

Fundamentals of Coding and Modulation

Tutorial at ECOC 2009, Vienna, Austria

Gerhard Kramer

Department of Electrical Engineering - Systems
University of Southern California
Los Angeles, CA, USA

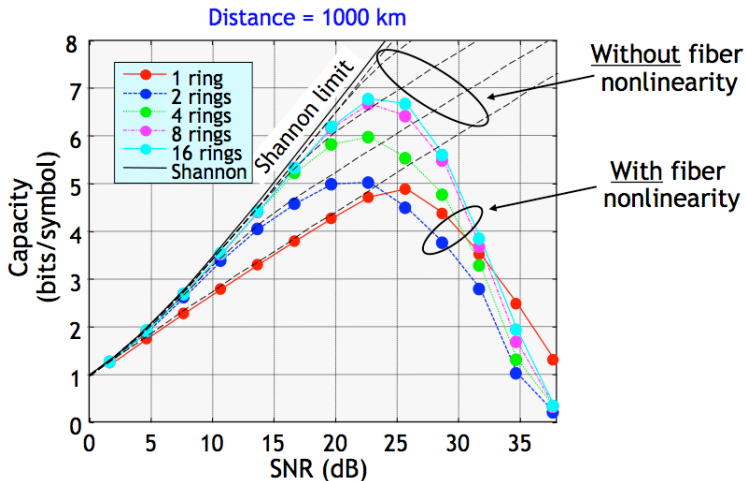
September 2009

Outline

- ① Part 1: Channels, Coding, and Capacity
- ② Part 2: Spectral Efficiency
- ③ Part 3: Linear Block Codes
- ④ Part 4: Hard and Soft Decoding
- ⑤ Part 5: Fiber Capacity Estimate

Acknowledgment: Several graphics were borrowed from R.-J. Essiambre and M. Magarini. USC students provided feedback on presentation.

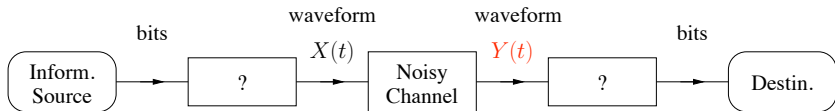
- One goal: review (**classic communications**) concepts that led to the following plot from our OFC 2009 tutorial.



Channels, Coding, and Capacity

1. Channels, Coding, and Capacity

Communication Problem



- Information source for our purposes: **bits**
- **Channel**: the part of a system one is **unable** or **unwilling** to change.
Idea: Everything else can be optimized. Example channels:
 - fiber (leaving many parameters, e.g., fiber type, filters, etc.)
 - specific fiber + filters + noncoherent detector
 - specific fiber + filters + specific modulator + coherent soft detector
- Goal: transmit information **quickly** and **reliably** to the destination
- How should we design the bits-to-waveform mapping?
How should we design the waveform-to-bits mapping?

Continuous to Discrete in Time and Amplitude/Phase



- Suppose the channel is **bandlimited** and passes the frequencies

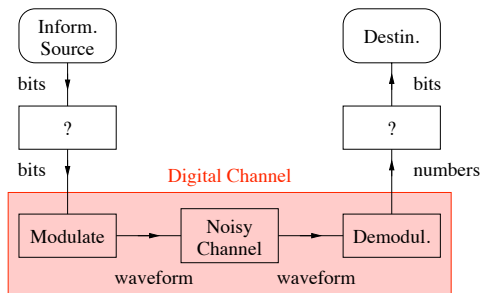
$$f_0 - W/2 < f < f_0 + W/2 \text{ Hertz}$$

where the center frequency f_0 is much larger than W

- Use Nyquist-Shannon sampling theory to represent signals by n regularly-spaced samples that are $T_s = 1/W$ seconds apart

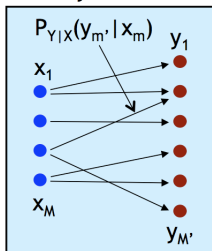
$$X(t) = \sum_{i=1}^n X_i^c \cdot \frac{\sin(\pi W t - \pi i)}{\pi W t - \pi i}$$

- So $X(t)$ and $Y(t)$ are represented by **discrete-time** and **continuous amplitude/phase** signals $X_1^c, X_2^c, \dots, X_n^c$ and $Y_1^c, Y_2^c, \dots, Y_n^c$
- We can “closely” approximate these signals with **discrete amplitude/phase** signals X_1, X_2, \dots, X_n and Y_1, Y_2, \dots, Y_n



- Result: an essentially **optimal** transmission structure is:
 - modulator: use **discrete levels** to approx. optimal input statistics
 - waveform: use a **compact spectrum** that approx. $\sin(x)/x$ signaling
 - demodulator:
 - Sample **signal** (field) in time, i.e., use **coherent detection**
 - Sample signal in amplitude/phase; use many sampling levels to avoid losing information, i.e., use **soft outputs**
- Summary: the mod/demod may as well convert the **continuous-time waveform** channel into a **digital** (discrete-time/alphabet) channel

Memoryless Channel



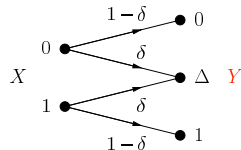
- **Noisy** channel: a **cond. probability distrib.** $P_{Y_1 Y_2 \dots Y_n | X_1 X_2 \dots X_n}(\cdot | \cdot)$. This model permits memory. We will process to remove memory.
- **Memoryless** channel: X represents **one** input and $Y = f(X, Z)$ where $f(\cdot)$ is some function and Z is **noise**, e.g., $Y = X + Z$
- Some common memoryless channels (see next page):
 - binary erasure channel (BEC)
 - binary symmetric channel (BSC)
 - additive white Gaussian noise channel (AWGN channel)

Binary Erasure Channel (BEC):

Example: Sudoku or crossword Puzzles

δ is erasure probability

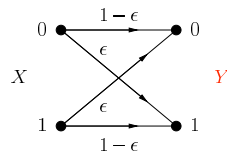
Δ is erasure symbol



Binary Symmetric Channel (BSC):

Example: OOK with hard detection

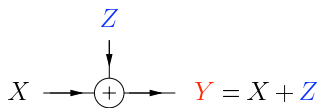
ϵ is crossover probability



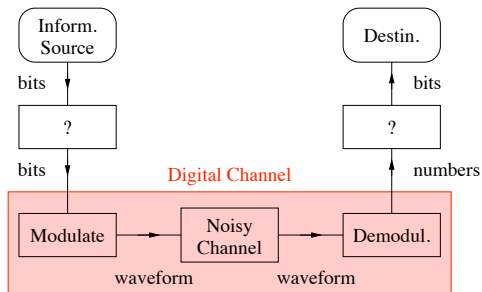
Additive white Gaussian noise (AWGN) channel:

complex input/output, Z is Gaussian with variance N

Gaussian Noise

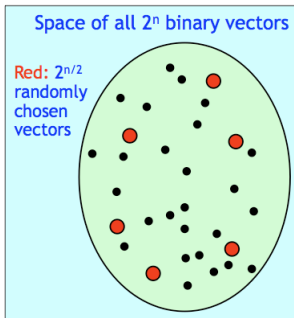


Coding

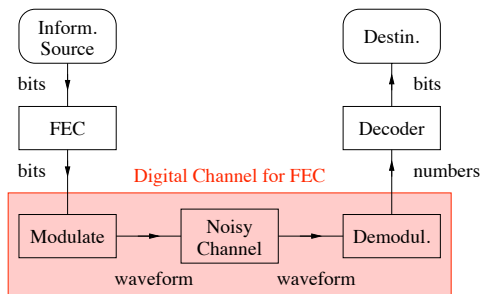


- How should we **use** the noisy digital channel $P_{Y|X}(\cdot|\cdot)$?
- Suppose **reliability** is paramount. Can we guarantee reliability?
- Simple answer: **yes** by **repeating** every symbol sufficiently often.
Sophisticated answer: **yes** and with positive rate (!) by using **coding**

- Example: suppose we use the BSC n times to transmit $k = n/2$ information bits (terminology: an (n, k) **code** with **rate** $k/n = 1/2$)
- There are 2^k information-bit vectors and 2^n length- n bit vectors. Map each information-bit vector to a unique length- n **code** vector.
- Table: the fraction of code vectors becomes **tiny** as n gets large
- Intuitive idea: the **“distance”** between code vectors becomes large, thereby ensuring reliability



n	2^n	$2^{n/2}$	$(2^{n/2})/2^n$
2	4	2	$1/2$
4	16	4	$1/4$
10	1024	32	$1/32$
100	2^{100}	2^{50}	2^{-50}



- How to choose 2^k code vectors? Shannon chose **random** vectors which is **essentially optimal** but **impractical** (huge look-up tables).
- **Code design** is the **art** of choosing **practical** codes, yet still achieve the maximum reliable rate called the **capacity**.
- Final communications block diagram is shown above, where FEC means “Forward Error Correction” or “Forward Error Control”

Capacity

- **Capacity** C is the maximum rate (bits/channel symbol) at which one can achieve arbitrarily small positive (non-zero) error probability
- **Shannon (1948)**: the capacity of the channel $P_{Y|X}(\cdot|\cdot)$ is



$$C = \max_{P_X} I(X; Y)$$

where

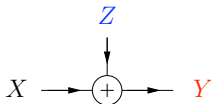
$$I(X; Y) = H(X) - \sum_y P(y) H(X|Y = y)$$

$H(X) = \sum_x -p(x) \log_2 p(x)$ is the **entropy** of X and $H(X|Y = y) = \sum_x -p(x|y) \log_2 p(x|y)$ is the entropy of X **conditioned** on $Y = y$.

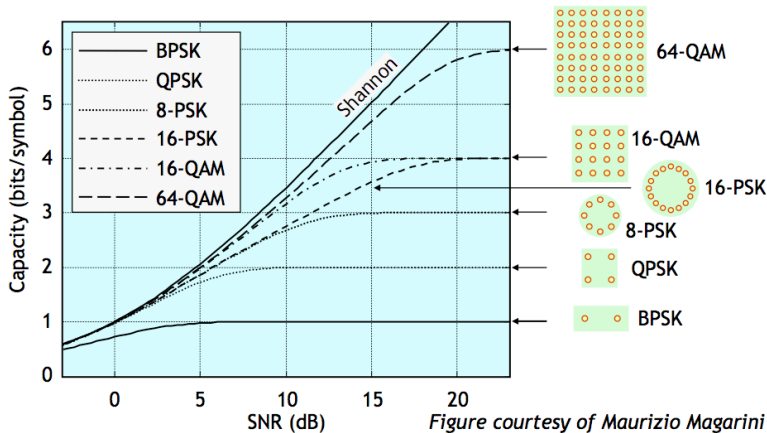
- So C is a **maximum entropy difference**.

- BEC: $C = 1 - \delta$ where δ is the erasure probability
- BSC: $C = 1 - h(\epsilon)$ where ϵ is the crossover probability and where $h(\epsilon) = -\epsilon \log_2(\epsilon) - (1 - \epsilon) \log_2(1 - \epsilon)$ is the **binary entropy function**

- **Complex** AWGN channel:

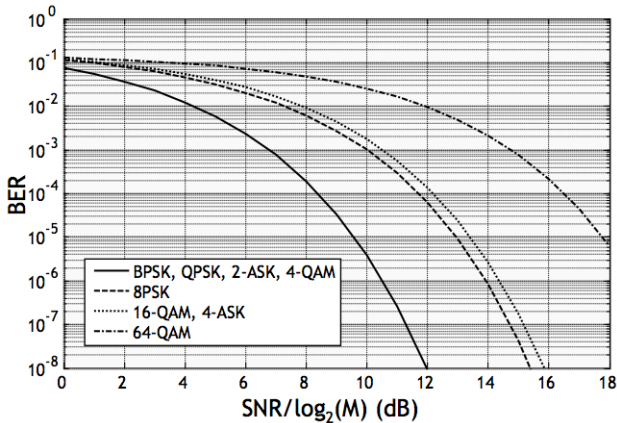


- Z is Gaussian, indep. real/imag. parts, each with variance $N/2$
- X has block power constraint $\sum_{i=1}^n |X_i|^2/n \leq P$
- Capacity: $C = \log_2(1 + \text{SNR})$ where $\text{SNR} = P/N$
Often use dB scale: $\text{SNR}(\text{dB}) = 10 \log_{10} \text{SNR}$
- Capacity achieved with **Gaussian** X with indep. real/imag. parts, each with zero mean and variance $P/2$
- Modulation: for practicality, use **discrete** X with M possible values to approximate a Gaussian random variable



- M-PSK=M-ary Phase-Shift Keying, B=Binary, Q=Quaternary, QAM=Quadrature Amplitude Modulation
- Each modulation has maximal capacity $\log_2(M)$
- Best mod. for complex AWGN channel is bi-dimensional Gaussian

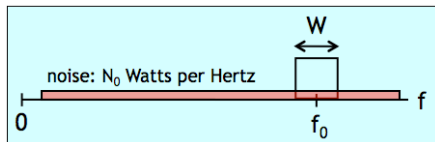
Comparison: Bit-Error-Rate (BER) Plots



- Observe: this plot does **not** give the right insight **with coding**. A large M is compensated by using a small code rate k/n .
- This and energy considerations suggest concept of **spectral efficiency**.

Spectral Efficiency

2. Spectral Efficiency



- Spectral efficiency: include effects of **energy** and **bandwidth**
- Noise power increases with bandwidth: $N = N_0 W$ where N_0 is the noise power per Hz
- Shannon capacity (now in **bits/s !!**) becomes

$$C \text{ [bits/s]} = W \log_2 \left(1 + \frac{P}{N_0 W} \right) \text{ [symbols/s]} \cdot \text{[bits/symbol]}$$

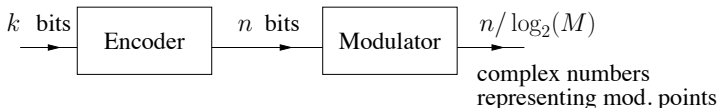
- Alternatively, in bits/symbol or **bits/s/Hz** we have

$$C/W = \log_2 \left(1 + \frac{P}{N_0 W} \right) \text{ [bits/s/Hz]}$$

Rate Definitions

Consider three rate definitions:

- 1 Code rate: $R_c = k/n$ where code maps k info. bits to n coded bits.
Note: the code **overhead** in percent is $100 \cdot (1/R_c - 1)$.
Example: $R_c = 15/16$ means the overhead is 6.67%.
- 2 Modulation rate: $R_m = \log_2(M)$ where M is the number of values X takes on. Example: 64-QAM has $R_m = 6$
- 3 Overall (coded modulation) rate: $R = R_c R_m = R_c \log_2(M)$



SNR Definitions

Consider three Signal-to-Noise Ratio (SNR) definitions:

- ① SNR defined as $\text{SNR} = (P/N)$
- ② SNR per (information) bit: recall that R is the overall rate
 - Energy per (information) bit is $E_b = (PT_s)/R$
 - Using $N = N_0W$ and $W = 1/T_s$ we have $\text{SNR} = (E_bR)/N_0$
so the SNR per bit is

$$\frac{E_b}{N_0} = \frac{\text{SNR}}{R}$$

- ③ OSNR (Optical SNR): let $B_{ref} = 12.5$ GHz and let p be the number of polarization states occupied by the signal. Now define

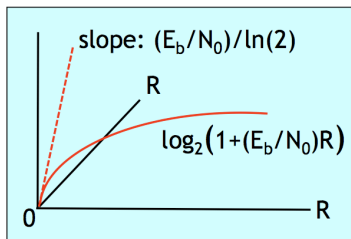
$$\text{OSNR} = \text{SNR} \cdot p / (2B_{ref}T_s)$$

Computing Spectral Efficiency

- AWGN channel: $R \leq C/W = \log_2(1 + \text{SNR})$ [bits/s/Hz]
- Substituting $\text{SNR} = (E_b/N_0)R$, we have

$$R \leq \log_2 \left(1 + \frac{E_b}{N_0} R \right)$$

- **Spectral efficiency** $\eta(E_b/N_0)$: **largest** solution for R of this equation
- Note: if $E_b/N_0 \leq \ln(2)$ then the only solution is $R = 0$



Modulation and Detection Spectral Efficiency

- Capacities for modulations (and detectors) generally have the form

$$R \leq C/W = \left[\max_{P_X} I(X; Y) \right] = f(\text{SNR})$$

where $f(x)$ is some non-decreasing function in x .

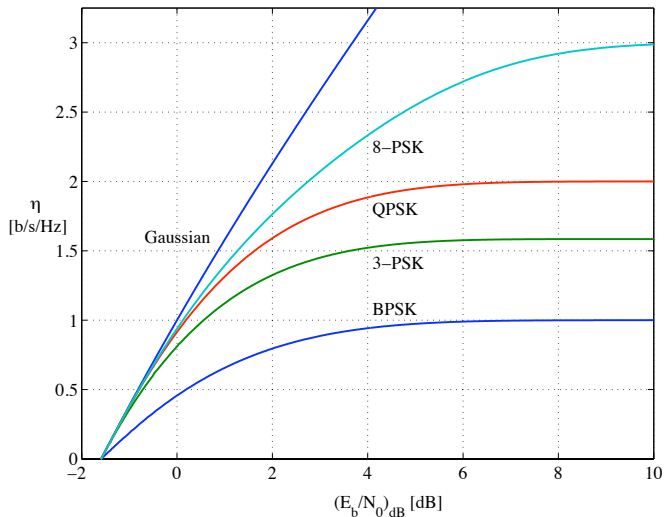
- Again substituting $\text{SNR} = (E_b/N_0) R$ we have

$$R \leq f\left(\frac{E_b}{N_0} R\right)$$

and $\eta(E_b/N_0)$ is the largest solution for R of this equation.

- The numerical results for M -PSK are shown below where $(E_b/N_0)_{dB} = 10 \log_{10}(E_b/N_0)$

Spectral Efficiency Plots

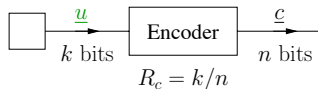


Spectral Efficiency Plots (Continued)

- Note: lowest possible $(E_b/N_0)_{dB}$ is $10 \log_{10}(\ln(2)) = -1.6$ dB
- For low E_b/N_0 and η , there is little transmission energy available per Hz of bandwidth. Typical applications: deep-space and ultra-wideband communication
- For high E_b/N_0 and η , should use modulations with large M . Note that $\eta \leq \log_2(M)$. Typical applications: DSL, wireless LAN, **optical fiber**

Linear Block Codes

3. Linear Block Codes



- Block encoding: map k information bits $\underline{u} = [u_1, u_2, \dots, u_k]$ into n coded bits $\underline{c} = [c_1, c_2, \dots, c_n]$
- Example: repetition code with $k = 1$, $n = 3$

$$0 \rightarrow [0 \ 0 \ 0]$$

$$1 \rightarrow [1 \ 1 \ 1]$$

- Example: single parity check code with $k = 2$, $n = 3$

$$[0 \ 0] \rightarrow [0 \ 0 \ 0]$$

$$[0 \ 1] \rightarrow [0 \ 1 \ 1]$$

$$[1 \ 0] \rightarrow [1 \ 0 \ 1]$$

$$[1 \ 1] \rightarrow [1 \ 1 \ 0]$$

General Block Codes

- The set of **all** $\{0, 1\}$ vectors of length n , together with the operations of vector (entry-by-entry) XOR addition and scalar multiplication by 0 or 1, forms a **vector space** \mathbf{V} of dimension n .
- The **standard basis** for \mathbf{V} is the set of n length- n vectors $[1, 0, \dots, 0]$, $[0, 1, \dots, 0]$, \dots , $[0, \dots, 0, 1]$.
- A **general** block code of rate $R_c = \log_2(N_c)/n$ is **any** choice of N_c distinct vectors from \mathbf{V} . A common choice is $N_c = 2^k$ for an (n, k) code with $R_c = k/n$.
- Disadvantage: difficult to encode and decode in general, since lack of structure requires using large look-up tables.

Linear Codes

- An (n, k) binary **linear** block code is a k -dimensional **subspace** \mathbf{C} of \mathbf{V} and has rate $R_c = k/n$.
- Let $\underline{g}_1, \underline{g}_2, \dots, \underline{g}_k$ be a **basis** for \mathbf{C} . Then every codeword \underline{c} in \mathbf{C} can be written uniquely as a linear combination

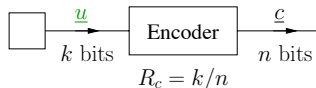
$$u_1 \underline{g}_1 + u_2 \underline{g}_2 + \dots + u_k \underline{g}_k$$

where u_1, u_2, \dots, u_k are the message bits.

- Equivalently, $\underline{c} = \underline{u} G$ where

$$G = \begin{bmatrix} \underline{g}_1 \\ \underline{g}_2 \\ \vdots \\ \underline{g}_k \end{bmatrix} \text{ is a } k \times n \text{ matrix with binary entries.}$$

- So it's easy to **encode** ... use matrix multiplication!



- Any matrix G whose rows are a basis for the linear code C is called a **generator matrix** (or encoding matrix) for C .
- An (n, k) linear code is called **systematic** if it has a generator matrix of the form $G = [I_k \ P]$ where I_k is the $k \times k$ identity matrix and P is a $k \times (n - k)$ matrix.

If it exists, such a G is called the **systematic generator matrix**.

If this G is used for encoding, the codeword is

$$\underline{c} = \underline{u}G = [\underline{u} \quad \underline{u}P]$$

- Every (n, k) linear code can be made systematic by reordering components of the codewords. Most encoders used in practice are systematic.

Parity-Check Matrices

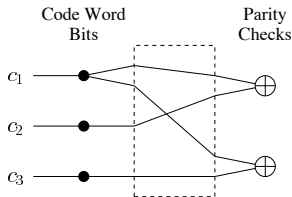
- A **parity-check matrix** for a linear code \mathbf{C} is any matrix H such that

$$\underline{c}H^T \begin{cases} = \underline{0} & \text{for all } \underline{c} \text{ in } \mathbf{C} \\ \neq \underline{0} & \text{else} \end{cases}$$

- Each row of H is a **parity-check**.
- Example: (3,1) repetition code and a graph for H

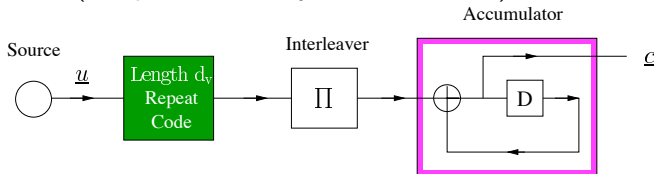
$$G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$



Repeat-Accumulate (RA) Codes

- Encoder (D represents a delay of one time unit):



- An RA code is an (n, k) linear code with generator matrix

$$G = G_R \cdot \Pi \cdot G_{ACC}$$

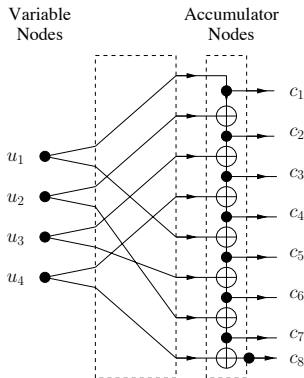
where Π is an $n \times n$ permutation matrix, and (for $d_v = 2$)

$$G_R = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \vdots & 1 & 1 \end{bmatrix}, \quad G_{ACC} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & 1 \\ 0 & 1 & 1 & \cdots & 1 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

- Example G and encoder graph:

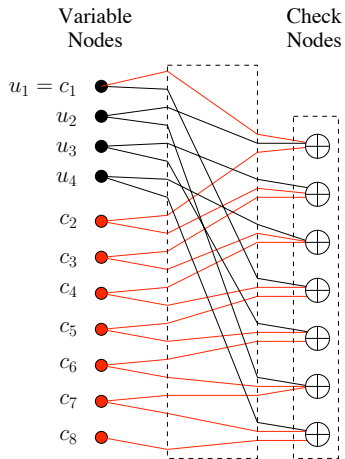
$$G = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Note that $c_8 = 0$ so we need not transmit c_8

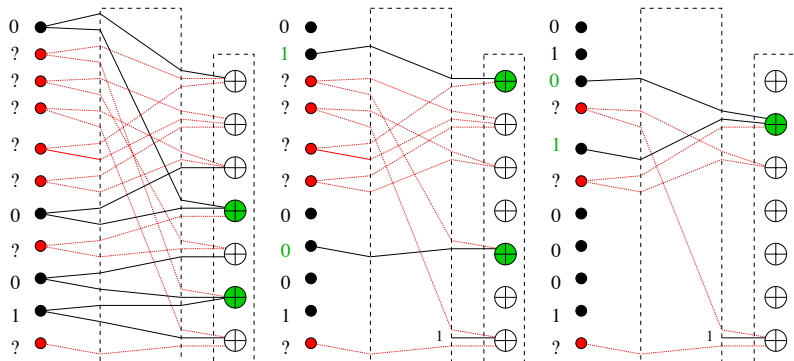


- Observe that these RA codes are nonsystematic. To make them systematic (commonly done) transmit \underline{u} also.
- For example, we can make the \underline{c} nodes variable nodes to get a low-density parity-check (LDPC) code graph (see next slide).

- Parity check matrix H graph:
- The variable nodes are effectively **repetition** codes and the check nodes are effectively **single parity-check** codes.
- Advantages of RA codes: “easy” to **encode** and to **iteratively decode** because the codes are simple
- Iterative decoding is done by **belief propagation** on the graph or some approximation thereof, e.g., bit-flipping algorithms



- Example decoding for Erasure Channel (a.k.a. “Sudoku Channel”):
 “?” and red lines represent unknown values; **iterate** to solve puzzle



- Two more iterations solve the “puzzle”: $\underline{u} = 0, 1, 0, 1$
- A third iteration solves for \underline{c} also: $\underline{c} = 0, 1, 1, 0, 0, 0, 1, 0$

Summary

Why Study Linear Block Codes ?

- Advantages of linear block codes over general block codes:
 - easier to implement encoder and, sometimes, decoder
 - easier to analyze and understand code properties
 - excellent performance for hard or soft decoding (see next section) or iterative decoding

Hard and Soft Decoding

4. Hard and Soft Decoding

Hard Decoding



- Recall that we prefer **soft** decoding and pass **numbers** to the decoder.
- Hard** decoding means passing **bits** to the decoder. This might be **necessary** when implementing very high-speed detectors.
- For BPSK or QPSK, the channel effectively becomes a **BSC** (binary symmetric channel). Code performance is thus characterized by the output BER vs. input BER curves.
- There is an **information loss** at the demodulator-decoder interface. Rate and power losses are shown on next slide.

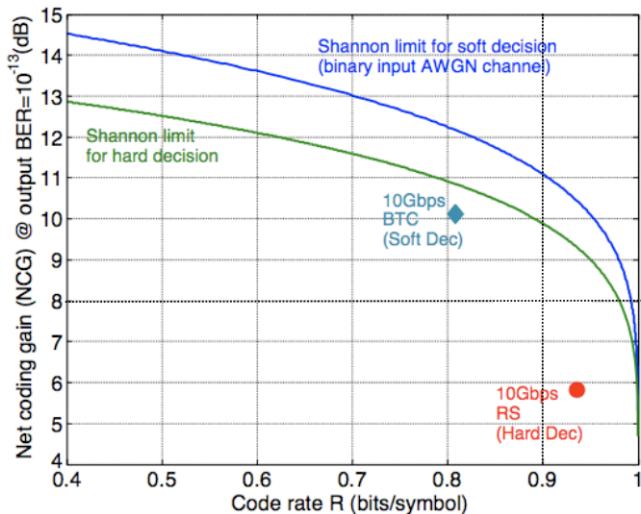


Figure reproduced from T. Mizuoichi presentation @ OFC/NFOEC2008.

- Block Turbo Code (BTC) with $R_c = 0.809$ (23.6% redundancy)
- Reed-Solomon (RS) code with $R_c = 0.9373$ (6.7% redundancy)

Soft Decoding

For **soft decoding**, one often distinguishes between

- Maximum-likelihood **sequence** decoding (MLSD)
- A Posteriori Probability (APP) decoding

MLSD:

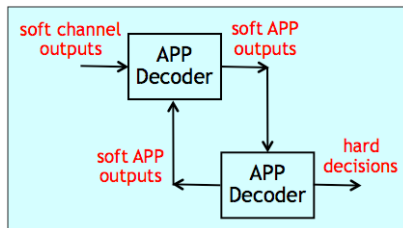
- Given \underline{y} the decoder chooses the \underline{u} that maximizes $p(\underline{y}|\underline{u})$.
- For linear block codes, one can implement MLSD efficiently using the **Viterbi Algorithm**

APP decoding:

- Given \underline{y} the decoder chooses the u_i that maximize $p(u_i|\underline{y})$ for all $i = 1, 2, \dots, k$.
- For linear block codes, one can implement APP decoding efficiently using the **Forward-Backward Algorithm** (also known as the Bahl-Cocke-Jelinek-Raviv Algorithm or BCJR Algorithm).

Turbo Decoding

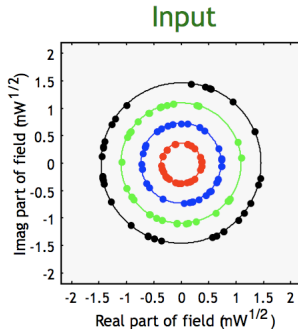
- APP decoding hardly gives better BER than MLSD detection.
Advantage: it provides **reliability** (soft) information about the u_i .
- The **soft outputs** let the APP decoder be **embedded** in an iterative (turbo) processing structure, i.e., the **decoder** soft outputs can be treated like **channel** soft outputs fed to a 2nd decoder. This decoder puts out soft outputs fed back to the first decoder, and so on.



Fiber Capacity Estimate

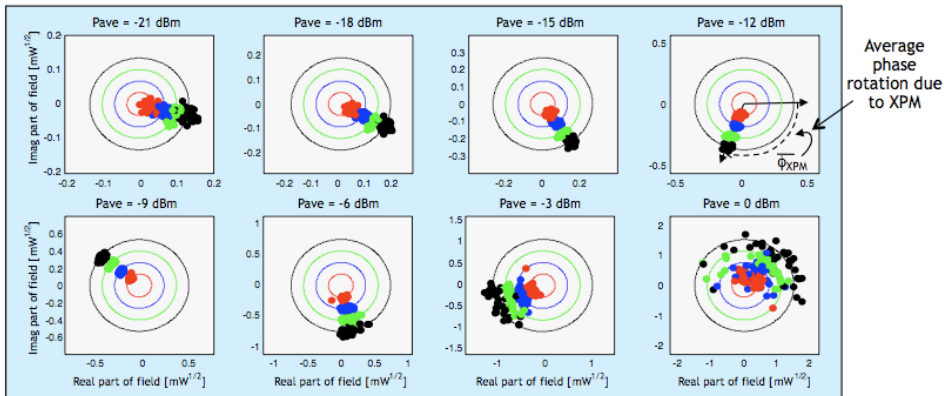
5. Fiber Capacity Estimate

- The following describes how the above ideas were used to compute a capacity lower bound in “Capacity Limits of Fiber-Optic Communication Systems,” OFC '08 tutorial by R.-J. Essiambre, G. Foschini, P. Winzer and G. Kramer.
- For more information on the fiber parameters, see the above tutorial.
- **Backpropagation** makes the channel “memoryless”.
- **Multiple-ring modulation** approximates Gaussian signaling.



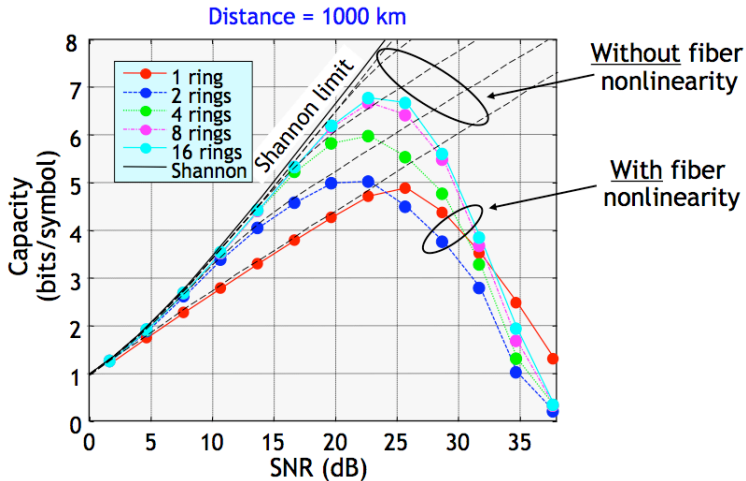
Evolution of Constellation with Signal Power

Symbol rate = 25 Gbaud, 50 GHz channel spacing



- At low launch powers, clouds are large because of the low SNR
- At high launch powers, clouds are large because of fiber nonlinearity

Capacity Estimates Including Fiber Nonlinearity



- Capacity reaches a maximum at some SNR due to fiber nonlinearity

List of Acronyms

APP	A Posteriori Probability	LAN	Local Area Network
ASK	Amplitude Shift Keying	LDPC	Low-Density Parity-Check
A/D	Analog to Digital	MLSD	Maximum-Likelihood Sequence Decoder
AWGN	Additive White Gaussian Noise	PSK	Phase Shift Keying
BCJR	Bahl-Cocke-Jelinek-Raviv	QAM	Quadrature Amplitude Modulation
BEC	Binary Erasure Channel	QPSK	Quadrature Phase Shift Keying
BER	Bit Error Rate	RA	Repeat Accumulate
BPSK	Binary Phase Shift Keying	RS	Reed-Solomon
BSC	Binary Symmetric Channel	SNR	Signal-to-Noise Ratio
BTC	Block Turbo Code	XPM	Cross-Phase Modulation
DSL	Digital Subscriber Line		
FEC	Forward Error Control		