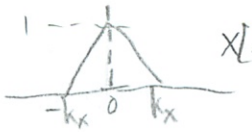
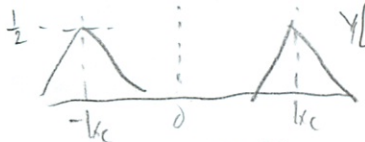
6.02 Cheat Sheet 3

Modulation - Have a band assigned to you
Only transmit in that band
Have signal around baseband



Freq of cos

Then multiply each timestep by $\cos(k_c \frac{2\pi}{N} n)$



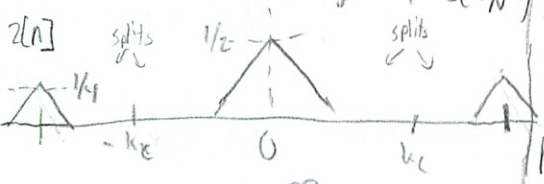
$Y[n] = x[n] \otimes \cos(k_c)$

$$y[n] = \sum_{k=-kx}^{kx} a_k e^{jk\frac{2\pi}{N}n} \cdot \frac{1}{2} e^{jk_c\frac{2\pi}{N}n} + \frac{1}{2} e^{-jk_c\frac{2\pi}{N}n}$$

Signal — the cos

$$= \frac{1}{2}\sum_{k=-kx}^{kx} a_k e^{j(k+k_c)\frac{2\pi}{N}n} + \frac{1}{2}\sum_{k=-kx}^{kx} a_k e^{j(k-k_c)\frac{2\pi}{N}n}$$

This is then transmitted over a wire
At reciever multiplied again by $\cos(k_c \frac{2\pi}{N} n)$



$$z[n] = y[n] \cdot \cos\left(k_c \frac{2\pi}{N} n\right)$$

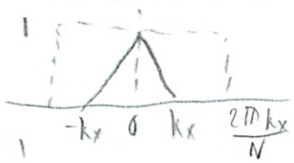
$$= y[n] \cdot \left[\frac{1}{2} e^{jk_c\frac{2\pi}{N}n} + \frac{1}{2} e^{-jk_c\frac{2\pi}{N}n}\right]$$

$$= \left[\frac{1}{2}\sum a_k e^{j(k+k_c)\frac{2\pi}{N}n} + \frac{1}{2}\sum \cdots\right] \cdot \left[\frac{1}{2}e^{+\frac{1}{2}e}\right]$$

$$= \frac{1}{4}\sum_{k=-kx}^{kx} a_k e^{j(k+2k_c)\frac{2\pi}{N}n} + \frac{1}{2}\sum_{k=-kx}^{kx} a_k e^{jk\frac{2\pi}{N}n} + \frac{1}{4}\sum_{k=-kx}^{kx} a_k e^{j(k-2k_c)\frac{2\pi}{N}n}$$

discard → keeping → discard

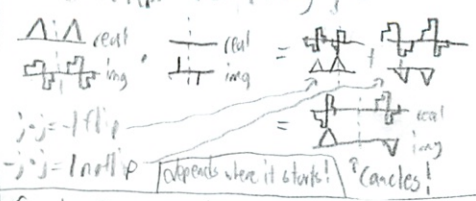
Then discard other pieces w/ LPF
And double (gain) on center piece



Done!
- this was perfect case

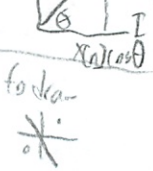
$e^{j\phi} = \cos(\phi) + j\sin(\phi)$
$\cos(\phi) = \frac{1}{2}e^{j\phi} + \frac{1}{2}e^{-j\phi}$
$\sin(\phi) = \frac{1}{2j}e^{j\phi} - \frac{1}{2j}e^{-j\phi}$

Sin is flipped when $j \cdot j = -1$
Notice flips real / imag positive axis



Can be freq or phase error
$\cos((\Omega+\varepsilon)n)$ $\cos(\Omega n + \varepsilon)$
results in amp scaling of $\cos(\varepsilon)$
delays look like phase errors

To fix: quadrature demodulation
- demodulate cos, sin seperatly
- $w[n] = I[n] + jQ[n]$
$= \sqrt{I[n]^2 + Q[n]^2}$



DFSL This was thing where it needs to learn line during training
or look at the differential

More complex - can have whole constellation w/ diff radii

Networks global multi-hop networks
- reliable, scalable, performance, cost, security
redundent, degregation, not failure
abstract away layers
simplex - one way communication
Half-duplex - bi directional, but one at a time
Full-duplex - simultaneous two-way comm

Fully Connected every point 1 to 1
Throughput $O(N^2)$ Latency $O(1)$ Cost $O(N^2)$

Star one central node - high prone to failure
Throughput $O(N)$ Latency $O(1)$ Cost $O(N)$

Circuit vs Packet switching
Time Division Multiplexing - reserve you a slot
C bit/sec channel, each R bits/sec
So C/R # of slots - bad for bursts
So go packet switching - best effort delivery
- good at bursty traffic
- best routing changes automatically
- have a queue

Little's Law N queue size T delay

$$N = \lambda D$$

N: mean # packets in queue
λ: rate P/T
D: mean delay per packet T/P
A/T over time

A = area under curve
T = time looking at

Say ammusement park
N = # people in line
λ = # people ride takes every minute
D = wait time for each rider

Depends where you define system
Line or Line + Ride
depends what you want to look

Want to find the min cost route; DV vs LS
in both periodic "HELLO" packets - 1 hop
to get list of neighbors
2. Advertise 3. Integrate

DV Sends its routing table (final dest, cost) to its immediate neighbors

Bellman-Ford (Integration)
For each entry recieved, add in used link cost
Are we currently using that neighbor?
- if yes update cost
- if no, if lower cost, update table

So routing table is
- for each dest, next link and total cost

if link breaks, ∞ costs should propagate
- as updates that neighbor
- if Hello says that link has failed

Look for next step lists for next links, costs
Sometimes get routing loop as packets sent back+forth till reach ∞ threshold

Path vector send entire path, So can check for duplicate entries

as packets go hop by hop figure out next best place to go

Shortest path X→Y via Z, There must be a shortest path X→Z
Proof on induction of # walks on
Converge time: proportional to (Shortest path / largest # hops considering all min cost paths)

LS

- sends info about its links to its neighbors
- (neighbors, link cost) to them
- Not DVs final dest, cost to final dest
- if adj has higher seqnum than before, forward it along all links

Each node should have every packet's adj

Then run Dijkstra's

- initially nodeset with all nodes
- have a table of costs and table of paths like before
- Find nodeset w/ smallest sp cost → call u
  - remove from nodeset
  - For v in u's neighbors
    - d = spcost(u) + cost(u,v)
    - if d < spcost(v) ← previously stored
      - spcost(v) = d
      - routes[v] = routes[u] ∪ link
        (note routes is dict of links that this node must take next for each dest)

↑ repeat, neg

Complexity: N = # nodes, L = # links

Finding u (N times): linear O(N), heap O(log N)

Updating spcost O(L) since each link 2x

O(N² + L) or O(N log N + L)

Same when packet forwarded next link and then next link decides where to send. But LS means it is stated in right dir.

Done in a Hierarchy so it scales

kilo	10^3	centi	.01
mega	10^6	milli	.001
giga	10^9	micro	10^{-6}
tera	10^{12}	nano	10^{-9}

Packets may become lost on the way
So sometimes need to retransmit
may also arrive out of order, duplicate
will need to fix

Stop + wait - Simple version

- Send packet
- when Ack received, send next
- if no Ack in timeout window, retransmit
- Choose timeout so no spurious retx
- would be RTT if things were perfect
- but are highly random
- So Chebychev $P(|X-\mu| \geq k\sigma) \leq \frac{1}{k^2}$
- pick k, so small prob
- Smooth it w/ SRTT

$SRTT[n] = \alpha\, RTT[n] + (1-\alpha)\, SRTT[n-1]$

α is .125 in TCP, use $.1 \leq \alpha \leq .25$

- Can also use deviation

$dev[n] = |RTT[n] - SRTT[n-1]|$

$SRTTDev[n] = \beta\, dev[n] + (1-\beta)\, SRTTDev[n-1]$

- $timeout[n] = SRTT[n] + k \cdot SRTTdev[n]$ (↑ k=4 for TCP)
- to have low tail for spurious Retx

Throughput = 1/T ← time b/w sucessful deliveries

T = RTT if things are perfect

Prob packet is lost $L = 1-(1-p)^N$ ← # links

So when message ← (p = pkt loss per link)

$T = (1-L)RTT + L(timeout + T)$

$= RTT + \frac{L}{1-L} timeout = RTT + \frac{P(loss)}{P(access)} \cdot RTO$

Suppose RTT same for every packet so timeout = RTT

$T = RTT + \frac{L}{1-L} RTT = \frac{1}{1-L} RTT$

$Throughput = \frac{1-L}{RTT} = \frac{(1-p)^N}{RTT}$

So max throughput = 100%!

$RTT = \underbrace{\sum \frac{S}{R}}_{\text{data}} + \underbrace{\sum \frac{k}{R}}_{\text{Ack}}$ n = # hops, R = rate of channel, S = size data in bits, k = size ack in bits

$Utilization = \frac{Throughput}{Max\ Throughput\ o\ bitrate} = \frac{data\ rate}{link\ rate}$

$Max\ throughput = \frac{1}{time\ 1\ packet} = \frac{1}{\frac{S}{R}} = \frac{R}{S}$

Watch kilo, mega byte/bit!

Rate = $\frac{bits}{time}$ How long to transmit x bits in X seconds?

trans for sucessful delivery = $\frac{1}{1-L}$

Improve w/ Sliding Window

- allow W packets "in the air" at once
- not really a window - I think
- if unacked list < W, transmit!
- Each packet has its own timestamp, for timeout
- Reciever has buffer so packets delivered in order

Want W just right

- too small: low throughput
- too large: queues fill - lots of latency
- Set $W = B \cdot RTT_{min}$ (B = rate of slowest link - in packets/sec!)
- use RTT_{min} - not including queue delays since feedback loop would ↑ W to ∞ as queue lengths ↑
- Or slightly longer so busy even w/ losses

$Throughput = \frac{W}{RTT} = \text{effective departure rate} \quad (Really = \min(\frac{W}{RTT}, B))$

- limited by B = rate of slowest link
- Find by ↑ W till packets dropping

Can double window size		depends what limiting factor is
Cut RTT/2	↑W	
Double B	RTT/2	
	2B	

if p link fails = p_i

Then $1 - \prod_{i=1}^{N}(1-p_i) = P(\text{packet fails})$ ← have already

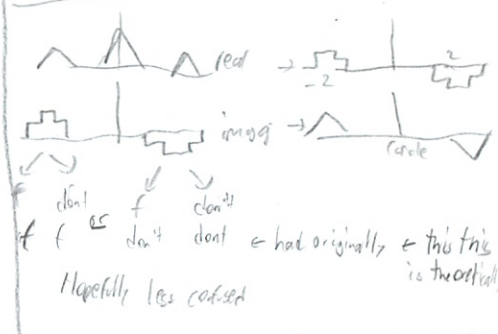
If W set ? W supposed to be $(B \cdot RTT_{min})$

Then queue is needed

queue delay = $\frac{W_{set} - W_{supposed}}{B = \text{bottleneck link rate}}$

$RTT = RTT_{min} + \text{queue delay}$

$Throughput = \frac{W}{RTT} = \text{same as } \frac{W_{suppost}}{RTT_{min}} = B!$ (note just B!)



if f or ← had originally, ← this fails is theoretically right

Hopefully less confused

Information - ↓ uncertainty

Info content $\log_2 \frac{1}{p(\text{seq})}$ in bits

Expected info content

$$H(X) = E(I(X)) = \sum_{i=1}^{n} p(x_i) \log_2 \left(\frac{1}{p(x_i)}\right)$$

If all pi's = uniform, $\frac{1}{N}$

$\log_2(N)$

Outcome reduced to M possible choices

Entropy after receipt

$\frac{1}{n} \log_2\left(\frac{1}{n}\right) = \log_2 M$

Entropy in message is change

$H_{before} - H_{after} = \log_2(N) - \log_2(M)$

$= \log_2\left(\frac{N}{M}\right)$ ← original # choices; ↑ # of choices left!

Example: Have 52 cards, tell you it's a ♢

So $\log_2\left(\frac{52}{13}\right) = 2$ bits info

I gave you ↑ assuming all prob =

Or add $\log_2\left(\frac{1}{12+12+12}\right)$ ← remaining probs

Additive

Fixed is easiest

Variable saves space - max down to entropy

Huffman coding - optimal

Compute avg length of code

$\sum P_{symbol}(L_{symbol})$

$p \approx \frac{1}{2^k}$

Log the power to which the base must be raised to to produce that #

$\log_2 16 \rightarrow 2^x = 16$

$\log(xy) = \log(x) + \log(y)$

$\log_b(x^p) = p \log_b x$

$e^{\ln(x)} = x \quad \ln(e^x) = x$

$\log_b\left(\frac{x}{y}\right) = \log_b(x) - \log_b(y)$

$\log_b(\sqrt[p]{x}) = \frac{\log_b(x)}{p}$

$\log_b(x) = \frac{\log_k(x)}{\log_k(b)}$

6.02 #1

Clock Recovery: The constant adjust (forward) backwards

Sample middle one

Other Stuff

How many samples/bit?

Where does byte start?

8b/10

1. Lots of bit transitions
2. DC balance 0s, 1s
3. Special sync symbol

$x[n]$ = input

$y(n)$ = output

$U[n]$ = unit step

$S[n]$ = unit step response

$\delta[n]$ = unit sample

$h[n]$ = unit sample response/channel description (fading coefficient)

break everything down to unit samples

Time invariant $x[n-N] \rightarrow y[n-N]$

Linear $ax_1[n] + bx_2[n] \rightarrow ay_1[n] + by_2[n]$ (weighted sums in → out)

Convolution (allows deconvolution)

- commutative $x[n] * h[n] = h[n] * x[n]$
- associative $x[n] * (h_1[n] * h_2[n]) = (x[n] * h_1[n]) * h_2[n]$
- distributive $x[n] * (h_1[n] + h_2[n]) = x[n]h_1[n] + x[n]h_2[n]$

Parallel ⊕

Series ⊗

Just a product of what came before

Just clearest point - not when transition is

Causal - depends only on current + previous values

Scalar - real #, not vector

$U[n] = S[n] - S[n-1]$

FSI

$B = \left\lceil \frac{\text{length } h[n] \text{ at } 0}{N} \right\rceil + 2$

test pattern $2^N \cdot B$

pick samples/bit

Diff than fast/slow channel

deconvolution

$w[n] = \frac{1}{h[0]}\left(y[n] - w[n-1]h[1] + \cdots\right)$

Stability

$\sum_{m=1}^{k} \left|\frac{h[m]}{h[0]}\right| < 1 \quad \sum_{m=1}^{k} h[m] < h[0]$

drop first $h[0]$ if is or close to 0

Noise

Mean $\mu_x = \frac{1}{N}\sum_{n=1}^{N} x[n]$

$P_x = \frac{1}{N}\sum_{n=1}^{N} x[n]^2 \quad \hat{P}_x = \frac{1}{N}\sum_{n=1}^{N}(x[n] - \mu_x)^2$

$E_x = \sum_{n=1}^{N} x[n]^2 \quad \hat{E}_x = \sum_{n=1}^{N}(x[n] - \mu_x)^2$

$SNR = \frac{\hat{P}_{signal}}{\hat{P}_{Noise}} \quad SNR(db) = 10\log\left(\frac{P_{signal}}{P_{noise}}\right)$

stationary vs ergodic random processes

$P(x_1 \le X \le x_2) = \int_{x_1}^{x_2} f_X(x)\,dx$

$\mu_X = \int_{-\infty}^{\infty} x f_X(x)\,dx$

$\sigma^2 = \int_{-\infty}^{\infty} (x - \mu_X)^2 f_X(x)\,dx$

PDF normal $= \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)}$

CDF - $erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\,dt$

$\phi_{\mu,\sigma}(x) = \phi\left(\frac{x-\mu}{\sigma}\right)$

BER bit error ratio

$$\mu_{Ynf} = \frac{1}{N}\sum_{n=1}^{N} Y_{nf}[n] = \frac{1}{N}\frac{N}{2} = \frac{1}{2}$$

$$\tilde{P}_{Ynf} = \frac{1}{N}\sum_{n=1}^{N}\left(Y_{nf}[n] - \frac{1}{2}\right)^2 =$$

$$= \frac{1}{N}\sum_{n=1}^{N}\left(\frac{1}{2}\right)^2 = \frac{1}{N}\frac{N}{4} = \frac{1}{4}$$

$$P(\text{error}) = P(0)\cdot P(\text{error}/\text{trans}0) + P(1)P(\text{error}/1)$$
$$= .5\cdot\phi(-.5/\sigma) + .5\cdot\phi(-.5/\sigma)$$
$$= \phi(-.5/\sigma)$$

$$SNR(db) = 10\log\left(\frac{\tilde{P}_{signal}}{\tilde{P}_{noise}}\right) = 10\log\left(\frac{.25}{\sigma^2}\right)$$

Eye diagram

all possible voltage sequences in a certain # of bits

If have a channel that recieves more 1s make it more likely to recieve a 1

Find ϕ w/ test signals

$$B = \left[\frac{n}{N}\right] + 1$$

Deconvolution w is estimated x

$$y[n] = h[0]w[n] + h[1]w[n-1] + \dots$$

$$w[n] = \frac{y[n] - (h[1]w[n-1] + h[2]w[n-2] + \dots)}{h[0]}$$

$$w[0] = \frac{y[0]}{h[0]}$$

$$w[1] = \frac{y[1] - h[1]w[n-1]}{h[0]}$$

$$w[2] = \frac{y[2] - (h[1]w[n-1] + h[2]w[n-2])}{h[0]}$$

Encode LZW

initalize TABLE[0 to 255] = code for individual bytes
STRING = get input signal
while there are still input symbols:
  SYMBOL = get input symbol
  if STRING + SYMBOL is in TABLE
    STRING = STRING + SYMBOL
  else:
    output the code for STRING
    add STRING + SYMBOL to TABLE
    STRING = SYMBOL
Output the code for STRING

Decode LZW

initalize TABLE[0 to 255] = code for individual bytes
CODE = read next code from encoder
STRING = TABLE[CODE]
Output STRING

while there are still codes to recieve:
  CODE = read next code from encoder
  if TABLE[CODE] is not defined:
    ENTRY = STRING + STRING[0]
  else:
    ENTRY = TABLE[CODE]
  output ENTRY
  add STRING + ENTRY[0] to TABLE
  STRING = ENTRY

SNR/BER

Mean $\mu_x = \frac{1}{N}\sum_{n=1}^{N} x[n]$

Power $P_x = \frac{1}{N}\sum_{n=1}^{N} x[n]^2$

often factor mean out $\hat{P}_x = \frac{1}{N}\sum_{n=1}^{N}(x[n]-\mu_x)^2 = \sigma^2 =$ var

Energy $E_x = \sum_{n=1}^{N} x[n]^2$

$\hat{E}_x = \sum_{n=1}^{N}(x[n]-\mu_x)^2$

$SNR = \frac{\hat{P}_{signal}}{\hat{P}_{noise}}$ in db $= 10\log\left(\frac{\hat{P}_{signal}}{\hat{P}_{noise}}\right)$

Stationary - not affected by shifts

Ergodic - " " " time

Find P_x, then P_{noise}

No noise $P_N = 0$ · $SNR = \infty$ BER = 0

Noise $\sim \sigma N + \mu$

$P(Noise < x) = P(\sigma N + \mu < x) = \Phi\left(\frac{x-\mu}{\sigma}\right)$

Sometimes $\mu_x = \frac{1}{2}\cdot 0 + \frac{1}{2}\cdot 1 = \frac{1}{2}$

$\sigma_x^2 = \frac{1}{2}\cdot 0^2 + \frac{1}{2}\cdot 1^2 - \left(\frac{1}{2}\right)^2 = \frac{1}{4}$

$BER = P(noise > .5 \mid 0)\cdot P(0) + P(< .5 \mid 1)P(1)$

$= \Phi\left(-\frac{\frac{1}{2}}{\sigma}\right)\cdot\frac{1}{2} + \frac{1}{2}\Phi\left(-\frac{\frac{1}{2}}{\sigma}\right)$

$= \Phi\left(-\sqrt{SNR}\right)$

$SNR(db) = 10\log\left(\frac{P_{signal}}{\sigma^2}\right)$

- So for BER Get σ^2 from here, plug from SNR

Intersymbol Interference ISI

- Eye diagram
  - all possible cases on top of each other

- for BER $P(1)P(\text{error}\mid 1) + P(1)P(\text{error}\mid 0) + \text{etc}$

$= \Phi\left(\frac{\text{midVro - voltage it will be}}{\sigma}\right)$

- V_{TH} to minimize error rate

6.02 Quiz 2

Ways to avoid errors

Packet = {#, msg, chk}

Hash to detect error

Checksum - Just add up bits

- easy for one error to offset another

$P(>1 \text{ error}) = 1 - P(\text{no error}) = 1-(1-BER)^k$

$P(>2 \text{ errors}) = 1 - P(\text{no error}) - P(1 \text{ error})$

$= 1-(1-BER)^k - k\cdot BER\cdot(1-BER)^{k-1}$

Hamming Distance = # digit positions wrong

D = min hamming dist - can correct $\left\lfloor\frac{D-1}{2}\right\rfloor$

Single error codes

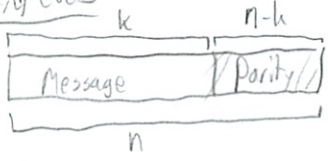
$E_0 = B_0 \oplus B_1 \oplus B_3 \oplus P_0$

$E_1 = B_0 \oplus B_2 \oplus B_3 \oplus P_1$

$E_2 = B_1 \oplus B_2 \oplus B_3 \oplus P_2$

(should all be 0s $\Rightarrow (n+1, n, 2)$

(n,k,d) Code



d = Min Hamming distance can correct

k/n = Code rate, ≤1, larger better

Rectangular — Need $n \le 2^{n-k} - 1$

B_0	B_1	P_0
B_2	B_3	P_1
P_2	P_3	

$\in (8,4,3)$

Replicating $(n,1,r)$ Good for code w/ Hamming = 2 = d

SECC $(n, n-p, 3)$ $n = 2^p - 1$ for $p > 1$

Interleaving for burst errors

- Needs framing to know where to start
- 8b10b works

Reed-Solomon

- the matrix, Galois field thing
- Common (255, 223)
- (n,k) solves up to $(n-k)/2$ errors

Convolutional Codes

4/10

- always a fn of what came before

$P_i[n] = \left(\sum_{j=0}^{k-1} g_i[j]\, x[n-j]\right) \bmod 2$

g_i = k-element generator polynomial for parity bit P_i

- either 1 or 0
- given

- interleave data parity for rate = 1/2

$= P_0[0]\,P_1[0]\,P_0[1]\,P_1[1]\ldots$

message never transmitted

- like a SM or Trellise (over time)
- look for min hamming for most likely then trace backward

Hard decision - digitized 1 or 0

Soft decision

For each pt - like OD $(V_{P_0}-0)^2 + (V_{P_1}-0)^2$ no $\sqrt{}$!

k = # places using a bit - higher better

Protocols for Multi User

- high utilization $U = \frac{\text{throughput all nodes}}{\text{max data rate of channel}}$
- fairness (goals)
- bounded wait
- Scalability

$F = \frac{\left(\sum_{i=1}^{N} x_i\right)^2}{N\sum_{i=1}^{N} x_i^2}$ $\quad \frac{1}{N} \le F \le 1$

TDMA - just give each person a slot

Aloha - p send packet

$U_{slotted} = \underbrace{N p (1-p)^{N-1}}_{P(\text{only 1 sender})}$

$p_{best} = \frac{1}{N}$ leads to $U = \frac{1}{e} \approx 37\%$

Can't p if collission

- if half = 2^{-k} = binary exp back off

but bound $p = \max(p_{min}, p/2)$

$p_{min} \ll 1/\max(N)$

$U_{slotted} \ge \frac{N p (1-p)^{k(2T-1)N-1}}{1/t} = \frac{tNp(1-p)^{\cdots}}{\exp[2T-1]N-1}$

with large N — w/ large N, T

$U_{max} \approx \left(\frac{T}{2T-1}\right)\frac{1}{e}$ — $U_{max} \approx \frac{1}{2e}$

$V_{max} = \frac{T}{2T-1}\left(1 - \frac{1}{(2T-1)N}\right)^{(2T-1)/N}$

(unslotted)

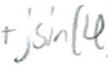
Carrier sense - even better
- need random wait time after backoff
  - inside a contention window

CDMA - using orthogonal vectors

Freq Division Multiplexing

- Seq must be periodic (repeat)
  $x[n] = x[n+N]$
- in N samples - fundamental freq $= \frac{2\pi}{N}$
- harmonics possible $k \cdot \frac{2\pi}{N}$
- negative means go other direction
- complex exponential

$e^{j\varphi} = \cos(\varphi) + j\sin(\varphi)$



$\cos(\varphi) = \frac{1}{2}e^{j\varphi} + \frac{1}{2}e^{-j\varphi}$

$\sin(\varphi) = \frac{1}{2}e^{j\varphi} - \frac{1}{2}e^{-j\varphi}$

When $\varphi = 0$

$e^{j0} = \cos(0) + j\sin(0)$

$= 1 + 0j$

$\varphi = \pm\pi$

$e^{j\pi} = e^{-j\pi} = -1$

$e^{j\pi n} = e^{-j\pi n} = (-1)^n$

Summing

$$\sum_{n=\langle N\rangle} e^{jk\frac{2\pi}{N}n} = \begin{cases} N & k = 0, \pm N, \pm 2N, \ldots \\ 0 & \text{otherwise} \end{cases}$$

Discrete-Time Fourier Seq

$x[n] = \sum_{k=\langle N\rangle} a_k e^{jk\left(\frac{2\pi}{N}\right)n}$ (Synthesis: freq → time)

k is over range of ints
- 0 for $0 \le \text{freq} < 2\pi$
- $-N/2$ for $-\pi \le \text{freq} \le \pi$

a_k = spectral coefficient (complex)

$a_k = \frac{1}{N}\sum_{n=\langle N\rangle} x[n] e^{-jk\left(\frac{2\pi}{N}\right)n}$ (Analysis: time to freq)

= kth spectral coefficient

↑ = index = # cycles in N samples

Need to truncate/limit to a freq
- loses fine detail, like an avg

$H(e^{j\Omega}) = \sum_m h[m] e^{-j\Omega m}$ (channel response)

↑ calc for a certain Ω

$\delta[n] \to \boxed{H(e^{j\Omega})} \to h(n)$ (unit sample response)

unit sample, periodic N timesteps

$a_k = \frac{1}{N}x[0]e^{-jk\frac{2\pi}{N}0} = \frac{1}{N}$

$h[n] = \sum_{k=\langle N\rangle} H\left(e^{jk\frac{2\pi}{N}}\right)e^{jk\frac{2\pi}{N}n}$

Finding the 0s at $\pm\varphi$

$H(e^{j\Omega}) = (e^{j\Omega})^2 - 2\cos(\varphi)(e^{j\Omega}) + 1) = 0$

$h[0] = 1 \quad h[1] = -2\cos(\varphi) \quad h[2] = 1$ (for these, 0 elsewhere)

convolution in time domain, multiplication in freq domain

poor man's low pass = convolve a bunch of 0s together

$x\alpha \to \square \to y\alpha$ — Random

$a_k e^{j\Omega kn} \to \square \to a_k H(e^{j\Omega k}) \cdot e^{j\Omega kn}$

Ω is the band you are assigned

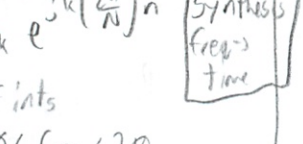
all on 1 channel

have 0s at start + end to bound

$\Omega = 2\pi\frac{f}{f_s} = 2\pi\frac{k}{N}$ (k = spectral coef, Ω = angular freq; signal freq cycles/sec, f_s = sample freq samples/sec)

Filter to only certain freq
- as an LTI channel - choose $h()$
- want 0 at many pts
- convolve many pts together

$h[n] = \delta[n] - \sqrt{2}\delta[n-1] + \delta[n-2]$



$H(e^{j\Omega}) = e^0 - \sqrt{2}e^{-j\Omega} + e^{-2j\Omega}$

$= 1 - \sqrt{2}\cos\Omega - \sqrt{2}\sin\Omega j + \cos 2\Omega + \sin 2\Omega j$

$= 1 - \sqrt{2}x + x^2$

if find 0, set = to 0, find x

$x = \cos\varphi + \sin\varphi$, find φ

$C(\sqrt{2}\text{ here}) = \frac{1 + e^{-2\Omega j}}{-e^{-\Omega j}}$

---

cos

$a_p = e^{\varphi}\cdot\frac{1}{2} + \frac{1}{2}e^{-\varphi}$ for $0 \le p \le N-1$

or $\left[-\frac{N}{2}\right] \le p \le \left[\frac{N}{2}\right]$

$\cos\left(p\left(\frac{2\pi}{11}\right)n\right)$ for $-5 \le p \le 5$

works out to $a_k = \begin{cases} 1/2 & k = \pm r \\ 0 & \text{otherwise} \end{cases}$

sin

$a_p = \frac{j}{2} + e^{\varphi} = \frac{j}{2} + \cos\varphi + \sin\varphi j$

$a_{-p} = -\frac{j}{2} + e^{\varphi} = -\frac{j}{2} + \cos\varphi + \sin\varphi j$

So $x[n] = 1 + 2\cos\left(3\frac{2\pi}{11}n\right) - 3\sin\left(5\frac{2\pi}{11}n\right)$

$a_{\pm 3} = 2\cdot\frac{1}{2} = 1$ (cos)

$a_{-5} = -3\frac{j}{2} = -1.5j$

$a_5 = -3\cdot\frac{-j}{2} = 1.5j$

$a_k = 0$ otherwise

---

states on state diagram $= 2^{k-1}$

$P(\text{noise} > 7.5) = P(\sigma N > 7.5) = P\left(N > \frac{7.5}{\sigma}\right)$

$= P\left(N < -\frac{5}{\sigma}\right) = \Phi\left(-\frac{1.5}{\sigma}\right) = 1 - \Phi\left(\frac{1.5}{\sigma}\right)$

(n,k) has at most 2^{n-k} patterns with min $d = 2t+1$ can correct t errors

k = Constraint length