Usefulness of linear algebra for sending bits.

Example: three users Alice, Bob, Mira

→ cell tower wants to send each user 1 bit

→ in parallel; not TDMA

Four steps:

(i) sender (i.e., cell tower): hide the 3 bits to be sent to Alice, Bob, Mira in 3-D vector

→ 3-D because 3 receivers

→ linear algebra

(ii) sender: transmit 3-D vector as EM wave

→ called CDMA (3G cellular technology)

→ ignore since not relevant

(iii) receiver (i.e., Alice, Bob, Mira): translate received EM into 3-D vector

→ Alice, Bob, Mira hear the same EM wave: broadcast

→ ignore EM to 3-D vector translation

(iv) receiver: Alice decodes her bit from 3-D vector

→ same goes for Bob and Mira

→ linear algebra

Example (cont.): assign each user a 3-D vector called code vector

→ $x = (x_1, x_2, x_3)$ for Alice

→ $y = (y_1, y_2, y_3)$ for Bob

→ $z = (z_1, z_2, z_3)$ for Mira

Rule: choose x, y, z so that

→ basis of 3-D space

→ mutually orthogonal

Recall: x and y are orthogonal (at right angle geometrically speaking) if their dot product is 0.

→ $x \circ y = x_1 y_1 + x_2 y_2 + x_3 y_3 = 0$

Dot product measures how similar two vectors are.

→ 0 means maximally dissimilar

Example (cont.): three mutually orthogonal vectors

→ Alice: (1,-2,1)

→ Bob: (3,5,7)

→ Mira: (19,4,-11)

We will allow positive and negative values.

→ not worry about converting to bits

The three code vectors are stored in Alice, Bob, Mira's smart phones.

→ their secret key

Suppose the 3 bits to be sent to Alice, Bob, Mira are:

- Alice: 1
- Bob: 0
- Mira: 1

We will use +1 to mean bit value 1, -1 to mean bit value 0.

→ convenient but not necessary

Step (i): cell tower computes 3-D vector by multiplying Alice's code vector by +1, Bob's code vector by -1, Mira's code vector by +1

→ since bits to send are 1, 0, 1

Then add the result to yield 3-D vector:

$$(+1)(1,-2,1)+(-1)(3,5,7)+(+1)(19,4,-11) = (17,-3,-17)$$

Step (ii): cell tower transmits (17,-3,-17) via EM broadcast

→ ignore how (17, -3, -17) is translated into electromagnetic wave

Step (iii): Alice's smartphone NIC converts EM wave into 3-D vector (17, -3, -17).

$\rightarrow$ same for Bob and Mira's smart phones

Step (iv): Alice retrieves her bit from 3-D vector (17, -3, -17) by performing dot product with her code vector (1, -2, 1):

$\rightarrow (17, -3, -17) \circ (1, -2, 1) = 17 + 6 - 17 = 6 > 0$

$\rightarrow$ positive means bit value 1

$\rightarrow$ negative means bit value 0

Same for Bob with his code vector (3,5,7) and Mira with her code vector (19,4,-11):

For Bob:

$\rightarrow (17, -3, -17) \circ (3, 5, 7) = 51 - 15 - 119 = -83 < 0$

$\rightarrow$ hence Bob received bit 0

For Mira:

$\rightarrow (17, -3, -17) \circ (19, 4, 11) = 323 - 12 + 119 = 187 = 498 > 0$

$\rightarrow$ hence Mira received bit 1

Why does this work?

$\rightarrow$ consider crucial role of orthogonality

Generalize: to send 1 bit per user to n users in parallel

- Set-up: assign n orthogonal code vectors in n-dimensional vector space

  $\rightarrow \mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^n$

  to n users

- Sender: to send n data bits $a_1, a_2, \ldots, a_n$ (+1 for 1, -1 for 0) in parallel, compute

  $\rightarrow \mathbf{z} = a_1\mathbf{x}^1 + a_2\mathbf{x}^2 + \cdots + a_n\mathbf{x}^n$

  $\rightarrow$ $\mathbf{z}$ is an n-dimensional vector that hides n bits in its coefficients (called spectra)

  $\rightarrow$ convert $\mathbf{z}$ into analog signal and transmit to all receivers

- Receiver: to decode user i'th bit a_i, receiver computes dot product with code vector $\mathbf{x}^i$

  $\rightarrow \mathbf{z} \circ \mathbf{x}^i = a_i(\mathbf{x}^i \circ \mathbf{x}^i) = a_i \times$ positive constant

  $\rightarrow$ by orthogonality

To send not single bit per user but a stream of bits per user:

$\rightarrow$ do the above sequentially in the parallel bit stream

Problem solved?

Limitations: heavy lifting from translating 3-D vectors to EM waves, and vice versa, remains.

→ challenging

Instead: go directly from bits to EM waves.

→ make EM waves our basis vectors

→ use multiple frequencies to carry multiple bit streams

→ called carrier frequencies

→ FDM (frequency division multiplexing)

Our goal:

→ understand limitations of FDM

→ mitigate using OFDM (orthogonal FDM)