\boldsymbol{x};\boldsymbol{y})$ represents the maximum number of bits per subsymbol that can be reliably transmitted over the communication channel. That is,

$$b \leq I(\boldsymbol{x};\boldsymbol{y}) \quad , \tag{2.122}$$

or on a per-subsymbol basis

$$\tilde{b} < \mathcal{I}(\tilde{\boldsymbol{x}};\tilde{\boldsymbol{y}}) \; ; . \tag{2.123}$$

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 $A_{\overline{N}}^{\epsilon}(\boldsymbol{x}/\boldsymbol{y})$ each contain one and only one codeword $\boldsymbol{x}$. Generally, $\mathcal{I}_{\boldsymbol{x};\boldsymbol{y}} \leq \mathcal{H}_{\boldsymbol{x}}$ and less by at least the minimum redundancy $\rho_{\boldsymbol{x};\boldsymbol{y}} \geq \mathcal{H}_{\boldsymbol{x}/\boldsymbol{y}}$. Different input distributions can lead to different mutual information. The best input then has $p_{\boldsymbol{x}}$ that maximizes the mutual information, a concept first introduced by Shannon[39] in his 1948 paper[1]:

Definition 2.3.5 (Channel Capacity) *The channel capacity in bits/subsymbol for a channel described by $p_{\boldsymbol{y}/\boldsymbol{x}}$ is defined by*

$$\widetilde{\mathcal{C}} \triangleq \max_{p_{\tilde{\boldsymbol{x}}}} I(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}}) \;\; \textit{bits/subsymbol} \, . \tag{2.124}$$

[39] After Claude E. Shannon, 1916-2001, an American Mathematician known for pioneering information and communication theory.

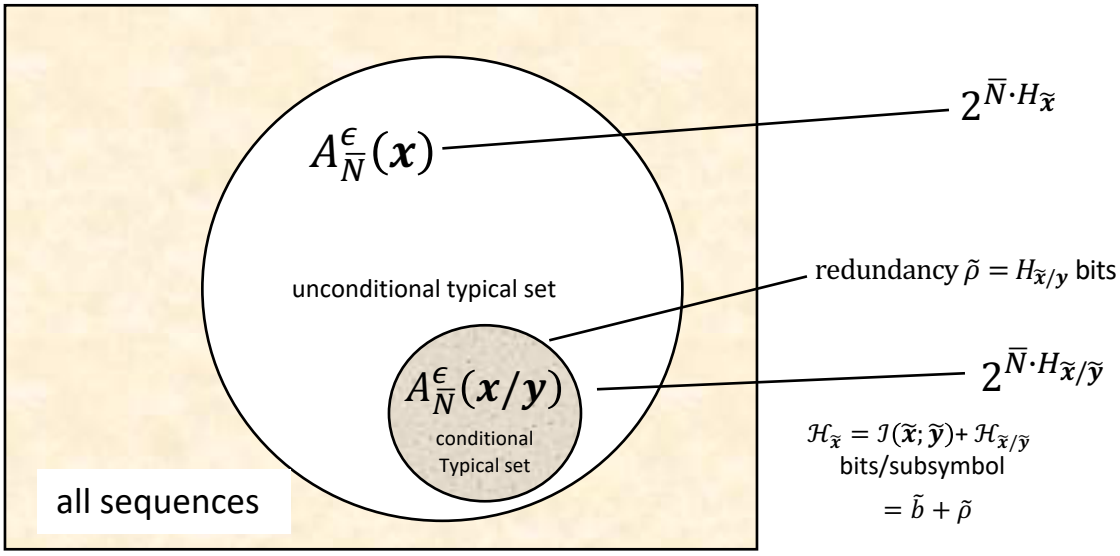


Figure 2.17: Illustration of sets corresponding to $\boldsymbol{x}$; and $\boldsymbol{x}$ conditioned on $\boldsymbol{y}$.

It is sometimes convenient to normalize $\widetilde{\mathcal{C}}$ to one dimension, by dividing $\widetilde{\mathcal{C}}$ by $\widetilde{N}$,

$$\overline{\mathcal{C}} \triangleq \frac{\widetilde{\mathcal{C}}}{\widetilde{N}} = \frac{\mathcal{C}}{N} \quad , \tag{2.125}$$

which is in bits/dimension. The capacity is measured in bits/dimension as $\overline{\mathcal{C}}$, bits/subsymbol $\widetilde{\mathcal{C}}$, or bits per codeword/symbol as $\mathcal{C}$. The calculation of (2.124) can be difficult in some cases, and may require numerical techniques to approximate the $\mathcal{C}$ value. Input constraints on the input $\boldsymbol{x}$ vectors' choice can affect the capacity value. Given the mutual information definition, Shannon's [1] famed channel coding theorem is:

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

proof: See the discussion surrounding Equations (2.121) and (2.122) and the AEP discussion preceding this theorem.

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

$$\widetilde{\mathcal{C}}_{awgn} = \overline{\mathcal{I}}(x;y) = \mathscr{H}_y - \mathscr{H}_{y/x} = \mathscr{H}_y - \log_2(\pi e \sigma^2) \quad , \tag{2.126}$$

which, because σ^2 is constant, is maximum when $\mathscr{H}_y$ 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

$$\widetilde{\mathcal{C}}_{awgn} = \tilde{I}(x;y) = \mathscr{H}_{x+n} - \log_2(\pi e \sigma^2) = \log_2\left(\frac{\bar{\mathcal{E}}_{\boldsymbol{x}} + \sigma^2}{\sigma^2}\right) = \frac{1}{2} \cdot \log_2(1 + SNR) \quad . \tag{2.127}$$

Some Gaussian Entropy Basics and Extension to MIMO: Given two complex jointly Gaussian random-vector subsymbols, $\tilde{\boldsymbol{x}}$ ($\widetilde{N}_x \times 1$) and $\tilde{\boldsymbol{y}}$ ($\widetilde{N}_y \times 1$) in a stationary vector random Gaussian process sequence of such variables, the conditional probability density $p_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}}$ also has a Gaussian probability density with mean $E[\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}] = R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{y}}} \cdot R_{\tilde{\boldsymbol{y}}}^{-1}{}_{\tilde{\boldsymbol{y}}} \cdot \tilde{\boldsymbol{y}}$ equal to the MMSE estimate of $\tilde{\boldsymbol{x}}$ given $\tilde{\boldsymbol{y}}$ as in Appendix D. (This result is easily proved as an exercise by simply taking the ratio of $p_{\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}}}$ to $p_{\tilde{\boldsymbol{y}}}$, which are both Gaussian with the general $(\widetilde{N}_x, \widetilde{N}_y)$-complex-dimensional $(2 \cdot (\widetilde{N}_x, \widetilde{N}_y)$ real dimensions) form. AEP style random codes, or random coding arguments, then apply to the selection of samples from $\boldsymbol{x}$'s stationary vector-Gaussian distribution. The (complex) Gaussian vector distribution is $1/(\pi^{\widetilde{N}_x}|R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|) \cdot e^{-(\boldsymbol{x}-\boldsymbol{u}_{\tilde{\boldsymbol{x}}})R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}{}^{-1}(\boldsymbol{x}^*-\boldsymbol{u}^*_{\tilde{\boldsymbol{x}}})}$, where $R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}$ is the subsymbol autocorrelation matrix and $\boldsymbol{u}_{\tilde{\boldsymbol{x}}}$ is the mean.) The differential entropy of a complex Gaussian subsymbol is

$$\mathscr{H}_{\tilde{\boldsymbol{x}}} = \log_2((\pi e)^{\widetilde{N}_x} \cdot |R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|) \quad , \tag{2.128}$$

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

$$H_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}} = \log_2((\pi e)^{\widetilde{N}_x} \left| R^{\perp}_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}} \right|) \quad , \tag{2.129}$$

where $R^{\perp}_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}} = R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}} - R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{y}}} \cdot R^{-1}_{\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}} \cdot R_{\tilde{\boldsymbol{y}}\tilde{\boldsymbol{x}}}$ is the autocorrelation matrix of the vector MMSE estimating $\tilde{\boldsymbol{x}}$ from $\tilde{\boldsymbol{y}}$, because $p_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}}$ also has a Gaussian distribution. For a scalar white Gaussian x, $\overline{\mathscr{H}}_x = \frac{1}{2} \cdot \log_2(2\pi e \bar{\mathcal{E}}_{\boldsymbol{x}})$ whether x is real or complex. Further, $\overline{R}_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}} = \frac{1}{2\widetilde{N}_x} \cdot R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}$. Again, per-real-dimensional quantities divide by the number of real dimensions $2\widetilde{N}_x$: $\overline{\mathscr{H}}_{\boldsymbol{x}} = \frac{1}{2}\log_2\{\pi \cdot e \cdot |R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|^{1/\widetilde{N}_x}\}$. If $\boldsymbol{x} = x$, i.e., a scalar, then $R_{\boldsymbol{x}\boldsymbol{x}} = 2\bar{\mathcal{E}}_{\boldsymbol{x}}$ in the entropy formula with $\widetilde{N}_x = 1$ and all per-dimensional results are consistent.[40]

For the scalar complex Gaussian case with $\widetilde{N}_x = 1$, and $\mathrm{SNR}_{mmse} \triangleq \bar{\mathcal{E}}_{\boldsymbol{x}}/\sigma^2_{mmse}$,

$$\begin{aligned}
\widetilde{\mathcal{C}}_{awgn} &= \log_2(1+\mathrm{SNR}) = \max\left(\mathscr{H}_y - \mathscr{H}_{y/x}\right) && (2.135)\\
&= \mathscr{H}_x - \mathscr{H}_{x/y} && (2.136)\\
&= \log_2(\pi e \mathcal{E}_x) - \log_2(\pi e \sigma^2_{mmse}) && (2.137)\\
&= \log_2(1+\mathrm{SNR}_{unbiased}) && (2.138)\\
&= \log_2(1+\mathrm{SNR}) \quad . && (2.139)
\end{aligned}$$

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

$$\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n} \quad , \tag{2.140}$$

[40] For the interested in alternative expressions (that provide the same entropy): If $\tilde{\boldsymbol{x}}$ is real, then $\mathscr{H}_{\tilde{\boldsymbol{x}}} = \frac{1}{2}\log_2\left[(2\pi e)^{2\widetilde{N}_x}|R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|\right]$ or

$$\begin{aligned}
\overline{\mathscr{H}}_{\tilde{\boldsymbol{x}}} &= \frac{1}{2\widetilde{N}_x}\log_2\left[(2\pi e)^{\widetilde{N}_x} \cdot |\widetilde{N}_x \bar{R}_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|\right] && (2.130)\\
&= \frac{1}{2}\log_2\left[(2\widetilde{N}_x\pi e) \cdot |\bar{R}_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|^{1/\widetilde{N}_x}\right] \quad , && (2.131)
\end{aligned}$$

which checks with one-dimensional formule. If $\tilde{\boldsymbol{x}}$ is complex, then $\widetilde{\mathscr{H}_{\tilde{\boldsymbol{x}}}} = \log_2\left[(\pi e)^{\widetilde{N}_x}|R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|\right]$ or

$$\begin{aligned}
\overline{H}_{\tilde{\boldsymbol{x}}} &= \frac{1}{2\widetilde{N}_x}\log_2\left[(\pi e)^{\widetilde{N}_x} \cdot |2\widetilde{N}_x \cdot \bar{R}_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|\right] && (2.132)\\
&= \frac{1}{2\widetilde{N}_x}\log_2\left[(\pi e)^{\widetilde{N}_x} \cdot (2\widetilde{N}_x)^{\widetilde{N}_x} \cdot |\bar{R}_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|\right] && (2.133)\\
&= \frac{1}{2}\log_2\left[(2\widetilde{N}_x\pi e) \cdot |\bar{R}_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|^{1/\widetilde{N}_x}\right] && (2.134)
\end{aligned}$$

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.

the mutual information is maximum when $\boldsymbol{x}$ is Gaussian; because if $\mathscr{H}_{\boldsymbol{x}}$ is Gaussian, then $\boldsymbol{y}$ is Gaussian following the expression in (2.126). Also, the minimum MSE error vector $\tilde{\boldsymbol{e}} = \tilde{\boldsymbol{x}} - R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{y}}} \cdot R_{\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}}^{-1} \cdot \boldsymbol{y}$ is then also Gaussian (linear combinations of Gaussians are Gaussian). The mutual-information formula is then (noting also the symmetry that $\mathcal{I} = \mathscr{H}_{\tilde{\boldsymbol{y}}} - \mathscr{H}_{\tilde{\boldsymbol{y}}/\tilde{\boldsymbol{x}}} = \mathscr{H}_{\tilde{\boldsymbol{x}}} - \mathscr{H}_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}}$)

$$\widetilde{\mathcal{I}}_{vector\ awgn} = \frac{1}{\widetilde{N}_x} \cdot \log_2 \left(\frac{|\ R_{\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}}\ |}{|\ R_{\tilde{\boldsymbol{n}}\tilde{\boldsymbol{n}}}\ |} \right) = \frac{1}{\widetilde{N}_x} \cdot \log_2 \left(\frac{|\ R_{\tilde{\boldsymbol{n}}\tilde{\boldsymbol{n}}} + H \cdot R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}} \cdot H^*\ |}{|\ R_{\tilde{\boldsymbol{n}}\tilde{\boldsymbol{n}}}\ |} \right) = \frac{1}{\widetilde{N}_x} \cdot \log_2 \left(\frac{|\ R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}\ |}{|\ R_{\tilde{\boldsymbol{e}}\tilde{\boldsymbol{e}}}\ |} \right) \quad . \tag{2.141}$$

The symbol $\widetilde{\mathcal{C}}_{vector\ awgn}$ later replaces $\widetilde{\mathcal{I}}_{vector\ awgn}$ in Subsection 2.3.5 that finds the best Gaussian distribution's autocorrelation matrix[41]. 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 (2.140), with specific autocorrelation $R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}$, a joint Gaussian distribution on all $\tilde{\boldsymbol{x}}$'s elements maximizes the mutual information $\mathcal{I}(\tilde{\boldsymbol{x}}; \tilde{\boldsymbol{y}})$.*

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 $p_{\tilde{\boldsymbol{x}}}$ in two real dimensions,

$$\mathscr{H}_{\tilde{\boldsymbol{x}}} = \log_2(6 \cdot \tilde{\mathcal{E}}_{\boldsymbol{x}}) \ \text{ uniform distribution} \quad , \tag{2.142}$$

in comparison to the larger value of $\mathscr{H}_{\tilde{\boldsymbol{x}}} = \log_2(\pi \cdot e \cdot \tilde{\mathcal{E}}_{\boldsymbol{x}})$ from Equation (2.94). For reasonable SNR, the MMSE estimate of $\tilde{\boldsymbol{x}}$ given $\boldsymbol{y}$ will be largely determined by the noise (which is Gaussian and independent of the input distribution). Thus, $\mathcal{H}_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}}$ 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

$$\text{loss SQ constellation} = \log_2(\pi \cdot e) - \log_2(6) \approx 0.5 \text{ bit/subsymbol} \ . \tag{2.143}$$

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

$$\mathcal{I}_{awgn,SQ} = \mathcal{C}_{awgn} - 0.5 \text{ bits/subsymbol} \quad . \tag{2.144}$$

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 (2.144). 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. While beyond this text's scope, peak-energy constraint on $\boldsymbol{x}$ has optimum **discrete** distribution $p_{\boldsymbol{x}}(i)$ (so not continuous nor Gaussian). This means the good code essentially can follow from QAM subsymbols, apart from the asymptotic shaping gain loss of $\gamma_s \leq 1.53$ dB, presuming any real system must have a peak instantaneous transmission constraint. Dr. Joel Smith first found peak-energy-limited optimizing distribution as discrete in his 1969 dissertation[42].

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

[42] after Joel Gorham Smith, 1935 - an Ameican engineer whose 1969 Berkeley dissertation first observed this optimizing discrete distribution. See also public document [11]

2.3.5 Simple Water-filling for the matrix AWGN

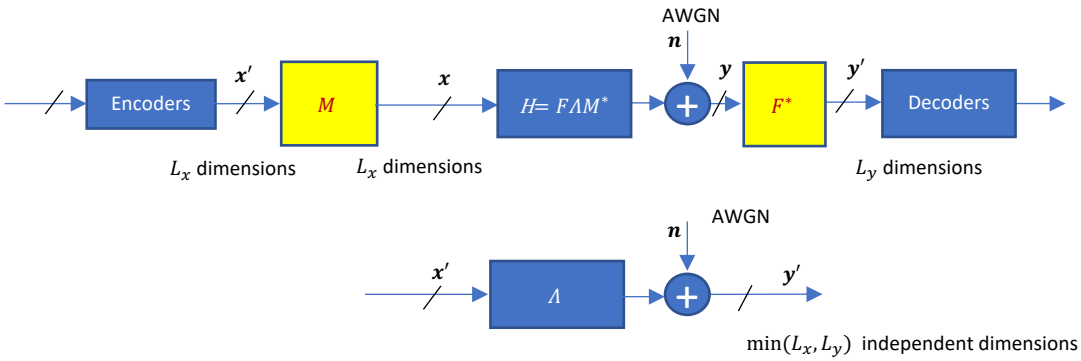


Figure 2.18: Simple Vector Coding "Parallelization" of a Filter AWGN with SVD.

The tilde notation on subsymbols is dropped in this subsection. Optimization over input distribution $p_{\boldsymbol{x}}$ can consider different $R_{\boldsymbol{xx}}$ for (2.140)'s channel, subject to constant input energy $trace\{R_{\boldsymbol{xx}}\} \leq \mathcal{E}_{\boldsymbol{x}}$. The matrix $R_{\boldsymbol{nn}}^{-1/2} \cdot H$ replaces H so that the noise-whitened channel equivalent has unit-variance noise in all (complex) dimensions. The $\widetilde{N}_y \times \widetilde{N}_x$ matrix H then has a singular value decomposition[43]

$$H = F \cdot \Lambda \cdot M^* \quad , \tag{2.145}$$

where the $\widetilde{N}_y \times \widetilde{N}_y$ matrix F is unitary, $FF^* = F^*F = I$, as is also the $\widetilde{N}_x \times \widetilde{N}_x$ matrix M, $MM^* = M^*M = I$ and Λ is a real (even if H is complex) diagonal-plus-zeros matrix. Figure 2.18 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 Λ when $\widetilde{N}_y > \widetilde{N}_x$ and to the right of diagonal Λ when $\widetilde{N}_x > \widetilde{N}_y$. The nonzero entries of Λ are the singular values. Transform of the input to $\boldsymbol{x}' = M \cdot \boldsymbol{x}$ does not change the energy, while transformation of the output to $\boldsymbol{y}' = F\boldsymbol{y}$ leaves $L^* = \min(\widetilde{N}_x, \widetilde{N}_y)$ nontrivial set of parallel independent "subchannel"

$$\boldsymbol{y}'_\ell = \lambda_\ell \cdot \boldsymbol{x}'_\ell + \boldsymbol{n}'_\ell \quad \ell = 1, ..., L^* \quad , \tag{2.146}$$

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 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

$$\mathcal{I}(\boldsymbol{x}; \boldsymbol{y}) = \sum_{\ell=1}^{L^*} \mathcal{I}(\boldsymbol{x}_\ell; \boldsymbol{y}_\ell) = \sum_{\ell=1}^{L^*} \log_2(1 + \mathcal{E}_\ell \cdot \lambda_\ell^2) \quad . \tag{2.147}$$

Maximization of this subject to the energy constraint yields

$$\mathcal{E}_\ell + \frac{1}{\lambda_\ell^2} = K \quad \wedge \quad \mathcal{E}_\ell \geq 0 \quad , \tag{2.148}$$

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 (2.148) is the **water-filling** energy distribution (studied in depth in Chapter 4). K derives from the total energy constraint $\mathcal{E} = trace\{R_{\boldsymbol{xx}}\} = \sum_{n=1}^{L^*} \mathcal{E}_n$, and then each energy is found from (2.148) before inserting the energies into (2.147) to compute the water-filling capacity.

[43] See Matlab "svd" command.

2.3.5.1 Simple Time/Dimension Sharing and Geometric SNRs

Vector coding's parallel subchannels most generally have different SNRs $\mathrm{SNR}_\ell = \mathcal{E}_\ell \cdot \lambda_\ell / \frac{\mathcal{N}_0}{2}$, each with it's own corresponding mutual information as in (2.147), so $\mathcal{I}_\ell \triangleq \mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = \log_2(1+SNR_\ell)$. Recognizing the sum-of-logs is also a log-of-products, $\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = \log_2(1+SNR_{geo})$ where

$$\mathrm{SNR}_{geo} \triangleq \left[\prod_\ell^L (1+\mathrm{SNR}_\ell)\right]^{1/L} - 1 \tag{2.149}$$

is the **geometric SNR**. The name arises from recognizing the geometric average of terms $(1+\mathrm{SNR}_\ell)$. Basically, the mutual information – equivalently the reliably achievable data rate with $\Gamma = 0$ dB codes – looks like it equivalently arose from L uses of a simple AWGN with $\mathrm{SNR} = \mathrm{SNR}_{geo}$.

A simple case might have two gain values, for instance with $L \in 2\mathbb{Z}^+$, such that $\mathrm{SNR}_\ell = \mathrm{SNR}_A$ for $\ell = 1, ..., L/2$ and $\mathrm{SNR}_\ell = \mathrm{SNR}_B$ for $\ell = L/2+1 ... L$. The system then also appears as if two channels with $\mathrm{SNR}_A \neq \mathrm{SNR}_B$ time-share (really dimension-sharing) an available dimension over multiple uses or symbols, in this case equally. Similarly other fractional uses for 2 or more SNR's could time-share the single equivalent channel, not necessarily equally. Each use would occur its respective fraction of "time." This is equivalent to having two, or correspondingly more, parallel channels operate separately. While SNR_{geo} and time-sharing are simple concepts they underly many more sophisticated concepts to come in multiuser channels where different codes with different b_ℓ (possibly for different users or even for decomposing a single user into multiple subusers) may share the channel's available dimensions, whether they are viewed as repeated "time-domain" uses of a single channel or separate parallel channels. Similarly Chapter 4's important Separation Theorem is at its root simply a restatement of time-sharing.

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 $C_{\boldsymbol{x}}$, the ML decoder will select the sequence of subsymbols, equivalently the codeword, that minimizes $\|\boldsymbol{y} - \widetilde{H} \cdot \boldsymbol{x}\|^2$ when the noise has $R_{\boldsymbol{nn}} = I$ finds and selects the codeword such that $\widetilde{H} \cdot \boldsymbol{x}$ is closest to $\boldsymbol{y}$ - that is the sequence of subsymbols. While the number of bits/subsymbol is finite, $\tilde{b} < \infty$, 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 $\boldsymbol{x}$ **over all its subsymbols** that minimizes

$$\min_{\{\tilde{\boldsymbol{x}}_k\}} \sum_{k=-\infty}^{\infty} \|\tilde{\boldsymbol{y}}_k - \widetilde{H} \cdot \tilde{\boldsymbol{x}}_k\|^2 \neq \sum_k \|\tilde{\boldsymbol{n}}_k\|^2 \ . \tag{2.150}$$

(2.150)'s inequality follows because with an effectively continuous distribution for $\tilde{\boldsymbol{x}}_k$, 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 $\boldsymbol{x}$ and minimize codeword-error probability; and indeed this best codeword has individual subsymbols $\tilde{\boldsymbol{x}}_k$ 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 $\boldsymbol{x}$, the sample MMSE estimate

$$MMSE = \min_W \lim_{K\to\infty} \frac{1}{2K+1} \sum_{k=-K}^{K} \|\tilde{\boldsymbol{x}}_k - W \cdot \tilde{\boldsymbol{y}}_k\|^2 \tag{2.151}$$

$$= \lim_{K\to\infty} \frac{1}{2K+1} \sum_{k=-K}^{K} \|\boldsymbol{e}_k\|^2 \ . \tag{2.152}$$

By the LLN, the sum of (on average over random coding) independent terms in (2.152) converges to the MMSE with probability 1. From Appendix D, $W = R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{y}}} \cdot R_{\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}}^{-1}$ and

$$\hat{\tilde{\boldsymbol{x}}}_{mmse} = W \cdot \tilde{\boldsymbol{y}} = E\left[\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}\right] \quad . \tag{2.153}$$

But since $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$ are on average with random coding jointly (as well as individually) Gaussian, the estimate $\hat{\tilde{\boldsymbol{x}}}_{mmse} = E\left[\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}\right]$ also must maximize the à posteriori distribution $p_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}}$ for any given channel output $\boldsymbol{y}$'s sequence of subsymbol values $\tilde{\boldsymbol{y}}$. Thus, on average, the MMSE estimate (sequence of subsymbol estimates) is MAP (and consequently ML) for the codeword $\boldsymbol{x}$. **QED**.

The convergence theorem holds for any input autocorrelation matrix $R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}$, noise correlation matrix $R_{\tilde{\boldsymbol{n}}\tilde{\boldsymbol{n}}}$ 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 (2.150).

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

$$\tilde{\boldsymbol{x}} \quad \rightarrow \quad \tilde{\boldsymbol{y}} \quad (H) \tag{2.154}$$

$$\tilde{\boldsymbol{y}} \quad \rightarrow \quad \tilde{\boldsymbol{x}} \quad (W)\,, \tag{2.155}$$

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

$$\begin{aligned}
\mathcal{I}(\tilde{\boldsymbol{x}};\tilde{\boldsymbol{y}}) &= \mathscr{H}_{\tilde{\boldsymbol{y}}} - \mathscr{H}_{\tilde{\boldsymbol{y}}/\tilde{\boldsymbol{x}}} && (2.156)\\
&= \log_2\left(\frac{\left|R_{\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}}\right|}{|R_{\tilde{\boldsymbol{n}}\tilde{\boldsymbol{n}}}|}\right) \text{ bits/subsymbol} && (2.157)\\
&= \mathscr{H}_{\tilde{\boldsymbol{x}}} - \mathscr{H}_{\tilde{\boldsymbol{x}}/\tilde{\boldsymbol{y}}} && (2.158)\\
&= \log_2\left(\frac{|R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|}{|R_{\boldsymbol{ee}}|}\right) \text{ bits/subsymbol} && (2.159)\\
&= \log_2|I - W\cdot H| && (2.160)\\
&= \log_2|I - H\cdot W| \quad . && (2.161)
\end{aligned}$$

The ML detector corresponds to the second problem in (2.155) that accepts $\tilde{\boldsymbol{y}}$ and produces $\tilde{\boldsymbol{x}}$. The first problem in (2.154) has MMSE $\tilde{\boldsymbol{n}}$ but is not the ML detector for $\boldsymbol{x}$.

The Zero-Forcing Solution: The first problem in (2.154) does produce a **zero-forcing estimate** when written as it's inverse (H^+ is the pseudoinverse, see Appendix C):

$$\hat{\tilde{\boldsymbol{x}}}_{ZF} = H^+ \cdot \tilde{\boldsymbol{y}} \quad . \tag{2.162}$$

While this $\hat{\tilde{\boldsymbol{x}}}_{ZF}$ 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 $\tilde{\boldsymbol{x}}$ reduces just enough that the benefit of reducing signal power to reduction of the squared error terms is best (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

$$E\left[\hat{\tilde{\boldsymbol{x}}}/\tilde{\boldsymbol{x}}\right] \leq \tilde{\boldsymbol{x}} \quad . \tag{2.163}$$

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

$$2^{\mathcal{I}(\tilde{\boldsymbol{x}};\tilde{\boldsymbol{y}})} = \frac{\left|R_{\tilde{\boldsymbol{y}}\tilde{\boldsymbol{y}}}\right|}{|R_{\tilde{\boldsymbol{n}}\tilde{\boldsymbol{n}}}|} = \frac{|R_{\tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}}|}{|R_{\tilde{\boldsymbol{e}}\tilde{\boldsymbol{e}}}|} \quad . \tag{2.164}$$

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

$$SNR_{mmse,k} = \frac{\mathcal{E}_k}{\sigma^2_{mmse,k}} = SNR_{unbias,k} + 1 \tag{2.165}$$

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

$$\boldsymbol{y}'' = \frac{SNR_{mmse,k}}{SNR_{mmse,k} - 1} \cdot \boldsymbol{y}' = \frac{SNR_{unbias,k} + 1}{SNR_{unbias,k}} \cdot \boldsymbol{y}' \quad . \tag{2.166}$$

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 $\Gamma > 0$ 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 $E\left[\hat{\tilde{\boldsymbol{y}}}/\tilde{\boldsymbol{y}}\right] \leq \tilde{\boldsymbol{y}}$ 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 $\widetilde{N}_x = \widetilde{N}_y = 1$ 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.

2.3.7 Ergodic Capacity and Random Coding

This section's entire AEP development and capacity-theorem development could view the joint distribution $p_{\boldsymbol{x},\boldsymbol{y}} = p_{\boldsymbol{y}/\boldsymbol{x}} \cdot p_{\boldsymbol{x}}$ with channel distribution $p_{\boldsymbol{y}/\boldsymbol{x}}$ as parametrized (so for instance by channel gain or matrix-MIMO channel H). If that parameter itself has a distribution, often representing ergodic time variation as in Section 1.6, which is also independent of $\boldsymbol{x}$, then all results could be average in a final step to obtain the ergodic mutual information and of course **ergodic capacity** when averaged over the independent distribution of the parameter.

Definition 2.3.6 *[Ergodic Capacity] The* **ergodic capacity** *averages the capacity over the independent distribution of the channel parameter* $\widetilde{H}$.

$$< \mathcal{C} > \triangleq \mathbb{E}_{\widetilde{H}}\left[\mathcal{C}_{\widetilde{H}}\right] \tag{2.167}$$

However, the issue of whether a single code exists that assures reliable decoding when used on the ensemble of averaged channels remains. Fortunately, such a code does exist:

Theorem 2.3.4 *[Ergodic Capacity Code Existence] At least one code exists that is applicable to the random channel as long as the data rate* $b \leq < \mathcal{C} >$.

Proof: Figure 2.17 is the key step in the AEP capacity-region development with random coding. The independence of $\boldsymbol{x}$ and $\widetilde{H}$ permits each random symbol selected in a codeword of a randomly constructed code to also correspond to a simultaneous choice from the (also presumably ergodic) channel distribution parameter $\widetilde{H}$'s distribution. While this simultaneous selection does not change $\mathcal{H}$ (or $\mathscr{H}$), it does affect $p_{\boldsymbol{x}/\boldsymbol{y}}$ through $\boldsymbol{y}$'s dependency on $\widetilde{H}$. Nonetheless, the AEP (law of large numbers) still applies to the convergence of the sample-average conditional entropy to its true mean value (which averages over both $p_{\boldsymbol{x}}$ and $p_{\widetilde{H}}$). Thus, while the $\bar{N}$ for a given ϵ in typical set construction may need to be larger when the channel is randomly parametrized, the asymptotic result remains that the sample average converges to the mean with probability 1. Subsection 2.3.3's proof remains the same, albeit tacitly at a possibly longer codeword length. Therefore a code exists with probability one that allows reliable decoding ($P_e \rightarrow 0$) that is applicable to the randomly parametrized channel. **QED**

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 $\bar{\mathcal{E}}_{\boldsymbol{x}}$ and (two-sided) noise power spectral density σ^2 has SNR $= \frac{\bar{\mathcal{E}}_{\boldsymbol{x}}}{\sigma^2}$. Dividing (2.139) by 2 real dimensions per subsymbol, this AWGN channel has maximum data rate or capacity

$$\bar{\mathcal{C}} = \frac{1}{2} \cdot \log_2(1 + \text{SNR}) \text{ bits/dimension.} \tag{2.168}$$

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 Γ 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

$$\bar{b} = \frac{1}{2} \cdot \log_2(1 + \frac{\text{SNR}}{\Gamma}) \text{ bits/dimension} \quad . \tag{2.169}$$

This gap approximation is the capacity formula with the SNR reduced by the gap. For these codes usually, the approximation[44] 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 $\bar{b} < \bar{\mathcal{C}}$. For gap-characterized codes, the gap is a constant function of the SNR and known bits/dimension for the given $\bar{P}_e$ and code class $C_{\boldsymbol{x}}$ as

$$\Gamma(C_{\boldsymbol{x}}, \bar{P}_e) \triangleq \frac{2^{2\bar{c}} - 1}{2^{2\bar{b}} - 1} = \frac{\text{SNR}}{2^{2\bar{b}} - 1} \quad . \tag{2.170}$$

Table 2.1 lists achievable $\bar{b}$ for the uncoded QAM scheme, which has constant $\Gamma = 8.8$ dB at $P_e = 10^{-6}$ using square constellations ($\widetilde{N} = 2$) versus AWGN SNR.[45]

$P_e = 10^{-6}$	$C_{\boldsymbol{x}}$ = SQ QAM					
$\bar{b}$	.5	1	2	3	4	5
SNR for $P_e = 10^{-6}$ (dB)	8.8	13.5	20.5	26.8	32.9	38.9
$2^{2\bar{b}} - 1$ (dB)	0	4.7	11.7	18.0	24.1	30.1
Γ (dB)	8.8	8.8	8.8	8.8	8.8	8.8

Table 2.1: Table of AWGN SNR Gaps for $\bar{P}_e = 10^{-6}$.

Table 2.1 shows that uncoded SQ QAM, or equivalently with PAM, at $\bar{P}_e = 10^{-6}$, has constant SNR gap $\Gamma = 8.8$ dB. With $C_{\boldsymbol{x}}$ as uncoded QAM or PAM and $P_e = 10^{-7}$, the gap is instead a constant 9.5

[44] For $\bar{b} < .5$, systems of codes can be constructed and viewed as one code that exhibit the same constant gap as for $\bar{b} \geq .5$.

[45] Thus, Table 2.1 assumes for $\bar{b} = .5$ that instead of simple BPSK transmission of $\begin{bmatrix} \pm\sqrt{2} & 0 \end{bmatrix}$ for an energy/symbol of 2, $\bar{\mathcal{E}}_{\boldsymbol{x}} = 1$, and $d^2_{min} = 8$ that instead the system transmits one of the following 4 symbol values for two successive QAM channel sub-symbols $2/\sqrt{3} \cdot [1\ 1\ 1\ 0]$, $2/\sqrt{3} \cdot [-1\ -1\ 1\ 0]$, $2/\sqrt{3} \cdot [1\ -1\ -1\ 0]$, or $2/\sqrt{3} \cdot [-1\ 1\ -1\ 0]$ for also an energy per dimension of $\bar{\mathcal{E}}_{\boldsymbol{x}} = 1$ and $\bar{b} = 0.5$, but with a minimum distance squared of $d^2_{min} = 32/3$. This improvement is 32/24= 1.3 dB. Using a few more dimensions like this can provide another .4 dB fairly easily, so 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 $\bar{b} \geq 0.5$ 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 Chapter 8) reduces the gap. Often these codes will have gaps that do not vary for $\bar{b} \geq .5$. A very well-coded system may may have a gap as low as .5 dB at $P_e \leq 10^{-6}$. Figure 2.19 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 with good codes.

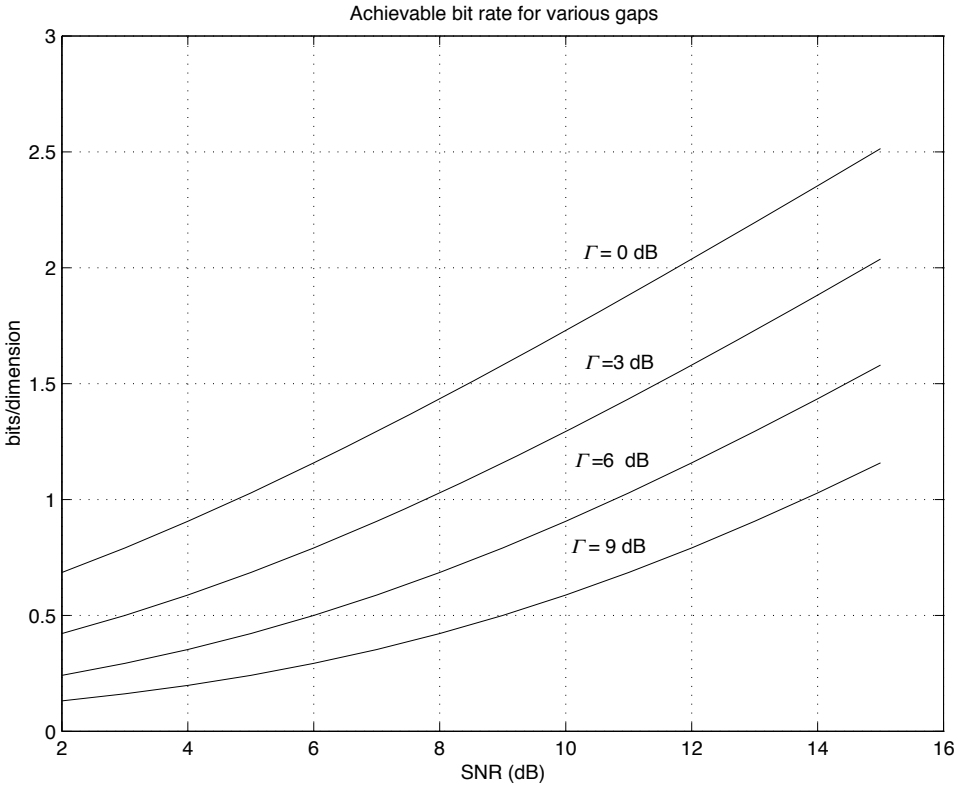


Figure 2.19: Illustration of bit rates versus SNR for various gaps.

For a given coding scheme, practical transmission designs often mandate a specific value for 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 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**, γ_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 $C_{\boldsymbol{x}}$ and $\bar{P}_e$ with gap Γ 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 follows through the use of the gap formula as*

$$\gamma_m = \frac{2^{2\bar{b}_{max}} - 1}{2^{2\bar{b}} - 1} = \frac{SNR/\Gamma}{2^{2\bar{b}} - 1} \quad . \tag{2.171}$$

The margin is the amount by which the SNR can lower before performance degrades to an error probability greater than the target $\bar{P}_e$ that defines the gap. A negative margin in dB means that the SNR must improve by the margin's magnitude to achieve the $\overline{P}_e$. The margin relation has alternative form

written as

$$\bar{b} = .5 \cdot \log_2 \left(1 + \frac{\text{SNR}}{\Gamma \cdot \gamma_m} \right) \quad , \tag{2.172}$$

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, but each contributes in the same fashion.

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

$$\bar{c} = .5 \cdot \log_2 (1 + \text{SNR}) = 3.5 \text{ bits/dim} \quad . \tag{2.173}$$

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

$$\bar{b} = .5 \cdot \log_2 \left(1 + \frac{\text{SNR}}{10^{.88}} \right) = 2 \text{ bits/dim} \quad . \tag{2.174}$$

With concatenated trellis and forward error correction (or with "turbo codes" – see Chapter 8), 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

$$\bar{b} = .5 \cdot \log_2 \left(1 + \frac{\text{SNR}}{10^{.18}} \right) = 3 \text{ bits/dim} \quad . \tag{2.175}$$

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

$$\gamma_m = \frac{2^{2 \cdot 3} - 1}{2^{2 \cdot 2.5} - 1} = \frac{63}{31} \approx 3 \text{ dB} \quad . \tag{2.176}$$

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 ($\bar{b} = 1$) over this channel and uses no code as in (2.174), then the margin is

$$\gamma_m = \frac{2^{2 \cdot 2} - 1}{2^{2 \cdot 1} - 1} = \frac{15}{3} \approx 7 \text{ dB} \quad . \tag{2.177}$$

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, $\boldsymbol{x}_0$, ... , $\boldsymbol{x}_{M-1}$ and J outputs $\boldsymbol{y}_0$,..., $\boldsymbol{y}_{J-1}$. The term "memoryless" means that $p_{\boldsymbol{y}/\boldsymbol{x}} = \prod_{n=1}^{N} p_{\boldsymbol{y}_n/\boldsymbol{x}_n}$.

BSC Capacity: Figure 2.9's BSC is probably the most widely cited DMC, and might be characterized for an underlying hard-coded AWGN by

$$p \triangleq \bar{P}_b = \frac{N_b}{b} \cdot Q \left(\frac{d_{min}}{2\sigma} \right) \quad . \tag{2.178}$$

The BSC's capacity follows by substitution into the mutual information formula:

$$
\begin{aligned}
I(x;y) &= \sum_{m=0}^{1}\sum_{j=0}^{1} p_{\boldsymbol{x}}(m)\cdot p_{\boldsymbol{y}/\boldsymbol{x}}(j,m)\cdot \log_2\left(\frac{p_{\boldsymbol{y}/\boldsymbol{x}}(j,m)}{p_{\boldsymbol{y}}(j)}\right) && (2.179)\\
&= p_x(0)\cdot(1-p)\cdot\log_2\left(\frac{1-p}{p_x(0)\cdot(1-p)+p_x(1)\cdot p}\right) && (2.180)\\
&+ p_x(0)\cdot(p)\cdot\log_2\left(\frac{p}{p_x(0)\cdot p+p_x(1)\cdot(1-p)}\right) && (2.181)\\
&+ p_x(1)\cdot(p)\cdot\log_2\left(\frac{p}{p_x(0)\cdot(1-p)+p_x(1)\cdot p}\right) && (2.182)\\
&+ p_x(1)\cdot(1-p)\cdot\log_2\left(\frac{1-p}{p_x(0)\cdot p+p_x(1)\cdot(1-p)}\right) && (2.183)
\end{aligned}
$$

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,

$$
\begin{aligned}
\bar{\mathcal{C}} &= (1-p)\cdot\log_2\left[2(1-p)\right]+p\cdot\log_2(2p) && (2.184)\\
&= 1-\mathcal{H}(p) \quad , && (2.185)
\end{aligned}
$$

where

$$
\mathcal{H}(p)\stackrel{\Delta}{=} -p\cdot\log_2(p)-(1-p)\cdot\log_2(1-p) \quad , \tag{2.186}
$$

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

BEC's Capacity: A second commonly encountered channel is the **binary erasure channel** (BEC) of Figure 2.12. The channel is again symmetric in $p_{\boldsymbol{x}}$, so that the maximizing input distribution for the mutual information is $p_x(0)=p_x(1)=.5$. The capacity is then

$$
\begin{aligned}
\bar{\mathcal{C}} &= \left[\frac{1}{2}(1-p)\log_2\frac{1-p}{\frac{1}{2}(1-p)}\right]2+\left[\frac{1}{2}p\log_2\frac{p}{2p\frac{1}{2}}\right]2 && (2.187)\\
&= 1-p && (2.188)
\end{aligned}
$$

Again, as $p\to 0$, there are no errors made and $\bar{\mathcal{C}}\to 1$ bit/symbol, otherwise $\bar{\mathcal{C}}\le 1$ for the BEC. When $p\le 0.5$, then $\bar{\mathcal{C}}_{BEC}\ge\bar{\mathcal{C}}_{BSC}$, 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 ($p_{output/input}$ 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

$$
\begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix} \tag{2.189}
$$

The symmetric DMC's capacity/maximizing distribution is again uniform by symmetry. A special case of interest is Figure 2.11'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 over the 2^b possible discrete messages. The capacity is

$$
\mathcal{C}=b-p_s\cdot\log_2\frac{2^b-1}{p_s}+(1-p_s)\cdot\log_2(1-p_s)\le b \text{ bits.} \tag{2.190}
$$

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 arises from 8 successive BSC, then

$$
p_s=1-(1-p)^b\approx b\cdot p=8\cdot p \text{ for small } p \quad . \tag{2.191}
$$

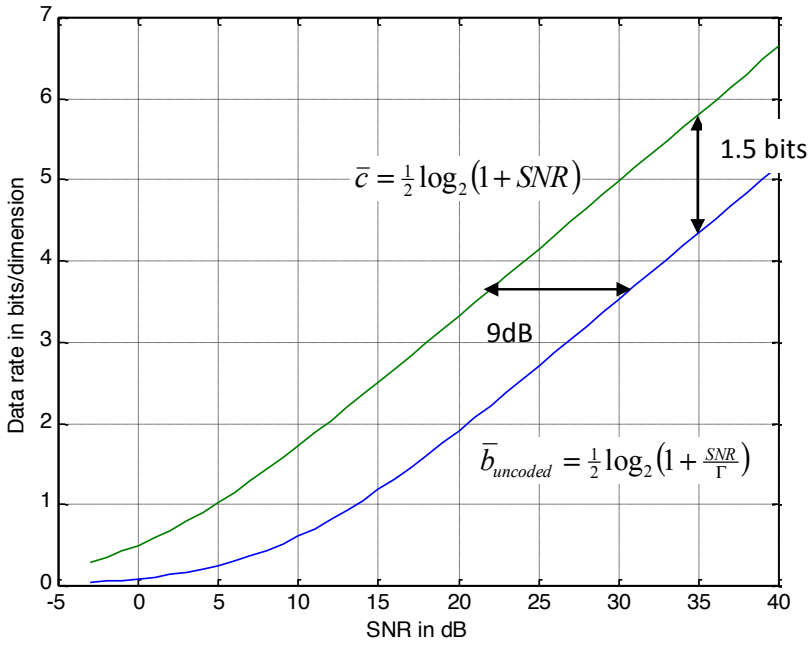


Figure 2.20: Illustration of capacity and gap to uncoded transmission with $\Gamma = 8.8$ dB.

UDSC 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 $\bar{P}_{UDSC,e}$ essentially to 0.

2.4.3 Capacity, Coding, and the Gap

Figure 2.20 plots the AWGN capacity $\bar{\mathcal{C}} = .5 \cdot \log_2(1 + \text{SNR})$ with expanded SNR on the horizontal axis. Figure 2.20's semi-log plot shows that as SNR becomes reasonably large (say 20 dB or more), that increase of capacity by $\bar{b} = 1$ bit per dimension requires SNR increase of 6 dB. Since QAM finds heavy use in practice, a bit per dimension of QAM corresponds to two bits per symbol or an often quoted QAM rule of "3 dB per bit". At low SNR (below 10 dB), the 6 dB 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

$$\lim_{\text{SNR}\to 0} \bar{\mathcal{C}} = \frac{.5}{\ln 2} \cdot \text{SNR} \quad . \tag{2.192}$$

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

Codes that use PAM- and QAM-like sub-symbols can recover this lost 9 dB of uncoded transmission. The price is not a higher error rate (and indeed the error probability can be asymptotically 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 8 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. (These statements assume $L_x = 1$.) There is essentially a variable 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

$$\begin{aligned} \bar{b} < \bar{\mathcal{C}} &= \frac{1}{2} \cdot \log_2 \left(1 + \frac{\bar{\mathcal{E}}_{\boldsymbol{x}}}{\sigma^2} \right) && (2.193) \\ &= \frac{1}{2} \cdot \log_2 \left(1 + \frac{\mathcal{E}_{\boldsymbol{x}}}{N\sigma^2} \right) && (2.194) \\ &= \frac{1}{2} \cdot \log_2 \left(1 + \frac{\bar{b}\mathcal{E}_{\boldsymbol{x}}}{b\sigma^2} \right) && (2.195) \\ &= \frac{1}{2} \cdot \log_2 \left(1 + \frac{\bar{b}\mathcal{E}_b}{\sigma^2} \right) \quad . && (2.196) \end{aligned}$$

Solving for $\frac{\mathcal{E}_b}{\sigma^2}$ in (2.196), and assuming that a capacity-achieving code is in use so that $\bar{b} = \bar{\mathcal{C}}$, yields

$$\frac{\mathcal{E}_b}{\sigma^2} = \frac{2^{2\bar{\mathcal{C}}} - 1}{\bar{\mathcal{C}}} \quad . \tag{2.197}$$

Equation (2.197) essentially is the minimum required $\mathcal{E}_b/\sigma^2$ for any given well-coded system's bits/symbol $\bar{\mathcal{C}}$ on the AWGN. Equation (2.197) also illustrates that any system that uses dimensionality, which is an increase of N at the same energy $\mathcal{E}_{\boldsymbol{x}}$ so smaller $\bar{\mathcal{E}}_{\boldsymbol{x}}$ and smaller $\bar{b}$, benefits by needing less energy per bit transmitted. Thus, again a choice between the same factor increase in dimensionality or power should take dimensionality. This gain diminishes as dimensions grow to infinity to a minimal level. Of fundamental interest is that level where $\bar{\mathcal{C}} \to 0$ (that is large redundancy in the code). Then (2.197) reduces to

$$\frac{\mathcal{E}_b}{\sigma^2} = 2 \cdot \ln 2 \;\; (1.4 \text{ dB}) \quad , \tag{2.198}$$

meaning that the energy/bit must be above this finite minimum value even if a code uses infinite redundancy (or infinite dimensionality or bandwidth). This result is sometimes phrased in terms of the quantity $\mathcal{E}_b/\mathcal{N}_0 = .5(\mathcal{E}_b/\sigma^2)$, which is equivalent to the statement that the minimum required $\mathcal{E}_b/\mathcal{N}_0$ is -1.6dB, an often-cited result. The dimensionality N may be limited by other constraints (like spectrum regulation for other systems occupying the same medium, like wireless) or physical contraints. The dimensional increase could also be in spatial dimensions and the same effect occurs if the available power ($\mathcal{E}_{\boldsymbol{x}}/T$) remains constant (presumably at fixed symbol rate, so larger dimensionality means more antennas and/or wider bandwidth).

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 Chapter 8.

2.4.5.1 Designing with Reed Solomon/Cyclic Codes

Cyclic block codes over the field $GF(q^i)$ for postive integer i and q have all codewords as circular shifts of one another. Reed Solomon[46] (a well used form of cyclc) codes are linear and also MDS and therefore achieve the Singleton Bound performance, namely $d_{free} = N - K + 1$. For the binary case of $q^i = 2$, only trivial codes are MDS and then have $d_{free} = n - k + 1$, but when $q > 2$, nontrivial codes exist and Reed Solomon (RS) are an example. A RS code with $\mathbb{GF}^{255}$ therefore works with bytes (octets) of 8 bits each,

[46] After USC Professor Irving S. Reed, 1923-2012, and the Jet Propulsion Laboratories' Gustave Solomon, 1930-1996.

and Figure 2.11's Symmetric DMC can be a potential byte-in-byte-out channel with $q-1=255$. Such RS codes allow $1<N<256$ with $\widetilde{N}=1$ byte. The number of parity subsymbols allowed within the codeword is $0 \le P \le N-1$ with correspondingly is $d_{free}=P-1$. RS codes with $q=256$ additionally impose the restriction that $P \le 32$. Usually the design selects P even so that d_{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 $(P-P')/2 \ge 0$ unknown erred subsymbol locations.

Equation (2.34) 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 (2.35) and (2.37) respectively.

A channel might have inner decoded bit-error probability $p=1.25\times10^{-4}$ and thus a byte-error probability of approximately 10^{-3}. By using the first term of (2.35), with $N_e=\binom{N}{d_{free}})/K$, the code must choose N and K to satisfy

$$N_e=\binom{N}{d_{free})}/K(p)^{\lfloor\frac{d_{free}-1}{2}\rfloor}<10^{-7} \tag{2.199}$$

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 $2.7\times10^{-8}<10^{-7}$. 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 (at fixed constellation size $|C|$), 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 $p^{\lceil\frac{d_{free}-1}{2}\rceil}=p^2$ 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 Example 2.2.1's 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 3dB. Thus, when the bandwidth can be doubled, this code gains 4dB 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 $r<\mathcal{C}$ asymptotically as $n\to\infty$. Such codes typically have long block lengths in practical use, and iterative decoders. These decoders construct LLR's for each bit (so ML or MAP decoder for each bit instead of the entire codeword) 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 $\mathcal{C}<1$. 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 next subsection's BICM case) and achieve also highest performance; except for shaping gain that must be separately considered.

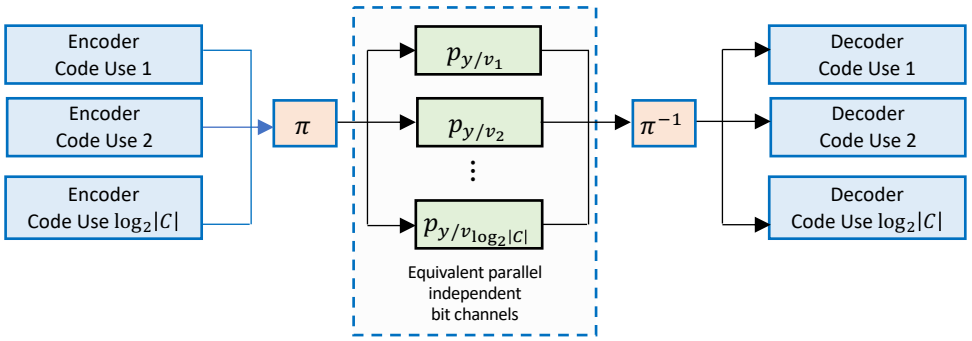


Figure 2.21: BICM's creation of parallel independent bit channels.

2.4.6 Bit-Interleaved Coded Modulation (BICM)

Bit-Interleaved Coded Modulation, [10] permits some good binary codes to map under certain conditions to the bit-labels of QAM (or PAM or more generally lower dimensionality subsymbols) and still retain the good code's performance in the multi-level system for the AWGN. Essentially good codes by the AEP correspond to sampling of codeword subsymbols from a continuous uniform distribution on the subsymbol. As long as the independent random selection effectively holds also for the larger-constellation, then the mapping thus preserves the good code asymptotically. Figure 2.21 illustrates the concept with an interleaver (see Section 8.3 on interleavers), which can sometimes be used to take a channel where time (more generally dimensions) upon which bits are encoded/modulated are so distant that they cause the conditional channel probabilities to be independent, creating effectively a set of parallel bit channels. Essentially 3 basic principles explain why BICM works in the next 3 subsections.

2.4.6.1 Continuous approximation and shaping gain

Section 1.3's "continuous approximation" (calculates average constellation energy by approximating a large constellation by a continuous uniform distribution over a Voronoi region that tightly bounds the constellation points. With an average energy constraint, the best region is a hypersphere and the marginal corresponding continuous distributions are Gaussian. Coding gain decomposes into two components, the fundamental gain γ_f that corresponds to the asymptotic equal uniform spacing of codewords within any volume, and the shaping gain γ_s that corresponds to bounding the constellation well. When the volume is restricted (as often in practice) to a hypercube, each dimension's continuous approximation will have for M equally spaced points $\bar{\mathcal{E}}_{\boldsymbol{x}} = \frac{M \cdot d^2}{12}$ where the inter-point spacing is d, and where the boundary in 1 dimension is a segment $(-\frac{(M-1)\cdot d}{2}, \frac{(M-1)\cdot d}{2})$. A good (AEP sense) code with such boundary in all dimensions as $N \to \infty$ produces codewords with uniform distribution asymptotically and a uniform distribution in each dimension also. If the constellation has a large M, then the shaping gain will be (as in Section 1.3) 0 dB shaping gain, so less than the best $\pi \cdot e/6 = 1.53$ dB (or loss of 1/4 bit/real dimension). For smaller M, the shaping gain possible is less than 1.53 dB.

Designers often accept the possible shaping loss (never more than 1/4 bit/dimension and usually less at practical SNRs or equivalently moderate M values). Thus, square constellations are often used, and most designs do not attempt to include shaping gain.

2.4.6.2 Discrete constellation points versus continuous uniform as quantization error

Practical codes will also use a discrete finite constellation. Such codes will need redundancy $\rho > 0$ bits, but the constellation need only have 1-2 bits more, so 2x to 4x the number of 2D points, than the minimum necessary $\bar{b}$. With inter-constellation-point spacing Δ (not to confuse with d_{min}), the miss between what random coding from a uniform continuous distribution and the discrete distribution is a

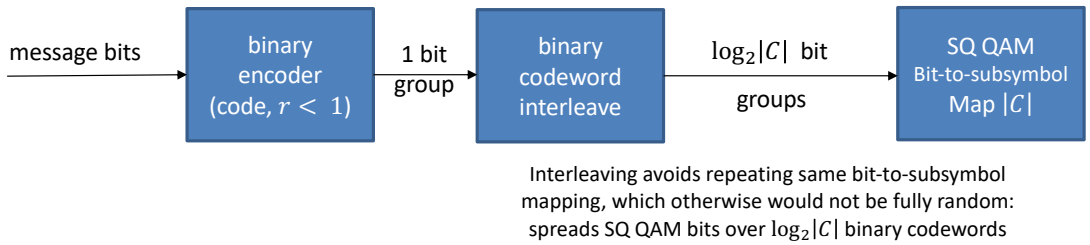


Figure 2.22: Bit-Interleaved Coded Modulation Transmitter.

random quantization error of variance $\Delta^2/12$. For a code with $\bar{\mathcal{C}} = 1$ bit/dimension, then $SNR = 3$ (4.7 dB). The noise variance is then $3/\bar{\mathcal{E}}_{\boldsymbol{x}}$. To make the codeword-miss small (.1dB = 1+1/10)) relative to the per-dimensional noise variance:

$$\frac{\Delta^2}{12} \leq \frac{1}{10} \cdot \frac{\bar{\mathcal{E}}_{\boldsymbol{x}}}{3} \quad , \tag{2.200}$$

which means $\Delta^2 = 0.4 \cdot \sigma^2$ or about 4 dB smaller than $\bar{\mathcal{E}}_{\boldsymbol{x}}$ when $\bar{\mathcal{C}} = 1$. This is achieved with 2x the number of points in the constellation or $\bar{\rho} = 1$; indeed $\bar{\rho} = 2/3$ would be sufficient for 0.1 dB loss with respect to a continuous uniform distribution as in AEP. In two dimensions this would be 4/3 bits, so using a square constellation with between 2x to 4x the number of 2D points suffices. The same argument applies to the spacing between points in larger constellations, just that the number of bits must also cover the average number of message points plus the extra 2/3 bit/dimension.

For this reason, well coded transmission systems with $\bar{b}$ rarely use more that 2x the constellation size (so $\tilde{\rho} = 1$) and little performance is lost. In some exceptional cases, $\tilde{\rho} = 2$ will make any loss with respect to continuous-distribution random code-design sampling negligible. Further, when the SNR is lower yet, the excess number of points further decreases and asymptotically systems with $\bar{\mathcal{C}} < 1$ effectively can achieve capacity with 2 level constellations, that is BPSK or binary PAM. Indeed such systems also cannot provide significant shaping gain so basically two levels is sufficient at low SNR.

2.4.6.3 Interleaving to preserve random uniform subsymbol sampling

Figure 2.22 completes the BICM description. A good binary code maps into a constellation (typically QAM with no more than twice the number of points necessary for $\tilde{b}$. However, the grouping of binary codeword bits for each subsymbol is not truly random and independent if the encoder uses the same mapping for every $\tilde{b}$ bits, even though a mapping may be 1-to-1 from a random-designed good binary code, that is not sufficient to ensure random well designed multi-level subsymbol selection. The solution is to randomize the mapping also, as in Figure 2.22's interleaver. Chapter 8 covers interleaving in detail, but the basic concept is a reordering of transmit bits that is restored at the receiver. If the interleaving has a depth that is not $\tilde{b}$ or some common multiple or divisor thereof, then the process of random codeword selection effectively passes from the binary system to the multi-level system. Some binary codes like turbo codes and LDPC codes (see Chapter 8) work very well with good interleaving and multi-level mapping because they have little regular structure. However, binary codes like Polar Codes, while approaching capacity in a canonical way such that suboptimum successive-decoding receiver detection strategies essentially come arbitrarily close to reliably achieving near-capacity reates, nonetheless have structure that leads to nonuniform multi-level codes. These polar codes simplified decoders do not work well with BICM. Solutions are unknown at present, other than to use a more complex ML decoder. Such an ML decoder voids the polar code's advantage.

2.5 Parallel Channels and the Multitone AWGN Channel

Sections 2.3 and 2.4 (for $\bar{\mathcal{C}}$) illustrate the most essential capacity concepts. In practice, engineers usually specify in data rates in units of bits/second. This section investigates previous results' conversion 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. **multitone** channels extend temporal dimensionality in frequency. The number of dimensions is again $N = \widetilde{N} \cdot \bar{N}$, but $\bar{N}$ takes a frequency or tone-index character in multitone, so that $n = 0, ...\bar{N} - 1$ equal-width tones occur with $\bar{N} \to \infty$ asymptotically. $\bar{N}$ is also the number of (frequency-indexed) subsymbols per symbol. Each such tone independently applies Section 2.3's AEP analysis for its use of capacity-achieving codes. Multitone systems have $\widetilde{N} = 2$ and $\bar{N}$ is the number of tones. This number of tones, $\bar{N}$, depends on the available usable bandwidth, as in Subsection 2.5.1. Subsection 2.5.2 expands to Shannon's asymptotic ($\bar{N} \to \infty$) multitone parallel-independent-channel set.

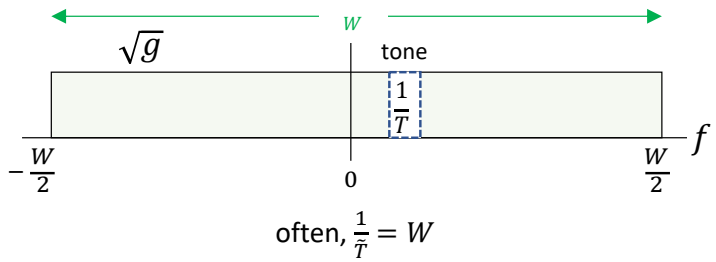
2.5.1 Capacity Conversion for Memoryless Channels

The DMC's and the AWGN channel are memoryless in that channel-dimensional uses are independent:

$$p_{\boldsymbol{y}/\boldsymbol{x}} = \prod_{n=1}^{\bar{N}-1} p_{y_n/x_n} \quad . \tag{2.201}$$

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."

Figure 2.23 shows such fixed bandwidth channel $h(t) = \sqrt{g} \cdot W \cdot \text{sinc}(Wt)$ with constant nonzero gain and no phase distortion over a frequency interval $f \in [-W/2, W/2)$ or of (two-sided in real baseband case) bandwidth W. The symbols[47] are essentially length $T = \bar{N} \cdot \widetilde{T}$. So $T \geq \widetilde{T}$; which implies $1/T \leq 1/\widetilde{T}$. The frequency band W thus decomposes into $\bar{N}$ equal width tonal channels in multitone, each of width $\frac{1}{T} = \frac{1}{\bar{N} \cdot \widetilde{T}}$. Multitone's nominal modulation filters are all channel-independent and equal to $\varphi_n(t) = \frac{1}{\sqrt{T}} \cdot \text{sinc}(\frac{t}{T}) \cdot e^{\jmath \cdot \frac{2\pi n}{T} \cdot t}$. For the flat channel, because $\text{sinc}(t) * \text{sinc}(t) = \text{sinc}(t)$, the multitone basis functions do not alter any channel tone's shape. The subsymbol rate is $1/\widetilde{T} = W$, which equals the fixed bandwidth. There are $\bar{N}$ frequency-indexed subsymbols per symbol.



$$h(t) = \sqrt{g} \cdot W \cdot sinc\,(Wt)$$

$$H(f) = \begin{cases} \sqrt{g} & |f| \leq \frac{W}{2} \\ 0 & |f| > \frac{W}{2} \end{cases}$$

Figure 2.23: Basic temporal subsymbol on a "flat" channel

The AWGN channel's noise whitened gain is $\sqrt{g}$ so that each tone input $x_n(t) = \sqrt{\mathcal{E}_n} \cdot \frac{1}{\sqrt{T}} \cdot \text{sinc}(t/T) \cdot e^{-\jmath 2\pi (n/T)}$ has subsymbol input energy $\mathcal{E}_n$ centered on its tone n/T. The leftmost and rightmost tones are half-width (SSB) in systems with even-integer $\bar{N}$, but can combine mathematically into a single two-dimensional channel in theory (since $\bar{N} \to \infty$, this is often ignored and these two unusual end-band channels carry zero energy and are ignored, avoiding an issue created by each possibly having a different

[47] Again, the AEP sense for each tone defines a codeword that has a different length tending to infinity than this $\bar{N}$, but this section and the sequel drop that implied asymptotic coding that tacitly underlies all results here.

gain). The multitone transmitter power is

$$P_x = \frac{\mathcal{E}_{\boldsymbol{x}}}{T} = \frac{\bar{N} \cdot \widetilde{\mathcal{E}}_x}{\bar{N} \cdot \widetilde{T}} = \frac{\widetilde{\mathcal{E}}_x}{\widetilde{T}} \tag{2.202}$$

that carries interpretation as energy per symbol divided by symbol period, as well as average energy per subsymbol divided by subsymbol period. The energy per subsymbol is thus $\widetilde{\mathcal{E}}_x = \mathcal{E}_x/\bar{N}$. Energy per (real) dimension remains $\mathcal{E}_{\boldsymbol{x}}/N$, while average energy per tone is $\widetilde{\mathcal{E}}_x$, equal to the average energy per subsymbol.

The (baseband) basis function $\varphi(t) = \frac{1}{\sqrt{T}} \cdot \mathrm{sinc}(t/T)$ allows successive transmitted modulated signals to be independently sampled[48] at times kT at the channel output without interference into one another, which is also intuitively obviously from the nature of the $\mathrm{sinc}(t)$ function's zero crossings and again that $\mathrm{sinc}(t) * \mathrm{sinc}(t) = \mathrm{sinc}(t)$. Thus, sampled tone outputs at the rate kT are independent, allowing AEP analysis to apply independently to each tonal channel. The term "bandwidth" W is approximate in practice.

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

$$\begin{aligned} \mathscr{C} &= 2W \cdot \bar{\mathcal{C}} = W \cdot \widetilde{\mathcal{C}} \text{ bits/sec} && (2.203) \\ &= W \cdot \log_2(1 + \mathrm{SNR}) \text{ bps} \ , && (2.204) \end{aligned}$$

a well-known restatement of Section 2.3's Shannon Capacity result in bits/second for a ("flat") AWGN channel of bandwidth W Hz.

EXAMPLE 2.5.1 [Computation of a Flat AWGN Capacity] Let Figure 2.23's $W =$ 10 MHz for a complex baseband system (so equivalently the positive-frequency signal bandwidth is 10 MHz). There are then 10^7 subsymbols sent per second. The number of symbols per second will be a factor of $\bar{N}$ less, so for instance $\bar{N} = 1000$ would mean 10 ksymbols/second, while $\bar{N} = 10,000$ would mean 1 ksymbol/second.

For this channel, the white noise has two-sided power spectra density $\sigma^2 = -120$ dBm/Hz and gain 10^{-3} so that transmit signals essentially reduce by 30 dB before entering the receiver. the $g = 10^{-3}/10^{-12} = 10^9$ (or 90 Hz/dBm). The transmit power is $P_x = 10$ mW (or 10 dBm). The transmit power spectral density is then $\widetilde{\mathcal{E}}_x = (10m\mathrm{W}/\widetilde{T} = 10/10^7 = 10^{-6}$ mW/Hz or equivalently -60 dBm/Hz. The SNR is thus SNR=$\widetilde{\mathcal{E}}_x \cdot g = -60 + 90 = 30$ dB (factor of 1000).

For this system,

$$\widetilde{\mathcal{C}} = \log_2(1 + 1000) \approx 10 \text{ bits/subsymbol} \tag{2.205}$$

and thus

$$\mathscr{C} = 10^7 \cdot \widetilde{\mathcal{C}} = 100 \text{ Mbs} \ , \tag{2.206}$$

of course in the AEP sense of infinite delay and best zero-gap code on each tone.

If W increases by a factor of 10, at the same transmit power now spread over this wider spectrum, then the capacity increases to

$$\mathscr{C} = 10^8 \cdot \log_2(1 + 100) \approx 700 \text{ Mbs} >> 100 \text{ Mbps} \ , \tag{2.207}$$

while if W decreases by a factor of 10, then

$$\mathscr{C} = 10^6 \cdot \log_2(1 + 10000) \approx 15 \text{ Mbs} << 100 \text{ Mbps} \ . \tag{2.208}$$

This example illustrates again that if wider bandwidth is available, best design exploits it and uses it, even with fixed transmit power.

[48] See Chapter 3.

2.5.2 Waveform Channels with Limited Bandwidth or Memory

Most practical channels are not memoryless. For example, the channel capacity for a filtered Gaussian noise channel with impulse response $h(t) \leftrightarrow H(f)$ and additive Gaussian noise with known power spectral density $\mathcal{S}_n(f)$ is of interest. Figure 2.24 illustrates **multitone modulation** where the transmission bandwidth W decomposes into $\bar{N}$ parallel non-overlapping equal-bandwidth channels. The $\bar{N}$ frequency-indexed subsymbols now may have different tonal channels, and indeed it may be wise to have different subsymbol input energies also. The tonal channels each have 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 $\widetilde{H}(f)$, which is a filtered AWGN. These tonal channels that sample $\widetilde{H}(f)\Big|_{f=n/T}$.

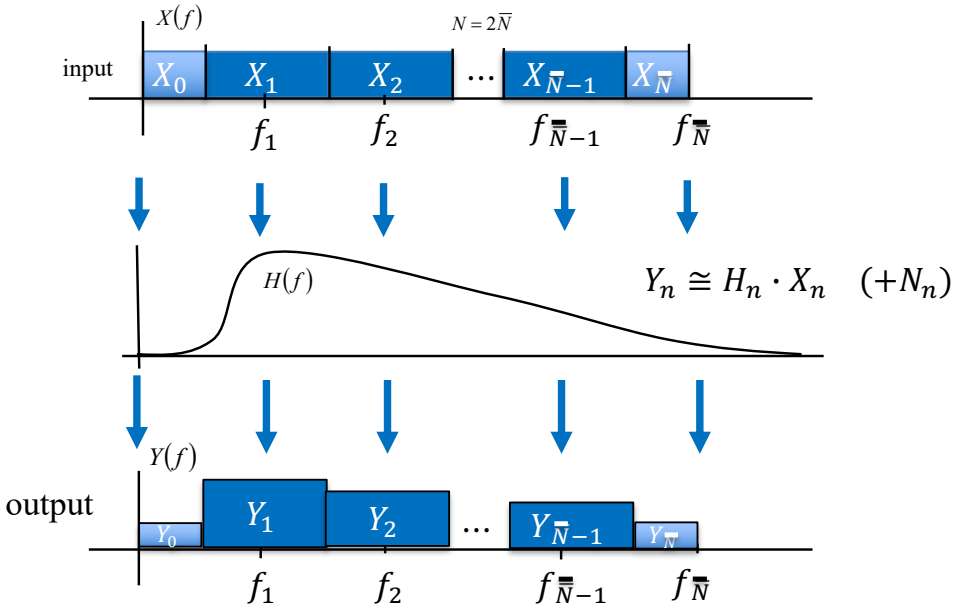


Figure 2.24: **Multitone Modulation:** set of parallel narrow frequency-indexed channels

The symbol period $T = \bar{N} \cdot \widetilde{T}$ becomes arbitrarily large as $\bar{N} \to \infty$, and analysis considers the channel as "one-shot" with infinite complexity and infinite detection delay for each tonal channel. The corresponding filtered-channel capacity here explicitly includes the symbol energy $\mathcal{E}_{\boldsymbol{x}} = P_x \cdot T$ and the symbol period T as arguments, $\mathcal{C} \to \mathcal{C}(\mathcal{E}_{\boldsymbol{x}}, T)$ bits/symbol. Then, the capacity, in bits/second, for a filtered Gaussian-noise channel is

$$\mathcal{C}_c = \lim_{T\to\infty} \underbrace{\left(\frac{1}{T}\right)}_{\text{symbols/second}} \cdot \underbrace{\mathcal{C}(P_x T, T)}_{\text{bits/symbol}} \tag{2.209}$$

$$= \lim_{T\to\infty} \frac{\mathcal{C}(P_x T, T)}{T} \quad . \tag{2.210}$$

$\mathcal{C}(P_x T, T)$ is still the capacity in bits/symbol defined earlier, but the notation now emphasizes the dependence upon the symbol period T, which increases to infinity for each tonal channel in the AEP sense. Because $\widetilde{H}(f)$ is a one-to-one[49] transformation of $H(f)$, the sub channels each have their own capacity. Each subchannel has a gain equal to that its Fourier transform as $\bar{N} \to \infty$.

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

The capacity of the overall channel set simply adds their capacities:

$$\mathcal{C}(P_x T, T) = \sum_{i=0}^{\infty} \max\left[0, \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_i}{\sigma_i^2}\right)\right] \tag{2.211}$$

with maximizing transmit-energy allocation[50]:

$$P_x \cdot T = \sum_{i=0}^{\infty} \max\left[0, \lambda' - \sigma_i^2\right] \quad . \tag{2.212}$$

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

$$\mathscr{C} = \lim_{T\to\infty} \frac{1}{T} \cdot \sum_{i=0}^{\infty} \max\left[0, \frac{1}{2}\log_2\left(\frac{\lambda'}{\sigma_i^2}\right)\right] \quad , \tag{2.213}$$

and

$$P_x = \lim_{T\to\infty} \frac{1}{T} \cdot \sum_{i=0}^{\infty} \max\left[0, \lambda' - \sigma_i^2\right] \quad . \tag{2.214}$$

Both sums above are nonzero over the same range for f, which this text calls $\mathcal{F}_{opt}$. In the limit,

$$\sigma_i^2 \to \frac{\mathcal{S}_n(f)}{|H(f)|^2} \tag{2.215}$$

and

$$\frac{1}{T} \to df \quad , \tag{2.216}$$

leaving Shannon's famous "water-filling" scheme for the waveform channel's capacity calculation:

$$\mathscr{C} = \frac{1}{2}\int_{\mathcal{F}_{opt}} \log_2 \frac{\lambda' \cdot |H(f)|^2}{\mathcal{S}_n(f)} df \tag{2.217}$$

and

$$P_x = \int_{\mathcal{F}_{opt}} \left(\lambda' - \frac{\mathcal{S}_n(f)}{|H(f)|^2}\right) df \quad , \tag{2.218}$$

where the transmit spectrum is chosen to satisfy

$$\lambda' = \frac{S_n(f)}{|H(f)|^2} + S_x(f) \quad , \tag{2.219}$$

which results in the equivalent capacity expression

$$\mathscr{C} = \frac{1}{2}\int_{\mathcal{F}_{opt}} \log_2\left(1 + \frac{S_x(f)\cdot|H(f)|^2}{S_n(f)}\right) df = \frac{1}{2}\int_{\mathcal{F}_{opt}} \log_2\left(\lambda' \frac{|H(f)|^2}{S_n(f)}\right) \cdot df \quad . \tag{2.220}$$

[50] 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, known as the water-fill distribution (see Chapter 4)

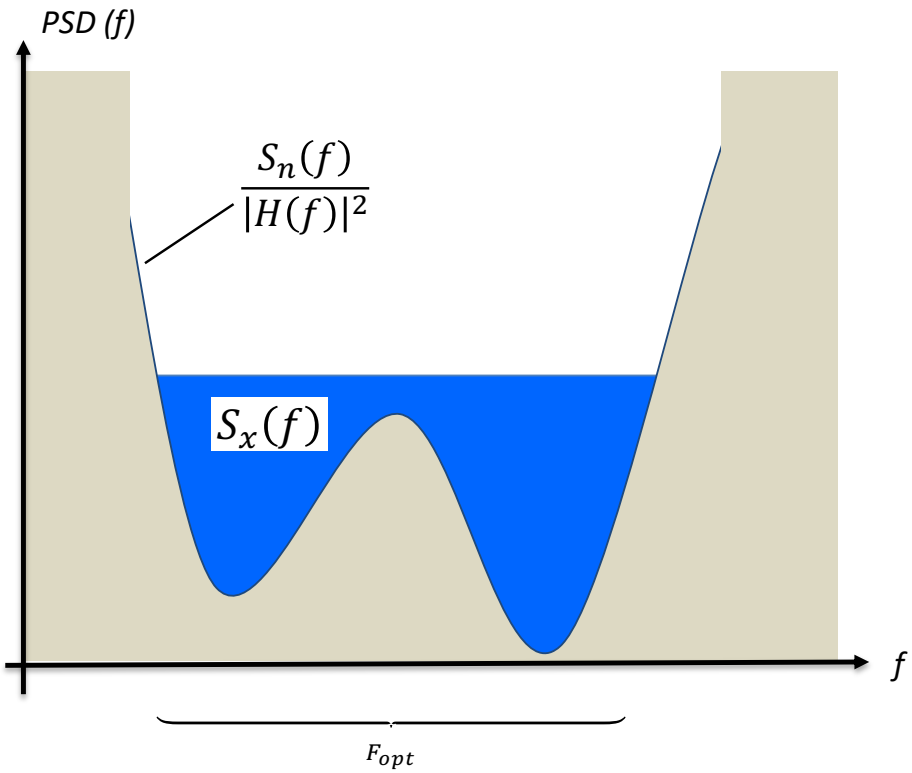


Figure 2.25: Continuous Water Filling.

Figure 2.25 illustrates the continuous water filling concept. The optimun transmit spectra computes as energy ("water-") "poured" into the inverted channel (multiplied by any noise power spectral density) until no energy remains, which determines both λ' and $\mathcal{F}_{opt}$. Then $\mathscr{C}$ is computed through (2.217) or (2.220).

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 $\mathscr{C} = W \cdot \log_2(1 + \text{SNR})$, or

$$\mathscr{C}_\infty = \lim_{W\to\infty} W \frac{1}{\ln 2} \ln\left(1 + \frac{P_x}{2W\sigma^2}\right) = \frac{1}{\ln 2} \cdot \frac{P_x}{2\sigma^2} \quad . \tag{2.221}$$

This result shows that even with infinite bandwidth, 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 AWGN channel with $H(f) = 1$ and $\mathcal{S}_n(f) = \frac{\mathcal{N}_0}{2}$. Then, one orthonormal set of basis functions is $\frac{1}{\sqrt{T}} \cdot \text{sinc}\left(\frac{t-iT}{T}\right) \cdot e^{-\jmath \frac{2\pi \cdot n \cdot t}{T}} \ \forall\ n \in \mathbb{Z}$; the noise samples have constant $\sigma_i^2 = \frac{\mathcal{N}_0}{2}\ \forall n$. Thus,

$$P_x = (\lambda' - \frac{\mathcal{N}_0}{2}) \cdot 2W \tag{2.222}$$

where $W = 1/2T$. Then,

$$\lambda' = \frac{P_x}{2W} + \frac{\mathcal{N}_0}{2} \tag{2.223}$$

and

$$\mathscr{C} = \left(\frac{1}{2} \cdot \log_2 \left[\frac{\frac{P_x}{2W} + \frac{\mathcal{N}_0}{2}}{\frac{\mathcal{N}_0}{2}} \right] \right) 2W = W \cdot \log_2 (1 + \text{SNR}) \quad , \tag{2.224}$$

the same result as obtained earlier in Equation (2.204).

EXAMPLE 2.5.2 ($1 + .9 \cdot e^{-j2\pi f}$ Channel Capacity) A second example is the low-pass channel with impulse response $h(t) = \text{sinc}(t) + .9 \cdot \text{sinc}(t-1)$ and AWGN with $\frac{\mathcal{N}_0}{2} = .181.$. Then

$$P_x = \int_{-W}^{W} \left(\lambda' - \frac{.181}{1.81 + 1.8 \cdot \cos(2\pi f)} \right) df \tag{2.225}$$

where W is implicitly in Hz for this example. If $P_x = 1$, the integral in (2.225) simplifies to[51]

$$\frac{1}{2} = \int_{0}^{W} \left(\lambda' - \frac{.181}{1.81 + 1.8 \cdot \cos(2\pi f)} \right) df \tag{2.226}$$

$$= \lambda' W - .181 \left\{ \frac{1}{\pi \cdot \sqrt{1.81^2 - 1.8^2}} \arctan \left[\sqrt{\frac{1.81 - 1.8}{1.81 + 1.8}} \cdot \tan(\pi W) \right] \right\} \tag{2.227}$$

At the bandedge W,

$$\lambda' = \frac{.181}{1.81 + 1.8 \cdot \cos(2\pi W)} \quad . \tag{2.228}$$

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

$$\frac{1}{2} = \frac{.181 W}{1.81 + 1.8 \cdot \cos(2\pi W)} - 0.3032 \cdot \arctan\left(.0526 \cdot \tan(\pi W)\right) \tag{2.229}$$

$W = .44$ approximately solves (2.229) and leaves $\lambda' = 1.33$.

The capacity is then

$$\mathscr{C} = \int_{0}^{.44} \log_2 \left(\frac{1.33}{.181} (1.81 + 1.8 \cos 2\pi f) \right) \cdot df \tag{2.230}$$

$$= \int_{0}^{.44} \log_2 7.35 \cdot df + \frac{1}{2\pi} \int_{0}^{.44} \log_2 (1.81 + 1.8 \cos 2\pi f) \cdot df \tag{2.231}$$

$$= 1.266 + .284 \tag{2.232}$$

$$\approx 1.55 \text{ bits/second} \quad . \tag{2.233}$$

Chapters 3, 4, and 5 will provide a more complete development of this example and band-limited channels' capacities.

[51] From a table of integrals with $a^2 > b^2$:

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

2.6 Multiuser Coding Basics

Transmission channels often accommodate more than one user, as in Figure 2.26'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 $p_{\boldsymbol{y}/\boldsymbol{x}}$, where Figure 2.26's two-user channel input has 2 users as elements $\boldsymbol{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2\}$, as does the channel output $\boldsymbol{y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2\}$. The conditional distribution $p_{\boldsymbol{y}/\boldsymbol{x}}$ and the input distribution $p_{\boldsymbol{x}}$ each derive from the joint distribution $p_{\boldsymbol{x}\boldsymbol{y}}$. These "multiuser" channels have three common foundational types that Subsection 2.6.1's Figures 2.29, 2.30, and 2.31 later illustrate respectively: the multiuser **multiple-access channel (MAC)**, **broadcast channel (BC)**, and **interference channel (IC)** for U users, $u = 1, ..., U$. The **user set** is $\boldsymbol{U} = \{1, 2, ..., U\}$, and thus $|\boldsymbol{U}| = U$. This chapter largely investigates the case where subsymbols can have L_x (usually complex) dimensions and tacitly in Section 2.3's AEP asymptotic sense $\bar{N}$ becomes infinite for each user's space-time MIMO channel. This AEP applies to each user independently when designing each and every users' encoder that maps $m_u \to \boldsymbol{v}_u$. Chapters 4 and 5 return Section 2.5's the multitone case where $\bar{N}$ becomes a number of frequency-indexed subsymbol dimensions within a symbol[52]. For most of this chapter's multiuser basics, $\bar{N} = 1$. Thus, an outer random-code-design process becomes tacit with reuse of the index n as a time-frequency symbol index within that tacit outer AEP-code-construction process.

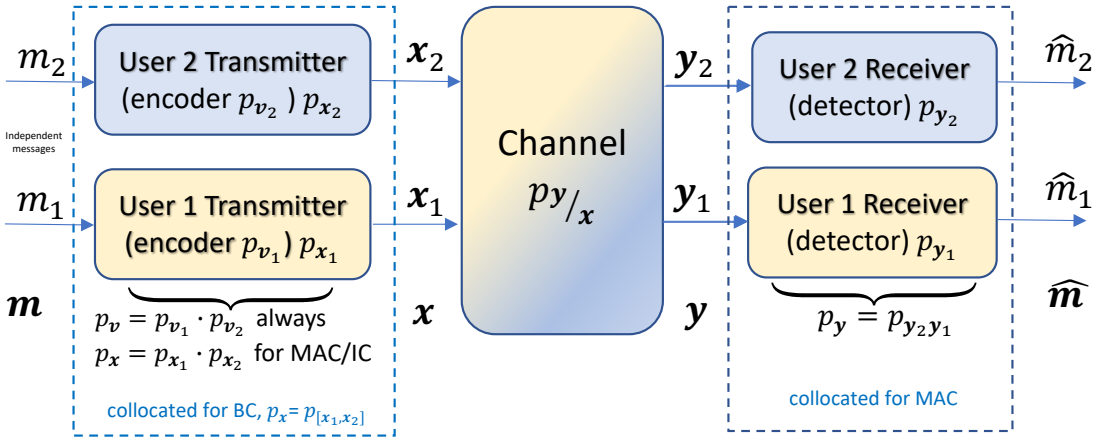


Figure 2.26: The multiuser channel.

These three multiuser-channel types respectively permit coordinated signal processing at the only the receiver/detector (MAC), only the transmitter/encoder (BC), or neither (IC). Other multiuser channels then are a nested combination and/or cascade of these 3 foundational types, as in Section 2.10. The term **"multiuser"** means all users' transmit symbols (and messages) are independent of any other user's corresponding transmit symbols (messages). This text's code designs are independent from user to user, and thus do not correspond to a joint typicality AEP design[53] Each user will use a good single-user code. Indeed all users may independently use different instances of the same good code, which is common in practice. 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 adhere to this multiuser definition. Relay channels are instead a special case of Section 2.10's expanded-user channels that circumvent inter-user dependencies with dependent cascades of independent expanded user sets.

[52] Allowing the possibility of $L_x \cdot \bar{N}$ total dimensions per "subsymbol" in the AEP context where codes apply overall to a sequence of such subbymbols. It is convenient here to call these subsymbols as "symbols," leaving the term "codeword" for the longer externally encoded system.

[53] This joint typicality design is possible but will not lead to better-performing systems, as becomes evident later.

The independent-user code construction supports Figure 2.27's and Definition 2.6.1's **capacity-region** $\mathcal{C}(\boldsymbol{b})$ characterization of the multiuser channel. Subsection 2.6.2 further expands Section 2.3's single-user mutual-information and capacity concepts, before then formally specifying this capacity rate region. More simply, multuser analysis expands "data-rate" to a two-dimensional **data-rate vector** $\boldsymbol{R} \triangleq [R_1, R_2]^t$, or similarly the corresponding **bits/symbol vector** is

$$\boldsymbol{b} \triangleq \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \boldsymbol{R} \cdot T = \begin{bmatrix} R_1 \cdot T \\ R_2 \cdot T \end{bmatrix} \quad , \tag{2.234}$$

where T remains symbol rate and common to all users[54].

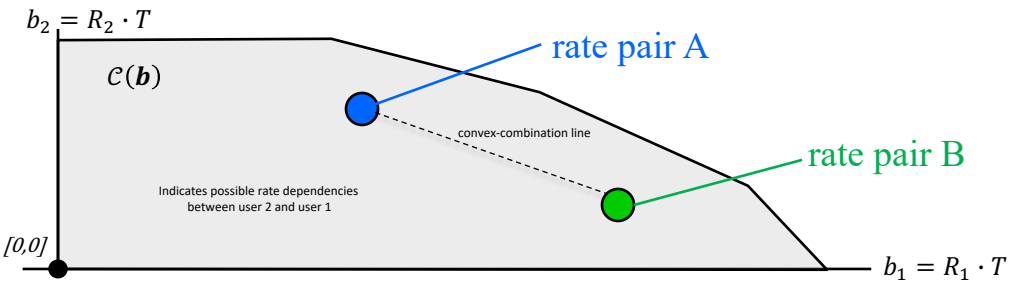


Figure 2.27: Rate Region.

Definition 2.6.1 *[Capacity Region] The capacity region $\mathcal{C}(\boldsymbol{b})$ is the set of all rate vectors $\boldsymbol{b}$ for users with independent messages, each encoded with its own single-user-capacity-achieving code, for which all users can be reliably decoded with average error probability $P_{e,u} \to 0$ by a MAP (ML with equally likely messages for all independent users' messages) detector at any receiver that must decode user u to deliver and forward its message.*

Figure 2.26'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 2.27. When Figure 2.27's shaded area is a rectangle, the users' reliably achievable data rates are independent, and thus single-user design and analysis applies to both users individually. Subsection 2.6.3 generalizes detection methods for multiuser channels. Subsection 2.6.2's various concepts and Subsection 2.6.3's detection lay a foundation for Subsection 2.6.4's general multiuser achievable-region and capacity-region specifications.

User Components or Subusers: Figure 2.28 illustrates **user components**. User components are independent message components of a single user, sometimes also called "subusers." The subuser does not fully carry the user's message, but it shares the same transmitter and receiver locations. The subusers' data rates (and energies for AWGN channels) simply sum to their parent user's data rate.

$$b_u = b_{u,a} + b_{u,b} \quad , \tag{2.235}$$

e.g. for the components a and b The subusers' utility is for other receivers that may elect to decode some of them (or all or none), if such detection and removal improves that other user's receiver performance. The subuser component decomposition may be important because only certain subuser-signal components (for instance the signal from effectively only one antenna of a MIMO transmitter) may reach a particular receiver. The identification of subuser possibilities typically depends upon the channel $p_{\boldsymbol{y}/\boldsymbol{x}}$ details - for instance, MIMO AWGN channels can decompose into a user component for each

[54] Such symbol-rate synchronization between users make take many forms that involve anything from global-positioning-satellite-based synchronization, to system symbol framing with various network clocks, to just making packets long enough in various approaches. Chapter 6 addresses synchronization methods.

specific user's (input, output, or both) dimensions/antennas. Other channels' potential user-component decompositions typically depend on the details of $p_{\boldsymbol{xy}}$ - generally speaking, any separately identified characteristic of the input $\boldsymbol{x}$ is a potential component index.

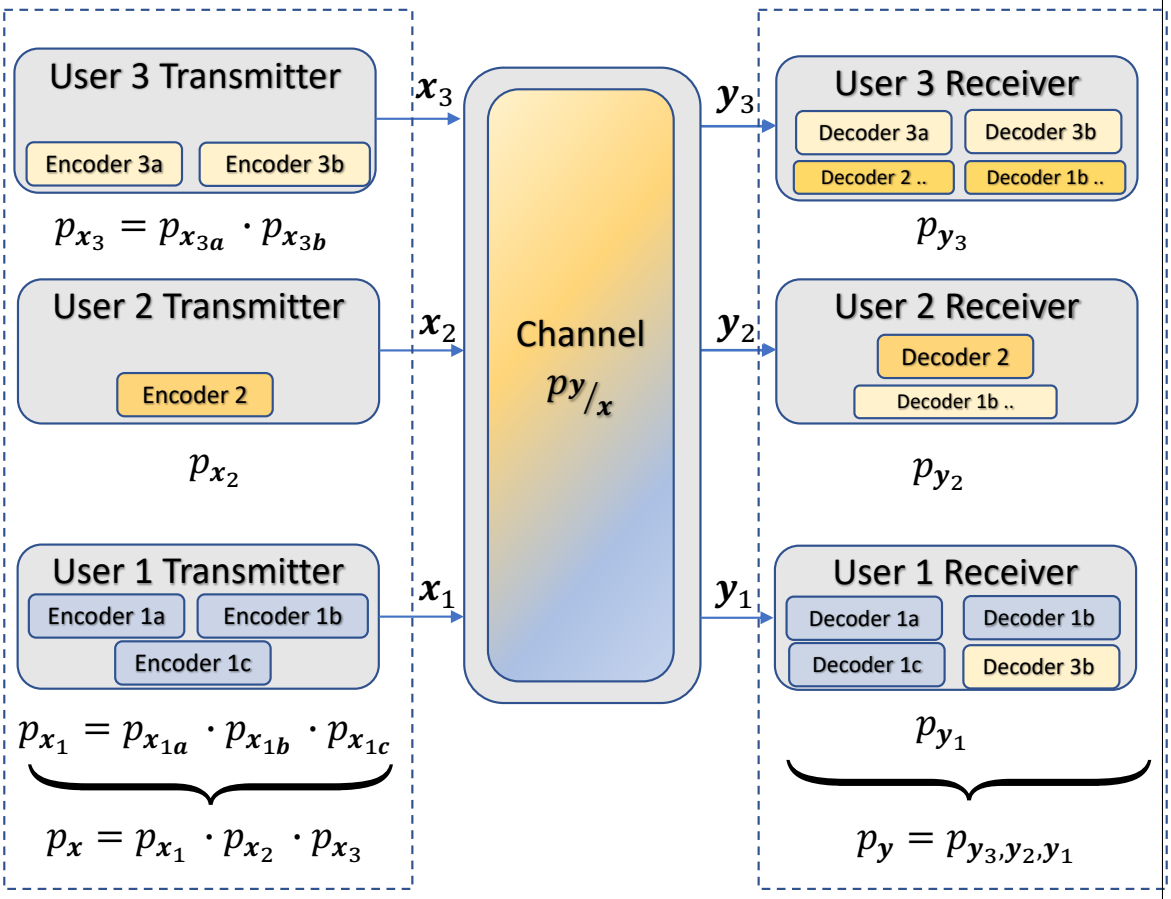


Figure 2.28: User Component Illustration.

Any particular receiver may then possibly decode that subuser if it is independent of another undetectable same-user component. Other subusers' prior detection allows their effect's removal from the desired user's detection. Figure 2.28 illustrates that receiver 3 may actually decode (and consequently remove) subuser 1b and user 2 before decoding user 3's two subusers. Similarly, receiver 2 may decode (and remove) 1b first, then decode 2, and so on. All subusers are independent, so the corresponding enlarged component-set input probability density $p_{\boldsymbol{x}}$ factors accordingly. In such cases, it is convenient to increase U to accommodate the total number of subusers perhaps $L_{x.u}$ of them each, and this text does so[55]. Thus, while there is a need in most general analyses to have only up to $U' = U^2$ potential subuser components, analytical simplification may increase components to even larger numbers when many antennas or dimensions find use. Then, the larger dimensional rate region can collapse to the original users through the rate sums, $u = 1...U$ (U is original number of users)

$$b_u = \sum_{\ell=1}^{L_u} b_{u,\ell} \quad . \tag{2.236}$$

[55]Reuse of $L_{x,u}$ for components/user and spatial dimensions/user is often consistent, avoiding further notational complexity; some situations may require further indexing that will find use only if needed in this text.

Convex Hull, a.k.a time/vertex sharing: User components also support the concept of "time-sharing," or more generally, "dimensional sharing." In effect, sharing a dimension means subdividing its use into two (or more) components that each apply a fraction of the "time" (dimensions). Each fraction (and its associated independent AEP-sense good-code encoder) effectively generates a component. This type of subuser/component indexing can be particularly useful also in MIMO AWGN channels with many antennas per user. Often this text uses U', instead of U, when the larger number of components' ($U' > U$) specific enumeration helps analysis.

The **convex hull** of a set of K rate vectors $\{\boldsymbol{b}_k\}_{k=1,...,K}$ is $\{\boldsymbol{b} \mid \sum_{k=1}^{K} \gamma_k \cdot \boldsymbol{b}_k\}$ where $0 \leq \gamma_k \leq 1$ ($\gamma_k \in \mathbb{R}^+ \; \forall \; k = 1, ..., K$) and $\sum_{k=1}^{K} \gamma_k = 1$. The convex-hull parameters γ_k in practice will have a least common divisor $\delta \triangleq LCD(\{\gamma_k\})$, which tacitly implies a dimensionality expansion by

$$L_x \to \delta^{-1} \cdot L_x \quad . \tag{2.237}$$

This dimensional subdivision into a larger-dimensional set allows application of multiple independent encoders for each user; each individual user's encoder operates (with rate vector $\boldsymbol{b}_k$) for a fraction γ_k of the expanded dimension set, constituting then a set of subusers. This convex hull operation thus creates and exploits user components to include possible other rate vectors in $\mathcal{C}(\boldsymbol{b})$. This text uses decoding order to enumerate reliably achievable rate vectors $\boldsymbol{b}_k$ or "vertices," followed by a convex hull over all vertex points corresponding to each of the decoding orders. This approach creates the largest superset of possible component decompositions for any multiuser channel and thereby simplifies the specification of $\mathcal{C}(\boldsymbol{b})$. It will raise operational questions about sample-average energy time period that later sections address. It also specifies by a finite-size order set an alternative to the multiuser random-code random-seeding in [12].

Often a user has a single component and so the terms "user" and "component" can interchange; however, there will be cases where an individual user may have more than one component.

Macro Users: A **macro user**, or more precisely possibly macro component, occurs when 2 or more independent components have identical impact; formally, their positions' interchange within $\boldsymbol{x}$ does not change $p_{\boldsymbol{xy}}$. Effectively, there is no need to design separately for each of these two identical components. These components' aggregation into a macro component can sometimes simplify $\mathcal{C}(\boldsymbol{b})$ construction. For instance the channel $\boldsymbol{y} = \boldsymbol{x}_1 + \boldsymbol{x}_2 + \boldsymbol{n}$ with $p_{\boldsymbol{x}_1} = p_{\boldsymbol{x}_2}$ is essentially a single-user channel for which the capacity would be $\bar{b} = \overline{\mathcal{C}} = \frac{1}{2} \cdot \log_2(1 + \frac{2\mathcal{E}_{\boldsymbol{x}}}{\sigma^2})$. Any rate splitting into non-negative bits/subsymbol $\bar{b}_1 + \bar{b}_2 = \bar{b}$ is trivially possible by reversing the macro user into the two original users (subuser components from the macro-user's perspective).

Some basic mutual-information bounds: Two multiuser mutual-information types of interest are the mutual information corresponding to the maximum of all users' **sum bits/symbol** $b \leq \mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$ and the **individual-user mutual information** corresponding to each user, $\mathcal{I}(\boldsymbol{x}_u; \boldsymbol{y})$, $u = 1, ..., U$.[56] The word maximum implies good code here, via Section 2.3's single-user asymptotic considerations. If each user has **per-user bits per symbol** b_u, $u = 1, ..., U$, then

$$b = \sum_{u=1}^{U} b_u \leq \mathcal{I}(\boldsymbol{x}; \boldsymbol{y}) \quad , \tag{2.238}$$

with the inequality constraint imposed because different users' channel inputs, channel outputs, or both may not permit coordination of modulators and/or detectors, even with the best codes.[57] Also, a bound for user u's **best average number of bits** is

$$\mathcal{I}(\boldsymbol{x}_u; \boldsymbol{y}) \leq \mathcal{I}(\boldsymbol{x}; \boldsymbol{y}) \quad , \tag{2.239}$$

[56] The individual mutual information rates for a particular output $\mathcal{I}(\boldsymbol{x}_u; \boldsymbol{y}_u)$ are also of interest in the interference channel.

[57] There is a presumption of a common symbol period in multiuser 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 needs consideration in practice, usually be synchronizing all users to a common symbol clock.

where $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$'s calculation averages the other $U-1$ users over their distributions. Equation (2.239) uses distributions with $\boldsymbol{\chi}$ as the integration variable for the random vector $\boldsymbol{x}$ and $\boldsymbol{v}$ as the integration variable for the random vector $\boldsymbol{y}$, so the marginal distribution is[58]

$$p_{\boldsymbol{x}_u}(\boldsymbol{\chi}_u) = \int_{\boldsymbol{\chi}\in\{\boldsymbol{x}\setminus\boldsymbol{x}_u\}} p_{\boldsymbol{x}}(\boldsymbol{\chi})\cdot d\boldsymbol{\chi} \tag{2.240}$$

and[59]

$$p_{\boldsymbol{x}_u,\boldsymbol{y}}(\boldsymbol{\chi}_u,\boldsymbol{v}) = \int_{\boldsymbol{\chi}\in\{\boldsymbol{x}\setminus\boldsymbol{x}_u\}} p_{\boldsymbol{x},\boldsymbol{y}}(\boldsymbol{\chi},\boldsymbol{v})\cdot d\boldsymbol{\chi} \quad . \tag{2.241}$$

The **set-removal operation** indicated by $\setminus$ removes an element (or subset) from a larger set. The mutual information $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$ has a conditional form when given other users $\boldsymbol{u}\subseteq\{\boldsymbol{U}\setminus u\}$, that is the **conditional mutual information per user** $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{x}_{\boldsymbol{u}})$. The conditional mutual information effectively presumes reliable ($P_e\to 0$) prior decisions on other users' ($\boldsymbol{x}_{\boldsymbol{u}}$'s) codewords, which essentially removes those other signals degrading effect upon subsequent user u's symbol detection. Thus, $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$ is not necessarily an upper bound for user u's data rate b_u in the multiuser channel.

2.6.1 Multiuser Channel Types

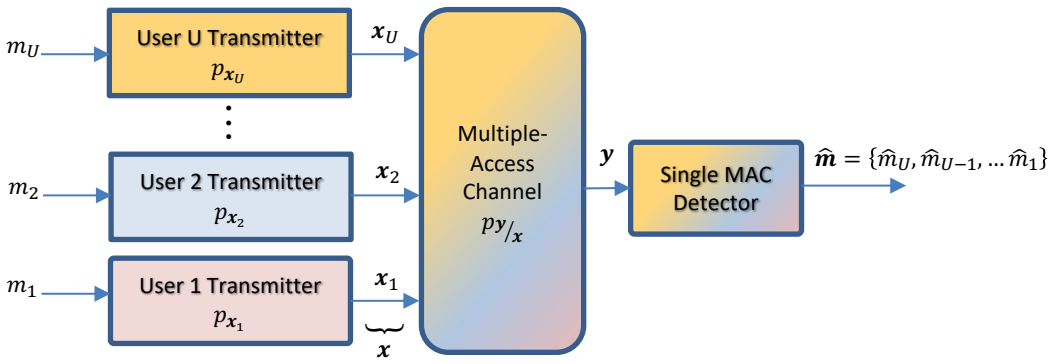


Figure 2.29: The multiple-access channel.

Multiple-Access Channel (MAC): Figure 2.29's single receiver characterizes the **multiple-access channel (MAC)**, which conveys independent messages $m_u, u=1,\ldots,U$ from U physically separated encoders, each with its own transmit symbol vector $\boldsymbol{x}_u$, $u=1,\ldots,U$. The single detector accepts the single channel-output vector $\boldsymbol{y}$ and provides the detected outputs $\hat{m}_u, u=1,\ldots,U$. There are thus multiple users that access the channel with a single output, whence the name "multiple-access" channel. The single output vector $\boldsymbol{y}$ may have $L_y>1$ and/or[60] $\bar{N}\geq 1$. Similarly, each input has $L_{x,u}$ spatial dimensions. Both input and output may also possibly have $\bar{N}$ frequency dimensions if Section 2.5's MT systems find use (Chapter 4 provides detail on finite-dimensional $\bar{N}<\infty$ MT systems). When $L_y>1$ or an of $L_{x,u}>1$, it is a **vector multiple access** channel. All users need not have the same number of dimensions – when this happens, MAC analysis ensures a consistent user-dimension bookkeeping[61]. The MAC's single output allows a single "coordinated" receiver to detect jointly all the transmitted user

[58] 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.

[59] This is a sum, instead of an integral, if $\boldsymbol{x}$ is a discrete random vector.

[60] Usually the case where $\bar{N}>1$ occurs in Chapter 4's multi-tone designs where each tone can be separately viewed as a multiuser channel by itself, with $\bar{N}\to\infty$ in the good-code/AEP sense.

[61] As becomes evident in some of the later matlab software's use of a cell array specifying the possibly variable number of spatial dimensions for users

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 own independent input message m_u.

Multiple-access channels abound in communication, 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" (Wi-Fi) or "base station" (cell tower). Such MAC architectures occur in cellular and most Wi-Fi systems, and are vector MACs when the base station (or access point) has multiple receive antennas. The upstream[62] direction of a cable or passive-optical network is also a scalar-output MAC example with various residential customers all sharing a common dimension 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. Similarly uplinks to a common satellite also qualify as MACs.

Of interest in multiple-access channels are the maximum reliably achievable data rates for each user (which may depend upon 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's best implementation in more detail.

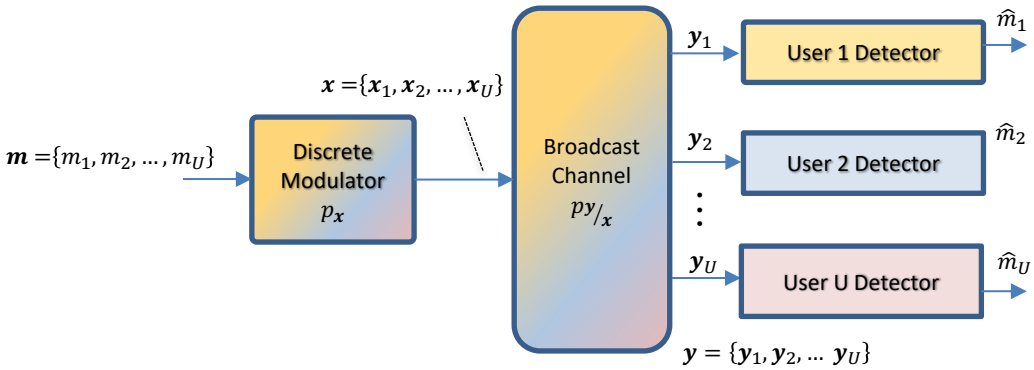


Figure 2.30: The broadcast channel.

Broadcast Channel (BC): The MAC's dual is Figure 2.30's **broadcast channel (BC)** in which a single BC transmitter generates all independent users' common channel-input symbol $\boldsymbol{x}$. The BC has U physically separated channel outputs $\boldsymbol{y}_u$. When the single input is a vector, the BC is **vector broadcast**. Each independent BC message $m_u, u = 1, ..., U$ is within the common single transmitter's channel-input ("broadcast") symbol $\boldsymbol{x}$, and so there is transmit-signal "coordination" – which has separate encoding of all users' message signals, so $p_{\boldsymbol{m}} = \prod_{n=1}^{\tilde{N}} p_{m,u}$, but the corresponding $p_{\boldsymbol{x}}$ may not so factor directly. The BC order convention typically reverses this (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 best BC transmitter/receiver designs, while Chapter 5 further investigates the vector BC.

BC examples include MAC's opposite-direction transmissions from a service provider to a customer: For instance a "downlink" in wireless cellular or Wi-Fi 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). The downlink of a satellite channel to multiple earth points is also a BC.

[62] Upstream is the term used for customers' transmissions to a central site in cable, fiber, powerline, and DSL internet access, while "uplink" is the term used in wireless for transmissions from "devices" to base stations or access points, even though it is not (yet) a (recognized) word (upstream has long been a word, prior to uplink).

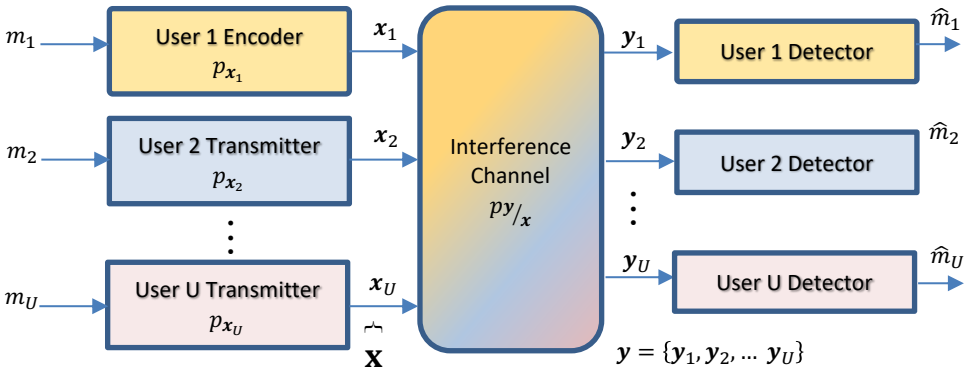


Figure 2.31: The interference channel.

Interference Channel (IC): The third channel type is Figure 2.31's **interference channel (IC)**. Coordination of neither inputs nor outputs is possible. Each of U possible receivers can attempt independent detection of one, some, or all the U transmitted messages (or their up to $U' = U^2$ components). In the **vector** interference channel, some 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.

Linear additive-noise multiuser 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 thus also $\bar{N} \to \infty$. However, there can remain spatially[63] $L_x < \infty$ transmit dimensions and $L_y < \infty$ receive dimensions, $L \triangleq \max(L_x, L_y)$. The presumed asymptotically Gaussian codes ($\Gamma = 0$ dB) will thus have N real dimensions or $\bar{N}$ complex dimensions (with $\tilde{N} = 2$) with presumed $\bar{N} \to \infty$ (but not shown explicitly).

Strictly speaking, in the infinite dimensional context of implied codewords, Sections 2.1 - 2.5's notation to this point would call the codeword's subsymbol channel inputs $\tilde{\boldsymbol{x}}$. 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 input notation will relax to simply $\boldsymbol{x}$.

Table 2.2 lists the possible input and output dimension sizes for each simple multiuser Gaussian channel.

When there are $\bar{N}$ frequency-indexed dimensions (like Section 2.5.4's multi-tone), then multiuser concepts can apply independently on each such indexed AWGN channel/tone (eliminating the need to carry the extra notation and index), as in Chapter 4. Effectively then, there will be $\overline{N}$ channels of the type in Table 2.2. The matrix AWGN nevertheless remains:

$$\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n} \quad . \tag{2.242}$$

For the MAC, any of Table 2.2's H_u (or $H_{u'u}$) has singular value decomposition (SVD) $H_u = F_u \cdot \Lambda_u \cdot M_u^*$ (or doubly indexed for IC) and that user's transmitter could pre-multiply $\boldsymbol{x}_u$ by M_u and/or that user's receiver could post-multply $\boldsymbol{y}$ by F_u^* without loss for that user u. This pre- and post-processing then leaves ϱ_{H_u} scalar dimensions to carry data optimally, where ϱ_{H_u} is the rank of H_u.

[63] Or otherwise equivalently separate user transmitters or receiver locations

Type	$\boldsymbol{x}$ Number of inputs	$\boldsymbol{y}$ Number of outputs	H
multiple access	$U \cdot L_x$ or $\mathscr{L}_x = \sum_{u=1}^{U} L_{x,u}$	L_y	$[H_U \ ... \ H_2 \ H_1]$
broadcast	L_x	$U \cdot L_y$ or $\mathscr{L}_y = \sum_{u=1}^{U} L_{y,u}$	$\begin{bmatrix} H_1 \\ \vdots \\ H_{U-1} \\ H_U \end{bmatrix}$
interference	$U \cdot L_x$ or $\mathscr{L}_x = \sum_{u=1}^{U} L_{x,u}$	$U \cdot L_y$ or $\mathscr{L}_y = \sum_{u=1}^{U} L_{y,u}$	$\begin{bmatrix} H_{UU} & ... & H_{U1} \\ \vdots & \ddots & \vdots \\ H_{2U} & ... & H_{21} \\ H_{1U} & ... & H_{11} \end{bmatrix}$

Table 2.2: Multiuser Gaussian ($\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n}$) Channel Dimensionality Table.

Each MAC input may have a user-variable $L_{x,u}$ and each BC output may have a user-variable $L_{y,u}$, and indeed the IC could have both. In most cases, multiuser analysis can add dummy dimensions so that the $L = \max_u(\varrho_{H_u})$ is the largest over all users, but this text's matlab programs often accommodate variable per-user dimensionality to match the theory here. 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 one user component for each such usable dimension.

Figure 2.26 characterizes a multiuser transmission channel by the conditional probability distribution $p_{\boldsymbol{y}/\boldsymbol{x}}$, consistent with single-user channels. The input probability distribution also consistently is $p_{\boldsymbol{x}}$. All other probability distributions consequently derive from $p_{\boldsymbol{x}\boldsymbol{y}} = p_{\boldsymbol{y}/\boldsymbol{x}} \cdot p_{\boldsymbol{x}}$. Because the user inputs are independent, the input distribution also must factor[64] as:

$$p_{\boldsymbol{m}} = \prod_{u=1}^{U} p_{m,u} \quad . \tag{2.243}$$

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 $p_{\boldsymbol{x}}$ may also factor, but this is not necessarily always true, although an equivalent input $p_{\boldsymbol{v}} = \prod_{u=1}^{U} p_{\boldsymbol{v}_u}$ will instead represent independent user messages with $\boldsymbol{v}_u$ in 1-to-1 relationship with non-zero-probability elements of the $\boldsymbol{x}_u$, the latter viewed as a Hilbert Space, and of course 1-to-1 with m_u. Then, $p_{\boldsymbol{v}} = \prod_{u=1}^{U} p_{v,u}$ always holds.

2.6.2 Order, Data Rates, and Rate Regions

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 transmitters' messages or subuser message components. However, perhaps not all receivers can decode them reliably. Any receiver-input signal correspondingly

[64] Subusers message components also will be independent when subusers find use.

decomposes into detectable and un-detectable user-components' messages. Within the aggregate of receivers (if there are more than 1, as in BC and IC or more general multiuser channel types), each subuser has a limit of the minimum mutual information over all receivers that necessarily must decode that ser for the given order, and then that subuser is "decodable everywhere needed" at that maximum data rate equal to this minimum mutual information. This minimum mutual information vector (or vertex) $\boldsymbol{\mathcal{I}}_{min}$ approach allows full characterization of all multiuser-channel capacity regions, specifically exploiting the observation that decomposition of any specific user maximally needs U subusers, so $U' \leq U^2 < \infty$ is the maximum number of necessary subusers needed to consider every possibility of decoding orders/choices.

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

A user u may have a subuser component that is decodable (or not) at each receiver, leading to a maximum of U^2 user components. Figure 2.32 illustrates this concept for $U = 3$ users, and thus up to $U^2 = 9$ components for a specific order $\pi_1(u, u') = \pi_1(i)$ at receiver 1 where the single i corresponds to the ordered pair of input indices for the users' transmitters and receivers, $i = 1, ..., U^2 = 9$.

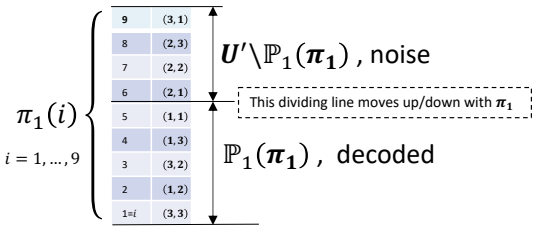


Figure 2.32: Example: receiver $u = 1$'s order/prior-user set.

There are sets of users

$$\begin{aligned} S_1 &\triangleq \{(1,1),(1,2),(1,3)\} \\ S_2 &\triangleq \{(2,1),(2,2),(2,3)\} \\ S_3 &\triangleq \{(3,1),(3,2),(3,3)\} \\ S &= S_1 \bigcup S_2 \bigcup S_3 \, . \end{aligned}$$

Receiver 1 must decode all elements in set S_1 and similarly for receivers $u = 2, 3$ with sets S_2 amd S_3. In Figure 2.32, receiver 1 decodes by position $i = 5$ all 3 user-1 components and one each from users 2 and 3. Any higher components are "noise" (more generally averaged in marginal-distribution calculation). For a given order, any receiver u' must decode also all elements i below the pair in the highest position that contains (u, u'). There are $U^2! = 9!$ possible orders for each of the 3 receivers (only receiver 1 appears). There are thus $(U^2!)^U = (9!)^3 < \infty$ possible $\boldsymbol{\Pi} = [\boldsymbol{\pi}_3 \ \boldsymbol{\pi}_2 \ \boldsymbol{\pi}_1]$ overall multiuser orders. Successive decoding occurs from the bottom up at receiver 1, following the chain rule.

The concept that a user component 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 or subusers) in each such mutual-information term, leaving only the unconditional marginal components to average in contribution to the specific mutual-information chain-rule term. There are many chain-rule decompositions that depend upon Definition 2.6.2's formalization of user order, $\boldsymbol{\Pi}$.

Definition 2.6.2 **[User Order and Prior-User Set]** *In general, the multiuser order matrix aggregates the user-receivers order vectors into a* $U^2 \times U$ **order matrix** $\mathbf{\Pi} = [\boldsymbol{\pi}_1 \ \ldots \ \boldsymbol{\pi}_U]$. *There are thus* $(U^2!)^U$ *possible orderings or* $|\mathbf{\Pi}| = (U^2!)^U$. *User* u*'s data rate (bits/symbol)* b_u *sums all its components that receiver* u *must decode, thus in set* S_u. *Then minimizing over the single index* i *from bottom to top in the given order* $\boldsymbol{\pi}_u$ *at receiver* u

$$i_u^*(\boldsymbol{\pi}_u) \stackrel{\Delta}{=} arg\min_j \left[\pi_u(S_u) \subseteq \{1:j\}\right] \quad , \tag{2.244}$$

and thus the **prior-user set** *is*

$$\mathbb{P}_u(\boldsymbol{\pi}_{i^*}) = \{(u,i) \mid \pi(u,i) \leq i_u^*\} \quad . \tag{2.245}$$

$\mathbb{P}_u(\boldsymbol{\pi})$*'s argument* $\boldsymbol{\pi}$ *can be any order vector, but requires that the index in (2.244) be found for that order.*

Each receiver u can attempt to process all users by decoding all those prior to (sub)user u's position in a $U' \times 1$ order vector $\boldsymbol{\pi}_u$. Alternatively, the discrete input/output function $j = \pi_u(i)$ provides the order for position i (counting $i = 1$ in bottom row and $i = U'$ at top). Receiver u decodes the users $\pi_u(i)$ that occur in lower positions than the i such that $\pi_u^{-1}(i) < \pi_u^{-1}(u)$. Thus, the vector $\boldsymbol{\pi}_u$ has specific form

$$\boldsymbol{\pi}_u = \begin{bmatrix} \pi_u(U') \\ \vdots \\ \pi_u(1) \end{bmatrix} \quad . \tag{2.246}$$

Correspondingly, $\pi_u^{-1}(j) = i$ is the position at which receiver u decodes user j. The inverse order vector trivially (but for completeness) is

$$\boldsymbol{\pi}_u^{-1} = \begin{bmatrix} U' \\ \vdots \\ 1 \end{bmatrix} \quad , \tag{2.247}$$

Those users above receiver/user u's position in $\pi_u^{-1}(i) > \pi_u^{-1}(u)$ are averaged (treated as noise) to compute the marginal probability distribution that the detector design uses.

Any mutual information across multiple user 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 another user, at data rate $b_{i \neq u}$, is not reliably decodable at any receiver in the given order – where i precedes u, then b_u is also not achievable for the given probability distribution $p_{\boldsymbol{xy}}$ and that given order.

vertices: After enumeration of reliably decodable $\boldsymbol{b}(\mathbf{\Pi})$, or the vertices, for each $\mathbf{\Pi}$ choice, a convex-hull operation in effect expands into subusers that are decodable, possibly sharing the two endpoint vertices (and corresponding sub codes) used to time/dimension-share/average to this intermediate non-vertex point. Subuser rates are added to form user rates for vertices.

The possible orders become fundamental to multiuser transmission's optima. Within an order, an un-detectable other-subuser's component simply becomes "noise" (the decoder design sums/integrates, effectively averages, that subuser's contribution to the channel's conditional-probability distribution) for the receivers that are unable to detect it. Within an order, the detectable subusers are best set as the "given" subusers. A search over all subusers' possible orders $\mathbf{\Pi}$ helps specify fundamental limits.

For an example where each user has 1 subuser component, suppose $\boldsymbol{\pi}_2 = [321]^*$, then receiver 2 attempts to decode first user 1 before decoding user 2; receiver 2 treats user 3 as noise for both detections. By contrast, if $\boldsymbol{\pi}_2 = [231]$, 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, $\mathbf{\Pi}$ can characterize up to $(U!)^U$ different multiuser-order possibilities for decoder

design. (Fortunately, as this chapter progresses, considerable elimination of such order possibilities often trivially occurs, but not always.). For any order $\boldsymbol{\pi}$, the set $\mathbb{P}_u(\boldsymbol{\pi})$ may or may not be reliably decodable, which depends on the vector $\boldsymbol{b}$ corresponding to the U users' code choices/data rates.

rcvr/ position i	$\pi_4(i)$	$\pi_3(i)$	$\pi_2(i)$	$\pi_1(i)$
$i=4$	3	3	4	3
$i=3$	4	2	3	2
$i=2$	1	4	2	1
$i=1$	2	1	1	4
$\mathbb{P}_u(\boldsymbol{\pi}_u)$	{1,2}	{2,4,1}	{1}	{4}

$\mathfrak{I}$	$\mathfrak{I}_4$	$\mathfrak{I}_3$	$\mathfrak{I}_2$	$\mathfrak{I}_1$
top	∞	$\mathfrak{I}_3(3/1,2,4)$ 20	∞	∞
	$\mathfrak{I}_4(4/1,2)$ 10	$\mathfrak{I}_3(2/1,4)$ 9	∞	∞
	$\mathfrak{I}_4(1/2)$ 5	$\mathfrak{I}(4/1)$ 4	$\mathfrak{I}_2(2/1)$ 4	$\mathfrak{I}_1(1/4)$ 2
bottom	$\mathfrak{I}_4(2)$ 1	$\mathfrak{I}(1)$ 2	$\mathfrak{I}_2(1)$ 2	$\mathfrak{I}_1(4)$ 5

$$\boldsymbol{\Pi} = [\boldsymbol{\pi}_4 \ \boldsymbol{\pi}_3 \ \boldsymbol{\pi}_2 \ \boldsymbol{\pi}_1] = \begin{bmatrix} 3 & 3 & 4 & 3 \\ 4 & 2 & 3 & 2 \\ 1 & 4 & 2 & 1 \\ 2 & 1 & 1 & 4 \end{bmatrix}$$

$$\mathfrak{I}_{min} = \begin{bmatrix} 4 \\ 20 \\ 1 \\ 2 \end{bmatrix}$$

Table 2.3: Order Example.

Table 2.3 further illustrates an order table (on the left) for a 4-user channel. To simplify in this example, each user has been verified externally to need only 1 subuser Thus, this is is one of $(4!)^4$ possible orders. More generally there would be $(16!)^4$ orders and 16 entries in each table row, where again subuser rates add to their parent user's rates. The prior-user sets for each receiver also appear in Table 2.3's bottom row: A **mutual-information-like vector** exists for each user's decoding at receiver u and for that receiver's order $\boldsymbol{\pi}_u$:

$$\boldsymbol{\mathscr{I}}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) = \begin{bmatrix} \mathscr{I}_u\left(\boldsymbol{x}_{\pi_u(U')}; \boldsymbol{y}_u / \mathbb{P}_{\pi_u(U)}(\boldsymbol{\pi}_u)\right] \\ \vdots \\ \mathscr{I}_u\left(\boldsymbol{x}_{\pi_u(i)}; \boldsymbol{y}_u / \mathbb{P}_{\pi_u(i)}(\boldsymbol{\pi}_u)\right) \\ \vdots \\ \mathscr{I}_u\left(\boldsymbol{x}_{\pi_u(1)}; \boldsymbol{y}_u / \mathbb{P}_{\pi_u(1)}(\boldsymbol{\pi}_u)\right] \end{bmatrix} , \tag{2.248}$$

with implied order $\boldsymbol{\Pi}$ and the joint distribution $p_{\boldsymbol{xy}}$ but not shown explicitly on (2.248)'s right to avoid burdensome notation. The mutual-information-like quantity $\mathscr{I}_u$ is

$$\mathscr{I}_u\left(\boldsymbol{x}_{\pi_u(i)}; \boldsymbol{y}_u / \mathbb{P}_{\pi_u(i)}(\boldsymbol{\pi}_u)\right) \triangleq \begin{cases} \infty & i > \pi_u^{-1}(u) \\ \mathcal{I}_u\left(\boldsymbol{x}_{\pi_u(i)}; \boldsymbol{y}_u / \mathbb{P}_{\pi_u(i)}(\boldsymbol{\pi}_u)\right) & i \leq \pi_u^{-1}(u) \end{cases} . \tag{2.249}$$

The ∞ 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 $\mathcal{I}_u$ value for the particular order. Table 2.3 provides finite example values in red, while listing ∞ 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. This vector exists for all choices of $\boldsymbol{\Pi}$, and the corresponding convex hull therefore creates an achievable rate region.

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

Definition 2.6.3 [Decodable Set and Minimum Mutual Information Vector] *For a given $\mathbf{\Pi}$, $p_{\boldsymbol{xy}}$, and $\boldsymbol{b}$, each receiver u will be able to detect reliably (on average in the AEP sense) other ($i \neq u$) subusers (user components) in the set*

$$i \in \mathscr{D}_u(\mathbf{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b}) \tag{2.250}$$

with $P_e \to 0$, When receiver u can detect no other users reliably, $\mathscr{D}_u(\mathbf{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b}) = \emptyset$, **with this order**.

Every multiuser channel has a minimum mutual-information vector with components

$$\mathcal{I}_{min,u}(\mathbf{\Pi}, p_{\boldsymbol{xy}}) = \min_j \left\{ \mathscr{I}_j \left(\boldsymbol{x}_{\pi_j(u)}; \boldsymbol{y}_j / \mathbb{P}_{\pi_j(u)}(\boldsymbol{\pi}_j) \right) \right\} \quad , \tag{2.251}$$

and thus the **minimum mutual-information vector**, *or vertex, is*

$$\boldsymbol{\mathcal{I}}_{min}(\mathbf{\Pi}, p_{\boldsymbol{xy}}) = \begin{bmatrix} \mathcal{I}_{min,U'}(\mathbf{\Pi}, p_{\boldsymbol{xy}}) \\ \vdots \\ \mathcal{I}_{min,u}(\mathbf{\Pi}, p_{\boldsymbol{xy}}) \\ \vdots \\ \mathcal{I}_{min,1}(\mathbf{\Pi}, p_{\boldsymbol{xy}}) \end{bmatrix} \quad . \tag{2.252}$$

In the most general case, the $\mathcal{I}_{min}$ vector entries are sums of each user's subuser component rates that are minimally decodable everywhere; where more than one single user u's subuser components at any receiver i are decodable within the order, then the $\mathcal{I}_{min}(i)$ calculations should sum those reliably decodable components' subrates at receiver i before comparing the minima across all receivers.

This definition helps then determine the best decodable set:

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

$$\boldsymbol{b} \preceq \boldsymbol{\mathcal{I}}_{min}(\mathbf{\Pi}, p_{\boldsymbol{xy}}) \quad , \tag{2.253}$$

then

$$\mathbb{P}_u(\boldsymbol{\pi}_u) \subseteq \mathscr{D}_u(\mathbf{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b}) \tag{2.254}$$

and receiver u reliably achieves the data rate $b = \mathcal{I}_u(\boldsymbol{x}_u; \boldsymbol{y}_u / \mathbb{P}_u(\boldsymbol{\pi}_u))$ with order $\boldsymbol{\pi}_u$.

proof: When (2.254) is met, then for user u to be decodable at receiver u, a search over other users' decoders $i \neq u$, and users $j \in \mathbb{P}_u(\boldsymbol{\pi}_u)$, can find a solution with each user reliably decodable wherever necessary if the condition

$$\mathscr{I}_u(\boldsymbol{x}_{\pi_u(j)}; \boldsymbol{y}_u / \boldsymbol{x}_{[\mathbb{P}_{\pi_u(j)}(\boldsymbol{\pi}_u)]})) \geq \mathscr{I}_i(\boldsymbol{x}_{\pi_u(j)}; \boldsymbol{y}_i / \boldsymbol{x}_{[\mathbb{P}_{\pi_u(j)}(\boldsymbol{\pi}_i)]})) \; \forall \; \{ i \neq u \; \wedge j \in \mathbb{P}_u(\boldsymbol{\pi}_u) \} \tag{2.255}$$

is met. The search in (2.255) may be best understood by looking at Table 2.3's example). If (2.253) holds, then receiver u can reliably decode all the prior users that any other receiver (including u at receiver u) for the given order $\mathbf{\Pi}$. This is because $\mathcal{I}_{min}(u)$ 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 $p_{\boldsymbol{x}_u}$ as in the above AEP observation. If (2.253) does not hold, then this

decoding order does not support user u's reliable detection with any codes chosen at the specified rate $\boldsymbol{b}$. **QED.**

Section 2.6.4 will use achievable $\boldsymbol{\mathcal{I}}_{min}$ vertices to specify the general capacity region. This text's approach tacitly presumes $U \to U' \leq U^2$ wherever subusers may be enumerated, enlarging the potential searches. This may be complex for some channels, and indeed perhaps makes the ensuing general capacity regions somewhat circular in that they will depend on knowledge of the user components (begging the criticism that the capacity region is somewhat circularly defined). However, the designs that this text considers have clear identifiable components.

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

Definition 2.6.4 [Cross-User Set] *The* **cross-user set** *of receiver indices at which u can be reliably decoded (on average in the AEP sense) is*

$$\mathcal{S}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b}) \tag{2.256}$$

with $P_e \to 0$. When no receiver i can detect user u, including receiver u, then $\mathcal{S}_u = \emptyset$.

The cross-user sets for the users $i \neq u$ 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 decodable maxima ensures that all receivers that must decode a user u can do so reliably. This concept later returns in Equation (2.281). The sets $\mathscr{D}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$ and $\mathcal{S}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$ depend on $\boldsymbol{\Pi}$, the joint probability distribution $p_{\boldsymbol{xy}}$ (equivalently, the input distribution and the channel distribution), and $\boldsymbol{b}$. 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 $\mathcal{S}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$, and in particular that $|\boldsymbol{\Pi}| < \infty$. Random characterization, instead of deterministic order enumeration, produces the same regions but cannot guarantee they are the largest [12]. The probability-distribution search tacitly implies an AEP-like capacity-achieving randcom code design for the user with other users cancelled or treated as noise.

Mutual Information of Interest: All four mutual-information quantities, $\mathcal{I}(\boldsymbol{x};\boldsymbol{y})$, $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$, $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{x}_{\boldsymbol{u}\setminus u})$, and $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\mathscr{D}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$ (implies averaging of $\mathcal{S}_u(\boldsymbol{\Pi}, p_{\boldsymbol{x},\boldsymbol{y}}, \boldsymbol{b})$ within the same average as $\mathcal{I}$) may be of interest for a particular user u's analysis. The mutual information $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$ 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, $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$ does not account for the potential decoding of other users (components). So it is possible for b_u to exceed $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$, but never to exceed $\mathcal{I}(\boldsymbol{x};\boldsymbol{y})$, the maximum user-bit/symbol sum. $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{x}_{\boldsymbol{u}\setminus u})$ attempts to characterize a decoder's prior removal of other users, or specifically $\boldsymbol{u} = \mathscr{D}_u(\boldsymbol{\Pi}, p_{\boldsymbol{x},\boldsymbol{y}}, \boldsymbol{b})$. There are 2^{U-1} possible choices for the subset $\boldsymbol{u} \in \{\boldsymbol{U} \setminus u\}$ along with a check to see if those other users are decodable. The bound of **best average conditional bits/symbol** is (using Lemma 2.253)

$$b_u \leq \max_{\boldsymbol{u} \subseteq \mathscr{D}_u(\boldsymbol{\Pi}, p_{\boldsymbol{x},\boldsymbol{y}}, \boldsymbol{b}_{\setminus b_u})} \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{x}_{\boldsymbol{u}\setminus u}) = \mathcal{I}_{min,u}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) \tag{2.257}$$

where the maximization is over the possible subsets $\boldsymbol{u} \subseteq \{\mathscr{D}_u(\boldsymbol{\Pi}, p_{\boldsymbol{x},\boldsymbol{y}}) \setminus u\} \subseteq \boldsymbol{U}$ 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 (2.257) is key, again depending on Definition 2.6.3, and allows the general capacity region specifications to be explicitly described in Subsection 2.6.4.

More formally, the capacity rate region $\mathcal{C}(\boldsymbol{b})$ is a plot of all possible data-rate vectors (U-tuples) that receivers can reliably achieve. Some $\mathcal{C}(\boldsymbol{b})$ 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 multiuser 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) multiuser channel. This chapter's later sections simplify $\mathcal{C}(\boldsymbol{b})$ construction for various special cases of the MAC (Section 2.7, the BC (Section 2.8), and sometimes the IC (Section 2.9). 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* $\boldsymbol{b}_\alpha \in \mathcal{C}(\boldsymbol{b})$, $\boldsymbol{b}_\beta \in \mathcal{C}(\boldsymbol{b})$, *and* $\alpha + \beta = 1$, *then* $\alpha \cdot \boldsymbol{b}_\alpha + \beta \cdot \boldsymbol{b}_\beta \in \mathcal{C}(\boldsymbol{b})$.

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 components' codes), 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. The components corresponding to the $b_{A,u}$ and $b_{B,u}$ are user u's subusers. The lemma trivially expands to a convex combination of U users. Furthermore, then each user may need decomposition into up to U subusers, essentially proving that $U' < U^2$.**QED.**

A simple example helps illustrate the noise averaging or other-user-cancellation concept:

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

$$y = x_1 + x_2 + x_3 + n \quad . \tag{2.258}$$

This is a MAC with separate energy constraints $\mathcal{E}_1$, $\mathcal{E}_2$, and $\mathcal{E}_3$. For the decoding orders, the following table summaries the data rate for each user's decoding:

Order $\boldsymbol{\Pi}$	b_1	b_2	b_3
$[1\ 2\ 3]^*$	$\frac{\log_2\left(1+\frac{\mathcal{E}_1}{\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_2}{\mathcal{E}_1+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_3}{\mathcal{E}_1+\mathcal{E}_2+\sigma^2}\right)}{2}$
$[1\ 3\ 2]^*$	$\frac{\log_2\left(1+\frac{\mathcal{E}_1}{\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_2}{\mathcal{E}_1+\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_3}{\mathcal{E}_1+\sigma^2}\right)}{2}$
$[3\ 1\ 2]^*$	$\frac{\log_2\left(1+\frac{\mathcal{E}_1}{\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_2}{\mathcal{E}_1+\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_3}{\sigma^2}\right)}{2}$
$[2\ 3\ 1]^*$	$\frac{\log_2\left(1+\frac{\mathcal{E}_1}{\mathcal{E}_2+\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_2}{\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_3}{\mathcal{E}_2+\sigma^2}\right)}{2}$
$[3\ 1\ 2]^*$	$\frac{\log_2\left(1+\frac{\mathcal{E}_1}{\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_2}{\mathcal{E}_1+\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_3}{\sigma^2}\right)}{2}$
$[3\ 2\ 1]^*$	$\frac{\log_2\left(1+\frac{\mathcal{E}_1}{\mathcal{E}_2+\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_2}{\mathcal{E}_3+\sigma^2}\right)}{2}$	$\frac{\log_2\left(1+\frac{\mathcal{E}_3}{\mathcal{E}_2+\mathcal{E}_3+\sigma^2}\right)}{2}$

If energies are $\sigma_n^2 = .001$, $\mathcal{E}_1 = 3.072$, $\mathcal{E}_2 = 1.008$, and[65] $\mathcal{E}_3 = .015$ with Gaussian codes ($p_{\boldsymbol{x}}$ Gaussian for AEP random coding design), the order $[123]^*$ corresponds to $b_1 = 1$, $b_2 = 3$, and $b_3 = 2$.

Technically, the concept of "time-sharing" or more generally vertex-sharing of two different single-user message components requires that user's sub-division into two (or more) sub-users. These subusers' time-shared codes may be different, but their time-shared combination also has view as a single code with averaged rates.

[65] With only a sum-energy constraint, this channel trivializes to a single-vector channel with any rate combination satisfying $b_1 + b_2 + b_3 \leq \frac{1}{2} \cdot \log_2\left(1 + \mathcal{E}_{\boldsymbol{x}}/\sigma^2\right)$.

Gap-Margin Review: Section 2.4's single-user AWGN channel coding gap Γ (again, a function of code choice and target P_e) and margin γ_m concepts combine to reduce SNR in the bits/dimension calculation $\bar{b} = \frac{1}{2} \cdot \log_2 (1 + SNR/(\Gamma \cdot \gamma_m))$, with $SNR = 2^{2\bar{C}} - 1$. Somewhat trivially evident also is that there is a related **bit gap** γ_b:

$$\gamma_b \triangleq \mathcal{C} - b \ , \tag{2.259}$$

where $\gamma_b \geq 0$ for reliable transmission to be possible. When $\bar{b} \geq 1/2$, the gap $\Gamma(P_e)$ is constant (or can be made so), possibly differing only with chosen code class. For instance a code class of "uncoded" PAM and SQ QAM has $\Gamma = 8.8$ dB when $P_e = 10^{-6}$. Very good codes (LDPC for instance), used with good shaping, may have constant gap of close to 0.5 dB at this same P_e. The single-user bit gap and gap (including any margin) are readily and easily related as

$$\gamma_m = \frac{1}{\Gamma} \cdot \frac{2^{2 \cdot (\bar{b} + \bar{\gamma}_b) \cdot b} - 1}{2^{2 \cdot \bar{b}} - 1} \approx 6 \cdot \gamma_b \text{ dB} \ . \tag{2.260}$$

Basically SNR-reduction $\Gamma \cdot \gamma_m$ from its best use at channel capacity occurs when either or both the coding gap $\Gamma > 1$ for practical codes and/or the margin against unforeseen noise $\gamma_m > 1$. In effect the product $\Gamma \cdot \gamma_m$ determines the SNR loss relative to capacity's best use. Several margin-design examples appear in Section 2.4.

The multiuser gap definition requires a nontrivial expansion of the bit gap:

Definition 2.6.5 **[Multiuser Capacity-Border Rate Image]** *The capacity-border image* $\boldsymbol{c}_{\boldsymbol{b}'} \in \mathscr{C}(\boldsymbol{b})$ *of a rate* $\boldsymbol{b}'$ *that corresponds to a bit gap* γ_b *satisfies*

$$\boldsymbol{c}_{\boldsymbol{b}'} = \boldsymbol{b}' + \gamma_b \cdot \boldsymbol{1} \tag{2.261}$$

Figure 2.33 illustrates this image for two different $\boldsymbol{b}'$ choices in a two-dimensional rate region, shown as $\boldsymbol{b}_A$ and $\boldsymbol{b}_B$. In two dimensions, the point and its image define a line that Figure 2.33 shows in point A's blue or point B's green as

$$b_2 = b_1 + (b_{A,2} - b_{A,1}) \tag{2.262}$$

$$b_1 = b_2 + (b_{B,1} - b_{B,2}) \ . \tag{2.263}$$

The rate images $\boldsymbol{c}_A$ and/or $\boldsymbol{c}_B$ may require sophisticated calculation, which then allows determination of γ_b via the difference of any two same-user u elements of $\boldsymbol{c}_{\boldsymbol{b}'}$ and $\boldsymbol{b}'$. Chapter 5's various matlab programs enable iterative calculation $\boldsymbol{c}_{\boldsymbol{b}'}$.

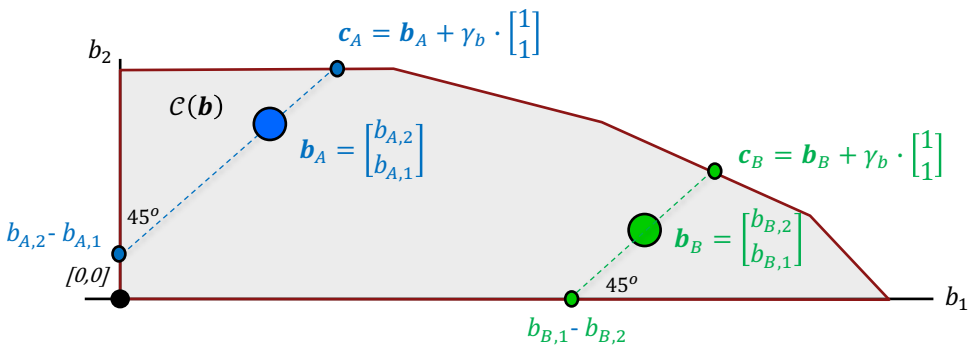


Figure 2.33: Illustration of multiuser fractional margin gap.

For $U \geq 2$, these equations generalize (with $\boldsymbol{b}'$ denoting the specific point and $umax \triangleq \arg\left(\max_u b'_u\right)$ to

$$b_{umax} = \left(\sum_{i \neq umax} b_i\right) + b'_{umax} - \left(\sum_{i \neq umax} b'_i\right) \quad . \tag{2.264}$$

The bit gap can apply to any channel (any $p_{\boldsymbol{y}/\boldsymbol{x}}$). Then there is a single line in U-dimensional space that extends from the design rate $\boldsymbol{b}'$ to the capacity region image $\boldsymbol{c}_{\boldsymbol{b}'} \in \mathscr{C}(\boldsymbol{b})$. The bit gap is the same on all dimensions incremented from that point to the image. The constant bit gap leads also to a scalar constant margin-gap when the channel has specifically AWGN. Then, the multiuser margin gap then follows Equation (2.260). Just as for single user, the designer may recall that the first jump from $\bar{b} = 1$ to $\bar{b} = 2$ is actually 7 dB, the next 6.3 dB, and then constant at 6 dB thereafter. If the channel is complex baseband, then the rule is instead 3 dB per bit/complex-subsymbol, but consistent in that a complex subsymbol has two real dimensions.

2.6.3 Optimum multiuser detection

The optimum multiuser detector generalizes Chapter 1's optimum single-user detector. The set of all possible coded multiuser channel inputs remains $C_{\boldsymbol{x}}$, and contains $M = |C_{\boldsymbol{x}}|$ possible distinct symbols, which may be a large number for the multiuser channel. $C_{\boldsymbol{x}}$ is thus a code for the set of all users that aggregates the individual users' codes. Section 2.3's AEP arguments most generally describe such a code with the probability density/distribution $p_{\boldsymbol{x}}$, from which designers may select good codes. For multiuser coding, by definition, the different users' codes are independent single-user-designed codes, each with $\Gamma = 0$ dB and independently designed by AEP random-coding principles for a specified b_u from the encoder-output distribution $p_{\boldsymbol{v}_u}$. The vector $\boldsymbol{v}$ will need a 1-to-1 transformation to $\boldsymbol{x}$, which may have dependency among its dimensions but is derived from the users' independently designed codes; for each user's message

$$\begin{aligned} m_1 &\rightarrow \boldsymbol{v}_1 \\ &\vdots \\ m_u &\rightarrow \boldsymbol{v}_u \\ &\vdots \\ m_U &\rightarrow \boldsymbol{v}_U \quad , \end{aligned}$$

when some or all users may have co-located transmitters.

> **Theorem 2.6.1** **[Optimum multiuser detection - maximum à posteriori]** *The multiuser-symbol-error probability is minimum when the detector selects $\hat{\boldsymbol{x}} \in C_{\boldsymbol{x}}$ to maximize $p_{\boldsymbol{x}/\boldsymbol{y}}$ and is the* **maximum à posteriori** *multiuser detector. When all possible multiuser input symbol values are equally likely, this optimum detector simplifies to maximization of the conditional probability $p_{\boldsymbol{y}/\boldsymbol{x}}$ over the choice for $\hat{\boldsymbol{x}} \in C_{\boldsymbol{x}}$, and is the* **maximum likelihood** *multiuser detector. This detector must respect multiuser-location restrictions on use of $\boldsymbol{y}_{i \neq u}$. Thus, the $\boldsymbol{y}$ in $p_{\boldsymbol{y}/\boldsymbol{x}}$ needs consistent understanding.*
>
> **Proof:** See Chapter 1's single-user proof. Nothing changes when considering any specific user's error probability, although coordination restrictions must be respected in that the detector may only use available components of $\boldsymbol{y}$. **QED.**

Even though the U users' codes are independent, the channel outputs may not (and usually are not) be independent. Then, for the BC and IC (and any multiuser channels with separate receiver locations), each physically distinct output $\boldsymbol{y}_u$ replaces the general $\boldsymbol{y}$ in the above Theorem 2.6.1, and each separate receiver may have different performance.

à posteriori and à priori densities with Detection Order: The individual user's à posteriori conditional probability distribution may condition on a user receiver's decodable other users. The conditional optimum detector then follows directly from the overall conditional distribution with known (at design time) other-user decodable set $\mathscr{D}_u(\mathbf{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$, which is here abbreviated by $\mathscr{D}_u(\mathbf{\Pi})$ with compliment $\overline{\mathscr{D}}_u(\mathbf{\Pi}) \triangleq \{\boldsymbol{U} \setminus \mathscr{D}_u(\mathbf{\Pi})\}$,

$$p_{\boldsymbol{x}_u/[\boldsymbol{x}_{i\in\mathscr{D}_u(\mathbf{\Pi})}\,\boldsymbol{y}]}(\boldsymbol{\chi}_u, \boldsymbol{x}_{i\in\mathscr{D}_u(\mathbf{\Pi})}, \boldsymbol{v}) = \int_{\boldsymbol{\chi}\in\boldsymbol{x}_{\{\overline{\mathscr{D}}_u(\mathbf{\Pi})\setminus u\}}} p_{\boldsymbol{x}/[\boldsymbol{y}\;\boldsymbol{x}_{\{i\in\mathscr{D}_u(\mathbf{\Pi})\}}]}(\boldsymbol{\chi}, \boldsymbol{x}_{i\in\mathscr{D}_u(\mathbf{\Pi})}, \boldsymbol{y}) \cdot d\boldsymbol{\chi} \quad . \tag{2.265}$$

This integral[66] averages $\boldsymbol{x}_{i\in\{\overline{\mathscr{D}}_u(\mathbf{\Pi})\setminus u\}}$, and for which variables no longer appear after the integration, effectively treating $\mathscr{D}_u(\mathbf{\Pi})$ users as "noise." The à posteriori conditional distribution inside (2.265)'s integral derives from the two known (channel) à priori distributions (using the independence of the users' messages) as

$$p_{\boldsymbol{x}/[\boldsymbol{y}\;\boldsymbol{x}_{i\in\{\mathscr{D}_u(\mathbf{\Pi})\}}]}(\boldsymbol{\chi}_u, \boldsymbol{\chi}_{i\in\mathscr{D}_u(\mathbf{\Pi})}, \boldsymbol{v}) = \frac{p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{\chi}_{i\in\overline{\mathscr{D}}_u(\mathbf{\Pi})}, \boldsymbol{x}_{i\in\mathscr{D}_u(\mathbf{\Pi})}, \boldsymbol{v}) \cdot p_{\boldsymbol{x}}(\boldsymbol{\chi}_{i\in\overline{\mathscr{D}}_u(\mathbf{\Pi})})}{p_{\boldsymbol{y}/\boldsymbol{x}_{\{i\in\mathscr{D}_u(\mathbf{\Pi})\}}}(\boldsymbol{x}_{i\in\mathscr{D}_u(\mathbf{\Pi})}, \boldsymbol{v})} \quad . \tag{2.266}$$

Equivalently, the conditional individual ML detector for $\boldsymbol{x}_u$ maximizes the integral of (2.266)'s left numerator term (assuming all $\boldsymbol{x}_u$ possible values are equally likely) or mathematically

$$\hat{\boldsymbol{x}}_u = \arg \max_{\boldsymbol{x}_u \in C_{\boldsymbol{x}_u}} \int_{\boldsymbol{\chi}\in\boldsymbol{x}_{\{\overline{\mathscr{D}}_u(\mathbf{\Pi})\setminus u\}}} p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{\chi}_{i\in\overline{\mathscr{D}}_u(\mathbf{\Pi})}, \boldsymbol{x}_{i\in\mathscr{D}_u(\mathbf{\Pi})}, \boldsymbol{v}) \cdot p_{\boldsymbol{x}}(\boldsymbol{\chi}_{i\in\overline{\mathscr{D}}_u(\mathbf{\Pi})}) \cdot d\boldsymbol{\chi} \quad . \tag{2.267}$$

Similarly the MAP maximizes (2.265). When the receivers do not coordinate, $\boldsymbol{y} \to \boldsymbol{y}_i$ in the above for receiver $i \in \{\boldsymbol{U}\}$. The expressions in Equations (2.265) and/or (2.266) then permit calculation of individual user error probabilities later in this subsection, for any order $\mathbf{\Pi}$ and of course also for a specific $p_{\boldsymbol{xy}}$. For subusers, a user's error probability corresponds to any of the subuser's incorrect decoding, which with equal-length symbols on all subuser's codes, is the sum of the error probabilities. However, individual subsymbol or bit-error probabilities may account for the fractional use of different codes in a weighted sum of the corresponding error probabilities.

Overall Error Probability: The overall error probability[67] for detecting all users is (as always)

$$P_e = 1 - P_c = 1 - \sum_{i=0}^{|C_{\boldsymbol{x}}|-1} P_{c/i} \cdot p_{\boldsymbol{x}}(i) \quad . \tag{2.268}$$

This single error probability has meaning for all multiuser 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 $\mathbf{\Pi}$:

$$\hat{\boldsymbol{x}}_u = \arg \left\{ \max_{\hat{\boldsymbol{x}}_u, \mathbf{\Pi}} p_{\boldsymbol{x}_u/[\boldsymbol{y}_u, \mathbf{\Pi}_u]} \right\} \quad , \tag{2.269}$$

which is (as always, even in single-user case) an implied function of $\boldsymbol{b}$. The designer pre-searches for best $\mathbf{\Pi}_{MAP}(u)$ that will maximize the average correct-detection probability

$$p_{c,u}(\mathbf{\Pi}_{MAP}) = \max_{\mathbf{\Pi}} \left\{ \sum_{i=0}^{|C_{\boldsymbol{x}}|} P_{c,u/i}(\mathbf{\Pi}) \cdot p_{u,i} \right\} \quad . \tag{2.270}$$

[66] Sum with discrete probability distributions.

[67] From Chapter 1, see Subsections 1.1.4, 1.1.6, and 1.3.2 specifically: $P_{c/i} = \sum_{\boldsymbol{v}\in\mathcal{D}_i} p_{\boldsymbol{y}/m_i}(\boldsymbol{v}, i)$ 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.

Then receiver implements a MAP (or ML when all long-length codewords are equally probable for these types of codes) by using $p_{\boldsymbol{x}_u/[\boldsymbol{y}_u,\boldsymbol{\Pi}_{MAP,u}]}$. The best order $\boldsymbol{\Pi}_{MAP,u}$ 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 $\mathcal{C}(\boldsymbol{b})$. A solution may use a weighted P_e so minimizes over the $\boldsymbol{\Pi}$ the sum

$$P_e = \sum_{u=1}^{U} w_u \cdot P_{e,u} \quad , \tag{2.271}$$

and uses this order for all receivers. Again, with known best single order, the remaining receiver designs follow directly for the applicable probability distribution. For the MAC, (2.271)'s weighting is not necessary.

For the BC and IC, the best detector for $\boldsymbol{y}_u$ also detects first all other users that are decodable (over any order), so effectively $\mathcal{D}_u = \mathbb{P}_u(\boldsymbol{\Pi}_{MAP}, p_{\boldsymbol{x}\boldsymbol{y}_u}, \boldsymbol{b})$. Thus, each receiver optimally detects each input that it can reliably, even though only user u may be of interest at receiver $\boldsymbol{y}_u$ in terms of message delivery. The error probability then becomes a function of u. When there are sub users, the symbol error probability is the sum of the sub-user symbol-error probabilities.

The individual decoders error probabilities on an AWGN channel's still follow the NNUB approximation as

$$P_e \leq N_e \cdot Q\left(\frac{d_{min}}{2\sigma}\right) \quad , \tag{2.272}$$

where the number of nearest neighbors, N_e, is the same, but the σ^2 now includes users treated as noise (later in the optimum applicable order).

The situation of independent channels greatly simplifies detectors:

Definition 2.6.6 [Crosstalk-Free Channel] *A* **crosstalk-free multiuser channel (CFC)** *has a conditional probability distribution that satisfies*

$$p_{\boldsymbol{y}/\boldsymbol{x}} = \prod_{u=1}^{U} p_{\boldsymbol{y}_u/\boldsymbol{x}_u} \quad . \tag{2.273}$$

That is, the channel probability distribution factors into independent terms for each of the users. When the channel is not CFC, it has **crosstalk**.

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 $p_{\boldsymbol{y}/\boldsymbol{x}}$, which factors into U independent terms that separate ML decoders independently maximize when (2.243) holds.

The capacity region for $\mathcal{C}(\boldsymbol{b})$ then trivially reduces to a rectangle for $U = 2$, or more generally to an orthotope.

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 upcoming Chapter 3's symbol-by-symbol detector, except now "user by user." The CFC assumption need not always occur, especially when design restrictions prohibit dimensionality increase (which typically requires larger bandwidth, longer delay, and/or more antennas).

The CFC channel's overall (all users) error probability is[68]

$$P_e = 1 - \prod_{u=1}^{U} P_{c,u} \quad , \tag{2.274}$$

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

$$P_e \geq P_{e,u} \; \forall \; u \in \{1, ..., U\} \quad . \tag{2.275}$$

When there are subusers, the error probability is $P_{e,u} = 1 - \prod_i (1 - P_{e,u,i})$ where i indexes the subusers for a particular user u.

2.6.3.1 MAC Detection Characterization for the linear multiuser AWGN

The linear multiuser AWGN is

$$\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n} \quad . \tag{2.276}$$

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

$$d_{min,u} = \min_{\boldsymbol{x}_u \neq \boldsymbol{x}'_u} \|H(\boldsymbol{x} - \boldsymbol{x}')\| \quad . \tag{2.277}$$

Trivially,

$$d_{min,u} \geq d_{min} \tag{2.278}$$

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

$$\min_u d_{\min,u} = d_{min} \quad . \tag{2.279}$$

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 $d_{min,u}$'s modification to $d_{min,u/\boldsymbol{u}\setminus u}$ if users in $\boldsymbol{u} \setminus u$ are given or removed, as per the following example:

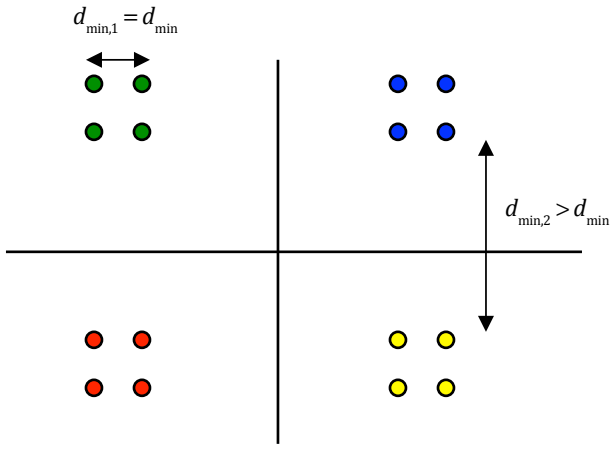


Figure 2.34: Illustration of two-dimensional two-user channel constellation.

[68] 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 MACusers both use 4-QAM with identical symbol periods, but different energy levels. Figure 2.34 shows the combined constellation that corresponds to these two signals' addition. The users' symbol contributions are not orthogonal, and both simultaneously occupy both dimensions. The first signal has the smaller $d_{min,1}$, which is equal to the overall d_{min}. The second signal has a larger $d_{min,2}$, and whose values correspond to the distance between Figure 2.34's 4 colors.

An ML detector for both signals simply selects the closest of the 16 points to a received two-dimensional value $\boldsymbol{y}$ 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 with noise well below either d_{min}. In this simple example, a detector that assumes user 1 is 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 $d_{min,2/1} > d_{min,2}$. but $d_{min,1/2} = d_{min,1}$.

2.6.4 A General multiuser Capacity Region

The general multiuser channel's capacity-region specification can require search over as many as $(U^2!)^U$ possible user orders $\boldsymbol{\Pi}$. Such a search may be very complex[69]. Some channels' searches need less than $U^2!$ order possibilities, as will be the case for the Matrix-AWGN MAC (Section 2.7), BC (Section 2.8), and sometimes the IC (Section 2.9).

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

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

For a general multiuser channel, $p_{\boldsymbol{y}/\boldsymbol{x}}$, user u's information-like quantity at receiver i with prior user set $\mathbb{P}_u(\boldsymbol{\pi}_i)$, is

$$\mathscr{I}_i\left(\boldsymbol{x}_u;\boldsymbol{y}_i/\mathbb{P}_u(\boldsymbol{\pi}_i)\right) \triangleq \begin{bmatrix} \infty & \pi_i(u) > \pi_i(i) \\ \mathcal{I}_i(\boldsymbol{x}_u;\boldsymbol{y}_i/\mathbb{P}_u(\boldsymbol{\pi}_i) & \pi_i(u) \le \pi_i(i) \end{bmatrix} \tag{2.280}$$

and again tacitly a function of $\boldsymbol{\Pi}$ and $p_{\boldsymbol{xy}}$ to avoid burdensome notation. This is the same information-like quantity as in (2.249) with reversed i and u.

Similarly, the minimum mutual-information arrives through the minimization process (through the same process, but with the index reversal, as in (2.251)) as

$$\mathcal{I}_{min,u}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) = \min_{i\in\{1,\ldots,U\}} \left\{\mathscr{I}_i\left(\boldsymbol{x}_u;\boldsymbol{y}_i/\mathbb{P}_u(\boldsymbol{\pi}_i)\right)\right\} \quad , \tag{2.281}$$

and thus as in (2.252):

$$\boldsymbol{\mathcal{I}}_{min}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) = \begin{bmatrix} \mathcal{I}_{min,U}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) \\ \vdots \\ \mathcal{I}_{min,u}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) \\ \vdots \\ \mathcal{I}_{min,1}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) \end{bmatrix} \quad . \tag{2.282}$$

Any $b_u \ge \mathcal{I}_{min,u}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}})$ cannot be reliably decoded, as needed for successive decoding, in this order, at one or more receivers, by the single-user capacity theorem. Thus again

$$\boldsymbol{b} \preceq \boldsymbol{\mathcal{I}}_{min}(\boldsymbol{\Pi}, p_{\boldsymbol{x};\boldsymbol{y}}) \quad . \tag{2.283}$$

All $\boldsymbol{\mathcal{I}}_{min}$ elements' (users') data rates can be reliably decoded at all corresponding receivers that decode its corresponding elements for the given order matrix $\boldsymbol{\Pi}$ and input distribution. For any

[69] 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.

receivers where a specific user is not decoded, it is treated as noise by that user's receiver. For Sections 2.7 and 2.8 on Gaussian matrix MAC and BC respectively, (2.281)'s minimization step is not necessary. For situations with more than one subuser/user, each of the subusers' decoding replaces a user decoding in this step.

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 $p_{\boldsymbol{x}}$ possibilities, and so there are correspondingly many points $\boldsymbol{b}(\boldsymbol{\Pi}, p_{\boldsymbol{x}})$. An achievable rate-region construction permits vertex-sharing or corresponding $\mathcal{I}_{min}(\boldsymbol{\Pi}, p_{\boldsymbol{x}})$ values for the possible order matrices $\{\boldsymbol{\Pi}\}$; equivalently the convex hull of the region formed by the set of points over all orders for any given allowed $p_{\boldsymbol{x}}$ is achievable as the **distribution-dependent achievable rate region**:

$$\mathcal{A}(\boldsymbol{b}, p_{\boldsymbol{x}}) = \bigcup_{\boldsymbol{\Pi}}^{conv} \boldsymbol{\mathcal{I}}_{min}(\boldsymbol{\Pi}, p_{\boldsymbol{x}}) \quad . \tag{2.284}$$

This convex-hull operation can require Section 2.6.2's subuser components, enlarging search by as much as $U' \leq U^2$, so $U'!$ orders at each receiver. These possiblities exhaust all achievable decompositions[70]. The convex hull of a finite set of points, in this case the up to $|\boldsymbol{\Pi}| = (U'!)^U$-term convex combinations $\sum_{i=1}^{|\boldsymbol{\Pi}|} \alpha_i \cdot \boldsymbol{\mathcal{I}}_{min}(i)$ where $\alpha_i \geq 0$ and $\sum_{i=1}^{|\boldsymbol{\Pi}|} \alpha_i = 1$. Any point outside this convex hull must use at least one term in such a convex combination that has an $\boldsymbol{\mathcal{I}}_{min}(i) + \boldsymbol{e}$ with $\boldsymbol{e} \succeq \boldsymbol{0}$ with at least one nonzero component. Reliable decoding of this point, for the order i and the given $p_{\boldsymbol{x}}$ and channel $p_{\boldsymbol{y}/\boldsymbol{x}}$, mandates use at least one receiver that cannot reliably 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 $p_{\boldsymbol{x}}$. Thus, points outside the achievable region have P_e for that user bounded away from zero. This order-convex-hull step may remove from consideration many superfluous orders in (2.281)'s search.

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

$$\mathcal{C}_{general}(\boldsymbol{b}) = \bigcup_{\text{allowed } \{p_{\boldsymbol{x}}\}}^{conv} \mathcal{A}(\boldsymbol{b}, p_{\boldsymbol{x}}) \tag{2.285}$$

where the convex hull over all possible input probability distributions allows independent-user input distributions that each must satisfy all of particular user's input constraints.

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

> **Theorem 2.6.2** [The multiuser Capacity Region Theorem] *The multiuser capacity region for any multiuser channel with $p_{\boldsymbol{y}/\boldsymbol{x}}$ is in Equation (2.285).* **Proof:** *See the preceding 3 steps.* **QED.**

Comment on auxiliary variables and common/private users: El Gamal[71] summarizes well the many previous approaches to capacity region development, including descriptions of Han-Kobayashi and Sato bounds, [5]. These approaches often introduce the concept of **private** and **common user** components. Essentially a private-user component reliably decodes at only its corresponding receiver. A common user component reliably decodes at more than one receiver. Each user partitions into private and common sub-user components, which then become conditional auxiliary variables in these developments. These methods lead to achievable regions that are subsets of the above region (sometimes attaining

[70] Effectively this is a finite (but large) enumeration alternative to approaches [12] using auxilliary variables that instead have uncountably infinite continuous-variables replacing this search, and consequently cannot in their cases unequivocally state it is optimum.

[71] Stanford Professor Abbas El Gamal (05/30/1950), an Egyptian-American educator and engineer who pioneered multiuser information-theory.

it, depending on situation), but not sure it is the capacity region. This text's approach then begs the question "Do the present text's various convex-hull operations cover all possible private/common auxiliary possibilities?" Clearly when all subuser components are considered, this region is achievable and any point outside it is not. The simple answer is "yes" following the 3-step proof above that has a converse for any point outside the region. In some ways this simplifies (although obviously more complex/complete searching occurs in this text's approach) but adds the burden of identifying all subusers. For most channels, like the matrix AWGN channels in subsequent sections, this identification is possible.

However, the following description of this text's doubly convex-hull approach (one hull for order, the outer hull for input distribution) offers an alternative view that may help connect the approaches:

1. **Chain Rule:** Every mutual information decomposes into $U'!$ possible chain-rule decompositions and subsequent reliably decoable user rates for the corresponding $p_{\boldsymbol{x}}$ for which the mutual information $\mathcal{I}(\boldsymbol{x};\boldsymbol{y})$ was computed.

2. **Minimum Mutual Information Vector:** Every mutual-information vector (vertex) $\boldsymbol{\mathcal{I}}_{min}(\boldsymbol{\Pi}, p_{\boldsymbol{x}})$, for the given $p_{\boldsymbol{x}}$ contains private components (either they are reliably decoded at only their receiver) or common components that are reliably decoded at more than one receiver, and components that may be decoded at various subsets of receivers.

3. **User Components:** User components corresponding to common/private user component decompositions thus can form through a convex combination of $\boldsymbol{\mathcal{I}}_{min}$ vertices. The range of possibilities are all such combinations, namely the convex hull – for the given $p_{\boldsymbol{x}}$ input distribution. Any point outside this convex hull, for the given $p_{\boldsymbol{x}}$ violates at least one single-user capacity theorem (for reliable decoding) in the chain-rule decomposition. Possible "private/common" decompositions are completely considered for this input distribution, along with other subuser compositions that private/common approaches do not consider.

4. **Input Distribution Range:** Repeating the process for every possible input distribution (possibly and usually) enlarges the region by synthesizing all the possible subuser decompositions for each $p_{\boldsymbol{x}}$ and then all possible combinations of different-$p_{\boldsymbol{x}}$ combinations. All these points can be realized, as per the chain-rule decomposition (and the time-sharing/dimension-sharing implied by the convex hull operations). Any point outside the region will not be reliably decodable for at least one user.

Basically, this text's multi-step process covers all the possible auxiliary variable bounding approaches (which variable-bounding approaches effectively attempt to combine into one convex combination and invevitibly miss some potential combinations.) It also leads in Chapter 5 to convex-design programs that produce transmitters and receivers that reliably decode for any $\boldsymbol{b} \in \mathcal{C}(\boldsymbol{b})$.

Best Rate Sum: The maximum sum b_{max} occurs when the plane $\sum_{u=1}^{U} b_u = b_{max}$ is tangent to $\mathcal{C}(\boldsymbol{b})$'s boundary, which necessarily must occur for at least one $\boldsymbol{b}$ 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 search quantizes the range of parameters. A $N_p \times 1$ vector of such parameters is $\boldsymbol{r}_{\boldsymbol{x}}$, each with range parameter $r_{x,n}$ having

$$r_{x,min,n} \leq r_{x,n} \leq r_{x,max,n} \tag{2.286}$$

A reasonable quantization step choice $\Delta_{x,n}$ partitions the range to cover each parameter's extent. An outer loop for (2.285)'s evaulation then becomes

$$\bigcup_{\boldsymbol{b}} \left\{ \sum_{r_{x,1}=r_{x,min,1}:\Delta_{x,1}}^{r_{x,max,1}} \sum_{r_{x,2}=r_{x,min,2}:\Delta_{x,2}}^{r_{x,max,2}} \cdots \sum_{r_{x,N_p}=r_{x,min,N_p}:\Delta_{x,N_p}}^{r_{x,max,N_p}} \mathcal{A}(\boldsymbol{b}, p_{\boldsymbol{x}}(\boldsymbol{r}_{\boldsymbol{x}})) \right\} . \tag{2.287}$$

While (2.287)'s search may be tedious, it is otherwise a straightforward "outer loop" to achievable region's internal calculation for each given $p_{\boldsymbol{x}}$ that corresponds to indices in (2.287)'s sum cascade. 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 $R_{\boldsymbol{xx}}$ (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 simplify to consider only energy-parameter ranges. When each user's input is a scalar, this simply corresponds to $k \cdot \Delta\mathcal{E}_u$, $u = 1, ..., U, k = 0, ..., \mathcal{E}_u/\Delta\mathcal{E}_u$. When the input has $L_x > 1$, the search becomes a search over the sub energies $\sum_{\ell}^{L_{x,u}} \mathcal{E}_{u,\ell} \leq \mathcal{E}_u$ for each u (again in the BC case, this is a single search over the dimensions for the BC's single input). The $R_{\boldsymbol{xx}}(u)$ in these MIMO cases forms as

$$M_u \cdot \text{Diag}\{\boldsymbol{\mathcal{E}}\} \cdot M_u^* \tag{2.288}$$

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} $i = 1, ..., U$ 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 parallels the IC search.

2.6.4.1 Admissibility

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

This leads to Chapter 5's Gaussian-channel alternative designs that attempt find for any $\boldsymbol{b}'$, 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 (without finding them all) then confirms the $\boldsymbol{b}$ is admissible (for reliable decoding).

2.6.5 The Matrix AWGN Gaussian Channel

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

$$\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n} \ . \tag{2.289}$$

Lemma 2.6.4 [Gaussian Inputs' Optimality on multiuser AWGN Channels] *For any linear AWGN $\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n}$ where $\boldsymbol{n}$ is Gaussian with autocorrelation $R_{\boldsymbol{nn}}$ and given input autocorrelation $R_{\boldsymbol{xx}}$, the optimum input distribution for each and every user $\boldsymbol{x}_{u \in \boldsymbol{U}}$ is Gaussian, and thus achievement of $\mathscr{C}_{awgn}(\boldsymbol{b})$ 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 $\mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$ and corresponding $R_{\boldsymbol{xx}}$. Lemma 2.3.4's Chain Rule applies to any order $\pi(u)$ of users. The last user U's input/output relationship in such an order corresponds to a single-user AWGN channel for any given specific values of $\boldsymbol{\chi}_{U-1} = \boldsymbol{x}_{U-1}, ... \boldsymbol{\chi}_1 = \boldsymbol{x}_1$, so $p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{\chi}_U, [\boldsymbol{x}_{U-1}, ..., \boldsymbol{x}_1], \boldsymbol{v})$. This single-user ($U$) channel has maximum value over $p_{\boldsymbol{x}_U}$ distributions that is Gaussian. Since $\boldsymbol{x}_U$ is then Gaussian, its crosstalk is Gaussian and then an induction to users $U - 1 ... 1$ applies in succession so that they all are Gaussian. This holds for any $R_{\boldsymbol{xx}}$, including a given specified autocorrelation. This also holds for any and all orders.

Equation (2.281) is a mutual information and thus has maximum when $p_{\boldsymbol{x}}$ is jointly Gaussian over all users (and infinite values are trivially not of interest). At this point with all Gaussian distributions, Equation (2.281)'s minimization is over the possible receiver points for a mutual

information but does not change the fact that $\mathcal{I}_{min,u}(\mathbf{\Pi}, p_{\boldsymbol{xy}})$ remains a mutual information, and thus is a highest data rate for any $R_{\boldsymbol{xx}}$ that satisfies the other decodabillty restrictions this section already addressed. Clearly Equation (2.283)'s stacking of user bits into a bit vector is just notational and does not compromise Gaussian optimality. For each order and jointly Gaussian (now optimal) input, a single-user capacity-achieving Gaussian code can be used with previously decoded users' crosstalk given and thus removed and subsequent-to-be-decoded users treated as Gaussian noise. This describes a code that achieves the point as the jointly Gaussian combination of the individual single-user (Gaussian) codes that achieve each the corresponding single-user capacity for the given Gaussian noise-crosstalk combination, so the point can be realized for the given inputs' Gaussian autocorrelation matrix $R_{\boldsymbol{xx}}$.

Equation (2.284)'s convex-hull union operation, over all orders, introduces dimension sharing. The dimensional sharing can be viewed as decomposing the vectors $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ each into two vectors, doubling dimensionality, so that

$$\boldsymbol{\beta}_1 \rightarrow \begin{bmatrix} \boldsymbol{\beta}_{1,\theta} \\ \boldsymbol{\beta}_{1,1-\theta} \end{bmatrix} \tag{2.290}$$

$$\boldsymbol{\beta}_2 \rightarrow \begin{bmatrix} \boldsymbol{\beta}_{2,\theta} \\ \boldsymbol{\beta}_{2,1-\theta} \end{bmatrix} \quad . \tag{2.291}$$

Lemma 2.3.3 however, with expansion of $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$, as in Equations (2.290) and (2.291), proved that for any θ and the consequent $R_{\boldsymbol{xx}}$, the entropy maximizing distributions remain Gaussian, and thus so does the consequent mutual information remain maximum. Thus, Gaussian $p_{\boldsymbol{x}}$ continues to maximize the boundary/extreme points in (2.284)'s $A(\boldsymbol{b}, p_{\boldsymbol{x}})$. The capacity region $\mathcal{C}(\boldsymbol{b})$ in (2.285)) then maximizes over all possible (allowed by $\mathbf{\Pi}$ earlier and as expanded by dimension sharing) auto-correlation matrices $R_{\boldsymbol{xx}}$ such that any applicable energy constraints are also satisfied, but all remain Gaussian in that optimization of auto-correlation possibilities. Again, all these can be realized by the single-user Gaussian codes that combine for any particular order (or convex combination thereof in dimensional-sharing expansion), so there is a realization.

Further, any $\boldsymbol{b}$ even slightly outside this region would have at least one user (or one user in some dimensions of dimension-sharing) that would violate the single-user capacity constraint and have error probability finite and positive, meaning at least one user cannot be reliably decoded and thus that $\boldsymbol{b}$ is not reliably decodable in the sense that all users must be reliably decodable by definition of the capacity rate region. Thus, the joint Gaussian-code distribution is optimum for any linear AWGN multiuser channel. **QED.**

The general search is complex with much bookkeeping on $\mathbf{\Pi}$, allowed $p_{\boldsymbol{xy}}$, 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 definition 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, but specific to the matrix AWGN.

Definition 2.6.7 **[(Subsymbol) Degraded multiuser Gaussian Channel]** *A* **(subsymbol)-degraded AWGN multiuser channel** *has matrix ranks for H and/or $R_{\boldsymbol{xx}}$ that are ϱ_H and $\varrho_{R_{\boldsymbol{xx}}}$ respectively, such that*

$$\min\left\{\varrho_{R_{\boldsymbol{xx}}}, \varrho_H\right\} < U \quad . \tag{2.292}$$

Otherwise, the channel is **non-degraded**. *The literature often omits the word "subsymbol," but it is tacit in degraded-channel definitions.*

This degraded multiuser channel basically forces users to share subsymbol dimensions, or equivalently the system does not have sufficient dimensionality to multiplex orthogonally 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.

2.6.5.1 The Capacity-Energy Region

The matrix AWGN's capacity energy region $\mathcal{C}_{\boldsymbol{b}}(\boldsymbol{\mathcal{E}})$ contains all user energy vectors $\boldsymbol{\mathcal{E}}$ for which a MAP receiver and good code reliably detect ($P_e \to 0$) specific rate vector $\boldsymbol{b}$. This convex region can be useful in design for a specific data rate because it guides on the user energy trade-offs to achieve a particular point. If an actual energy constraint is such that $\boldsymbol{\mathcal{E}}_x \in \mathcal{C}_{\boldsymbol{b}}(\boldsymbol{\mathcal{E}})$, then the rate $\boldsymbol{b} = \boldsymbol{b}'$ is admissible, or equivalently $\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})$, where a more descriptive name might be $\mathcal{C}_{\boldsymbol{\mathcal{E}}}(\boldsymbol{b})$. Figure 2.35 illustrates the basic concept where there are no energy limits, but the user's bits/symbol (b_u) instead have fixed values. This region is also trivially convex via a time-sharing argument. The energy-capacity region simply re-describes the capacity regions that would correspond to different energy limits. Trivially thus the rate $\boldsymbol{b}$ is reliably achievable if it corresponds to a point in the original rate region that corresponds to a particular given energy vector $\boldsymbol{\mathcal{E}}_x$.

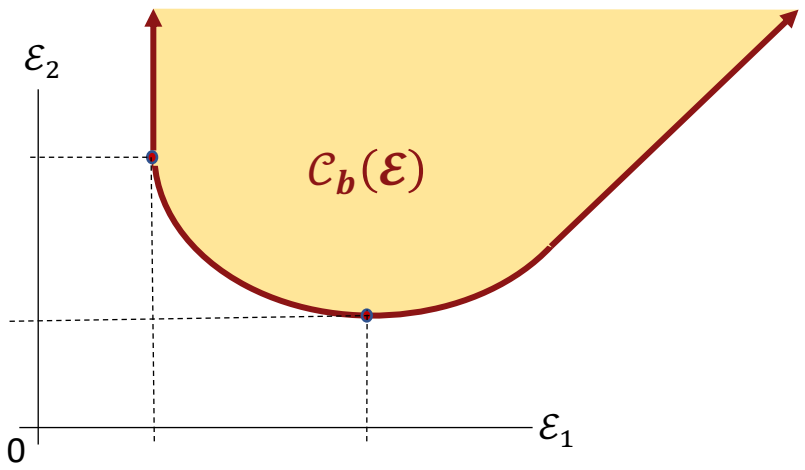


Figure 2.35: Capacity-Energy Region $\mathcal{C}_{\boldsymbol{b}}(\boldsymbol{\mathcal{E}})$.

2.6.6 Ergodic Capacity Region

Following Subsection 2.3.7's single-user ergodic capacity, the **ergodic capacity region** has definition:

Definition 2.6.8 *[Ergodic Capacity Region] for ergodic-channel parameter $\widetilde{H}$ (independent of all user inputs) is*

$$< \mathcal{C}(\boldsymbol{b}) >= \mathbb{E}_{\widetilde{H}} \left[\mathcal{C}(\boldsymbol{b}) \right] \quad . \tag{2.293}$$

Again the expectation is over a parameter distribution $\widetilde{H}$ that is independent of the channel input. Ergodic capacity suggests that long codewords or blocklengths (longer that coherence periods of $\widetilde{H}$), which implies situations of rapid channel variation be addressed by longer-delay codes[72]. These codes' design for each user is for the average (ergodic) capacity of that user and the ergodic capacity extension ensures that a good code will still be reliably decodable if its rate is less than the average in the ergodic capacity region. Such average design however may have unacceptable delay. However, if instead the channel is **quasi-static** in that designs for a fixed channel can occur over a coherence period, then delay may be less. This will introduce a variation in the channel capacity region and the individual user's data rates $b_u \to b_u(t)$, and thus perhaps a channel-input-acceptance (an arrival rate) when the data can be accepted for transmission. Section 2.6.7 addresses this type of variation, as well as variation in the arrival times of message packets in **scheduling**.

The designer then may have two objectives:

1. design for the ergodic rate with long codewords and accept the consequent decoding complexity and delay, but the need to know the channel well is less - thus design for $< \mathcal{C}(\boldsymbol{b}) >$, or

2. design more completely for shorter intervals less than the coherence time with less delay and decoding complexity, but also a need to know the instantaneous channel and corresponding current capacity region, $\mathcal{C}(\boldsymbol{b})$, noting that of course the best codes theoretically require $\bar{N} \to \infty$ for $P_e \to 0$ – nonetheless, very good codes close to $\Gamma = 0$ dB can have finite block lengths that would presumably be less than the coherence time for this approach to be viable.

2.6.7 Scheduling and Multiuser Rate-Vector Selection

Tacit so far has been the specific multiuser data-rate selection, $\boldsymbol{b} = \boldsymbol{b}'$. The capacity region $\mathcal{C}(\boldsymbol{b})$ identifies trade-off between different user rates. The design and/or system can exploit the capacity region's offered tradeoffs, but must allocate transmit information in terms of what is admissible, $\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})$. The capacity region $C_{\widetilde{H}}(\boldsymbol{b})$ may vary with time/$\widetilde{H}$ as in Subsection 2.6.6, which includes noise-variance variation. Actual implementation often considers a specific sample from a measured channel-characterizing distribution (channel identification, see Chapter 9) for a symbol (or a series of symbols) for which the channel is relatively constant or coherent (as in Section 1.6). The user inputs' arrivals may also vary with time. So far, and largely in this text, inputs have constant bits/symbol $\boldsymbol{b}$. However, communication users may not transmit continuously and thus have silent periods of no data between "arrivals" of data packets for transmission, as in Figure 2.36. This creates the appearance of a time-varying, or more precisely random stationary-ergodic, input data-rate vector, often called $\boldsymbol{\lambda}_q$ in network communication. Clearly $\mathbb{E}[\boldsymbol{\lambda}_q] \preceq \mathbb{E}[\boldsymbol{b}]$. However, instantaneous input rates (samples of $\boldsymbol{\lambda}_q$) may exceed the available data rates, in which case Figure 2.37 illustrates that the delay in transmission caused by input data queuing.

[72] Clearly channels that have large coherence times can be treated as if deterministic in terms of design.

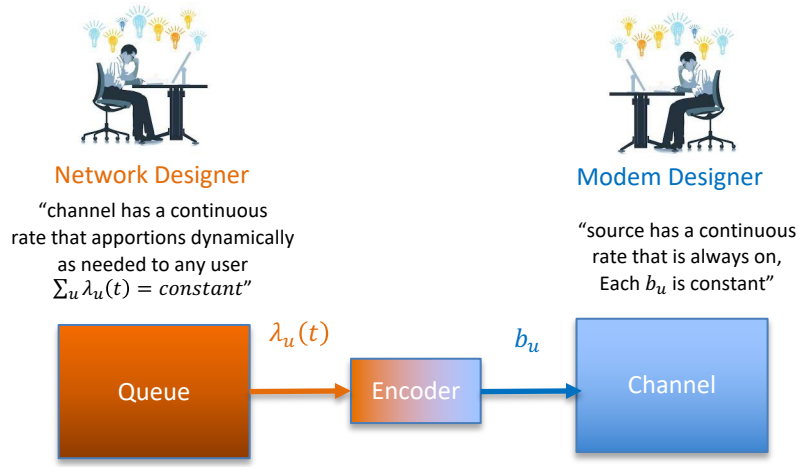


Figure 2.36: **Modem versus network viewpoints.**

Figure 2.36 illustrates the different viewpoints that communication engineers may have. This text's readers immediately recognize the modem designer who views the supplied message streams as constant b_u for each user, and then designs a system (the modem) to transport those constant rates. The channel may vary, so the maximum data rates (or $\boldsymbol{b} \in \mathcal{C}(\boldsymbol{b})$) may also vary, but the source appears as fixed. If the data rate is too large ($\boldsymbol{b} \notin \mathcal{C}(\boldsymbol{b})$, it cannot be transmitted reliably. The modem designer thereby oversimplifies the challenge. Actual data sources usually are not so constant. The network designer instead views the individual users as randomly time varying with arrival rates (bits/symbol) $\lambda_u(t)$. When the user has no data, $\lambda_u(t) = 0$. When the user has too much data for reliable channel transmission, the excess data is delayed in a queue for transmission at future possible times when the channel rate supports the excess. The network designer views the channel as one big constant rate $\sum_u \lambda_u(t) = b$ that is to be partitioned to users using queues. Where the modem designer assumes continuous data rates, the network designer assumes the channel can be arbitrarily partitioned to support user-queue needs. Neither designer is always correct (and indeed they are both almost always wrong). The network designer's presumption that partitioning of user data rates to accommodate the queue does not usually consider that the channel may not support what is best for the queue or users, while also the modem designer's presumption that all the users are continuously arriving at constant rates to support essentially does not exploit fully the capacity region of tradeoffs.

The solution essentially involves Appendix A's queueing theory that allows for variable channel (server) rates. This theory applies to **scheduling** in present context. Broader scheduling theory often partitions a high-rate channel of constant rate, often referred to as **time-division multiple access (TDMA)**. TDMA often exhibits Figure 2.36's network-designer fallacy that the channel can be so arbitrarily partitioned. The fallacy arose in wired channels (like a coaxial cable or fiber connection) that carry many users messages and are viewed as a single-user rate sum. Such systems are point-to-point and design/resource-allocation simply allocates users to dimensions/time-slots. In this viewpoint, such systems have trivial capacity regions (for instance a time-sharing triangle for two users), see Prob 2.24. This limited and oversimplified capacity region is a regular (U-dimensional) pyramid

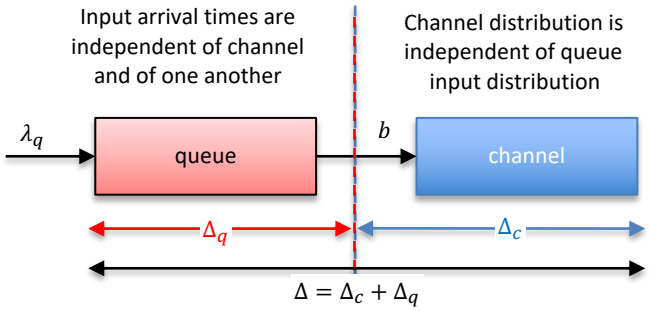
$$\mathcal{C}(\boldsymbol{b}) = \{\boldsymbol{b} \,|\mathbf{1}^* \boldsymbol{b}^*(t) = b \;\wedge\; \boldsymbol{b}(t) \succeq \mathbf{0} \;\forall\, t = n \cdot T \text{ where } n \in \mathbb{Z}\} \quad ,$$

or in words the sum-rate is maximally a constant and user data rates can be allocated to users arbitrarily as long as the maximum sum b is not exceeded.

Wireless systems, and particularly so those with many antennas and users in different locations, simply do not follow often this pyramid-region model. The oversimplified region often corresponds to TDMA partitioning as a function of user-source statistics. Elaborate treatment is left to other texts

(Appendix A has a summary) because it is not channel focused.[73] Essentially the TDM system typically has an average data rate that is b/U, but of course certainly users can have larger averages while others are smaller, but the total b cannot be exceeded. TDM in the form of allocating each user dedicated bandwidth incurs a major loss[74]. When users are not active at the same time, dynamic bandwidth allocation can have advantage (even if the channel is stationary).

Figure 2.37 introduces a more sophisticated scheduling where the queue and channel attempt to work together to best support user need while also considering the channel's tradeoff between users, $\mathcal{C}(\boldsymbol{b})$.



$\mathcal{B} = \lambda_q \cdot \Delta \longrightarrow \mathbb{E}[\mathcal{B}] = \mathbb{E}[\lambda_q] \cdot \mathbb{E}[\Delta]$ = number of bits in system (Little's Theorem)

$\mathbb{E}[\lambda_q] \leq \mathbb{E}[b]$ for stable operation

Figure 2.37: Input queuing for random data rates.

The input λ_q is in bits/symbol; just like b, both can be random and ergodic. Thus user subscript of u is tacit, but not shown. The two processes: input packet arrivals and channel symbol processing are independent. However, the choice of a specific $\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})$ can of course be made to depend on the input and queue. Appendix A addresses random variables and processes and also their use in queuing, which is often modeled by state-machines or Markov Processes. There is a delay (in symbol periods) through each of the queues and the channel, Δ_q and Δ_c respectively, and the total number of bits in the system (queue and/or channel) is $\mathcal{B}$. **Little's Theorem** basically notes that since the input arrival process (effectively characterized in the distribution of λ_q) is independent of the channel (random or deterministic) that follows, so since $\mathcal{B} = \lambda_q \cdot \Delta$, then

$$\mathbb{E}[\mathcal{B}] = \mathbb{E}[\lambda_q] \cdot \mathbb{E}[\Delta] \tag{2.294}$$

$$\mathbb{E}[\boldsymbol{\mathcal{B}}] = \mathbb{E}[\boldsymbol{\lambda}_q] \odot \mathbb{E}[\boldsymbol{\Delta}] \quad \text{vector multiuser form} \ . \tag{2.295}$$

Thus, notational simplification here in this multiuser context simply replaces each quantity's variable by its mean. Intuitively, stability requires

$$\lambda_q \leq b \ \ (\textbf{stable queue}). \tag{2.296}$$

The instantaneous values of λ_q and thus $\Delta \geq 0$ can however vary, and indeed in extreme cases the delay could become arbitrarily large (presumably exceeding any practical implementation's memory limit) but with vanishing probability. A more formal definition of stability is that $\lim_{\tau \to \infty} Pr\{\Delta > k\} \to 0$ for any $k < \infty$, with τ being time. Equivalently, this means that at some point in time, the queue will become empty so $\Delta_q \to 0$ at various points in time. Basically, on average the channel goes faster than the queue

[73] Other than to assume the channel supports some fixed data rate that may vary with time as the sum of all users' data rate maximally possible.

[74] For instance a cable-modem system advertising 1 Gbps home connection speed would need, if there were 1000 users (typical) sharing the medium, to instead advertise 1 Mbps.

filling, so eventually the queue will zero and then start filling again. The average delay will be finite for input distributions of interest. The same analysis can be applied to every user so for multiuser system stability

$$\boldsymbol{\lambda}_q \preceq \boldsymbol{b} \quad , \tag{2.297}$$

which means every user is stable in terms of its average arrival rate does not exceed its corresponding average channel rate. The average delay and moments (particularly second moment, or its square-root **jitter**) follow for various input and channel statistics. Various queuing analyses apply for different arrival distributions (first letter), channel inter-arrival processing-time distribution (second letter), and number of users U, so for instance $M/M/1$ corresponds to a Poisson distribution on the the number of arrivals in a period of time τ_q (so λ_q parametrizes the mean interarrival time distribution), with channel-input arrival-time separations exponential distributed as $1/b \cdot e^{-b \cdot \tau_c}$, and $U = 1$ single user. Appendix A summarizes some of the more basic results from queuing theory for different queue and channel distributions and numbers of users/servers. These statistics help compute average extra delay beyond the channel-coding delay (which of course approaches infinity with zero-gap codes anyway).

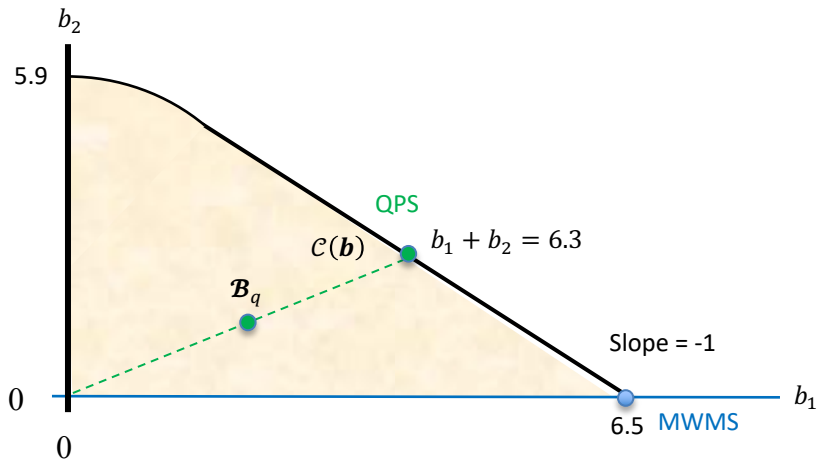


Figure 2.38: Comparison of MWMS and QPS Scheduling Strategy.

For any parameter-sample channel (function or series) from the channel distribution, there is a corresponding capacity region. It is convenient for ergodic processes to think of such a sample channel as an $\widetilde{H}$ at some point in time, so then the capacity region is a function of this $\widetilde{H}$, which usually has many reliably implementable designs corresponding to different points $\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})$. For a corresponding sample of the input queue, there is a specific queue depth vector $\boldsymbol{\mathcal{B}}$. Clearly, one objective is to reduce the queue depth for users who already have large depths at some time point, but also the channel itself may favor some users with higher data rates in $\mathcal{C}(\boldsymbol{b})$. Thus, while there exists (yet) no optimal queuing strategy, heuristics suggest that maximization of the inner product

$$\max_{\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})} \sum_u b_u \cdot \mathcal{B}_u \quad , \tag{2.298}$$

where clearly this point will occur on the capacity region boundary $\mathscr{C}(\boldsymbol{b})$ for a given $\boldsymbol{\mathcal{B}} \succeq \mathbf{0}$. This heuristic strategy was introduced by McKeown[75]) et al [8] in 1996, but is stable and known as **Maximum Weight Matching Scheduling (MWMS)**. A better **queue-proportional scheduling (QPS)** strategy emerged 10 years later from Seong et al [9] and is also stable:

[75]Stanford Prof N. McKeown, (04/07/1963 - A English-American pioneer in computer networking and software-defined networks.

Definition 2.6.9 (Queue Proportional Scheduling (QPS)) *satisfies*

$$\max_{\gamma_{mu,b}} \quad \boldsymbol{b}' = (1+\gamma_{mu,b}) \cdot \boldsymbol{\mathcal{B}} \tag{2.299}$$

$$ST \quad \boldsymbol{b}' \in \mathscr{C}(\boldsymbol{b}) \ . \tag{2.300}$$

Basically QPS equalizes the delay for the users as a form of fairness and also empties the queue faster than MWMS. Either solution can prioritize different users by creating a priority vector $\boldsymbol{\theta} \succeq \mathbf{0}$ such that $\boldsymbol{b}' \rightarrow \boldsymbol{b}' \odot \boldsymbol{\theta}$. Figure 2.38 illustrates the two methods. Basically, QPS maximizes the slowest user's gap relative to choice of $\boldsymbol{\mathcal{B}}_q$ as the current rate. As the name implies, QPS chooses the operational point on the capacity region boundary that is collinear with the input queue depth vector. This sets all users delays equal so $\Delta_{q,u} = \Delta_q \ \forall \ u = 1, ..., U$. Indeed this is fair (or if priority is given within $\boldsymbol{\theta}$, fair relative to said priority's emphasis) and intuitively (and [9] shows mathematically) empties the queues fastest. The MWMS illustration in Figure 2.38 pictorially suggests that this method will also have greater jitter in the queue as the algorithm searches for more extreme boundary points that maximize the inner product of the queue depth vector and the selected channel-rate vector. For the MAC, QPS and MWMS coincide.

Thus, QPS provides a way (with equal priority $\boldsymbol{\theta} = \mathbf{1}$ or otherwise) to specify a $\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})$ for transmission. Design can proceed directly. If the input rate vector $\boldsymbol{\lambda}$ is constant, then the design picks the point corresponding to the same line always; and indeed if the channel is constant, it is the point on the line that is in the fixed capacity-region boundary.

Control: The multiuser 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 multiuser case also implies a common design assumption, but goes further to imply, even in the IC or more general multiuser configurations, that there is some common agreement on the codes, data rates ($\boldsymbol{b}$ 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 multiuser channels, design, assignments of codes/dimensions may need coordination. There are basically four approaches to this coordination.

Central Management These systems have a policy controller that informs each transmitter what dimensions and codes to use (and when - See Section 2.6.7.1)..

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. (See Section 2.6.7.2).

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. (See Section 2.6.7.3).

Random Access Random access systems make design assumptions based on dimensional availability probabilities and on collision detection and retransmission (See Section 2.6.7.4).

2.6.7.1 Central Management or Network Slicing

Central management has been tacitly presumed throughout this chapter. Essentially this means the designer (or some controller adopting the designs) has access to the channel $\widetilde{H}$ (more generally $p_{\boldsymbol{y}/\boldsymbol{x}}$) and of the input queue depth vector $\boldsymbol{\mathcal{B}}$. Cellular wireless networks use central management as do almost

all wireline transmission systems. However, even if designs are centrally managed, variation may occur too rapidly for the full optimal designs to be updated before the channel changes.

2.6.7.2 Consensus Management or Block Chain

Consensus management ideally attempts to have each and every user transmitter and receiver know $\widetilde{H}$ and $\boldsymbol{\mathcal{B}}$ so that they all know and use the same design strategy and arrive at the same settings. However, because the calculation is local to each transmitter, the result of one design's conclusions can affect other designs. It is theoretically possible if the message passing time is much smaller than the coherence time or rate of change of both the channel and the queue depth that consensus management can achieve the same optima as central management. However, consensus management often instead targets a system where only one user can occupy any dimension. The dimension owner (think a service provider) can contract with any user for that dimension and the users' contracts can be broadcast to all users. No other user then attempts to use another user's dimensions. Any dimensional sharing by users occurs only through different-time uses of the shared dimension, which of course can increase delay. Consensus management typically does not effect full central management, but rather improves random access in Subsection 2.6.7.4 to come. Consensus management may improve significantly over static-management systems, but at at best only achieves the central-management optima. Consensus management typically occurs in network and transport layers (OSI layers 3, 4, and above).

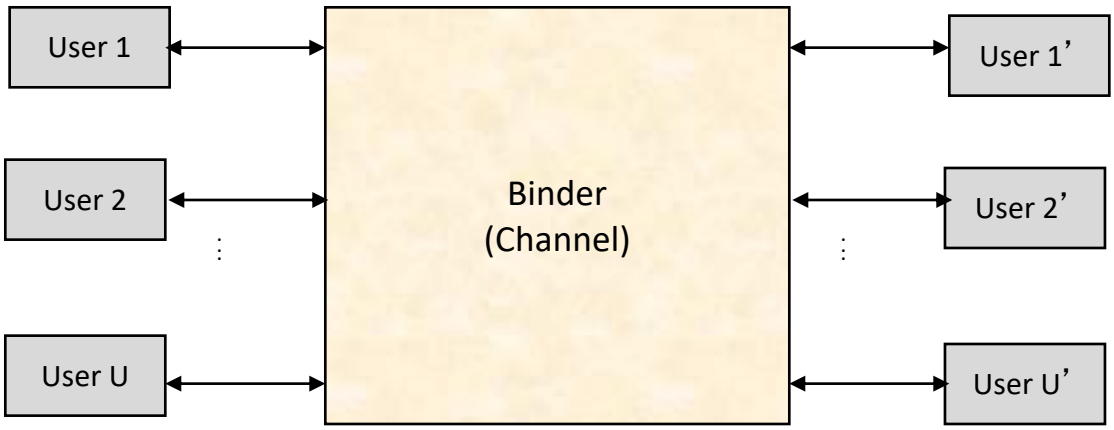


Figure 2.39: **Binder architecture for IC.**

2.6.7.3 Distributed Management

Distributed management essentially has each transmitter make local decisions based on its observation of its own channel $\widetilde{H}_u$. That channel observation may indeed sense other users' crosstalk into the local channel, and so is not devoid completely of overall channel information, as in Figure 2.39. U transmitters and U receivers each communicate only with a counterpart, but experiencing interference from the other users of the channel. There is no central control in the channel. While the architecture is different from Figure 2.40's bus, it is also an example of a vector IC. The channel may "duplex" to support both transmission directions, in which case it may be best modeled as having $2U$ users. The binder situation corresponds to crosstalking channels. Chapter 5, Section 6 discusses strategies for such distributed management.

2.6.7.4 Random Access

Figure 2.40 illustrates the bus architecture where several users attach to a common medium/dimension. This architecture indeed could represent any multiuser system if the medium has many dimensions like a MIMO wireless channel, or a single dimension's shared use. Random access schemes address "accidental" TDM in that uncoordinated transmitters may transmit on the same dimension simultaneously. There is no guarantee however of reliable decoding when the receivers do not know the channel crosstalk, nor the codes used by other users.

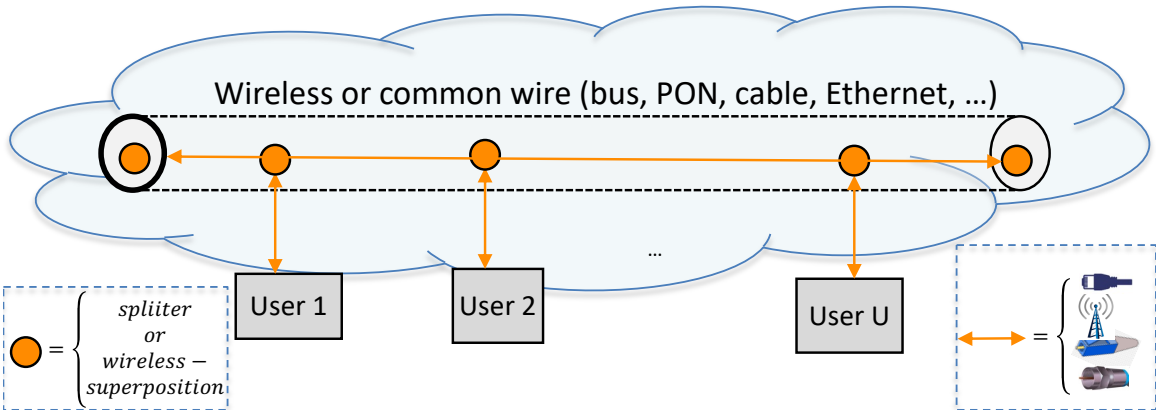


Figure 2.40: **Shared-dimension (NOMA) architecture for multiuser channels.**

Indeed, Figure 2.40 more generally applies to central and other forms of management also, but this diagram best visualizes random access.

In common random access, the individual transmitters simply transmit whenever they want and hope no other user is active. If users either infrequently transmit or are few in number, this works. If two or more transmit at the same time, there usually is a symbol loss. Any erred symbol needs retransmission. Early ethernet used this scheme successfully by simply tapping a single coaxial cable in different places. If certain packets were not received correctly, a higher-level protocol would determine this (via CRC check as in Section 2.2.3) and retransmit any lost symbol. Even with no overhead, this of course increases delay and reduces data rate to **throughput** by (one minus) the time fraction lost to retransmitting the same data. This basic random-access throughput decays rapidly with more active users. The BSC capacity formula from Section 2.4.2 provides insight. Application of this formula with $p = 1/2$ could correspond to two users' ($U = 2$) packets, each running 1/2 the total speed but randomly accessing the channel. The capacity in this case is zero, while centrally controlled TDM would achieve complete reliability with good codes. Even for values of $p = 1/4$, the capacity is greatly reduced. Nonetheless, the appeal of distributed users not needing central control drove several improvements with varying degrees of external (central) "agreement," generally known as Carrier-Sense Multiple Access (CSMA):

> **Definition 2.6.10** [CSMA] **Carrier Sense Multiple Access (CSMA)** senses the channel (so implies a receiver colocates with each transmitter that can sense/detect energy of other users on the target dimensions for intended use), whence the "CS." The multiple access (MA) is analogous to the MAC in that several uncoordinated transmitters access the same channel, although CSMA presumes a single receiver "on the shared dimensions."

Basic CSMA is random access with a local sensor. The sensor allows **collision detection (CD)**.

Early ethernet systems (IEEE 802.3 standard) used CSMA-CD. Because ethernet packets are fairly large (over 10,000 bits typically), continuing a packet's transmission when it is known to collide simply wastes bandwidth. In wired systems, all the sensors can see the channel so the detection is reliable and all will immediately silence transmission and wait a random time period before an another transmission attempt. The data loss is then considerably less than the above simple analysis with the BSC capacity formula. Effectively only a tiny portion of transmission is lost if collisions are not too frequent. The loss factor (p in BSC capacity formula) is reduced substantially by the average fraction of bits sent before detection and silence. However, as users attempt transmission more frequently, the incidence of retransmission has a threshold where performance rapidly degrades with respect to centrally controlled TDM. CSMA-CD typically degrades busy-network throughput to roughly 1/3 of the maximum (sum) rate of the transmission channel. For this reason, most modern ethernet systems use separate-wire connections between users and an ethernet switch, so the switch becomes the central controller. Effectively this system adjusts each user's queue lengths to accommodate each user's fixed channel rate b_u. Such systems will still have delay when all users active bandwidth exceeds the switches' total memory capacity and will have a certain "blocking" probability with any finite-delay (memory-size) constraint.

Collision Avoidance (CA) improves upon CD in that no packets are simultaneously transmitted (so no collisions) without a central authority granting access to random requests. This is used for some wireless MACs, particularly uplink Wi-Fi. Different uncoordinated Wi-Fi systems (think different routers in local proximity) may still collide with simultaneous transmission, so CD also remains present. However, within a specific "access point," uplink users request and are granted or denied permission to transmit. Simultaneous transmission may be allowed by different uplink users when there are multiple antennas. The CA systems will also transmit at a random time later upon a collision, and may also deny permission to transmit, in which case the uplink user waits a random period of time before requesting permission again (so like a collision but with zero packet already transmitted). CA is slightly better than CD, but mainly used in wireless because transmissions from some users may not be sensed by all users (unlike the wireline case). CA systems also see about 1/3 the maximum (sum) rate as throughput. Like modern ethernet, modern Wi-Fi standards evolution increasingly allows both central control (time-slot allocation in CA system instead of transmission-request denial) and consensus management through what are called "coloring schemes" (different adjacent Wi-Fi networks agree to use different channels/times/space denoted by colors with distant Wi-Fi sensing irrelevant).

2.7 MAC Capacity Rate Regions and Designs

This section investigates further the MAC's capacity region, and in so doing also provides design methodology for the matrix AWGN MAC. The general capacity region in Section 2.6.3's Equations (2.281) - (2.285) simplifies considerably through the use of bits/subsymbol sums for the MAC with any particular input and channel probability distributions, $p_{\boldsymbol{x}}$ and $p_{\boldsymbol{y}/\boldsymbol{x}}$. 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 (sub-)symbol rate. This rate-sum formulation exploits the MAC's single receiver to reduce (2.283)'s complexity. Subsection 2.7.1 uses rate sums to address the MAC capacity region $\mathcal{C}(\boldsymbol{b})$'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 2.7.4 expands to continuous time-frequency MACs.

2.7.1 The General Multiple-access Rate Region

Simplest MAC analysis focuses on rate sums

$$b_{\boldsymbol{u}} = \sum_{u \in \{\boldsymbol{u}\}} b_u \ . \tag{2.301}$$

notation: The notation $\{\boldsymbol{u}\}$, or sometimes more simply $\boldsymbol{u}$ when understood in context, refers to a set of users so that $\{\boldsymbol{u}\} \subseteq \{\boldsymbol{U}\}$. Thus (2.301) adds users data rates from users $u \in \{\boldsymbol{u}\}$ to compute the rate sum for the set of users in $\{\boldsymbol{u}\}$. Equation (2.301) applies to any subset $\boldsymbol{u} \subseteq \boldsymbol{U}$. The full rate sum has simple notation $b \triangleq b_{\boldsymbol{U}}$. The notation $\{\boldsymbol{x}\}$ also refers to the set containing the subsymbol vector $\boldsymbol{x}$'s elements. Again, sometimes simply $\boldsymbol{x}$ instead of $\{\boldsymbol{x}\}$ appears when the context is clear. The matrix multiplication expression $\boldsymbol{y} = H \cdot \boldsymbol{x}$ clearly uses vectors, while indices on sums, or limits on integrals, often refer to the sets of elements contained in $\boldsymbol{x}$. The overall rate sum has many equivalent notational specifications such as

$$b = \sum_{u=1}^{U} b_u = \sum_{u \in \{\boldsymbol{U}\}} b_u = \sum_{u \in \{\boldsymbol{U}\}} b_{\pi(u)} = \mathbf{1}^* \boldsymbol{b} = \sum_{u=1}^{|\boldsymbol{U}|} b_u \ . \tag{2.302}$$

A similar expression for any $b_{\boldsymbol{u}} \leq b$ also follows directly for any subset $\boldsymbol{u}$, $b_{\boldsymbol{u}}$, and user u and associated 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 $\boldsymbol{x}$ and $\boldsymbol{b}$, but retain the same rate sums for the same user subsets. The MAC's matrix order $\boldsymbol{\Pi}$ simplifies to a single column vector

$$\boldsymbol{\Pi} = \boldsymbol{\pi} = \begin{bmatrix} \pi(U) \\ \vdots \\ \pi(1) \end{bmatrix} \ , \tag{2.303}$$

or equivalently $\pi_u(i) \equiv \pi(i) \ \forall \ u = 1, ..., U$. The MAC detector treats all individual users' inputs as a single input vector

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_U \\ \boldsymbol{x}_{U-1} \\ \vdots \\ \boldsymbol{x}_1 \end{bmatrix} \ . \tag{2.304}$$

The multi-user channel's mutual information is the same as single-user channel's mutual information $\mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$ when the two allowed input distributions $p_{\boldsymbol{x}}$ are the same. However, the MAC user inputs are independent:

$$p_{\boldsymbol{x}} = \prod_{u=1}^{U} p_{x_u} \ , \tag{2.305}$$

and so often the MAC $p_{\boldsymbol{x}}$ must meet more restrictions than the single-user $p_{\boldsymbol{x}}$. Clearly then, the MAC rate-sum $b \leq \mathcal{I}(\boldsymbol{x};\boldsymbol{y})$, which is the single-user maximum with the MAC's independent-input restriction for the MAC's given $p_{\boldsymbol{x}}$.

Partial rate-sums and independent MAC macro users: As in (2.239) and (2.257), an individual MAC user's b_u may exceed $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$ 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

$$b_u \leq \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{x}_{i\in\{\boldsymbol{U}\setminus u\}}) \quad . \tag{2.306}$$

Subsets $\{\boldsymbol{u}\} \subseteq \{\boldsymbol{U}\}$ each have mutual information with MAC channel output $\boldsymbol{y}$; which are

$$\mathcal{I}(\boldsymbol{x}_{\boldsymbol{u}};\boldsymbol{y}) \; \forall \; \boldsymbol{u} \subseteq \boldsymbol{U} \quad . \tag{2.307}$$

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 $\mathcal{I}(\boldsymbol{x}_\emptyset;\boldsymbol{y}) = 0$. Equation (2.307)'s mutual information represents the partial maximum rate sums over subset of user indices in $\boldsymbol{u}$ with undetected remaining users $i \in \{\boldsymbol{U} \setminus \boldsymbol{u}\}$, so $\mathcal{I}(\boldsymbol{x}_{\boldsymbol{u}};\boldsymbol{y})$'s calculation averages[76] over $\{\boldsymbol{U} \setminus \boldsymbol{u}\}$. These (macro-user) partial rate sums are maximums over the excluded users' i averaged statistics, so these remaining users' effect is as if they are "noise." A higher (or equal) partial rate sum can be achieved for any (macro-) user group $\boldsymbol{u}$ by conditioning the mutual information on all the remaining users, $\{\boldsymbol{U} \setminus \boldsymbol{u}\}$.

Lemma 2.7.1 *[Condition Mutual Information never decreases] Conditional mutual information never decreases below the original mutual information.*

$$\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{X}_{\boldsymbol{U}\setminus u}) \geq \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}) \quad . \tag{2.308}$$

Proof: *By the mutual-information chain rule of Lemma 2.3.4:*

$$\begin{aligned} \mathcal{I}\left(\left[\boldsymbol{X}_{\boldsymbol{U}\setminus u} \;\; \boldsymbol{y}\right];\boldsymbol{x}_u)\right) &= \underbrace{\mathcal{I}(\boldsymbol{X}_{\boldsymbol{U}\setminus u};\boldsymbol{x}_u)}_{0} + \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{X}_{\boldsymbol{U}\setminus u}) \\ &= \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}) + \underbrace{\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{X}_{\boldsymbol{U}\setminus u}/\boldsymbol{y})}_{\geq 0} \quad . \end{aligned}$$

Thus $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{X}_{\boldsymbol{U}\setminus u}) \geq \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y})$. **QED**.

This conditioning essentially means the detector treats all $u \in \boldsymbol{U} \setminus u$, effectively decoded and then canceled in their effect on $\boldsymbol{u}$'s decoding, as in Figure 2.41. Knowledge of the independent set $\{\boldsymbol{U} \setminus \boldsymbol{u}\}$'s specific actual values can only help an optimal detector and thus increases the rate sum bound as per Lemma 2.7.1: $\mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}) \leq \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/\boldsymbol{U} \setminus u)$, with equality if and only if users $\boldsymbol{u}$ and channel output $\boldsymbol{y}$ are jointly independent of $\boldsymbol{U} \setminus \boldsymbol{u}$. Since both (2.306) and (2.307) are mutual information quantities, they each represent maximum (sum) data rates for users in $\boldsymbol{u}$ under their respective conditions. Then the following bound holds for all 2^U possible macro users $\boldsymbol{u}$ and any given input probability distribution and channel:

$$b_{\boldsymbol{u}} \triangleq \left\{ \sum_{u\in\boldsymbol{u}\subseteq\boldsymbol{U}} \mathcal{I}_u(\boldsymbol{x};\boldsymbol{y}) \right\} \leq \mathcal{I}(\boldsymbol{x}_{\boldsymbol{u}};\boldsymbol{y}/\{\boldsymbol{U} \setminus \boldsymbol{u}\}) \quad . \tag{2.309}$$

Equations (2.306), (2.307), and all the instances of (2.309) over user subsets, simplify the achievable MAC multi-user rate-region specification for any given input distribution $p_{\boldsymbol{x}}$. This process averts use of Section 2.6's sets $\mathcal{S}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$, $\mathscr{D}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$, and $\mathbb{P}_u(\boldsymbol{\pi}_i)$.

[76] MAC user subsymbols $\boldsymbol{x}_u$ are independent over $u = 1, ..., U$; in general, all multiuser channels have independent messages $m_u, u = 1, ..., U$. For the MAC, this independence of messages passes also directly to the subsymbols $\boldsymbol{x}_u$.

Chain-Rule Use and Implications: The proof that 2.309)'s achievable region is the bound on $\boldsymbol{b}$ for any given probability distribution uses the single-user chain-rule mutual information bound from Lemma 2.3.4 that

$$\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = \mathcal{I}(\boldsymbol{x}_1;\boldsymbol{y}) + \mathcal{I}(\boldsymbol{x}_2;\boldsymbol{y}/\boldsymbol{x}_1) + ... + \mathcal{I}(\boldsymbol{x}_U;\boldsymbol{y}/[\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ ...,\boldsymbol{x}_{U-1}]) \tag{2.310}$$

$$= \sum_{u=1}^{U} \mathcal{I}(\boldsymbol{x}_u;\boldsymbol{y}/[\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ ...,\boldsymbol{x}_{u-1}])\ . \tag{2.311}$$

The chain rule applies[77] for all user orders $\boldsymbol{\pi}$ with the same rate-sum bound $b \leq \mathcal{I}(\boldsymbol{x};\boldsymbol{y})$, so 2.311 simply emphasizes the multiuser set partitioning in the use of the more general chain rule. Each chain-rule term corresponds to a MMSE estimate when $\boldsymbol{n}$ is Gaussian (and $\boldsymbol{x}$ is a Gaussian code). The chain rule also applies within any user subset $\boldsymbol{u}$ and their partial (macro-user) rate sum bounds are the same, with remaining users given, $I(\boldsymbol{x_u};\boldsymbol{y}\ /\ \boldsymbol{x_{U\setminus u}})$. Thus, a simplification of Section 2.6.3's numerous-situational enumeration instead considers only (2.309)'s possible rate sums for different macro-user subsets $\boldsymbol{u} \subseteq \boldsymbol{U}$.

MAC Successive Decoding: Variation of $\boldsymbol{\pi}$ re-indexes users so that, without loss of generality, the MAC analysis can assume that $\boldsymbol{u} = \{u, ..., U\}$ while $\boldsymbol{U} \setminus \boldsymbol{u} = \{1, 2, ..., u-1\}$. The chain rule suggests a **successive decoding** strategy[78] that detects users within $\{\boldsymbol{U}\setminus\boldsymbol{u}\}$ prior to user u and that then removes their distorting effect on u's detection. This successive-decoder averages[79] over users $i \in \{\boldsymbol{u} \setminus u\}$, so experiences them as noise, but not the prior-decoded users, as in Figure 2.41.

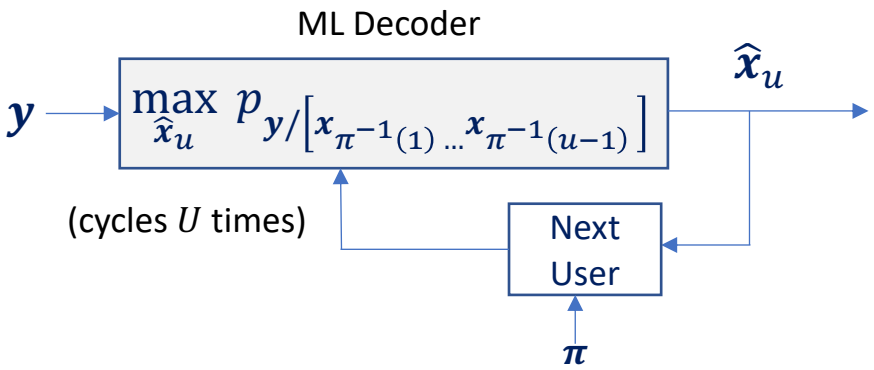


Figure 2.41: MAC successive decoder.

Figure 2.41's conditional probability distribution is

$$p_{\boldsymbol{y}/[\boldsymbol{x}_{\pi^{-1}(1)}...\boldsymbol{x}_{\pi^{-1}(u-1)}]}\left(\boldsymbol{x}_u, [\boldsymbol{x}_{\pi^{-1}(1)}...\boldsymbol{x}_{\pi^{-1}(u-1)}], \boldsymbol{v}\right) \tag{2.312}$$

The order-inverse notation is unnecessary for the MAC. Thus, the remainder of this chapter and text assumes that the MAC's single order relabels users so that user U is at the top (best position) and user 1 is at the bottom, with respect to the chain-rule decomposition. This then simplifies notation so that $\pi^{-1}(u) = \pi(u) = u$ and Figure 2.41 and the previous equation simplify so that

$$p_{\boldsymbol{y}/[\boldsymbol{x}_1...\boldsymbol{x}_{u-1}]}\left(\boldsymbol{x}_u, [\boldsymbol{x}_1...\boldsymbol{x}_{u-1}], \boldsymbol{v}\right) = \tag{2.313}$$

$$= \int_{\boldsymbol{\chi}\in\boldsymbol{X}_{\boldsymbol{U}\setminus\boldsymbol{u}}} p_{\boldsymbol{y}/\boldsymbol{x}}\left([\boldsymbol{x}_1...\boldsymbol{x}_{u-1}], \boldsymbol{x}_u, \boldsymbol{\chi}_{\boldsymbol{U}\setminus\boldsymbol{u}}, \boldsymbol{v}\right) \cdot p_{\boldsymbol{X}_{\boldsymbol{U}\setminus\boldsymbol{u}}}\left(\boldsymbol{\chi}_{\boldsymbol{U}\setminus\boldsymbol{u}}\right) \cdot d\boldsymbol{\chi}_{\boldsymbol{U}\setminus\boldsymbol{u}} \tag{2.314}$$

[77] And is true even if the users are not independent.
[78] This also is called **successive detection**.
[79] Integrates (sums) the distribution $p_{\boldsymbol{y}/\boldsymbol{x}}$ over $\boldsymbol{\chi} \in \boldsymbol{x}_{\boldsymbol{u}\setminus u}$.

Figure 2.41's ML estimate is (for the given $[\hat{\boldsymbol{x}}_1 ... \hat{\boldsymbol{x}}_{u-1}]$ and $\boldsymbol{y} = \boldsymbol{v}$)

$$\hat{\boldsymbol{x}}_u = \arg\max_{\boldsymbol{x}_u} \left\{ p_{\boldsymbol{y}/[\boldsymbol{x}_1 ... \boldsymbol{x}_{u-1}]} \left(\boldsymbol{x}_u, [\hat{\boldsymbol{x}}_1 ... \hat{\boldsymbol{x}}_{u-1}], \boldsymbol{v} \right) \right\} \quad . \tag{2.315}$$

Thus, the MAC detector for a specific channel output $\boldsymbol{y} = \boldsymbol{v}$ progresses from detecting user 1 first as if all others are noise with marginal distribution $p_{\boldsymbol{y}/\boldsymbol{x}_1}(\boldsymbol{\chi}_1, \boldsymbol{v})$, then proceeds to user 2 with higher indexed users $i > 2$ as noise, but user 1's detected symbol $\hat{\boldsymbol{x}}_1$ specifically inserted into marginal $p_{\boldsymbol{y}/[\boldsymbol{x}_1, \boldsymbol{x}_2]}([\hat{\boldsymbol{x}}_1, \boldsymbol{\chi}_2], \boldsymbol{v})$, and proceeding sequentially to user U's detection with (non-marginal)

$$p_{\boldsymbol{y}/\boldsymbol{x}}(\{\hat{\boldsymbol{x}}_1 ..., \hat{\boldsymbol{x}}_{U-1}, \boldsymbol{\chi}_U\}, \boldsymbol{v})$$

.

Successive decoding intuition emphasis: Code design with successive decoding also treats higher-indexed users' crosstalk as "noise," *i.e.* the corresponding decoder averages the marginal distributions for those users $i > u$. Successive decoders perform as if lower-index $i < u$ users are not present. User 1 likely has a lower data rate than user U, for which user U's design only considers the non-user "noise." Re-ordering with various $\boldsymbol{\pi}$ choices allows the designer to trade users' possible b_u values. The single receiver correctly decodes user 1 (with $P_{e,1} \to 0$), $\hat{\boldsymbol{x}}_1 \to \boldsymbol{x}_1$ 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 $\mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$. There are $U!$ orderings, and thus $U!$ data-rate U-tuples, or vertices, that achieve this same $\mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$. Achievable rate-sum calculation computes $\mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$ only once for all these orders. However, finding an acceptable trade between individual data rates investigates the full rate-region, where order variation helps construct $\mathcal{C}(\boldsymbol{b})$. A set of $2^U - 1$ nontrivial corresponding mutual-information quantities bound these macro-user partial rate-sums (bits/subsymbol-sum) through the hyperplane partial sums $\sum_{j=1}^{u} b_j = \mathcal{I}(\boldsymbol{x}_{\boldsymbol{u} \subseteq \boldsymbol{U}}; \boldsymbol{y})$.

Vertex sharing and user components: When $\boldsymbol{u} = \boldsymbol{U}$, the macro-user becomes equivalently a single user. In effect macro user $\boldsymbol{U}$ (all users) partitions into the original users, now from this perspective as user components. All these users have the same sum-rate (independent of $\boldsymbol{\pi}$). An order-independent partial rate-sum also associates within each proper subset $\boldsymbol{u} \subset \boldsymbol{U}$. For $\boldsymbol{u} = \boldsymbol{U}$, the original $i = 1, ..., U!$ orders in the set $\{\boldsymbol{\pi}\}$ each have a vertex rate tuple $\boldsymbol{b}_i$ that sums to the same $\mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$, for which time/dimensional sharing creates a $\boldsymbol{b}$ as

$$\boldsymbol{b} = \sum_{i=1}^{U!} \theta_i \cdot \boldsymbol{b}_i \tag{2.316}$$

$$\text{where} \tag{2.317}$$

$$\sum_{i=1}^{U!} \theta_i = 1 \tag{2.318}$$

$$\theta_i \geq 0 \; \forall \; i = 0, ..., U! \quad , \tag{2.319}$$

is also achievable through simple use of the corresponding codes in the proportions θ_i. Such sharing however effectively increases subsymbol dimensionality.

Rate Regions: Vertex sharing corresponds to partitioning some/all original users into user components over multiple subsymbols with use fractions θ_i. The decoder separately detects each component set for each such macro-to-component set's corresponding MAC usage on $\theta \times 100$ percent of the subsymbols (equivalently creating larger subsymbols). The remaining rate-region calculation finds the 2^{U-1} proper subsets $\{\boldsymbol{u}\} \subset \{\boldsymbol{U}\}$ corresponding to all maximum partial bits/subsymbol sums and hyperplanes (equivalently an order search within each macro group), and finding their intersection's interior, using also the trivial 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 $\boldsymbol{u}$. There are thus $\sum_{u=1}^{U} \binom{U}{u} = 2^U - 1 < U!$ (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-$\boldsymbol{\Pi}$-matrices search procedure as U becomes large.

The chain rule thus helps formalize the MAC rate region for any input distribution as:

Theorem 2.7.1 (MAC Achievable Rate Region) *The general* **multiple-access region** $\mathcal{A}(\boldsymbol{b}, p_{\boldsymbol{x}})$ *for a given user input distribution* $p_{\boldsymbol{x}}$ *is the set of all* U*-dimensional rate vectors* $\boldsymbol{b}$ *that satisfy:*

$$\mathcal{A}(\boldsymbol{b}, p_{\boldsymbol{x}}) = \bigcap_{\boldsymbol{u} \in \boldsymbol{U}} \left\{ \boldsymbol{b} \; \middle| \; 0 \leq \sum_{i \in \{\boldsymbol{u}\}} b_i \leq \mathcal{I}(\boldsymbol{x}_i; \boldsymbol{y} / \boldsymbol{x}_{\{\boldsymbol{u} \setminus i\}}) \right\} \quad . \tag{2.320}$$

Proof: *The successive-decoding discussion leading to this theorem describes achievement of any* $\boldsymbol{b}' \in \mathcal{A}(\boldsymbol{b}, p_{\boldsymbol{x}})$ *point with independent single-user capacity-achieving codes. Through that same discussion, any point outside* $\mathcal{A}(\boldsymbol{b}, p_{\boldsymbol{x}})$ *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* $\{\boldsymbol{U} \setminus \boldsymbol{u}\}$ *given.* **QED**.

The hyperplane's intersection occurs through the convex hull in Equation (2.320) that considers essentially 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 excluded users $\{\boldsymbol{U} \setminus \boldsymbol{u}\}$ for that rate-sum's computation. The computation thus simplifies to one calculation per different macro-user group. The subsequent capacity region simply aggregates over all possible multi-user input distributions $p_{\boldsymbol{x}}$:

Lemma 2.7.2 (MAC Capacity Region) *The MAC capacity region* $\mathcal{C}(\boldsymbol{b})$ *is the maximum, or precisely the union's convex hull of the achievable regions* $A(\boldsymbol{b}, p_{\boldsymbol{x}})$*, over all the possible joint input probability distributions* $p_{\boldsymbol{x}}$ *that the MAC permits:*

$$\mathcal{C}(\boldsymbol{b}) = \left\{ \boldsymbol{b} \, \middle| \boldsymbol{b} \in \bigcup_{p_{\boldsymbol{x}}}^{conv} \mathcal{A}\left(\boldsymbol{b}, p_{\boldsymbol{x}}\right) \right\} \quad , \tag{2.321}$$

where the convex hull (special union operation in (2.321)) appears in (2.316) - (2.319).

Proof: *Any possible input distribution leads to an achievable rate region according to Theorem 2.7.1. 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 component decompositions) some of the different possible probability distributions – assuming such a component partitioning is within the allowed input probability-distribution set.*

By contrast, any point outside the region $C(\boldsymbol{b})$ *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 2.42 provides a 2-user achievable rate-region example for a particular $p_{\boldsymbol{x}}$ that uses both Subsection 2.6.3's method in Equations (2.284) - (2.285) and this subsection's rate-sum method in Equation (2.311).

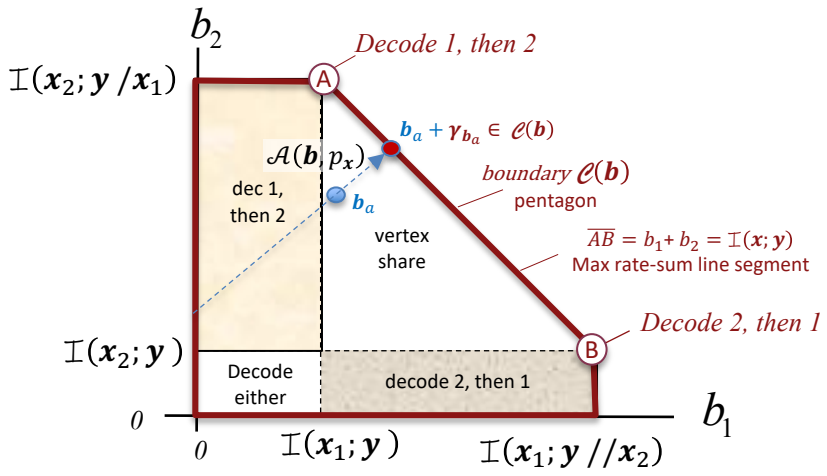


Figure 2.42: Achievable Rate Region for two-user multiple-access channel.

Figure 2.42's 2-user achievable region $\mathcal{A}(\boldsymbol{b},p_{\boldsymbol{x}})$, is the pentagon and its interior (thick cardinal-color outline is $\mathscr{C}(\boldsymbol{b})$ when only a single $p_{\boldsymbol{x}}$ is allowed). The boundary shares solutions through the convex-hull operation along the segment $\overline{\text{AB}}$ from (A) to (B). 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 $p_{\boldsymbol{x}}$. Section 2.6.3's acheivable-region calculation is also fairly easy to compute for any given $p_{\boldsymbol{x}}$ and produces the same region. 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 "vertex-share" region more precisely corresponds to proportionately using (convex hull or union) the two component/users' codes and decoding orders that corresponds to component/user combinations that follow either vertex (A) or (B). Each particular $p_{\boldsymbol{x}}$ again corresponds to a pentagon after the vertex-sharing, for which the 4 non-zero vertices may change. The convex hull of all such pentagons over all allowed $p_{\boldsymbol{x}}$ is $\mathcal{C}(\boldsymbol{b})$, and for which the boundary $\mathscr{C}(\boldsymbol{b})$ may no longer be a pentagon.

Figure 2.42 shows a blue point $\boldsymbol{b}_a\in\mathcal{A}(\boldsymbol{b})$, under the assumption of a Gaussian channel. The scalar AWGN could correspond to a particular energy vector constraint on the two inputs $\boldsymbol{\mathcal{E}}$. The point $\boldsymbol{b}_a$ is in the dimension sharing region so would share orders in a design. A single order may also be implemented by finding a pentagon within $\mathcal{A}(\boldsymbol{b})$ for which $\boldsymbol{b}_a$ is a vertex, but this is not always guaranteed to exist so dimension-sharing of the two decoding orders may be necessary in general. The radial extension of $\boldsymbol{b}_a$ to a second point $\boldsymbol{b}_a+\gamma_{b_a}$ on the boundary $\mathscr{C}(\boldsymbol{b})$ then would have margin $6\cdot\gamma_{b_a}$ dB against noise if capacity-achieving codes were used. The point on the boundary can be difficult to find in general, although Chapter 5 will provide ways of finding it for the MAC (and BC).

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

$$2^U-1+U \qquad \text{faces} \tag{2.322}$$

$$\sum_{k=0}^{U}\binom{U}{k}\cdot k! \qquad \text{vertices}\ . \tag{2.323}$$

(2.322)'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 trivial planes that bound the all-positive orthant. (2.323)'s vertices follow from $k!$ different orders among

the k-user macro groups times $\binom{U}{k}$ possible choices of macro-group k-user sets from U users. There are $U!$ orders that correspond to all U users active (with nonzero energy) on the region's face.

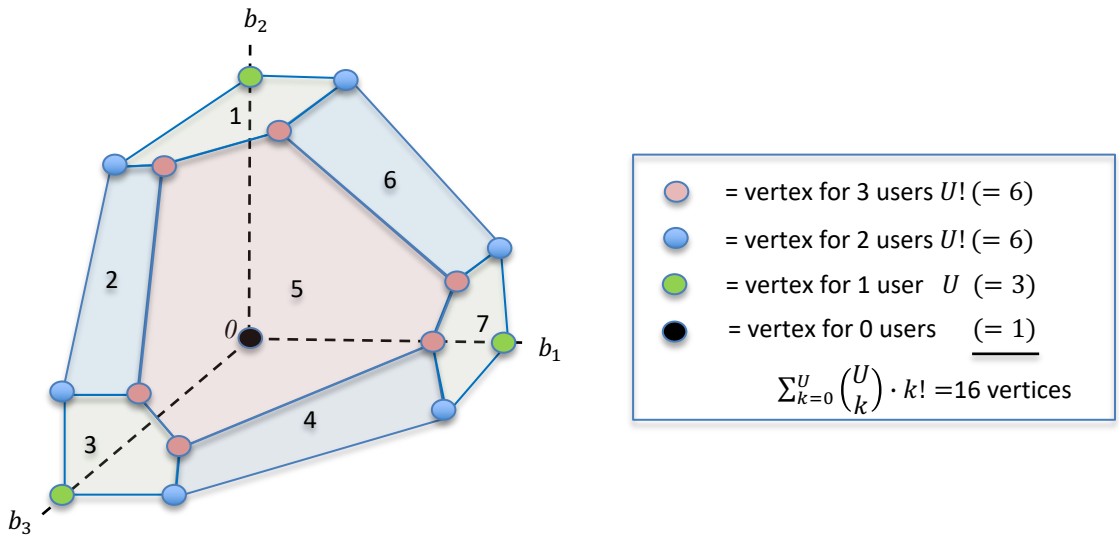


Figure 2.43: Achievable Rate Region for three-user multiple-access channel.

Figure 2.44 illustrates the capacity region formed from Figure 2.43 through the union of different input probability distributions.

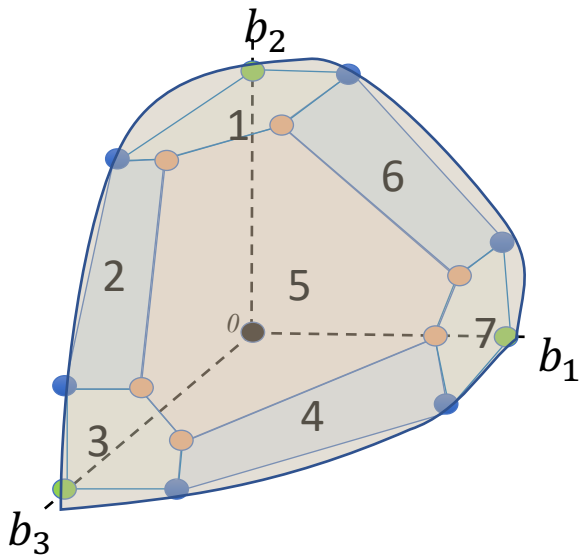


Figure 2.44: Capacity Region for three-user multiple-access channel in Figure 2.43.

Figure 2.43 depicts the achievable region for $U = 3$. There are $2^U - 1 = 7$ numbered (non-trival) faces plus the $U = 3$ (un-numbered) trivial boundaries for the first octant. The vertices are the $U! = 3! = 6$ shown as rose-colored points for all $\binom{3}{1} = 3$ users active, $\binom{3}{2} \cdot 2! = 3 \cdot 2 = 6$ 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[80] to 16, as per Equation (2.323).

2.7.2 The Gaussian MAC Capacity Region and Design.

Figure 2.45's Gaussian MAC has a rate region $A(\boldsymbol{b}, p_{\boldsymbol{x}})$ that 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 (spatial) dimensions now generalizes from L_x to $L_{x,u}$ for user u; thus each MAC user may have different dimensionality. The temporal dimensionality $\bar{N}$ is usually $\bar{N} = 1$ here in this subsection; however, it occasionally increases in anticipation of Subsection 2.7.4's continuous-time multiuser multitone generalization of Section 2.5's results. Chapters 4 and 5 provide more detail on larger $\bar{N}$.

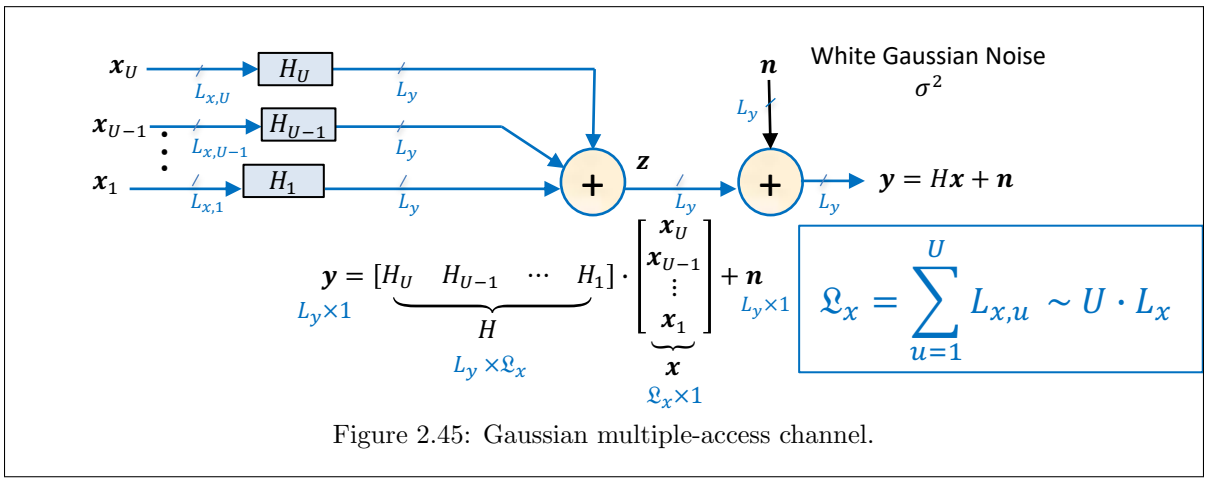


Figure 2.45: Gaussian multiple-access channel.

Each MAC user has its own channel matrix processing, H_u, before addition of other users and of a common Gaussian noise $\boldsymbol{n}$ to construct $\boldsymbol{y}$. When MAC inputs are Gaussian, best design achieves points on or near the rate-region boundary $\mathscr{C}(\boldsymbol{b})$. Gaussian inputs' sufficiency and necessity to achieve boundary points also follow from the last-decoded MAC user:

Lemma 2.7.3 **[All Matrix AWGN MAC users best use Gaussian Distributions]** *The matrix AWGN MAC has largest data rate for any given user autocorrelation-matrix set* $\{R_{\boldsymbol{xx}}(u)\}_{u=1,\ldots,U}$ *when all users' input distributions are Gaussian.* **Proof:** *The MAC distribution with specific MAC output sample* $\boldsymbol{y} = \boldsymbol{v}$*, and with earlier-order users* $\boldsymbol{x}_{u=1,\ldots,U-1}$ *given, is*

$$p_{\boldsymbol{y}/\boldsymbol{x}}(\boldsymbol{\chi}_U, \boldsymbol{v}) = p_{\boldsymbol{n}}\left(\boldsymbol{v} - H_U \cdot \boldsymbol{\chi}_U - \sum_{u=1}^{U-1} H_u \cdot \boldsymbol{x}_u\right) \tag{2.324}$$

$$p_{\boldsymbol{n}}(\boldsymbol{v}' - H_U \cdot \boldsymbol{\chi}_U) \tag{2.325}$$

where $\boldsymbol{v}' = \boldsymbol{v} - sum_{u=1}^{U-1} H_u \cdot \boldsymbol{x}_u$ *at some fixed value. This is a single-user AWGN and has a best input distribution for* $\boldsymbol{x}_U$ *as Gaussian. In turn, because this last user* $\boldsymbol{x}_U$ *has Gaussian distribution, the noise plus crosstalk seen by the 2nd to last decoded user, user* $U-1$*, also has Gaussian distribution* $p_{\boldsymbol{n}+H_U \cdot \boldsymbol{x}_u}\left(\boldsymbol{v} - H_{U-1} \cdot \boldsymbol{\chi}_{U-1}\right)$*. Continuing, user 1 and all lower-indexed users must also be Gaussian by induction.* **QED.**

Lemma 2.7.3 essentially is a special case of Lemma 2.6.4 for the MAC.

[80] But the general formula is not necessarily 2^{U+1}, as evident already for $U = 2$ where there are 5 vertices ($= 1+2\cdot1!+1\cdot2!$

2.7.2.1 The scalar Gaussian MAC

The scalar Gaussian MAC is a very special, and often encountered, case with $L_y = L_{x,u} = 1 \; \forall \; u = 1, ..., U$. When $U = 1$, the MAC is trivially single user. When $U > 1$, the scalar MAC is always degraded when $\bar{N} = 1$. There is a user-specific energy limit $\mathcal{E}_u$ for each user, or equivalently

$$\boldsymbol{\mathcal{E}} \succeq \boldsymbol{\mathcal{E}}_x = \begin{bmatrix} \mathcal{E}_U \\ \vdots \\ \mathcal{E}_1 \end{bmatrix} \tag{2.326}$$

With Gaussian MAC inputs, each and every rate sum corresponds to a macro-user. For the largest (single) macro-user rate, the macro-user distribution must be vector Gaussian. Because this is true for each and every macro-user, each individual user's marginal distribution is then also Gaussian; which yet further simplifies intuition from Lemma 2.7.3. Any proper subgroup of users $\boldsymbol{u} \subset \boldsymbol{U}$ has a marginal distribution for $u \in \boldsymbol{u}$ that is also Gaussian.

Interior points: For interior points $\boldsymbol{b}' \in \mathcal{S}(\boldsymbol{b})$, the energy of one or more users can reduce to create a polytope with a face that includes $\boldsymbol{b}'$. Dimension sharing of that face's vertices' orders/solutions may be needed to achieve the point exactly. The sharing of Gaussian solutions retains Section 2.3's Gaussian AEP code-design properties, although the energy characterizing the component Gaussians can vary with order as long as $\boldsymbol{\mathcal{E}} \preceq \boldsymbol{\mathcal{E}}_x$. Any such point also has extension image $\boldsymbol{c}_{\boldsymbol{b}'} = \boldsymbol{b} + \gamma_b \cdot \boldsymbol{1}$ on the boundary $\mathscr{C}(\boldsymbol{b})$ that corresponds to a multi-user margin gap with respect to energy use; such margin gap also applies to each user's component energy individually. For instance, with complex baseband, $\gamma_b = 1$ is one bit/complex subsymbol, or $\gamma_m = 3 \cdot \gamma_b$ dB.

Boundary points: The mutual information quantities of the scalar ($\widetilde{N} = 1$) MAC capacity region rate-sum boundary descriptions become log-one-plus-signal-to-noise-ratio quantities, where any user's signal energy is the product of the input energy $\bar{\mathcal{E}}_u$ and the channel gain $|H_u|^2$, and the noise is the sum of the Gaussian noise and any uncancelled other-users' mean-square crosstalk. Mathematically, for any $\{\boldsymbol{u}\} \subseteq \{\boldsymbol{U}\}$:

$$\bar{I}(x_{\boldsymbol{u}}; y / \boldsymbol{x}_{\boldsymbol{U} \setminus \boldsymbol{u}}) = \frac{1}{2} \log_2 \left(1 + \frac{\sum_{i \in \boldsymbol{u}} \bar{\mathcal{E}}_i \cdot |H_i|^2}{\sigma^2} \right) \quad . \tag{2.327}$$

This mutual information, as for all Gaussian noise channels, corresponds to a MMSE SNR with $\boldsymbol{x}_{\boldsymbol{U} \setminus \boldsymbol{u}}$ given. For a scalar AWGN MAC (so $L_y = 1$) example with $U = 2$, the 2×1 channel is $H = [h_1 \; h_2]$, so then

$$y = h_1 \cdot x_1 + h_2 \cdot x_2 + n \quad . \tag{2.328}$$

Each rate sum (pentagon, more generally for $U > 2$ polytope, face) maps to an SNR:

$$\text{SNR}_1 = \frac{\mathcal{E}_1 \cdot |h_1|^2}{\sigma^2} \tag{2.329}$$

$$\text{SNR}_2 = \frac{\mathcal{E}_2 \cdot |h_2|^2}{\sigma^2} \tag{2.330}$$

$$\text{SNR} = \frac{\mathcal{E}_1 \cdot |h_1|^2 + \mathcal{E}_2 \cdot |h_2|^2}{\sigma^2} \quad , \tag{2.331}$$

and

$$\bar{b}_1 \leq \overline{\mathcal{I}}_1 = \frac{1}{2} \cdot \log_2 (1 + \text{SNR}_1) \tag{2.332}$$

$$\bar{b}_2 \leq \overline{\mathcal{I}}_2 = \frac{1}{2} \cdot \log_2 (1 + \text{SNR}_2) \tag{2.333}$$

$$\bar{b}_1 + \bar{b}_2 \leq \overline{\mathcal{I}} = \frac{1}{2} \cdot \log_2 (1 + \text{SNR}) \quad . \tag{2.334}$$

The SNR's, and thus rate-sum bounds in (2.332) - (2.334) 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 (2.332) - (2.334) also specify the capacity region, along with $\bar{b}_1 \geq 0$ and $\bar{b}_2 \geq 0$. Chapter 5 finds that there may be other energy-sum minimizing solutions for that interior point with a single order and possibly preferable to this vertex-shared realization.

Order-Dimension share from vertex set: In general with $U > 2$ scalar (degraded and $L_y = 1$) Gaussian MACs sum users, and the pentagon generalizes to an U-dimensional polytope with $2^U + U - 1$ sides in U-space ($\mathbb{R}^U$). 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 $\bar{\mathcal{C}}_u = .5 \cdot \log_2(1 + \text{SNR}_u)$ where $\text{SNR}_u = \mathcal{E}_u \cdot |H_u|^2/\sigma^2 = \mathcal{E}_{\boldsymbol{x}} \cdot g_u$. There are hyperplanes for all possible subsets $\boldsymbol{u}$, each as $\sum_{u \in \boldsymbol{u}} b_u \leq \mathcal{C}_{\boldsymbol{u}} = 1/2 \cdot \log_2(1 + \text{SNR}_{\boldsymbol{u}})$ where $\text{SNR}_{\boldsymbol{u}} = \sum_{u \in \boldsymbol{u}} \mathcal{E}_u \cdot |H_u|^2/\sigma^2$.

In general for a facial point with K vertex rate vectors $\boldsymbol{\beta}_k$, $u = 1..., K$, with components $\beta_{u,k}$ $k = 1, ..., K$. These will be shared in proportions α_k where $0 \leq \alpha_k \leq$ each with $1 = \sum_{k=1}^{U} \alpha_k$. Then

$$\underbrace{\begin{bmatrix} \beta_{K,U} & ... & \beta_{K,1} \\ \vdots & \ddots & \vdots \\ \beta_{1,U} & ... & \beta_{1,1} \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} \alpha_K \\ \vdots \\ \alpha_1 \end{bmatrix}}_{\boldsymbol{\alpha}} = \boldsymbol{b} \tag{2.335}$$

is a nonsingular matrix equation that can be solved for $\boldsymbol{\alpha} = B^{-1} \cdot \boldsymbol{b}$ to obtain the order/dimension-sharing proportions.

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

$$b_{max} = \log_2 (1 + \mathcal{E}_{\boldsymbol{x}} \cdot g_{max}) \text{ bits/complex-subsymbol} \tag{2.336}$$

where $g_{max} \triangleq \max_u g_u$ by definition. In this case, the maximum rate sum depends on order and in fact must have user U at the top of $\boldsymbol{\pi}_U$ with all energy allocated to this unique "primary" user component.

Vertex-sharing design choices: Time, or more generally, vertex-sharing really implies alternating between $\boldsymbol{b}$ MAC-rate-region vertices that may have different different decoding orders $\boldsymbol{\pi}$. Subtly tacit in this sharing is a presumption that the average energy/symbol remains the same for the shared points. However, further reflection reveals that indeed the energy itself also could be varied for the design's shared vertices, expanding a space of possible consideration. This point is abjectly missing in most information-theory texts and consideration. Energy-sharing may enlarge the capacity region because, depending on the average-energy calculation's presumptions. The shared vertices, each must use different good-codes' encoders because the vertices' user order differs. means effectively partitioning the AEP process selecting different numbers of samples from the input distribution for the different user-code rates in the different shared points. This may mean up to U such codes for each user, but the average data rate vector is the desired user-rate combination. Under the dubious pretense of subsymbol energy cannot be averaged in the code construction, the capacity rate region for a given "fixed"' energy vector is a polytope (pentagon when $U = 2$); the capacity rate region $\mathcal{C}_{MAC}(\boldsymbol{b})$ is not necessarily a polytope/pentagon with dimension sharing over a set of shared subsymbols/orders and averaged energies. The convex-hull operation over the sharing essentially raises the issue of how is the sharing done? Holding the sampling rate constant, and enlarging the symbol period to allow essentially time-fractioning of the different vertices within the larger symbol, will not change the capacity region. However, if a fixed subsymbol period is divided into a larger number of subsymbol periods with a higher (faster) subsymbol rate, Chapter 1's spread-spectrum effect occurs, altering $\mathcal{C}_{MAC}(\boldsymbol{b})$. If the channel energy remains the same for each new subsymbol, then $\mathcal{C}_{MAC}(\boldsymbol{b})$ remains a polytope (although an enlarged one), although the data-rate vector $\boldsymbol{b}$ varies over the subsymbols within a symbol. Such dimension sharing introduces no delay (although the codes

already are asymptotic, the growth towards asymptotic is slower and hence effectively increases latency). Inevitably, some bandwidth limit must apply for the capacity region to have useful meaning. Example 2.7.1 illustrates this effect in a simple situation.

Scalar-AWGN MAC Capacity-Energy Region: The scalar AWGN MAC with $\widetilde{N} = 1$ has an energy-capacity region $\mathcal{C}_{\boldsymbol{b}}(\boldsymbol{\mathcal{E}})$ with $U!$ finite vertices. These vertices correspond to the $U!$ possible decoding orders. There is a $\mathscr{C}(\boldsymbol{b})$ boundary hyperplane containing the $U!$ vertices. Figure 2.47 in the Example 2.7.1 that follows is an example for $U = 2$ with a line segment being the nontrivial boundary. $U = 3$ correspondingly has a planar face as in Figure 2.43 with 6 vertices. Each Figure 2.43 non-facial line segment corresponds to the face of a two-dimensional capacity-energy face (line) like that in Figure 2.47(b) extending to infinity instead of to the trivial boundary pentagons as in Figure 2.43. For $\mathcal{C}_{\boldsymbol{b}}(\boldsymbol{\mathcal{E}})$, the single $U-1$-dimensional hyperplane face with $U!$ vertices is of most interest for the MAC. Rate vectors within the face correspond to dimension/order sharing of that face's vertices. Any interior point $\boldsymbol{\mathcal{E}}' \in \mathcal{S}_{\boldsymbol{b}}$ uses more energy than necessary and has a positive excess-energy ratio

$$\Gamma_{mu,\boldsymbol{\mathcal{E}}}(\boldsymbol{b}) = \frac{\sum_{u=1}^{U} \mathcal{E}'_u}{\sum_{u=1}^{U} \mathcal{E}^o_u} \tag{2.337}$$

where $\boldsymbol{\mathcal{E}}^o \in \mathscr{C}_{\boldsymbol{b}}(\boldsymbol{\mathcal{E}})$ is the radial image point in the capacity-energy-region boundary.

The following example illustrates both the capacity region and the capacity-energy region for a two-user scalar MAC:

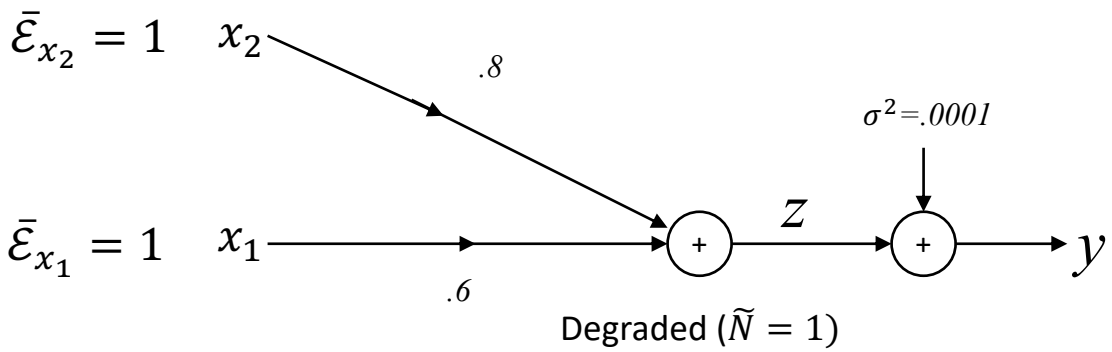


Figure 2.46: Example 2.48's scalar 2-user MAC AWGN capacity calculation.

EXAMPLE 2.7.1 (Scalar AWGN channel) Figure 2.46 illustrates a simple 2-user AWGN channel with capacity regions in Figure 2.47. This channel's sum-rate capacity follows easily by recognizing $\boldsymbol{z}$ is a single-user (macro-user) AWGN channel input. This single-user capacity is $\mathcal{I}(x;y) = \mathcal{I}(z;y) = .5 \cdot \log_2(1 + 10^4)$ =6.6439 bits/dimension. However, consideration of either user's signal as Gaussian noise, which distorts the other users, leads to marginal capacities of

$$\begin{aligned}
\mathcal{I}(x_1;y) &= \frac{1}{2} \cdot \log_2\left(1 + \frac{.36 \cdot 1}{.0001 + .64}\right) = .3219 \text{ bits/dimension} && (2.338)\\
\mathcal{I}(x_2;y) &= \frac{1}{2} \cdot \log_2\left(1 + \frac{.64 \cdot 1}{.0001 + .36}\right) = .7368 \text{ bits/dimension} && (2.339)\\
\text{total} &= 1.0587 \text{ bits/dimension} \quad , && (2.340)
\end{aligned}$$

which is well below the maximum rate sum of $\bar{b}_{max} = 6.6439$ bits/dimension. Bits/dimension here corresponds to the single dimension that both users share.

Decoding x_1 first (point Ⓐ in Figure 2.47(a)) leads to

$$\mathcal{I}(x_2; y/x_1) = \frac{1}{2}\cdot\log_2(1+\mathrm{SNR}_2) = \frac{1}{2}\cdot\log_2\left(1+\frac{.64\cdot 1}{.0001}\right) = 6.3220 \text{ bits/dimension}\ , \quad (2.341)$$

or decoding x_2 first (point Ⓑ) leads to

$$\mathcal{I}(x_1; y/x_2) = \frac{1}{2}\cdot\log_2(1+\mathrm{SNR}_1) = \frac{1}{2}\cdot\log_2\left(1+\frac{.36\cdot 1}{.0001}\right) = 5.9071 \text{ bits/dimension} \quad (2.342)$$

In either case, the sum is the same as .3219+6.3220 =.7368+5.9071=6.6439 bits/dimension. Figure 2.47(a) illustrates the capacity rate region and the two successive-decoding vertices, Ⓐ and Ⓑ. Figure 2.48 illustrates this channels' two successive-decoding receivers. Other points on the line $b_1 + b_2 = 6.64$ between Ⓐ and Ⓑ correspond to vertex sharing of the two orders. In the Gaussian case, vertex-sharing of two different distributions always corresponds to a single vector Gaussian distribution for each user, but there are two codes (AEP sense) selected from that single distribution, one for the user rate corresponding to vertex Ⓐ for a fraction of the time (uses) and one for the user rate corresponding to vertex Ⓑ the remaining fraction of the time (uses), which effectively means the input is not stationary (possibly cyclo-stationary if a regular pattern is used). These different codes are a simple form of user components - each user now has two components corresponding to the two time-shared codes. Each component encodes and decodes separately. Such vertex sharing also exemplifies the bandwidth question, specifically the period over which the energy constraint applies - is it every symbol individually or the average over the time-shared interval? If time-sharing creates longer symbols, then the $\mathcal{C}(\boldsymbol{b})$ remains as shown.

Nonetheless continuing, a time-sharing of 3/4 vertex Ⓑ and 1/4 vertex Ⓐ yields point Ⓒ with

$$\begin{aligned}
\mathcal{E}_{x_2} &= \frac{1}{4}(1) + \frac{3}{4}(1) = 1 && (2.343)\\
\mathcal{E}_{x_1} &= \frac{1}{4}(1) + \frac{3}{4}(1) = 1 && (2.344)\\
b_2 &= \frac{1}{4}(6.3220) + \frac{3}{4}(.7638) = 2.1331 && (2.345)\\
b_1 &= \frac{1}{4}(.74) + \frac{3}{4}(6.32) = 4.5108 && (2.346)\\
b_1 + b_2 &= 6.6439\quad . && (2.347)
\end{aligned}$$

This example specifically illustrates that vertex sharing can increase the number of subsymbol temporal dimensions $\widetilde{N}$ - in this example from 1 dimension per subsymbol to 4 dimensions per subsymbol, 3 with code Ⓑ and 1 with code Ⓐ, achieving $\boldsymbol{b}' = [2.1331\ 4.5108]^*$. If the energy vector remains as $\boldsymbol{\mathcal{E}} = [1\ 1]^* \triangleq \mathbf{1}$ on BOTH code Ⓐ and Ⓑ, then Ⓒ is the only point that achieves the rate $\boldsymbol{b}_{\prime}$; however dimension sharing admits other energy vectors as long as $\mathbb{E}[\boldsymbol{\mathcal{E}}] = \mathbf{1}$. Figure 2.47(a) also shows $\mathcal{C}_{\boldsymbol{b}'}(\boldsymbol{\mathcal{E}})$, which enumerates all the possible energy vectors where good codes can reliably achieve $\boldsymbol{b}'$. For instance, point Ⓓ achieves $\boldsymbol{b}'$ with minimum sum energy $\mathcal{E}_{\boldsymbol{b}'_{prime,min}} = 1.4815 + .1145 = 1.6255 < 2$, but would violate user 2's energy constraint. Similarly, Ⓔ is another vertex point, but violates user 1's energy constraint. However, Point Ⓒ achieves $\boldsymbol{b}$ by 67% point Ⓓ and 33% point Ⓔ; so with $\bar{N} = 3$, one encoder implements $\mathcal{E}_{\text{Ⓓ}}$ and the other encoder implements $\mathcal{E}_{\text{Ⓔ}}$. All 3 have $\boldsymbol{b} = [2.1331\ \ 4.5108]$, and the average energy-vector constraint is met. The line $\mathcal{E}_1 + \mathcal{E}_2 = \mathcal{E}_sum$ is tangent at Ⓓ.

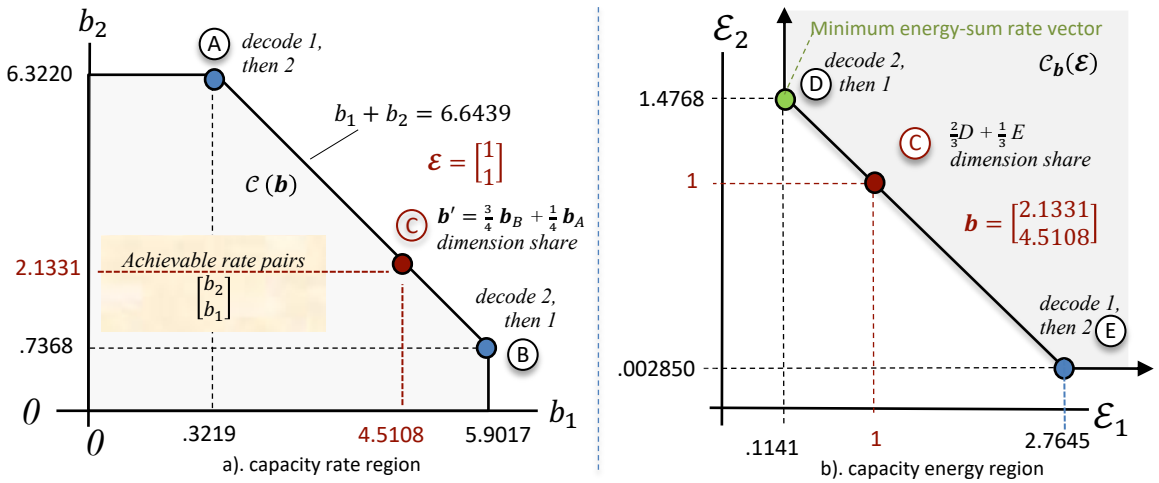


Figure 2.47: 2-user AWGN rate and energy regions.

EXAMPLE 2.7.1 CONTINUED

For constant subsymbol-level average energy, dimension-sharing for point (C) groups 4 subsymbols, 3 for solution (B) and 1 for solution (A).

Point (C) is the common operating point for both the capacity region and the capacity-energy region and dimensionally shared. The rate $\boldsymbol{b} = [2.1331\ 4.5108]$ is achievable with less sum energy at point (D) than at point (C), but violates user 2's individual energy constraint. Similarly, Figure 2.49 shows rate-sums of up to 6.82 are achievable for 2 units of sum energy, but user 1's data rate of 4.5108 would not be achieved despite this maximum sum rate.

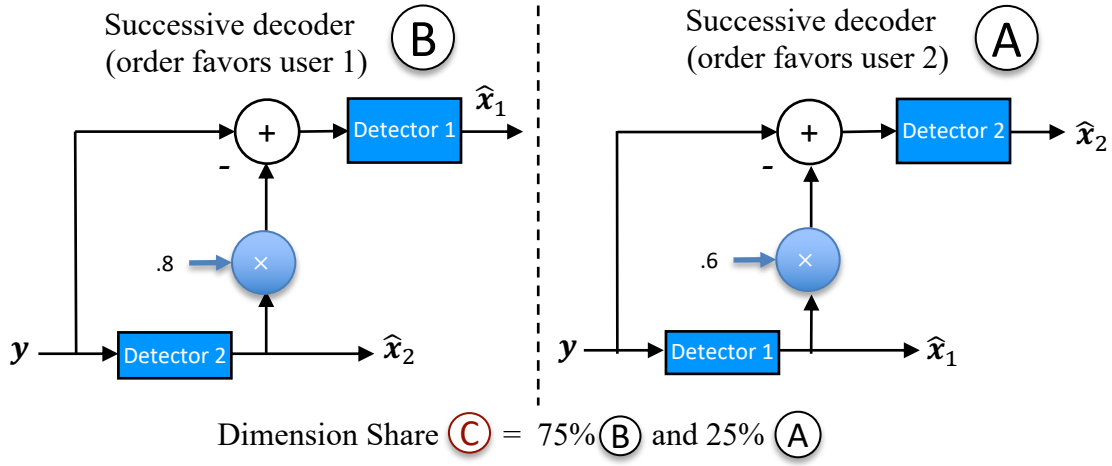


Figure 2.48: Example successive decoders.

The energy calculations for Points Ⓓ and Ⓔ are:

$$\mathcal{E}_{Ⓓ,2} = \left(2^{2(2.1331)}-1\right)\cdot\frac{1}{6400} = .002850 \tag{2.348}$$

$$\mathcal{E}_{Ⓓ,1} = \left(2^{2(4.5108)}-1\right)\cdot\frac{\cdot 2^{2\cdot 2.1331}}{3600} = 2.7645 \tag{2.349}$$

$$\mathcal{E}_{Ⓔ,1} = \left(2^{2\cdot 4.5108}-1\right)\cdot\frac{1}{3600} = .1441 \tag{2.350}$$

$$\mathcal{E}_{Ⓔ,2} = \left(2^{2\cdot 2.1331}-1\right)\cdot\frac{2^{2\cdot 4.5108}}{6400} = 1.4768 \ . \tag{2.351}$$

To find the point Ⓒ's convex combination,

$$\alpha\cdot\begin{bmatrix}1.4768\\.1141\end{bmatrix}+(1-\alpha)\cdot\begin{bmatrix}.002850\\2.7645\end{bmatrix}=\begin{bmatrix}1\\1\end{bmatrix} \tag{2.352}$$

has solution $\alpha = 2/3$. The averaged-energy solution (at fixed rate $\boldsymbol{b}'$) is easier in that each user has a fixed rate (and presumably scaled constellation set of points as implemented) and requires less latency because the time-shared larger subsymbols are 3 original subsymbols long (1 subsymbol Ⓔ for every 2 subsymbols Ⓓ).

Figure 2.49's energy-sum-MAC $\mathcal{C}(\boldsymbol{b})$-construction relaxes input energy constraints to an energy-sum MAC, so

$$\mathcal{E}_1+\mathcal{E}_2=\mathcal{E}_{\boldsymbol{x}} \ , \tag{2.353}$$

which might characterize multiuser total energy with a common energy source (even if in different locations). All energy on user 2, $\mathcal{E}_2 = \mathcal{E}_{\boldsymbol{x}} = 2$, provides the largest data rate of $b_2 = b_{max} = 6.82$ bits/subsymbol. All energy on user 1 leads to $b_1 = 6.4 < b_{max}$ bits/subsymbol. The region is curved with slope -1 (corresponding to the line $b_1+b_2 = b_{max}$) 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 2.49. Any single point corresponds to a single pentagon vertex (single order $\boldsymbol{\pi} = [12]'$) for some $\boldsymbol{\mathcal{E}} \ni \mathbf{1}^*\boldsymbol{\mathcal{E}} = \mathcal{E}_{\boldsymbol{x}}$; but not necessarily such that an original $\boldsymbol{\mathcal{E}}_{\boldsymbol{x}}$ constraint is met by all users' energies. Meeting such a constraint may require vertex-sharing (dimension/time-sharing) of the two orders on the same (or even different) pentagons. A "longer-term" vertex-sharing constraint could be viewed over shorter-term averages as an energy-sum constrain (with different energy sums over longer time averaging to the target $\boldsymbol{\mathcal{E}}_{\boldsymbol{x}}$.

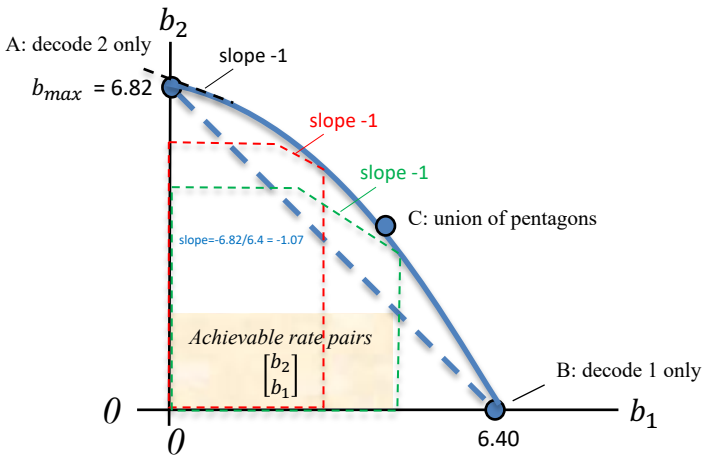


Figure 2.49: Capacity and Energy-Capacity regions for constant power with doubled sampling rate.

EXAMPLE 2.7.1 FURTHER CONTINUED

Another design strategy shares different energies that average to the same energy vector $\boldsymbol{\mathcal{E}} = [1\ 1]'$. Such design then admits dividing the subsymbol (holding latency constant) but exploiting the AWGN's essentially unrestricted bandwidth (increase W in $N = 2WT$) and holding power constant (which means energy/dimension for example halves if there are twice as many dimensions/second). The channel itself maintains its gain over twice as many dimensions if $\bar{N} = 2$, which means H remains $[80\ 60]'$ for both dimensional uses; however input energy $\bar{\mathcal{E}}_{\boldsymbol{x}}$ halves. Because the two dimensions are identical, by symmetry they are essentially interchangeable. The spread-spectrum effect then leads to a larger rate region because the input energy constraint now applies to the sum of energies on the two identical channels, thus effectively changing one of the parameters for the rate region's construction.

Figure 2.50 illustrates the corresponding capacity-rate region $\mathcal{C}(\boldsymbol{b})$ and capacity-energy region $\mathcal{C}(\boldsymbol{\mathcal{E}})$ with dimension-sharing subdividing a subsymbol into 2 half subsymbols that each have 1/2 the energy. The margin-gap is roughly 2.4 dB. Chapter 1's spread spectrum effect is evident on both users' data rates, even though the power remains the same (1 unit) for each user, equally divided now 1/2 on each of the two dimensions. The rate vector $\boldsymbol{b} = [2.113\ 4.5108]$ now is well within the region allowing many time-sharing solutions. The capacity energy region to the right shows the point $\boldsymbol{\mathcal{E}} = [.048\ .024]$ as the point with lowest energy sum, and indeed both users have energies well below the maximum, exploiting the spread spectrum effect to retain data rate and reduce power. The lowest latency uses the new point (D) with a single order on all subsymbols at the desired rate. The users each also use only 1 code for all subsymbols. This example illustrates that when the continuous-time channel's bandwidth supports expansion, that expansion likely eliminates the need for time sharing of codes (which is a theoretical abstraction for information theory, but likely not a desired implementation). Of course, use of wider spectrum W is not always possible in practice.

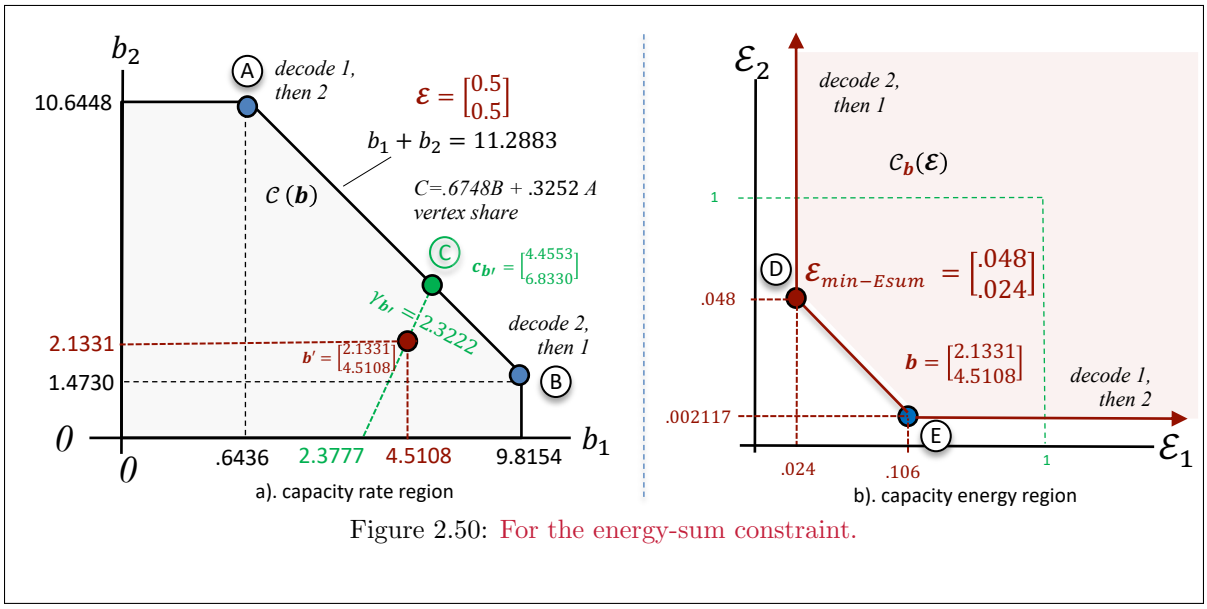


Figure 2.50: For the energy-sum constraint.

Scalar MAC primary and secondary user components: Example 2.7.1 illustrates that one scalar MAC user component is **"primary"** in that this channel's energy-sum-MAC maximum data rate computes as if there is no crosstalk (again, for this example). The other user component is **secondary** in that its signal shares the primary user's common output dimension but experiences that primary-user component as crosstalk noise energy that reduces its reliably achievable data rate. The energy-sum MAC's primary and secondary users/components retain their energy-sum-based characterization even if there is an energy vector $\boldsymbol{\mathcal{E}}$ constant. This channel has $L_{x,u} = 1$ for all users, and so each user is equal to its component if the design uses energy-vertex sharing; however, when rate-vertex sharing, there are two components for each user. The primary user component is user 2's active component. For channels with $L_{x,u} > 1$ for any user(s), the user has multiple components and the distinction of primary and secondary becomes more complex. The sum-energy MAC becomes of interest later (in duality with BC's) as $\mathbf{1}^* \boldsymbol{\mathcal{E}} \leq \mathcal{E}_{\boldsymbol{x}}$ (=2 in Example 2.7.1), although it can also be of interest when the user group's energy use is a constraint.

For the scalar MAC with $L_y = 1$ and $U > 2$ users, one user (largest channel gain g_U after ordering) always has the primary component and the others are all secondary components. 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 $g_u = \frac{|h_u|^2}{\sigma^2}$ such that $g_U \geq g_{U-1} \geq ... \geq g_1$. When each user has a different and separate non-zero energy constraint, then the order generalizes to

$$\mathcal{E}_U \cdot g_U \geq \mathcal{E}_{U-1} \cdot g_{U-1} \geq ... \geq \mathcal{E}_1 \cdot g_1 \quad . \tag{2.354}$$

The tangent line/plane at the maximum sum rate generalizes to

$$\mathbf{1}^* \boldsymbol{b} = b_{max} \text{ when } U > 2 \quad . \tag{2.355}$$

2.7.2.2 The Vector Gaussian MAC by Design

With the vector MAC's larger output dimensionality $L_y > 1$, this subsection shows that is possible for more than 1 MAC user component to be "primary" and thus associate simultaneously with the maximum rate-sum when the MAC has only a energy-sum constraint. This subsection also provides a MMSE MAC design that reliably achieves any capacity-region rate vector, along with a software-design program mu_mac.m that provides the basic design elements for realization.

The vector Gaussian MAC has either or both of $L_y > 1$ and/or $L_{x,u} > 1$. Each user subchannel H_u has SVD $H_u = F_u \cdot \Lambda_u \cdot M_u^*$. Two useful concepts are

Pass Space

$$\mathcal{P}_{H_u} \triangleq \{\boldsymbol{m}_i \mid \lambda_{u,i} > 0 \; \forall \; i \in \{1, ..., L_{x,u}\}\} \quad . \tag{2.356}$$

Null Space

$$\mathcal{N}_{H_u} \triangleq \{\boldsymbol{m}_i \mid \lambda_{u,i} = 0 \; \forall \; i \in \{1, ..., L_{x,u}\}\} \quad . \tag{2.357}$$

Clearly $\mathcal{P}_{H_u} \bigcup \mathcal{N}_{H_u} = \mathbb{C}^{L_{x,u}}$. Any input pass-space component for an $R_{\boldsymbol{xx}}(u)$ has nonzero channel output. Any null–space component for $R_{\boldsymbol{xx}}(u)$ does not pass. If a good code had two codewords that were both in the null space, they would be indiscernible for a detector and thus neither would be reliably detectable. Any energy for such codewords in such a dimension would have to have the same subsymbol value, which is simply wasteful. Good code design does not energize null-space components. For a general input $R_{\boldsymbol{xx}}(u)$ such that $\boldsymbol{x}_u = R_{\boldsymbol{xx}}^{1/2}(u) \cdot \boldsymbol{v}_u$ it is possible there are null space components in $\boldsymbol{x}_u$ that satisfy, for any $\boldsymbol{m}_i \in \mathcal{N}_{H_u}$,

$$0 = \boldsymbol{m}_i \cdot R_{\boldsymbol{xx}}^{1/2}(u) \cdot \boldsymbol{\nu}_u \quad . \tag{2.358}$$

These components contribute nothing to transmission. So useful input construction then focuses upon

$$\boldsymbol{x}_u = \underbrace{\boldsymbol{m}_i \cdot R_{\boldsymbol{xx}}^{1/2}(u)}_{R_{\boldsymbol{xx}}^{o/2}(u)} \cdot \boldsymbol{\nu}_u \quad . \tag{2.359}$$

When $R_{\boldsymbol{xx}}(u) \neq R_{\boldsymbol{xx}}^{o}(u)$, the difference corresponds to user u's (wasted) energy in the channel null space. Again, good design zeros such energy.

Subsection 2.7.3 provides a simultaneous water-filling process that determines which users carry non-zero energy for the maximum-rate energy-vector MAC. The maximum-energy-sum MAC data-rate might be found by using this procedure for all possible energy vectors or computing with a convex-solver as in the macmax.m program later in this section. Sections 2.8 and 5.5 also provides a method using duality that can also more simply find this point and corresponding best energy-sum $trace\{R_{\boldsymbol{xx}}\}$. Energization of $R_{\boldsymbol{xx}}^{o}$ for any given $R_{\boldsymbol{xx}}$ enables a good code to achieve the maximum mutual information bound reliably. The maximum rate sum for the energy-sum MAC energizes only primary components, by definition of primary components. When $L_y > 1$ and/or $L_{x,u} > 1$, the users may have multiple components, so the previous sentence does not say that the maximum rate sum only energizes some primary users, rather components. When $L_y = L_{x,u} = 1$, then the statement is true.

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 $H_u \cdot \hat{\boldsymbol{x}}_u$ from $\boldsymbol{y}$ to detect users $i > u$. Subsection 2.3.6 showed that mutual information corresponds to a MMSE estimate, which only simplifies to $h_u \cdot \hat{x}_u$ when $L_y = 1$ and $L_{x,u} = 1$. Subsection 2.3.6 showed the MMSE estimate to be the ML decoder decision, which only equals the ZF solution ($h_u \cdot \hat{x}_u$) when there is only 1 user component. When $L_y > 1$, typically the maximum number of components is $L_y \times \mathscr{L}_x$ because each of a MAC-input user's dimensions (per subsymbol) could have a component that reaches any output dimension, if energized. Often the (information-lossless) inclusion of M_u into a redefined input $R_{\boldsymbol{xx}}(u) \rightarrow M_u \cdot R_{\boldsymbol{xx}} \cdot M_u^*$ is one way to analyze the MAC with sub-users (with $\widetilde{H}_u = F_u \cdot \Lambda_u \cdot M_u^*$ SVD). If $\varrho_{\widetilde{H}_u} < L_{x,u}$, then the number of sub-user components will be less than this maximum, as will also be the case if $\varrho_{\boldsymbol{x},u} < L_{x,u}$.

Figure 2.45's more general Gaussian MAC has matrix[81] H that is $L_y \times \mathscr{L}_x$, where

$$\mathscr{L}_x \triangleq \sum_{u=1}^{U} L_{x,u} \quad . \tag{2.360}$$

[81] $n = 1, ..., N$ 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.

The MAC situations where $L_y > 1$ and/or $L_{x,u} > 1$ need further examination for primary and secondary user components.

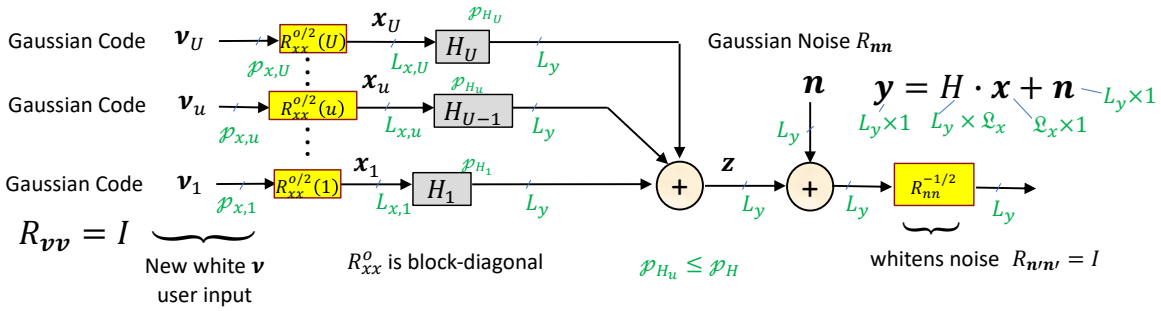


Figure 2.51: Vector MAC transformation.

To avoid cumbersome notation, the remainder of this text's discussion uses $R_{\boldsymbol{xx}}(u)$ to represent $R^o_{\boldsymbol{xx}}(u)$ because null-space components carry no information. Each user's encoder presumably then avoids null-space energy.

Vector Gaussian MAC Normalization with known block diagonal $R_{\boldsymbol{xx}}$: The following prepares the vector MAC for common analysis with a fixed given set of user autocorrelation matrices $\{R_{\boldsymbol{xx}}(u)\}_{u=1,...,U}$ The MAC-user sub-matrices H_u normalize through definition of the $L_y \times L_{x,u}$ matrices (for $u = 1, ..., U$)

$$\widetilde{H}_u \triangleq R_{\boldsymbol{nn}}^{-1/2} \cdot H_u \cdot R_{\boldsymbol{xx}}^{1/2}(u) \quad . \tag{2.361}$$

User u partitions into $\varrho_{x,u} \triangleq \varrho_{R_{\boldsymbol{xx}}(u)}$ components, one for each non-zero probability component of the random Hilbert-Space vector process $\boldsymbol{x}_u$. Figure 2.51 illustrates this normalization that creates a new "white input" $\boldsymbol{\nu}$ such that $\boldsymbol{x}_u = R_{\boldsymbol{xx}}^{1/2} \cdot \boldsymbol{\nu}_u$ in each block diagonal element. The $R_{\boldsymbol{xx}}^{1/2}(u)$ (redefined from $R_{\boldsymbol{xx}}^{o/2}$) in (2.359) may use any square root for which it's pseudoinverse will produce $\varrho_{x,u}$ independent unit-energy inputs $\boldsymbol{\nu}_u$, $u = 1, ..., \varrho_{x,u}$.

$$R_{\boldsymbol{\nu\nu}} = \begin{bmatrix} R_{\boldsymbol{\nu\nu}}(U) & ... & 0 \\ \vdots & \ddots & \vdots \\ 0 & ... & R_{\boldsymbol{\nu\nu}}(1) \end{bmatrix} = I \quad . \tag{2.362}$$

Chapter 5 finds good $R_{\boldsymbol{xx}}(u)$ designs that appropriately excite primary and secondary components to acheive a given $\boldsymbol{b} \in \mathcal{C}(\boldsymbol{b})$. The individual unit-energy sub-user scalar (possibly complex) components are $\nu_{u,\ell}$, $\ell = 1, ..., \varrho_{x,u}$ and

$$\boldsymbol{\nu}_u = \begin{bmatrix} \nu_{u,\varrho_{x,u}} \\ \vdots \\ \nu_{u,1} \end{bmatrix} \quad . \tag{2.363}$$

Transmitter u's modulator $R_{\boldsymbol{xx}}^{1/2}(u)$ ensures $R_{\boldsymbol{xx}}(u)$ on the actual channel input $\boldsymbol{x}_u$. $\widetilde{H}_u$'s transformation in (2.361) also whitens the noise. So then the set of distinct codewords subsymbols formed from $\varrho_{x,u}$ values of average unit-energy $\nu_{u,\ell}$, per subsymbol map uniquely (and invertible with probability 1) to codewords $\boldsymbol{x}_u$ that have autocorrelation $R_{\boldsymbol{xx}}(u)$. The mapping is unique and invertible because any distinct vector $\boldsymbol{x}_u$ that exists (with nonzero probability) has a linear invertible mapping between

$\boldsymbol{x}_u$ and a corresponding $\boldsymbol{\nu}_u$, namely $R^{+/2}_{\boldsymbol{x}\boldsymbol{x}}$ or the unique pseudo-inverse of the selected square-root[82]. Any inversion ambiguity disappears by choosing $\boldsymbol{\nu}_u$ to have the pseudoinverse's unique $\|\boldsymbol{\nu}_u\|^2 = \|\boldsymbol{x}_u\|^2$ choice. By Lemma 2.3.5, there is no information loss. Clearly $\varrho_{x,u} \leq L_{x,u}$, or effectively the input $\boldsymbol{v}_u$ has autocorrelation $\begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}$ if $\boldsymbol{x}_u$ has $L_{x,u} > \varrho_{u,x}$ dimensions. Singular inputs can be useful generally in that they avoid energy transmission on poor channel modes (e.g., water-filling).

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 the above step's execution. It will be convenient to read the following passage as if $L_{x,u} = 1$ and $\varrho_{\widetilde{H}} = L_y = U = U'$, 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.

Canonical Forward/Backward MAC: Receiver processing for Steps 1 and 2's modified Gaussian MAC begins with a receiver matched-matrix (filter) $\widetilde{H}^*$ that (as always) retains all information:

$$\boldsymbol{y}' = \widetilde{H}^* \cdot \boldsymbol{y} = \widetilde{H}^* \cdot \left[\widetilde{H} \cdot \boldsymbol{x} + \boldsymbol{n}\right] = \underbrace{\widetilde{H}^* \cdot \widetilde{H}}_{R_f} \cdot \boldsymbol{\nu} + \underbrace{\widetilde{H}^* \cdot \boldsymbol{n}}_{\boldsymbol{n}'} \quad , \tag{2.364}$$

which however produces noise $\boldsymbol{n}'$ with autocorrelation $R_{\boldsymbol{n}'\boldsymbol{n}'} = R_f$. The matrix R_f is also the MAC's **canonical forward-channel matrix**. It may be tempting to Cholesky-factor R_f and 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 the best possible 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) with Cholesky facotrization

$$R_b^{-1} \triangleq R_f + I = G^* \cdot S_0 \cdot G \quad , \tag{2.365}$$

where[83] G is a $\mathscr{L}_x \times \mathscr{L}_x$ monic[84] upper-triangular matrix, and S_0 is a diagonal matrix of positive-real Cholesky factors. (Appendix D describes Cholesky Factorization.) The relation

$$2^{2 \cdot \overline{\mathcal{I}}(\boldsymbol{x};\boldsymbol{y})} = \mid R_b^{-1} \mid = \mid S_0 \mid \quad , \tag{2.366}$$

follows immediately.

MMSE MAC: Appendix D's MMSE estimator of $\boldsymbol{x}$ (where the relabeled $\boldsymbol{x}$ arises from $\widetilde{H}_u$'s Step 1 absorption of $R^{-1/2}_{\boldsymbol{x}\boldsymbol{x}}(u)$ that defines $R_{\boldsymbol{\nu}\boldsymbol{\nu}} = I$), given $\boldsymbol{y}'$ is

$$R_{\boldsymbol{\nu}\boldsymbol{y}'} \cdot R^{-1}_{\boldsymbol{y}'\boldsymbol{y}'} = R_f \cdot [R_f \cdot R_f + R_f]^{-1} = [R_f + I]^{-1} = R_b \quad , \tag{2.367}$$

so R_b is this MMSE estimator for $\boldsymbol{\nu}$. Also, R_b is always non-singular with nonsingular $R_{\boldsymbol{n}\boldsymbol{n}}$ and thus always has Cholesky factorization:

$$R_b = G^{-1} \cdot S_0^{-1} \cdot G^{-*} \quad . \tag{2.368}$$

Thus G and S_0 are also invertible. The receiver next forms (recalling that a linear combination's MMSE estimate is the same linear combination of the estimates)

$$\begin{aligned} \boldsymbol{y}'' &= S_0^{-1} \cdot G^{-*} \boldsymbol{y}' & (2.369)\\ &= S_0^{-1} \cdot G^{-*} \left[R_f \cdot \boldsymbol{\nu} + \boldsymbol{n}'\right] & (2.370)\\ &= S_0^{-1} \cdot G^{-*} \cdot \left[R_b^{-1} - I\right] \cdot \boldsymbol{\nu} + S_0^{-1} \cdot G^{-*} \cdot \boldsymbol{n}' \quad , & (2.371) \end{aligned}$$

[82] Specifically if the selected square root has SVD $R^{1/2}_{\boldsymbol{x}\boldsymbol{x}} = V_{\boldsymbol{x}} \cdot D_{\boldsymbol{x}} \cdot U_{\boldsymbol{x}}$, then the unique pseudoinverse square root is $V_{\boldsymbol{x}} \cdot D^{-1/2}_{\boldsymbol{x}} \cdot U_{\boldsymbol{x}}$ where the nonzero diagonal elements of $D_{\boldsymbol{x}}$ are replaced by their inverse positive square roots. This pseudoinverse choice also zeros null-space components.

[83] This G is not the linear finite-field generator, although the two have analogous functions and whence the common notation.

[84] Monic means 1's on the diagonal.

Further from (2.367)

$$\boldsymbol{\nu} = R_b \cdot \boldsymbol{y}' + \boldsymbol{e} \ , \tag{2.372}$$

so

$$\begin{aligned} \boldsymbol{y}' &= R_b^{-1} \cdot \boldsymbol{\nu} - R_b^{-1} \cdot \boldsymbol{e} && (2.373) \\ S_0^{-1} \cdot G^{-*} \boldsymbol{y}' &= G \cdot \boldsymbol{\nu} - G \cdot \boldsymbol{e} && (2.374) \\ \boldsymbol{y}'' &= G \cdot \boldsymbol{\nu} - \boldsymbol{e}' \ , && (2.375) \end{aligned}$$

where the MMSE error vector $\boldsymbol{e}' \triangleq G \cdot \boldsymbol{e}$ has diagonal autocorrelation

$$R_{\boldsymbol{e}'\boldsymbol{e}'} = S_0^{-1} \ . \tag{2.376}$$

The cascade of receiver matrix multiplies is (See Figure 2.52.) is the (unbiased) Mean-Square Whitened Matched Filter (**MSWMF**)

$$\boldsymbol{y}''' = \underbrace{(S_0^{-1} - I) \cdot G^{-*} \cdot H^* \cdot R_{\boldsymbol{nn}}^{-1}}_{\mathcal{W}} \cdot \boldsymbol{y} \ . \tag{2.377}$$

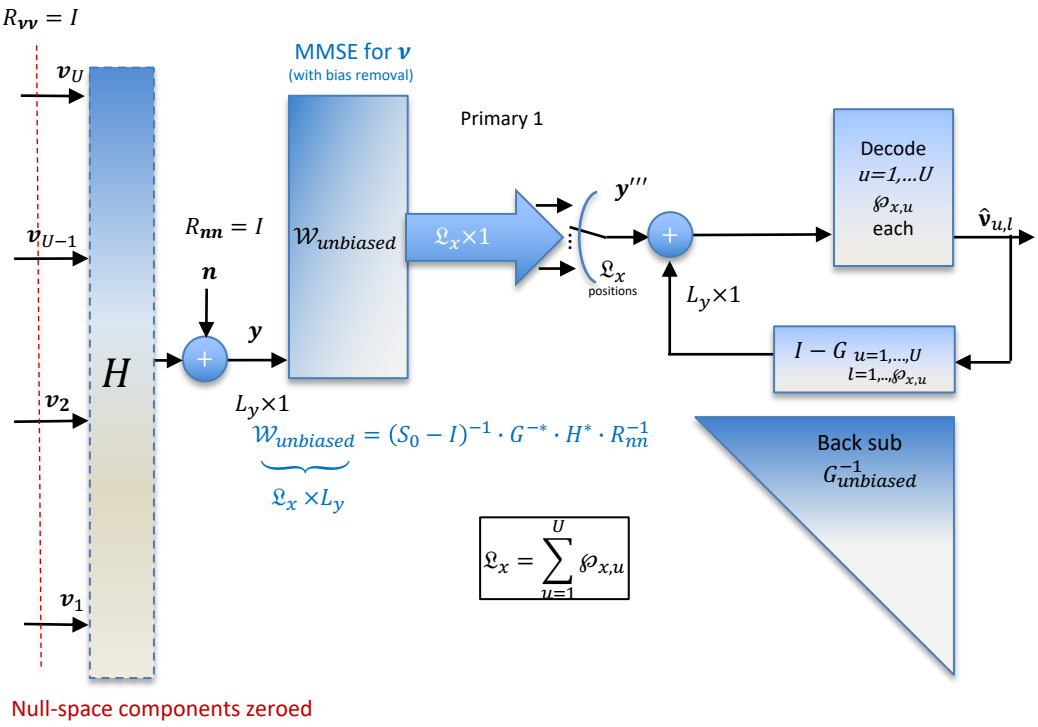


Figure 2.52: Vector MAC Design Illustration.

Correct Successive Decoding: The monic upper-triangular matrix G has structure

$$G = \begin{bmatrix} 1 & g_{U',U'-1} & \cdots & g_{U',2} & g_{U',1} \\ 0 & 1 & \cdots & g_{U'-1,2} & g_{U'-1,1} \\ \vdots & \ddots & \cdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & g_{2,1} \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \tag{2.378}$$

The receiver processes a subsymbol $\boldsymbol{y}''$ by its user components successively[85] from user 1 up through user U:

$$\begin{aligned} \hat{\boldsymbol{\nu}}_1 &= \text{decision}\,(\boldsymbol{y}_1'') \\ \hat{\boldsymbol{\nu}}_2 &= \text{decision}\,(\boldsymbol{y}_2'' - g_{2,1}\cdot\hat{\boldsymbol{\nu}}_1) \\ &\vdots \quad \vdots \quad \vdots \\ \hat{\boldsymbol{\nu}}_{U'} &= \text{decision}\left(\boldsymbol{y}_U'' - \sum_{i=1}^{U-1} g_{U,i}\cdot\hat{\boldsymbol{\nu}}_i\right) \quad . \end{aligned} \tag{2.379}$$

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 $\hat{\boldsymbol{\nu}}_1 = \boldsymbol{\nu}_1$. This "decision" operation may well introduce decoder delay to process a long symbol before earlier users' subsymbols become available for later-user decoders' use. This $\boldsymbol{\nu}_1$ now multiplies the red $g_{2,1}$ component in Equation (2.378)'s second row up from bottom and disappears through the operation $\boldsymbol{y}_2'' - g_{2,1}\cdot\boldsymbol{\nu}_1 = \boldsymbol{\nu}_2 + \boldsymbol{e}'$, 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 (2.375) that is (are diagonal elements of) $S_0(u)$ (recalling $R_{\boldsymbol{\nu}\boldsymbol{\nu}}$ has been normalized to the identity $R_{\boldsymbol{\nu}\boldsymbol{\nu}} = I$ in the modified channel definition's Step 1 $\widetilde{H}$ formation).

SNR's and User bits/subsymbol: Each $S_0(u)$ is a diagonal $\mathscr{L}_x \times \mathscr{L}_x$ matrix with diagonal SNR elements

$$SNR_{mmse,u,\ell} = S_0(u,\ell) \quad . \tag{2.380}$$

All the above signal processing preserves the mutual information, which is then

$$\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = \log_2(|\underbrace{\widetilde{H}^*\widetilde{H} + I}_{R_b^{-1}}|) \tag{2.381}$$

$$= \log_2 |\, S_0 \,| \tag{2.382}$$

$$= \log_2 \left\{ \underbrace{\prod_{u=1}^{U} \underbrace{\prod_{\ell=1}^{L_{x,u}} SNR_{mmse,u,\ell}}_{\text{user components}}}_{\text{users}} \right\} \text{ bits / complex symbol} \quad . \tag{2.383}$$

Thus, this processing creates a set of individual decoders whose $SNR_{mmse,i,\ell}$'s, and corresponding individual subsymbol mutual information quantities $\mathcal{I}_{u,\ell} \triangleq \log_2(SNR_{mmse,u,\ell})$, all sum to the MAC's mutual information. As per Section 2.3.6 and Appendix D, the individual residual outputs at each decoding stage of $\boldsymbol{y}_u''' \triangleq \boldsymbol{y}_u'' - \sum_{i=1}^{u-1} g_{u,i}\cdot\hat{\boldsymbol{\nu}}_i$ have residual bias and so

$$\mathbb{E}\,[\boldsymbol{y}_u'''/\boldsymbol{\nu}_u] = \left\{I_{L_y} - S_0^{-1}(u)\right\}\cdot\boldsymbol{\nu}_u \quad . \tag{2.384}$$

The reduction in energy accrues to the MMSE-receiver-processing signal reduction, along with the Gaussian noise $\boldsymbol{n}$, to create a smaller error $\boldsymbol{e}_u$. Thus $SNR_{mmse,u} = |S_0(u)|$ is slightly higher than the unbiased (scaled) channel SNR and indeed each dimension actually has unbiased $SNR_{u,\ell} = SNR_{mmse,u,\ell} - 1$. The receiver removes bias according to (See Figure 2.52)

$$\boldsymbol{y}''' = \underbrace{(S_0 - I)^{-1}\cdot G^{-*}\cdot H^*\cdot R_{\boldsymbol{n}\boldsymbol{n}}^{-1}}_{\mathcal{W}}\cdot\boldsymbol{y} \quad , \tag{2.385}$$

[85] The operation of $\hat{\boldsymbol{\nu}}_u$ corresponds to a decision that finds the subsymbol value for user u from $\boldsymbol{y}_u$ 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 exactly with the correct subsymbol in user u's codeword; then user 2's values could be found for each subsymbol $\boldsymbol{\nu}_1$ already known infinite delay and $\Gamma = 0$ dB code.

or equivalently

$$W_{unb} = (S_0 - I)^{-1} \cdot S_0 \cdot W \quad . \tag{2.386}$$

The bias can be of consequence with codes that have $\Gamma > 0$ dB, and best design removes it. Nonetheless, the overall mutual information remains the same as

$$\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = \log_2(\mid \widetilde{H}^* \widetilde{H} + I \mid) = \log_2 \mid S_0 \mid = \log_2 \left\{ \prod_{u=1}^{U} \prod_{\ell=1}^{L_y} (1 + SNR_{u,\ell}) \right\} \text{ bits / complex symbol} \quad . \tag{2.387}$$

The MMSE/mutual-information-chain-rule solution has residual components of all other users when $L_y > 1$, because the MMSE criterion trades some crosstalk for noise amplitude reduction in a way that improves SNR. The MMSE receiver removes (reduces) lower-index crosstalk components in the successive-decoding/back-substitution process. The higher-index components are part of the error $\boldsymbol{e}$. The receiver processing produces a signal component (ignoring noise $\boldsymbol{n}$ components)

$$\begin{aligned} \text{signal part of } \boldsymbol{y}'' &= S_0 \cdot \left[S_0^{-1} \cdot G^{-*} \cdot R_f \cdot \boldsymbol{\nu} \right] && (2.388) \\ &= S_0^{-1} \cdot G^{-*} \left[R_b^{-1} - I \right] \boldsymbol{\nu} && (2.389) \\ &= G \cdot \boldsymbol{\nu} - S_0^{-1} \cdot G^{-*} \cdot \boldsymbol{\nu} \quad . && (2.390) \end{aligned}$$

Bias removal in the back-substitution process based on $\boldsymbol{\nu}$ produces an unbiased successive decoder that uses

$$\mathcal{G} = I + [S_0 - I]^{-1} \cdot S_0 \cdot [G - I] \quad . \tag{2.391}$$

The processing also produces a lower-triangular user-error component that is

$$- S_0^{-1} \cdot G^{-*} \cdot \boldsymbol{\nu}. \quad . \tag{2.392}$$

Thus, unlike a zero-forcing solution, the MMSE will have some crosstalk from other users in every error except the last (upper component) that has only bias from itself (so simple scaling renders this one component to have unbiased MMSE and ZF the same, while all other users experience crosstalk in the MMSE approach from later users in the given order), 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.

The mu_mac.m matlab program: Appendix G's mu_mac.m program finds the MMSE matrix filters (forward linear and backward decision-feedback/successive cancellation) for a matrix AWGN MAC. The program accepts the (noise-whitened) H matrix as a succession of user submatrices from $u = U, ..., 1$. Thus, the matrix input H is $L_y \times \mathscr{L}_x$ for mu_mac.m. The corresponding input modulator (square-root-autocorrelation) matrices are in a block-diagonal matrix A (so that the product $H \cdot A$ represents the input transfers from each independent input component to the output. The variable Lxu is a $1 \times U$ vector that successively lists the number of user-input dimensions, so for instance Lxu= [1 2] corresponds to user $U = 2$ with one antenna and user 1 with two antennas, so the channel is 3×3. The parameter cb indicates the channel is complex baseband (cb =1) or real baseband (cb = 2). The outputs are the user bit vector b ($\boldsymbol{b}$), the unbiased feedback matrix GU ($\mathcal{G}$, the unbiased feedforward matrix W_{unb}, the diagonal SNR matrix S0 (S_0), and the unbiased MS-WMF, MSWMFU ($\mathcal{W}$).

```
function [b, GU, WU, S0, MSWMFU] = mu_mac(H, A, Lxu , cb)
-------------------------------------------------------------------------
 Per-tonal (temporal dimension) multiuser mac receiver and per-user bits

 Inputs: H, A , Uind , cb
 Outputs: b, GU, WU, S0, MSWMFU
```

```
definitions

H: noise-whitened channel matrix [HU ... H1] Ly x sum-Lxu
A: Block Diag sq-root sum-Lxu x sum-Lxu discrete modulators,
   blkdiag([AU ... A1]);  The Au entries derive from each MAC user's
   Lxu x Lxu input autocorrelation matrix, where the trace of each such
   autocorrelation matrix is user u's energy/symbol. This is per-tone.

Lxu: # of dimensions for each user U ... 1 in 1 x U row vector
cb: = 1 if complex baseband or 2 if real baseband channel

GU: unbiased feedback matrix sum-Lxu x sum-Lxu
WU: unbiased feedforward linear equalizer sum-Lxu x sum-Lxu
SO: sub-channel channel gains sum-Lxu x sum-Lxu
MSWMFU: unbiased mean-squared whitened matched filter, sum-Lxu x Ly
b - user u's bits/symbol 1 x U
   the user should recompute SNR and b if there is a cyclic prefix

-------------------------------------------------------------------------
```

The program mu_mac.m's comment on cyclic prefix relates to the exponent on $S_0^{-\mathscr{L}_x}$, where nominal use assumes a MIMO channel with $\mathscr{L}_x = \sum_u L_{x,u}$, but some future uses may need to reduce these by an effective "excess-bandwidth" quantity that Chapters 4 and 5 later address.

Gaussian's Optimality via Chain Rule: Any order's chain-rule decomposition of $\mathcal{I}(\boldsymbol{x};\boldsymbol{y})$ has the last decoded user with all other users given and cancelled, and as in Appendix D and Section 2.3 corresponds to MMSE estimation with Gaussian noise. This last term corresponds to a Gaussian conditional distribution. Thus this last-decoded user sees a single-user crosstalk-free channel with only the Gaussian nosie. 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 (including some uncancelled user in unbiased case, but that remains Gaussian), and thus the best distribution for the second-last user is also Gaussian. Inductively, even with bias, all users' best (marginal) distributions remain Gaussian, consistent with earlier findings. The Gaussian MAC input autocorrelation matrix $R_{\boldsymbol{xx}}$ is always block diagonal. The above singularity-removal step forces this to I, but energy constraints apply as originally given on $trace\{R_{\boldsymbol{xx}}(u)\}$ and correspond to energy/symbol $\mathcal{E}_{x,u} \cdot \varrho_{x,u}$ on the normalized inputs $\boldsymbol{\nu}_u$, $trace\{R_{\boldsymbol{\nu\nu}}(u)\} = \varrho_{x,u}$. There are two consequent maximum rate-sum situations of interest:

Energy-Vector MAC The individual energy-constraints $trace\{R_{\boldsymbol{xx}}(u)\} \leq \mathcal{E}_u$, or equivalently $\boldsymbol{\mathcal{E}} \preceq \boldsymbol{\mathcal{E}_x}$, has the same rate sum for any particular $R_{\boldsymbol{xx}}$ and order $\boldsymbol{\pi}$, and this rate sum is $\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = \log_2 | \, |\widetilde{H} \cdot \widetilde{H}^* + I \, |$ bits/subsymbol.

Energy-Sum MAC The total energy-constraint of $trace\{R_{\boldsymbol{xx}}\} \leq \mathcal{E}_{\boldsymbol{x}}$ will have the same maximum rate sum for all $U!$ orders. The rate sum is not maximum over $trace\{R_{\boldsymbol{xx}}\} \leq \mathcal{E}_{\boldsymbol{x}}$ if any $\boldsymbol{x}_u$ has a nonzero energy component in $\mathcal{N}_{H_u}$.

EXAMPLE 2.7.2 *[Full-rank channel]* A 2×2 real baseband MAC with $U = 2$ has noise-whitened channel matrix and unit energy independent inputs.

$$\widetilde{H} = \begin{bmatrix} 5 & 2 \\ 3 & 1 \end{bmatrix} \quad . \tag{2.393}$$

The mu_mac program allows calculation of the quantities of interest:

```
H=[5 2 ; 3 1];
[b, GU, WU, S0, MSWMFU] = mu_mac(H, eye(2), [1 1] , 2)

b =
    2.5646    0.1141
GU =
    1.0000    0.3824
         0    1.0000
WU =
    0.0294         0
   -2.1667    5.8333
S0 =
   35.0000         0
         0    1.1714
MSWMFU =
    0.1471    0.0882
    0.8333   -0.6667
```

The program internally implements the following commands that may be also of interest:

```
>> 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 b_2 and smaller b_1 for this chosen $R_{\boldsymbol{xx}} = I$. The matlab results below illustrate several of Figure 2.52's receiver-filtering Gaussian-MMSE MAC concepts. Figure 2.52 shows the post-matched filter processing, MMSE with biases removed, and WU (which is triangular above as expected). The unbiased filter/matched-filter combination is MSWMFU acting on the 2×2 channel (non triangular).

```
>> MSWMFu*H =
    1.0000    0.3824
    2.1667    1.0000
```

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 represents this residual error, 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 for this example's small SNR.

The zero-forcing[86] (ZF) solution helps contrast receiver processing. The ZF solution can be implemented by Cholesky factorization of R_f or also directly QR factorization of $\widetilde{H} = Q_{ZF} \cdot S_{ZF} \cdot G_{ZF}$ (S_{ZF} is positive real diagonal matrix, G_{ZF} is monic upper triangular, and Q_{ZF} is unitary, and this can be computed with matlab's qr command) , as per Figure 2.53. Figure 2.53 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 (2.379). This is the straightforward implementation of successive decoding, but it is not ML in 2 dimensions. Indeed the second dimension has a lower SNR and bit rate than the canonical MMSE implementation displayed above. While the 1-to-1 processing produces a set of parallel independent channels, this set has lower SNR than is possible and indeed does not minimize the ML sum in (2.150). Figure 2.53'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, nor even canonically equivalent, in the multi-dimensional zero-forcing case.

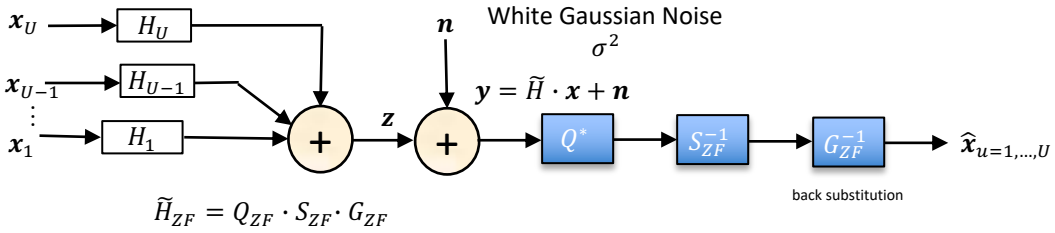


Figure 2.53: Zero-Forcing MAC Receiver.

```
>> [Q,R]=qr(H) =
Q=
   -0.8575   -0.5145
   -0.5145    0.8575
R =
   -5.8310   -2.2295
         0   -0.1715
>> Q*R =
    5.0000    2.0000
    3.0000    1.0000 (checks)
>> Szf=diag(diag(R)) =
   -5.8310         0
         0   -0.1715
>> Gzf=inv(Szf)*R =
    1.0000    0.3824
         0    1.0000
>> det(Szf*Szf) =    1.0000
>> W=inv(Szf)*Q' =
    0.1471    0.0882
    3.0000   -5.0000
>> W*H =
```

```
    1.0000    0.3824
         0    1.0000.  (checks = Gzf)
>> bzf = 0.5*log2(diag(eye(2)+Szf*Szf)) =
    2.5646
    0.020
>> b =
    2.5646
    0.1141
>> sum(b) = 2.6787
>> b - bzf =
         0
    0.0932
    (MMSE and ZF same in user 2, but not user 1.)
 >> Wzf=inv(Szf)*Q' =
    0.1471    0.0882
    3.0000   -5.0000
>> Wzf*H =
    1.0000    0.3824
         0    1.0000
```

In this zero-forcing example, the highest priority user has the same (optimum) data rate, but the second user has reduced data rate and SNR with respect to MMSE. The use of a full-rank H 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 next example shows the first user has the same performance, but not the remaining user(s).

EXAMPLE 2.7.3 [degraded-rank channel] This MAC's expansion to a 2×3 channel where user 3 may have also (now extra) energy $\mathcal{E}_3 = 1$ would have the same data rates on users 1 and 2 if user 3 is cancelled first. So, the sum rate must increase.

```
>> H=[5 2 1
3 1 1];
[b, GU, WU, S0, MSWMFU] = mu_mac(H, eye(3), [1 1 1] , 2)

b =    2.5646    0.1141    0.1137
GU =
    1.0000    0.3824    0.2353
         0    1.0000    0.1667
         0         0    1.0000
WU =
    0.0294         0         0
   -2.1667    5.8333         0
   -1.2857   -0.1429    5.8571
S0 =
   35.0000         0         0
         0    1.1714         0
         0         0    1.1707
MSWMFU =
    0.1471    0.0882
    0.8333   -0.6667
```

```
   -0.8571    1.8571
>> sum(b) =    2.7925

>> MSWMFU*H =
    1.0000    0.3824    0.2353
    2.1667    1.0000    0.1667
    1.2857    0.1429    1.0000
>> SNR = 10*log10(diag(S0)) =   (in dB)
   15.4407
    0.6872
    0.6846
```

Indeed, the sum rate increased with 3 units of energy from 2.7 to .9 roughly. Users 3 and 2 are the same as Example 2.7.2's users 2 and 1 respectively and have the same data rates, while user 3 adds roughly .11 more.

Example 2.7.3 continued:
If the channel reduces input energy to the original 2 units without allocating this energy to the original two users, say evenly over the 3 users with an energy-vector MAC, then the data rate reduces further

```
>> [b, GU, WU, S0, MSWMFU] = mu_mac(H, (2/3)*eye(3), [1 1 1] , 2)

b =    2.0050    0.1009    0.0696
GU =
    1.0000    0.3824    0.2353
         0    1.0000    0.3878
         0         0    1.0000
WU =
    0.0662         0         0
   -2.3878    6.6582         0
   -2.0000   -0.5000    9.8750
S0 =
   16.1111         0         0
         0    1.1502         0
         0         0    1.1013
MSWMFU =
    0.2206    0.1324
    0.9184   -0.3367
   -0.7500    2.2500
>> sum(b) =    2.1755
```

Both user 3 and user 2 have lost data rate with respect to the case when only they are excited.

The above example illustrates that while more dimensions is usually helpful, the distribution of energy to them becomes increasingly important relative to adding new users on those antennas.

2.7.2.3 Achievable Regions with Non-Zero Gaps

Figure 2.54 illustrates a gap-reduced capacity region. Basically this convex region has bound $\mathscr{C}(\boldsymbol{b})-\gamma_b\odot\mathbf{1}$. The designer computes the capacity region and then removes an equal-element constant vector (γ_b in each position) from it to trace the gap-reduced capacity region.

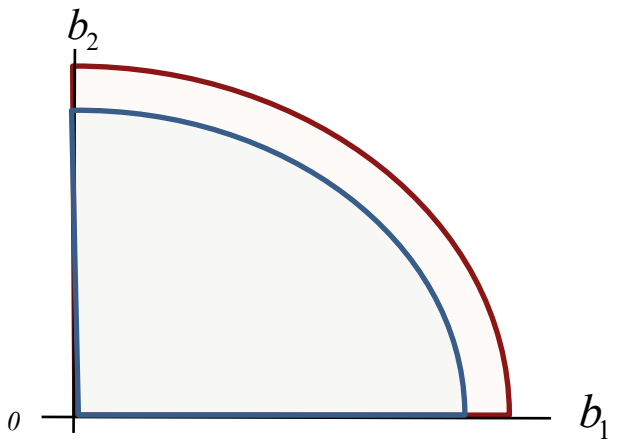


Figure 2.54: Gap-Reduced Capacity Region.

2.7.3 Maximum rate-sum calculation for the matrix AWGN

The matrix-AWGN MAC rate sum is always

$$\tilde{b}=\sum_{u=1}^{U}\tilde{b}_u\le\mathcal{I}(\boldsymbol{x};\boldsymbol{y})=\log_2\frac{|H\cdot R_{\boldsymbol{xx}}\cdot H^*+R_{\boldsymbol{nn}}|}{|R_{\boldsymbol{nn}}|}\quad . \tag{2.394}$$

The maximum over input $R_{\boldsymbol{xx}}$ corresponds to maximizing the numerator inside the log or equivalently

$$\max_{\{R_{\boldsymbol{xx}}(u)\}}|H_u\cdot R_{\boldsymbol{xx}}(u)\cdot H_u^*+\underbrace{\sum_{i\neq u}H_i\cdot R_{\boldsymbol{xx}}(i)\cdot H_i^*+R_{\boldsymbol{nn}}}_{R_{noise}(u)}| \tag{2.395}$$

The maximization separates into U optimizations where each user views all the rest as "noise," as indicated by $R_{noise}(u)$ in (2.395), specifically

$$R_{noise}(u)\triangleq\sum_{i\neq u}H_i\cdot R_{\boldsymbol{xx}}(i)\cdot H_i^*+R_{\boldsymbol{nn}}\quad . \tag{2.396}$$

Each optimal $R_{\boldsymbol{xx}}(u)$ satisfies the water-filling criteria for the vector-coded channel $R_{noise}^{-1/2}(u)\cdot H_u$. 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 2.7.2 **[Optimality of SWF for the matrix-AWGN MAC]** *Any autocorrelation matrix set $\{R_{\boldsymbol{xx}}(u)\}$ that simultaneously water-fills, with respect to all other users as noise, maximizes the matrix-AWGN MAC rate sum.*

proof: See the preceding paragraph and note that the optimization problem is separable for each of the $R_{\boldsymbol{xx}}(u)$ as a 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

$$R_{noise}^{-1/2}(u) \cdot H_u = F_u \cdot \Lambda_u \cdot M_u^* \quad . \tag{2.397}$$

Then the energies are assigned according to (where $g_{u,\ell} = \lambda_{u,\ell}^2$)

$$\mathcal{E}_{u,\ell} + \frac{1}{g_{u,\ell}} = \text{constant over } \ell \text{ for each } u \ , \tag{2.398}$$

such that

$$\sum_{\ell=1}^{L_x} \mathcal{E}_{u,\ell} = \mathcal{E}_u \quad , \tag{2.399}$$

and

$$\mathcal{E}_{u,\ell} \geq 0 \quad . \tag{2.400}$$

The u^{th} user's input is

$$\boldsymbol{x}_u = M_u \cdot \boldsymbol{X}_u \quad , \tag{2.401}$$

where $\boldsymbol{X}_u$'s elements are independent and each has energy $\mathcal{E}_{u,\ell}$. These energies then characterize the components. If the constraints on $\mathcal{E}_u$ become a single total-energy constraint $\mathcal{E}_{\boldsymbol{x}} = \sum_{u,\ell} \mathcal{E}_{u,\ell}$, then the nonzero $\mathcal{E}_{u,\ell} > 0$ identify the primary components.

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 MAC users as noise, will converge.[87] Figure 2.55 illustrates the basic 2-user capacity region for two users and also illustrates iterative water-filling (IW).

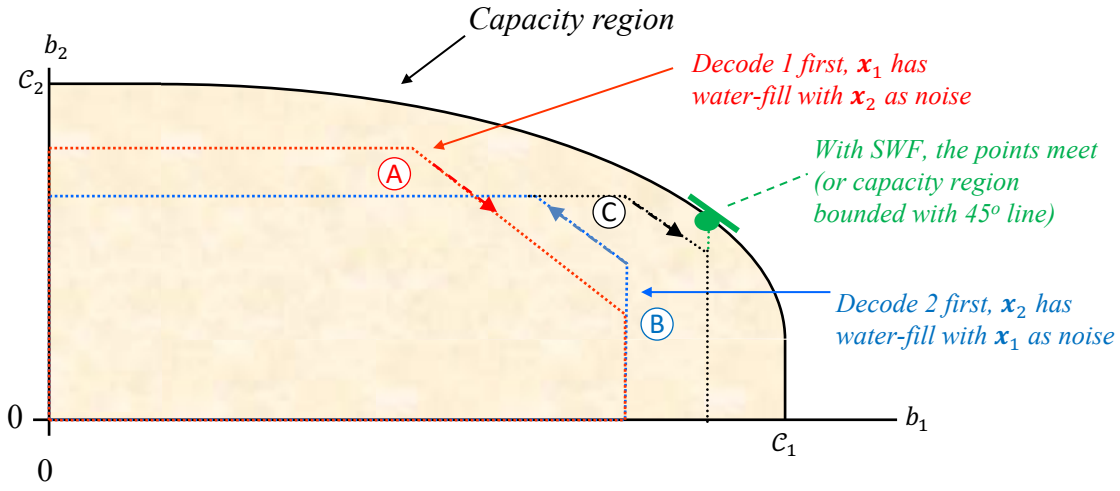


Figure 2.55: Illustration that SWF solution achieves MA channel maximum rate sum.

[87] Since the SWF condition corresponds to a convex optimization, and descent algorithm can be used and will thus converge, but the objective here is a more heuristic explanation.

IW Convergence: The two users $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ contribute to the common MAC output $\boldsymbol{y}$. Then, by the chain rule,

$$\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) = I(\boldsymbol{x}_1;\boldsymbol{y}) + I(\boldsymbol{x}_2;\boldsymbol{y}/\boldsymbol{x}_1) = I(\boldsymbol{x}_2;\boldsymbol{y}) + I(\boldsymbol{x}_1;\boldsymbol{y}/\boldsymbol{x}_2) \quad , \tag{2.402}$$

and there are thus two ways to achieve the same rate sum $I(\boldsymbol{x};\boldsymbol{y})$. Since either order can be used, the receiver can consider $\boldsymbol{x}_1$ first, and water-fill for user $\boldsymbol{x}_1$ first with user $\boldsymbol{x}_2$ as noise to obtain point (A). 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 $R_{\boldsymbol{xx}}(1)$, user 2 now water fills. Since the receiver decodes user 1 last and does not alter $R_{\boldsymbol{xx}}(1)$, 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 blue pentagon. Since user 2 now maintains the same $R_{\boldsymbol{xx}}(2)$, then user 2's rate b_2 does not change. However, now again reversing order, user 1's rate increases to point (C) (in black) 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^o line that bounds the capacity region $\mathscr{C}(\boldsymbol{b})$. Thus, the result is established for $U = 2$. By setting a single user $\boldsymbol{b}$ to correspond to the rate sum $\bar{b}_1 + \bar{b}_2$ and introducing a 3rd user $\boldsymbol{x}_3$, the concept can be repeated with $b_2 + b_1$ replacing b_1 and b_3 replacing b_2. Thus by induction, the iteration of water-filling must converge to a rate-sum-maximizing point and thus SWF.

2.7.4 Frequency-Indexed 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 frequency-indexed Gaussian MAC's maximum sum rate.

2.7.4.1 Capacity Regions for the Scalar Filtered Gaussian MAC

Subsection 2.7.1's MAC bounds each user's energy by $\mathcal{E}_u$. This bound generalizes for linear time-invariant channels with pulse response set $\{h_u(t)\}$ and corresponding Fourier Transform set $\{H_u(f)\}$. $\mathcal{E}_u$ then translates 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 period and starting time 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 $U \times 1$ $\boldsymbol{S}_x(f)$. Any user's spectrum can simultaneously occur with any of the other users' possible spectra, leading to a large range of possibilities for construction of $\mathcal{A}(\boldsymbol{b}, \boldsymbol{S}_x(f))$ and $\mathcal{C}(\boldsymbol{b})$. Each and every different power spectral density vector $\boldsymbol{S}_x(f)$ creates a possible Gaussian input distribution that contributes to $\mathcal{C}(\boldsymbol{b})$'s convex-hull construction.

Figure 2.56 illustrates $\mathcal{C}(\boldsymbol{b})$'s construction via such a convex-hull/union of pentagon regions for $U = 2$. For a specific choice power spectral density choice $\boldsymbol{S}_x(f)$, $\mathcal{A}(\boldsymbol{b}, \boldsymbol{S}_x(f))$ is just one pentagon. Different $\boldsymbol{S}_x(f)$ 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 $\mathcal{C}(\boldsymbol{b})$ boundary or interior point may correspond to a different $\boldsymbol{S}_x(f)$. 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.[88] Figure 2.56's capacities $\mathcal{C}_1$, or $\mathcal{C}_2 > \mathcal{C}_1$, with the other user zeroed, limit the maximum data rate for that user – the capacity-region boundary may be a straight line (horizontal or vertical lines for $\mathcal{C}_1$ and $\mathcal{C}_2$ in Figure 2.56 respectively) at these $\mathcal{C}_1$ and $\mathcal{C}_2$ points. Usually, the capacity region curves gradually to these points for practical channels' large dimensionality (many frequencies equivalently, and there are infinitely many

[88] 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.

in ideal multi-tone) and there are no such straight lines (unlike the simpler pentagons earlier). These straight lines occur only on channels with a user who can use the entire band available or more (which is usually caused by under sampling in modern transmission systems and practical channels, or possibly some wireless regulatory spectral limits).

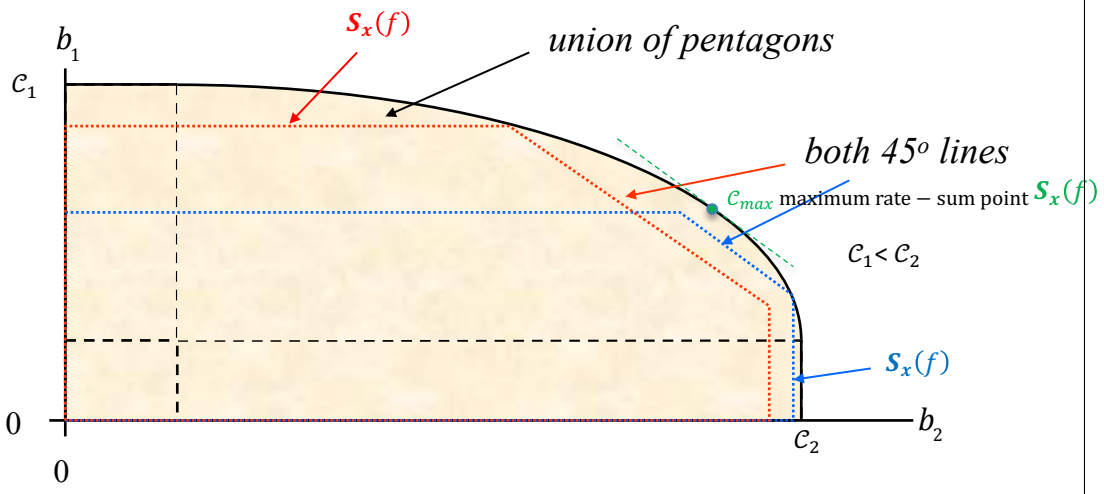


Figure 2.56: 2-user AWGN multiple-access capacity region.

The green line with slope -1 is the rate-sum line and is tangent to $C(\boldsymbol{b})$ at the maximum rate sum $\mathcal{C}_{max}$. While there are infinite dimensions and time, each frequency corresponds to a scalar MAC channel. At each such frequency, there is a primary-user component and a secondary-user component 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 $\frac{\mathcal{N}_0}{2}$). This could be a different user as largest at different frequencies. Effectively, each frequency has a single component, one for each user. Only the larger component can be primary. There is a rate sum in the absence of individual power constraints so that only a total energy constraint $\int S_{x_1}(f)df + \int S_{x_2}(f) \leq \mathcal{E}_1 + \mathcal{E}_2 = \mathcal{E}_{\boldsymbol{x}}$ applies, which then would place all energy on the primary sub-user component 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. However, it is only necessarily optimum in the SISO case where $L_{x,u} = L_y = 1$ and there is only a sum-energy constraint.

Any channel's rate-region calculation (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 $p_{\boldsymbol{x}}$. 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.

2.7.4.2 Simultaneous Water-Filling for the Scalar Filtered Gaussian MAC

Figure 2.57 illustrates this continuous time/frequency system for two users. Section 2.5's limiting aggregate MT sum rate for all users with filters $H_u(f)$, input power spectral densities $S_{x,u}(f)$, and noise

power spectral densities $S_n(f)$ is

$$\bar{b} = \sum_{u=1}^{U} \bar{b}_u \leq \overline{\mathcal{I}}(\boldsymbol{x};\boldsymbol{y}) = \int_{-\infty}^{\infty} \frac{1}{2} \cdot \log_2 \left[1 + \frac{\sum_{u=1}^{U} S_{x,u}(f) \cdot |H_u(f)|^2}{S_n(f)} \right] df \quad , \tag{2.403}$$

where the bits/subsymbol sum becomes an integral over all the infinitesimally 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

$$S_{x,u}(f) + \frac{\sigma^2 + \sum_{i \neq u} S_{x,i}(f) \cdot |H_i(f)|^2}{|H_u(f)|^2} = \lambda_u \quad . \tag{2.404}$$

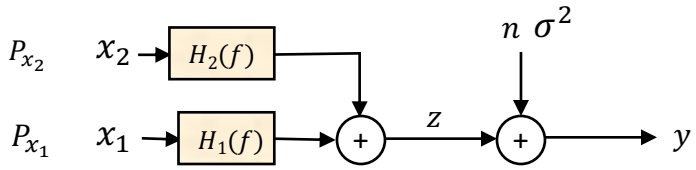


Figure 2.57: 2-user AWGN channel with ISI.

Since the original function's integrand was convex and all the constraints are convex (linear inequality), then a solution exists. Each instance of (2.404) for $u = 1, ..., U$ is water-filling with all the other users viewed as "noise." This leads to the following SWF Lemma extension:

Lemma 2.7.4 (MT Simultaneous Water-Filling Optimum) *The maximum rate sum* $\bar{b} = \sum_{\boldsymbol{u}} \bar{b}_{u \in \boldsymbol{u}}$ *for the scalar* ($L_y = L_x = 1$, $\boldsymbol{y} = y$, $\boldsymbol{x}_u = x_u$) *continuous time-frequency Gaussian MAC with* u^{th} *user's input power spectral density* $S_{x,u}(f)$ *satisfies*

$$S_{x,u}(f) + \frac{\sigma_u^2(f)}{|H_u(f)|^2} = \lambda_u \tag{2.405}$$

where

$$\sigma_u^2(f) = \sigma^2 + \sum_{i \neq u} S_{x,i}(f) \cdot |H_i(f)|^2 \tag{2.406}$$

with

$$S_{x,u}(f) \geq 0 \quad , \tag{2.407}$$

and

$$\int_{-\infty}^{\infty} S_{x,u}(f) \cdot df = P_u \quad . \tag{2.408}$$

$H_u(f)$ *is the Fourier transform of the* u^{th} *channel impulse response, and* P_u *is the* U^{th} *user's power constraint.* **proof:** *See above paragraph.*

Figure 2.57 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 2.58 shows with $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_{Total}$ all being water-filling capacities for the multiple-access channel. Three simultaneous water-filling spectra situations appear in Figure 2.58. 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.

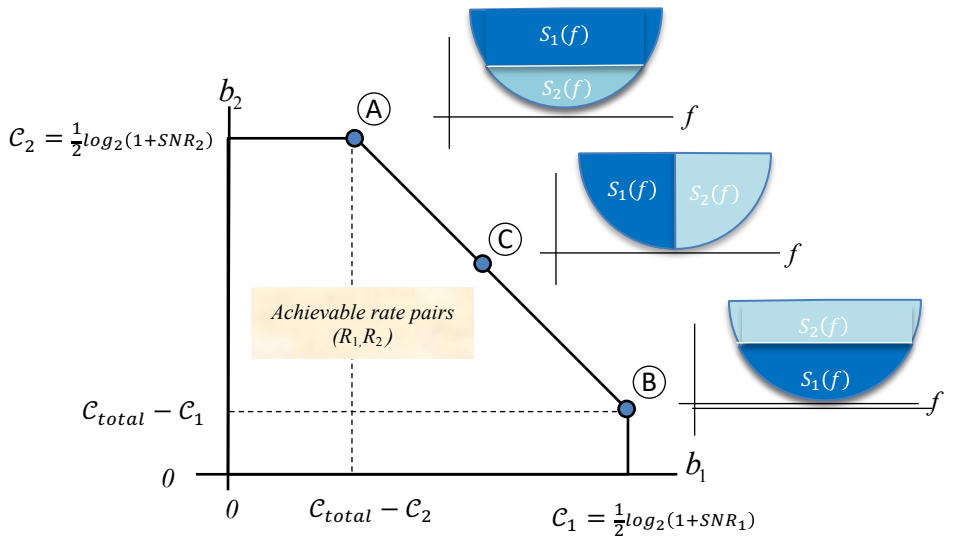


Figure 2.58: 2-user AWGN capacity region for $P_1(D) = P_2(D)$ case.

Figure 2.58 illustrates the corresponding water-fill spectra for the 3 different situations, (A), (B), and (C). The two corner points need not correspond to the use of a water-fill spectra for one of the two users. However, at point (A), user 1 has a WF spectra when user 2 is viewed as noise, while user 2 has WF spectra as if user 1 were zero (i.e., removed by successive decoding). The corresponding opposite is true for point (B). Both points (A) and (B) also satisfy SWF, as is evident in the power-spectral-density diagrams in Figure 2.58. Points in between can correspond to frequency-division or any other division of energy in the total water-fill region that satisfies each of the individual water-filling constraints also. Each such division corresponds to a different pair of rates with $R_1 + R_2 = C_{Total}$ (R_1, R_2) along the slanted boundary. At least one point corresponds to non-overlapping spectra (frequency-division multiplexing or FDM) if the channel's output and imputs are one-dimensional ($L_y = L_{x,u} = 1$). This FDM point clearly satisfies SWF individually as each user is zero in the others band. At this FDM 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 often uses "dimension-sharing or vertex-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 unitary (energy and volume preserving). Chapter 5 investigates more deeply this important concept of simultaneous water-filling.

Simultaneous Water-Filling Examples

EXAMPLE 2.7.4 (Example 2.7.1 revisited) Figure 2.47 shows the capacity region with $L_{x,u} = L_y = N = 1$ and $U = 2$. Example 2.7.1 investigated doubling the subsymbol rate and showed an enlarged capacity region. The time-sharing for the same symbol rate effectively can occur over an infinite period because capacity-achieving codes have infinite codeword length.

To extend this channel's fixed-subsymbol-rate analysis, the designer could let the frequency index $N \to \infty$ while keeping $L_x = L_y = 1$ and $\widetilde{T} = 1$ (so $T = \bar{N}$ and $W = 1$). With this infinite $\bar{N}$, Figure 2.59 shows 3 of an infinite number of simultaneous water-filling solutions possible. Solution (a) corresponds to both users flat over the entire band and use of a successive decoder at one of the vertices. Solution (b) corresponds to an FDMA solution

that is also SWF and has 41% of the bandwidth allocated to user 1 and remaining 59% of the bandwidth allocated to user 2. Both are flat over their ranges, and clearly both satisfy SWF criteria. The FDM case's decoder is 2 independent decoders (no feedback or successive decoding necessary). Solution (c) corresponds to a mixture of the two that is part FDMA and part sharing of bandwidth. Because $\mathscr{C}(\boldsymbol{b})$ is a non-zero length flat line with slope -1, several SWF solutions are possible, each corresponding to a point on this line. The representative decoder structure appears below each spectral choice.

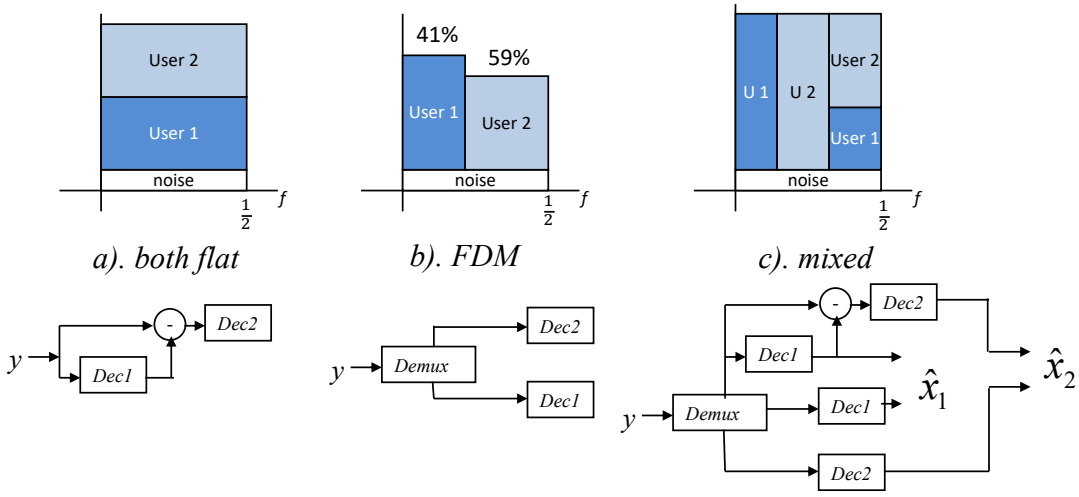


Figure 2.59: Example SWF spectra for Example 2.7.4.

The 59% was found from the zero-energy point of the function below in terms of the allocation of bandwidth fraction $1-w$ to one user and w to the other.

```
>> myfun = @(w) 0.5*w*log2(1+6400/w)+(1-w)*0.5*log2(1+3600/(1-w))-6.64;
>> fzero(myfun, .5) =    0.5894
```

2.7.4.3 Matlab max-rate-sum MAC programs

Matlab macmax.m Program: The macmax.m program computes simultaneous water-filling for N equally spaced frequency points over the range of $[0, 1/T' = \bar{N}/T)$. This chapter sets $\bar{N} = 1$, while Chapter 5 addresses $\bar{N} > 1$. The macmax program accepts a time-domain input $\boldsymbol{h}$ that is (when $\bar{N} = 1$) simply this chapter's channel matrix H. The sum of user energies is the input Esum, The program input Lxu specifies possibly variable number of transmit antennas . Here set N = 1; cb =1 for complex baseband and cb=2 for real baseband. The program outputs are the resultant Rxx (for each user), which is just an energy for each user when all have single-antenna transmitters, but whose trace is the specified input energies for each user when multiple antennas occur per user. Both the maximum rate sum, and the maximum rate sum with only linear receiver processing are also outputs. The program macmax.m uses the CVX (and Mosek) software packages.

```
function [Rxx, bsum , bsum_lin] = macmax(Esum, h, Lxu, N , cb)

Simultaneous water-filling Esum MAC max rate sum (linear & nonlinear GDFE)
The input is space-time domain h, and the user can specify a temporal
block symbol size N (essentially an FFT size).

This program uses the CVX package
```

```
the inputs are:
Esum The sum-user energy/SAMPLE scalar.
     This will be increased by the number of tones N by this program.
     Each user energy should be scaled by N/(N+nu)if there is cyclic prefix
     This energy is the trace of the corresponding user Rxx (u)
     The sum energy is compouted as the sum of the Eu components
     internally.
h    The TIME-DOMAIN Ly x sum(Lx(u)) x N channel for all users
Lxu  The number of antennas for each user 1 x U
N    The number of used tones (equally spaced over (0,1/T) at N/T.
cb   cb = 1 for complex, cb=2 for real baseband

the outputs are:
Rxx A block-diagonal psd matrix with the input autocorrelation for each
    user on each tone. Rxx has size (sum(Lx(u)) x sum(Lx(u)) x N .
    sum trace(Rxx) over tones and spatial dimensions equal the Eu
bsum the maximum rate sum.
bsum  bsum_lin - the maximum sum rate with a linear receiver

    b is an internal convergence (vector, rms) value, but not sum rate
```

Matlab SWF.m Program: The program SWF.m also computes simultaneous water-filling for N equally spaced frequency points over the range of $[0, 1/T' = \bar{N}/T)$. It however does so for an energy-vector MAC. It's inputs also differ from macmax in that the SWF.m program accepts a **frequency-domain** input H that coincidentally happens to be (when $\bar{N} = 1$) simply this chapter's channel matrix H. The energy vector for all users is an input, Eu. The program input Lxu specifies the (possibly variable) number of transmit antennas/user. Here set N = 1; cb =1 for complex baseband and cb=2 for real baseband. The SWF.m program also permits specification of the noise autocorrelation matrix (same dimensionality as output of H), which spares the program user from having to compute the white-noise equivalent channel themselves. The program outputs are the resultant Rxx (for each user), which is just an energy for each user when all have single-antenna transmitters, but whose traces add to the specified input energies for each user when multiple antennas occur per user. Both the maximum rate sum, and the maximum rate sum with only linear receiver processing are also outputs. The SWF.m program does not use CVX.

```
 function [Rxx, bsum , bsum_lin] = SWF(Eu, H, Lxu, Rnn, cb)

Simultaneous water-filling MAC max rate sum (linear and nonlinear GDFE)
The input is space-time domain h, and the user can specify a temporal
block symbol size N (essentially an FFT size).

Inputs:
Eu  U x 1 energy/SAMPLE vector. Single scalar equal energy all users
    any (N/N+nu) scaling should occur BEFORE input to this program.
H   The FREQUENCY-DOMAIN Ly x sum(Lx(u)) x N MIMO channel for all users.
    N is determined from size(H) where N = # tones
    (equally spaced over (0,1/T) at N/T.
    if time-domain h, H = 1/sqrt(N)*fft(h, N, 3)
Lxu 1xU vector of each user's number of antennas
Rnn The Ly x Ly x N noise-autocorrelation tensor (last index is per tone)
cb  cb = 1 for complex, cb=2 for real baseband

Outputs:
```

```
  Rxx A block-diagonal psd matrix with the input autocorrelation for each
      user on each tone. Rxx has size (sum(Lx(u)) x sum(Lx(u)) x N .
      sum trace(Rxx) over tones and spatial dimensions equal the Eu
  bsum the maximum rate sum.
  bsum  bsum_lin - the maximum sum rate with a linear receiver
      b is an internal convergence sum rate value, not output

  This program is modified version of one originally supplied by student
  Chris Baca
--------------------------------
```

EXAMPLE 2.7.5 *[2×2 Simultaneous Water-Filling]* This example provides a 2×2 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:

```
>> h=[10 8
3 5];
>> [b, GU, WU, S0, MSWMFU] = mu_mac(h, eye(2), [1 1] , 2)
b =    3.3907    1.4959
GU =
    1.0000    0.8716
         0    1.0000
WU =
    0.0092         0
   -0.1242    0.1438
S0 =
  110.0000         0
         0    7.9545
MSWMFU =
    0.0917    0.0275
   -0.0915    0.3464
>> sum(b) =    4.8866
```

($L_y = 2$, $L_{x,1} = L_{x,2} = 1$) For this 2×2 2-user MAC with necessarily diagonal $R_{\boldsymbol{xx}}$ and energy-vector constraint $\boldsymbol{\mathcal{E}} = [1\ 1]^*$, 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 $\mathcal{E}_2 + \mathcal{E}_1 = 2$, 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

$$
\begin{aligned}
|\, S_0 \,| &= |H \cdot R_{\boldsymbol{xx}} H^* + I| && (2.409)\\
&= \left| H \begin{bmatrix} \mathcal{E}_2 & 0 \\ 0 & \mathcal{E}_1 \end{bmatrix} H^* + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right| && (2.410)\\
&= -676 \cdot \mathcal{E}_2^2 + 1372 \cdot \mathcal{E}_2 + 179 \ \text{ uses } \mathcal{E}_1 = 2 - \mathcal{E}_2 \ , && (2.411)
\end{aligned}
$$

which has maximum at

$$\mathcal{E}_2 = \frac{1372}{1352} = 1.0148. \ \ \mathcal{E}_1 = .9852 \quad . \tag{2.412}$$

This value corresponds to

```
>> E2 = 1372/1352 =
    1.0148
>> E1=2-E2 =
    0.9852
```

```
>> det(H*diag([E2 E1])*H'+eye(2)) =
  875.1479
>> bsum =   .5*log2(ans) =  4.8867.
```

(Note h=H when h is $L_y \times \mathscr{L}_x \times 1$.) The use of SWF.m on this channel produces

```
>> [Rxx, bsum, bsumlin] = SWF([1 1]', h, [1 2], eye(2), 2)

Rxx =
    1.0000         0
         0    1.0000
bsum =    4.8866
bsumlin =    3.1365
```

The two user energies trivially match the inputs for this simple channel on the diagonal of the Rxx, which must be diagonal for the MAC. If instead the energy-sum is desired, then macmax.m can be used

```
>> H=h;
>> [Rxx, bsum , bsum_lin] = macmax(1, H, [1 1], 1 , 2)

Rxx =    0.5147         0
         0    0.4853
bsum =    4.0361
bsum_lin =    2.4217
```

The rate sum appears lower , but the total energy input was 1 unit. If this is increased to two units so then including the possibility of 1 energy unit per user, then

```
>> [Rxx, bsum , bsum_lin] = macmax(2, H, [1 1], 1 , 2)

Rxx =
    1.0147         0
         0    0.9853
bsum =    4.8867
bsum_lin =    3.1369
```

This shows that 1 unit each is close to optimum and the gain is .0001 bits/symbol for the energy-sum.

A rank 2 channel with 3 users extends the previous channel by 1 user.

```
>> H=[10 8 3
3 5 3 ];
>> [b, GU, WU, S0, MSWMFU] = mu_mac(H, eye(3), [1 1 1] , 2)
b =    3.3907    1.4959    0.3467
GU =
    1.0000    0.8716    0.3578
         0    1.0000    0.7647
         0         0    1.0000
WU =
    0.0092         0         0
   -0.1242    0.1438         0
    0.3611   -1.0833    1.620
S0 =
```

```
  110.0000         0         0
         0    7.9545         0
         0         0    1.6171
MSWMFU =
    0.0917    0.0275
   -0.0915    0.3464
   -0.1944    0.5278
>> bsum = sum(b) =   5.2333
```

However, this system has 3 units of energy. Instead holding energy at $\mathcal{E}_{\boldsymbol{x}} = 2$ leads instead to

```
>> [Rxx, bsum] = SWF([2/3 2/3 2/3]', h, [1 1 1], eye(2), 2)
Rxx =
    0.6667         0         0
         0    0.6667         0
         0         0    0.6667
bsum =    4.7020
```

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 reduces rate sum when energized. Indeed, the following so verifies: The overall energy-sum maximum is

```
>> [Rxx, bsum , bsum_lin] = macmax(2, H, [1 1 1], 1 , 2)

Rxx =
    1.0148         0         0
         0    0.9852         0
         0         0    0.0000   (zero user 1)
bsum =    4.8867
bsum_lin =    3.1369
```

2.8 Broadcast Channel Capacity Rate Regions and Designs

Section 2.6 describes the general capacity region in Equations (2.283) - (2.285). The single Gaussian BC capacity region also follows more directly from this section's methods. 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 corresponding vector Gaussian BC's design (precoder and detectors).

Similar to Subsection 2.7.4, Subsection 2.8.3 also investigates the design for reliable communication for any $\boldsymbol{b} \in \mathcal{C}_{BC}(\boldsymbol{b})$, again using MMSE criteria. The BC is a dual of the MAC in many ways. Figure 2.60 simplifies this duality by showing the MAC receiver that attains capacity region points (or convex time-shared combinations of those points) for all $\boldsymbol{b} \in \mathcal{C}_{MAC}(\boldsymbol{b})$. This all-user-coordinated MAC receiver processes linearly points in a (nearly) unitary "postcoder" matrix multiply W to create a triangular noise (plus "white" error sequence). The nonlinear postcoder is the successive-user decoding for the chosen rate vector's given user order. That nonlinear processing has a decision operation within it that creates the nonlinearity, but also prevents any noise increase on the inversion.

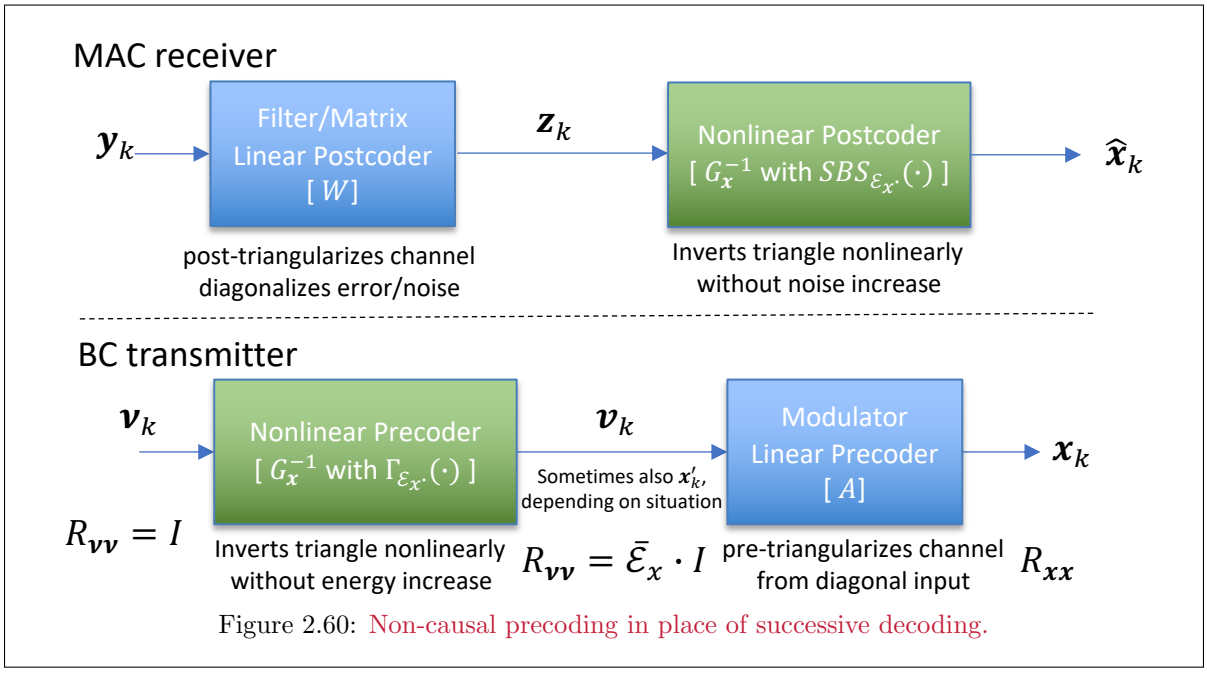


Figure 2.60: Non-causal precoding in place of successive decoding.

Figure 2.60's lower diagram illustrates the dual BC transmitter. This transmitter uses a precoder matrix that preserves the desired $R_{\boldsymbol{xx}}$ channel input (and thus energy) from a "white" input, $\boldsymbol{v}_k$ while also triangularizing the product of itself and the channel matrix. This white input $\boldsymbol{v}$ itself is generated, without energy increase, from the input message values $\boldsymbol{\nu}$ through a nonlinear precoder that inverts the subsequent triangular channel. A nonlinear "modulo" operation is included (in a position that is the same as the successive decoder in the MAC's nonlinear postcoder) that prevents energy increase and retains the independent dimensions of $\boldsymbol{v}$.

Subsection 2.8.3 uses two specifc vector-Gaussian BC design criteria where:

All User Autocorrelation Matrices Are Known All input autocorrelation matrices $\{R_{\boldsymbol{xx}}(u)\}_{u=1,\ldots,U}$ are available.

Only User-summed Autocorrelation Matrix Are Known Only $R_{\boldsymbol{xx}} = \sum_{u=1}^{U} R_{\boldsymbol{xx}}(u)$ is available.

The latter case requires the a worst-case-noise design choice that selects inter-user noise correlation to minimize a vector channel's mutual information. This latter case's worst-case noise also diagonalizes

MMSE receiver processing when combined with Subsection 2.8.1's lossless precoding. The former case does not require the worst-case noise and proceeds as a series of MMSE solutions. The energy-sum MAC concept of primary- and secondary-user components also applies to the vector BC, where the BC's (sub-)user input partitioning corresponds to a dual MAC's input. Indeed, worst-case noise calculations inherently separate the primary- and secondary-user components. Section 2.8.4 then introduces scalar duality (a concept first appearing in this text's earlier versions) to follow the observed symmetric transformation 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 to align indices. Subsection 2.8.5 address continuous-time extension of the Gaussian BC, while Subsection 2.8.6 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 Gaussian process decomposition into user message components

Lemma 2.6.4 establishes that the matrix-AWGN BC's best single input $\boldsymbol{x}$, and consequently all its marginal components $\{\boldsymbol{x}_{u=1,...,U}\}$, have Gaussian distributions. BC AWGN analysis and design, as with the AWGN single-user channel and the AWGN MAC, thus best considers all users as encoded with Gaussian-code inputs and $\Gamma \to 0$ dB. The Gaussian BC input vector $\boldsymbol{x}$ is a linear combination of its marginal (also Gaussian) components in vector

$$\boldsymbol{x}' = \begin{bmatrix} \boldsymbol{x}_1 \\ \vdots \\ \boldsymbol{x}_U \end{bmatrix} \quad , \tag{2.413}$$

A simple combination for the scalar Gaussian BC is $x' = \sum_{u=1}^{U} x_u = \mathbf{1}^* \boldsymbol{x}'$, but that is not the only linear combination for the scalar channel, nor for any other channel with $L_x > 1$.

Entropy Chain Rule: Following Subsection 2.3.2's entropy chain rule, each conditional-entropy chain that sums to $\mathscr{H}_{\boldsymbol{x}'}$ corresponds to an MMSE estimate chain for $\boldsymbol{x}_u$ from $\boldsymbol{x}_{i>u}$, defining

$$\boldsymbol{v}_u = \boldsymbol{x}_u - \hat{\boldsymbol{x}}_{u/\{\boldsymbol{x}_{u+1} ... \boldsymbol{x}_U\}} \quad . \tag{2.414}$$

These MMSE estimates $\boldsymbol{v}_u$ depend tacitly on the order $\boldsymbol{\Pi}$, which shortly simplifies to a single order for all receivers $\boldsymbol{\pi}$ or $\pi_u(i) = \pi(i) \; \forall \; u = 1, ..., U$. All Gaussian MMSE estimates are always linear, so effectively

$$\boldsymbol{v} = \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ \vdots \\ \boldsymbol{v}_U \end{bmatrix} = \begin{bmatrix} 1 & g_{1,2} & ... & g_{1,U} \\ 0 & 1 & ... & g_{2,U} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & ... & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_U \end{bmatrix} = G_{\boldsymbol{x}}^{-1} \cdot \boldsymbol{x}' \quad , \tag{2.415}$$

equivalently $\boldsymbol{x}' = G_{\boldsymbol{x}} \cdot \boldsymbol{v}$, where both $G_{\boldsymbol{x}}$ and $G_{\boldsymbol{x}}^{-1}$ are upper triangular and monic (1's on the diagonal). The autocorrelation matrix is diagonal $R_{\boldsymbol{v}\boldsymbol{v}} = \mathbb{E}[\boldsymbol{v} \cdot \boldsymbol{v}^*] = S_x$ and $R_{\boldsymbol{x}\boldsymbol{x}} = G_x \cdot S_x \cdot G_x^*$. A normalized input $\boldsymbol{\nu}$ is such that $R_{\boldsymbol{\nu}\boldsymbol{\nu}} = I$ where $\boldsymbol{v} = S_x^{1/2} \cdot \boldsymbol{\nu}$ and a square-root matrix is $R_{\boldsymbol{x}\boldsymbol{x}}^{1/2} = G_x \cdot S_x^{1/2}$. Equivalently

$$S_x(u) = \mathbb{E}\left[|v_u|^2\right] \quad \text{and} \quad \nu_u = \frac{v_u}{|v_u|} \quad . \tag{2.416}$$

Clearly $|G_{\boldsymbol{x}}| = 1$ so transformation of a region by $G_{\boldsymbol{x}}$ preserves the region's volume, and $G_{\boldsymbol{x}}$ is invertible, preserving entropy and mutual information. There can be a different $G_{\boldsymbol{x}}$ for each possible user-order

function $\pi(u)$. Each successive estimation error $\boldsymbol{v}_u$ (and $\boldsymbol{\nu}_u$) must (by Appendix D's Orthogonality Principle) be independent of all $\boldsymbol{x}_{i>u}$ and thus then also $\boldsymbol{v}_{i>u}$ (and $\boldsymbol{\nu}_{i>u}$). The random vectors $\boldsymbol{\nu}_u$ each have differential entropies $\mathscr{H}_{\boldsymbol{\nu}_u}$ and via the chain rule then

$$\mathscr{H}_{\boldsymbol{x}} = \mathscr{H}_{\boldsymbol{x}'} = \sum_{u=1}^{U} \mathscr{H}_{\boldsymbol{v}_u} = \mathscr{H}_{\boldsymbol{\nu}} \tag{2.417}$$

for any and all orders. Theorem 2.3.5 and (2.415)'s invertible relationship then permits design to view $\{\boldsymbol{v}_1, ..., \boldsymbol{v}_U\}$ as independent BC encoder inputs to a discrete-modulator matrix $G_{\boldsymbol{x}}$. This $G_{\boldsymbol{x}}$ produces a channel input $\boldsymbol{x}$ that has autocorrelation matrix $R_{\boldsymbol{x}\boldsymbol{x}}$. $R_{\boldsymbol{x}\boldsymbol{x}}$'s dimensional components are not necessarily independent, but they carry the same information as $\boldsymbol{v}$. Indeed, further invertible processing may also follow the generator $G_{\boldsymbol{x}}$ and will still preserve entropy and allow a view of $\boldsymbol{\nu}$ as the actual users' unit-variance inputs without information loss. These $\boldsymbol{\nu}_u$ each map to the independent messages m_u. The final BC channel input vector will have name $\boldsymbol{x}$, while earlier versions in 1-to-1 correspondence with all users' messages may have names like $\boldsymbol{x}'$, $\boldsymbol{v}$, or $\boldsymbol{\nu}$. If the $\boldsymbol{x}_u$ are already independent and $R_{\boldsymbol{x}\boldsymbol{x}}$ is diagonal, then $\boldsymbol{v}_u = \boldsymbol{x}_u$. The sequence $\boldsymbol{v}_{u=1,...,U}$ is sometimes also called the **innovations** of the vector $\boldsymbol{x}$, see [13]. The Gaussian BC's single input $\boldsymbol{x}$ or equivalently $\boldsymbol{v}$, admits Equation (2.148)'s water-filling energy use over the set of all user dimensions in some special situations earlier in Subsection 2.8.3.3. 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

The MAC's canonical decision feedback has a transmit-side equivalent "precoder" implementation that matches well BC constraints. A BC precoder uses a modulo operation that needs definition with respect to an L_x-dimensional lattice Λ:

Definition 2.8.1 (Modulo Operation) *The modulo operation $(\boldsymbol{\nu})_\Lambda$ produces the vector signal in the Lattice Λ's origin-centered Voronoi Region $\mathcal{V}(\Lambda)$ for the vector input $\boldsymbol{\nu}$ that is equal to the difference between $\boldsymbol{\nu}$ and the closest lattice point $\boldsymbol{\lambda} \in \Lambda$. Further the notation $\boldsymbol{\nu} \oplus_\Lambda \boldsymbol{\mu}$ means $(\boldsymbol{\nu} + \boldsymbol{\mu})_\Lambda$.*

Thus for any complex vector $\boldsymbol{\nu}$, then $(\boldsymbol{\nu})_\Lambda = \boldsymbol{\nu} - \lambda_{\boldsymbol{\nu}}$ where $\lambda_{\boldsymbol{\nu}} \in \Lambda$ 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*

$$(\boldsymbol{\mu} + \boldsymbol{\nu})_\Lambda = (\boldsymbol{\mu})_\Lambda \oplus_\Lambda (\boldsymbol{\mu})_\Lambda \quad . \tag{2.418}$$

Proof:

$$\begin{aligned}
(\boldsymbol{\mu} + \boldsymbol{\nu})_\Lambda &= \boldsymbol{\nu} + \boldsymbol{\mu} - \lambda_{(\boldsymbol{\nu}+\boldsymbol{\mu})} && (2.419)\\
&= (\boldsymbol{\nu})_\Lambda + \lambda_{\boldsymbol{\nu}} + (\boldsymbol{\mu})_\Lambda + \lambda_{\boldsymbol{\mu}} - \lambda_{(\boldsymbol{\nu}+\boldsymbol{\mu})} && (2.420)\\
\lambda &\triangleq \lambda_{\boldsymbol{\nu}} + \lambda_{\boldsymbol{\mu}} - \lambda_{(\boldsymbol{\nu}+\boldsymbol{\mu})} && (2.421)\\
(\boldsymbol{\nu} + \boldsymbol{\mu})_\Lambda &= (\boldsymbol{\nu})_\Lambda + (\boldsymbol{\mu})_\Lambda + \lambda && (2.422)\\
&= (\boldsymbol{\mu})_\Lambda \oplus_\Lambda (\boldsymbol{\mu})_\Lambda \quad , && (2.423)
\end{aligned}$$

where the Lattice Λ's closure under addition means that $\lambda \in \Lambda$ and thus λ in (2.422) must be the lattice point at minimum distance from the complex sum $\boldsymbol{\nu} + \boldsymbol{\mu}$. The operation $\oplus_\Lambda$ by definition removes this λ so then (2.418) holds directly . **QED**.

Figure 2.61 illustrates lossless or **non-causal precoding**, which is sometimes also known as **"dirty paper" precoding** and uses the modulo operation. This precoding observes that the channel's addition of a transmitter-known sequence $\boldsymbol{s}$ does not change channel capacity. In 1983, M. Costa[89] 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 2.61's left.

Lemma 2.8.2 (Forney's Crypto Lemma) *Given the additive-noise channel in Figure 2.61 with both $\boldsymbol{\nu}$ and $\boldsymbol{s}$ uniform in distribution over some Voronoi region Λ, then $\boldsymbol{x} = mod_{\Lambda}(\boldsymbol{x}')$ has energy $\mathcal{E}_{\boldsymbol{x}} = \mathcal{E}_{\boldsymbol{\nu}}$, differential entroppy $\mathscr{H}_{\boldsymbol{x}'} = \mathscr{H}_{\boldsymbol{\nu}}$, and $\boldsymbol{x}$ is independent of $\boldsymbol{s}$ and $\boldsymbol{\nu}$.*

Proof: *Addition modulo Λ is a group operation so that any two points' addition produces another point within the group. Thus by Lemma 2.8.1, $\boldsymbol{x} = (\boldsymbol{x}')_{\Lambda} = (\boldsymbol{\nu} - \boldsymbol{s})_{\Lambda} = \boldsymbol{\nu} \ominus \boldsymbol{s}$ since both $\boldsymbol{\nu} \in \mathcal{V}(\Lambda)$ and $\boldsymbol{s} = \boldsymbol{x}' - \boldsymbol{\nu}$. Let $\boldsymbol{\nu}$ and $\boldsymbol{s}$ first be random and uniform over $\mathcal{V}_{\Lambda}$ with $\boldsymbol{s}$ independent of $\boldsymbol{\nu}$, then $p_{\boldsymbol{x}/\boldsymbol{\nu}}(\boldsymbol{\nu} \ominus \boldsymbol{x}) = p_{\boldsymbol{s}}(\boldsymbol{\nu} \ominus \boldsymbol{x}) \equiv$ constant (since s has uniform distribution) for all $\boldsymbol{x}$ and any, and all, particular $\boldsymbol{\nu}$. Thus $\boldsymbol{x}$ has uniform distribution over $\mathcal{V}_{\Lambda}$ and is independent of $\boldsymbol{\nu}$. Further, since the distributions are the same, $\mathscr{H}_{\boldsymbol{x}'} = \mathscr{H}_{\boldsymbol{x}}$.* **QED.**

Continuing if $\boldsymbol{s}$ is deterministic and not random nor uniform over $\mathcal{V}(\Lambda)$, then a precomputed uniform (over $\mathcal{V}(\Lambda)$) random dither sequence $\boldsymbol{d}$ may be added to $\boldsymbol{s}$ at the transmitter and subtracted at the receiver without loss. The sum $\boldsymbol{s} \oplus \boldsymbol{d}$ 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 2.61's right half shows the ML decoder's form for the lossless non-causal precoder. Figure 2.61 shows neither the good-code's $\Gamma = 0$ dB encoder nor decoder because they are presumed present just before and just after Figure 2.61's precoder and corresponding decoder operation; the omitted encoder/decoder both span infinite time over many subsymbols $\boldsymbol{\nu}$ and lay outside the use of Λ (which may itself also separately have large dimension) and the precoder.

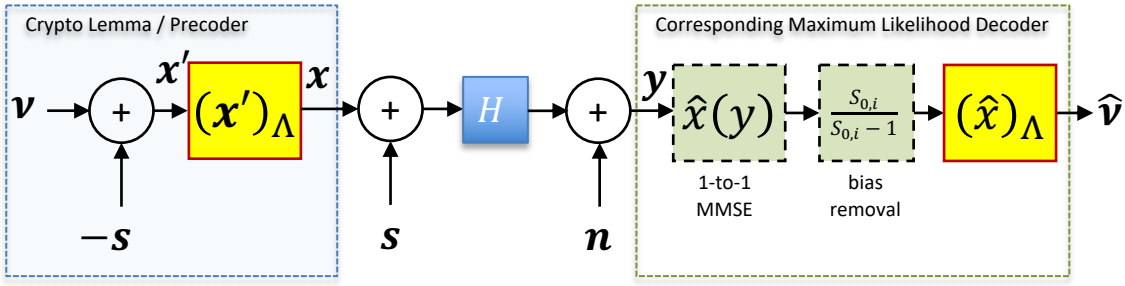


Figure 2.61: Non-causal precoding in place of successive decoding.

Codewords have subsymbols independently selected from a continuous uniform distribution over $\mathcal{V}(\Lambda)$ in Subsection 2.3's sense. Indeed for such infinite length, Λ may become a hypersphere, and then $\boldsymbol{x}$ is a Gaussian code. When $\boldsymbol{y} = y$ is a scalar, the green shaded boxes combined function simply passes y unscaled. In this simple scalar case with $H = h = 1$, then the modulo device directly processes y. This

[89] After Max Henrique Machado Costa, circa 1952 –, a Brazilian academic and electrical engineer, and former Stanford student.

leads to

$$
\begin{aligned}
(y)_\Lambda &= (x+s+n)_\Lambda && (2.424)\\
&= ((\nu-s)_\Lambda + s + n)_\Lambda && (2.425)\\
&= (\nu)_\Lambda \oplus_\Lambda s \ominus s \oplus (n)_\Lambda && (2.426)\\
&= \nu \oplus_\Lambda (n)_\Lambda \ . && (2.427)
\end{aligned}
$$

The channel output is basically the channel input ν 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 Λ. Generalizing slightly $H = I$ can be multidimensional, but is essentially several scalar channels in parallel. Again, the shaded green boxes become independent passages for each of $\boldsymbol{y}$'s dimensions. The lattice Λ now can apply across these L_y dimensions. The same result in (2.427) holds that ML subsymbol processing (apart from the actual outer AWGN code's ML decoder) is simply a modulo for the lattice Λ. The scalar case's MMSE (from Appendix D) is $\sigma^2_{mmse} = \bar{\mathcal{E}}_{\boldsymbol{x}} - (|h|^2 \cdot \bar{\mathcal{E}}_{\boldsymbol{x}})/\sigma^2$.

However, when $L_x > 1$ and H is a non-diagonal matrix, then the MMSE estimation is non-trivial and will maximize the ratio $S_0 \triangleq \bar{\mathcal{E}}_{\boldsymbol{x}}/\sigma^2_{mmse}$ since $\sigma^2_{mmse} = trace\{R_{\boldsymbol{ee}}\}$ is, by definition minimum. As with the MAC and single-user MMSE cases, if the MMSE output is $\boldsymbol{z}$, then $E[\boldsymbol{z}/\boldsymbol{x}] = (I - S_0^{-1}) \cdot \boldsymbol{x}$, then the subsymbol amplitude needs to increase by $(S_0 - I)^{-1} \cdot S_0$ 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 precoding and successive-decoding literature where a zero-forcing precoder is used incorrectly, so the reader might best be vigilant in alternative review. Successive decoding's use of exact other-user crosstalk removal is suboptimum. MMSE decision feedback however is better and canonical in that for $P_e \to 0$ for good AWGN-channel code's use when the bits/symbol is less than the mutual information. Part of the literature confusion on this arises from failure to distinguish the subsymbol bias from the infnite-length codeword bias. The latter codeword bias does not change the infinite-length ML decoder decision regions prior to the outer AWGN ML decoding.[90] However, it does affect decoders that use subsymbols where receiver should remove the bias.

The receiver's MMSE estimator also automatically addresses any dimensional differences when $L_{y,u} \neq L_x$ and will produce an L_x-dimensional output consistent with $\boldsymbol{x}$ and $\boldsymbol{v}$ that are both L_x dimensional.

More precoder observations The name "crypto" arises from the independence of $\boldsymbol{\nu}$ from $\boldsymbol{x}$, which basically means the input ν is not recoverable[91] if $\boldsymbol{s}$ is an encryption key. The name "dirty paper" arises from a view of $\boldsymbol{s}$ as existing writing on paper, and then $\boldsymbol{x}$ is new writing on the same paper that contains a "key" ($-\boldsymbol{s}$ hidden within the new writing) such that the receiver can "cleanse" the paper through modulo operation. Figure 2.61'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 shortly).

Implication for the AWGN and Gaussian capacity-achieving codes For coding applications, the crypto lemma applies to any lattice Λ of L_x dimensions from which a good code effectively selects the subsymbols $\boldsymbol{\nu}$ (or equivalently $\boldsymbol{x}$) 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 $\mathcal{I}(\boldsymbol{\nu}; \boldsymbol{y})$ for that uniform distribution. The ML detector begins with Figure 2.61's modulo (and unbiased MMSE) and then proceeds to find the codeword closest to the sequence of outputs $\boldsymbol{y}$ without influence of $\boldsymbol{s}$. When the input has an average energy constraint $\mathcal{E}_{\boldsymbol{x}}$ then

$$
\boldsymbol{x} \in \{\boldsymbol{x} \mid E[\|\boldsymbol{x}\|^2] \leq \mathcal{E}_{\boldsymbol{x}}\} \ , \qquad (2.428)
$$

[90] 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.

[91] If an eavesdropper does also know $\boldsymbol{s}$.

and $\mathcal{V}(\Lambda)$ and the energy constraint are not quite consistent; unless the number of subsymbols in a codeword becomes infinite $\bar{N} \to \infty$. Then essentially the subsymbol selected from Λ tends towards selection from a Gaussian distribution where $\Lambda \to \mathcal{S}_{L_x}$, the on-average $\widetilde{N}$-dimensional complex hypersphere's interior where

$$\mathcal{S}_{L_x} \triangleq \{\boldsymbol{x} \in \mathbb{C}^{L_x} \mid \mathbb{E}[\|\boldsymbol{x}\|^2] \leq \mathcal{E}_{\boldsymbol{x}}\} \quad . \tag{2.429}$$

The subsymbol's finite-dimensional ($L_x < \infty$) marginal distribution becomes Gaussian with average energy $\widetilde{\mathcal{E}}_{\boldsymbol{x}}$ while the overall symbol/codeword distribution is uniform within the infinite-dimensional hypersphere $\mathcal{S}_\infty$. The latter uniform distribution over the ball[92] of $\mathcal{S}_\infty$ meets directly the crytpo lemma requirement. However, the crypto lemma's dither-argument proof for non-uniform $\boldsymbol{s}$ also applies to the subsymbol's non uniform (that is, Gaussian) distribution. While encoding of $\boldsymbol{\nu}$ 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 $\Lambda \approx \mathcal{S}_{\widetilde{N}}$, with some shaping loss that reduces as L_x and/or $\widetilde{N}$ becomes large[93]. The AEP suggests more generally that any codeword distribution limited to being uniform in its subsymbols' marginals is optimum and meets $b \to \mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$; when those subsymbols have themselves large finite-dimensional constellations $|C| \to \infty$. 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 $b \to \mathcal{C}$. Clearly $\mathcal{I} \leq \mathcal{C}$.

Uniform over any Voronoi region separates fundamental gain from shaping gain: The Crypto Lemma also implies that random code generation (in the AEP sense), given a fixed shaping lattice, independently and uniformly in the limit as $\bar{N} \to \infty$ with $\widetilde{N} < \infty$ (dimension of shaping lattice) preserves the good code properties that any rate below the corresponding mutual information is reliably decodable with vanishingly small P_e. There will be a shaping loss, but the MMSE receiver estimate with associated decoding for the good fundamental-gain code, is as good as possible. Thus, if the shaping loss is acceptable, asymptotic results for MMSE/ML decoding allow $P_e \to 0$. Forney approximated the the corresponding mutual-information reduction (with MMSE) as

$$\mathcal{C} - \mathcal{I} \leq \log_2 \left(\pi e \cdot \frac{\mathcal{E}_{\boldsymbol{x}}}{\mathcal{V}^{2/\widetilde{N}}(\Lambda_s)} \right) \quad , \tag{2.430}$$

which computes to well known result of 1.53 dB when $]\Lambda_s = \mathbb{Z}^{\widetilde{N}}$, and of course will be smaller for other lattices. Thus, results (like for instance with the MAC) where there is a feedback section in the decoder that uses other users' decisions will remain canonical for the given shaping region, even when it is not an infinite-dimensional hypershpere. So Γ need not be zero, but the codes fundamental gain must essentially become infinite (which also happens for capacity-achieving codes) with the MMSE receiver. In this context, "good code" then means the fundamental gain is such that the points are spaced uniformly within the codewords formed from randomly sampling symbol values from a finitie $\widetilde{N}$-dimensional lattice's continuous uniform distribution over its Voronoi region and $\bar{N} \to \infty$. This is what modern code designers nearly universally assume, even though they may view it in different ways.

Implication for the Gaussian BC The crypto lemma has important use in Gaussian BCs: This use is asymptotic as $\bar{N} \to \infty$. Here, the side information $\boldsymbol{s}$ 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 $\boldsymbol{s}$. Then the received signal component $\boldsymbol{v}_u$, or equivalently $\boldsymbol{x}_u$, passes to its ML decoder through Figure 2.61's initial decoder processing as if those other users are not present. Of course those other users may reduce the amount of $\mathcal{E}_{\boldsymbol{x}}$ that is available for user u, but they create no crosstalk[94]. 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

[92] A ball means the hypersphere's interior.

[93] A "shaping" code might also be applied to temporal groups of L_x dimensional subsymbols. So even with finite L_x, groups of L_x dimensions that are large can have Gaussian-approaching L_x-dimensional marginal distributions.

[94] Again, as with the MAC, this statement qualifies as "no harmful" crosstalk.

so the order $\pi(u)$ also applies to the subsymbols in implementation. 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 Λ, the modulo operation here generalizes from modulo $\mathcal{V}_\Lambda$ to $(\bullet)_{\mathcal{E}_u}$; 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 infinite-dimensional hypersphere as in Section 2.1's Gaussian-code sphere packing with marginal distributions appropriately approaching Gaussian. The total energy will be $\mathcal{E}_{\boldsymbol{x}} = \sum_{u=1}^{U} \mathcal{E}_u$ 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 $\boldsymbol{s} = -\sum_{i=u+1}^{U} g_{u,i} \cdot \boldsymbol{x}_i$, where $g_{u,i}$ has been trivially $g_{u,i} = 1$ when $H = I$, and will be specifically determined in other cases[95]. The precoder's pre-modulo subtraction of this same "side information as other users" ensures that the receiver's second modulo operation restores $\boldsymbol{\nu}$. The encoder re-adds the signal, $\boldsymbol{s}$ to $\boldsymbol{x}$ prior to channel entry, which is the BC;s addition of all message signals, each sharing the available transmit energy. The ideal precoder modulos in Figures 2.61 and 2.66 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 2.66 and 2.64, as Subsection 2.8.2 further discusses.

2.8.2 Scalar Gaussian BC Channel Capacity regions

Figure 2.62 illustrates three equivalent versions of the two-user scalar AWGN BC with $L_x = L_{y,1} = L_{y,2} = 1$. The signal x adds user 1 and user 2 messages' modulated subsymbols. Figure 2.62a shows the 2-user BC as is. Figure 2.62b shows the channel with presumed receiver noise-whitening/normalization. Figure 2.62c 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 $|h_1| \geq |h_2|$.

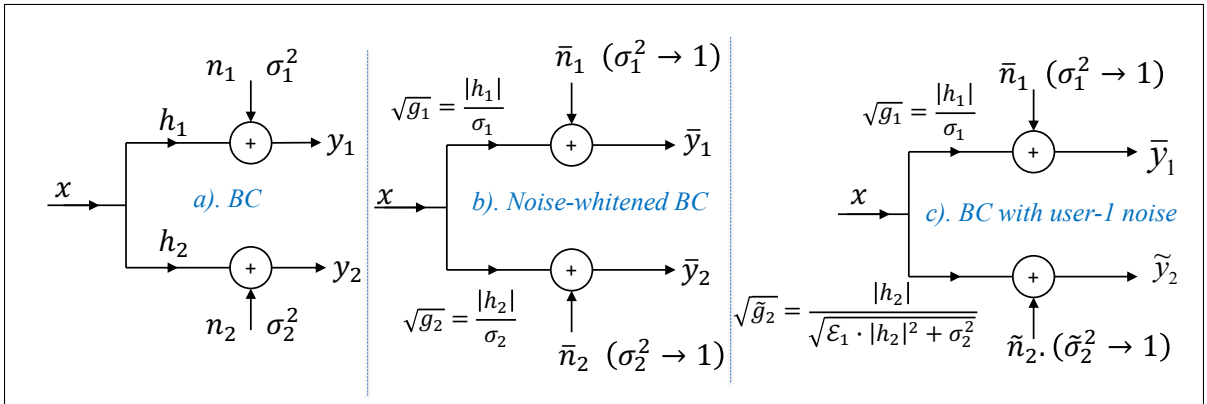


Figure 2.62: Simple **SCALAR** two-user broadcast AWGN channel with $g_1 \geq g_2$: 3 equivalent forms

More generally, the U-user **scalar**[96] Gaussian BC has given channel gains

$$g_u = \frac{|h_u|^2}{\sigma_u^2} \quad , \tag{2.431}$$

with a chosen order such that

$$g_1 \geq g_2 \geq \ldots \geq g_U \quad , \tag{2.432}$$

[95] Usually $\neq \sqrt{h/\sigma^2}$, but MMSE based.

[96] when either or both of $L_x > 1$ or $L_y > 1$, then the g_u definition is $g_u = | H_u \cdot R_{\boldsymbol{nn}}^{-1}(u) \cdot H_u |$ and then also orders the users.

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

$$\begin{aligned} \mathcal{E}_1 &= \alpha \cdot \mathcal{E}_{\boldsymbol{x}} && (2.433) \\ \mathcal{E}_2 &= (1-\alpha) \cdot \mathcal{E}_{\boldsymbol{x}} && (2.434) \\ \mathcal{E}_1 + \mathcal{E}_2 &= \mathcal{E}_{\boldsymbol{x}} \quad , && (2.435) \end{aligned}$$

where $0 \leq \alpha \leq 1$. The dimensionality of $\mathcal{E}_1$ and $\mathcal{E}_2$ is the same as $\mathcal{E}_{\boldsymbol{x}}$ and of 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 2.63 shows the successive decoders that apply to Figure 2.62b and 2.62c channels. (Figure 2.63 removes the bars from $\boldsymbol{y}$'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:

$$\begin{aligned} \bar{b}_1 &\leq I(x_1; y_1/x_2) = \frac{1}{2}\log_2\left(1 + \alpha \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_1\right) && (2.436) \\ \bar{b}_2 &\leq I(x_2; y_2) = \frac{1}{2}\log_2\left(1 + \frac{(1-\alpha) \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_2}{1 + \alpha \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_2}\right) \quad , && (2.437) \end{aligned}$$

which when added are not an instance of the chain rule (because $y_1 \neq y_2$).

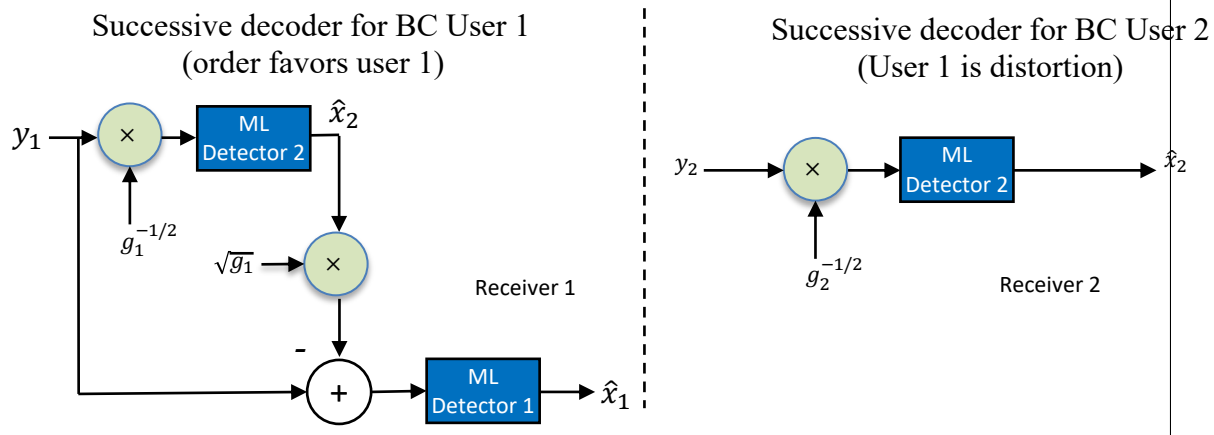


Figure 2.63: 2-User Scalar Gaussian BC successive decoders.

The bound (2.437) presumes that user 2's receiver considers any x_1 component as (Gaussian) noise; however this is optimum as per Lemma 2.6.4. Each $\mathcal{I}$ term corresponds to an MMSE estimate. $\mathcal{I}(x_U; y_U)$ has $x_{1,\ldots,U-1}$ all as noise; $\mathcal{I}(x_{U-1}; y_{U-1}/x_U)$ has $x_{1,\ldots,U-2}$ as noise, and so on. Even though the these mutual information quantities have different n_u (unlike MAC), they still correspond each to an MMSE-based decoder.

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 maximum data rate 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 2.64 generalizes the 2-user scalar Gaussian BC to U users. Figure 2.64 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, receiver-side successive decoding appears more complex than precoding for BC implementation because of the repeated receiver-side implementations. Figure 2.64 also uses the individual h_u directly rather than the scalar g_u values that Figures 2.62 and 2.63 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 2.63'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 2.61's MMSE estimate with bias necessitates removal.

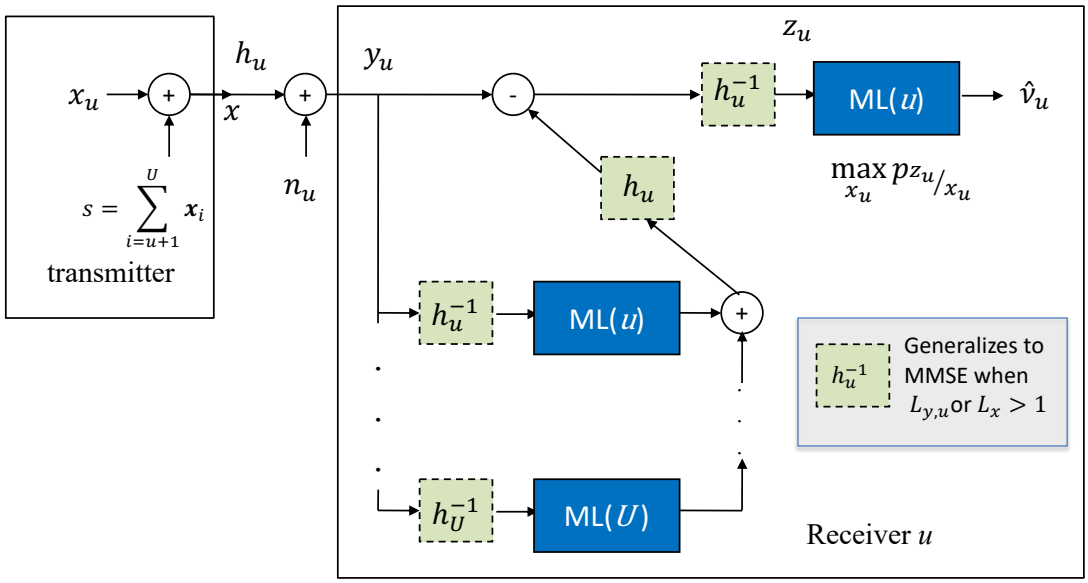


Figure 2.64: Transmitter for **SCALAR** BC with successive decoding receivers.

2.8.2.2 Scalar precoding

Figure 2.65 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 ν_1 and ν_2 respectively into codewords x_1 and x_2 that are ideally infinitely long (Gaussian) subsymbol sequences, with $\bar{N} \to \infty$. The modulo device presumes this infinite-length encoding creates Gaussian-distributed subsymbols x_u, and consequently the modulo would need infinite delay to precode perfectly a signal inside the origin-centered hypersphere $\mathcal{S}_\infty$. Specifically, the ideal non-causal modulo device thus translates any codeword sequence ν_1 into signal sequence $x_1 = \left(\sqrt{\alpha \cdot \mathcal{E}_{\boldsymbol{x}}} \cdot \nu_1 - x_2\right)_{\mathcal{E}_1}$ (where $\mathcal{E}_1 \triangleq \alpha \cdot \mathcal{E}_{\boldsymbol{x}}$). This $\text{mod}(\bullet)_{\mathcal{E}_{\boldsymbol{x}}}$ operation precodes sequence values outside the hypersphere of radius

$\sqrt{\mathcal{E}_1}$ 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 $\mathcal{E}_1$, no matter what the input energy. User 2's signal $x_2 = \sqrt{(1-\alpha)\cdot\mathcal{E}_{\boldsymbol{x}}}$ has energy $\mathcal{E}_2 = (1-\alpha)\cdot\mathcal{E}_{\boldsymbol{x}}$. As Figure 2.65 shows, $\mathcal{E}_{\boldsymbol{x}} = \mathcal{E}_1 + \mathcal{E}_2$ on the BC input.

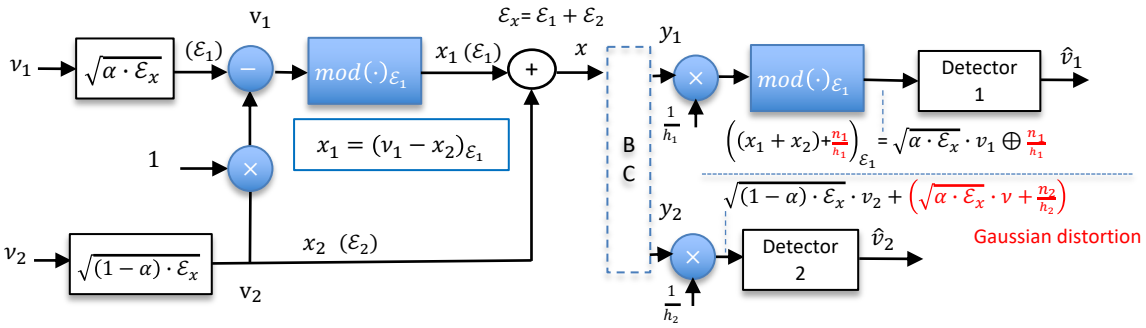


Figure 2.65: 2-User Scalar Gaussian BC with precoder.

Understanding Figure 2.65: 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 α, 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 $\mathcal{E}_1$. Users 1 and 2 consequently have two components in

$$\begin{aligned} y_1 &= \sqrt{g_1}\cdot x + n_1 && (2.438)\\ &= h_1\cdot(x_1+x_2)+n_1 && (2.439)\\ &= h_1\cdot\left((\sqrt{\mathcal{E}_1}\cdot\nu_1 \ominus x_2)_{\mathcal{E}_1} + x_2\right) + n_1 && (2.440)\\ \left(\frac{y_1}{h_1}\right)_{\mathcal{E}_1} &= \sqrt{\mathcal{E}_1}\cdot\nu_1 \oplus \frac{n_1}{h_1} && (2.441)\\ &\approx \sqrt{\alpha\cdot\mathcal{E}_{\boldsymbol{x}}}\cdot\nu_1\ , && (2.442) \end{aligned}$$

which corresponds to $\mathcal{I}(x_1; y_1/x_2)$ or its MMSE-estimate equivalent that removes/mitigates $\sqrt{g_1}\cdot x_2$, and

$$\begin{aligned} y_2 &= h_2\cdot(x_2+x_1)+n_2 && (2.443)\\ &= h_2\cdot\left[x_2 + \underbrace{(\nu_1 \ominus x_2)_{\mathcal{E}_1}}_{\sqrt{\alpha\cdot\mathcal{E}_{\boldsymbol{x}}}\cdot\nu}\right] + n_2 && (2.444)\\ \frac{y_2}{h_2} &= \sqrt{(1-\alpha)\cdot\mathcal{E}_{\boldsymbol{x}}}\cdot\nu_2 + \left(\sqrt{\alpha\cdot\mathcal{E}_{\boldsymbol{x}}}\cdot\nu + \frac{n_2}{h_2}\right)\ , && (2.445) \end{aligned}$$

which corresponds to $\mathcal{I}(x_2; y_2)$ with x_1 as noise. Equations (2.441), (2.445), and Figure 2.65 show 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 2.65's receiver signal ν is random (Gaussian) with unit energy and is the signal at the precoder modulo output that is independent of user 2. The component ν thus has the same (Gaussian) distribution as both ν_1 and/or ν_2. User 1's receiver modulo element removes both the user 2 components because they are equal and opposite in sign, modulo $\mathcal{E}_1$, leaving only the noise $g_1^{-1/2}\cdot n_1$. Effectively, user 2's receiver sees ν equivalent to x_1

with energy $g_2 \cdot \mathcal{E}_1 = \alpha \cdot g_2 \cdot \mathcal{E}_{\boldsymbol{x}}$. The crosstalk noise $\sqrt{|h_2|^2 \cdot \alpha \cdot \mathcal{E}_{\boldsymbol{x}}} \cdot \nu$ 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 decode user 1 reliably. The precoder is equivalent to the set of successive decoders, although arguably simpler, being implemented only once in the single BC transmitter. Again, the simple h_1^{-1} and h_2^{-1} receiver multiplies become MMSE estimates in the non-scalar case.

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 $\mathcal{C}(\boldsymbol{b})$ 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 α_u such that

$$\mathcal{E}_u = \alpha_u \cdot \mathcal{E} \quad \forall\, u = 1, ..., U \tag{2.446}$$

where $\alpha_U = 1 - \sum_{i=1}^{U-1} \alpha_i$ and

$$0 \le \alpha_u \le 1 \quad . \tag{2.447}$$

The U-user scalar precoder essentially implements the crypto lemma as in Figure 2.66.

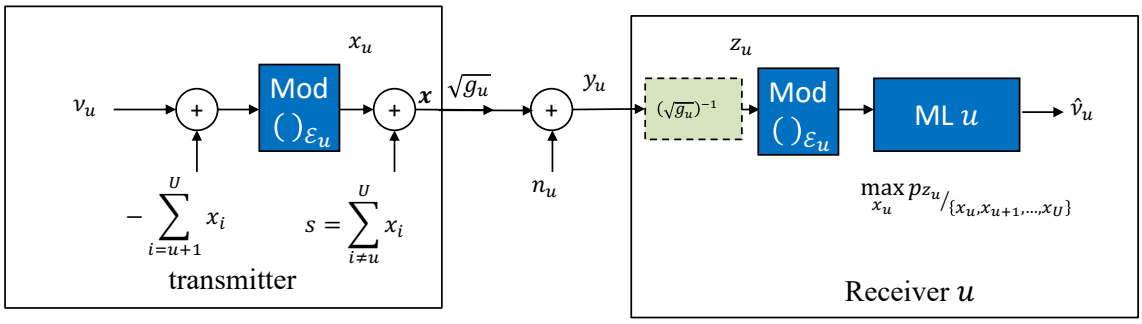


Figure 2.66: Transmitter and receiver u for BC with noiseless precoder.

Figure 2.66 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 2.62 and sets $h_1 = .8$ and $h_2 = .5$ and $\sigma^2 = .0001$. The single-user mutual information upper bound from (2.141), which may require receiver coordination and thus not be attainable, is

$$I(x; \boldsymbol{y}) = \frac{1}{2} \log_2 \left(\frac{|R_{\boldsymbol{yy}}|}{|R_{\boldsymbol{nn}}|} \right) = \frac{1}{2} \log_2 \left(\frac{(.6401) \cdot (.2501) - .4^2}{.0001^2} \right) = 6.56 \text{ bits/dimension.} \tag{2.448}$$

The sum of the data rates thus cannot exceed 6.56 bits/dimension for the BC. Using the exact formulas in (2.436) and (2.437) for various α produces the following table:

α	$\bar{b}_1$	$\bar{b}_2$	$\bar{b} = \bar{b}_1 + \bar{b}_2$
1.0	6.32	0	6.32
.75	6.12	.20	6.32
.50	5.82	.50	6.32
.25	5.32	1.0	6.32
.10	4.66	1.66	6.32
.05	4.16	2.15	6.31
.01	3.01	3.29	6.30
.001	1,44	4.74	6.18
0	0	5.64	5.64

Table 2.4: Example 2.8.1's BC energy choices and corresponding data rates.

The rate sum simplifies with algebra for $\alpha >> \mathcal{E}_{\boldsymbol{x}}/g_2 = 1/2500 = .0004$ to

$$\bar{b} = \frac{1}{2}\log_2 \frac{(1+\mathcal{E}_{\boldsymbol{x}} \cdot g_2)\cdot(1+\alpha\cdot\mathcal{E}_{\boldsymbol{x}}\cdot g_1)}{1+\alpha\cdot\mathcal{E}_{\boldsymbol{x}}\cdot g_2} = \frac{1}{2}\log_2(\mathcal{E}_{\boldsymbol{x}}\cdot g_1) = 0.5\cdot\log_2(80^2) = 6.32 \qquad (2.449)$$

The corresponding rate region appears in Figure 2.67.

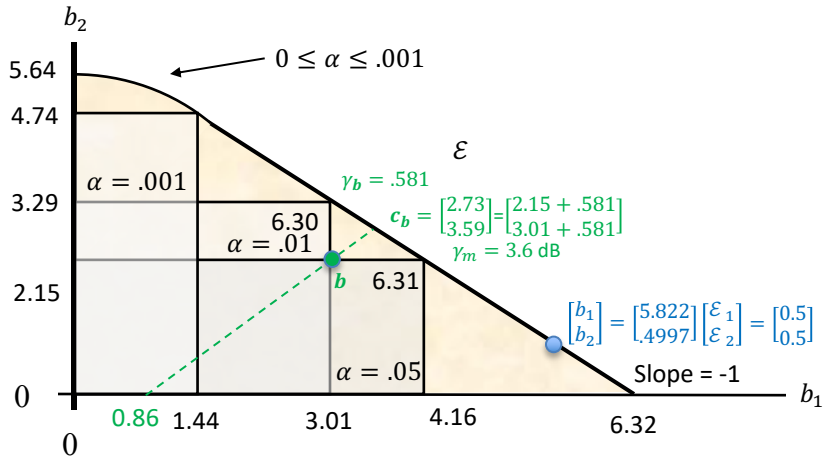


Figure 2.67: Rate region for Example 2.8.1's broadcast channel

For this channel, Table 2.4's input energy allocations correspond to $\mathcal{E}_{\boldsymbol{x}} = 1$. Figure 2.67's gray boxes corresponding to specific choices in Table 2.4. Essentially then the sum rate equals user 1's maximum rate until α is sufficiently small. At this low α level, user 2's energization begins to decrease the sum $\bar{b}$ as well as rapidly $\bar{b}_1$. Receiver 1 basically has best use of the single input dimension as a primary component/user, while significant energy on component/user 2 leads to overall rate loss as this secondary user component increasingly uses energy and thus channel resources. This is similar to the scalar MAC with only an energy-sum constraint.

The capacity rate region $\mathcal{C}(\boldsymbol{b})$ does not attain Equation (2.448)'s single-user maximum rate sum, illustrating loss from the BC's uncoordinated-user-receiver constraint. The BC loss is thus

$$\gamma_{BC} = \frac{2^{2\cdot 6.56}-1}{2^{2\cdot 6.32}-1} = 1.395 \;\; (1.45 \text{ dB}) \; . \qquad (2.450)$$

This example's rate sum is relatively constant except for very small values of α, which is not true in general and derives from the specific $h_u \cdot \mathcal{E}_{\boldsymbol{x}} >> \sigma^2$ in this example. Figure 2.67

also shows the boundary-image point $\boldsymbol{c_b}$ for the interior point $\boldsymbol{b} = [2.15\ 3.01]^*$, which follows from

```
gammab=(6.322-5.16)/2 =    0.5810 dB
>> bvec=[3.01 ; 2.15];
>> cvec=bvec+ones(2,1)*gammab =
    3.5910
    2.7310
```

General Scalar BC: Example 2.8.1 illustrates that the two users share a common dimension, and that user 1's receiver can decode both signals reliably. The broadcast capacity region for general $1 \times U$ Gaussian BC is then with ordering

$$|g_1| \geq |g_2| \geq |g_3| \geq ... \geq |g_U| \tag{2.451}$$

is then

$$b_1 \leq \mathcal{I}(x_1; y_1/x_2, x_3, ..., x_U) = \frac{1}{2} \cdot \log_2 \left(1 + \alpha_1 \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_1\right) \tag{2.452}$$

$$b_2 \leq \mathcal{I}(x_2; y_2/x_3, ..., X_U) = \frac{1}{2} \cdot \log_2 \left(1 + \frac{\alpha_2 \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_2}{1 + \alpha_1 \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_2}\right) \tag{2.453}$$

$$b_3 \leq \mathcal{I}(x_3; y_3/x_4, ..., X_U) = \frac{1}{2} \cdot \log_2 \left(1 + \frac{\alpha_3 \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_3}{1 + (\alpha_1 + \alpha_2) \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_3}\right) \tag{2.454}$$

$$\vdots \tag{2.455}$$

$$b_U \leq \mathcal{I}(x_U; y_U) = \frac{1}{2} \cdot \log_2 \left(1 + \frac{\alpha_U \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_U}{1 + (\sum_{u=1}^{U-1} \alpha_u) \cdot \bar{\mathcal{E}}_{\boldsymbol{x}} \cdot g_U}\right) \tag{2.456}$$

2.8.3 The Vector Gaussian BC Design

This subsection describes two basic BC design approaches. Subsection 2.8.3.1 addresses design that initially specifies the user-autocorrelation-matrix set

$$\begin{aligned} &\{R_{\boldsymbol{xx}}(u)\}_{u=1,...,U}\ , \\ \text{with}\quad &R_{\boldsymbol{xx}} = \textstyle\sum_u R_{\boldsymbol{xx}}(u) \quad . \end{aligned} \tag{2.457}$$

This first **MMSE BC Design** is simpler, but depends upon knowing all the users' specific $R_{\boldsymbol{xx}}(u)$. Subsection 2.8.3.2 develops a second **Worst-Case Noise (WCN) Design** approach that only depends on $R_{\boldsymbol{xx}}$ without user-specific decomposition, leaving user decomposition to be optimized instead of given. The WCN approach treats the BC effectively as a single-user channel with the additional restriction that certain output dimension sets cannot coordinate. This output-processing independence connects to a worst-case noise that helps separate user components into two sets: primary and secondary components. The worst-case noise depends on only the primary-user components. Again, secondary user components will "freeload" on the primary-user components' better channel use and reduce overall sum rate. Chapter 5 further bridges these MMSE-BC and WCN designs using vector duality and describes best input decomposition of $R_{\boldsymbol{xx}}$ into $\{R_{\boldsymbol{xx}}(u)\}$ for a given $\boldsymbol{b} \in \mathcal{C}_{BC}(\boldsymbol{b})$. Subsection 2.8.3.4 provides a general theory of primary/secondary-user components with nonlinear precoders that combines insight from the WCN approach with the MMSE approach. Subsection 2.8.3.5 investigates primary-user-component properties in the WCN approach.

Figure 2.68 illustrates the vector Gaussian BC with individual-user noise whitening and input generation from a white "innovations" representation $\boldsymbol{v}$ that collects all BC users into ϱ_x (the rank of the given $R_{\boldsymbol{xx}}$) independent subsymbol dimensions, and $\varrho_x \leq L_x$. When $\varrho_x < L_x$, the input autocorrelation may correspond to a well-designed $R_{\boldsymbol{xx}}$, such as water-filling that may not use all possible BC input

dimensions. As with the MAC, the single-channel input's decomposes into independent sub-user/user dimensional components simplifies analysis. A good number-of-components choice is

$$U' = \sum_{u=1}^{U} \min(\varrho_x, \varrho_{H_u}) \leq \mathscr{L}_y \cdot L_x \quad . \tag{2.458}$$

Again, the BC has only an energy-sum constraint $trace\{R_{\boldsymbol{xx}}\} \leq \mathcal{E}_{\boldsymbol{x}}$, thus directly analogous to the energy-sum MAC. The latter energy-sum condition permits excitation of every possible nonzero energy/information channel-transfer passage to the channel output dimensions. However first, the specification of each and every user's specific subcomponent falls easily again into MMSE design and produces the next subsection's specific noiseless precoder.

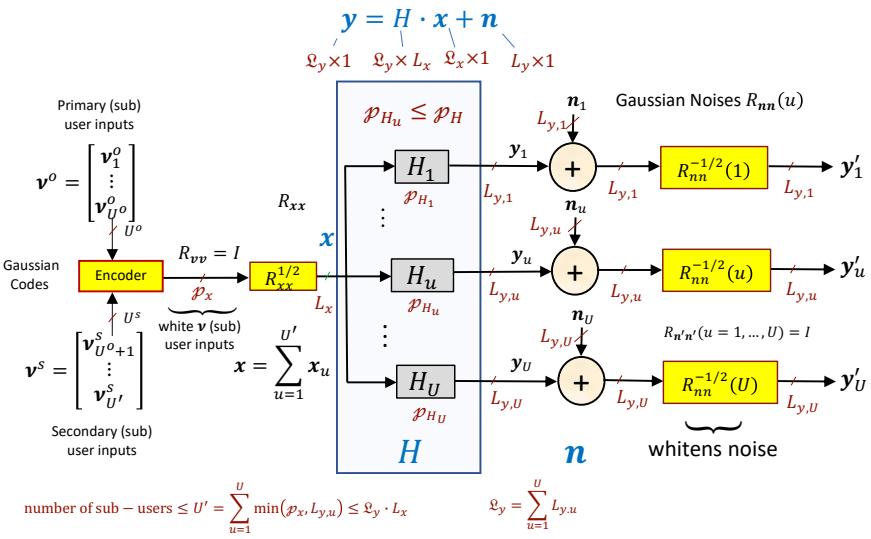


Figure 2.68: Vector Gaussian BC

2.8.3.1 MMSE BC Precoders with known user autocorrelation-matrix set

The MMSE-BC design directly finds the precoder from the known specified $R_{\boldsymbol{xx}}(u)$ for U independent user-input components, H, and $R_{\boldsymbol{nn}}(u)$. The MMSE-BC design also directly follows the MMSE MAC approach that produces the feedback matrix $\mathcal{G}$ and forward matrix $\mathcal{W}$, given all known user input autocorrelation matrices. The MMSE-BC precoder determination follows the MAC's decision-feedback design repeatedly for each receiver u, determining that user's precoder coefficients, with that precoder preceding a given known linear transmitter matrix $A_u = R_{\boldsymbol{xx}}^{1/2}(u)$. The corresponding user receiver's unbiased MSWMF matrix rows $\mathcal{W}_u$ and the transmitter's unbiased precoder rows $\mathcal{G}_u$ contain L_x correspondingly indexed subset rows. This exercise repeats for each receiver. So where the MAC had one MMSE solution, the BC has U MMSE solutions. Row subsets from each BC solution aggregate into a size U larger-matrix design.

Individual receivers: MMSE-BC design individually processes each separately noise-whitened individual user-channel output signal:

$$\boldsymbol{y}_u = R_{\boldsymbol{nn}}^{-1/2}(u) \cdot H_u \cdot \boldsymbol{x} + \boldsymbol{n}'_u \tag{2.459}$$

where $R_{\boldsymbol{n}'\boldsymbol{n}'}(u) = I$. The assignment of users to integer indices within the matrix determines the order $\boldsymbol{\pi}$. For all BC designs and that order, the single input $\boldsymbol{x}$ decomposes into known components

$$\boldsymbol{x} = \left[R_{\boldsymbol{xx}}^{1/2}(1) \vdots R_{\boldsymbol{xx}}^{1/2}(2) \vdots \ldots \vdots R_{\boldsymbol{xx}}^{1/2}(U) \right] \cdot \boldsymbol{v} \quad , \tag{2.460}$$

where receiver u attempts to decode user u reliably after a modulo device removes (MMSE sense) users $i > u$ and receiver u treats users $i < u$ as noise. The MMSE-BC design expands the number of user (square-matrix) precoder/input-output dimensions fo $U \cdot L_x$ so that there is an BC-input message component corresponding to every potential receiver-output dimension (or component). Effectively, the input autocorrelation matrix is a block diagonal $R_{\boldsymbol{XX}}$, with each diagonal block holding a single $R_{\boldsymbol{xx}}(u)$ component. The channel-input autocorrelation matrix, however, remains the sum $R_{\boldsymbol{xx}} = \sum_{u=1}^{U} R_{\boldsymbol{xx}}(u)$. Each receiver sees an equivalent $L_{y,u} \times [U \cdot L_x]$ channel

$$\widetilde{H}_u = \underbrace{R_{\boldsymbol{nn}}^{-1/2}(u) \cdot H_u}_{L_{y,u} \times L_x} \cdot \underbrace{\left[R_{\boldsymbol{xx}}^{1/2}(1) \vdots R_{\boldsymbol{xx}}^{1/2}(2) \vdots ... \vdots R_{\boldsymbol{xx}}^{1/2}(U) \right]}_{L_x \times U \cdot L_x} \tag{2.461}$$

and consequently the influence of all users. The $\boldsymbol{v}$ is the output of a precoder determined by the MMSE-BC design's Cholesky factor G_u that could process all $(U \cdot L_x)$ inputs ν_u in succession from $U \cdot L_x ... 1$, but from which only the L_x-row subset corresponding to user u finds use for construction of $\boldsymbol{v}_u$ from $\boldsymbol{\nu}_u$. Then $\boldsymbol{v}_u$ forms the u^{th} component of the input $\boldsymbol{x}$, namely $\boldsymbol{x}_u = A_u \cdot \boldsymbol{v}_u$. The design for receiver u ultimately thus uses only L_x rows of the $U \cdot L_x \times U \cdot L_x$ square matrix to determine the $L_x \times U \cdot Lx$ $\mathcal{G}_u$ precoder component for user u. The MMSE approach again for the matrix AWGN (with $\Gamma = 0$ dB codes) corresponds to realization of the best data rate possible for the given order $\mathcal{I}(\boldsymbol{x}_u; \boldsymbol{y}_u / [\boldsymbol{x}_{u+1} ... \boldsymbol{x}_U])$, equivalently $\mathcal{I}(\boldsymbol{x}_u; \boldsymbol{y}_u / [\boldsymbol{v}_{u+1} ... \boldsymbol{v}_U])$ because of the triangular matrix generation using G_u. Thus for the given $\{R_{\boldsymbol{xx}}(u)\}$, H_u, and $R_{\boldsymbol{nn}}(u)$, the performance is canonically maximum and reliably achievable.

Create the Precoder and Receiver Linear Matrix: The MMSE Cholesky factorization based on receiver u's MMSE backward channel model for $\boldsymbol{y}'_u = \widetilde{H}_u^* \cdot \boldsymbol{y}$ follows from, as with the MMSE MAC Equations (2.368) - (2.377):

$$R_{b,u}^{-1} = \widetilde{H}_u \cdot \widetilde{H}_u^* + I = G_u \cdot S_{0,u} \cdot G_u^* \quad , \tag{2.462}$$

an $(L_x U) \times (L_x U)$ factorization. As with the MMSE MAC, the path from $\boldsymbol{v}$ to $(S_{0,u} - I)^{-1} \cdot G_u^{-*} \cdot \boldsymbol{y}'$ produces

$$\boldsymbol{y}''_u = \mathcal{G}_u \cdot \boldsymbol{v} + \boldsymbol{n}''_u \tag{2.463}$$

with unbiased monic, and upper-triangular, precoder

$$G_u \triangleq I + (S_{0,u} - I)^{-1} \cdot S_{0,u} \cdot (G_u - I) \quad . \tag{2.464}$$

User u's relevant SNR entries are in its corresponding L_x rows of $S_{0,u}$. The design ignores the remainder of G_u and $S_{0,u}$'s rows[97]. The MSWMF output's processing $\mathcal{W}_u \cdot \boldsymbol{y}_u$, where $\mathcal{W}_u$ is the $(L_x U) \times L_{y,u}$ matrix

$$W_u = (S_{0,u} - I)^{-1} \cdot G_u^{-*} \cdot \widetilde{H}_u^* \cdot R_{\boldsymbol{nn}}^{-1}(u) \quad . \tag{2.465}$$

Similarly the MSWMF uses only the L_x rows of W_u that correspond to user u's position in the order. The final design aggregates all U of these L_x row subsets of G_u and W_u to form the actual precoder $\mathcal{G}$ and what could be viewed as a block diagonal $\mathcal{W}$.

[97] A design exercise for interested reader would be an algorithm that computes directly on the desired rows and thus uses less complexity.

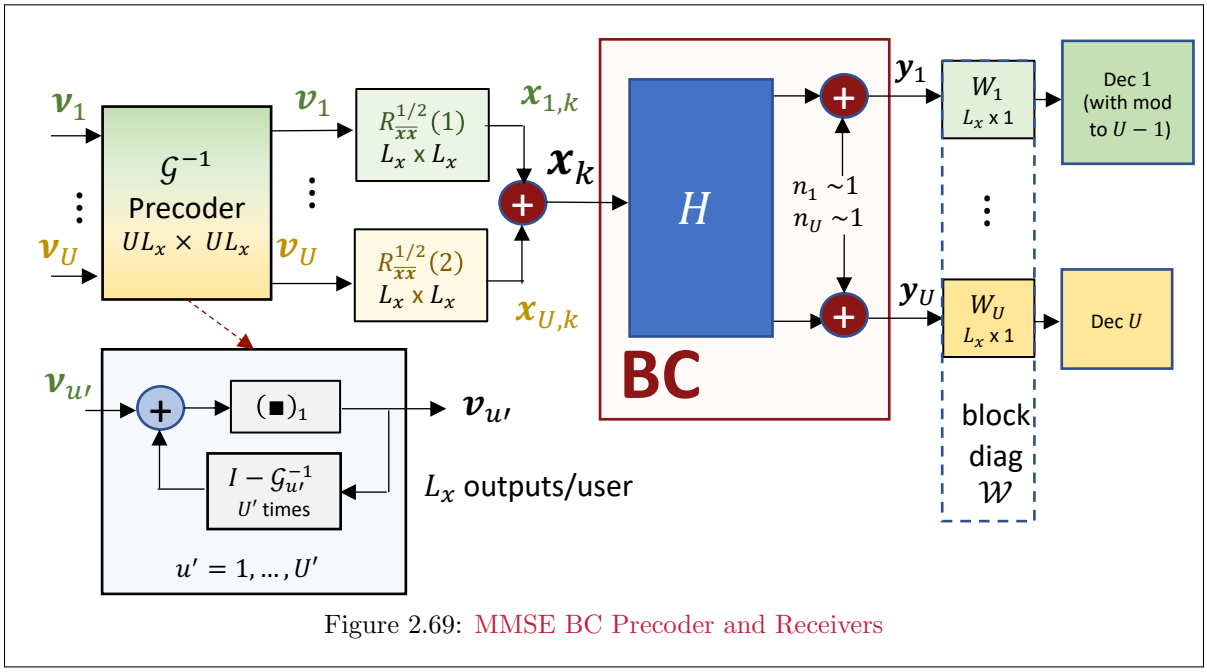


Figure 2.69: MMSE BC Precoder and Receivers

Figure 2.69 illustrates the MMSE precoder and processing. The precoder acts like the feedback section of the MMSE MAC, except that its successive input single-dimensional samples are not decisions but the modulo-device output from previous users' dimensional processing.

The mu_bc.m matlab program The mu_bc.m program is similar to the mu_mac.m program and computes the precoder and receivers' feedforward matrices. The user may select whatever matrix square root desired for the individual user-component $A_u \triangleq R_{\boldsymbol{xx}}^{1/2}(u)$ that are set into an array of such modulators with sizes determined in Lyu. The outputs are the bit vector, unbiased precoder GU $= \{\mathcal{G}_u\}$, unbiased MSWMF(s) stacked in a column of such, and S_0. This mu_bc.m handles directly $\bar{N}$ tones of MT, but that is best left to Chapter 5. For now, $\bar{N} = 1$.

```
function [Bu, GU, S0, MSWMFunb , B] = mu_bc(H, AU, Lyu , cb)
-------------------------------------------------------------------------
Inputs: Hu, AU , Usize, cb
Outputs: Bu, Gunb, Wunb, S0, MSWMFunb

H: noise-whitened BC matrix [H1 ; ... ; HU] (with actual noise, not wcn)
   sum-Ly x Lx x N
AU: Block-row square-root discrete modulators, [A1 ... AU]
    Lx  x (U * Lx) x N
Lyu: # of (output, Lyu) dimensions for each user U ... 1 in 1 x U row vector
cb: = 1 if complex baseband or 2 if real baseband channel

GU: unbiased precoder matrices: (Lx U) x (Lx U) x N
    For each of U users, this is Lx x Lx matrix on each tone
S0: sub-channel dimensional channel SNRs: (Lx U) x (Lx U) x N
MSWMFunb: users' unbiased diagonal mean-squared whitened matched matrices
   For each of U cells and Ntones, this is an Lx x Lyu  matrix
Bu - users bits/symbol 1 x U
   the user should recompute SNR if there is a cyclic prefix
B - the user bit distributions (U x N) in cell array
```

EXAMPLE 2.8.2 *[MMSE BC precoder for the 2-user scalar AWGN with equal energies]* Referring to Example 2.8.1, the MMSE BC precoder simplifies quickly placing half energy on each user with $L_{y,u} = 1$ and $\bar{N} = N = 1$.:

```
>> H =
    80
    50
>> Lyu=[1 1];
>> [Bu, Gunb, S0, MSWMFunb] = mu_bc(H, [1/sqrt(2) 1/sqrt(2)], Lyu, 2)
Bu =
    5.8222    0.4997
>> Gunb{:,:} =
    1.0000    1.0000
---
     0     1
S0 =
  2 x 1 cell array
    {[3.2010e+03]}        User 1 SNR
    {[    1.9992]}        User 2 SNR
MSWMFunb =
  2 x 1 cell array
    {[0.0177]}             multiplies y1
    {[0.0283]}             multiplies y2
>> sum(Bu) =    6.3219
```

This BC's MMSE precoder with equal user energies basically can simplify Figure 2.65 to adding users 1 and 2 prior to the modulo addition with unit energy output and then adding again user 2 with unit energy output before scaling that final sum by $1/\sqrt{2}$ prior to transmission. The MSWMFunb filters are also the values 0.0177 for receiver 1 (followed by unit-energy modulo prior to decoding) and .02833 for receiver 2 that simply accepts user 1's (precoding combination) energy as added noise to its nominal noise.

A second example examines a broadcast channel with two transmitters:

EXAMPLE 2.8.3 *[MMSE BC precoder for the 2-user scalar-output BC with equal energies]* Enlarging to a BC that has two input dimensions with total energy 1, so 1/4 unit (so amplitude 1/2) on each dimension.

```
 H=[50 30
10 20];
>> A =
    0.5000         0    0.5000         0
         0    0.5000         0    0.5000
[Bu, Gunb, S0, MSWMFunb] = mu_bc(H, A, [1 1], 2)

Bu =
    4.8665    0.4971
>> Gunb{:,:} =

    1.0000    0.6000    1.0000    0.6000
         0    1.0000    1.6667    1.0000
  -----------------
```

```
         0         0    1.0000    2.0000
         0         0         0    1.0000
>> S0{:,:} =
  626.0000         0
         0    1.3594
------------------
    1.1984         0
         0    1.6623
>> MSWMFunb{:,:} =
    0.0400
    0.0667
------
    0.2000
    0.1000
>> sum(Bu) =    5.3636
```

User 2 has a low data rate, which suggests that equal energy on both users is not a good choice. Chapter 5 addresses better BC input energy allocations. Each receiver MSWMF expands a single channel output dimension to two dimensions. The decoder processes both dimensions for their desired user. That user uses two successive subsymbols in its symbol. Receiver 1 then also has a modulo operation for each dimension; receiver 2 uses a single modulo operation for the upper dimension, but none for its lower dimension. The precoder is 4×4 and starts with user 2's first dimension, then it's second dimension, working up through user 1's two dimensions respectively.

2.8.3.2 Finding BC Primary- and Secondary-User Components - The Rate-Sum Approach

The WCN design works with only a given $R_{\boldsymbol{xx}}$, specifying only the users' rate sum and not its constituent user data rates. Those rates are left to further design choices, as in Chapter 5. Secondary user components must zero for the maximum-rate sum to occur. Figure 2.68's U' user components map into the primary-user components' encoder input signals $\boldsymbol{v}_1, ..., \boldsymbol{v}_{U^o}$ and the secondary-user components' encoder input signals $\boldsymbol{v}_{U^o+1}, ..., \boldsymbol{v}_{U'}$. The total number of user components is $U' = U^o + U^s$. The secondary-user components must use dimensions that otherwise best carry only primary-user components. The number of channel outputs is

$$\mathscr{L}_y = \sum_{u=1}^{U} L_{y,u} \geq \frac{U'}{L_x} \quad , \tag{2.466}$$

where the bound by U' follows from (2.458); also $L_{y,u} \geq \varrho_{H_u}$. The number of subuser components U' need not include zero-energy dimensions when there is a specified $R_{\boldsymbol{xx}}$. Both $\boldsymbol{y}$ and $\boldsymbol{n}$ are $\mathscr{L}_y \times 1$ vectors. The nonsingular noise's user-indexed components all have standard inverse square-root noise whitening to produce Figure 2.68's $\boldsymbol{y}'_u$, $u = 1, ..., U$ and $\boldsymbol{n}'_u$ where $R_{\boldsymbol{n}'\boldsymbol{n}'}(u) = I_{L_{y,u}}$. This subsection briefly returns to the scalar Gaussian BC for guidance and then extends to the vector case.

Scalar Gaussian BC primary and secondary components/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 $\varrho_H < U'$. Scalar BCs have all users equal to their single components (primary or secondary). Only one scalar BC user/component is primary. In the maximum-rate scalar Gaussian BC, user 1 fully uses all energy as if user 2 is not present. User 1 thus is the primary user/component. User 2 treats user 1 as noise as is thus a secondary user/component. The mixing of secondary user's dimensions into primary users' dimensions occurs at the BC input. These dimensions associate with the components. BC User 1's successive-decoding receiver can decode all primary-user components' 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 2.64 through 2.66. to user 1

as primary and users 2, ..., U as secondary. The scalar Gaussian BC's secondary users/components share the primary user's single component/dimension and reduce rate sum if any have nonzero energy. Any dimension- (time-)sharing's expansion indirectly[98] increases $L_{y,u}$. Such expansion effectively increases channel rank ϱ_H, and input rank ϱ_x, so that the BC is no longer scalar but do not enlarge $\mathcal{C}(\boldsymbol{b})$. Secondary components have smaller contribution to rate sum (at the same energy level) as a primary component.

Designs that consider other users as noise, i.e. those that have at least one secondary-user component, sometimes have the name **Non-Orthogonal Multiple Access (NOMA)**.[99] The set of primary user components is $\boldsymbol{u}^o$. There is always at least 1 primary user component. The set of secondary-user components is $\boldsymbol{u}^s$. The primary set size is $U^o \triangleq |\boldsymbol{u}^o|$, while the secondary set size is $U^s =| \boldsymbol{u}^s |$. As with the energy-sum MAC, $U' = U^o + U^s$, and $U \to U'$, and as well $\boldsymbol{U} \to \boldsymbol{U}'$ if there are sub-users.

Vector Gaussian BC primary- and secondary-user components: The vector BC case uses (2.458) and then compares U' with ϱ_H. If $U' \leq \varrho_H$, the channel is non-degraded. When $U' > \varrho_H$, there are secondary components (if energized, these reduce the BC rate sum). The original-user matrices individually "noise-whiten" at each receiver to ($R_{\boldsymbol{nn}}(u)$, by definition are each non singular.)

$$\widetilde{H}_u \triangleq R_{\boldsymbol{nn}}^{-1/2}(u) \cdot H_u \quad . \tag{2.467}$$

$\widetilde{H}_u$ is a function only of $R_{\boldsymbol{nn}}(u)$, not the full $R_{\boldsymbol{nn}}$, and so dependent only on $R_{\boldsymbol{nn}}(u)$'s (block) diagonal elements. A related matrix is

$$\widetilde{H}_{BC} \triangleq \begin{bmatrix} \widetilde{H}_1 \\ \widetilde{H}_2 \\ \vdots \\ \widetilde{H}_U \end{bmatrix} \quad . \tag{2.468}$$

The rows of $\widetilde{H}_{BC}$ essentially enumerate the users' component set, which is as large as L_x. When $L_{y,u} = 1 \; \forall \; u$, then each row of $\widetilde{H}$ associates with a user/receiver, which assumption helps to understand the following process initially. With this simplifying assumption temporarily, there will be $U^o = \min(\varrho_H, \varrho_x)$ primary user components. Each such scalar-user receiver has output

$$y_u = \tilde{h}_{u,1} \cdot x_1 + ... + \tilde{h}_{u,L_x} \cdot x_{L_x} \quad . \tag{2.469}$$

$\widetilde{H}_{BC}$'s row indices will be $i \in I_{BC}$ and the column indices $j \in J_{BC}$. **If any and all** input dimensions carry non-zero input energy for the maximum rate sum, then an iterative process for finding primary components follows: Since any of the x_u may linearly combine $\nu_{u=1,...,U}$ inputs, the largest element

$$\tilde{h}_{max} = \max_{i,j} \left| \tilde{h}_{i,j} \right| \; \forall \; i \in I_{BC} \wedge j \in J_{BC} \tag{2.470}$$

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 higher-indexed subsequent user component, while decoding without crosstalk interference from other users. Any second primary components' identification (so $U^o \geq 2$) now examines

$$\tilde{h}_{max} = \max_{i,j} \left| \tilde{h}_{i,j} \right|^2 / (\left| \tilde{h}_{i,i_1} \right|^2 + 1) \; \forall \; i \in \{I_{BC} \setminus i_1\} \wedge j \in \{J_{BC} \setminus \{i_1\}\} \tag{2.471}$$

which determines the second primary component as i_2. For yet larger U^o, the search continues now extracting both i_1 and i_2 (generally $i_m, m = 1, ..., U^o$) by finding possible channel gains with the earlier

[98] Indirectly because this sharing splits a dimension into two dimensions, typically through higher-dimensional subsymbols that result from concatenation of several subsymbols that may correspond each to one of the time-dimensionally shared solutions. Dimension sharing increases subsymbol dimensionality (usually in time) through subsymbol packets that use 2 or more codes for respective fractions (of time). Thus ϱ_H and ϱ_x increase and all-primary components may become possible through the enlarged subsymbol packets.

[99] The "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.

users as noise. The maximum determination becomes increasingly complex as the number of primary users increases. The set of primary (and thus secondary) components thus found are invariant to (non-singular) $R_{\boldsymbol{xx}}$, and thus depend only on the channel (and noise).

Lemma 2.8.3 *[Primary- and Secondary-User Component independence of input] The scalar-output ($L_{y,u} = 1$, but $L_x \geq 1$) Gaussian BC's set of primary and secondary components is invariant to nonsingular $R_{\boldsymbol{xx}}$.*

Proof: *For nonsingular $R_{\boldsymbol{xx}}$, see Equations (2.469) - (2.471) above. Singular $R_{\boldsymbol{xx}}$ means that some of the steps in the above input-independent process may misidentify a zero-energy component.* **QED.**

The primary-secondary component pre-qualification-association with maximum data rate renders the scalar ($L_{y,u} = 1$) primary-secondary characterization unique. While it is possible to investigate particular $R_{\boldsymbol{xx}} \neq R_{\boldsymbol{xx}}^{opt}$ that may have different user partitioning that may shift some users for that $R_{\boldsymbol{xx}}$, particularly with a channel that has $\varrho_{\widetilde{H}} < U$, to act like primary components for that $R_{\boldsymbol{xx}}$, this nevertheless does not make those components primary. Because the primary/secondary characterization corresponds to the maximum rate-sum point, the sets $\boldsymbol{u}^o$ and $\boldsymbol{u}^s$ remain invariant to $R_{\boldsymbol{xx}}$.

When the BC outputs individually have multiple dimensions with $L_{y,u} > 1$, the MMSE equivalents must be used in the search in Equation (2.471). This can be complicated, so the next subsection's worst-case noise provides algorithms and software to determine the primary and secondary components.

Worst-Case Noise (WCN) Autocorrelation Approach to Primary/Secondary Component Identification For any linear Gaussian noise channel (multiuser or single-user)

$$\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n} \tag{2.472}$$

with given input autocorrelation matrix $R_{\boldsymbol{xx}}$, there is a worst-case-noise autocorrelation. The overall noise autocorrelation matrix is R_{wcn}, and is such that[100] (where each $R_{wcn}(u)$ is a $L_{y,u} \times L_{y,u}$ matrix):

$$\text{Diag}_{block}\{R_{wcn}\} = \begin{bmatrix} R_{\boldsymbol{nn}}(1) & 0 & ... & 0 \\ 0 & R_{\boldsymbol{nn}}(2) & ... & 0 \\ \vdots & ... & \ddots & 0 \\ 0 & ... & 0 & R_{\boldsymbol{nn}}(U) \end{bmatrix} . \tag{2.473}$$

That is, the diagonal elements match the original noise autocorrelation matrix' diagonal elements. This $\mathscr{L}_y \times \mathscr{L}_y$ matrix R_{wcn} (to be determined below) minimizes mutual information. For the BC, $R_{\boldsymbol{nn}}(u)$, $u = 1, ..., U$ are the given BC noise autocorrelation matrices at each corresponding receiver.

Theorem 2.8.1 (Worst-Case Noise) *For a given linear AWGN with channel matrix H and noise energies specified by (2.473), the mutual information $\mathcal{I}(\boldsymbol{x};\boldsymbol{y})$ has minimum when R_{wcn} satisfies*

$$R_{wcn}^{+} - \left[H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{wcn}\right]^{+} + R_{psd} = \mathcal{S}_{wcn} \tag{2.474}$$

where $\mathcal{S}_{wcn}$ is a $\mathscr{L}_y \times \mathscr{L}_y$ block-diagonal matrix (block sub-entries are $L_{y,u} \times L_{y,u}$), and R_{psd} is an $\mathscr{L}_y \times \mathscr{L}_y$ positive semi-definite matrix that is nonzero when $|R_{wcn}| = 0$. When $|R_{wcn}| > 0$, then (2.474) simplifies to

$$R_{wcn}^{-1} - \left[H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{wcn}\right]^{-1} = \mathcal{S}_{wcn} \quad , \tag{2.475}$$

[100] BC channel indexing reverses order top/left to bottom/right on highest index.

which latter form more typically appears in most WCN treatments that ignore the possible R_{wcn} singularity; typically these treatments pre-condition that $|H \cdot R_{\boldsymbol{xx}} \cdot H^*| > 0$, but said pre-condition is not necessary in general.

Proof: Mutual information minimization first differentiates this convex problem's Lagrangian with respect to the $\mathscr{L}_y \times \mathscr{L}_y$ elements of $R_{\boldsymbol{nn}}$, where these diagonal sub-blocks have side constraints in that Lagrangian. This Lagrangian has $\left(\sum_{u=1}^{U} L_{y,u}^2\right)$ Lagrange multipliers $S_{wcn,i,j}(u)$, where $i = 1, ..., L_{y,u}$, and $j = 1, ..., L_{y,u}$ for each diagonal $R_{\boldsymbol{nn}}(u)$ constraint; R_{psd} is an $\mathscr{L}_y \times \mathscr{L}_y$ Lagrange-multiplier positive-semi-definite matrix that ensures $R_{wcn} \succeq 0$ (positive semidefinite worst-case noise autocorrelation) , and $u = 1, ..., U$. The Lagrangian is

$$\min_{R_{\boldsymbol{nn}}} \mathcal{L}(R_{\boldsymbol{nn}}, \{\mathcal{S}_{wcn,i,j}(u), R_{psd}\}) \quad , \tag{2.476}$$

in bits per complex subsymbol, for all $(i, j) \in \{1, ..., L_{y,u}\} \otimes \{1, ..., L_{y,u}\}$. In more detail:

$$\begin{aligned} \mathcal{L}(R_{\boldsymbol{nn}}, \{\mathcal{S}_{wcn,i,j}(u)\}) \quad &= \quad \underbrace{\log_2 | \, H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{\boldsymbol{nn}} \, | - \log_2 |R_{\boldsymbol{nn}}|}_{\mathcal{I}(\boldsymbol{x};\boldsymbol{y})} \\ &+ \sum_{u=1}^{U} \sum_{i=1}^{L_{y,u}} \sum_{j=1}^{L_{y,u}} [\mathcal{S}_{wcn}(u)]_{i,j} \odot [R_{\boldsymbol{nn},i,j}(u) - R_{wcn,i,j}(u)] \\ &+ \langle R_{\boldsymbol{nn}}, R_{psd} \rangle_F \quad , \end{aligned} \tag{2.477}$$

where $\odot$ denotes Hadamard product of two same-size matrices (matrix of same size that has $(i, j)^{th}$ entry equal the products of the $(i, j)^{th}$ entries of the two matrices, see Appendix C) and $\langle A, B \rangle_F = trace\{A \cdot B^*\}$ is the Frobenius inner product of the two same-size matrices, see Appendix C. Appendix C also provides the scalar function $\ln |R|$'s derivative with respect to (real) symmetric matrix R as

$$2 \cdot R^+ - \text{Diag}(R^+) \quad . \tag{2.478}$$

Thus, the Lagrangian's first term has derivative

$$\begin{aligned} \frac{d\mathcal{I}(\boldsymbol{x};\boldsymbol{y})}{d|R_{\boldsymbol{nn}}|} \quad &= \quad \frac{1}{\log_2 e} \cdot \left\{ 2 \cdot [H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{\boldsymbol{nn}}]^+ - 2 \cdot |R_{\boldsymbol{nn}}|^+ \right\} \\ &+ \text{Diag} \left\{ R_{\boldsymbol{nn}}^+ - [H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{\boldsymbol{nn}}]^+ \right\} \end{aligned} \tag{2.479}$$

The Lagrangian's second-term's derivative is simply a (block over i, j) diagonal matrix Blkdiag $\{\mathcal{S}_{wcn,i,j}(u)\} \triangleq \mathcal{S}_{wcn}$. The Lagrangian's third term similarly has derivative R_{psd}. When $R_{psd} = 0$ (so $|R_{wcn}| > 0$ because this is the positive-definite R_{wcn} constraint is already met), setting the Lagrangian derivative to zero means that the first term's derivative must be a diagonal, specifically $\frac{1}{2}\mathcal{S}_{wcn}$ where the factor of $1/2$ can be absorbed into the Langrange multiplier definition $\mathcal{S}_{wcn} \to \frac{1}{2}\mathcal{S}_{wcn}$. (2.479)'s diagonal components simply then correct the diagonal portion of (2.479), because the off diagonal components must be zero to match $\mathcal{S}_{wcn}$).

$$R_{wcn}^+ - [H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{wcn}]^+ = \mathcal{S}_{wcn} \, . \tag{2.480}$$

When $|R_{wcn}| = 0$, then the mutlipliers $R_{psd} \neq 0$ become active, because R_{wcn} borders constraint violation, and thus

$$R_{wcn}^+ - [H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{wcn}]^+ + R_{psd} = \mathcal{S}_{wcn} \, , \tag{2.481}$$

with any scale factors absorbed into $\mathcal{S}_{wcn}$ and R_{psd}. **QED**.

WCN Solution: (2.481)'s solution finds the Lagrange multipliers S_{wcn} (and R_{psd} when R_{wcn} is singular) through the side (specified $R_{\boldsymbol{nn}}(u)$ and positive semi-definite $R_{\boldsymbol{nn}}$) constraints. Solution only requires the specific additional Lagrange multiplier R_{psd} for positive semi-definite $R_{\boldsymbol{nn}}$ when $|R_{wcn}| = 0$. When $H \cdot R_{\boldsymbol{xx}} \cdot H^* \succ 0$, and thus $|R_{wcn}| > 0$, from (2.479) the mutual information and the constraint are strictly convex, and the solution has a unique minimum. Singular R_{wcn} can occur when $|R_{\boldsymbol{xx}}| = 0$, e.g. for an optimized input. In such situations the determinant $|H \cdot R_{\boldsymbol{xx}} \cdot H^*| = 0$; or H and/or $R_{\boldsymbol{xx}}$ can be such that $|H \cdot R_{\boldsymbol{xx}} \cdot H^*|$ is zero, so WCN also can become arbitrary, including zero, on such dimensions. This means that certain dimensions of $H \cdot R_{\boldsymbol{xx}} \cdot H^*$ with eigendecompositon $\sum_{u=1}^{<U'} s_{x,u} \cdot \boldsymbol{q}_{h,u} \cdot \boldsymbol{q}_{h,u}^*$ carry no energy[101], so $s_{x,u} = 0$ for those dimensions, and WCN also can best use its energy on other dimensions to minimize mutual information.

Mutual information $\mathcal{I}(\boldsymbol{x}; \boldsymbol{y})$ is a maximum (for any $R_{\boldsymbol{xx}}$) rate-sum, minimized over $R_{\boldsymbol{nn}}$ with worst-case noise, which associates with a particular input $R_{\boldsymbol{xx}}$. The rate-sum can have interpretation as a single macro user. Worst-case noise however also distinguishes primary components from secondary components in that only the primary components carry energy if the $\mathcal{I}$ has maximum sum rate over all possible $R_{\boldsymbol{xx}}$. The non-zero Lagrange multipliers ($\mathcal{S}_{wcn}$'s nonzero diagonal entries) represent the different BC receivers' noise-infinitesimal-change affect on this mutual information that nevertheless remains a maximum rate-sum bound for the $R_{\boldsymbol{xx}}$ and $R_{\boldsymbol{nn}}(u)$. $\mathcal{S}_{wcn}$'s zero **diagonal** entries reflect noise components that cannot affect the minimization for a given nonsingular $R_{\boldsymbol{xx}}$ and H. Singularity complicates the secondary-component identification somewhat. Intuitively, such users' BC channel outputs are "useless" in the sense that other (primary) users' outputs support a higher data rate for these secondary-user components (but at the wrong receiver). From the single-user perspective, this is very acceptable. From a multiple-user BC perspective, it means reliable decoding of the secondary-user components at their actual BC outputs reduces their reliably decoded data rates (at all receivers) and the rate sum. This provides intuition for the term "secondary." The following theorem exploits this observation:

Theorem 2.8.2 *[Secondary-user component identification via worst-case noise] The WCN or the maximum BC rate sum identifies secondary components according to the zeroed columns of the RQ* factorization*

$$R_{wcn}^{+} \cdot H = R \cdot Q^*. \tag{2.482}$$

Any permutation of dimensional order needs to be accommodated in such factorization to identify correctly the secondary components.

Proof: The maximum rate sum only energizes primary components. Further in such best rate-sum use, the BC user receivers cannot coordinate the processing of $\boldsymbol{y}_u$. The R matrix has a generalized triangular form where the user receivers correspond to R's rows (or block rows when $L_{y,u} > 1$). The primary components for the maximum rate will have other nonzero energy and thus the R matrix' linearly independent rows can have primary-component crosstalk reduced (almost zeroed in MMSE sense) through transmit lossless precoding. Any other generalized-triangular nonzero rows (zeroed diagonal element) above the full triangular part (nonzero diagonal elements) correspond to user receivers whose non-noise input is linearly dependent on the other receivers' noiseless input components. If the system has maximum rate sum, then those components at the other receivers must be entirely used to improve those corresponding users' data rates. Their rates at any receiver corresponding to the linearly dependent rows must be no greater at these rows' receivers (which will not contribute to the maximum rate sum and need not be decoded at this receiver). Thus the primary components correspond to the linearly independent lower rows of R, while the secondary components are the remaining users in the upper (non-triangular) rows. When upper secondary rows are linearly

[101] Careful inspection of the worst-case noise equation shows that the the noise can be arbitrary, including zero, on these same dimensions, leading to singular R_{wcn}.

dependent on only the top row, there is a"tie" and the last primary and a secondary component can be interchanged.

Permutations will not change the unitary/Hermetian nature of the Q matrix, so these must be tracked to align the zeros that will always otherwise occur in the first positions of the rq command's output with the actual original output indices. This is automatically done in a feature of Matlab's rq.m command. An illustrated use occurs in Example 2.8.4. **QED.**

For other $R_{\boldsymbol{xx}} \neq R_{\boldsymbol{xx}}^{opt}$, there exists an R_{wcn}, and there also can be users who are best decoded at other receivers, rather than their own receivers. The worst-case noise autocorrelation R_{wcn} is not necessarily (block) diagonal, but $\text{Blkdiag}_{block}\{R_{wcn}\} = \text{Blkdiag}_{block}\{R_{\boldsymbol{nn}}\}$. Worst-case noise has other useful implications for the Gaussian BC. Indeed, worst-case noise renders coordination among (primary) users' component receivers useless, as later in this section. Thus any optimum primary-component-only detector that experiences worst-case noise consequently would have independent optimum user/component receivers. The Gaussian BC then basically can perform no better than if the noise were worst case.

In general, there are 3 BC cases that may arise, presuming all inputs, including nonsingular, in particular are possible:

3 Primary/Secondary BC cases:

Perfect MIMO: $U^o = U'$. This case is non-degraded since $U' = \varrho_H = \varrho_x$; there are no secondary components. Perfect MIMO users each have equal number of used transmitter dimensions (antennas) and total number of receiver dimensions (antennas). Perfect MIMO's (all-primary-component) dimensions each have a path largely (MMSE sense) free of other users' crosstalk, as becomes evident shortly. Essentially, all the user components get their own dimension.

Degraded (NOMA): $U^o < U'$. In NOMA, $U^o = \min(\varrho_h, \varrho_x) <= U'$ and secondary components have no dimensions (antennas) to themselves. In this case, energy-sharing (or sharing the secondary components' data rates on common dimensions) is necessary if the secondary components must carry non-zero information. However, the $H_{u \in \boldsymbol{u}^s}$ determine these secondary components' maximum reliably decodable rates, even though the primary-component receivers could reliably decode a higher rate for these secondary components.

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 components may share dimensions to receivers, and there are secondary components.
2. $U < U^o = U'$ (non-degraded enlarged MIMO). In this case, each component has at least one dimension that is largely free (MMSE sense) of crosstalk from other components.

When $U^o >> U$, enlarged (non-degraded) MIMO often has the name "**Massive MIMO**."

2.8.3.3 Maximum BC sum rate

The Gaussian BC channels so far have a specified $R_{\boldsymbol{xx}}$. Optimization of $\mathcal{I}_{wcn}(\boldsymbol{x};\boldsymbol{y})$ over $R_{\boldsymbol{xx}}$ for the corresponding R_{wcn} is indeed possible. This process follows Figure 2.70's iterative process:

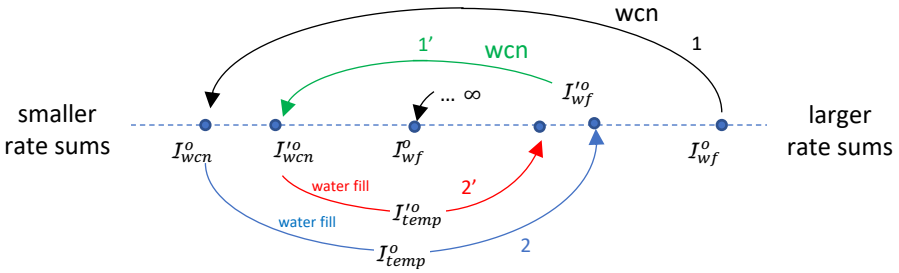


Figure 2.70: Iterative calculation of maximum rate sum.

Iterative Calculation of the BC Maximum Rate Sum:

Waterfill Maximize the single-user $R_{\boldsymbol{xx}}$'s rate-sum maximum with energy-sum constraint This step executes water-fall on the independent primary-components' dimensions and SNR's found from vector coding, which were determined initially for an R_{wcn} based on $R_{\boldsymbol{xx}} = \frac{\mathcal{E}_{\boldsymbol{x}}}{U} \cdot I$ and subsequently for whatever R_{wcn} applies after the next step. The water-fill solution provides energies $\{\mathcal{E}_u\}_{u=U,\ldots,1}$.

Worst Case Noise The previous step now provides a new $R_{\boldsymbol{xx}}$ for worst-case-noise update to R_{wcn}.

Termination Check worst-case noise rate sum versus value at last instance of the worst-case-noise step. If difference is less than some small tolerance, end the algorithm.

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-user components have non-zero energy, so they always carry zero energy for the maximum rate-sum calculation. Thus, this iteration identifies secondary-user components through Theorem 2.8.2. The minimax corresponds to a saddle point.

The bcmax.m program: The bcmax program[102] accepts an initial $R_{\boldsymbol{xx}}$ matrix (only trace is important), individually noise-whitened BC matrices H_u, and $L_{y,u}$ values. The program iterates water-filling and worst-case noise optimizations until convergence. The output is the best $R_{\boldsymbol{xx}}$, corresponding R_{wcn}, and the BC's maximum rate sum. (The parameters N and ν can be ignored or set so $N = 1$ and $\nu = 0$ in this chapter. Chapters 4 and 5 will generalize to other values.)

```
function [Rxx, Rwcn, bmax] = bcmax(iRxx, H, Lyu)

 Uses cvx_wcnoise.m and rate-adaptive waterfill.m (Lagrange
 Multiplier based)
  Arguments:
      - iRxx: initial input autocorrelation array, size is Lx x Lx x N.
          Only the sum of traces matters, so can initialize to any valid
          autocorrelation matrix Rxx to run wcnoise.
          needs to include factor N/(N+nu) if nu ~= 0
      - H: channel response, size is Ly x Lx x N, w/o sqrt(N)
      normalization
      - Lyu: number of antennas at each user
            can create variable-u by just using dummy zero rows in H for
            all output receivers that have less than max (=Lyu input)
  Outputs:
      - Rxx: optimized input autocorrelation
```

[102]This program is based on one first provided by Stanford graduate student Oleksiy Krutko to remove dependencies on need for special packages like CVX.

```
        - Rwcn: optimized worst-case noise autocorrelation, with white local noise
                SO IF H is noise-whitened for Rnn, then actual noise is
                Rwcn^(1/2)*Rnn*Rwcn^(*/2)
        - b: maximum sum rate/real-dimension - user must mult by 2 for
             complex case
 --------
function [En] = waterfill(total_en, gn, gap)

  Waterfill for any set of channel gains, just produces energy

  Water accepts the channel gains as input, and finds water-fill solution
  for the given (input) total energy, with codes of gap "gap."
  The gap is specified as linear (so not in dB).
  The program uses any gain set and does not compute an SNR, nor does it
  know the original of the gains (so no guard-band penalty is presumed)

  Inputs
        total_en is the total energy/symbol for all dimensions
        gn is a vector containing all the gains (energy xfer)
        gap is the LINEAR gap of the code, so gap =1 means 0 dB
  Output
        En is the set of energies for the original dimensions
```

Example 2.8.8 continued for best rate sum The previous singular 3×3 BC has a maximum rate sum given by

```
Rxx =
     3     0     0
     0     4     0
     0     0     2
>> H =
    80    60    40
    60    45    30
    20    20    20
[RxxA, RwcnA, bmax] = bcmax(Rxx, H, 1)
RxxA =
    3.7515    1.5032   -0.7451
    1.5032    1.5019    1.5007
   -0.7451    1.5007    3.7465
RwcnA =
    1.0000    0.7500    0.0008
    0.7500    1.0000    0.0006
    0.0008    0.0006    1.0000
bmax =   12.1084
```

Mohseni's Worst-Case Noise Program Former student Dr. Mehdi Mohseni provides this WCN matlab software wcnoise.m for the case $|R_{wcn}| > 0$ **without** need of the CVX software package. This program can vary the dual_gap (which is the duality gap) from its default 1e-6, while similarly vary nerr (which is Newton's method acceptable error, with default 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 full program is in Appendix G. This program does not include the R_{psd} that is nonzero with singular WCN.

```
function [Rwcn, bsum] = wcnoise(Rxx, H, Ly, dual_gap, nerr)

inputs
H is U*Ly by Lx, where
      Ly is the (constant) number of antennas/receiver,
      Lx is the number of transmit antennas, and
      U is the number of users. H can be a complex matrix
Rxx is the Lx by Lx input (nonsingular) autocorrelation matrix
      which can be complex (and Hermitian!)
dual_gap is the duality gap, defaulting to 1e-6 in wcnoise
nerr is Newton's method acceptable error, defaulting to 1e-4 in wcnoise

outputs
Rwcn is the U*Ly by U*Ly worst-case-noise autocorrelation matrix.
bsum is the rate-sum/real-dimension.

I = 0.5 * log(det(H*Rxx*H'+Rwcn)/det(Rwcn)), Rwcn has Ly x Ly diagonal blocks
    that are each equal to an identity matrix
```

A second more sophisticated wcnoise program allows variable $L_{y,u}$ and also singular $R_{\boldsymbol{xx}}$ as inputs, and is from Dr. Yun Liao. This program uses the CVX package from matlab and thus needs to rewrite the optimization, first recognizing that[103][104] (with $\widetilde{H}$ basically stacking the $\widetilde{H}_u = R_{\boldsymbol{nn}}^{-1/2}(u) \cdot H_u$ to make each individual user-noise constraint the identiy matrix)

$$\begin{aligned}
\mathcal{I}(\boldsymbol{x};\boldsymbol{y}) &= \frac{1}{2} \cdot \log_2 \left(\frac{\left| \widetilde{H} \cdot \widetilde{H}^* + R_{\boldsymbol{nn}} \right|}{|R_{\boldsymbol{nn}}|} \right) && (2.483) \\
&= \frac{1}{2} \cdot \log_2 \left(\left| R_{\boldsymbol{nn}}^{-1/2} \cdot \widetilde{H} \cdot \widetilde{H}^* R_{\boldsymbol{nn}}^{-*/2} + I \right| \right) && (2.484) \\
&= \frac{1}{2} \cdot \log_2 \left(\left| \widetilde{H} \cdot R_{\boldsymbol{nn}} {-} 1 \cdot \widetilde{H}^* + I \right| \right) && (2.485) \\
&= \frac{1}{2} \cdot \log_2 \left[\left| I - \widetilde{H}^* \cdot \left(R_{\boldsymbol{nn}} + \widetilde{H} \cdot \widetilde{H}^* \right)^{-1} \cdot \widetilde{H} \right| \right]^{-1} && (2.486) \\
&= -\frac{1}{2} \cdot \log_2 \left[\left| I - \widetilde{H}^* \cdot \left(R_{\boldsymbol{nn}} + \widetilde{H} \cdot \widetilde{H}^* \right)^{-1} \cdot \widetilde{H} \right| \right] \quad . && (2.487)
\end{aligned}$$

Equation (2.487 allows rewriting the optimization as

$$\begin{aligned}
\underbrace{\text{maximize}}_{R_{\boldsymbol{nn}}} \quad & \tfrac{1}{2} \log_2 |I - Z| \\
\text{subject to:} \quad & R_{\boldsymbol{nn}}(u) = I_{L_{y,u}} \quad u \in \boldsymbol{U} \\
& R_{\boldsymbol{nn}} \succeq 0 \\
& Z \succeq \widetilde{H}^* \cdot \left(R_{\boldsymbol{nn}} + \widetilde{H} \cdot \widetilde{H}^* \right)^{-1} \cdot \widetilde{H}
\end{aligned} \qquad (2.488)$$

[103] (**??** follows because $|I + A \cdot B| = |I + B \cdot A|$.

[104] (2.486) uses matrix-inversion lemma from Appendix A inside the determinant.

The last constraint implies the extra variable matrix Z's positive semidefinite nature in addition to $R_{\boldsymbol{nn}}$. The equivalent Lagrangian to (2.477) is then

$$\begin{aligned}\mathcal{L}(R_{\boldsymbol{nn}}, \{\mathcal{S}_{wcn,i,j}(u)\}, R_{psd}) &= \log_2 |I - Z| \\ &+ \sum_{u=1}^{U} \sum_{i=1}^{L_{y,u}} \sum_{j=1}^{L_{y,u}} [\mathcal{S}_{wcn}(u)]_{i,j} \odot [R_{\boldsymbol{nn},i,j}(u) - R_{wcn,i,j}(u)] \\ &+ \langle R_{\boldsymbol{nn}}, R_{psd} \rangle_F \\ &+ \left\langle \begin{bmatrix} R_{\boldsymbol{nn}} + \widetilde{H} \cdot \widetilde{H}^* & \widetilde{H} \\ \widetilde{H}^* & Z \end{bmatrix}, \underbrace{\begin{bmatrix} S_{4,11} & S_{4,12} \\ S_{4,21} & S_{4,22} \end{bmatrix}}_{S_4} \right\rangle \quad . \end{aligned} \tag{2.489}$$

The last term in (2.489) is sometimes called the Schur compliment, but basically says the matrix on the left must be positive definite, which is the same as Equation (2.488)'s last constraint. The gradient with respect to $R_{\boldsymbol{nn}}$ has only the constraint terms active so then has the block diagonal

$$S_{wcn} = R_{psd} + S_{4,11} \tag{2.490}$$

when the corresponding R_{wcn} is inserted.

This program includes $R_{psd} = S_3$ and S_4; S_1 is the block-diagonal real constraint and S_2 is the imaginary part constraint (imaginary part should be zero).:

```
function [Rnn, sumRatebar, S1, S2, S3, S4] = cvx_wcnoise(Rxx, H, Lyu)

cvx_wcnoise This function computes the worst-case noise for any given input
autocorrelation Rxx and channel matrix.
   Arguments:
       - Rxx: input autocorrelation, size(Lx, Lx)
       - H: channel response, size (Ly, Lx)
       - Lyu: number of antennas at each user, scalar/vector of length U
   Outputs:
       - Rnn: worst-case noise autocorrelation, with white local noise
       - sumRatebar: maximum sum rate/real-dimension
       - S1 is the lagrange multiplier for the real part of Rnn diagonal
              elements.  Zero values indicate secondary-user components.
       - S2 is the imaginary part
       - S3 is for the positive semidefinite constraint on Rwcn
       - S4 is for a larger Schur compliment used in the optimization
```

Appendix G has the full cvx_wcnoise.m program listing.

EXAMPLE 2.8.4 (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 2.67 shows the corresponding rate region. For this channel $R_{\boldsymbol{xx}} = \mathcal{E}_{\boldsymbol{x}} = 1$ and is thus trivially nonsingular. The wcnoise program's third argument is thus $L_{y,u} = 1$ here. This channel has a worst-case noise and the steps to this BC's characterization are:

```
>> H=[80
50] =
    80
    50              (noise-whitened/normalized channel)
>> [Rwcn,b]=wcnoise(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 and it is nonsingular. Input energy 1 is clearly optimum for this trivial scalar case, so maximum data rate is indeed 6.322 bits/subsymbol. Continuing:

```
>>Swcn=inv(Rwcn)-inv(H*H' + Rwcn). =
    0.9998    0.0000
    0.0000    0.0000
>> Htilde=inv(Rwcn)*H =
   80.0000
    0.0000   (zeroed row is secondary-user component)
>> 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!)
```

Since $R_{\boldsymbol{xx}}^{opt} = 1$ trivially, and $R_{wcn} \succ 0$, the observation $\mathcal{S}_{wcn,2} = 0$ confirms user 2 as entirely secondary. The designer can share the one useful dimension from both users' inputs to the BC outputs, with a precoder or a successive decoder. The best MMSE worst-case-noise design assumes all energy on user 1 because it focuses only on the rate sum. The program's rate-sum output $b = 6.3220$ is the same as Examples 2.8.4, 2.8.6, and 2.8.2's rate sum that occurs for $R_{\boldsymbol{xx}} = 1$ from previous work on this same example. Neither the wcnoise programs, nor the worst-case noise concept from which they arise, specify to which users the energy maps. In general, the WCN approach does not individually allocate b_1 and b_2, although in this trivial case $b_1 = 6.322$ and $b_2 = 0$.

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 component reliably as can user 1, but energy on user 2 reduces the sum rate and user 1's data rate below 6.322. Worst-case noise simply implies correctly that receiver 2 is not necessary for the best sum rate $\mathcal{I}_{wcr}(\boldsymbol{x}; \boldsymbol{y})$, 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. Alternately, MMSE Design's mu_bc.m can be used, but requires the specification of all $\{R_{\boldsymbol{xx}}(u)\}$, so the designer there must first allocate exact energy to each user. Allocation in mu_bc.m to user 1 achieves 6.322.

The mu_bc.m program with maximum-rate-sum energy allocation follows:

```
>> AU=[1 0 ; 0 0];
>> [Bu, GU, S0, MSWMFunb , B] = mu_bc(H, AU, 1 , 2)
Bu =    6.3220
GU{:,:} =     1     0
>> S0{:,:} =   6.4010e+03
>> MSWMFunb{:,:} =    0.0125         0
B =    {[6.3220]} ,
```

which indicates that the zeroed energy of user 2 essentially reduces the precoder to one dimension, but shows the zero contribution from the other (so the precoder is a kind of trivial monic triangular, but nonsquare) and the SNR and bit rate are correct. the MSWMFunb again shows that the best MMSE receiver strategy ignores receiver 2 in terms of data rate.

A complex 4×4 Gaussian BC example with rank-3 channel and rank-2 input further illustrates aspects beyond the simple Example 2.8.4.

EXAMPLE 2.8.5 *[Singular BC and Singular $R_{\boldsymbol{xx}}$]* The complex Gaussian BC has 4 scalar outputs and a 4-dimensional vector input (which combines 4 users into $\boldsymbol{x}$). The channel and

$R_{\boldsymbol{xx}}$ appear below in matlab. The channel has rank 3 and the input has rank 2 so $U^o \geq 2$ and there should be 3 primary and 1 secondary components; however recall primary/secondary corresponds to a maximum rate sum (which is not necessarily the case for a given $R_{\boldsymbol{xx}}$ as is the case here). A first pass at this channel uses the given $R_{\boldsymbol{xx}}$.

```
>> H =
   0.4054 - 0.1990i   0.3641 + 0.6869i   3.6004 + 0.5569i   0.5318 + 0.0080i
   1.8406 + 1.3469i  -1.3014 - 1.1630i   2.7217 + 1.0820i   0.0947 - 1.0710i
  -2.3367 + 1.1594i  -0.3949 + 0.7899i  -1.4024 + 0.8380i   0.8085 + 0.3019i
   1.1210 + 1.4423i   0.2611 + 1.6376i   2.9534 - 0.3945i   0.2962 - 0.8347i

>> Rxx =
   0.1382 + 0.0000i   0.0077 - 0.0664i  -0.0701 + 0.0449i  -0.0594 - 0.0207i
   0.0077 + 0.0664i   0.2005 + 0.0000i  -0.0155 - 0.0152i   0.0281 - 0.0130i
  -0.0701 - 0.0449i  -0.0155 + 0.0152i   0.0522 + 0.0000i   0.0262 + 0.0288i
  -0.0594 + 0.0207i   0.0281 + 0.0130i   0.0262 - 0.0288i   0.0331 + 0.0000i
>> rank(H) =     3
>> rank(Rxx) =     2
```

Simple 4-digit display can hide numerical errors that increase the rank. So the reader reproducing this example (by pasting it in on your computer) should perform singular value decomposition on H and Rxx and eliminate the row/column vectors corresponding to the smallest singular value (should be very close to zero) and the two smallest singular values of Rxx (both are close to zero) before proceeding.

```
[V, D]=eig(Rxx);
D =
   -0.0001         0         0         0
         0    0.0001         0         0
         0         0    0.1418         0
         0         0         0    0.2821
Rxx=V(:,3)*V(:,3)'*D(3,3)+ V(:,4)*V(:,4)'*D(4,4);
rank(Rxx) = 2
[F,L,M]=svd(H);
L =
    6.7116         0         0         0
         0    3.1056         0         0
         0         0    2.0988         0
         0         0         0    0.0000
H=F(:,1)*M(:,1)'*L(1,1)+ F(:,2)*M(:,2)'*L(2,2)+F(:,3)*M(:,3)'*L(3,3);
rank(H) =     3

>> [Rwcn, b , S1, S2, S3, S4] = cvx_wcnoise(Rxx, H, ones(1,4))

Rwcn =
   1.0000 + 0.0000i  -0.5084 + 0.0458i   0.1251 + 0.5916i  -0.0042 + 0.3065i
  -0.5084 - 0.0458i   1.0000 + 0.0000i  -0.4394 + 0.2473i  -0.6341 + 0.0932i
   0.1251 - 0.5916i  -0.4394 - 0.2473i   1.0000 + 0.0000i   0.7215 + 0.3923i
  -0.0042 - 0.3065i  -0.6341 - 0.0932i   0.7215 - 0.3923i   1.0000 + 0.0000i
b =    0.8241
S1 =  4 x 1 cell array
    {[    0.2288]}
    {[    0.3149]}
    {[    0.3073]}
```

```
    {[5.9295e-09]}
S2 =  4 x 1 cell array
    {[0]}
    {[0]}
    {[0]}
    {[0]}
S3+S4(1:4,1:4) =
    0.2288         0         0         0
         0    0.3149         0         0
         0         0    0.3073         0
         0         0         0    0.0000
>> pinv(Rwcn)*H =   1.0e+09*
   1.0647 + 0.7898i  -0.1492 + 0.2605i   2.1940 + 0.5313i   0.1632 - 0.5248i
   0.4982 + 1.1166i  -0.2378 + 0.1418i   1.5228 + 1.4199i   0.3687 - 0.3479i
  -0.6578 + 1.1464i  -0.2754 - 0.1173i  -0.2699 + 2.2345i   0.5387 + 0.1003i
  -0.0000 - 0.0000i   0.0000 - 0.0000i  -0.0001 - 0.0000i  -0.0000 + 0.0000i
```

So, the bottom user (4 in this text's BC indexing) is better decoded at other receivers since $S_{wcn}(4,4) = 0$; however the rate sum is not best whether or not user 4 is energized with this $R_{\boldsymbol{xx}}$. For this channel another rank-3 $R_{\boldsymbol{xx}}$ corresponds to the maximum rate sum (this can be found through the bcmax.m software). The upper 4×4 entries of S3 + S4 match the S1 entries as should always be the case, and both indicate user 4 is not helping rate sum, which also indicate user 4 as the secondary-user component but again this channel has singular WCN so that user 4 as a secondary component is not definitive.

To find the $R_{\boldsymbol{xx}}$ corresponding to maximum rate sum, the bcmax program produces:

```
>> [Rxxopt, Rwcnopt, bmax] = bcmax(eye(4), H, 1)

Rxxopt =
   1.1187 + 0.0000i   0.0353 - 0.0463i   0.1305 + 0.0349i  -0.4055 - 0.2217i
   0.0353 + 0.0463i   1.2565 - 0.0000i  -0.0312 - 0.0639i   0.1240 + 0.0690i
   0.1305 - 0.0349i  -0.0312 + 0.0639i   1.3569 + 0.0000i   0.2026 - 0.0858i
  -0.4055 + 0.2217i   0.1240 - 0.0690i   0.2026 + 0.0858i   0.2680 + 0.0000i

Rwcnopt =
   1.0000 + 0.0000i  -0.0057 + 0.0471i  -0.1909 + 0.1583i   0.2870 + 0.1204i
  -0.0057 - 0.0471i   1.0000 + 0.0000i  -0.2764 - 0.1283i   0.0466 + 0.2162i
  -0.1909 - 0.1583i  -0.2764 + 0.1283i   1.0000 + 0.0000i   0.2482 + 0.6487i
   0.2870 - 0.1204i   0.0466 - 0.2162i   0.2482 - 0.6487i   1.0000 + 0.0000i

bmax =    5.7221  > 0.8242
(Ensure no numerical issues by forcing a zeroed eigenvalue:)
>> [Vopt, Dopt]=eig(Rxxopt)

Vopt =
   0.3519 + 0.1746i   0.2952 + 0.4505i   0.4330 - 0.1522i  -0.2021 - 0.5515i
  -0.0986 - 0.0746i   0.7104 + 0.0000i  -0.0837 - 0.0805i   0.6830 + 0.0000i
  -0.1774 + 0.0516i   0.0573 - 0.3628i   0.8594 + 0.0000i   0.0258 + 0.3029i
   0.8923 + 0.0000i  -0.0937 - 0.2481i   0.0139 + 0.1926i   0.2507 + 0.1825i
Dopt =
   0.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   1.2144 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   1.4479 + 0.0000i   0.0000 + 0.0000i
```

```
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.3376 - 0.0000i
>> Rxxopt=Vopt(:,2)*Vopt(:,2)'*Dopt(2,2)+Vopt(:,3)*Vopt(:,3)'*Dopt(3,3)+...
Vopt(:,4)*Vopt(:,4)'*Dopt(4,4);
>> rank(Rxxopt) =     3
>> [Vwcn,Dwcn]=eig(Rwcnopt);
Dwcn =
    0.0000         0         0         0
         0    1.0344         0         0
         0         0    1.2568         0
         0         0         0    1.7087

>> Rwcnopt=Vwcn(:,2)*Vwcn(:,2)'*Dwcn(2,2)+Vwcn(:,3)*Vwcn(:,3)'*Dwcn(3,3)+...
Vwcn(:,4)*Vwcn(:,4)'*Dwcn(4,4);
>> rank(Rwcnopt) =     3
```

Again the Rxxopt and Rwcnopt can have very small eigenvalues (or singular) values and the ones very close to zero should be removed to best follow this example and avoid finite-precision errors clouding understanding.

Nominally, Matlab's qr.m, and our added rq.m program, create the desired factorization. However, the WCN-BC design's factorization is of a matrix $R_{wcn}^{-1} \cdot H$ that does not exist anywhere in actual implementation. H is the channel and has an order associated with its input and output dimensions. The rq.m command will always produce a generalized upper triangular matrix that makes the top user appear as entirely a secondary component if the input matrix is $R_{wcn}^{-1} \cdot H$; however, the dimensions have been reordered. Matlab's additional output argument P in [R, Q, P] =rq(matrix) summarizes the reordering in that matrix(P,:)=RQ*. Matlab also labels the top row as user 1, same as BC if there were no P reordering (P=[1 2 3 ... U]). Since the RQ are what the WCN-BC design uses, this is the new order. Thus P relates that order.

CAUTION - Matlab's Reordering:

```
 >> Htilde=pinv(Rwcnopt)*H =
   0.1617 - 0.1930i   0.4556 + 0.5176i   2.8679 + 0.5764i   0.5089 + 0.0885i
   1.4786 + 1.1517i  -1.1892 - 1.1034i   2.1957 + 0.7506i   0.0830 - 0.9427i
  -1.4599 + 0.6341i  -0.1581 + 0.4667i  -1.0483 + 0.2059i   0.4469 + 0.1754i
   0.7309 + 0.6790i   0.0985 + 1.1198i   1.9228 - 0.5057i   0.1148 - 0.4906i

>> [R,Q,P]=rq(Htilde)

[R,Q,P]=rq(Htilde)

R =
  -0.0000 + 0.0000i  -0.5119 - 1.2144i   0.2833 - 0.0287i   1.1435 - 1.0112i
   0.0000 + 0.0000i  -1.3253 + 0.0000i  -1.3058 + 0.3437i  -1.3686 + 1.0062i
   0.0000 + 0.0000i   0.0000 + 0.0000i  -2.6014 + 0.0000i  -1.5866 + 0.2793i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i  -3.5249 + 0.0000i
Q =
  -0.3854 + 0.0764i  -0.1548 + 0.6591i   0.2288 - 0.2284i  -0.4195 + 0.3267i
   0.1236 - 0.0010i  -0.1603 + 0.6711i  -0.4145 + 0.3536i   0.3374 - 0.3130i
   0.1091 - 0.1491i   0.0846 - 0.1033i  -0.6997 + 0.1586i  -0.6229 + 0.2130i
  -0.7070 + 0.5444i  -0.0067 - 0.2184i  -0.2100 + 0.1997i  -0.0235 - 0.2674i
P =  3     4     1     2
```

The original (non optimum) $R_{\boldsymbol{xx}}$ essentially zeroed primary user component 4 (bottom), allowing another non-zero-energy user to occupy its passage, making another user component appear artificially primary and user component 4 as secondary.

```
>> eig(Rxx) =
  -0.0000 - 0.0000i
   0.2821 - 0.0000i
   0.1418 - 0.0000i
  -0.0000 + 0.0000i
```

Indeed, the original $R_{\boldsymbol{xx}}$ had rank 2 and happened to zero both user 1's and 4's (top and bottom) input energy. That original $R_{\boldsymbol{xx}}$ had a much lower data rate as well. The maximum rate is actually 5.7221 bits/symbol for this BC. In this case, user 3's component is secondary, and both user 1 and user 4 components carry energy for maximum data rate. Indeed the order is user 4 in best (top) position, then user 1, and finally user 2. This best rate sum actually occurs when (as with bcmax) the input is waterfill for the equivalent channel with WCN for that waterfill input.

The example continues with a different $R_{\boldsymbol{xx}}$ with rank less than the rank of H and what looks initially like 2 secondary user/components

[Singular BC and Singular $R_{\boldsymbol{xx}}$] The same channel H with a different rank-2 input is

```
>> H =
   0.4054 - 0.1990i   0.3641 + 0.6869i   3.6004 + 0.5569i   0.5318 + 0.0080i
   1.8406 + 1.3469i  -1.3014 - 1.1630i   2.7217 + 1.0820i   0.0947 - 1.0710i
  -2.3367 + 1.1594i  -0.3949 + 0.7899i  -1.4024 + 0.8380i   0.8085 + 0.3019i
   1.1210 + 1.4423i   0.2611 + 1.6376i   2.9534 - 0.3945i   0.2962 - 0.8347i
>> Rxx =
   0.3189 + 0.0000i   0.0108 - 0.2816i  -0.0423 + 0.0438i  -0.1069 - 0.0815i
   0.0108 + 0.2816i   0.3222 + 0.0000i  -0.0938 - 0.1478i   0.0593 - 0.1102i
  -0.0423 - 0.0438i  -0.0938 + 0.1478i   0.2222 + 0.0000i   0.0296 + 0.0212i
  -0.1069 + 0.0815i   0.0593 + 0.1102i   0.0296 - 0.0212i   0.0601 + 0.0000i
>> rank(H) =     3
>> rank(Rxx) =     2

>> [Rwcn, b , S1, S2, S3, S4] = cvx_wcnoise(Rxx, H, ones(1,4))

Rwcn =

   1.0000 + 0.0000i   0.1553 - 0.3908i   0.1513 + 0.0367i   0.6484 + 0.4078i
   0.1553 + 0.3908i   1.0000 + 0.0000i  -0.7047 - 0.3153i   0.0346 + 0.0185i
   0.1513 - 0.0367i  -0.7047 + 0.3153i   1.0000 + 0.0000i   0.2524 + 0.4824i
   0.6484 - 0.4078i   0.0346 - 0.0185i   0.2524 - 0.4824i   1.0000 + 0.0000i

b =    2.4349
S1 =  4 x 1 cell array
    {[2.8091e-09]}
    {[    0.8172]}
    {[1.5658e-08]}
    {[    0.8073]}
S2 =  4 x 1 cell array
    {[0]}
    {[0]}
    {[0]}
```

```
    {[0]}
S3 =  1.0e-08 *
   0.1315 + 0.0000i  -0.0140 + 0.0493i  -0.0007 + 0.0005i  -0.0770 - 0.0520i
  -0.0140 - 0.0493i   0.5503 + 0.0000i   0.5679 + 0.2363i  -0.0563 - 0.3012i
  -0.0007 - 0.0005i   0.5679 - 0.2363i   0.7919 + 0.0000i  -0.2245 - 0.3785i
  -0.0770 + 0.0520i  -0.0563 + 0.3012i  -0.2245 + 0.3785i   0.3652 + 0.0000i

>> S3+S4(1:4,1:4) =
    0.0000         0         0         0
         0    0.8172         0         0
         0         0    0.0000         0
         0         0         0    0.8073
```

So, users 1 and 3 are better decoded at other receivers since $S_{wcn}(1,1) = 0$ and $S_{wcn}(3,3) = 0$; however the rate sum is not best. Again the secondary-user component is user 3 from the bcmax exercise above, which was included this time.

(Secondary-user components with singular input)

```
>> H
    1.0719   -0.8627   -0.1901    0.2952
    1.0498   -0.7245    0.2568    0.2757
   -0.4586    0.5595    1.0027    0.0530
    0.4107    0.0496    0.3965   -0.7740
[F,L,M]=svd(H);
L =
    2.0671         0         0         0
         0    1.1449         0         0
         0         0    0.8130         0
         0         0         0    0.0000
H=F(:,1)*M(:,1)'*L(1,1)+ F(:,2)*M(:,2)'*L(2,2)+F(:,3)*M(:,3)'*L(3,3);
>> H13 = H(1:3,:);
>> H24=H(2:4,:);
index124=[1 2 4];
index134=[1 3 4];
H124=H(index124,:);
H134=H(index134,:);

>> [Rxxopt, Rwcnopt, bmax] = bcmax(eye(4), H13, 1)
Rxxopt =
    1.3528   -0.9949    0.0527    0.3143
   -0.9949    0.8134    0.3356   -0.1731
    0.0527    0.3356    1.7189    0.2790
    0.3143   -0.1731    0.2790    0.1149
Rwcnopt =
    1.0000    0.9064   -0.4843
    0.9064    1.0000   -0.0950
   -0.4843   -0.0950    1.000
bmax = 2.0408

>>  [Rxxopt, Rwcnopt, bmax] = bcmax(eye(4), H24, 1)

Rxxopt =
```

```
    1.1514   -0.7316    0.1121   -0.0693
   -0.7316    0.6322    0.2787   -0.2159
    0.1121    0.2787    1.3264   -0.0207
   -0.0693   -0.2159   -0.0207    0.8900
Rwcnopt =
    1.0000   -0.1318    0.1136
   -0.1318    1.0000    0.1047
    0.1136    0.1047    1.0000
bmax =    2.1903

>>  [Rxxopt, Rwcnopt, bmax] = bcmax(eye(4), H134, 1)

Rxxopt =
    1.1632   -0.7640   -0.0317   -0.1158
   -0.7640    0.6848    0.3747   -0.2344
   -0.0317    0.3747    1.1936   -0.1357
   -0.1158   -0.2344   -0.1357    0.9584
Rwcnopt =
    1.0000   -0.2779    0.0653
   -0.2779    1.0000    0.1142
    0.0653    0.1142    1.0000
bmax =   2.1472

>> [Rxxopt, Rwcnopt, bmax] = bcmax(eye(4), H124, 1)

Rxxopt =
    1.4923   -0.9375    0.1705   -0.1023
   -0.9375    0.7922    0.1570   -0.3967
    0.1705    0.1570    0.5258   -0.4655
   -0.1023   -0.3967   -0.4655    1.1897
Rwcnopt =
    1.0000    0.7524   -0.0328
    0.7524    1.0000    0.1628
   -0.0328    0.1628    1.0000
bmax =    1.9497
```

Running on the full H also confirms the same maximum rate sum, although the $R_{\boldsymbol{xx}}^{opt}$ and R_{wcn} are different.

```
>> [Rxxopt, Rwcnopt, bmax] = bcmax(eye(4), H, 1)

Rxxopt =
    1.1559   -0.7381    0.0995   -0.0691
   -0.7381    0.6399    0.2862   -0.2207
    0.0995    0.2862    1.3091   -0.0326
   -0.0691   -0.2207   -0.0326    0.8951
Rwcnopt =
    1.0000    0.9163   -0.4792   -0.0191
    0.9163    1.0000   -0.0991    0.1251
   -0.4792   -0.0991    1.0000    0.1044
   -0.0191    0.1251    0.1044    1.0000
bmax =    2.1911
```

One of the eigenvalues of each is very close to zero, so we use the cvx_wcnoise program that

works even when the worst-case noise is singular.

```
>> inv(Rwcnopt) - inv(H*Rxxopt*H'+Rwcnopt)
 -69.9931   62.4186  -26.7687   -6.3512
   62.4186  -54.9656   23.8385    5.6560
  -26.7687   23.8385   -9.5873   -2.4256
   -6.3512    5.6560   -2.4256   -0.1050

NOT DIAGONAL (subject to numerical issues so different runs may produce different results)
>> [Rwcnopt, sumRatebar, S1, S2, S3, S4] = cvx_wcnoise(Rxxopt, H, [1 1 1 1])

Rwcnopt =
    1.0000    0.9163   -0.4792   -0.0191
    0.9163    1.0000   -0.0991    0.1251
   -0.4792   -0.0991    1.0000    0.1044
   -0.0191    0.1251    0.1044    1.0000
sumRatebar =    2.1911

>> S3+S4(1:4,1:4) =
    0.0978         0         0         0
         0    0.6204         0         0
         0         0    0.6360         0
         0         0         0    0.4705
```

Because $\rho_H = 3$, there is one secondary user/component, and thus it would be the top usercomponent (user 1 in our BC notation and this example. Design may now proceed for the GDFE/precoder for the channel H24, effectively zeroing receiver 1's processing because any energy that does go into receiver 1 is not sufficient to decode reliably the larger data rate that energy allows in sum across the other users.

```
----- ON THE RQ -------------
>> [Vopt,Dopt]=eig(Rwcnopt)

Vopt =
   -0.7169   -0.0870   -0.0146   -0.6915
    0.6379   -0.3439    0.2918   -0.6242
   -0.2736   -0.7165    0.5295    0.3626
   -0.0649    0.6007    0.7964   -0.0250
Dopt =
    0.0000         0         0         0
         0    0.8067         0         0
         0         0    1.1156         0
         0         0         0    2.0778

(NOTE that the first eigenvalue is zero for user at top, consistent with only 3 users excited
for maximum sum rate on this channel, although this does not say which user is secondary.)

>> Rwcnopt=Dopt(2,2)*Vopt(:,2)*Vopt(:,2)'+Dopt(3,3)*Vopt(:,3)*Vopt(:,3)'+Dopt(4,4)*Vopt(:,4)*Vo

Rwcnopt =
    1.0000    0.9163   -0.4794   -0.0191
    0.9163    1.0000   -0.0992    0.1251
   -0.4794   -0.0992    1.0000    0.1044
   -0.0191    0.1251    0.1044    1.0000
```

```
rank(Rwcnopt) ans =     3
>> Htilde=pinv(Rwcnopt)*H =
    0.5056   -0.4129   -0.0607    0.1863
    0.5191   -0.3199    0.3795    0.2332
   -0.2041    0.3248    0.9857    0.2517
    0.3768    0.0478    0.2450   -0.8259

>> [R,Q,P]=rq(Htilde)

R =
   -0.0000    0.6416    0.0041    0.2302
         0    0.7210   -0.0912   -0.2051
         0         0   -0.9411    0.0255
         0         0         0   -1.0872
Q =
    0.5342    0.7233   -0.3953    0.1877
    0.7873   -0.5361   -0.0589   -0.2988
   -0.2071    0.2325   -0.2849   -0.9066
    0.2278    0.3679    0.8713   -0.2315

P =     1     2     4     3
```

Again recalling that matlab has the same order as BC, this confirms that user 1 is secondary (as evident also in the Rxxopt's eigendecomposition). The order for maximum sum rate is user 2 in last precoder position, user 4 next to last, and user 3 first. User 1 freeloads as a secondary user if it were energized (which it is not for maximum data rate) and would freeload on the other 3 users' best-used dimensions.

2.8.3.4 Precoder Generalization

Primary/secondary component characterization abstracts a higher-level precoder (and corresponding decoder) architectural understanding. The BC transmitter uses a square root to process an energy/dimension-normalized ($R_{\boldsymbol{vv}} = I$) input vector $\boldsymbol{v}$ from the precoder output and to transform it into the correct $R_{\boldsymbol{xx}}$. Square roots are not unique, and indeed this generalized approach finds an $R_{\boldsymbol{xx}}$ that satisfies

$$R_{\boldsymbol{xx}} = A \cdot R_{\boldsymbol{vv}} \cdot A^* = A \cdot A^* \quad , \tag{2.491}$$

where A is a special square root that depends on both R_{wcn} and $R_{\boldsymbol{xx}}$, or equivalently on $R_{\boldsymbol{xx}}$ and the BC $\widetilde{H}$. This A need not decompose into a specific $A = \left[A_1 \vdots A_2 \vdots \ldots \vdots A_U\right]$ and instead characterizes only the sum $R_{\boldsymbol{xx}}$. 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. This A effectively presumes primary components share their dimensions, with consequently their own and total rate-sum reduction, with energized secondary components.

Precoders for the Vector BC: Figure 2.71 generalizes Figure 2.65 to the vector BC case. The primary and secondary user-component precoders[105] both appear as generating identity autocorrelation matrices with respective fractions of primary- and secondary-component energy, so $R_{\boldsymbol{vv}} = I$, while $R^o_{\boldsymbol{vv}} = \alpha \cdot I$ and $R^s_{\boldsymbol{vv}} = (1-\alpha) \cdot I$. While secondary user components can always be reliably decoded at primary-user components' receivers, the secondary-user components' data rates are determined by the lowered data rates reliably decoded at their own receivers.

[105] Again, determined with noise-whitening but only pure energy constraint for any given set of (block) diagonal elements $\{R_{\boldsymbol{nn}}(u)\}$.

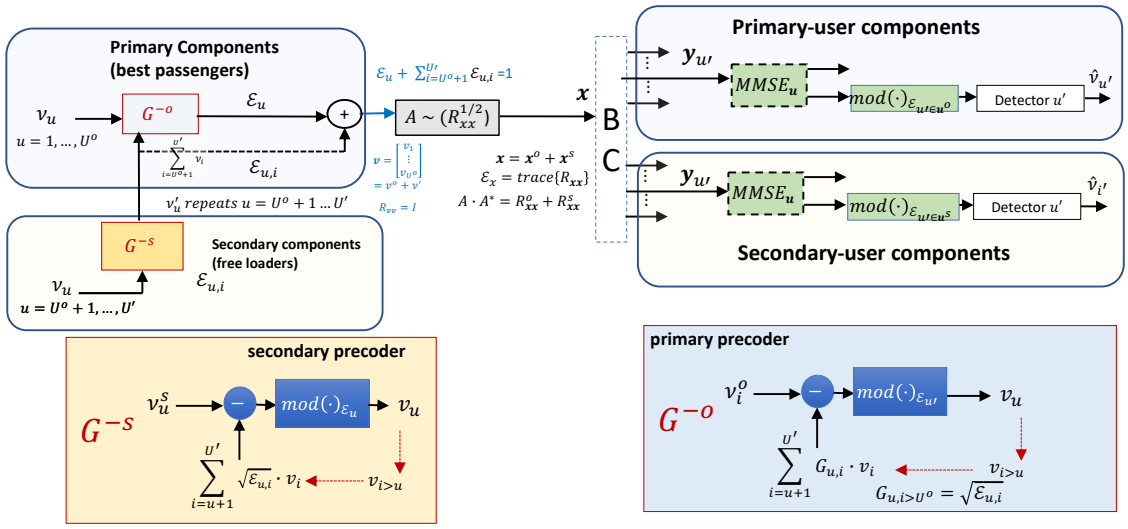


Figure 2.71: Vector Gaussian BC Precoders.

Subsection 2.8.3.5 shows that a MMSE design largely cancels that crosstalk between primary components' dimensional energies (be those energies all primary or shared with secondary). The crosstalk caused by secondary user components' sharing of primary components' dimensions, however, is not cancelled.

The precoder input remains $\boldsymbol{\nu}$ and the precoder output $\boldsymbol{v}$. Both primary and secondary precoders process input user components recursively, first with decreasing index from $u = U'$ to $U^o + 1$ in the secondary precoder G^{-s} and then from U^o to 1 in the primary precoder. Figure 2.71's transmitter's square-root matrix A then converts $R_{\boldsymbol{vv}} = I$ to $R_{\boldsymbol{xx}}$ on the channel input, per requirement. Individual primary sub-user energy components are for primary components $\mathcal{E}_i \; \forall \, i \in \boldsymbol{u}^o$ and

$$1 = \mathcal{E}_u + \sum_{i=U^o+1}^{U'} \mathcal{E}_{u,i} \; \forall \; u = 1, ..., U^o \quad , \tag{2.492}$$

where $\mathcal{E}_{u,i}$ is the energy from secondary users $i = U^o + 1, ..., U'$ that superimposes upon primary components' dimensions. Any amplification to the actual $\mathcal{E}_{\boldsymbol{x}}$ value occurs through the $A = R^{1/2}_{\boldsymbol{xx}}$ linear precoder. While the primary/secondary approach provides insight in analogy with the simple 2-user scalar BC, the entire set of $\{R_{\boldsymbol{xx}}(u)\}$ is necessary for reasonable design complexity. Section 5.5 combines the insights and the simple design for any point in the capacity region.

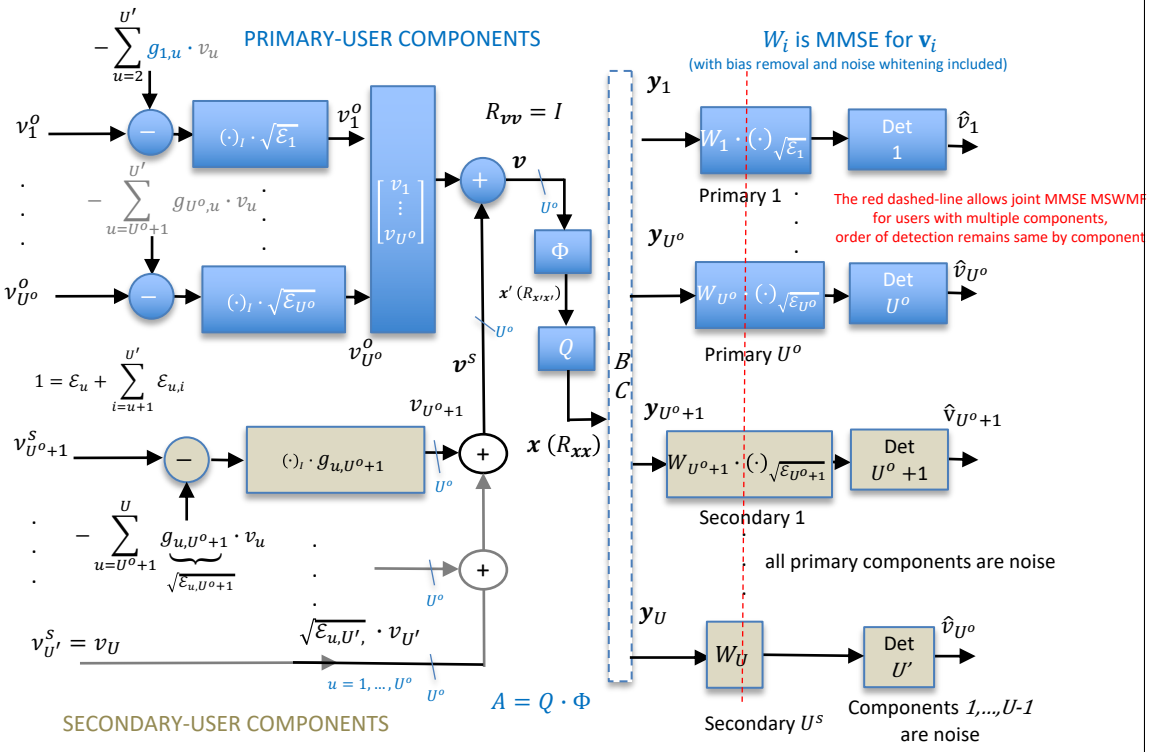


Figure 2.72: General Gaussian BC Precoder and corresponding receiver.

The primary-group components' rate sum b^o is the same for all orders within the primary-component set $\boldsymbol{u}^o$ if the secondary set $\boldsymbol{u}^s$ has zero energy $\mathcal{E}^s = 0$ (or equivalently the BC becomes non-degraded with only primary user components). Order within the primary-component set affects each primary-component individual rate $\{b_u \mid u \in \boldsymbol{u}^o\}$, but the sum $b^o = \sum_{u=1}^{U^o} b_u$ remains constant for the given $R_{\boldsymbol{xx}}$. The secondary-component set's individual rates $\{b_u \mid u \in \boldsymbol{u}^s\}$ and rate sum $b^s = \sum_{u=U^o+1}^{U'} b_u$ both vary with secondary set's energy apportionment from the energy-possibility set (or $\{R_{\boldsymbol{xx}}(u)\}$ set)

$$\{\mathcal{E}_x^s\} = \left\{ trace\{R_{\boldsymbol{xx}}(s,u)\} \mid u \in \boldsymbol{u}^s \wedge \left[R^s_{\boldsymbol{xx}} = \sum_{u=1}^{U^s} R_{\boldsymbol{xx}}(s,u) \right] \wedge [R_{\boldsymbol{xx}} = R^o_{\boldsymbol{xx}} + R^s_{\boldsymbol{xx}}] \right\} . \quad (2.493)$$

There is (at least) one overall order, that achieves best rate sum for any input energy assignment (i.e., the θ_i's and α choices). Subsection 2.8.3.3 provided the iterative algorithm to compute the best $R_{\boldsymbol{xx}}$ and corresponding BC maximum rate sum.

2.8.3.5 Worst-Case Noise Design

Figure 2.72 illustrates the individual primary and secondary components in blue and gray colors respectively. It provides a useful reference for this section that concentrates on the primary-components' separation that occurs with WCN.

Precoder and Receiver: This subsection shows that worst-case noise (block) diagonalizes primary-components'-optimum receiver processing with a lossless-precoder transmitter. First, Subsection 2.8.3.4's

vector precoder recursively computes outputs

$$v_u = \left(\nu_u - \sum_{i>u}^{U^o} g_{u,i} \cdot v_i - \sum_{j=U^o+1}^{U'} \sqrt{\mathcal{E}_{u,j}} \cdot v_j \right)_{\mathcal{E}_u} \quad , \forall\ u = U', ..., 1 \ . \tag{2.494}$$

In WCN design, the resultant $U^o \times 1$ vector $\boldsymbol{v} = \boldsymbol{v}^o + \boldsymbol{v}^s$ is input to Figure 2.72's $A = R_{\boldsymbol{xx}}^{1/2} = Q \cdot \Phi$ matrix filter. The design only focuses on $R_{\boldsymbol{xx}}$, which effectively in highest rate-sum situations corresponds only to energized primary users. These are the blue precoder devices in Figure 2.72. The blue precoder recursively produces this $U^0 \times 1$ vector one dimension at a time. Each of the secondary components may share one of the primary component dimensions, while preserving $R_{\boldsymbol{vv}} = I$, which occurs at the blue summing junction. The vector $\boldsymbol{v}^s$ varies through the energy factors θ_u within the secondary group and the factor $(1-\alpha)$ relative to the primary component set's energy. These secondary components are also in the blue-gray sum of the upper blue precoders that occur before the blue modulo operations. These sums depend on higher indices ν_u and execute sequentially. User $u's$ receiver modulo operation reproduces (after all receiver processing at each output) the quantity $\hat{\nu}_u$. WCN design estimates signals on primary-component dimensions. Secondary components see all lower-indexed components as noise.

The WCN-design process first deletes rows/columns corresponding to secondary components from (2.481). These secondary user components instead use the secondary precoders in Figures 2.71 and 2.72. The precoder's removal of these secondary user components renders $R_{wcn} > 0$ as long as primary-component dimensions carry nonzero energy. The corresponding $U^o \times U^o$ worst case-noise Equation (2.481) uses the $[L_x \times U^o]$ matrix square-root A such that $R_{\boldsymbol{xx}} = A \cdot A^*$, which the factorization uniquely determines.

$$A = Q \cdot \Phi \ . \tag{2.495}$$

Figure 2.72's precoder output retains the statistical independence in the dimensional outputs $\boldsymbol{v}_u$, $u = 1, ..., U^o$ because the lossless-precoder output's $R_{\boldsymbol{vv}} = I$. Then the primary-component-onlyt in (2.481) becomes

$$R_{wcn}^{-1} - \left[H \cdot A \cdot A^* \cdot H^* + R_{wcn} \right]^{-1} = \mathcal{S}_{wcn} \ . \tag{2.496}$$

There are (up to) U block-diagonal (positive definite) entries in $\mathcal{S}_{wcn}(u)$ in $\mathcal{S}_{wcn}$, each of which is now nonsingular because secondary-users' dimensions no longer remain. The $U^o \times U^o$ matrix $\mathcal{S}_{wcn}$ has an (all-real-positive-eigenvalue) eigen-decomposition

$$Q_{wcn}^* \cdot \mathcal{S}'_{wcn} \cdot Q_{wcn} \ , \tag{2.497}$$

where $\mathcal{S}'_{wcn}$ is positive-definite diagonal. Indeed, Q_{wcn} is not only unitary ($Q_{wcn} \cdot Q_{wcn}^* = Q_{wcn}^* \cdot Q_{wcn} = I$, but also block diagonal with each block also unitary so that $Q_{wcn}^*(u) \cdot Q_{wcn}(u) = I = Q_{wcn}(u) \cdot Q_{wcn}^*(u)$.

The $L_x \times U^o$ square-root matrix A, and the $L_x \times L_x$ autocorrelation matrix $R_{\boldsymbol{xx}}$ have the same rank $\varrho_A = \varrho_x = U^o \leq L_x$. As per Figure 2.72, all secondary users' energies tacitly superimpose upon the primary-component dimensions and retain ϱ_x as the $R_{\boldsymbol{xx}}$ rank (while decreasing the rate sum). WCN analysis handles the U^o independent dimensions regardless of which dimensions secondary users may share them. The inverted matrices in (2.496) are consequently nonsingular, allowing all following matrix inversions to exist. The Matrix Inversion Lemma's[106] use on (2.496)'s second term (on the left), and flipping equality sides, leads to

$$\begin{aligned}
\mathcal{S}_{wcn} &= R_{wcn}^{-1} - \left[R_{wcn}^{-1} - R_{wcn}^{-1} \cdot H \cdot A \cdot \left(I + A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot H \cdot A \right)^{-1} \cdot A^* \cdot H^* \cdot R_{wcn}^{-1} \right] \\
Q_{wcn}^* \cdot \mathcal{S}'_{wcn} \cdot Q_{wcn} &= R_{wcn}^{-1} \cdot H \cdot A \underbrace{\left(I + A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot H \cdot A \right)^{-1}}_{R_b \triangleq G^{-1} \cdot S_0^{-1} \cdot G^{-*}} A^* \cdot H^* \cdot R_{wcn}^{-1} \qquad (2.498) \\
\mathcal{S}'_{wcn} &= Q_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot A \cdot R_b \cdot A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot Q_{wcn}^* \ , \qquad (2.499)
\end{aligned}$$

[106] The matrix inversion lemma is

$$[A + BCD]^{-1} = A^{-1} - A^{-1} B [C^{-1} + DA^{-1}B]^{-1} DA^{-1} \ .$$

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 $R_b^{-1} = G \cdot S_0 \cdot G^*$ is therefore a $U^o \times U^o$ positive definite matrix with Cholesky[107] factor G. Section 2.7 found R_b is the MMSE matrix for the backward-channel MMSE estimation of $\boldsymbol{\nu}$ from $\boldsymbol{y}$. Continuing (2.499)

$$\mathcal{S}'_{wcn} = Q_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot A \cdot G^{-1} \cdot S_0^{-1} \cdot G^{-*} \cdot A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot Q_{wcn}^* \quad . \tag{2.500}$$

The unique $R_{\boldsymbol{xx}}^{1/2}$ choice of the matrix A remains ambiguous at this point, and thus so do consequently G and S_0.

Two steps to find the unique A: 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 the first step's completion. $\mathscr{L}_y = U^o$ (number of energized dimensions).

The channel triangular component The $U^o \times L_x$ matrix $R_{wcn}^{-1} \cdot H$ has generalized "QR" factorization[108]

$$Q_{wcn} \cdot R_{wcn}^{-1} \cdot H = \left[\underbrace{\mathbf{0}}_{(L_x - U^o) \times U^o} \quad \underbrace{R}_{U^o \times U^o} \right] \left[\begin{array}{c} \boldsymbol{q} \\ \underbrace{Q^*}_{U^o \times L_x} \end{array} \right] = R \cdot Q^* \ \text{ STEP 1} \quad . \tag{2.501}$$

The input triangular component The $U^o \times U^o$ square nonsingular upper-triangular matrix Φ satisfies the Cholesky square-root factorization:

$$\Phi \cdot \Phi^* = Q^* \cdot R_{\boldsymbol{xx}} \cdot Q \ \text{ STEP 2} \quad . \tag{2.502}$$

The desired unique specific square root is then

$$R_{\boldsymbol{xx}}^{1/2} = A = Q \cdot \Phi \ , \tag{2.503}$$

which unambiguously follows from Q and Φ. When $U^o < L_x$, then $\boldsymbol{x}$'s Hilbert-space ($R_{\boldsymbol{xx}}$) components in the channel null space[109] do not pass to the channel output. Thus, strictly $A \cdot A^* \neq R_{\boldsymbol{xx}}$ **only when** $U^o < L_x$ occurs. Nevertheless, and $A \cdot A^*$ represents the only information content of interest because A is unique. Equivalently, this good A-matrix design zeros energy in the BC's null space.

Finding The Overall MMSE BC Triangular Channel: A $U^o \times U^o$ diagonal matrix D_A has definition

$$D_A \triangleq \text{Diag}\{R \cdot \Phi\} \quad . \tag{2.504}$$

Then (2.498)'s upper triangular matrix G and the diagonal factor S_0 respectively satisfy:

$$G = D_A^{-1} \cdot R \cdot \Phi \tag{2.505}$$

$$S_0 = D_A \cdot (S')_{wcn}^{-1} \cdot D_A \quad , \tag{2.506}$$

which checks through

$$G^{-1} \cdot S_0^{-1} \cdot G^{-*} = \left(\Phi^{-1} \cdot R^{-1} \cdot D_A\right) \cdot \left(D_A^{-1} \cdot S_{wcn} \cdot D_A^{-1}\right) \cdot \left(D_A \cdot R^{-*} \cdot G^{-*}\right) \tag{2.507}$$

$$= \Phi^{-1} \cdot R^{-1} \cdot S_{wcn} \cdot R^{-*} \cdot \Phi^{-*} \tag{2.508}$$

$$= \Phi^{-1} \cdot R^{-1} \cdot \left[Q_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot A \cdot R_b \cdot A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot Q_{wcn}^*\right] \cdot R^{-*} \cdot \Phi^{-*}$$

$$= \Phi^{-1} \cdot R^{-1} \cdot R \cdot Q^* \cdot Q \cdot \Phi \cdot R_b \cdot A^* \cdot Q \cdot \Phi^* \cdot R^* \cdot R^{-*} \cdot A \cdot Q^* \cdot \Phi^{-*} \tag{2.509}$$

$$= R_b \tag{2.510}$$

and the relations

$$Q_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot A \ = \ R \cdot \Phi \ = \ D_A \cdot G \quad . \tag{2.511}$$

also follow immediately.

[107] See Matlab's "chol" command, or for more general block Cholesky, see Appendix D.
[108] See matlab qr command, or the rq.m program in Appendix E.
[109] The null space will be all vectors $\boldsymbol{v}$ such that $Q_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot \boldsymbol{v} = 0$.

The MMSE User Channels Are Canonical: Also of interest is the worst-case-noise mutual information

$$
\begin{aligned}
2^{\mathcal{I}_{wcn}(\boldsymbol{x};\boldsymbol{y})} &= \frac{\mid H \cdot R_{\boldsymbol{xx}} \cdot H^* + R_{wcn} \mid}{\mid R_{wcn} \mid} && (2.512)\\
&= \mid R_{wcn}^{-1/2} \cdot H \cdot R_{\boldsymbol{xx}} \cdot H^* \cdot R_{wcn}^{-*/2} + I \mid && (2.513)\\
&= \mid R_{wcn}^{-1/2} \cdot H \cdot A \cdot A^* \cdot H^* \cdot R_{wcn}^{-*/2} + I \mid && (2.514)\\
&= \mid A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot H \cdot A + I \mid \quad \text{follows from SVD of } R_{wcn}^{-1/2} \cdot H \cdot A && (2.515)\\
&= \mid R_b^{-1} \mid && (2.516)\\
&= \mid S_0 \mid && (2.517)\\
\mathcal{I}_{wcn}(\boldsymbol{x};\boldsymbol{y}) &= \log_2(\mid S_0 \mid) \quad \text{bits/complex subsymbol.} && (2.518)
\end{aligned}
$$

Equation s(2.513)and (2.515) also relate that single-user vector-coding's singular value decomposition (SVD) of the channel path $R_{wcn}^{-1/2} \cdot H \cdot A$ produces a valid set of individual channel SNRs. These SNRs are the squared singular values plus 1, so vector-coded SNR's would be $SNR_{vc,wcn}(u) = \frac{\lambda^2_{vc,wcn,u} \cdot 1}{\sigma^2} + 1$ and thus

$$\mid S_0 \mid = \prod_{u=1}^{U^o} S_{0,u} = \prod_{u=1}^{U^o} (SNR_{vc,wcn}(u) + 1) \quad . \tag{2.519}$$

So also

$$
\begin{aligned}
\widetilde{\mathcal{I}}_{wcn}(\boldsymbol{x};\boldsymbol{y}) &= \sum_{u=1}^{U^o} \log_2(1 + SNR_{vc,wcn}(u)) && (2.520)\\
&= \sum_{u=1}^{U^o} \log_2(1 + SNR_{BC,wcn}(u)) \quad \text{bits/complex subsymbol,} && (2.521)
\end{aligned}
$$

recalling that Equations (2.518) and (2.521) can be divided by 2 to be interpreted in terms of sums over real dimensions. Also evident is VC's usable channels span the same space as the BC's primary components. Thus, vector coding with WCN coincides with maximum BC sum rate for the given $R_{\boldsymbol{xx}}$.

Cannonical User Channel Set Has Diagonal Receiver: Following the vector-coding equivalence, an optimum single-user receiver that sees all users' components of $\boldsymbol{y}$ would begin with the 1-to-1 mapping cascade with noise-whitening matched-matrix filter, recalling also Lemma 2.3.5,

$$
\begin{aligned}
\mathcal{W} &= \underbrace{S_0^{-1} \cdot G^{-*}}_{1-to-1} \cdot \underbrace{A^* \cdot H^* \cdot R_{wcn}^{-1}}_{\text{noise-white-match}} \cdot \underbrace{Q_{wcn}^* \cdot Q_{wcn}}_{I} && (2.522)\\
&= S_0^{-1} \cdot G^{-*} \cdot \Phi^* \cdot Q^* \cdot Q \cdot R^* \cdot Q_{wcn} && (2.523)\\
&= S_0^{-1} \cdot G^{-*} \cdot \Phi^* \cdot R^* \cdot Q_{wcn} && (2.524)\\
&= S_0^{-1} \cdot G^{-*} \cdot G^* \cdot D_A \cdot Q_{wcn} && (2.525)\\
&= S_0^{-1} \cdot D_A \cdot Q_{wcn} \quad , && (2.526)
\end{aligned}
$$

which is (block, because of the Q_{wcn} block diagonal unitary matrix) diagonal and so **needs no coordination** among the U^o primary components. This equals Subsection 2.8.3.1's MSWMF if $\boldsymbol{u}^s = \emptyset$. This also follows directly from Equation (2.511), but without the intermediate decomposition into a noise-whitening-matched matrix and triangular components. Thus, a (block, when any specific user has more than one primary component dimension) diagonal $\mathcal{W}$ can be used on the Gaussian BC primary-component dimensions with no rate-sum loss. ML detectors then follow independently on each user at the corresponding receiver with a modulo operation that corresponds to the BC transmitter's implementation of G^{-1}. The Q_{wcn} matrix exploits local noise correlation at a single user's receiver, within the overall user-to-user worst-case noise structure.

Further, the data path from the transmitter's input innovations/data vector $\boldsymbol{\nu}$ to the receiver's detector input (with the precoder equivalent to G^{-1} if the coefficients of the monic upper-triangular matrix G characterize the primary-component precoder) is

$$\begin{aligned}
\boldsymbol{z} = \mathcal{W} \cdot (\boldsymbol{y} - \boldsymbol{n}) &= \left(S_0^{-1} \cdot D_A \cdot Q_{wcn}\right) \cdot \left(H \cdot A \cdot G^{-1}\right) \boldsymbol{\nu} && (2.527)\\
&= S_0^{-1} \cdot D_A \cdot Q_{wcn,u} \cdot R_{wcn} \cdot R_{wcn}^{-1} \cdot H \cdot Q \cdot \Phi \cdot G^{-1} \cdot \boldsymbol{\nu} && (2.528)\\
&= S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot R_{wcn} \cdot R \cdot Q^* \cdot Q \cdot \Phi \cdot G^{-1} \cdot \boldsymbol{\nu} && (2.529)\\
&= S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot R_{wcn} \cdot R \cdot \Phi \cdot G^{-1} \cdot \boldsymbol{\nu} && (2.530)\\
&= S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot R_{wcn} \cdot R \cdot \Phi \cdot \Phi^{-1} \cdot R^{-1} \cdot D_A \boldsymbol{\nu} && (2.531)\\
&= S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot R_{wcn} \cdot D_A \cdot \boldsymbol{\nu} \ . && (2.532)
\end{aligned}$$

From (2.532), the BC outputs of interest to the (uncoordinated BC primary components') receivers are

$$\boldsymbol{z}_u = \left(S_{0,u}^{-1} \cdot D_{A,u}\right) \cdot \sum_{i=1}^{U^o} Q_{wcn}(u) \cdot R_{wcn}(u,i) \cdot D_{A,i} \cdot \boldsymbol{\nu}_i \ . \tag{2.533}$$

Specifically, $\boldsymbol{z}_u$ is not only a function of $\boldsymbol{\nu}_u$; it depends on all primary-component inputs. Also,

$$\mathbb{E}[\boldsymbol{z}_u / \boldsymbol{\nu}_u] \neq \boldsymbol{\nu}_u \ , \tag{2.534}$$

which biases the detector. The BC receivers together allow reliable detection of the sum rate $b \leq \mathcal{I}_{wcn}$ because Equation (2.527) 's processing is 1-to-1 (invertible over all complex vectors), which by Lemma 2.3.5 retains all information $\mathcal{I}_{wcn}$ at this invertible transformation's output. Further, each receiver (Gaussian case) implements a MMSE estimate with SNR product $\mid R_{\boldsymbol{xx}} \mid / \mid R_{\boldsymbol{ee}} \mid$ because the MMSE estimator checks as

$$\begin{aligned}
\mathbb{E}[\boldsymbol{\nu} \cdot \boldsymbol{y}^*] \cdot \{\mathbb{E}[\boldsymbol{y} \cdot \boldsymbol{y}^*]\}^{-1} &= \mathbb{E}[\boldsymbol{\nu} \cdot \boldsymbol{\nu}^*] A^* H^* \left[H \cdot A \cdot E[\boldsymbol{\nu}\boldsymbol{\nu}^*] \cdot A^* \cdot H + \mathbb{E}[\boldsymbol{n} \cdot \boldsymbol{n}^*]\right]^{-1} && (2.535)\\
&= G \cdot A^* \cdot H^* \left[H \cdot A \cdot I \cdot A^* \cdot H + R_{wcn}\right]^{-1} && (2.536)\\
&= G \cdot \left[A^* \cdot H^* \cdot R_{wcn}^{-1} \cdot H \cdot A + I\right]^{-1} A^* \cdot H^* \cdot R_{wcn}^{-1} && (2.537)\\
&= G \cdot R_b \cdot A^* \cdot Q \cdot R^* \cdot Q_{wcn} && (2.538)\\
&= G \cdot G^{-1} \cdot S_0^{-1} \cdot G^{-*} \cdot \Phi^* \cdot R^* \cdot Q_{wcn} && (2.539)\\
&= S_0^{-1} \cdot G^{-*} \cdot G^* \cdot D_A \cdot Q_{wcn} && (2.540)\\
&= \mathcal{W} \ , && (2.541)
\end{aligned}$$

confirming that the diagonal BC receiver processing $\mathcal{W}$ is MMSE and canonical for the Gaussian BC with WCN. Simple extension of

$$\mathcal{W} \rightarrow [\mathcal{W} \ \ 0] \tag{2.542}$$

is also MMSE for all users, and indeed allows reliable detection of the rate sum $\mathcal{I}$ for the given $R_{\boldsymbol{xx}}$. However, any secondary user components on primary-component dimensions carry an information that exceeds their reliably decodable data rate at their respective receivers, hence the zeros in 2.542.

MMSE Bias Removal: The bias removal is

$$(S_0 - I)^{-1} \cdot S_0 \ . \tag{2.543}$$

Thus

$$\mathcal{W}_{unb} = (S_0 - 1)^{-1} \cdot D_A \cdot Q_{wcn} \ . \tag{2.544}$$

Returning to all primary components, the signal power of interest (the component that arises from $\boldsymbol{\nu}_u$) to the detector is slightly smaller than the full signal power of $\boldsymbol{v}_u$. The noise path in the receiver is (since W is diagonal) is

$$S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot \boldsymbol{n} \ , \tag{2.545}$$

with noise autocorrelation matrix

$$R_{\boldsymbol{n}'\boldsymbol{n}'} = S_0^{-1} \cdot D_A \cdot Q_{wcn} \cdot R_{wcn} \cdot Q_{wcn}^* \cdot D_A \cdot S_0^{-1} \ . \tag{2.546}$$

Individual MMSE User Channel Set Is Cannonical: The ratio of signal energy to noise energy on the diagonals is indeed $S_{0,u}$, and this is a MMSE biased SNR. An ML detector maximizes $p_{\boldsymbol{z}/\boldsymbol{\nu}}$, which has the subsymbol bias (non-zero conditional mean) in Equation (2.534). Subsection 2.3.6 finds the exact relationship in situations with minimum mean-square-error to be with

$$SNR_{bc,wcn,u} \triangleq S_{0,u} - 1 \quad , \tag{2.547}$$

and then

$$E[\boldsymbol{z}_u/\boldsymbol{\nu}_u] = (1 - \frac{1}{S_{0,u}}) \cdot \boldsymbol{\nu}_u \tag{2.548}$$

so the unbiased SNR is not $S_{0,u}$ but $SNR_{BC,wcn,u} = S_{0,u} - 1$ when the receiver scales the subsymbol decision regions to remove (2.534)'s bias. While

$$\mid S_0 \mid = \prod_{u=1}^{U^o} S_{0,u} = \prod_{u=1}^{U^o} (SNR_{BC,wcn}(u) + 1) = \prod_{u=1}^{U^o} (SNR_{VC,wcn}(u) + 1) \quad , \tag{2.549}$$

and $\mathcal{I}_{wcn}(\boldsymbol{x};\boldsymbol{y})$ remain the same, it is not necessarily true that the individual-user SNRs for single-user, WCN-based, 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 (2.549). $\mathcal{W}$ is an MMSE estimator (see Appendix D), which follows from (2.547) - (2.549).

The diagonalized-receiver's primary components' rate sum b^o is independent of the order within $\{\boldsymbol{u}^o\}$, but $b = b^o + b^s$ reduces along with $b^o = b_{u \leq U^o}$ as $b^s = b_{u \geq U^s}$ increases and shares the primary-component' dimensions. Receiver modulo operations remove secondary components and output energy based on $\mathcal{E}_u \cdot \alpha_u$ (where the $\alpha_u \leq 1$ characterizes the reduction that allows secondary components to share the primary-component dimensions). Returning again to Example 2.62:

EXAMPLE 2.8.6 (Simple Broadcast Continued) Example 2.62 earlier continued with worst-case noise. For this example, $L_x = 1$, $L_{y,u} = 1$, and $U = 2$. the channel has rank $\varrho_{\widetilde{H}} = 1$ and is trivially degraded NOMA. There is one primary component (user 1) and one secondary component (user 2). In this case, the WCN-design's R_b matrix is a scalar, so $G = 1$, and thus $S_0 = R_b$. S_0 also follows 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 components** and of course then shows zero value for secondary user (which is user 2). This zeroing of secondary users simply means that the primary-component dimensions carry any nonzero user-2 energy at the expense of reducing primary component/user 1's energy $\mathcal{E}_1$, and consequently bits per dimension $\bar{b}_1$. When user 2 carries nonzero energy, the 0 is of course not implemented and isntead Figure 2.72's structure finds use; the b_2 at receiver 2 is less than its value could be at receiver 1. The maximum (for $R_{\boldsymbol{xx}} = 1$) rate sum can only occur at user 1's receiver.

A primary-component-only case occurs when $L_x = 2$ with 2 users in the following example:

EXAMPLE 2.8.7 (Simple 2×2 non-degraded BC) A perfect-MIMO BC has white Gaussian noise with 2×2 autocorrelation $R_{\boldsymbol{nn}} = .0001 \cdot I$ and channel matrix

$$H = \begin{bmatrix} .8 & .7 \\ .5 & .6 \end{bmatrix} \quad , \tag{2.550}$$

with rank $\varrho_{\widetilde{H}} = 2 = U = U'$. The channel is non-degraded and both user 1 and user 2 solely as primary components. There is no secondary component. Thus, R_b is 2×2, as is G. $L_y = 1$, while $L_x = 2$ (and $l_x = 1$).

```
>> H = [    80    70
            50    60 ];
>> Rxx=[1 .8
       .8 1];
```

The energy is $\mathcal{E}_{\boldsymbol{x}} = trace\{R_{\boldsymbol{xx}}\} = 2$: for the BC $R_{\boldsymbol{xx}}(1) = R_{\boldsymbol{xx}}(2) = 1$ does not imply equal user treatment. There are many $R_{\boldsymbol{xx}}$ choices for which trace $trace\{R_{\boldsymbol{xx}}\} = 2$ that correspond to different rate sums and user rates. This differs from the 2×2 MAC earlier, for which there was only 1 rate sum in Example 2.7.1, when an energy-vector constraint applies.

```
>> [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
>> [R,Q,P]=rq(Htilde)
R =
  -12.4389  -74.6780
         0      -104.5673
Q =
    0.6565   -0.7544
   -0.7544   -0.6565
P =     2     1

ORDER IS REVERSED SO SWITCH USERS!
>>J=[0 1 ; 1 0];

>> >> Htilde-J*R*Q' =
     0     0
     0     0   checks.
 >> Rxxrot=Q'*Rxx*Q;
>> Phi=lohc(Rxxrot) =
    0.4482    0.0825
         0    1.3388
>> DA=diag(diag(R*Phi));
>> G=inv(DA)*R*Phi  =
    1.0000   18.1182
         0    1.0000
>> A=Q*inv(R)*DA*G =
    0.2942   -0.9557
   -0.3381   -0.9411
>> SO=DA*inv(Swcn)*DA =   1.0e+04 *
    0.0032   -0.0000
   -0.0000    2.0229
```

```
Wunb=inv((S0)-eye(2))*DA*J
-0.0000   -0.1822
-0.0069   -0.0000
Indeed diagonal with order switch!
>> Gunb=eye(2)+S0*inv(S0-eye(2))*(G-eye(2)) =
    1.0000   18.7103
         0    1.0000
>> b=0.5*log2(diag(S0))' =   2.4909    7.1521
>> sum(b) =    9.6430
>> J*Wunb*H*A*inv(Gunb)  =

    0.0386  >>    -0.0004
    0.0236  <<   25.4902
Diagonally dominant. (MMSE)
```

The diagonal receiver matrix is $\mathcal{W}$ (this example's matlab output denotes it Wunb). The G_{unb} matrix is nontrivial and defines the channel's input precoder. There is crosstalk after the $\mathcal{W}$ diagonal filtering that occurs through the channel and special square-root transmitter. The precoder coefficient of 18 largely cancels (MMSE sense) the interference from user 1 into user 2 (which is at a level roughly 18/59 below user 1) relative to user 2. The proper rq.m function reorders the users. Any remaining crosstalk is simply best traded for noise (and user 2 crosstalk for user 1) reduction in a MMSE sense. The sum data rate is as high as it can be for the BC and the given $R_{\boldsymbol{xx}}$, 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 $R_{\boldsymbol{xx}}$). If the same channel had to accommodate a third user, then that at least one user would have secondary components, , the primary-compoents' dimensions would carry it with some $\bar{b}_1$ and/or $\bar{b}_2$ loss and sum-rate loss.

```
------------------- 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;
>> Swcn = inv(Rwcn)-inv(H*Rxx*H'+Rwcn) =
    0.9979    0.0000
    0.0000    0.9962
>> [R,Q,P]=rq(Htilde)

R =
  -12.2520  -76.8651
         0 -106.1055
Q =
    0.6583   -0.7528
   -0.7528   -0.6583
P =     2     1
ORDER REVERSED AGAIN
HERE IS SEQUENCE OF COMMANDS repeated and outputs:
>> Rxxrot=Q'*Rxx*Q
Phi=lohc(Rxxrot)
DA=diag(diag(R*Phi))
```

```
G=inv(DA)*R*Phi
A=Q*inv(R)*DA*G
S0=DA*inv(Swcn)*DA
Wunb=inv((S0)-eye(2))*DA*J
Gunb=eye(2)+S0*inv(S0-eye(2))*(G-eye(2))
b=0.5*log2(diag(S0))'
sum(b)
J*Wunb*H*A*inv(Gunb)

Rxxrot =
    1.7929   -0.1067
   -0.1067    0.2071
Phi =
    1.3183   -0.2344
         0    0.4551
DA =
  -16.1520         0
         0  -48.2917
G =
    1.0000    1.9881
         0    1.0000
A =
    0.8678   -0.4969
   -0.9924   -0.1232
S0 =   1.0e+03 *
    0.2614   -0.0000
   -0.0000    2.3410
Wunb =
   -0.0000   -0.0620
   -0.0206   -0.0000
Gunb =
    1.0000    1.9958
   -0.0000    1.0000
b =
    4.0152    5.5965
sum(b) =   9.6116
equalized channel
   0.0008    0.9966
   1.0017   -0.0000
        ---------- with cross correlation .2 ----------------
>> Rxx=[1 .2
.2 1];
>> [Rwcn,b]=wcnoise(Rxx,H,1) =
    1.0000    0.0060
    0.0060    1.0000
b =   10.3240
----------- identity input -----
>> Rxx=eye(2);
>> [Rwcn,b]=wcnoise(Rxx,H,1) =
    1.0000    0.0048
    0.0048    1.0000
b =   10.3517
-----------  may need to change dual gap to 1e-5 ---------------
>> Rxx=[ 1.5 .8
```

```
.8 .5];
>> [Rwcn,b]=wcnoise(Rxx,H,1) =
    1.0000    0.0682
    0.0682    1.0000
b =    8.8612
-------------
>> Rxx=[1.5 0
0 .5];
>> [Rwcn,b]=wcnoise(Rxx,H,1) =
    1.0000    0.0063
    0.0063    1.0000
b =   10.1467
--------------
>> Rxx=[2 0
0 0];
>> [Rwcn,b]=wcnoise(Rxx,H,1,1e-6,1e-3) =
    1.0000    0.6250
    0.6250    1.0000
b =    6.8220
```

The input autocorrelation matrix $R_{\boldsymbol{xx}} = [2\ 0; 0\ 0]$ does not correspond to user 2 with all energy. User 2 having all energy would instead correspond to $R_{\boldsymbol{vv}} = [1\ 0; 0\ 0]$ and an $A \cdot \boldsymbol{v}$ (or $A \cdot R_{\boldsymbol{vv}} \cdot A^*$) that generates $R_{\boldsymbol{xx}}$ being such that $trace\{R_{\boldsymbol{xx}}\} = 2$ and that best adjusts the two-dimensional single-vector BC input.

```
    -----------
>> [Rwcn,b]=wcnoiseplus(Rxx,H,1,1e-6,1e-3) =
    1.0000    0.8571
    0.8571    1.0000
b =    6.6294
    ---
>> Rxx=[1.9 .3
.3 .1];
>> [Rwcn,b]=wcnoiseplus(Rxx,H,1,1e-6,1e-3) =
    1.0000    0.0549
    0.0549    1.0000
b = 8.7729
>> Rxx=[.1 -.3
-.3 1.9];
>> [Rwcn,b]=wcnoiseplus(Rxx,H,1,1e-6,1e-3) =
    1.0000    0.0326
    0.0326    1.0000
b =    8.7316
>> Rxx=[.1 .3
.3 1.9];
>> [Rwcn,b]=wcnoiseplus(Rxx,H,1,1e-6,1e-3) =
    1.0000    0.0570
    0.0570    1.0000
b =    8.7700
```

Example 2.8.7 here also illustrates the variation of sum rate with $R_{\boldsymbol{xx}}$ that Subsection 2.8.3.3 investigates further.

A third Example 2.8.8 with a 3×3 singular channel is more potent in illustrating primary and secondary components and the overall handling:

EXAMPLE 2.8.8 (Singular 3×3 degraded BC) A real-baseband degraded-NOMA BC has white Gaussian noise with 3×3 autocorrelation $R_{\boldsymbol{nn}} = .0001 \cdot I$ and channel matrix

$$H = \begin{bmatrix} .8 & .6 & .4 \\ .6 & .45 & .3 \\ .2 & .2 & .2 \end{bmatrix} , \tag{2.551}$$

with rank $\varrho_{\widetilde{H}} = 2 = U^o < U$. The channel is degraded. The input autocorrelation matrix is given as

$$R_{\boldsymbol{xx}} = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 2 \end{bmatrix} . \tag{2.552}$$

The objective is a precoded design and the corresponding data rates.

This begins by finding the secondary user:

```
>> H=[80 60 40
60 45 30
20 20 20];
>> rank(H)  =     2
>> Rxx=diag([3 4 2]);
>> [Rwcn, b]=wcnoise(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   -0.0000    0.0000
    0.0000    0.0000    0.9948
```

As an aside, we use bcmax to check the maximum data rate

```
>> [Rxxopt, Rwcnopt, bmax] = bcmax((9/3)*eye(3), H, ones(1,3))

Rxxopt =
    3.7515    1.5032   -0.7451
    1.5032    1.5019    1.5007
   -0.7451    1.5007    3.7465
Rwcnopt =
    1.0000    0.7500    0.0006
    0.7500    1.0000    0.0005
    0.0006    0.0005    1.0000
bmax =   12.1084
>> eig(Rwcnopt) =
    0.2499
    1.0000
    1.7501  (so WCN is nonsingular)
>> Swcn=inv(Rwcnopt)-inv(H*Rxxopt*H'+Rwcnopt)
Swcn =
    1.0000   -0.0002    0.0008
   -0.0002    0.0000   -0.0004
    0.0008   -0.0004    0.9971
```

Row/column 2 is almost zero, so corresponds to user 2 having a single component that is secondary. So the best data rate corresponds to a nonsingular R_{wcn} and user 2 is secondary and users 1 and 3 are single primary components each. The sum rate (for the original nonoptimum input $R_{\boldsymbol{xx}}$) if all energy is on users 1 and 3 would then be 11.3777 bits/dimension. BC receiver 2 in practice will simply treats the other two users as noise on whatever emanates from its channel output. If user 2's energy instead reallocates to users 1 and 3, the best reate sum is 12.1 bits/symbol. To show this, this example returns to the non-optimized input case, the remaining matrix for primary-component users 1 and 3 is then

```
>> H1=[H(1,1:3)
H(3,1:3)] =
    80    60    40
    20    20    20
>> [Rwcn, b]=wcnoise(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
>> [Rxxopt, Rwcnopt, bmax] = bcmax(diag([3 4 2]), H1, [1 1]')
Rxxopt =
    3.7515    1.5032   -0.7451
    1.5032    1.5019    1.5007
   -0.7451    1.5007    3.7465
Rwcnopt =
    1.0000    0.0002
    0.0002    1.0000
bmax =   12.1084
```

So R_{wcn}^{opt} is clearly nonsingular and thus there are no longer secondary users because all S_{wcn} diagonal entries are nonzero. (Caution should be exercised with initial conditions on bcmax - if a zero rate occurs, attempt another initial condition, indeed that happens with the initial Rxx as an identity on this channel H1). The special square root is next:

```
>> [R,Q,P]=rq(inv(Rwcn)*H1)
R =
         0    9.1016  -33.2537
         0         0 -107.6507
Q =
    0.4082   -0.5306   -0.7429
   -0.8165    0.1517   -0.5571
    0.4082    0.8340   -0.3713
P =     2     1
ORDER IS REVERSED (Here it is order of users 1 and 3 since 2 was eliminated)
>> R1=R(1:2,2:3) =
    9.1016  -33.2537
         0 -107.6507
>> Q1=Q(1:3,2:3) =
   -0.5306   -0.7429
    0.1517   -0.5571
    0.8340   -0.3713
>> Rxxrot=Q1'*Rxx*Q1  =
```

```
    2.3275    0.2251
    0.2251    3.1725
>> Phi=lohc(Rxxrot)  =
    1.5204    0.1264
         0    1.7811
>> DA=diag(diag(R1*Phi)) =
   13.8379         0
         0 -191.7414
>> G=inv(DA)*R1*Phi  =
    1.0000   -4.1971
         0    1.0000
>> A=Q1*inv(R1)*DA*G =
   -0.8067   -1.3902
    0.2306   -0.9730
    1.2679   -0.5559
>> A*A'  =
    2.5833    1.1667   -0.2500
    1.1667    1.0000    0.8333
   -0.2500    0.8333    1.9167
```

The product $A \cdot A^* \neq R_{\boldsymbol{xx}}$ only because $R_{\boldsymbol{xx}}$ places energy into the BC null space. The part of the energy that will pass is $A \cdot A^*$ and the remaining difference $R_{\boldsymbol{xx}} - A \cdot A^*$ is not relevant and carries no useful information, although it does represent energy waste. Thus, this $R_{\boldsymbol{xx}}$ is not a very good design. Other choices would be better that have no such waste. The design now completes for primary components with

```
>> S0=DA*inv(Swcn)*DA =   1.0e+04 *
    0.0192    0.0000
    0.0000    3.6957
>> MSWMFunb=(inv(S0)-eye(2))*DA*J  =
    0.0000    0.0726
   -0.0052   -0.0000
>> b=0.5*log2(diag(S0))'
    3.7909    7.5868
>> Gunb=eye(2)+S0*inv(S0-eye(2))*(G-eye(2))  =
    1.0000   -4.2191
   -0.0000    1.0000
>> b=0.5*log2(diag(S0))'  =
    3.7909    7.5868
>> sum(b) =   11.3777
```

These are data rates with the two primary components/users, but they are not maximum because the input is not optimized.

The transmitter input to the A matrix has unit energy on each of the two primary component/user dimensions. The channel outputs thus have matrix description $\boldsymbol{y} - \boldsymbol{n} = \widetilde{H}\boldsymbol{v}$

```
>> H*A =
    0.0219 -191.8333
    0.0164 -143.8749
   13.8379  -58.3825
```

Thus, with the order reversal,

$$y_1 = 13.84 \cdot v_1 - 58.38 \cdot v_3 \tag{2.553}$$

$$y_2 = 0.0164 \cdot v_1 - 143.87 \cdot v_3 \tag{2.554}$$

$$y_3 = .0219 \cdot v_1 - 191.83 \cdot v_3 \tag{2.555}$$

Problem 2.30 is similar to this example. Further

$$\begin{aligned} v_1 &= \sqrt{\mathcal{E}_1} \cdot v_1^o + \sqrt{\mathcal{E}_{1,2}} \cdot v_2 \\ v_3 &= \sqrt{\mathcal{E}_3} \cdot v_3^o + \sqrt{\mathcal{E}_{3,2}} \cdot v_2 \end{aligned}$$

These two dimensions then enter matrix A so

$$\boldsymbol{x} = \begin{bmatrix} -0.8067 & -1.3902 \\ 0.2306 & -0.9370 \\ 1.2679 & -0.5559 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_3 \end{bmatrix} \tag{2.556}$$

$$= \begin{bmatrix} -0.8067 & -1.3902 \\ 0.2306 & -0.9370 \\ 1.2679 & -0.5559 \end{bmatrix} \cdot \begin{bmatrix} \sqrt{\mathcal{E}}_1 & \sqrt{\mathcal{E}}_{12} & 0 \\ 0 & \sqrt{\mathcal{E}}_{23} & \sqrt{\mathcal{E}}_3 \end{bmatrix} \begin{bmatrix} v_1^o \\ v_2 \\ v_3^o \end{bmatrix} \tag{2.557}$$

Energy distribution to user 2 reduces the rate sum, but user 2 must carry data in this example. 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 (recalling $\mathcal{E}_{v_1} = \mathcal{E}_{v_2} = 1$ while $\mathcal{E}_{x,1} = 3\ \mathcal{E}_{x,2} = 4\ \mathcal{E}_{x,3} = 2$ with scaling in A)

$$\mathcal{E}_1 = 1 \tag{2.558}$$

$$\mathcal{E}_3 = 1/3 \tag{2.559}$$

and correspondingly then

$$\mathcal{E}_{1,2} = 0 \tag{2.560}$$

$$\mathcal{E}_{3,2} = 2/3 \tag{2.561}$$

The primary component/user data rates must account for the secondary component/user so

```
> b=0.5*log2(diag([ 1 1/3]) *diag(S0)) =
    3.7909
    6.7943
```

User 2 remains with signal energy component

$$(60)^2 \cdot 2/3 = 2400 \ ,$$

with noise components from both users 1 and 3 that can be computed from the middle row of $H \cdot A$

$$\mathcal{E}_1 \cdot .0164^2 + \mathcal{E}_3 \cdot 143.8^2 = 6893 \ .$$

$$\begin{aligned} b_2 &= \frac{1}{2} \cdot \log_2 \left(1 + \frac{2400}{6893} \right) \\ &= 0.22 \text{ bits/subsymbol.} \end{aligned}$$

The rate sum is now $3.7909 + 6.7943 + .22 = 10.8 < 11.3777$). A reduction in rate sum occurs for any nonzero energy allocation to user 2. Figure 2.73 summarizes the design, with the choices of energy illustrated specifically and generically.

The worst-case-noise-based design emphasizes finding a (total) square root matrix A that provides maximum rate sum for primary-user components.

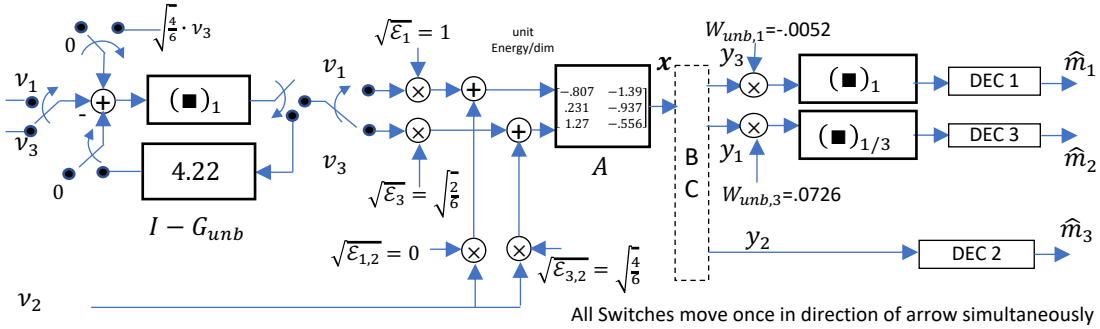


Figure 2.73: Illustration of Example 2.8.8

The continuation of Example 2.8.8 illustrates that just any decomposition of $R_{\boldsymbol{xx}}$ into sum components (without regard to the channel H specifics) does not achieve the best sum rate of (11.3777), thus illustrating that the initial set $\{R_{\boldsymbol{xx}}(u)\}$'s assignment to users is also important:

Example 2.8.8: MMSE design and suboptimal user energy allocation The input covariance $R_{\boldsymbol{xx}} = \text{diag}([3\ 4\ 2])$ might simply decompose into (zeroing user 2 to attempt best rate sum) two equal-energy diagonal autocorrelation matrices for users 1 and 3 as:

```
>> Rxx1 =
    1.5000         0         0
         0    2.0000         0
         0         0    1.0000
>> Rxx2 = zeros(3,3);
>> Rxx3 = Rxx1;
>> AU1=sqrtm(Rxx1);
AU2=zeros(3,3);
AU3=AU1;
AU=[ AU1 AU2 AU3];
>> AU*AU' =
    3.0000         0         0
         0    4.0000         0
         0         0    2.0000
>> [Bu, Gunb, S0, MSWMFunb] = mu_bc(H, AU , [1 1 1] , 2);

Bu =  7.0837         0    0.4998
>> sum(Bu) =    7.5835 < 11.3777
```

Other diagonal allocations also seem to produce rate sums well below best. Thus, the decomposition of $R_{\boldsymbol{xx}}$ into user components is important. The direct MMSE design approach, while easier than WCN, ignores how good $\{R_{\boldsymbol{xx}}(u)\}$ can occur. As Subsection 2.8.3.3 will show, this BC has a maximum rate sum (with 9 units of input energy) of 12.1084 , which is slightly below the single-user bound of

```
0.5*log2(det(H*RxxA*H'+eye(3))) =   12.4303
```

or a loss of $\gamma_{BC} = .16$ dB.

Thus, the simple MMSE precoder design still needs help to get best performance, which means to design a good set of $\{R_{\boldsymbol{xx}}(u)\}$. That design will be addressed through duality, which Subsection 2.8.4 next introduces, and Chapter 5, Section 5 more completely develops, to use a MAC channel for which

good $R_{\boldsymbol{xx}}(u)$ sets are less complex to find, transforms to the BC through duality, and then proceeds with the MMSE precoder design, creating an alternative to the difficult analysis and precoder, for example as in Figures 2.72 and 2.73. However, primary and secondary separation only directly is understood via WCN design. Later, a better design process will avoid worst-case noise, although another convex optimization (Section 5.4's minPMAC program) will then instead become necessary.

2.8.4 Scalar duality for Gaussian MAC/BC Channels

Duality relates the MAC and BC when $H_{MAC} = J_x \cdot H_{BC}^* \cdot J_y$ in multiuser transmission (and its first earliest forms appeared before publication elsewhere in this text's earlier on-line turn-century editions), where the J matrices are reversal matrices (1's on antidiagonal in the scalar case, further $J_y = 1$). Effectively,

$$H_{MAC} = [h_U \ \ldots \ h_2 \ \ h_1] \tag{2.562}$$

$$H_{BC} = \begin{bmatrix} h_1^* \\ h_2^* \\ \vdots \\ h_U^* \end{bmatrix} . \tag{2.563}$$

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

$$\sum_{u=1}^{U} \mathcal{E}_u^{BC} \leq \mathcal{E}_x \quad , \tag{2.564}$$

while the MAC has additional restrictions of energy per user $\mathcal{E}_u$ individually bounded or an energy vector $\boldsymbol{\mathcal{E}}$ such that

$$\begin{bmatrix} \mathcal{E}_1^{MAC} \\ \vdots \\ \mathcal{E}_U^{MAC} \end{bmatrix} \preceq \boldsymbol{\mathcal{E}}_x \tag{2.565}$$

where $\preceq$ denotes component-wise $\leq$. Duality is instead with energy-sum MAC's, so

$$\{\boldsymbol{\mathcal{E}} \mid \mathbf{1}^* \boldsymbol{\mathcal{E}} \leq \mathcal{E}_x\} \tag{2.566}$$

matches the BC energy constraint. Duality also reverses order to simplify user indexing. The coding gap is $\Gamma = 0$ dB for all users, and thus all users employ Gaussian capacity-achieving codes. Figure 2.74 illustrates the basics.

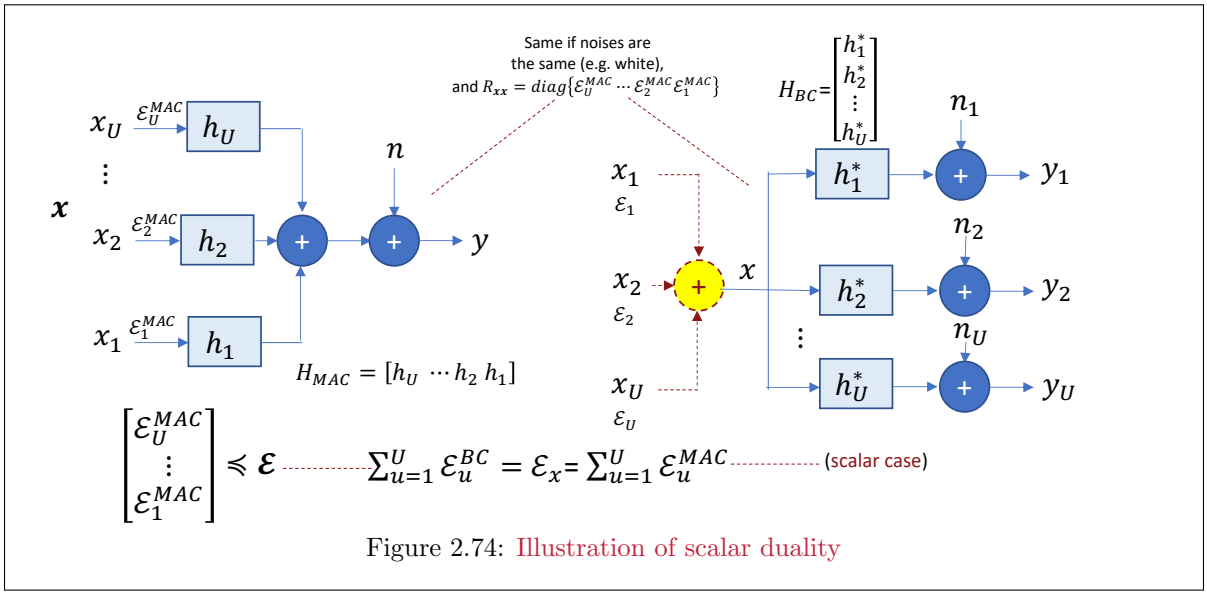


Figure 2.74: Illustration of scalar duality

Table 2.5 lists the corresponding MAC and BC $\bar{b}_u$ with the order reversal in most preferred position evident. The table's bits per user-subsymbol are equal, thus constraining the corresponding energies[110]

MAC	BC
$\bar{b}_1 = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_1^{MAC}\cdot g_1}{1+\mathcal{E}_2^{MAC}\cdot g_2+\ldots+\mathcal{E}_U^{MAC}\cdot g_U}\right)$	$\bar{b}_1 = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_1^{BC}\cdot g_1}{1}\right)$
$\bar{b}_2 = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_2^{MAC}\cdot g_2}{1+\mathcal{E}_3^{MAC}\cdot g_3+\ldots+\mathcal{E}_U^{MAC}\cdot g_U}\right)$	$\bar{b}_2 = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_2^{BC}\cdot g_2}{1+\mathcal{E}_1^{BC}\cdot g_2}\right)$
$\vdots$	$\vdots$
$\bar{b}_U = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_U^{MAC}\cdot g_U}{1}\right)$	$\bar{b}_U = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_U^{BC}\cdot g_U}{1+\left[\mathcal{E}_1^{BC}+\ldots+\mathcal{E}_{U-1}^{BC}\right]\cdot g_U}\right)$

Table 2.5: Scalar Duality bits/user

[110] 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.

For equality of bit rates, these equations follow:

$$\begin{aligned}
\mathcal{E}_1^{BC} &= \mathcal{E}_1^{MAC} \cdot \frac{1}{1+\mathcal{E}_2^{MAC}\cdot g_2 + ... + \mathcal{E}_U^{MAC}\cdot g_U} && (2.567)\\
\mathcal{E}_2^{BC} &= \mathcal{E}_2^{MAC} \cdot \frac{1+\mathcal{E}_1^{BC}\cdot g_2}{1+\mathcal{E}_3^{MAC}\cdot g_3 + ... + \mathcal{E}_U^{MAC}\cdot g_U} && (2.568)\\
\vdots &= \vdots && (2.569)\\
\mathcal{E}_U^{BC} &= \mathcal{E}_U^{MAC} \cdot \left(1+\left[\mathcal{E}_1^{BC} + ... + \mathcal{E}_{U-1}^{BC}\right]\cdot g_U\right) && (2.570)\\
& && (2.571)
\end{aligned}$$

Theorem 2.8.3 **[Equal Sum Energy for Duality]** *Scalar duality has the same MAC and BC energy sums when the bit rates are set equal, namely*

$$\sum_{u=1}^{U}\mathcal{E}_u^{BC} = \sum_{u=1}^{U}\mathcal{E}_u^{MAC} \quad . \tag{2.572}$$

Proof: The sum of the equations:

$$\begin{aligned}
\mathcal{E}_1^{BC}\cdot\left[1+\mathcal{E}_2^{MAC}\cdot g_2 + ... + \mathcal{E}_U^{MAC}\cdot g_U\right] &= \mathcal{E}_1^{MAC} && (2.573)\\
\mathcal{E}_2^{BC}\cdot\left[1+\mathcal{E}_3^{MAC}\cdot g_3 + ... + \mathcal{E}_U^{MAC}\cdot g_U\right] &= \mathcal{E}_2^{MAC}\cdot\left[1+\mathcal{E}_2^{BC} g_2\right] && (2.574)\\
\vdots &= \vdots && (2.575)\\
\mathcal{E}_U^{BC} &= \mathcal{E}_U^{MAC}\cdot\left(1+\left[\mathcal{E}_1^{BC} + ... + \mathcal{E}_{U-1}^{BC}\right]\cdot g_U\right)
\end{aligned}$$

equals

$$\sum_{i=1}^{U}\mathcal{E}_i^{BC} + \sum_{i=1}^{U}\mathcal{E}_i^{BC}\cdot\sum_{k=i+1}^{U} g_k\cdot\mathcal{E}_k^{MAC} = \sum_{i=1}^{U}\mathcal{E}_i^{MAC} + \sum_{i=2}^{U}\mathcal{E}_i^{MAC}\cdot g_i\cdot\sum_{k=1}^{i-1}\mathcal{E}_k^{BC} \tag{2.576}$$

Inspection of the 2^{nd} term on the left and on the right in (2.576) when $U=3$ provides insight:

$$\begin{aligned}
\mathcal{E}_1^{BC}\left(g_2\cdot\mathcal{E}_2^{MAC} + g_3\cdot\mathcal{E}_3^{MAC}\right) &+ \mathcal{E}_2^{BC}\cdot g_3\cdot\mathcal{E}_3^{MAC}\\
= g_2\cdot\mathcal{E}_1^{BC}\cdot\mathcal{E}_2^{MAC} &+ g_3\cdot\left(\mathcal{E}_1^{BC}\cdot\mathcal{E}_3^{MAC} + \mathcal{E}_2^{BC}\cdot\mathcal{E}_3^{MAC}\right) \quad . \quad (2.577)
\end{aligned}$$

The remaining terms on the left and right of (2.576) are also equal. In general, the 2nd term on the left then can be rewritten

$$\sum_{i=2}^{U}\mathcal{E}_i^{MAC}\cdot g_i\cdot\sum_{k=1}^{i-1}\mathcal{E}_k^{BC} \tag{2.578}$$

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/MIMO BC and MAC duality forms.

EXAMPLE 2.8.9 *[Example 2.8.1 revisited with duality]* Returning again to Examples 2.8.4 and 2.8.1 and Figure 2.62, $h_1 = .8$ and $h_2 = .5$ and $\sigma^2 = .0001$. As an example point for duality, let $\mathcal{E}_1^{BC} = \mathcal{E}_2^{BC} = .5$, or equivalently $\alpha = .5$. Then, $g_1 = 80^2$, $g_2 = 50^2$ and

$$\mathcal{E}_2^{MAC} = \frac{\mathcal{E}_2^{BC}}{1+\mathcal{E}_1^{BC}\cdot g_2} = \frac{1}{2}\left(\frac{1}{1+2500(.5)}\right) = \frac{1}{2502} \quad , \tag{2.579}$$

while then

$$\mathcal{E}_1^{MAC} = \mathcal{E}_1^{BC} \cdot (1 + g_2 \cdot \mathcal{E}_2^{MAC}) = (.5) \cdot \left(1 + \frac{2500}{2502}\right) = \frac{2501}{2502} = 1 - \mathcal{E}_2^{MAC} \quad . \tag{2.580}$$

The users' energy sum is again 1 as it should be. Then the users' bits/subsymbol for the dual MAC channel are

$$b_1 = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_1^{MAC} \cdot g_1}{1 + \mathcal{E}_2^{MAC} \cdot g_2}\right) = 5.82 \tag{2.581}$$

$$b_2 = \frac{1}{2}\log_2\left(1 + \frac{\mathcal{E}_2^{MAC} \cdot g_2}{1}\right) = .50 \quad , \tag{2.582}$$

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 energy-sum MAC channels with $\sum_u \mathcal{E}_u = 1$. Duality avoids the need to find worst-case noise. It also uses a suboptimal order for the MAC. With care to reverse MAC order (with respect to its best order when analyzed separately) in duality, Subsection 2.7.2.2's MMSE MAC follows for this dual channel as, through mu_mac:

```
>> H=[50 80];
>> Rxx=diag([1/2502 2501/2502]) =
    0.0004         0
         0    0.9996
>> A=sqrtm(Rxx);
[b, GU, WU, S0, MSWMFU] = mu_mac(H, A, [1 1] , 2)

b =    0.4997    5.8222
GU =
    1.0000   80.0160
         0    1.0000
WU =
    1.0008         0
   -0.0125    0.0003
S0 =   1.0e+03 *
    0.0020         0
         0    3.2010
MSWMFU =
    1.0004
    0.0125
>> sum(b) =    6.3219
```

The two users' bits/subsymbol match for 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 as per the dual-BC's use of mu_bc). In general, design most prudently uses mu_bc after duality fines user rates and energies.

For the dual BC, the transmitter and receiver are:

```
>> Hbc=[80
50];
```

```
>> AU=[sqrt(.5) sqrt(.5)];
>> [Bu, Gunb, S0, MSWMFunb] = mu_bc(Hbc, AU, [1 1] , 2)
Bu =    5.8222    0.4997
>> Gunb{:,:} =
     1     1
     0     1
S0 =  2 x 1 cell array
     3201
    1.9992
MSWMFunb =  2 x 1 cell array
    0.0177
    0.0283
```

In this case, since duality found the equivalent MAC and BC $R_{\boldsymbol{xx}}(u)$ sets, the direct MMSE-precoder approach then is easiest since they are given.

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

$$\boldsymbol{\mathcal{E}} = \begin{bmatrix} \mathcal{E}_{1,max} \\ \vdots \\ \mathcal{E}_{U,max} \end{bmatrix} \quad . \tag{2.583}$$

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 $\{\mathcal{C}(\boldsymbol{b})\} = \emptyset$.
2. for all $\boldsymbol{\mathcal{E}}$ such that $\mathbf{1}^* \boldsymbol{\mathcal{E}} = \mathcal{E}_{\boldsymbol{x}}$ (Using the discretization process to make computation finite as in (2.288)
   - Compute $a(\boldsymbol{b})$ for the MAC defined by g_1, ..., g_U with $\boldsymbol{\mathcal{E}}$ as energy-vector constraint.
   - Form $\{c(\boldsymbol{b})\} = \bigcup \{\{c(\boldsymbol{b})\}, a(\boldsymbol{b})\}$.

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 The continuous-time Gaussian BC

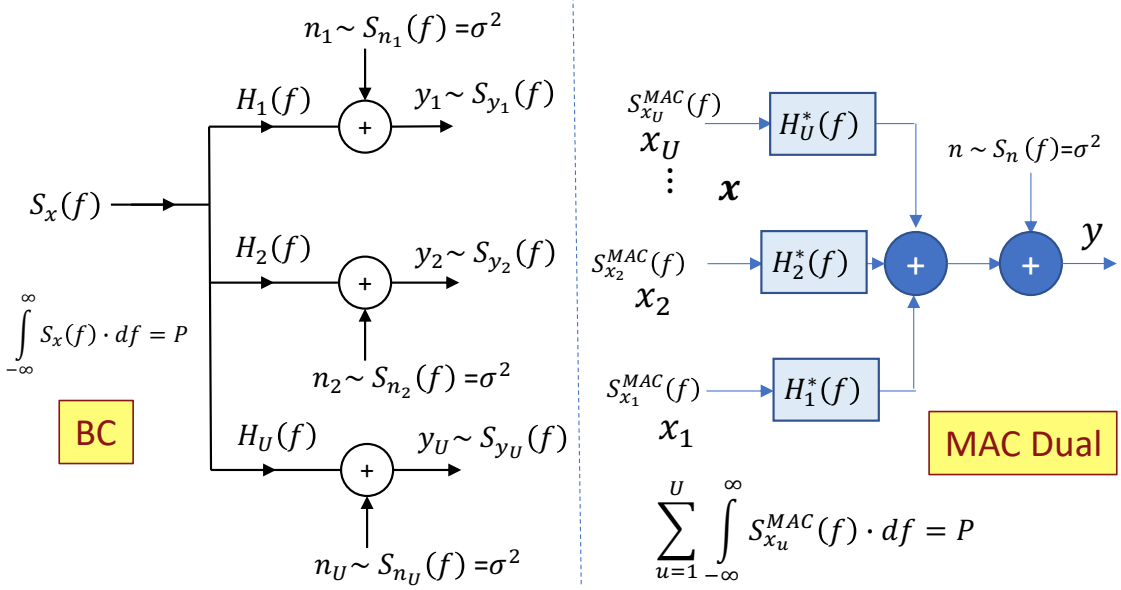


Figure 2.75: Continuous-time duality.

The continuous time scalar ($L_x = L_{y,u} = 1$) BC follows Section 2.5's MT analysis where every temporal channel $h_u(t)$ component converts into a infinitesimally narrow MT channel with $H_u(f)$. All expressions then are integrated over frequency.

$$b_1 \leq \mathcal{I}(x_1; y_1/x_2, x_3, ..., x_U)df = \int_{-\infty}^{\infty} \frac{1}{2} \cdot \log_2 \left(1 + S_{x_1}(f) \cdot |H_1(f)|^2\right) \cdot df \tag{2.584}$$

$$b_2 \leq I(x_2; y_2/x_3, ..., X_U) = \int_{-\infty}^{\infty} \frac{1}{2} \cdot \log_2 \left(1 + \frac{S_{x_2}(f) \cdot |H_2(f)|^2}{1 + S_{x_1}(f) \cdot |H_2(f)|^2}\right) \cdot df \tag{2.585}$$

$$b_3 \leq I(x_3; y_3/x_4, ..., X_U) = \int_{-\infty}^{\infty} \frac{1}{2} \cdot \log_2 \left(1 + \frac{S_{x_3}(f) \cdot |H_3(f)|^2}{1 + [S_{x_1}(f) + S_{x_2}(f)] \cdot |H_3(f)|^2}\right) \cdot df \tag{2.586}$$

$$\vdots \tag{2.587}$$

$$b_U \leq I(x_U; y_U) = \int_{-\infty}^{\infty} \frac{1}{2} \cdot \log_2 \left(1 + \frac{S_{x_U}(f) \cdot |H_U(f)|^2}{1 + \left[\sum_{u=1}^{U-1} S_{x_u}(f)\right] \cdot |H_U(f)|^2}\right) \cdot df \tag{2.588}$$

However, these integrals follow from scalar duality with the dimensional gains becoming transfer functions and energies becoming power spectral densities, both then as functions of frequency f. Then the MAC continuous-time scalar duality algorithm can be applied directly to the continuous-time scalar Gaussian BC. Chapters 4 and 5 investigate further the discrete-frequency implementation. Indeed, the program mu_bc.m already. has within it an N parameter, nominally $\bar{N} = 1$ in this chapter, which specifies the number of tones within a a symbol period. When N is large, this will then approximate the use of MT on the BC. Figure 2.75 illustrates the basics.

Continuous-time duality: Continuous-time duality follows discrete duality with a frequency index for every infinitessimmally narrow MT index f:

$$S_{x_1}^{BC}(f) = S_{x,1}^{MAC}(f) \cdot \frac{1}{1 + S_{x,2}^{MAC}(f) \cdot G_2(f) + ... + S_{x_U}^{MAC}(f) \cdot G_U}(f) \tag{2.589}$$

$$S_{x_2}^{BC}(f) = S_{x_2}^{MAC}(f) \cdot \frac{1 + S_{x_1}^{BC}(f) \cdot G_2(f)}{1 + S_{x_3}^{MAC}(f) \cdot G_3(f) + ... + S_{x_U}^{MAC}(f) \cdot G_U(f)} \tag{2.590}$$

$$\vdots = \vdots \tag{2.591}$$

$$S_{x,U}^{BC}(f) = S_{x,U}^{MAC}(f) \cdot \left(1 + \left[S_{x_1}^{BC}(f) + ... + S_{x_{U-1}}^{BC}(f)\right] \cdot G_U(f)\right) \quad . \tag{2.592}$$

The MAC and BC will have the same total transmit energy and data rates with the above transformations.

2.8.6 Non-Gaussian BC rate regions

Section 2.6's general capacity region applies to any BC. In particular, Equations 2.284 and 2.285 still apply, along with finding the $\boldsymbol{\mathcal{I}}_{min}$ vector vertices for the allowed orders $\boldsymbol{\pi}$ and joint input distributions $p_{\boldsymbol{x}}$. The single transmitter leads to $\boldsymbol{\Pi} \rightarrow \boldsymbol{\pi}$ so only $U!$ different BC orders for consideration. However, some further simplification is possible, leveraging a generalized dual MAC, although mostly in definitional appearance rather than the direct dual mapping in the Gaussian BC case. Indeed, even the Gaussian BC construction with duality is complex. Chapter 5 however finds significantly less complex ways to compute designs for any single point within the capacity region $\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})$ for both the Gaussian MAC, and then through a more generalized form of vector-Gaussian duality, that then apply also to the Gaussian BC. Once the specific $\{R_{\boldsymbol{xx}}(u)\}$ that correspond to $\boldsymbol{b}'$, the MMSE approach earlier in this section applies directly. This subsection guides generalization to the non-Gaussian situation where a MAC design method can also extend to a corresponding dual BC.

Gaussian to AEP Mapping: At this point, the reader can extrapolate insight from the Gaussian case to the more general AEP channels (any number of users, any channel) through the following 3 conceptual expansions:

> **Energy to Entropy** $\mathcal{E}_x \rightarrow \mathscr{H}_x$,
> **MMSE to Conditional Entropy** $\sigma^2_{mmse} \rightarrow \mathscr{H}_{x/y}$,
> **SNR to Mutual Information** $SNR \rightarrow \mathcal{I}$.

The (zero-mean) Gaussian case essentially reduces a distribution to a single parameter (or single autocorrelation matrix), namely the energy, which is then in 1-to-1 relationship with the entropy $\mathscr{H}$. That energy in the conditional case becomes MMSE. The SNR is similarly then in 1-to-1 relationship with the mutual information in the Gaussian case. More general distributions may have more parameters, and entropy and mutual information essentially reduce these again to a single-parameter characterization.

Generalized Overall Duality Relations: BC duality somewhat tacitly implies 3 overall relations that can be inferred for the set of users from the above extrapolations:

1. **Equal Sum Rates:** $\mathcal{I}(\boldsymbol{x}_{MAC}; y_{MAC}) = \mathcal{I}(x_{BC}; \boldsymbol{y}_{BC})$,
2. **Equal Sum Entropy:** $\sum_u \mathscr{H}_{x_{u,MAC}} = \sum_u H_{x_{u,BC}}$.
3. **Equal Sum Conditional Entropy:** $\sum_u \mathscr{H}_{x_{u,MAC}/[y_{MAC}, x_{u-1},...,1,MAC]} = \sum_u H_{x_{U,BC}/[y_{u,BC} x_{u+1},...,U,BC]}$.

Individual User Duality Relations: Duality also preserves individual user bit rates and corresponds to these generalized-duality relations:

$$
\begin{aligned}
b_1 \mid \quad & H_{x_{1,MAC}} - H_{x_{1,MAC}/y_{MAC}} && = H_{x_{1,BC}} - H_{x_{1,BC}/[y_{1,BC},x_{2,\ldots,U,BC}]} && (2.593)\\
b_2 \mid \quad & H_{x_{2,MAC}} - H_{x_{1,MAC}/[y_{MAC},x_{1,MAC}]} && = H_{x_{2,BC}} - H_{x_{2,BC}/[y_{2,BC},x_{3,\ldots,U,BC}]} && (2.594)\\
\vdots \quad & \vdots && \vdots && (2.595)\\
b_U \mid \quad & H_{x_{U,MAC}} - H_{x_{U,MAC}/[y_{MAC},x_{1,\ldots,U-1MAC}]} && = H_{x_{U,BC}} - H_{x_{U,BC}/y_{U,BC}} && (2.596)
\end{aligned}
$$

The rate sum is indeed the mutual information (for the given $p_{\boldsymbol{x}}$), and this sum is order independent for the MAC, and thus also for the dual BC. Individual user data-rates usually depend on order, not just the sum. Thus, any MAC channel $p_{y_{MAC}/\boldsymbol{x}}$ and input distribution $p_{\boldsymbol{x}}$ that satisfy the dual equations, both individual and overall, is a dual for the BC. If such a MAC has a known or simple construction for the capacity region, then the BC capacity region is also constructed. The Gaussian case is an example and may have direct methods for computing the energy vector for a given $\boldsymbol{b}'$ that is within $\mathcal{C}(\boldsymbol{b})$. Similarly if there is a method for computing the $p_{\boldsymbol{x}}$ for the MAC, and thus consequenty the user entropies and conditional entropies, then a dual input probability distribution for the BC that satisfies the equalities will achieve the same $\boldsymbol{b}'$ on the BC reliably. This distribution is often not unique, although it may be unique if $\boldsymbol{b} \in \mathscr{C}(\boldsymbol{b})$, the capacity-region boundary. There may be considerable more information in the literature about methods for computing MAC capacity regions, or for designs that correspond to a "best" MAC design, which then apply through generalized duality to the BC.

2.9 The Gaussian Interference Channel

IC analysis uses Equation (2.283)'s minimum mutual-information vector, $\mathcal{I}_{min}(\mathbf{\Pi}, p_{\boldsymbol{xy}})$. Sections 2.7 and 2.8's capacity-region construction for the single-receiver MAC and BC respectively did not need $\mathcal{I}_{min}$. The Gaussian IC $\mathcal{C}_{IC}(\boldsymbol{b})$ (for the channel $\boldsymbol{y} = H \cdot \boldsymbol{x} + \boldsymbol{n}$) however uses $\mathcal{I}_{min}$, but the region's construction still can simplify with respect to Section 2.6's general capacity-region construction.

An IC has (at least) two possible interpretations:

MAC set The Gaussian IC is a set of U Gaussian MACs with a common (block) diagonal $R_{\boldsymbol{xx}}$.

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 $\mathcal{J} \cdot \widetilde{H}^*$ that can translate the MAC-set approach to/from the BC-set approach.

2.9.1 The MAC-set approach

For the Gaussian IC, (2.251)'s $\mathcal{I}_{min}$ minimization occurs for a given order $\mathbf{\Pi}$ and a given (necessarily [block] diagonal) input autocorrelation $R_{\boldsymbol{xx}}$. Some two-user examples first provide insight, with two scalar Gaussian ICs first in Subsection 2.9.1.1 and then a vector IC with all-primary users in Subsection 2.9.2. The first examples have 2 users, one primary and one secondary for the IC's $U = 2$ MAC receivers. **In this first example**, only $U = 2$ orders need consideration, not $(2!)^2 = 4$ orders). The second example considers all 4 orders or worse $(2^U!)^U = (4!)^2 = 576$, although ultimately reduces to 3. In the second example $L_y = 2$ for both receivers (with $L_x = 1$). Subsection 2.9.2's examples allow partition of a more general situation into sets of primary and secondary users components that can be viewed as 1st and 2nd users.

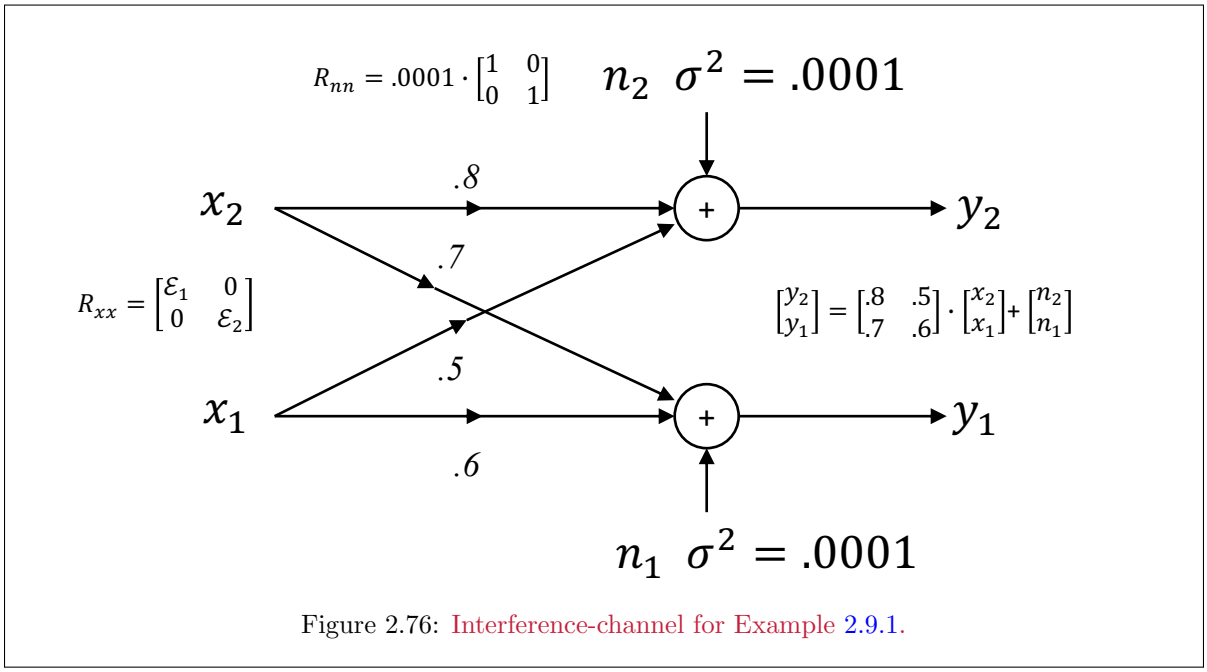


Figure 2.76: Interference-channel for Example 2.9.1.

Two-User IC Simplification: The two-user IC can be a special case where $U = U' < U^2$: If either user had multiple (maximally 2 in this case) subusers, the intended receiver of course must decode them all (both), and thus subuser division never directly benefits the intended receiver. With $U = 2$, the unintended receiver could elect to decode or treat as noise the interfering user. Decoding only a subuser portion of the interfering user simply leaves more noise and degrades the intended user's detection, although it may increase that interfereing user's achievable rate.

2.9.1.1 The Scalar Gaussian IC

EXAMPLE 2.9.1 **[Simple 2×2 scalar IC]** Figure 2.76 illustrates a 2×2 scalar IC, within which are two scalar MACs to each physically separated output. This IC has a diagonal input $R_{\boldsymbol{xx}}$. Its noise $R_{\boldsymbol{nn}}$ is also diagonal. 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.

$\frac{\log_2\left(1+\frac{.64}{.0001}\right)}{2}$ =6.3220	$\frac{\log_2\left(1+\frac{.36}{.0001}\right)}{2}$ =5.9071	$\frac{\log_2\left(1+\frac{.64}{.2501}\right)}{2}$ =0.9157	$\frac{\log_2\left(1+\frac{.25}{.6401}\right)}{2}$ =0.2378
$\frac{\log_2\left(1+\frac{.36}{.4901}\right)}{2}$ =0.3973	$\frac{\log_2\left(1+\frac{.49}{.3601}\right)}{2}$ =0.6196	$\frac{\log_2\left(1+\frac{(.5).64}{.0001}\right)}{2}$ =5.8222	$\frac{\log_2\left(1+\frac{(.5).36}{.0001}\right)}{2}$ =5.4073
$\frac{\log_2\left(1+\frac{.64}{.1251}\right)}{2}$ =1.3063	$\frac{\log_2\left(1+\frac{.18}{.4901}\right)}{2}$ =0.2257	$\frac{\log_2\left(1+\frac{.49}{.1801}\right)}{2}$ =0.9478	$\frac{\log_2\left(1+\frac{.36}{.2451}\right)}{2}$ =0.6519
$\frac{\log_2\left(1+\frac{.5(.64)}{.2501}\right)}{2}$ =0.5944	$\frac{\log_2\left(1+\frac{.5(.49)}{.3601}\right)}{2}$ =0.3744	$\frac{\log_2\left(1+\frac{.5(.25)}{.6401}\right)}{2}$ =0.1287	$\frac{\log_2\left(1+\frac{.25}{.3201}\right)}{2}$ =0.4163

Table 2.6: Some useful calculations for the upcoming example

Since .8 > .7 and .6 > .5, and the noises have the same variance, the input clearly need not have a separate subcomponent for each receiver, thus 4 orders are the maximum to consider $(2!)^2$. instead of $(4!)^2 = 576$. With Table 2.6's calculations enumerated, Table 2.7 evaluates initially 4 possible orders $\boldsymbol{\Pi}$ (two become obviously superfluous in so doing) and the enumeration of $\boldsymbol{\mathcal{I}}_{min}$ for each order and for the input energies $\mathcal{E}_1 = \mathcal{E}_2 = 1$:

$\boldsymbol{\Pi}$	u	$\mathcal{E}_u$	$\mathbb{P}_u(\boldsymbol{\pi}_u)$	$\mathscr{I}_u(x_u; y_u/\mathbb{P}_u(\boldsymbol{\pi}_u))$	$\mathscr{I}_{\neg u}(x_u; y_{\neg u}/\mathbb{P}_{\neg u}(\boldsymbol{\pi}_{\neg u}))$	$\mathcal{I}_{min,u}(\boldsymbol{\Pi}, \boldsymbol{\mathcal{E}})$
$\lceil 2 \quad 2 \rceil$	2	1	{1}	6.322	∞	6.322
$\lfloor 1 \quad 1 \rfloor$	1	1	$\emptyset$	.3973	.2378	.2378
$\lceil 1 \quad 1 \rceil$	2	1	$\emptyset$	.9157	.6196	.6196
$\lfloor 2 \quad 2 \rfloor$	1	1	{2}	5.9071	∞	5.9071
$\lceil 1 \quad 2 \rceil$	2	1	$\emptyset$	.9157	∞	.9157
$\lfloor 2 \quad 1 \rfloor$	1	1	$\emptyset$	.3973	∞	.3973
$\lceil 2 \quad 1 \rceil$	2	1	{1}	6.322	.2378	.2378
$\lfloor 1 \quad 2 \rfloor$	1	1	{2}	5.9071	.6196	.6196

Table 2.7: Evaluation of $\boldsymbol{\mathcal{I}}_{min}$ for different orders.
Red color indicates a limiting value and affects $\mathcal{I}_{min}$, while green does not.

The table's first row (which is really two subrows together) considers an order that is decode user 2 (top) last at each receiver, when both users have full energy of one unit. There is a prior-user set containing user 1 for user 2's decoding, but the prior-user set is empty for user 1's decoding. The data rate for user 2 is in the top subrow, and is infinite at receiver 1 because it need not be decoded there. The $\mathcal{I}_{min}$ vector takes the smallest of the subrow entries for each of its subrows. Similarly, the table's second row considers an order that is decode user 1 last at each receiver, when both users have full energy of one unit. There is a prior user set containing user 2 for user 1's decoding, but this set is empty for user 2's decoding. The data rate for user 1 is in the bottom subrow, and is infinite at receiver 2 because it need not be decoded there. The $\mathcal{I}_{min}$ vector takes the smallest of the subrow entries for each of its subrows. Since both MAC-set channels have common primary/secondary components/users, only $U = 2$ orders need consideration. Also, no subuser partitioning can help on this IC

The shaded cells represent orders that are superfluous because these rate vectors are within the region that in the next achievable-rate-region step of dimension sharing (convex hull over order/vertices) would be a subset. These rows are not necessary to compute the region's boundary. Figure 2.77 plots the two points from the table (the unshaded rows), which allows first estimate of the rate region with the dimension-sharing pentagon for this energy

combination of $\boldsymbol{\mathcal{E}} = [1\ 1]^*$. Reducing the energy of one user's energy by 3 dB , as well as by the smaller .22 dB ($\mathcal{E} = 0.95$) produces a few more points, as in Table 2.8:

$\boldsymbol{\Pi}$	u	$\mathcal{E}_u$	$\mathbb{P}_u(\boldsymbol{\pi}_u)$	$\mathscr{I}_u(x_u; y_u/\mathbb{P}_u(\boldsymbol{\pi}_u)$	$\mathscr{I}_u(x_{\neg u}; y_{\neg u}/\mathbb{P}_{\neg u}(\boldsymbol{\pi}_{\neg u})$	$\mathcal{I}_{min,u}(\boldsymbol{\Pi}, \boldsymbol{\mathcal{E}})$
$\lceil 2\ \ 2 \rceil$	2	1	$\{1\}$	6.322	∞	6.322
$\lfloor 1\ \ 1 \rfloor$	1	0.5	$\emptyset$	.2257	.1287	.1287
$\lceil 1\ \ 1 \rceil$	2	1	$\emptyset$	1.3063	.9478	.9478
$\lfloor 2\ \ 2 \rfloor$	1	0.5	$\{2\}$	5.4073	∞	5.4073
$\lceil 2\ \ 2 \rceil$	2	0.5	$\{1\}$	5.822	∞	5.822
$\lfloor 1\ \ 1 \rfloor$	1	1	$\emptyset$	.6519	.4163	.4163
$\lceil 1\ \ 1 \rceil$	2	0.5	$\emptyset$	5.822	.3744	.3744
$\lfloor 2\ \ 2 \rfloor$	1	1	$\{1\}$	5.9071	∞	5.9071
$\lceil 2\ \ 2 \rceil$	2	1	$\{1,2\}$	6.322	∞	6.322
$\lfloor 1\ \ 1 \rfloor$	1	0.95	$\{1\}$	.3818	.2276	.2276
$\lceil 1\ \ 1 \rceil$	2	1	$\{2\}$	.9425	.6412	.6412
$\lfloor 2\ \ 2 \rfloor$	1	0.95	$\{1,2\}$	5.8701	∞	5.8701

Table 2.8: More example points with the best orders

Some of these points also appear in Figure 2.77(a). The $\mathcal{I}_{min}$-vertex-sharing line has formula

$$b_2 = \frac{6.322 - .6196}{5.9071 - .2378} \cdot b_1 + (6.322 + 1.006 \cdot .2378) = -1.006 \cdot b_1 + 6.5612 \tag{2.597}$$

Figure 2.77(a)'s dimension-shared pentagon is for these IC specific values and is $\mathcal{C}(\boldsymbol{b})$'s outer boundary via the following argument: Calculation of the two derivatives at the upper vertex elements with respect to $\mathcal{E}_2$ finds:

$$\ln(2) \cdot \frac{db_2}{d\mathcal{E}_2} = \frac{3200}{6400 \cdot \mathcal{E}_2 + 1} = 0.4999 \tag{2.598}$$

$$\ln(2) \cdot \frac{db_1}{d\mathcal{E}_2} = -\frac{3200 \cdot 2500 \cdot \mathcal{E}_1}{6400 \cdot (\mathcal{E}_2 + 2500 \cdot \mathcal{E}_1 + 1) \cdot (6400 \cdot \mathcal{E}_2 + 1)} = -.1404 \quad . \tag{2.599}$$

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 magnitude line-slope of -3.56. It slopes downward more quickly than rightward than (2.597). Thus, the curvature $d\mathcal{E}/db_2$ is within the pentagon shown (this need not occur in general and depends on $\widetilde{H}$). Figure 2.77(a) shows that points evaluated for either user's reduced energy therefore fall in the interior also. Search therefore need not include the 3^{rd} mixed order. The convex combination of the vertices corresponds to the same order at all receivers, thus U such orders are sufficient.

An enlarged capacity region also appears in Figure 2.77(b) for the energy-sum IC case where the energy constraint relaxes to $\mathcal{E}_{\boldsymbol{x}} = \mathcal{E}_1 + \mathcal{E}_2$. This region has 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 the vertex $b_2 = 6.822$ and $b_1 = 0$ and tends to infinity when $\mathcal{E}_1 \to 0$, so dimension-sharing again dominates for this $\widetilde{H}$. Indeed any relaxed energy constraint will ensure that curvature would be below the $\mathcal{I}_{min}$-vertex-sharing line.

Maximum rate sum: For two vertices as in this 2-user example, then it is possible to see that a line with slope -1 , so $b_1 + b_2 = b_{max}$, lowers from the upper right with b_{max} and must first touch the $\mathcal{I}_{min}$ vertex with maximum sum rate. This method is easier in two dimensions than the partial derivatives.

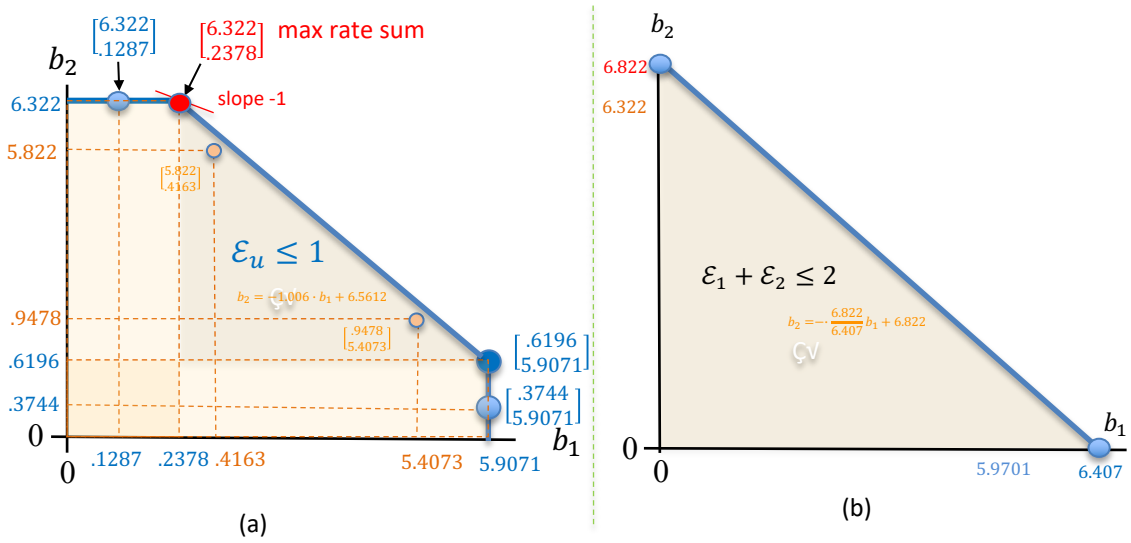


Figure 2.77: Capacity region for Example 2.9.1's IC.

A second example is the co-called **weak interference channel**, which leads to different primary and secondary users within the IC's MAC set and requires a larger search of orders $\mathbf{\Pi}$.

EXAMPLE 2.9.2 (Weak Symmetric IC) The scalar 2×2 **weak symmetric interference channel** has channel description

$$\begin{bmatrix} y_2 \\ y_1 \end{bmatrix} = \begin{bmatrix} 1 & \alpha \\ \alpha & 1 \end{bmatrix} \cdot \begin{bmatrix} x_2 \\ x_1 \end{bmatrix} + \begin{bmatrix} n_2 \\ n_1 \end{bmatrix} \tag{2.600}$$

with $\alpha \to 0$ with $R_{\boldsymbol{nn}} = I$ and $\boldsymbol{\mathcal{E}} \preceq \mathbf{1}$. The limiting channel is clearly two independent users, each with capacity $\mathcal{C}_u = \frac{1}{2} \cdot \log_2(2) = \frac{1}{2}$ bit/symbol. The sum rate's upper bound is then clearly 1 and attained when $\alpha = 0$. For $0 < \alpha < 1$, the interference is "weaker" than the main user of interest at both receivers, and the primary user is that corresponding main user at each receiver in the MAC set; thus, so not the same primary at all receivers as in Example 2.9.1. The corresponds to Figure 2.78's black rate vector, and the order where each user treats the other as noise is better than either decoding the other first (because of the minimum operation that occurs with $\boldsymbol{\mathcal{I}}_{min}$ that relates this data rate reduces too much consequently at the other receiver that did not decode it). In the figure, this black rate-pair vector moves outside the pentagon of the other two orders when $\alpha < 1$ and moves to the upper right maximum point when $\alpha = 0$. The third order choice is clearly necessary, while the 4th order choice (not shown) basically forces zero data rate at both users through this minimization and is thus trivially ignored. Each users-energy pair produces different black rate-pair vectors so then the two convex hull operations look like first taking the $\boldsymbol{\mathcal{I}}_{min}$-vertices' union/hull to create the 5-or-6-sided polygons (5 or 6 depends on the channel's choice of α). When $\alpha = 0$, $\mathcal{C}(\boldsymbol{b})$ is a 4-sided special case or square. Figure 2.78 illustrates the achievable region when the energies are exactly $\mathcal{E}_1 = \mathcal{E}_2 = 1$ (so not less than 1); when energies are less than one, the convex hull operation over energy pairs can lead to curved regions replacing the pentagons or 6-sided polygons. The second convex hull operation corresponds to varying the different choices of the energy vector within the constraint (and of course repeating the first convex hull operation within a first step for each such energy-pair choice). As $\alpha \to 0$, the upper right black point increasingly separates from the line between the other two vertices. In this case, the number of searched orders is 3.

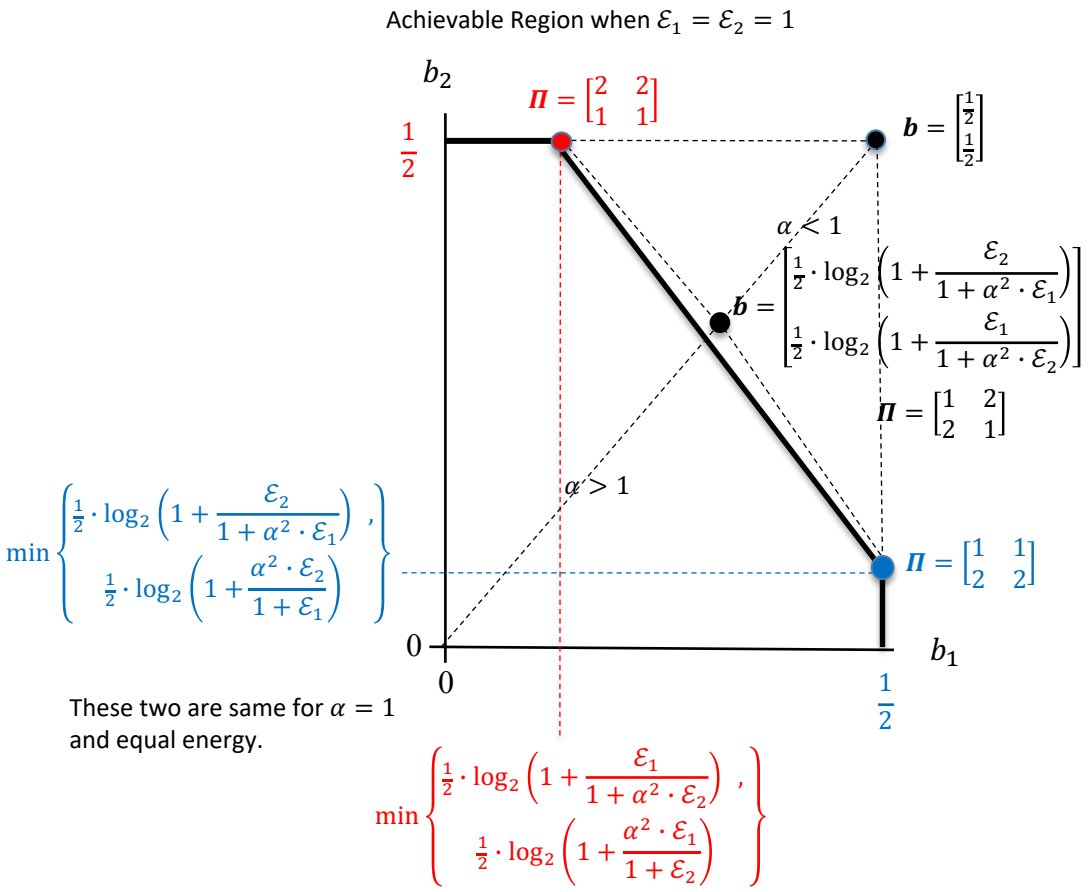


Figure 2.78: Weak symmetric interference-channel example of Subsection 2.9.1.1; Achievable region when $\mathcal{E}_1 = \mathcal{E}_2 = 1$.

Both these examples illustrate for the $U = 2$ case that the search of up to $(U!)^U$ orders may be necessary, depending on the channel. The general method here provides the capacity region, but may require a larger order search and nested convex hull operations. It will produce or exceed the best previously known[111] capacity regions such as the Han-Kobayashi region [5], depending on the channel. The key is the embedded multiple convex-hull operations in Section 2.6's general capacity region. While complex, the multiple embedded hull operations obviate the need for the complex auxiliary variables found in other approaches. The commonl/private user decdompositions subsume into this approaches' larger order search. The converse readily follows from the MMSE (with feedback or successive decoding) for each and every point that is found. Any user rate vector that has even one user outside the rate region by $\epsilon > 0$ necessarily violates the single user capacity theorem in resultant AWGN that results. Thus points outside the region are not reliably achievable for at least one user.

Scalar IC Order Generalization: Examples 2.9.1 and 2.9.2 provide insight: The order matrix $\mathbf{\Pi}$ with user u simultaneously at the top (most preferred position) for all users in the MAC-set approach is important, particularly to determine some 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 data-rate values for this same user at other receivers then become irrelevant (set at ∞) because

[111] Prior to this text.

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. However, receiver u itself has users $\{i \neq u\} \in \mathcal{D}_u$ necessary for decoding, and these users may severely limit at least one $i \neq u$ at another receiver where they are the desired user or part of that user's decodable set $\mathcal{D}_i$ for the given order $\mathbf{\Pi}$. Example 2.9.2 illustrates this affect as $\alpha \to 0$ because decoding the other user's signal with very low SNR then limits the data rate that might otherwise be very high at the intended receiver. This situation illustrates how more orders from the set of $|\mathbf{\Pi}|$ possible orders may be necessary for search, even though the same-at-all-receivers $U!$ users usually is important to check first. Clearly one of the 4 orders in Example 2.9.2 is trivially unnecessary, and for larger U there may be also many obviously poor orders that search can omit trivially. Ultimately, primary-user components common to all MACs reduces order search. The area of search-space reduction is open for further contribution.

The convex combination of the two larger-$\boldsymbol{\mathcal{I}}_{min}$ vertex points creates Figure 2.78's line, 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 means that the maximum sum point b_{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. Example 2.9.1 has all points below the slope=-1.006 line; but in general this need not be true.

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 $\mathcal{C}_{G-IC}(\boldsymbol{b})$'s construction. In this final step, the vertices may simply fade into a smooth region boundary.

2.9.2 Scalar IC Generalizations on curvature and convex-hull:

The 2×2 scalar Gaussian IC generalizes from Example 2.8.7 with (but now as an IC)

$$\begin{array}{ll} 6400 \to g_{22} & 4900 \to g_{21} \\ 2500 \to g_{12} & 3600 \to g_{11} \end{array} \quad . \; . \tag{2.601}$$

The two general mutual-information derivative expressions then are

$$\ln(2) \cdot \frac{db_2}{d\mathcal{E}_2} = \frac{1/2}{\mathcal{E}_2 + 1} \tag{2.602}$$

$$\ln(2) \cdot \frac{db_1}{d\mathcal{E}_2} = -\frac{1/2 \cdot g_{12} \cdot \mathcal{E}_1}{(\mathcal{E}_2 + g_{12} \cdot \mathcal{E}_1 + 1) \cdot (g_{22} \cdot \mathcal{E}_2 + 1)} \quad \textbf{or} \tag{2.603}$$

$$\ln(2) \cdot \frac{db_1}{d\mathcal{E}_2} = -\frac{1/2 \cdot g_{11} \cdot \mathcal{E}_1}{(\mathcal{E}_2 + g_{11} \cdot \mathcal{E}_1 + 1) \cdot (g_{21} \cdot \mathcal{E}_2 + 1)} \quad , \tag{2.604}$$

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 capacity region instead of a pentagon. This is the case in Example 2.9.2 where the mixed-order needs inclusion. When this happens, a construction that finds other pentagons (orthotopes) that are not necessarily within the original pentagon (orthotope). Then the convex hull (union) of energy combinations becomes more complex and tedious. Further, the pentagon could be come a rectangle for very small g_{21} and/or g_{12}. Figure 2.79 shows the 3 possibilities. Chapter 5, Section 6 has algorithms that attempt to circumvent the large order search step for design at a single $\boldsymbol{b}' \in \mathcal{C}(\boldsymbol{b})$.

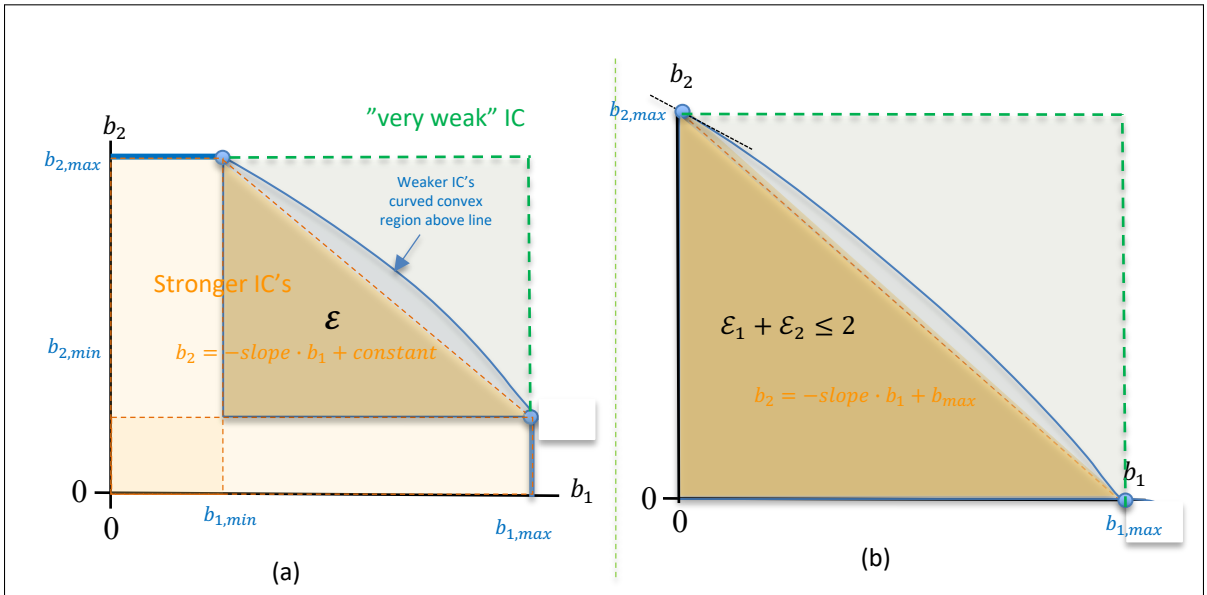


Figure 2.79: IC capacity-region shape types.

Derivative test: Some authors, who use a different more situationally dependent characterization of posed capacity regions, basically enumerate different magnitude of the g_{ij} for the 2×2 case (see for instance [5], See p. 140, Figure 6.3). Alternatively, the successive steps of first $\boldsymbol{\mathcal{I}}_{min}$ calculations and then convex combinations of these possibilities over the necessary searched orders, and then variation of energy, in an outer step possibly involving a second level of convex hull and dimension sharing provides the general IC capacity region. Fortunately, the various shapes thereof relate to the relative derivative magnitudes. In general, the derivative test generalizes to finding each vertex' possible derivative sets

$$\{\partial b_u / \partial \mathcal{E}_u\} \, . \tag{2.605}$$

There is a dimension-sharing plane of all the $b_{i \neq u, max}$ with b_u 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 $\mathcal{C}(\boldsymbol{b})$ will curve above the plane to that other point $b_{i \neq u}$. More orders need search. When the derivative's magnitudes reverse, the dimension-sharing plane bounds the region, search and need only consider $U!$ common-at-all-receivers orders. This derivative test, while not trivial, is perceptively easier than characterizing into strong, weak, very strong, etc, particularly when $U > 2$ when that characterization becomes overly complex. The basic derivative-set test(s) at each vertex is (are) sufficient to sketch the region.

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 2.9.3 [4×2 IC with 2 users.] A Gaussian IC has a 4×2 channel matrix

$$\underbrace{\boldsymbol{y}}_{4\times 1} = \begin{bmatrix} \boldsymbol{y}_2 \\ \boldsymbol{y}_1 \end{bmatrix} = \underbrace{\begin{bmatrix} H_2 \\ H_1 \end{bmatrix}}_{4\times 2} \underbrace{\begin{bmatrix} x_2 \\ x_1 \end{bmatrix}}_{2\times 1} + \underbrace{\begin{bmatrix} \boldsymbol{n}_2 \\ \boldsymbol{n}_1 \end{bmatrix}}_{4\times 1} \, , \tag{2.606}$$

where the constituent MACs are

$$H_2 = \begin{bmatrix} \boldsymbol{h}_{22} & \boldsymbol{h}_{21} \end{bmatrix} = \begin{bmatrix} .9 & .3 \\ .3 & .8 \end{bmatrix} \tag{2.607}$$

and

$$H_1 = \begin{bmatrix} \boldsymbol{h}_{12} & \boldsymbol{h}_{11} \end{bmatrix} = \begin{bmatrix} .8 & .7 \\ .6 & .5 \end{bmatrix} \tag{2.608}$$

and $R_{\boldsymbol{nn}} = 0.01 \cdot I_4$. Each receiver's two dimensions, with two users ($U = 2$), allows each constituent MAC-set element to be all 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

$$R_{\boldsymbol{xx}} = \begin{bmatrix} \mathcal{E}_2 & 0 \\ 0 & \mathcal{E}_1 \end{bmatrix} \quad . \tag{2.609}$$

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 example again requires search of only two orders, each with the same user in upper priority position at both receivers, if the separate energy constraints of (2.609) apply individually[112]. The two orders are $\mathbf{\Pi} = [1\ 1; 2\ 2]$ and $\mathbf{\Pi} = [2\ 2; 1\ 1]$.

Receiver 2's best order places user 2 in the upper position where there is no crosstalk while receiver 1's best 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.

```
>> H2 = [9     3
        3     8 ];
>> Rb2inv=H2'*H2+diag([1 1]) =
    91    51
    51    74
>> Gbar2=chol(Rb2inv) =
    9.5394    5.3463
         0    6.7393
>> G2=inv(diag(diag(Gbar2)))*Gbar2 =
    1.0000    0.5604
         0    1.0000
>> S02=diag(diag(Gbar2))*diag(diag(Gbar2)) =
   91.0000         0
         0   45.4176
>>  0.5*log2(diag(S02)) =
b2 =  3.2539
b1 =  2.7526

>>  H1 = [8     7
         6     5 ];
>> Rb1inv=H1'*H1+diag([1 1]) =
   101    86
    86    75
>> Gbar1=chol(Rb1inv);
>> S01=diag(diag(Gbar1))*diag(diag(Gbar1));
>> 0.5*log2(diag(S01)) =
b2 =  3.3291
b1 =  0.4128
```

These two points determine vertex A by taking the smallest value for each user so (A) at $\boldsymbol{b}_A = [3.2539\ .4128]^*$ in Figure 2.80. To find the other vertex (B) at $\boldsymbol{b}_B$:

[112] However, with a sum-energy constraint, there is only 1 order that is determined by ranking $\widetilde{H}_u$ according to $|\widetilde{H}_u \cdot \widetilde{H}_u^*|$.

```
>> J2=hankel([0 1]);
>> Rb2inv=J2*H2'*H2*J2+diag([1 1]) =
    74    51
    51    91
>> Gbar2=chol(Rb2inv);
>> S02=diag(diag(Gbar2))*diag(diag(Gbar2));
>> 0.5*log2(diag(S02)) =
b1 =  3.1047
b2 =  2.9018

>> Rb1inv=J2*H1'*H1*J2+diag([1 1]);
>> Gbar1=chol(Rb1inv);
>> S01=diag(diag(Gbar1))*diag(diag(Gbar1));
>> 0.5*log2(diag(S01)) =
b1 =  3.1144
b2 =  0.6275
```

So, $\boldsymbol{b}_B = [3.1047\ .6275]^*$.

For $\mathcal{E}_2 = 0.5$, then (C) $\boldsymbol{b}_C = [2,7618\ .6581]^*$ and (B') $\boldsymbol{b}_C = [.3799\ .6581]^*$:

```
>> Rb2inv=diag([1/sqrt(2) 1])*H2'*H2*diag([1/sqrt(2) 1])+diag([1 1]) =
   46.0000   36.0624
   36.0624   74.0000

>> Gbar2=chol(Rb2inv);
>> S02=diag(diag(Gbar2))*diag(diag(Gbar2));
>> 0.5*log2(diag(S02)) =
b2=    2.7618
b1=    2.7575

>> Rb1inv=diag([1/sqrt(2) 1])*H1'*H1*diag([1/sqrt(2) 1])+diag([1 1]) =
   51.0000   60.8112
   60.8112   75.0000
>> Gbar1=chol(Rb1inv);
>> S01=diag(diag(Gbar1))*diag(diag(Gbar1));
>> 0.5*log2(diag(S01)) =
b2=    2.8362
b1=   0.6581
>>J2=hankel([0 1]);

>> Rb2inv=diag([1 1/sqrt(2)])*J2*H2'*H2*J2*diag([1 1/sqrt(2)])+diag([1 1]) =
   74.0000   36.0624
   36.0624   46.0000
>> Gbar2=chol(Rb2inv);
>> S02=diag(diag(Gbar2))*diag(diag(Gbar2));
>> 0.5*log2(diag(S02)) =
b1 =    3.1047
b2 =    2.4146

>> Rb1inv=diag([1 1/sqrt(2)])*J2*H1'*H1*J2*diag([1 1/sqrt(2)])+diag([1 1]) =
   75.0000   60.8112
   60.8112   51.0000
>> Gbar1=chol(Rb1inv);
```

```
>> S01=diag(diag(Gbar1))*diag(diag(Gbar1));
>> 0.5*log2(diag(S01)) =
b1 =    3.1144
b2 =    0.3799
```

For $\mathcal{E}_1 = 0.5$ and (D) $\boldsymbol{b}_D = [.9407\ 2.6144]^*$ and (A') $\boldsymbol{b}_{A'} = [3.2539\ .2355]^*$::

```
>> 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
```

The slope of the line connecting the two vertices is -.9722, which has magnitude less than 1, so this means the curvature at the $\boldsymbol{\mathcal{I}}_{min}$ vertex with the larger rate sum (B) must have greater magnitude than the line. Correspondingly, the maximum rate-sum point occurs when a -1 slop line is tangent and occurs along Figure 2.80's curved region that curves away from point (B) and upward and to the left eventually touching the other vertex with slope magnitude less than .9722.

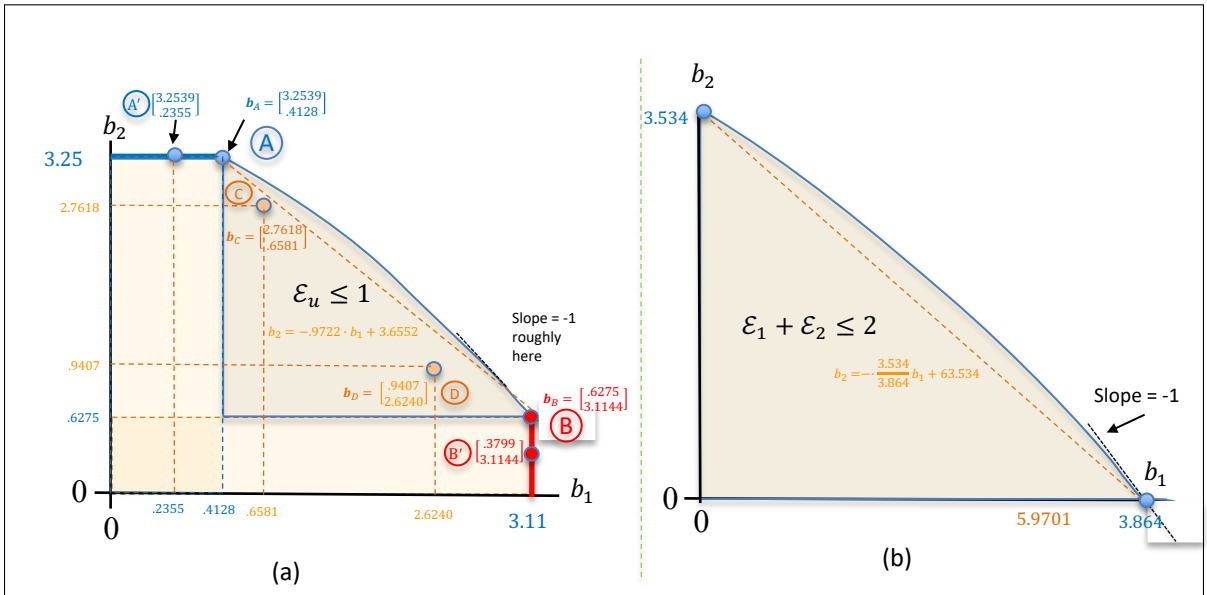


Figure 2.80: Achievable regions for given energy and 4×2 IC, as in Example 2.9.3

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

$$\mathcal{A}_{GIC}(\boldsymbol{b}, R_{\boldsymbol{xx}}) = \bigcup_{\boldsymbol{\Pi}}^{conv} \mathcal{I}_{min}(\boldsymbol{\Pi}, R_{\boldsymbol{xx}}) \ . \tag{2.610}$$

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 for the given $R_{\boldsymbol{xx}}$. The convex hull operation corresponds to the vertex-sharing. This vertex sharing essentially reduces the order search because for the primary users, the only $\boldsymbol{\Pi}$'s that matter are the ones 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) subsume all the other orders. When the set $\boldsymbol{u}^s \neq \emptyset$, then there is a possibility of $(U^s+1)!$ different orders that use 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 $(U^s+1)! \leq (U!)^U$ possible orders (often significantly less).

The convex hull of all permitted $R_{\boldsymbol{xx}}$ that conform to the energy limit $\boldsymbol{\mathcal{E}}$ then completes the capacity region as

$$\mathcal{C}_{GIC}(\boldsymbol{b}) = \bigcup_{trace\{R_{\boldsymbol{xx}}\} \leq \boldsymbol{\mathcal{E}}}^{conv} \mathcal{A}(\boldsymbol{b}, R_{\boldsymbol{xx}}) \ . \tag{2.611}$$

The examples illustrated a portion of this step by allowing some additional energy choices (namely total sum energy), but not all choices.

2.9.3 The BC-set approach

The BC-set approach uses duality. Each of the MAC's in the MAC approach is the dual of a corresponding BC with $\widetilde{H}^*$ 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 $\mathcal{I}_{min}$ step where the minima of different possibilities correspond to the exclusion of energy from the precoders. This BC-set approach is useful in Section 2.11's relay/mesh channels' $\mathcal{C}(\boldsymbol{b})$ characterization.

2.10 Generalized multiUser Channels

Many multiuser 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 2.81 provides an example mixture that Subsection 2.10.1 investigates in more detail. Subsection 2.10.1 also 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[113]. Subsection 2.10.1 computes the desired capacity region from expanded achievable regions. Expanded multiuser channels leverage the concept of order and successive decoding (or precoding). Recall, for instance, that a 2×2 two-user Gaussian BC has a 4×4 precoder, with essentially two sub-users/user that correspond to each users' influence at each of the two channel-input dimensions with each component having its own sub-user code and corresponding contribution to user data rate.

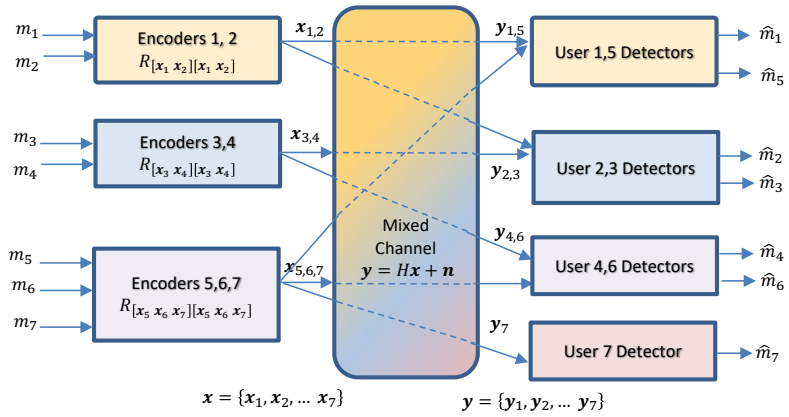


Figure 2.81: Mixed Gassian Channel.

On the other hand, a design could use many users within a sub-communications channel also as a macro user. For instance, a macro-level IC might have (probably un-cancellable or un-decodable) crosstalk between (think radio towers) nodes that each have a combination of downlink BC and uplink MAC. The individual MACs and BC's could view other systems' MAC's and BC's as just additional crosstalk (most likely not decodable, but possibly so) with macro decisions on spectrum/space (dimensions) to be used by each node, and then that dimensionality suballocated within each of the nodes. This is an example of a **nested** multiuser communication channel. Figure 2.81 would illustrate nesting if there were 3 decoders respectively coordinating for the user groups (1,2), (3,4), and (5, 6,7) and all crosstalk between those groups viewed as noise (or possibly a macro user that might be reliably decoded all or in part) at the 3 receivers.

2.10.1 User Expansion and the Capacity Region

Figure 2.81 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 multiuser 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 multiuser channel has a number of sub-users U that exceeds the number of transmit (MAC, IC, or general multiuser) or receive (BC, IC, or general multiuser) locations

[113] "larger" means that U is enlarged so that the rate-region plot has $|\boldsymbol{b}| > U$ dimensions.

in the channel description. For instance, the Figure 2.81's 2-user MAC $[5, 1]$ may expand its user set to all 7 users, call this 7-dimensional bit vector then $\boldsymbol{b}_e$. The input autocorrelation matrix $R_{\boldsymbol{xx}}$ spans all 7 users. For this expanded-user MAC, for any $0 \leq b_{i\neq 1,5} < \infty$ and the given $R_{\boldsymbol{xx}}$, the achievable region $\mathcal{A}(\boldsymbol{b}_e, R_{\boldsymbol{xx}})$ is the same pentagon for b_1 and b_5. Indeed, if Figure 2.81's lines represent 1 dimension each, this 7-user region expands in 5 positive half spaces to ∞ on the users not received, essentially viewing these users' receivers with zero-energy noises. The concept generalizes conceptually if the users have more than 1 dimension.

For the same $R_{\boldsymbol{xx}}$, 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 $R_{\boldsymbol{xx}}(7)$ that is within $R_{\boldsymbol{xx}}$. Repeating from (2.251):

$$\mathcal{I}_{min,u}(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}) = \min_{i\in\{\mathbb{P}_{\pi_i^{-1}(u)},u\}} \left\{ \mathscr{I}_i \left(\boldsymbol{x}_{\pi_i^{-1}(u)}; \boldsymbol{y}_i / \mathbb{P}_{\pi_i^{-1}(\boldsymbol{\pi}_i)} \right) \right\} \quad . \tag{2.612}$$

The achievable region then follows as

$$\begin{aligned} \mathcal{A}_{GMUC}(\boldsymbol{b}, R_{\boldsymbol{xx}}) \quad = \quad & \bigcup_{\boldsymbol{\Pi}}^{conv} \{\mathcal{A}_{5,1}(\boldsymbol{b}_{e,5,1}, \boldsymbol{\pi}_{5,1}, R_{\boldsymbol{xx}}); \mathcal{A}_{3,2}(\boldsymbol{b}_{e,3,2}, \boldsymbol{\pi}_{3,2}, R_{\boldsymbol{xx}}); \\ & \mathcal{A}_{6,4}(\boldsymbol{b}_{e,6,4}, R_{\boldsymbol{xx}}); \mathcal{A}_7(\boldsymbol{b}_{e,7}, R_{\boldsymbol{xx}})\} \quad . \end{aligned} \tag{2.613}$$

is compact for each $R_{\boldsymbol{xx}}$. Then, the capacity region searches over all $R_{\boldsymbol{xx}}$ that satisfy the energy constraints $R_{\boldsymbol{xx}} \in \{R_{allowed}\}$:

$$\mathcal{C}_{GMUC}(\boldsymbol{b}) = \bigcup_{R_{\boldsymbol{xx}}\in\{R_{allowed}\}}^{conv} \mathcal{A}(\boldsymbol{b}, R_{\boldsymbol{xx}}) \quad . \tag{2.614}$$

The set $R_{allowed}$ in this case would likely have 3 constraints:

$$trace\{R_{[\boldsymbol{x}_1\boldsymbol{x}_2][\boldsymbol{x}_1\boldsymbol{x}_2]}\} \leq \mathcal{E}_{1,2} \tag{2.615}$$

$$trace\{R_{[\boldsymbol{x}_3\boldsymbol{x}_4][\boldsymbol{x}_3\boldsymbol{x}_4]}\} \leq \mathcal{E}_{3,4} \tag{2.616}$$

$$trace\{R_{[\boldsymbol{x}_5\boldsymbol{x}_6\boldsymbol{x}_7][\boldsymbol{x}_5\boldsymbol{x}_6\boldsymbol{x}_7]}\} \leq \mathcal{E}_{5,6,7} \quad , \tag{2.617}$$

and these would guide the search over possible input autocorrelation matrices, each with its own independent minimization in (2.617) and convex hull over orders in (2.617).

Generalized BC Set: The BC-set approach again would repeat the MAC region generation for each BC's dual MAC. In this case the dual channels would decompose as (with a superscript of $\perp$ signifying dual channel)

$$\mathcal{A}_{GMUC}(\boldsymbol{b}, R_{\boldsymbol{xx}}) = \bigcup_{\boldsymbol{\Pi}}^{conv} \left(\mathcal{A}_{1,2}^{\perp}(\boldsymbol{b}_{e,1,2}, \boldsymbol{\pi}_{1,2}, R_{\boldsymbol{xx}}); \mathcal{A}_{3,4}^{\perp}(\boldsymbol{b}_{e,3,4}, \boldsymbol{\pi}_{3,4}, R_{\boldsymbol{xx}}); \mathcal{A}_{5,6,7}^{\perp}(\boldsymbol{b}_{e,5,6,7}, R_{\boldsymbol{xx}}) \right) \quad . \tag{2.618}$$

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 (2.614). The BC-set approach finds use in Subsection 2.10.2's relay-channel capacity-region construction.

2.10.2 Nesting

This section to be added later.

2.11 Relay and Mesh Multiuser channels

Subsection 2.11.1 addresses relay channels while Subsection 2.11.2 addresses **mesh networks**.

2.11.1 Relay Channel Capacity Region

An alternative extended-user channel example is Figure 2.82'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 2.82'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 $U \times k_r$ **bit matrix** $\boldsymbol{B}$, where each column is a $U \times 1$ bit vector $\boldsymbol{b}_k$ with $k = 1, ..., k_r$ elements that represents users' bits/subsymbols to relay k. Each relay has a first-stage-side achievable region $\mathcal{A}_k^{(1)}\left(\boldsymbol{B}^{(1)}, R_{\boldsymbol{xx}}^{(1)}\right)$, paralleling the IC approach but retaining the extended users through the subscript $k = 1, ..., k_r$ so $U \cdot k_r$ total users. The first (left) stage thus has a $k_r \times U$ bit matrix:

$$\boldsymbol{B}^{(1)} \triangleq \left[\boldsymbol{b}_{k_r}^{(1)}, ..., \boldsymbol{b}_1^{(1)}\right] \quad , \tag{2.619}$$

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 $k = 1, .., k_r$ are

$$\boldsymbol{b}_{k,min}^{(1)}(\boldsymbol{\pi}_k^{(1)}, R_{\boldsymbol{xx}}^{(1)}) \triangleq \begin{bmatrix} \min_{u \in \mathcal{S}_U(\boldsymbol{\Pi}^{(1)}, R_{\boldsymbol{x},\boldsymbol{x}}^{(1)})} \left\{ b_{[\pi^{(1)}]^{-1}(u)}(k, R_{\boldsymbol{xx}}^{(1)}) \right\} \\ \vdots \\ \min_{u \in \mathcal{S}_1(\boldsymbol{\Pi}^{(1)}, R_{\boldsymbol{x},\boldsymbol{x}}^{(1)})} \left\{ b_{[\pi^{(1)}]^{-1}(u)}(k, R_{\boldsymbol{xx}}^{(1)}) \right\} \end{bmatrix} \quad , \tag{2.620}$$

where the minimum is over Definition 2.6.4's cross-user set.

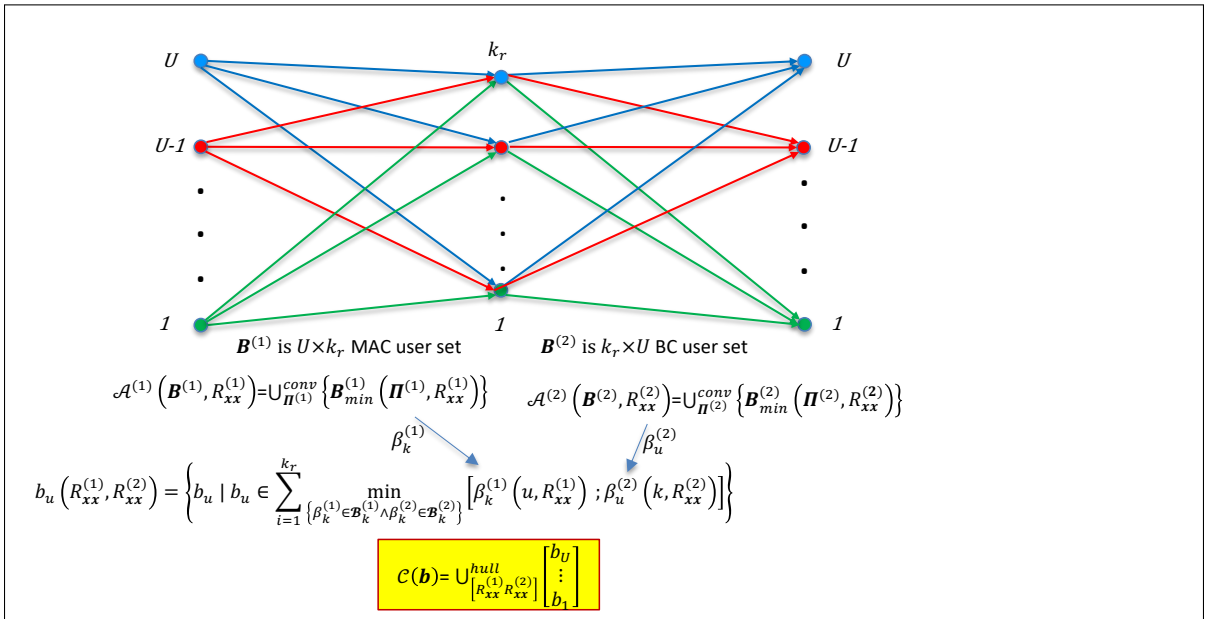


Figure 2.82: Simple Two-Stage relay channel.

The $U \cdot k_r$-dimensional first-stage extended-user achievable region is then the convex-hull combination

$$\mathcal{A}^{(1)}(\boldsymbol{B}^{(1)}, R_{\boldsymbol{xx}}^{(1)}) = \bigcup_{\boldsymbol{\Pi}^{(1)}}^{conv} \left\{ \boldsymbol{B}_{min}^{(1)}\left(\boldsymbol{\Pi}^{(1)}, R_{\boldsymbol{xx}}^{(1)}\right) \right\} \quad . \tag{2.621}$$

The k_r achievable first-stage rate vector possibilities are then all the vectors $\boldsymbol{\beta}_k^{(1)}$ that satisfy

$$\boldsymbol{\mathcal{B}}_k^{(1)}(R_{\boldsymbol{xx}}^{(1)}) = \left\{ \boldsymbol{\beta}_k^{(1)} \mid \boldsymbol{\beta}_k^{(1)} \left(R_{\boldsymbol{xx}}^{(1)} \right) \in \mathcal{A}^{(1)} \left(\boldsymbol{B}^{(1)}, R_{\boldsymbol{xx}}^{(1)} \right) \right\} \quad . \tag{2.622}$$

The $\boldsymbol{\beta}_k^{(1)}$ vectors have individual vector elements $\beta_k^{(1)}(u, R_{\boldsymbol{xx}}^{(1)})$ 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

$$\boldsymbol{B}^{(2)} \triangleq \left[\boldsymbol{b}_U^{(2)}, ..., \boldsymbol{b}_1^{(2)} \right] . \quad , \tag{2.623}$$

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

$$\boldsymbol{b}_{u,min}^{(2)}(k, \boldsymbol{\pi}_u^{(2)}, R_{\boldsymbol{xx}}^{(2)}) \triangleq \begin{bmatrix} \min_{k \in \mathcal{S}_{k_r}(\boldsymbol{\Pi}^{(2)}, R_{\boldsymbol{x},\boldsymbol{x}})^{(2)}} \left\{ b_{[\pi^{(2)}]^{-1}(k)}(u, R_{\boldsymbol{xx}}^{(2)}) \right\} \\ \vdots \\ \min_{k \in \mathcal{S}_1(\boldsymbol{\Pi}^{(2)}, R_{\boldsymbol{x},\boldsymbol{x}}^{(2)})} \left\{ b_{[\pi^{(1)}]^{-1}(k)}(u, R_{\boldsymbol{xx}}^{(2)}) \right\} \end{bmatrix} . \tag{2.624}$$

The convex-hull-over-orders step becomes the $k_r \cdot U$-dimensional region

$$\mathcal{A}^{(2)}(\boldsymbol{B}^{(2)}, R_{\boldsymbol{xx}}^{(2)}) = \bigcup_{\boldsymbol{\Pi}^{(2)}}^{conv} \left\{ \boldsymbol{B}_{min}^{(2)}(\boldsymbol{\Pi}^{(2)}, R_{\boldsymbol{xx}}^{(2)}) \right\} \quad . \tag{2.625}$$

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 $\boldsymbol{\beta}_k^{(2)}$ that satisfy

$$\boldsymbol{\mathcal{B}}_u^{(2)}(R_{\boldsymbol{xx}}^{(2)}) = \left\{ \boldsymbol{\beta}_u^{(2)} \mid \boldsymbol{\beta}_u^{(2)} \left(R_{\boldsymbol{xx}}^{(2)} \right) \in \mathcal{A}^{(2)} \left(\boldsymbol{B}^{(2)}, R_{\boldsymbol{xx}}^{(2)} \right) \right\} \quad , \tag{2.626}$$

with individual elements $\beta_u^{(2)}(k, R_{\boldsymbol{xx}}^{(2)})$

A stage-combination step contracts the number of users to the original U and computes for each possible autocorrelation-stage pair $[R_{\boldsymbol{xx}}^{(1)}, R_{\boldsymbol{xx}}^{(2)}]$ the corresponding achievable region's entries through

$$b_u(R_{\boldsymbol{xx}}^{(1)}, R_{\boldsymbol{xx}}^{(2)}) = \left\{ b_u \mid b_u \in \sum_{k=1}^{k_r} \min_{\left\{ \beta_k^{(1)} \in \boldsymbol{\mathcal{B}}_k^{(1)} \wedge \beta_u^{(2)} \in \boldsymbol{\mathcal{B}}_u^{(2)} \right\}} \left[\beta_k^{(1)}(u, R_{\boldsymbol{xx}}^{(1)}), \beta_u^{(2)}(k, R_{\boldsymbol{xx}}^{(2)}) \right] \right\} . \tag{2.627}$$

The combined two-stage relay channel has U users and bits/subsymbol vector $\boldsymbol{b}$. 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 $\boldsymbol{\Pi}$ in each stage arises to compute the achievable regions $\mathcal{A}^{(1)} \left(\boldsymbol{B}, R_{\boldsymbol{xx}}^{(1)} \right)$ for the first relay stage and $\mathcal{A}^{(2)} \left(\boldsymbol{B}, R_{\boldsymbol{xx}}^{(2)} \right)$ 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 (2.627) then form a $U \times 1$ vector $\boldsymbol{b}$ and the capacity region is

$$\mathcal{C}_{RC2}(\boldsymbol{b}) = \bigcup_{[R_{\boldsymbol{xx}}^{(1)}, R_{\boldsymbol{xx}}^{(2)}]} \left\{ \begin{bmatrix} b_U(R_{\boldsymbol{xx}}^{(1)}, R_{\boldsymbol{xx}}^{(2)}) \\ \vdots \\ b_1(R_{\boldsymbol{xx}}^{(1)}, R_{\boldsymbol{xx}}^{(2)}) \end{bmatrix} \right\} \quad , \tag{2.628}$$

which is the two-stage **relay-channel capacity region**.

2.11.2 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

$$U^e = k_r^{(S-1)} \cdot k_r^{(S-2)} \cdot k_r^{(1)} \cdot U \tag{2.629}$$

users. $\boldsymbol{B}$ 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

$$\boldsymbol{\Pi} = \otimes_{s=1}^{S-1} \boldsymbol{\Pi}^s \quad . \tag{2.630}$$

The last stage has order matrix $\boldsymbol{\Pi}^S$ also with U^e users that contract to the original U users. The sum in (2.627) becomes an $S-1$ stage sum and the minimum is overall all the stages in the sum of the form:

$$\begin{aligned} b_u(R_{\boldsymbol{xx}}^{(1)}, R_{\boldsymbol{xx}}^{(2)} ... R_{\boldsymbol{xx}}^{(S)}) \quad = \\ \sum_{k=1}^{k_r^{S-1}} \sum_{k=1}^{k_r^{S-2}} ... \sum_{k=1}^{k_r^1} \min_{\left\{\beta_k^{(1)} \in \boldsymbol{\mathcal{B}}_k^{(1)} \wedge \beta_k^{(2)} \in \boldsymbol{\mathcal{B}}_k^{(2)} \wedge \beta_k^{(S-1)} \in \boldsymbol{\mathcal{B}}_k^{(S-1)} \wedge ... \wedge \beta_u^{(S)} \in \boldsymbol{\mathcal{B}}_u^{(S)}\right\}} \\ \left[\boldsymbol{\beta}_{k_r^{(1)}}^{(1)}(u, R_{\boldsymbol{xx}}^1), \boldsymbol{\beta}_{k_r^{(2)}}^{(2)}(k_r^{(1)}, R_{\boldsymbol{xx}}^2), \boldsymbol{\beta}_{k_r^{(3)}}^{(3)}(k_r^{(2)}, R_{\boldsymbol{xx}}^3), ..., \boldsymbol{\beta}_u^{(S)}(k_r^{(S-1)}, R_{\boldsymbol{xx}}^S)\right] \quad . \end{aligned} \tag{2.631}$$

2.11.3 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 (2.631). The minimum operation in (2.631) must consider any such recombination of sub users at any intermediate stage (or at the final stage) as the sum of $b_u(R_{\boldsymbol{xx}})$ 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 (2.631).

2.11.4 Reflective Intelligent Surfaces (RIS)

An Reflective Intelligent Surface (RIS) essentially is an "analog" repeater that can adjust the channel transfer $H(f)$ at an intermediate channel point. The modeling and optimization of channel adjustment can improve multiuser transmission, essentially by improving the channel capacity, or for multiple users, enlarging the channel-capacity region. RIS applies to the matrix AWGN. An RIS channel can be modeled by an H that is

$$\boldsymbol{y} = \underbrace{\begin{bmatrix} H_{los} \\ H_{out} \cdot Q_H \cdot H_{in} \end{bmatrix}}_{H_{RIS}} \cdot \boldsymbol{x} + \underbrace{\begin{bmatrix} \boldsymbol{n}_{los} \\ \boldsymbol{n}_{out} + Q_H \cdot \boldsymbol{n}_{in} \end{bmatrix}}_{\boldsymbol{n}_{RIS}} \quad . \tag{2.632}$$

Effectively, the channel has a controllable intermediate gain matrix[114] Q_H that corresponds to the RIS. The line-of-sight path channel H_{los} can be MIMO.

Different RIS strategies constrain the intermediate gain matrix so that

- Q_H is arbitrary subject to a constraint that $\|Q_H\|_F^2 \leq G_H$ (the RIS gain), an "active RIS," or
- Q_H is unitary so $Q_H^* \cdot Q_H^* = I$, meaning that energy is not increased by the RISl ("passive RIS"), and/or
- Q_H is diagonal, and/or

[114] An one-dimensional channel including the RIS provides no gain, so the RIS channel cannot be 1×1 .

- some or all elements of Q_H are individually constrained to be limited in magnitude and offer discrete phase adjustment choices $q_{H,i,j} = |q_{H,i,j}| \cdot e^{\jmath \cdot \theta_k}$ with $\theta_k \in \{0, \theta_1, ...\theta_K\}$ and $|q_{H,i,j}| \leq 1$.

The RIS basically consists of an $L_{in} \times L_{out}$ array of input to output antennas (usually $L_{in} = L_{out}$ because they are the same antennas. The diagonal cases correspond to simple gain/phase adjustment of the intermediate received signal on a particular antenna before returning it to that same channel. The first two cases allow a spatially filtered replica while the passive version ensures overall energy is not increased.

For a fixed $R_{\boldsymbol{xx}}$, the best (single-user) choice of Q_H maximizes

$$\mathcal{I}(\boldsymbol{y}; \boldsymbol{x}) = \log_2 |R_{n,RIS} + H_{RIS} \cdot R_{\boldsymbol{xx}} \cdot H_{RIS}^*| \quad , \tag{2.633}$$

where

$$R_{\boldsymbol{nn},RIS} = \begin{bmatrix} R_{\boldsymbol{nn}} & 0 \\ 0 & R_{\boldsymbol{nn},out} + Q_H \cdot R_{\boldsymbol{nn}in} \cdot Q_H^* \end{bmatrix} \quad . \tag{2.634}$$

The rate maximization problem, for a fixed $R_{\boldsymbol{xx}}$, is convex (concave), and the $\|Q_H\|_F^2 \leq G_H$ is similar to an energy constraint. The Q_H-dependent input autocorrelation matrix, which effectively is the transmitter to the RIS output stage, is

$$R_{x,in} \triangleq \begin{bmatrix} I \\ Q_H \cdot H_{in} \end{bmatrix} \cdot R_{\boldsymbol{xx}} \cdot [I \;\; H_{in}^* \cdot Q_H^*] \quad . \tag{2.635}$$

Equation (2.633), or the determinant within, is maximized over Q_H for a fixed $R_{\boldsymbol{xx}}$. Because of the noise dependency on Q_H, similar to crosstalk dependency on $R_{\boldsymbol{xx}}$ in the MAC, this maximization can occur through a modified iterative water-filling process. The Q_H-related energy component in each dimension must satisfy a water-fill criterion with all other dimensional contributions appearing as noise additions. Such a solution when complete will not decrease the value the mutual information (sum rate) for the given $R_{\boldsymbol{xx}}$.

The solution algorithm progresses and then views the current Q_H as constant while maximizing over $R_{\boldsymbol{xx}}$ subject to the convex constraint $trace\{R_{\boldsymbol{xx}}\} \leq \mathcal{E}_{\boldsymbol{x}}$, which is a simple water-fill problem. Since the same $\mathcal{I}$ value increases at each step, a global maximum will eventually occur if the above super-iteration of (1) iterative water-fill for fixed (last) $R_{\boldsymbol{xx}}$ on Q_H and (2) water-fill with fixed (last) Q_H for $R_{\boldsymbol{xx}}$ A program to implement this awaits an enterprising student's (extra credit) effort, both the dual water-fill and the diagonal case. The various Q_H constraints are all convex and simply will enter the first step optimization.

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 $d_{min} = 2$. (2 pts)

b. Find this translated 16 HEX's 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? Hint: $\log_2(7) \approx 14/5$. (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

$$\boldsymbol{g}_1 = \frac{d}{\sqrt{2}} \cdot [1\ 1\ 0] \tag{2.636}$$

$$\boldsymbol{g}_2 = \frac{d}{\sqrt{2}} \cdot [0\ 1\ 1] \tag{2.637}$$

$$\boldsymbol{g}_3 = \frac{d}{\sqrt{2}} \cdot [1\ 0\ 1] \tag{2.638}$$

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 2.83 shows an 8PAM constellation with "Gray" bit encoding[115] for transmission over an AWGN with noise σ^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.

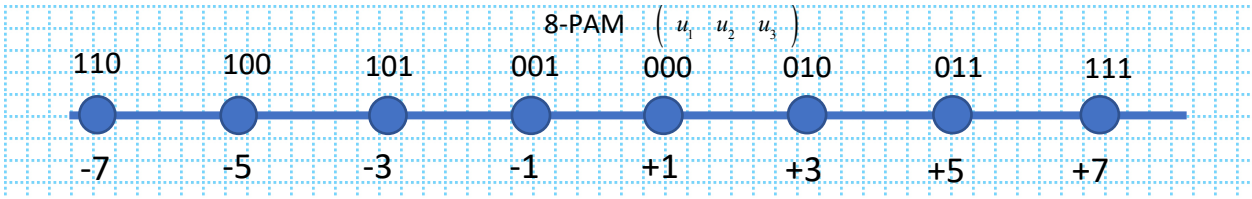


Figure 2.83: 8 PAM for bit MAP decoding.

a. Show that (with $x(u_j)$ being the 8-PAM value corresponding to u_j having a particular value, 1 or 0)

$$p_{u_i,y} \propto \sum_{u_j \forall j \neq i} e^{-\frac{(y-x(u_j))^2}{2\sigma^2}} \quad \text{(2 pts)}.$$

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_i is approximated for reasonable noises by

$$LLR_i \approx \frac{1}{2\sigma^2} \cdot [y - x^*(i = 0)]^2 - \frac{1}{2\sigma^2} \cdot [y - x^*(i = 1)]^2 \quad . \text{ (2 pts)}$$

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 $\mathbb{Z}^2$, 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 $\mathbb{Z}^2$, 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 $D_2 = G_{D_2} \cdot \mathbb{Z}^2$, which means the matrix operates on any ordered pair vector of integer inputs to generate corresponding outputs as elements of D_2.

[115] After Frank Gray, who is credited for designing constellations in which adjacent points differ in only one bit in the bit-level encoding.

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)

b. A coset adds the constant vector [0 1] to each point of D_2 to form $D_2 + [0\ 1]$. Identify the union $D_2 \bigcup D_2 + [0\ 1]$. (1 pt).

c. Following Part b, D_2 partitions $\mathbb{Z}^2$ 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 $V(\mathbb{Z}^2)$? Validate your answers by comparing with Part c (2 pts)

e. Suppose instead of viewing $\mathbb{Z}^2$ 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 $\mathbb{Z}^4 \triangleq \mathbb{Z}^2 \otimes \mathbb{Z}^2$ to form 4-dimensional integer points (and $\mathbb{Z}^2 = \mathbb{Z} \otimes \mathbb{Z}$). 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, $\bigcup$ 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 $\mathbb{Z}^4$? What is $V(D_4)$? (2 pts)

i. Find the determinant $|G_{D_4}|$. (1 pt)

j. Find d_{min} and γ_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 $\boldsymbol{y}$, how many real calculations would the straightforward exhaustive ML decoder have to perform? A real calculation would be and 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. 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. (HInt: A sketch may be the 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 a DMC 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. This problem's coded systems must 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)

Initially, the 16SQ QAM system's symbol-error probability P_e will lead to a $\bar{P}_b$ for a BSC - the bits are gray-coded to 16SQ QAM symbol values. The outer coded system uses one parity bit of redundancy per QAM subsymbol that is the modulo-2 sum of the other 3 bits, mapping then the extra 8 redundant points into 16SQ selections that effectively enlarge the constellation-choices possibilities to be among those of 16SQ QAM. These are hard-decoded into 4 bits the binary code's decoder treats those bits as a BSC. The QAM constellation symbol rate is 1 MHz.

b. What are N, $\widetilde{N}$ and $\overline{N}$ for the uncoded 16QAM system that would have $b = 4$? What are they for the coded system viewing each bit as a subsymbol? (2 pts)

c. For the bit-level coded system what is the redundancy ρ per symbol, and $\bar{\rho}$ per subsymbol? (2 pts)

d. What is d_{free} for the coded system? (2 pts)

e. Using the correct ML decoders for each respective system (AWGN uncoded and BSC coded), 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, creating a symmetric DMS with the $\bar{P}_b = p$ of Part a. An MDS code is applied as an outer code. There are now 8 input bits per each use of this inner system. This MDS code (a byte-wide Reed Solomon for the expert reader) has $\overline{N} \leq 254$ total bytes per codeword. The code uses $P \leq 16$ parity bytes from the $\overline{N}$ for implementation-complexity reasons. This problem will use only even-number values for parity and block length.

f. What are $\widetilde{N}$ and $\overline{N}$ for the uncoded inner system of 2 successive 16QAM symbols? (1 pt)

g. In terms of P and $\overline{N}$, what is the redundancy of the coded system in bits per codeword? And, in terms of bits per double-16QAM, or equivalently, byte $\tilde{\rho}$ (1 pt)

h. 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 on the SDMC? (1 pt)

i. 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:

```
% ------------------------------------------------------------
% [outvec] = choosevec(N,vec);
%
% Computes a vector of values with size on input vec (1 x something)
%   N   = top of choose
%   vec  = vector of values for choose
%
% created by J. Cioffi, 12/26/18
% ------------------------------------------------------------
function [outvec] = choosevec(N,vec);
% ------------------------------------------------------------
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.

j. What is the minimum number of parity bytes that can be used to ensure that the coded system has better data rate and lower error probability than the original 8SQ uncoded system? The answer may approximate error probability by the first term corresponding expressions. Also, find the corresponding N value. (3 pts)

k. What is the maximum data rate that could be achieved with this code (MDS $q = 2^8$)? (2pts)

2.6 *Bounds' Tightness for the DMC - Subsection 2.2.2 - 17 pts*

This problem explores the Singleton and Hamming bounds on 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 $d_{free} = 5$. What would be this bound's code rate it 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 Singleton Bound's tightness in Part a in view of the (correct) answer for Part b ? (1 pt)

d. Repeat Parts a - c for $n = 12$ with both $d_{free} = 1$ and then again for $d_{free} = 12$ and for 2.(2 pts).

e. A well-known binary Hamming code has $n = 7$, $d_{free} = 3$ 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_{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_{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_{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 symbol-error probability? What might be this approach"s drawbacks? (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 (2.64) 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);
% ---------------------------------------------------------------

size = length(idx);
temp = [];
temp2=zeros(1,size);
for in=1:size
temp = [temp, nchoosek(N,dh+idx(in))];
ln=idx(in);
temp2(in)=0;
minusone=1;
for jn=0:ln
    temp2(in)=temp2(in)+minusone*nchoosek(dh+ln,jn)*(q^(ln+1-jn)-1);
    minusone=minusone*-1;
end
end
Ni=temp.*temp2;
for in=1:size
if Ni(in) <0
    Ni(in)=0;
    in
    'MDS code does not exist for these parameters'
end
end
```

a. 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 .01. What is d_{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 codeword-error probability equation. (2 pts)

d. A How do these terms compare, and does a single term well represent the overall error probability if $p = 10^{-5}$? (1 pt)

e. code with $q = 16$, $N = 14$, $P = 4$ is instead used. Now find the 5 largest contributions and comment similarly thereupon, as in Part d.

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 problem. However, quantitatively predict your expectation as N grows, and as P grows for any particular N. (3 pts).

2.8 *Bandwidth Expansion (spread-spectrum effect), Hamming Codes, and Bit-Interleaved Coded Modulation - Subsection 2.2.2 - 18 pts*

This problem first exploits the AWGN's essentially infinite bandwidth by noting that power $P = \mathcal{E}/T$ decreases linearly with increased symbol rate at fixed symbol energy $\mathcal{E}$, but power increases exponentially with integer numbers of bits $\bar{b} \in Z$ 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 use any symbol rate that meets the performance objectives.

a. Is 128SQ a good design in terms of use of fixing any two quanties of the set {power P, data rate R, 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 $\bar{b} < 1$ 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 $\boldsymbol{v} = \boldsymbol{u} \cdot G$. 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 BSPK 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 $\bar{\mathcal{E}}_{\boldsymbol{x}}/T$ 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 $\frac{\mathcal{N}_0}{2}$ 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 i 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 $\mathcal{E}/b$. Hint: use the capacity formula for this. (2 pts)

Now suppose bandwidth is limited and the output bits $\boldsymbol{v}$ 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)

m. "**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 i (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, ARQ - Subsection 2.2.3 - 14 pts*

For this problem, refer to Figures 2.9 and 2.12, 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 $LLR(y)$ 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 **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 **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 and 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 block transmission 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_{tput}. (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 Δ. Determine the minimum Δ 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 this contellation's 4-dimensional entropy $\mathcal{H}_{\boldsymbol{x}}$? Also, then what are $\widetilde{\mathcal{H}}_{\boldsymbol{x}}$ and $\overline{\mathcal{H}}_{\boldsymbol{x}}$ (1 pt)

d. What is this constellations 2-dimensional entropy $\mathcal{H}_{\tilde{\boldsymbol{x}}}$? Also, then what is $\overline{H}_{\tilde{\boldsymbol{x}}}$ What are the 2D and 1D redundancies $\tilde{\rho}$ and $\bar{\rho}$. (3 pts)

e. What is this constellation's 1-dimensional entropy $\mathcal{H}_x$. How dowa thia compare to the entropy? (2 pts)

f. Compare the answers in Parts c to e to the corresponding maximum entropy in those dimensions. (2 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 continous entropy difference. (1 pt)

b. A continuous random variable g with the exponential density $p_g(u) = \frac{e^{-u/\mathcal{E}_g}}{\mathcal{E}_g}$ $\forall$ $g \geq 0$. What happens as $\mathcal{E}_g$ gets small? (2 pts)

c. A continuous random variable a with Rayleigh distribution $p_a(u) = \frac{u}{\mathcal{E}_a} \cdot e^{-\frac{u^2}{2\mathcal{E}_a}}$ $\forall a \geq 0$. Hint: Use that the definite integral $\int_0^\infty e^{-x} \cdot \ln(x) dx = -\gamma = -.577216$ (The Euler Macheroni constant). (3 pts)

d. A continuous random variable x with lognormal distribution $p_x(u) = \frac{1}{\sigma \cdot u \cdot \sqrt{2\pi}} \cdot e^{-\frac{(\ln(u)-\mu_x)^2}{2\sigma^2}}$ $\forall$ 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 2.84 has an AWGN channel for which a shaping lattice Λ_s is used with a good fundamental-gain code on its codewords with $\widetilde{N}$-dimensional subsymbols $\tilde{\boldsymbol{x}}$. The $\boldsymbol{x}$ 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 $\tilde{\boldsymbol{x}} \in C$ where C is a discrete constellation with $|C|$ points selected from a lattice Λ. This problem focuses on an approach to shaping codes that uses the shaping lattice Λ_s's Voronoi Region $\mathcal{V}(\Lambda_s)$ as the subsymbol constellation C's outer Voronoi boundary $\mathcal{V}_{\boldsymbol{x}}$. This means that the union of the $|C|$ constellation points' Voronoi Regions have Voronoi Boundary equal to the Voronoi region of Λ_s

$$\bigcup_{i=0}^{|C|-1} \mathcal{V}_i(\Lambda) = \mathcal{V}(\Lambda_s) \quad .$$

(For this to happen, a sufficient condition is that Λ_s be a sub-lattice of Λ.)

The "dither" sequence $\boldsymbol{\delta}$ is fixed and known to transmitter and receiver, but in any given subsymbol $\tilde{\boldsymbol{\delta}}$ it was generated by randomly selecting the subsymbol from a continuous distribution with the Voronoi Boundary region $\mathcal{V}_{\boldsymbol{x}} = \mathcal{V}(\Lambda_s)$. Such a sequence is sometimes known in cryptography as a **key**; this sequence's generation may be complicated depending on the region; $\boldsymbol{\delta}$'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 $\Lambda_s = Z^{\widetilde{N}}$, then $V(Z^{\widetilde{N}}) = 1$ and $\bar{\mathcal{E}}_{\Lambda_s} = 1/12$.

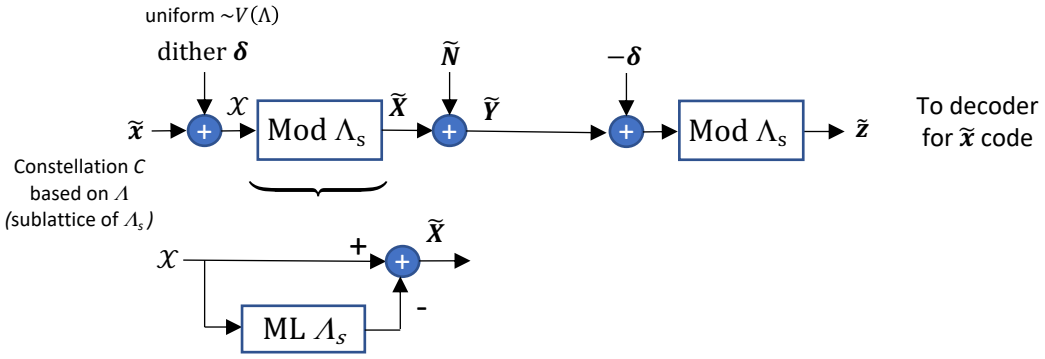


Figure 2.84: Dither Shaper.

As in Figure 2.84's lower portion, the modulo-Λ_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 Λ_s to that dithered signal $\mathcal{X}$ and then finds the difference $\tilde{\boldsymbol{X}}$ between the ML-detector's choice and the input. The modulo-Λ_s (sometimes called a **Voronoi Shaping Code Encoder**. Clearly $\tilde{\boldsymbol{X}} \in \mathcal{V}(\Lambda_s)$. $\tilde{\boldsymbol{X}}$ has mean value equal to the origin (all zeros). The modulo of any point $\tilde{\boldsymbol{z}}$ is abbreviated $(\boldsymbol{z})_{\Lambda_s}$. Application of modulo to the sum of two values (real or complex) is abbreviated $\text{mod}_{\Lambda_s}(\boldsymbol{z}_1 + \boldsymbol{z}_2) = \boldsymbol{z}_1 \oplus_{\Lambda_s} \boldsymbol{z}_2$.

a. Show that $\boldsymbol{z}_1 \oplus_{\Lambda_s} \boldsymbol{z}_2 = (\boldsymbol{z}_1)_{\Lambda_s} \oplus_{\Lambda_s} (\boldsymbol{z}_2)_{\Lambda_s}$. (2 pts)

b. Show that the transmitted sequence $\tilde{\boldsymbol{X}}$ and the input subsymbol sequence $\tilde{\boldsymbol{x}}$ are independent. (This independence essentially renders the message unrecoverable by any observer of the link, this securing transmission from eavesdropping[116].) (1 pt)

c. Show that the transmitted sequence $\tilde{\boldsymbol{X}}$ 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-Λ_s device. Such a system is also known as **writing on dirty paper** in information theory because the sequence $\tilde{\boldsymbol{x}}$ is written on top of "dirt" or existing writing that is the dither sequence. (See also Section 2.8.)

e. If the noise has $\sigma^2 = 0$, show that $\boldsymbol{z} = \tilde{\boldsymbol{x}}$ 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)

$$\tilde{I}_{\boldsymbol{x},\boldsymbol{y}} \leq \tilde{\mathcal{C}} - \log_2 \left(2\pi \cdot e \cdot \frac{\bar{\mathcal{E}}_x}{|V^{2/N}(\Lambda_s)|} \right) \quad .$$

g. Evaluate your bound for an AWGN with known SNR when $\Lambda_s = Z^{\tilde{N}}$ if $\boldsymbol{x}$ is a very good code. (1 pt).

[116] Of course no dither sequence or key will ever be perfect and so this method will not absolutely protect transmission in practice.

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 mean for a finite-field channel, particularly looking at the AEP's interpretation in terms of the Hamming and Singleton Bounds.

a. Use differential entropy $\mathscr{H}_{\tilde{\boldsymbol{x}}}$ to relate an AWGN's hypersphere squared radius (average 2D energy) $\mathcal{E}_{\tilde{\boldsymbol{x}}}$ to the size of the size of the typical set $|A^{\epsilon}_{\boldsymbol{x}}|$ for large n. (1 pt)

b. Using Part a's finding, similarly interpret the size of the conditional typical set$|A^{\epsilon}_{\boldsymbol{x}/\boldsymbol{y}}|$. (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)

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 similarly 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 *Bandwidth versus Power- 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. The power $\mathcal{E}_x/T$ is fixed, so if the symbol rate is increased, the energy per symbol must decrease by the same factor.

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 $\boldsymbol{x}$ with transmitted flat two-sided power spectral density $\bar{\mathcal{E}}_{\boldsymbol{x}} = -60$ dBm/Hz and white-noise power spectral density $\frac{\mathcal{N}_0}{2} = -140$ dBm/Hz. (dBm/Hz 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 $\bar{\mathcal{E}}_x$ 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 $H(f) = 10^{-5.3}\ \forall\ f$.

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 *Gaps and Codes - 14 pts*

(Section 2.2) An uncoded 2-level PAM modulator is used on an AWGN channel. The matlab programs qfunc.m and qfuncinv.m may be useful in this problem.

a. Find an expression for the symbol-error probability per dimension , $\bar{P}_e$, as a function of SNR. Does the bit error probability differ from this expression. (2 pts)

b. At what SNRs will the $\bar{P}_e = 10^{-6}$? $= 10^{-12}$? What is the difference in energy/power needed? (2 pts)

c. Find an expression for the gap Γ for this 2PAM in terms of SNR and $\bar{b}_{2-PAM}$ and evaluate it for both Part b's error probabilities. Find and explain the difference in gaps. (3 pts)

A byte-level MDS outer code ($\mathbb{GF}^{256}$) with is instead applied with individual (hard) bit decisions made AWGN channel with the SNR such that $p = \bar{P}_e = 10^{-6}$ for a BSC. The symbol rate for the 2-PAM inputs may be increased to offset any code rate loss. A single byte is then approximately in error with probability roughly $8p$.

d. Find an expression for the loss in SNR caused by increasing the 2-PAM symbol rate from $1/T$ to $1/T_{new} > 1/T$. (1 pt)

e. Find an expression for small $\lceil p$ for the new error probability in terms of N_e and d_{free}. (1 pt)

f. Evaluate this expression from Part e with $\bar{P}_b \approx \frac{\bar{P}_e}{8} < 10^{-12}$ to find a d_{free} value for the code that achieves the desired lower bit-error probability. (2 pts).

g. How many parity bytes are necessary to attain this d_{free}? What is the consequent code rate $R = K/N$? (2 pts)

h. Use your answer in Part b to find the loss (in dB) of the necessary rate increase. Compare this to the loss implied instead in Part c (1 pt)

2.17 *Channel Capacity for the BSC and the BEC - 12 pts*

Subsection 2.3.2 discusses the BSC and BEC.

a. Graph the BSC and BEC capacities 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 pt)

d. Find the AWGN channel capacity with SNR of 10 dB. (1pt)

e. Find P_b for binary PAM transmission on the channel of Part d. (2pts)

f. Letting p for a BSC be equal to Part e's, find the BSC's capacity. (2 pts)

g. Compare Part f's capacity with Part d's capacity. Why are they different? (2 pts)

2.18 *Universal DMC (Subsection 2.4.2) - 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

$$p(i/j) = \begin{cases} \frac{p_s}{255} & \forall i \neq j \\ 1 - p_s & i = j \end{cases} \tag{2.639}$$

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 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

$$R_{x,y} = \begin{bmatrix} \mathcal{E}_x & R_{xy} \\ R_{xy}^* & \mathcal{E}_y \end{bmatrix} \tag{2.640}$$

The minimum mean-square estimate of x given y has variance $\sigma^2_{x/y}$. The orthogonality principle of Appendix D may be useful throughout.

a. (1 pt) Find $\sigma^2_{x/y}$ in terms of $\mathcal{E}_x$, R_{xy}, and $\mathcal{E}_y$.

b. (1 pt) Relate the conditional entropy $H_{x/y}$ to $\sigma^2_{x/y}$ and therefore to the MMSE estimate.

c. (2 pts) Interchange the roles of x and y in the results in Parts **??** and b and compare the SNRs for the two results.

d. (1 pt) Relate the mutual information to Part c's SNR.

e. (3 pts) Suppose y becomes a complex vector $\boldsymbol{y}$ but x remains a scalar with $\boldsymbol{y} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} x + \boldsymbol{n}$ where $\boldsymbol{n}$ has independent zero-mean Gaussian components all with variance σ^2. Does Part d's answer change?

2.20 *MMSE and Entropy (Appendix D) - 14 pts*

A matrix AWGN has (real baseband or simply real in this case)

$$H = \begin{bmatrix} 10 & 4 \\ 3 & 1 \end{bmatrix} \quad . \tag{2.641}$$

The input vector is $\boldsymbol{x}$ and the output vector is $\boldsymbol{y}$, while the noise is $\boldsymbol{n}$, with $R_{\boldsymbol{nn}} = I$ and $R_{\boldsymbol{xx}} = I$. 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 $\boldsymbol{x}$ given the channel output $\boldsymbol{y}$. (1 pt)

c. Find the MMSE estimator for $\boldsymbol{y}$ given the channel input $\boldsymbol{x}$. (1 pt)

d. What type of decoder satisfies (1 pt)

$$\hat{\boldsymbol{x}} = \arg\min_{\hat{\boldsymbol{x}}} \left\{ \sum_{n=-\infty}^{\infty} \|\boldsymbol{y} - H \cdot \hat{\boldsymbol{x}}\|^2 \right\} \ ?$$

e. What type of estimator satisfies (1pt)

$$\hat{\boldsymbol{x}} = \arg\min_{\hat{\boldsymbol{x}}} \left\{ \sum_{n=-\infty}^{\infty} \|\hat{\boldsymbol{x}} - W \cdot \boldsymbol{y}\|^2 \right\} \ ?$$

In the asymptotic (law of large numbers or AEP) sense, what type of detector is this?

f. Compute $\widetilde{\mathscr{H}}_{\boldsymbol{x}/\boldsymbol{y}}$ and $\widetilde{\mathscr{H}}_{\boldsymbol{y}/\boldsymbol{x}}$. Compare them. (2 pts)

g. Suppose that Part b's W multiplies Part c's $\hat{\boldsymbol{y}}$, instead of $\boldsymbol{y}$. What does $W \cdot \hat{\boldsymbol{y}}$ estimate? Similarly, what does $H \cdot \hat{\boldsymbol{x}}$ estimate? What does this mean intuitively? (3 pts)

h. Reflecting on Part g's answer, 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 Multiuser Channel Types Subsection 2.6.1 - 8 pts

This problem provides several multi-user channels for type identification (e.g., MAC, BC, ...)

a. Identify Figure 2.85'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 stands for **optical line terminal** and typically at a network edge owned by the internet service provider, while the ONT stands for **optical network terminal** and is at the edge owned by the internet-service consumer.

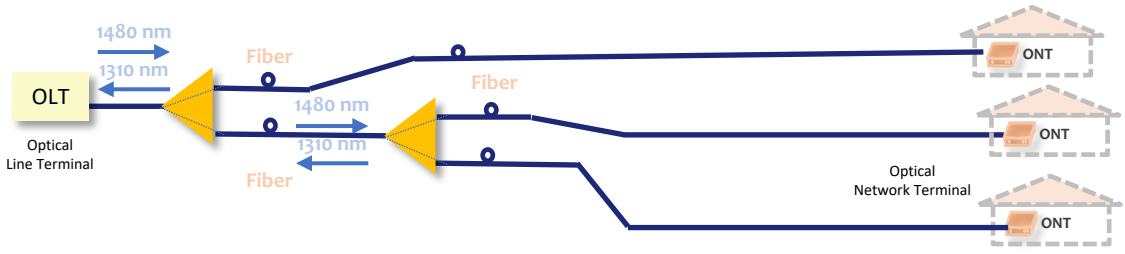


Figure 2.85: Channel type for Part a.

b. Identify Figure 2.86's channel type when transmitting in one direction only, and when transmitting in two direction simultaneously. What is U in each case? (3 pts)

Hint: 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. The two directions do not otherwise cooperate.

Figure 2.86: Channel type for Part b.

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).

2.22 MU Detector **Multiuser Margin**- 10 pts

Figure 2.87 provides a capacity region for a 2-user scalar-AWGN complex-baseband MAC. Both users may use up to 1 energy unit (each) per subsymbol.

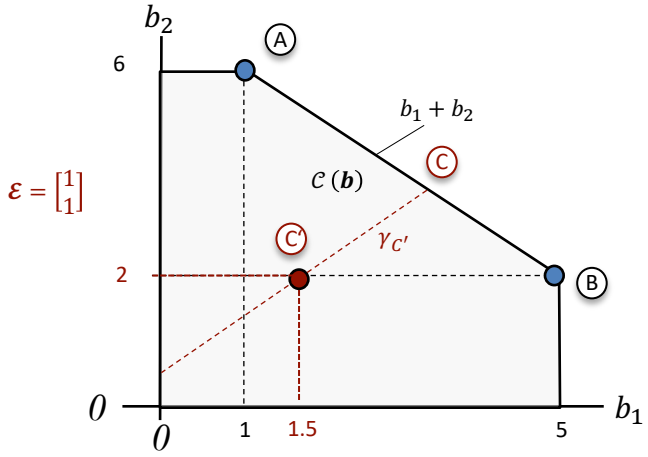


Figure 2.87: Capacity region for Problem 2.22.

a. What is the maximum rate-sum for this channel (2 pts)

b. Find the two users' SNRs in the absence of crosstalk, SNR_1 and SNR_2. (2 pts).

c. Find the two data-rate vector points for (C) and (C') (2 pts).

d. Find the corresponding multiuser margin, both $\gamma_{(C')}$ and γ_m. (2 pts)

e. Could this point be achieved with uncoded transmission at $P_e = 10^{-6}$? If not, how far away is the point from being achieved? If not, what is minimum coding gain required? (2 pts)

2.23 Mutual-Information Vector Subsection cp6.4 - 6 pts

This problem determines the minimum information vector for a multi-user channel for the orders and information-like

rcvr/ user	π_5	π_4	π_3	π_2	π_1
top	5	3	3	5	5
	4	4	5	2	4
	3	5	4	4	1
	2	1	2	3	3
botto m	1	2	1	1	2
$\mathbb{P}_u(\boldsymbol{\pi}_u)$					

rcvr/ User $\mathfrak{I}$	π_5	π_4	π_3	π_2	π_1
top	10	∞	5	∞	∞
	8	7	8	4	∞
	5	8	5	6	3
	3	2	3	4	3
bottom	2	2	2	2	1

Table 2.9: Tables for Problem 2.23.

a. How many users are there? (1 pt)

b. Find the values of $\mathbb{P}_u(\boldsymbol{\pi}_u)$ for each of the users. (2.5 pts)

c. Find the minimum information vector for this channel. (2.5 pts).

2.24 Time-Division Multiplexed Multiuser Channel Subsection 2.7.2 - 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 $\mathcal{E}_{\boldsymbol{x}}$, with noise power spectral density σ^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 may faces has $\mathcal{C}(\boldsymbol{b})$? (2 pts).

d. How many vertices has $\mathcal{C}(\boldsymbol{b})$ (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 periods equal for all users. What happens to the capacity region? (2 pts)

2.25 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

$$R_{\boldsymbol{nn}} = \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{-1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix} \cdot \begin{bmatrix} \frac{9}{256} & 0 \\ 0 & \frac{1}{16} \end{bmatrix} \cdot \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix} .$$

Input energy follows from

$$R_{\boldsymbol{xx}} = 2 \cdot I \ .$$

Hint: Parts e and f may solve easily using Cholesky factorization of R_b^{-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 $R_{\boldsymbol{nn}}$, 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. (1 pt)

d. Find the largest rate sum for the given input $R_{\boldsymbol{xx}}^{1/2}$. (1 pt).

e. Find the 3 two-dimensional pentagon regions that correspond to $\mathcal{E}_3 = 0$, $\mathcal{E}_2 = 0$, and $\mathcal{E}_1 = 0$ respectively. (3 pts)

f. Find the 6 vertices when $\mathcal{E}_u = 2$, $\forall u = 1, 2, 3$ that characterize the maximum rate-sum plane boundary of $\mathcal{A}(\boldsymbol{b}, R_{\boldsymbol{xx}})$. (6 pts)

g. Is the point $\boldsymbol{b}^* = [b_3 \ b_2 \ b_1] = [0.1 \ 2.4 \ 3.2]$ in the achievable region? (2 pts)

h. Relax the energy constraint to $trace\{R_{\boldsymbol{xx}}\} \leq 6$ and find the maximum rate sum. (1 pt)

i. For the relaxed energy constraint of the energy-sum MAC in Part h, find $\boldsymbol{u}^o$ and $\boldsymbol{u}^s$ (assuming the conventional user indexing of highest number MAC user at top/left and the requisite energy-sum MAC has the same energy as the given $R_{\boldsymbol{xx}}$). Find possible user orders that would achieve maximum rate-sum on a energy-sum MAC version of this channel. (3 pts).

2.26 Pre-Triangular MAC - 12 pts

A 2-user real baseband MAC has channel

$$H = \begin{bmatrix} 1 & 0.9 \\ 0 & 1 \end{bmatrix}$$

with Gaussian noise autocorrelation $R_{\boldsymbol{nn}} = 0.1 \cdot I$. Each independent user has unit energy. The gap is zero dB.

a. Is there a single preferred order for this triangular channel? (1 pt).

b. Try two zero forcing designs for the two possible designs that simply detect one user from the output with the other as noise, subtract that other user, and then detect the remaining user. Is this optimum design when the original channel is triangular? (3 pts)

c. Find $\mathcal{C}(\boldsymbol{b})$ and the maximum rate sum, along with two designs corresponding to the vertices (4 pts)

d. Find a MMSE-based LINEAR (no feedback nor successive decoding allowed) receiver design for this MAC. Find also the loss with respect to the optimum (nonlinear design with feedback from Part c. (3 pts)

e. What does this imply for a designer (or designs) that in choosing linear versus zero-forcing (successive cancellation)? (1 pt)

2.27 Vestigially Symmetric Continuous MAC rate sum Subsection 2.7.4 - 12 pts

(See Section 2.5 end example.) Figure 2.88's 2-user continuous-time Gaussian MAC has two channel responses

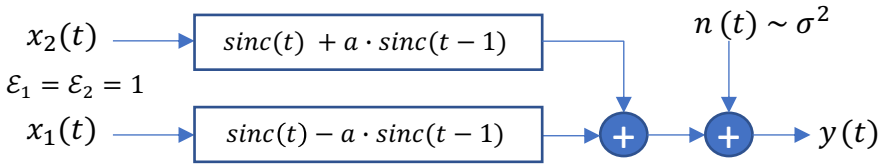


Figure 2.88: Simultaneous Water Filling for Problem 2.27.

$$\begin{aligned} h_2(t) &= \text{sinc}(t) + a \cdot \text{sinc}(t-1) \\ h_1(t) &= \text{sinc}(t) - a \cdot \text{sinc}(t-1) \end{aligned}$$

with AWGN with (two-sided) power spectral density σ^2 and $0 \le a \le 1$, with power $P_1 = P_2 = 1$. The integral

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

may be useful.

a. Sketch the inverse frequency responses of the two users' channels $|H_2^{-1}(f)|$ and $|H_1^{-1}(f)|$ on the same graph with $a = 0.9$. (2pts)

b. Describe the frequency-division SWF (simultaneous water fill) solution's spectra in terms of a given water level λ for any value of a and σ^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 Subsection 2.8.1 - 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 $2\mathbb{Z}^2$.

a. Find the result of the following operations for $(\boldsymbol{v})_{4\mathbb{Z}^2}$: (4 pts)

(i) $\boldsymbol{v} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

(ii) $\boldsymbol{v} = \begin{bmatrix} 4.5 \\ 1 \end{bmatrix}$

(iii) $\boldsymbol{v} = \begin{bmatrix} 4.2 \\ -4.2 \end{bmatrix}$

(iv) $\boldsymbol{v} = \begin{bmatrix} -17 \\ -21 \end{bmatrix}$

This is the **rectangular lattice** (scaled by 4).

b. Using the generators for the **hexagonal lattice** $\Lambda = A2$, $\boldsymbol{g}_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $g_2 = \begin{bmatrix} \frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{bmatrix}$ so that any $\lambda \in A_2$ can be written as $\lambda = z_1 \cdot \boldsymbol{g}_1 + z_2 \cdot \boldsymbol{g}_2$ where $z_1 \in \mathbb{Z}$ and $z_2 \in \mathbb{Z}$, find $(\boldsymbol{v})_{A_2}$: (4 pts)

(i) $\boldsymbol{v} = \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix}$

(ii) $\boldsymbol{v} = \begin{bmatrix} 2.1 \\ 0 \end{bmatrix}$

(iii) $\boldsymbol{v} = \begin{bmatrix} 0 \\ -2.1 \end{bmatrix}$

(iv) $\boldsymbol{v} = \begin{bmatrix} \sqrt{3}/2 + \epsilon \\ 1/2 + \epsilon \end{bmatrix}$ with $0 < \epsilon < 0.5$.

c. The **face-centered cubic** $D3$ lattice has $\lambda \in \Lambda$ such that $\lambda = z_1 \cdot \boldsymbol{g}_1 + z_2 \cdot \boldsymbol{g}_2 + z_3 \boldsymbol{g}_3 = G \cdot \boldsymbol{z}$, with $\boldsymbol{z} \in \mathbb{Z}^3$ where: (4 pts)

$$G = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

(i) $\boldsymbol{v} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

(ii) $\boldsymbol{v} = \begin{bmatrix} 2.1 \\ 1 \\ 1 \end{bmatrix}$

(iii) $\boldsymbol{v} = \begin{bmatrix} -1 \\ -1.1 \\ -2.1 \end{bmatrix}$

(iv) $\boldsymbol{v} = \begin{bmatrix} 5 \\ 4 \\ 5 \end{bmatrix}$.

2.29 Scalar BC - Subsection 2.8.2 8 pts A scalar real Gaussian BC has

$$H = \begin{bmatrix} -2 \\ 1 \end{bmatrix}$$

with noise per dimension $\sigma^2 = .01$, and $\overline{\mathcal{E}}_x = 2 = \overline{\mathcal{E}}_1 + \overline{\mathcal{E}}_2$.

a. Show the two receivers for successive decoding. (1 pts)

b. Show a precoder and corresponding two receivers if each user has one unit of energy. (2 pts)

c. Find $\mathcal{C}_{BC}(\boldsymbol{b})$. (4 pts)

d. Find $\mathcal{C}_{MAC}$ for an energy-sum MAC with $H = [-2\ 1]$ with $\overline{\mathcal{E}}_x = 2$. (1 pt)

2.30 Vector Gaussian BC Subsection 2.8.2 - 12 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 complex dimension $2 \cdot \sigma^2 = .01$, and $\mathcal{E}_x = \mathcal{E}_1 + \mathcal{E}_2 + \mathcal{E}_3 = 3$. $L_y = 1$ for each receiver.

a. Find the noise-whitened channel, and determine if the channel is degraded. (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 $R_{\boldsymbol{xx}}$. 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. Design a precoder (using the new order) for $R_{\boldsymbol{xx}} = \text{Diag}(1\ 1\ 4)$. 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 $R_{\boldsymbol{xx}}$. (5 pts)

e. Design the decoder for the solution in Part d. (1 pts)

2.31 Duality Subsection 2.8.4 - 7 pts
A real baseband scalar BC has

$$H_{BC} = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}$$

with noise per complex dimension $\sigma^2 = 1$, and $\mathcal{E}_x = \mathcal{E}_1 + \mathcal{E}_2 + \mathcal{E}_3 = 3$. $L_y = 1$ for each receiver.

a. What is the $trace\{R_{\boldsymbol{xx}}\}$ limit for the MAC's dual BC channel? (1 pt)

b. Find H_{MAC} for the dual channel (1 pt)

c. Find a MAC vertex point that corresponds to $\boldsymbol{\mathcal{E}} = [1\ 1\ 1]^*$ and for which user 1 has the largest rate and user 3 has the lowest. Can you draw any conclusions for scalar channels from this result? (3 pts)

d. Find the corresponding dual BC's energies for the answer in Part c (1 ptRxx)

2.32 MU BC Design **Secondary User Determination** Subsection 2.8.4 - 9 pts
A real baseband scalar BC has

$$H_{BC} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

with noise per complex dimension $\sigma^2 = .01$, and $\mathcal{E}_x = 4$. $L_y = 1$ for each receiver.

a. Find the maximum sum rate for this BC and the corresponding best $R_{\boldsymbol{xx}}$ and worst-case noise R_{wcn} (4 pts)

b. Ensure your autocorrelation matrices have the appropriate rank (eliminate finite-precision errors leading to appearance of any nonsingularity) and find the any secondary and primary users. (1 pt)

c. Use the best $R_{\boldsymbol{xx}}$ found in Part b but allocate all energy to any secondary user. What is the new sum rate? (2 pts)

d. Suppose all energy were allocated to the best user, what would the data rate be (2 pts)

2.33 IC Vertices Section 2.9 - 7 pts

A 2-user real baseband IC has channel

$$H = \begin{bmatrix} 1 & 0.9 \\ 0 & 1 \end{bmatrix}$$

with Gaussian noise autocorrelation $R_{\boldsymbol{nn}} = I$. Each independent user has unit energy. The gap is zero dB.

a. Find $\boldsymbol{\mathcal{I}}_{min}(\boldsymbol{\pi}, \boldsymbol{\mathcal{E}})$ for the two orders and for both users at unit energy. (4 pts)

b. Find the achievable regions and the capacity region corresponding to the convex hull of the two solutions. Show this pentagon is the capacity region. (2 pts)

c. Find the capacity region $\mathcal{I}(\boldsymbol{b})$ and the maximum rate sum. (2 pts)

d. Describe the two receivers for the maximum rate-sum point. (2 pts)

e. Repeat Part d for any point that is on the upper right boundary that is $\alpha \in (01)$ fraction of the way from the vertex on the right to the vertex on the left. (2 pts)

2.34 Mixed-Dimensional IC Subsection 2.9.3 - 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 2 (corresponding to the lower 2 rows and rightmost columns for this IC is two-dimensional complex (4 real dimensions), while user 1 is one-dimensional complex (two real dimensions). The noise per dimension $\sigma^2 = .01$ and independent in all dimensions, and $\mathcal{E}_x = \mathcal{E}_1 + \mathcal{E}_2 = 2$, 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 $\mathcal{C}_{IC}(\boldsymbol{b})$ by finding points. Show the max rate-sum point and slope of tangent at that point (3 pts)

2.35 MAC Design (17 pts):

Figure 2.89 illustrates a real baseband matrix-AWGN MAC with $\mathcal{E}_1 = \mathcal{E}_2 = 1$ with fixed subsymbol rate $1/T = 1$..

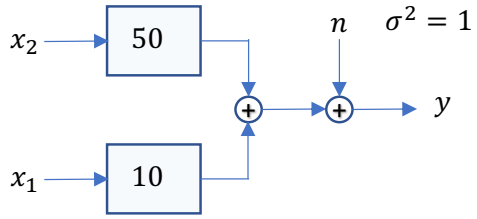


Figure 2.89: Multiuser MAC Design - Problem 2, 2023 Midterm

a. Find the capacity region. (3 pts)

b. Find the maximum rate-sum point that also satisfies $b_2 = \frac{1}{2} \cdot b_1$ (2 pts)

c. Describe a successive-decoding receiver for Part b (3 pts)

d. Is there a larger rate sum possible? If so, what is it? (2 pts)

e. Suppose a design must have at least 6 dB of single-user margin for both users with respect to the data rate found in Part b. What is that data rate while maintaining $b_1 = 2 \cdot b_2$, What happens to the data-rate ratio if both users have the a fixed multiuser margin of 6 dB? (3 pts)

Return to $\gamma_m = 0$ dB. The sampling rate and symbol rate cannot change.

f. If $\mathcal{E}_{sum} = 2$ is the only input energy restriction, what is the maximum sum-rate point with $b_1 = 2 \cdot b_2$. (3 pts)

g. Repeat part f for any input data rates. [Hint: this is easy, no fancy software is necessary] (1 pts)

2.36 Simple Relay Channel Section 2.11- 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 $\mathcal{E} = 1$ 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 users 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

[1] Claude E. Shannon *A Mathematical Theory of Communication* Bell Systems Technical Journal, 1948. Vol. 27, No. 3, pp. 628-634.. See also *Communication in the Presence of Noise* Proceedings of the IRE, 1949 Vol. 37, No. 1, pp. 10-21.

[2] Thomas M. Cover and Joy A. Thomas *"Elements of Information Theory"*. Wiley, New York, 1991.

[3] Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *"Linear Estimation"*. Prentice Hall, USA, 2000.

[4] G. David Forney, Jr. *"On the role of MMSE estimation in approaching the information-theoretic limits of linear Gaussian channels: Shannon meets Wiener"*. Proceedings Allerton Conference, pp.430-439 October 2003

[5] Abbas El Gamal and Young-Han Kim *"Network Information Theory"*. Cambridge University Press, UK 2011

[6] Sumanth Jagannathan *INTERFERENCE AND OUTAGE OPTIMIZATION IN MULTI-USER MULTI-CARRIER COMMUNICATION SYSTEMS* Ph.D. Dissertation, Stanford University. 2008

[7] Stephen B. Wicker *"Error Control Systems for Digital Communication and Storage"* Prentice Hall, New Jersey 1995

[8] N. McKeown, A. Mikkittikul, V. Anantharam, and J. Walrand *"Achieving 100% Throughput in an Input-Queued Switch"* Proceedings IEEE Infocom 1996, San Francisco, CA, pp. 296-302

[9] K. Seong, R. Narsimhan, and J.M. Cioffi *"Queue Proportional Scheduling in Gaussian Broadcast Channels"* Proceedings IEEE International Conference on Communications 2006, Isltanbul, TURKEY , pp. 1647-1652

[10] G. Caire and G. Taricco, and E. Biglieri *"Bit-Interleaved Coded Modulation"* IEEE Transactions on Information Theory May 1998, May , Vol 44, No. 3, pp. 927- 946

[11] Joel Gorham Smith *"The Information Capacity of Amplitude and Variance-Constrained Scalar Gaussian Channels"* Information and Control April 1971, Vol 18, No. 3, pp. 203 - 219

[12] Bernd Bandemer, Abbas El Gamal, and Young-Han Kim *"Optimal Achievable Rates for Interference Networks with Random Codes"* IEEE Transactions on Information Theoryl December 2015, Vol 61, No. 12, pp. 6536-6548

[13] Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *"Linear Estimation"*. Prentice Hall, USA, 2000.

Index

1

2

3

4

5

6

7

8

9

10

11