Lecture 5

Spatial Modulation & Discrete Loading

April 13, 2026

JOHN M. CIOFFI

Hitachi Professor Emeritus of Engineering

Instructor EE379B – Spring 2026

Announcements & Agenda

- Announcements
  - Problem Set #1 Solutions posted – see PS1 (also link at course page and connects to canvas where actual solution is stored)
  - Problem Set #2 Wed April 19 at 17:00
  - Sections **1.6**, 4.4
  - Problem Set #3 due April 26 at 17:00

- Agenda
  - Spatial Modulation
  - Spatial Loading
  - Discrete Loading

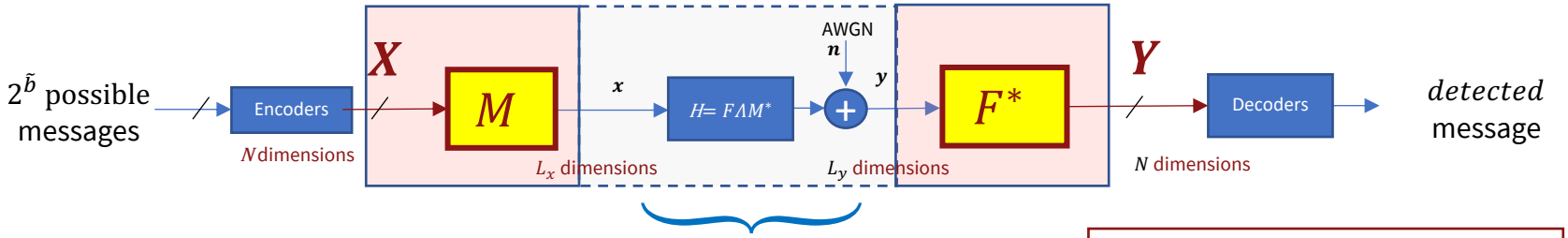
- Problem Set 3 = PS3, due 4/26
  1. 4.13 Coded-OFDM loading
  2. 4.14 ergodic waterfill
  3. 4.22 wireless spatial loading
  4. 4.16 estimating gain distribution
  5. 4.15 Wi-Fi Loading

Spatial Modulation
“Space-Time Block Codes (STBC)”

Spatial Vector Coding is Optimal

- The symbols have crosstalk or inter-spatial-dimension interference.
- However, symbols otherwise have no intersymbol interference, $\nu = 0$ (or ISI is separately handled).
  - With Vector DMT/OFDM on all crosstalking channels, there is no ISI.

- Spatial Vector-Code channel partitioning remains for each tone n (index not shown).



$$y_l = \lambda_l \cdot x_l + n_l$$

- Parallel spatial channels index as:
  - $l = 1, \ldots, L \leq \wp_H \leq \min(L_x, L_y)$.

$$SNR_l = \frac{\lambda_l^2 \cdot \bar{\mathcal{E}}_l}{\sigma^2}$$

$$\mathbb{E}[\boldsymbol{n} \cdot \boldsymbol{n}^*] = R_{\boldsymbol{nn}} = R_{\boldsymbol{nn}}^{1/2} \cdot R_{\boldsymbol{nn}}^{*/2}$$

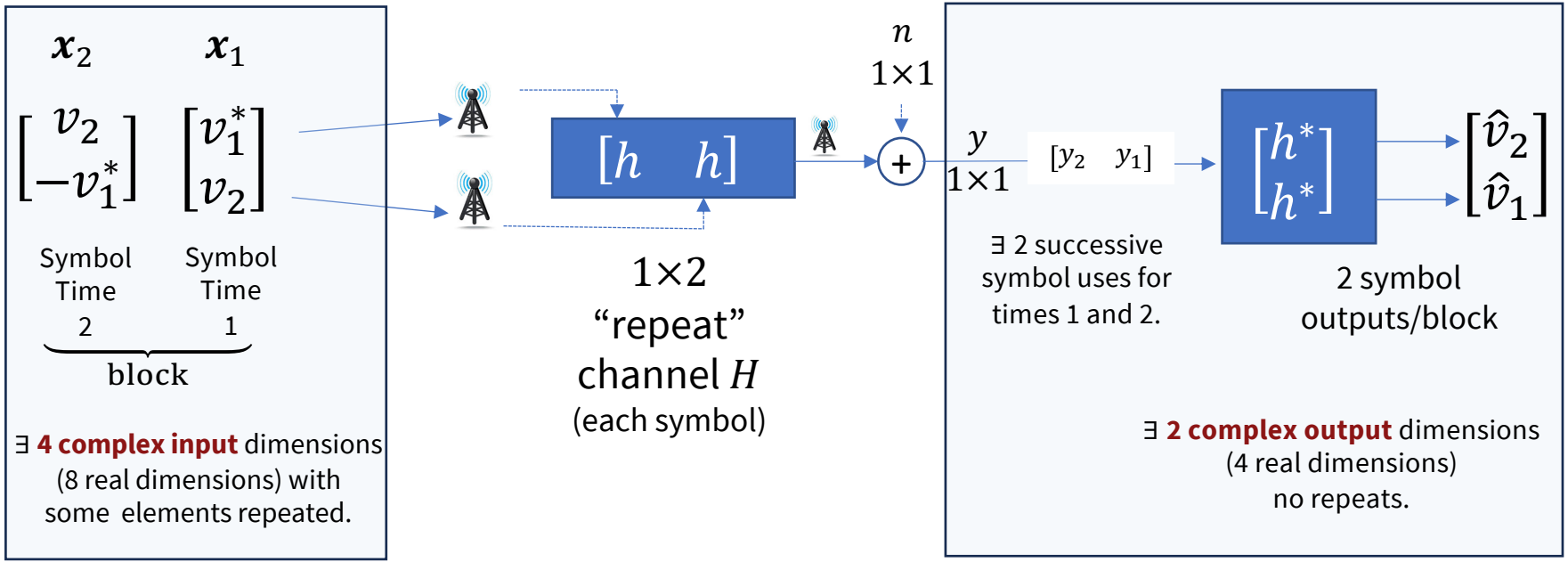
Noise-Equivalent Channel

$$\boldsymbol{y} \leftarrow R_{nn}^{-1/2}\boldsymbol{y} = \left(R_{nn}^{-1/2} \cdot H\right) \cdot \boldsymbol{x} + \tilde{\boldsymbol{n}}$$

Spatial Modulator (Matrix)

- Best matrix modulator is M for any energy distribution.
- For this choice of M, it follows easily that the unbiased MMSE receiver matrix is F^*.
- In practice, it may be difficult for the transmitter to know M, since only the receiver can measure it.
  - This requires a reverse control channel.
  - The channel may change by the time it is reversed communicated.
- This leads to a variety of spatial approximations, among them Space-Time Block Codes (STBC).

Alamouti's Code (1998)



- **The trivial 1 x 2 "repeat" channel is the ONLY channel** for which Alamouti's code is Vector Coding.
  - It obviously does better by 3 dB than a single channel use $2 \cdot [h]^2$ instead of $[h]^2$
- Basically, two line-of-sight paths to the same single-antenna receiver that must have the same gain.

Alamouti with more general 2x2 H ??

- Symmetric matrices: $J_2 \triangleq \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad J_1 \triangleq \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad R \triangleq \begin{bmatrix} a & c \\ c & b \end{bmatrix}$

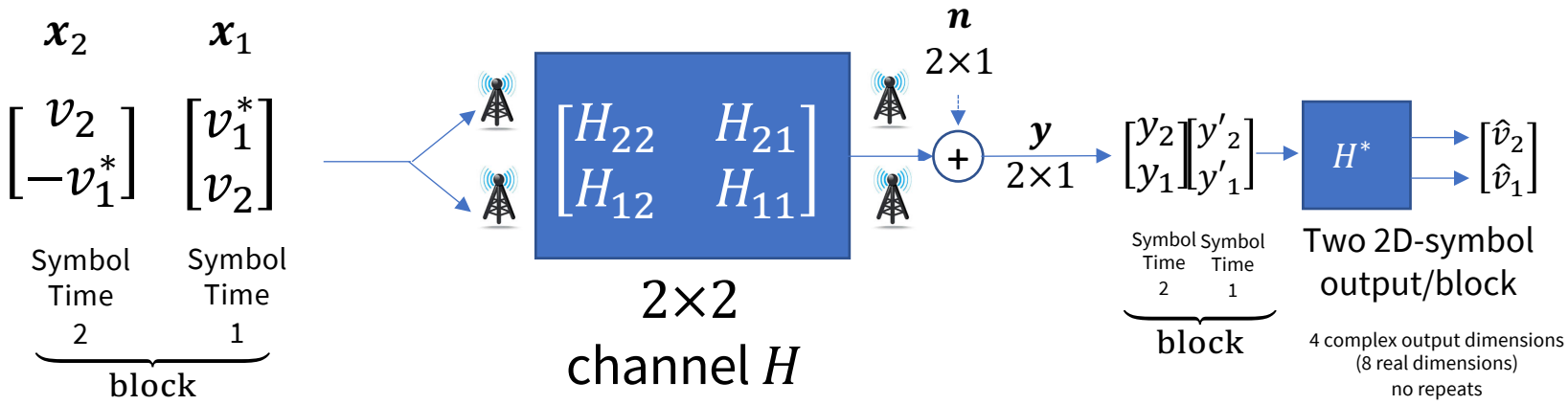
- Identity: $J_2 \cdot R \cdot J_2 + J_1 \cdot R \cdot J_1 = \begin{bmatrix} a+b & 0 \\ 0 & a+b \end{bmatrix}$

- Alamouti modulator: $\underbrace{\boldsymbol{x}}_{4\times 1} = \underbrace{\begin{bmatrix} J_2 \\ J_1 \end{bmatrix}}_{4\times 2} \cdot \underbrace{\begin{bmatrix} v_2 \\ v_1 \end{bmatrix}}_{\boldsymbol{v},2\times 1} \qquad \underset{4\times 1}{\boldsymbol{y}} = \underbrace{\begin{bmatrix} H & 0 \\ 0 & H \end{bmatrix}}_{\boldsymbol{H}\ 4\times 4} \cdot \boldsymbol{x} + \boldsymbol{n} = \begin{bmatrix} H \cdot J_2 \\ H \cdot J_1 \end{bmatrix} \cdot \boldsymbol{v} + \boldsymbol{n}$

- Forward channel autocorrelation is the block diagonal $\boldsymbol{R}_f = \boldsymbol{H}^* \cdot \boldsymbol{H} = \begin{bmatrix} R_f & 0 \\ 0 & R_f \end{bmatrix}.$

- Each 2×2 sub-block diagonal is $\begin{aligned} R_f &= J_2^* \cdot H^* \cdot H \cdot J_2 + J_1 \cdot H^* \cdot H \cdot J_1 \\ &= \begin{bmatrix} H_{22}^2 + H_{21}^2 + H_{12}^2 + H_{11}^2 & 0 \\ 0 & H_{22}^2 + H_{21}^2 + H_{12}^2 + H_{11}^2 \end{bmatrix} \end{aligned}$

SUBOPTIMAL Alamouti's Code Use, 2 x 2 channel



4 complex input dimensions
(8 real dimensions)
some symbol values repeated

$$R_f = J_2^* \cdot H^* \cdot H \cdot J_2 + J_1 \cdot H^* \cdot H \cdot J_1$$
$$= \begin{bmatrix} H_{22}^2 + H_{21}^2 + H_{12}^2 + H_{11}^2 & 0 \\ 0 & H_{22}^2 + H_{21}^2 + H_{12}^2 + H_{11}^2 \end{bmatrix}$$

$$\gamma_{repeat} = \frac{1}{2}$$

- The *transmitter remains channel independent*, and the receiver remains simple matched matrix.
- The SNR is higher, BUT the data rate is halved. **Vector coding is better** ($\gamma_{repeat} = 1$).

STBC Codebooks

- Variety of different sizes, but rate loss $\gamma_{repeat} \geq$ ½ (with more than one receive antenna).

- They perform a lot worse than vector-coding in practice.

Example: $\bar{\mathcal{E}}_x = 1\ ;\ \sigma^2 = 1$

```
>> H
H =
    1.0000    0.9000
    0.8000    1.0000
>> [F,L,M]=svd(H)
F =
   -0.7245   -0.6892
   -0.6892    0.7245
L =
    1.8512         0
         0    0.1512
M =
   -0.6892   -0.7245
   -0.7245    0.6892
bvc=log2(det(L^2+eye(2))) = 2.1790
bstbc=log2(sqrt(norm(H,'fro')^2+1)) = 1.0769
10*log10((2^bvc-1)/(2^bstbc - 1))  =  5.0245 dB!
```

Sqrt same as ½ in front, must include the repeat Factor on the two inputs.

At higher data rates, the VC advantage grows;

Wireless Raleigh fading

$$\langle P_e \rangle \propto (SNR)^{d_{free}}$$

but still large VC advantage.

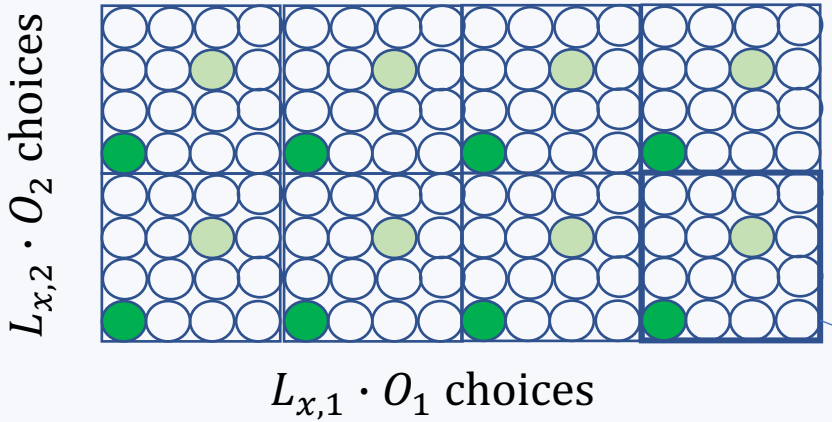
Wireless field use has STBC as "option"

- STBC will essentially repeat symbols, so if the channel is very poor and the codes (nonideal) are fixed, then it can increase SNR and create a reliable link at significant data-rate loss.
- This may be acceptable if no connection is otherwise reliably possible.
- For ideal outer codes ($\Gamma = 0$ dB), there is no such repeat-energy advantage theoretically.
- Most wireless systems (Cellular and Wi-Fi) allow use of STBC as an option, but also permit use of vector-coding approximations to avoid the STBC losses.

Spatial Loading

precoder-matrix and rank indications

Cellular: Type 1 (Linear) Precoders



O_i is oversampling factor (angle)

$L_x = L_{x,1} \cdot L_{x,2}$ = number used antennas

Each $O_1 \times O_2$ rectangle is a "subarray" of choices.
$O = O_1 \cdot O_2$

- $Q_{L_{x,1} \cdot O_1}$ is a DFT matrix of size $L_{x,1} \cdot O_1$; $Q_{L_{x,2} \cdot O_2}$ is a DFT matrix of size $L_{x,2} \cdot O_2$.
  - These 2 DFT matrices' columns are then decimated by O_1 and O_2 respectively to generate respectively $L_{x,1} \times 1$ and $L_{x,2} \times 1$ column vectors
    - $\boldsymbol{q}_{L_{x,1} \cdot O_1}(l_1)$ for $l_1 \in [0: L_{x,1}\}$, and
    - $\boldsymbol{q}_{L_{x,2} \cdot O_2}(l_2)$ for $l_2 \in [0: L_{x,2}\}$.
- Cartesian (Kronecker) products create $L_x \times 1$ vectors $\boldsymbol{a}(l_1, l_2) = \boldsymbol{q}_{L_{x,1} \cdot O_1}(l_1) \otimes \boldsymbol{q}_{L_{x,2} \cdot O_2}(l_2)$, They reindex to $l = 1: (L_x \cdot O)$ such vectors.
- For the precoder matrix A, the rcvr specifies the
  - **Rank Index (RI)** $\leq L_x$, $L_x \times 1$ columns of these $\boldsymbol{a}(l_1, l_2)$
  - **Precoder Matrix Indicator (PMI)** ~ set of RI $(l_{1,l}, l_{2,l})$ and a phase $\phi_{l_{1,l}, l_{2,l}} \in \{1, j, -1, -j\}$ for each (chooses these as closest to M from SVD of H)
  - **Channel Quality Index (CQI)** ~ $[r\ , |C|]$

Cellular: Type 2 Precoders

- Type 2 uses reindexed Type 1 columns $\boldsymbol{a_l} = \boldsymbol{a}(l_{1,l}, l_{2,l})$ into a weighted sum.
- Weights $a_{l,ss}$ can have amplitudes $2^{-i/2}$ for $i \in \{0, \ldots, 6\}$ and phases $\frac{2\pi m}{M}$ $m \in \{0, \ldots, M-1\}$.
- Receiver selects $L > RI$ beam vectors $\boldsymbol{a}_l$, from which RI new $L_x \times 1$" beams" form as, per spatial stream (ss):

$$\boldsymbol{a}'_{ss} = \sum_{l=1}^{L>RI} a_{l,ss} \cdot \underbrace{\boldsymbol{a}(l_{1,l}, l_{2,l})}_{\boldsymbol{a}_l} \qquad ss \in [1:RI]$$

$$a_{l,ss} = 2^{-i/2} \cdot e^{j\frac{2\pi m}{M}}$$

$$i \in [0:6], m \in [0:M-1]$$

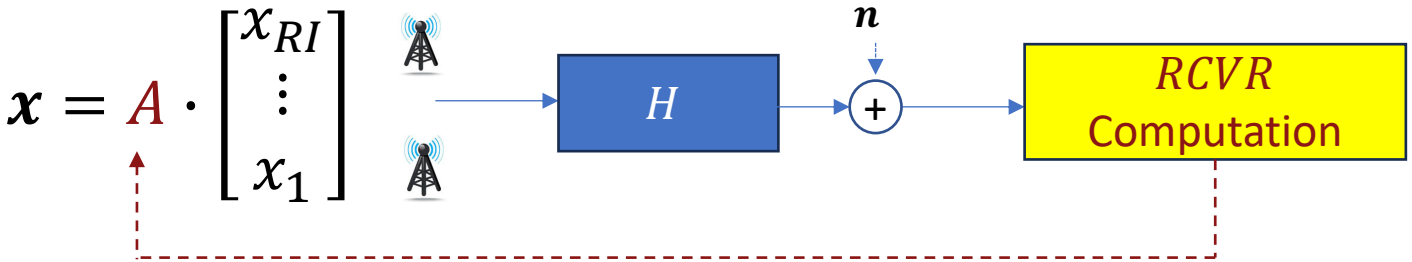
- PMI includes beam indices $(l_{1,l}, l_{2,l})$ for $l \in [1:L]$ and the $(i_{l,ss}, m_{l,ss})$ for $l \in [1:L]$ & $ss \in [1:RI]$
  - Type 1 $\phi_{l_{1,l},l_{2,l}}$ is subsumed in the Type 2 additional quantities, so does not need separate specification.

$$A = [\boldsymbol{w}_1 \cdots \boldsymbol{w}_{RI}] = A' \cdot C \qquad A' = [\boldsymbol{a}'_{RI} \cdots \boldsymbol{a}'_1] \qquad C = \left[a_{l,ss}\right]_{\substack{ss=RI:-1:1 \\ l=L:-1:1}}$$

$L_x \times RI$ $\quad$ $L_x \times L$ $\quad$ $L \times RI$

$$rank(A) \leq RI \leq L \leq L_x$$

Spatial Loading chooses the "A"



- The $\mathbb{I} = \sum_{n=0}^{N_{sc}-1} \log_2 |I + H_n \cdot A \cdot S_x \cdot A^* \cdot H_n^*|$ can be maximized over the choices for A.
  - The A and S_x do not vary with n, apart from the RI used spatial dimensions (in cellular).
    - Of course water-filling solves this in general when they can vary.
  - Indeed, the S_x is a constant diagonal and Aa constant $L_x \times RI$ matrix independent of tone index n.
- One approach is to try all the possible $\boldsymbol{a_l}, i, m, A$ possibilities and pick the largest one.
  - This has enormous complexity for reasonable $L_{x,1}$ and $L_{x,2}$.
- Spatial loading addresses this selection more efficiently.

A Spatial Loading Strategy

- **Step 1:** find RI and A: Form $R_{f,ave} = \frac{1}{N_{sc}} \cdot \sum_{n=0}^{N_{sc}-1} H_n^* \cdot H_n$, which will be square and of size $L_x \times L_x$.
  - Find the eigenvalue decomposition $R_{f,ave} = \sum_{l=1}^{L_x} \lambda_l \cdot \boldsymbol{v}_l \cdot v_l^*$.
  - Compute waterfill for subchannels with gains λ_l and the total energy budget $\mathcal{E}_x$.
  - Set L to be waterfill's L^* , presume any sorting and unsorting of the gains and indices
  - Dominant subspace is $A_{wf} = [\boldsymbol{v}_1 \cdots \boldsymbol{v}_{L^*}]$ and $R_f = \sum_{l=1}^{L^*} \lambda_l \cdot \boldsymbol{v}_l \cdot v_l^*$. (and $RI = L^*$)

- **Step 2:** Compute $L_x \cdot O$ candidates' scores $\gamma_k = \boldsymbol{a}_k^* \cdot R_f \cdot \boldsymbol{a}_k$ and keep the K top survivors from the $\{\boldsymbol{a}_k\}$.
  - $K > L^*$ but somewhat discretionary as to what the γ_k threshold is (looking for significant drop beyond).
  - Complexity reduced from $\binom{L_x \cdot O}{L^*}$ to $\binom{K}{L^*}$

- **Step 3:** Form $T = \binom{K}{L^*}$ candidate subsets, where each subset t selects L^* beams from K possible survivors $\boldsymbol{a}_k$:

$$\boldsymbol{a'}_{ss,t} = \sum_{l=1}^{L^*} a_{l,ss} \cdot \boldsymbol{a}_{l,t} \quad t \in [1:T] \qquad A_t = [\boldsymbol{a}_{L^*,t} \cdots \boldsymbol{a}_{1,t}]$$

$$a_{l,ss} = 2^{-i/2} \cdot e^{j\frac{2\pi m}{M}}$$

$$i \in [0:6], m \in [0:M-1]$$

- **Step 4:** Final A is found as

$$A = \arg\max_t tr\{A_t^* \cdot R_f \cdot A_t\} \text{ or to maximize directly}$$

$$\mathcal{I} \cong \sum_{n=0}^{N_{sc}-1} \log_2 |I + H_n \cdot A_t \cdot S_x \cdot A_t^* \cdot H_n^*|$$

Wi-Fi Loading

- Fewer antennas and slower channels (less variation/Doppler)

- So Wi-Fi does the SVD per tone n (or per groups of tones like 4, 16, …, with wider bands).
  - Sends back SVD's M or an eigenvector equivalent when groups of tones are used, and with either: $SNR_{geo,ss}$.
  - Or subset of M's columns.
- The Wi-Fi is packet-based (CSMA) so trains/updates for each
  - Transmitter decides L^* (useful rank based on feedback) and the MCS $[r\ , |C|]$ per USER (so MCS common to all ss for single user)
  - Cellular is continuously on and changing more rapidly, and thus feedback would take too much bandwidth.

- The M is quantized by standardized series of "Complex Givens" rotations.

- But again, the water-fill energy is only approximated by silencing some weaker spatial streams.

- Cellular has Type 1 and Type 2, allowing for lots of antennas and oversampling in space.
  - This has dubious value in Wi-Fi as the AP could be placed anywhere.

Wi-Fi Steps (downlink)

- **Step 1:** Receiver identifies the channel H_n from training packets (see Chapter 9, L17).

$$H_n = F_n \cdot \Lambda_n \cdot M_n^* \qquad M_n \text{ is the best } \textit{linear} \text{ precoder.}$$

For groups $R_{f,ave} = \frac{1}{N_{sc}} \cdot \sum_{n=0}^{N_{group}-1} H_n^* \cdot H_n = \sum_{l=1}^{L_x} \lambda_l \cdot \boldsymbol{v}_l \cdot v_l^*$, and V_{group} is the selected *linear* precoder (pretend this is M_n for rest below)

- **Step 2:** Receiver does QR factorization of $M_n \neq Q_n \cdot R_n = Q_n$ for a unitary matrix.
  - The factorization process that leads to upper triangular $R_n = I$ matrix uses complex Givens Rotations
  - Starting column 1 bottom up, then column 1 up until only 1 (upper left) nonzero entry remains $(L_x - 1)$ rotations.
  - Then columns 2 through $L_{max} \leq L_y$.
  - The **complex** rotations have **two** angles, each angle quantized.

$$Q_2(\phi, \varphi) = \begin{bmatrix} cos(\varphi) & -e^{-j\phi} \cdot sin(\varphi) \\ e^{-j\phi} \cdot sin(\varphi) & cos(\varphi) \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \sqrt{|x|^2 + |y|^2} \\ 0 \end{bmatrix}$$

High or Low SNR:

$\phi = \angle x - \angle y \in [0: 2\pi]$ (4-6 bits)

$\varphi = \tan^{-1} \frac{|y|}{|x|} \in [0: \frac{\pi}{2}]$ (2-4 bits)

For multiuser: (7-9 bits)

(5-7 bits)

- Return the angles for L_{max} columns (dominant subspace) to transmitter to implement as M_n.
- Transmitter (AP) may subsequently reduce L_{max} on later uses
  - (it pre-decides on each use – I don't like this, "standards mistake!")

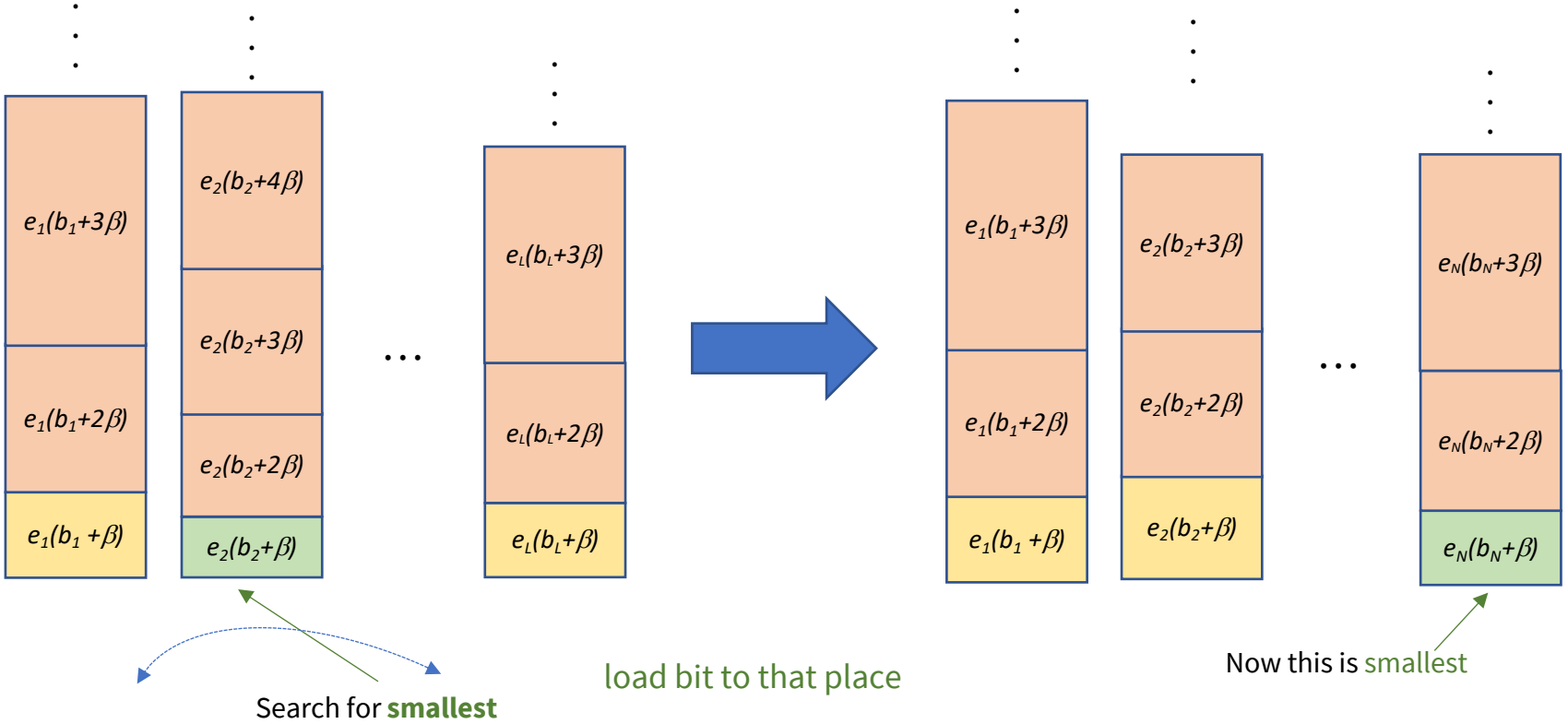
Discrete Loading

Bit and Energy Distributions

- Bit distribution $\boldsymbol{b} = [b_1 \quad b_2 \quad \cdots \quad b_N]$
- Energy distribution $\boldsymbol{\mathcal{E}} = [\mathcal{E}_1(b_1) \quad \mathcal{E}_2(b_2) \quad \cdots \quad \mathcal{E}_N(b_N)]$
- Incremental energy $e_l(k \cdot \beta) \triangleq \mathcal{E}_l(k \cdot \beta) - (k \cdot \beta - \beta)$
  - β is smallest info unit (e.g., 1 bit)
  - $\mathcal{E}_l(0) = 0, k = 1, \ldots$

Incremental Energy Table (Greedy Algorithm)

- Create table of incremental energies for each dimension, some are better – some worse



- Repeat until RA (all energy used) or until MA (desired rate attained)

Some aspects of the incremental energy table

- Dependencies
  - Constellation, code/gap (can now be different on different dimensions if desired)
  - Channel gain g_l
  - Bit caps $b_l \leq b_{max}$ (any real system cannot carry infinite number of bits)
  - Spectrum or antenna-energy masks

$$\mathcal{E}_n = \sum_{k=1}^{b_n} e_n(k \cdot \beta)$$

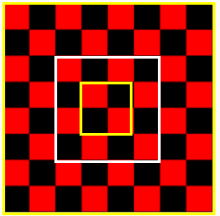
- "easier" than water-filling for adding constraints
  - Spectrum or antenna-energy masks
    - Set incremental energy to infinite (large) value if no more bits allowed
  - Different dimensions can have different total energy constraints
    - (MIMO antennas)
  - Bit caps $b_n \leq b_{max}$ (any real system cannot carry infinite number of bits)

Example Common Table for SQ QAM

- Use Square QAM on all tones with β=1
  - *Each tone's xmit distance d_n scales inversely by square root g_n to provide constant d^2/noise at detector*

Xmit Energy =

$$\mathcal{E}_n(b_n) = \begin{cases} \frac{2^{b_n}-1}{6} d_n^2 & b_n \text{ even} \\ \frac{2^{b_n+1}-1}{12} d_n^2 & b_n \text{ odd} \end{cases}$$



- **Incremental energy** can be computed generally as function of d_n also

$$e_n(b_n) = \begin{cases} 2^{b_n-1}/12 \; d_n^2 & b_n \text{ even} \\ 2^{b_n+1}/12 \; d_n^2 & b_n \text{ odd} \end{cases}$$

- Can tabulate for each tone/group after inverse-square-root-channel-gain scaling

Example continued

- Before Scaling:

b_n	0	1	2	3	4	5	6	7	8
$\mathcal{E}_n(b_n)$	0	$\frac{d_n^2}{4}$	$\frac{d_n^2}{2}$	$\frac{5d_n^2}{4}$	$\frac{5d_n^2}{2}$	$\frac{21d_n^2}{4}$	$\frac{21d_n^2}{2}$	$\frac{85d_n^2}{4}$	$\frac{85d_n^2}{2}$
$e_n(b_n)$	0	$\frac{d_n^2}{4}$	$\frac{d_n^2}{4}$	$\frac{3d_n^2}{4}$	$\frac{5d_n^2}{4}$	$\frac{11d_n^2}{4}$	$\frac{21_n^2}{4}$	$\frac{43d_n^2}{4}$	$\frac{85d_n^2}{4}$

- The incremental energy is monotonic in this example. Why?
  - This is important for suggested algorithm to be optimum
  - Most rationally designed constellation approaches have this property

- The receiver stores only one column, but scales the next-out entry to be evaluated (the yellow/green ones earlier) by

$$\frac{1}{\sqrt{g_n}} \; or \; \frac{1}{\sqrt{g_{geo,n}}}$$

Works for $g_{geo,n}$ to find $[r\,, |C|]$ for different spatial dimension (or groups)

Does this really work? Is it optimum? Yes & Yes!
(when $e_n(b_n)$ function is monotonically non-decreasing with bits)

Efficiency Concept introduced by Campello

- Adding one more bit (unit of β) takes more energy than energy savings for deletion of any bit
- An **efficient** bit distribution $\max_n e_n(b_n) \leq \min_{m \neq n} e_m(b_m + \beta)$
- If the bit distribution is efficient (and the incremental energy is monotonic):
  - No other bit distribution (which would need to move bits) could use less energy → optimum
- Rate Adaptive (RA) optimum solution if $\mathcal{E}_x = N \cdot \bar{\mathcal{E}}_x = \sum_{n=1}^{N} \mathcal{E}_n$
- Margin Adaptive optimum solution if $b = N \cdot \bar{b} = \sum_{n=1}^{N} b_n$

“Efficientize” a bit distribution (Campello)

EF Algorithm

① $m \leftarrow \arg\left\{ \min_{1 \le i \le N} \left[e_i(b_i + \beta) \right] \right\}$ position of least among next — Incremental energy

② $n \leftarrow \arg\left\{ \max_{1 \le j \le N} \left[e_j(b_j) \right] \right\}$ position of most among current — Incremental energy

③ while $e_m(b_m + \beta) < e_n(b_n)$

swap “bit” from n to m

$b_m \rightarrow b_m + \beta$

$b_n \rightarrow b_n - \beta$

update m, n from steps 1, 2

- Relies on monotonicity, but the algorithm will converge to an efficient bit distribution (always)

$1+.9D^{-1}$ Example Revisted for EF

- First Step? Compute the incremental energy table

$$\mathcal{E}_{PAM,n}(b_n) = \frac{10^{.88}}{g_n}\left(2^{2b_n}-1\right) \quad n=0,4 \qquad \mathcal{E}_{QAM,n}(b_n) = 2\cdot\frac{10^{.88}}{g_n}\left(2^{b_n}-1\right) \quad n=1,2,3$$

- *n=0*:

$$\mathcal{E}_0 = \frac{\Gamma}{g_0}\left(2^{2b_0}-1\right) = \frac{10^{.88}}{20}\left(2^{2b_0}-1\right)$$

$$e_0 = \frac{10^{.88}}{20}\left(2^{2b_0}-2^{2(b_0-1)}\right)$$

PAM

- *n=1,2,3*:

$n = 1,2,3$

$$\mathcal{E}_n = 2\frac{\Gamma}{g_n}\left(2^{b_n}-1\right) = \frac{2(10^{.88})}{17,10,\text{or } 3}\left(2^{b_n}-1\right)$$

$$e_n = \frac{2(10^{.88})}{17,10,3}\left(2^{b_n}-2^{b_n-1}\right)$$

QAM

- *n=4*:

$$\mathcal{E}_4 = \frac{\Gamma}{g_4}\left(2^{b_4}-1\right)$$

$$e_4 = \frac{10^{.88}}{.055}\left(2^{b_4}-2^{b_4-1}\right)$$

SSB

1 bit added is
1 bit/symbol (8 real dims)
or .125 bits per dimension

Whether added PAM or QAM
For QAM, 2 successive 3's are same

Incremental Energy Table for $1+.9D^{-1}$ uncoded

- Table – note smallest energy on top here

n	0	1	2	3	4
$e_n(1)$	1.14	.891	1.50	5.112	412.3
$e_n(2)$	4.56	1.78	3.03	10.2	—
$e_n(3)$	18.3	3.56	6.07	20.4	—
$e_n(4)$	—	7.13	12.1	—	—
$e_n(5)$	—	14.2	24.3	—	—

note 6dB/bit in colum 0 (4)

3dB/bit in colum 1-3

- Start with 8 bits allocated as [0 5 0 2 1]
  - 5 iterations
  - Energy =

.891+1.14+1.5 +1.78+3.56 +3.03 +4.56+5.11 = 21.6 > 8 !

1. $\boldsymbol{b} = [0\ 5\ 0\ 2\ 1]$
2. $\boldsymbol{b} = [1\ 5\ 0\ 2\ 0]$
3. $\boldsymbol{b} = [1\ 4\ 1\ 2\ 0]$
4. $\boldsymbol{b} = [1\ 4\ 2\ 1\ 0]$
5. $\boldsymbol{b} = [2\ 3\ 2\ 1\ 0].$

- Round MA Water-filling?

$b_0 = 1.71 \rightarrow 2$

$b_1 = 3.19 \rightarrow 3$

$b_2 = 2.42 \rightarrow 2$

$b_3 = .678 \rightarrow 1$

same this time

Energy Tightness

- E-tight Definition for efficient distribution:

$$0 \le N\bar{\mathcal{E}}_{\boldsymbol{x}} - \sum_{n=1}^{N} \mathcal{E}_n(b_n) \le \min_{1 \le i \le N} [e_i(b_i + \beta)]$$

No more bits can be added

- E-tight Algorithm:

WHILE $(N\bar{E}_x - S) < 0$ OR $N\bar{E}_x - S \geq \min_i e_i(b_i + \beta)$

IF $(N\bar{E}_x - S) < 0$

(a). $n \leftarrow \arg\{\max_i e_i(b_i)\}$ index of costly

(b). $S \leftarrow S - e_n(b_n)$

(c). $b_n \leftarrow b_n - \beta$

} Delete "bit"

ELSE

(a) $m \leftarrow \arg\{\min_i e_i(b_i + \beta)\}$ index of cheapest

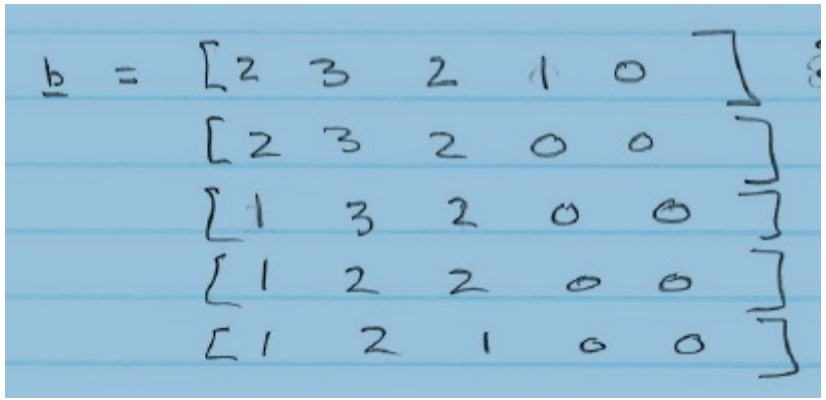
(b) $S \leftarrow S + e_m(b_m + \beta)$

(c) $b_m \leftarrow b_m + \beta$

} Add "bit"

E-Tighten earlier RA Example

- Reduce most costly bits first until energy < 8



21.6
16.5
11.9
8.3
5.3

$\gamma = 8/5.3 = 1.8$ dB $\quad \bar{b} = 1/2 \quad b = 4$

note transmission has $P_e < 10^{-6}$ at $b = 4$

- Note how such a system allows bits/dimension granularity of 1/8 bit dimension
- The symbol blocking (of 8 dimensions together) indirectly allows fractional bits

RALC

- Matlab software at the web site

```
function [gn,En,bn,b_bar]=LC(P,SNRmfb,Ex_bar,Ntot,gap)

% Levin Campello's Method
%
% P is the pulse response
% SNRmfb is the SNRmfb in dB
% Ex_bar is the normalized energy
% Ntot is the total number of real subchannels, Ntot>2
% gap is the gap in dB
%
% gn is channel gain
% En is the energy in the nth subchannel (PAM or QAM)
% bn is the bit in the nth subchannel (PAM or QAM)
% Nstar is the number of subchannel used
% b_bar is the bit rate
%
% The first bin and the last bin is PAM, the rest of them are QAM.
```

```
>> [gn,En,bn,b_bar]=LC([.9 1],10,1,8,8.8)

gn =

   19.9448   17.0320   10.0000    2.9680    0.0552


En =

    1.1410    2.6723    1.5172         0         0


bn =

     1     2     1     0     0


b_bar =

    0.5000
```

Bit Tightness

- T-tight definition for efficient distribution:

$$b = \sum_{n=1}^{N} b_n$$

- BT Algorithm:

$\tilde{b} = \sum b_n$

while $\tilde{b} \neq b$

if $\tilde{b} > b$

$n \leftarrow \arg\{\max_i [e_i(b_i)]\}$

$\tilde{b} \leftarrow \tilde{b} - \beta$

$b_n \leftarrow b_n - \beta$

} "bit" delete

ELSE

$m \leftarrow \arg\{\min_i [e_i(b_i)]\}$

$\tilde{b} \leftarrow \tilde{b} + \beta$

$b_n \leftarrow b_n + \beta$

bit add

$1+.9D^{-1}$ again for BT

- Reverse the previous example to get back to 8 bits / symbol (with -4.3 dB margin)

- From zero-bit start (which is always efficient trivially)

1. $\boldsymbol{b} = [0\ 0\ 0\ 0\ 0]$
2. $\boldsymbol{b} = [0\ 1\ 0\ 0\ 0]$
3. $\boldsymbol{b} = [1\ 1\ 0\ 0\ 0]$
4. $\boldsymbol{b} = [1\ 1\ 1\ 0\ 0]$
5. $\boldsymbol{b} = [1\ 2\ 1\ 0\ 0]$
6. $\boldsymbol{b} = [1\ 2\ 2\ 0\ 0]$
7. $\boldsymbol{b} = [1\ 3\ 2\ 0\ 0]$
8. $\boldsymbol{b} = [2\ 3\ 2\ 0\ 0]$
9. $\boldsymbol{b} = [2\ 3\ 2\ 1\ 0]$

n	0	1	2	3	4
$e_n(1)$	1.14	.891	1.50	5.112	412.3
$e_n(2)$	4.56	1.78	3.03	10.2	—
$e_n(3)$	18.3	3.56	6.07	20.4	—
$e_n(4)$	—	7.13	12.1	—	—
$e_n(5)$	—	14.2	24.3	—	—

note
6dB/bit
in colum
0 (4)
3dB/bit
in colum
1-3

MALC

- Matlab software at the web site

```
function [gn,En,bn,b_bar_check,margin]=MALC(P,SNRmfb,Ex_bar,b_bar,Ntot,gap)

%
% Levin Campello's Method
%
% P is the pulse response
% SNRmfb is the SNRmfb in dB
% Ex_bar is the normalized energy
% Ntot is the total number of real/complex subchannels, Ntot>2
% gap is the gap in dB
% b_bar is the bit rate

% gn is channel gain
% En is the energy in the nth subchannel (PAM or QAM)
% bn is the bit in the nth subchannel (PAM or QAM)
% b_bar_check is the bit rate for checking - this should be equal to b_bar
% margin is the margin

% The first bin and the last bin is PAM, the rest of them are QAM.
```

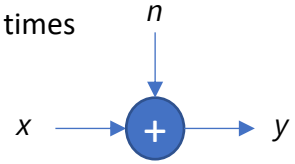
```
>> [gn,En,bn,b_bar_check,margin]=MALC([.9 1],10,1,1,8,8.8)

gn =

   19.9448   17.0320   10.0000    2.9680    0.0552


En =

    5.7051    6.2354    4.5515    5.1117         0


bn =

     2     3     2     1     0


b_bar_check =

     1


margin =

   -4.3144
```

Finding an equivalent single AWGN

- **Sum the bits** – for total

$$\bar{b} = \frac{1}{2}\log_2\left[\prod_{n=1}^{N}\left(1+\frac{SNR_n}{\Gamma}\right)\right]^{1/N} = \frac{1}{2}\log_2\left(1+\frac{SNR_{m,u}}{\Gamma}\right)$$

- **Geometric SNR** – for equivalent single channel
  - Use this channel N times



Equivalent SNR_{geo}

$$SNR_{m,u} = \Gamma\cdot\left\{\left[\prod_{n=1}^{N}\left(1+\frac{SNR_n}{\Gamma}\right)^{1/N}\right]-1\right\}$$

When $\frac{SNR_n}{\Gamma} \gg 1$, then

$$SNR_{geo} \approx \sqrt[N]{SNR_1 \cdots SNR_N}$$

$$\bar{b} = \frac{1}{2}\log_2\left(1+\frac{SNR_{m,u}}{\Gamma}\right)$$

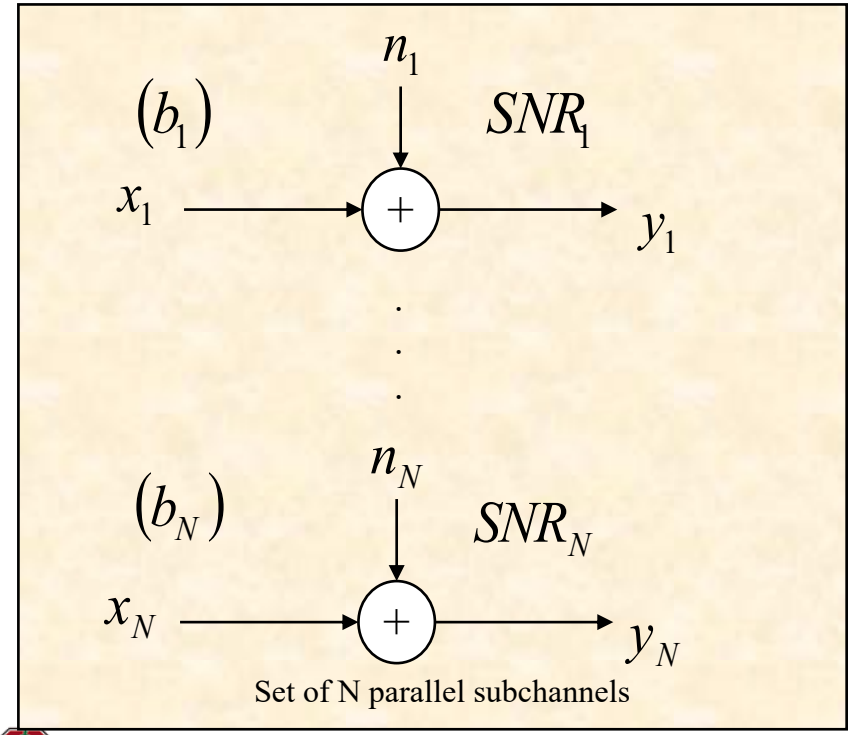
- **Allows comparison – an easier calculation than SNR_{geo} when loading is done?**

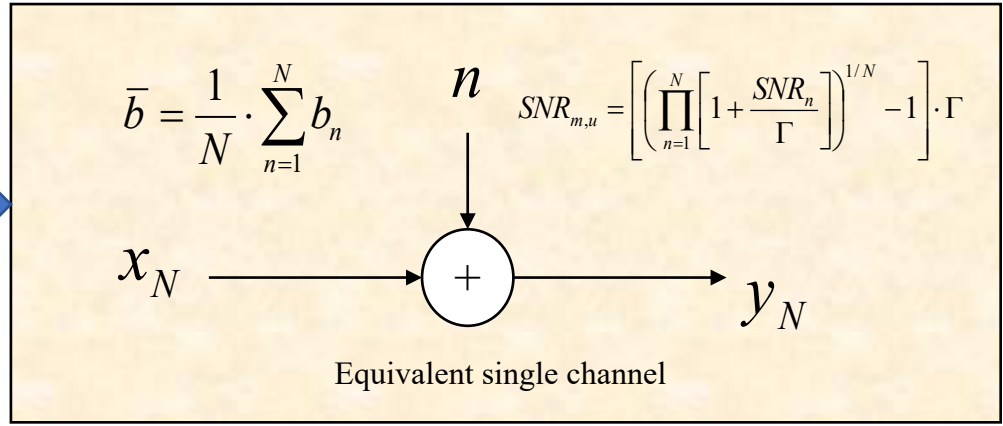
$$SNR = \Gamma\cdot\left(2^{2\bar{b}}-1\right)$$

Just need total bits/N

Once Loading is Done: Important concept

- Set of dimensional channels can each be individually addressed (coded and designed)
  - But there is an equivalent single channel





With whatever bits and energy on each subchannel

End Lecture 5