STANFORD

Lecture 18

IC and Conclusion

June 4, 2024

JOHN M. CIOFFI

Hitachi Professor Emeritus of Engineering

Instructor EE379B – Spring 2024

Announcements & Agenda

- Announcements
  - Final is Friday at 2pm, get email from Helen helen.niu@stanford.edu
  - Due Saturday at 3pm.
  - Course evaluation link is open at [eval]()
  - Solutions go up early tomorrow morning (Wed).

- Agenda
  - CIC Optimization (from Thursday)
  - LIC Design
  - Research & Machine-Learning/AI Challenges (optional)

CIC Optimization

CIC's "Optimum" Spectrum Balancing (no GDFEs)

- **OSB** minimizes weighted energy sum for given $\boldsymbol{b}_{min}$.
  - There is no crosstalk cancellation

$$\min_{\{R_{\boldsymbol{xx}}(u,n)\}} \quad \sum_{u=1}^{U} w_u \cdot \mathcal{E}_u$$
$$ST: \quad \text{for } \; u = 1, \ldots, U$$
$$0 \le \sum_n \text{trace}\left\{R_{\boldsymbol{xx}}(u,n)\right\} \le \mathcal{E}_{u,max}$$

- OSB relates $\boldsymbol{b}$ to $R_{\boldsymbol{xx}}(u,n)$ by:

$$b_u = \sum_n \log_2 \frac{\left| H_{uu,n} \cdot R_{\boldsymbol{xx}}(u,n) \cdot H^*_{uu,n} + \mathcal{R}_{noise}(u,n) \right|}{\left| \mathcal{R}_{noise}(u,n) \right|}$$

- $\mathcal{R}_{noise}(u,n) = I + \sum_{i \neq u} \widetilde{H}_{u,i,n} \cdot R_{\boldsymbol{xx}}(i,n) \cdot \widetilde{H}^*_{u,i,n}$.
  - **All** other users are crosstalk-noise additions.

- Tonal Lagrangian for minPOSB
  - Minimize each individually, and sum.
  - It's not convex (no sequential-differences' transformation).
  - The $\boldsymbol{w}$ is determined as an output of optimization.
  - **Optimization can also maximize negative for the maxROSB.**

$$L_n(R_{\boldsymbol{xx}}(u,n), \boldsymbol{b}_n, \boldsymbol{w}, \boldsymbol{\theta}) = \sum_{u=1}^{U} w_u \cdot \mathcal{E}_{u,n} - \theta_u \cdot b_{u,n}$$

Still has minimum, "integer programming"

- $SNR(u,n) = \frac{|\tilde{H}_{u,u,n} \cdot R_{xx}(u,n) \cdot \tilde{H}^*_{u,u,n}|}{|R_{noise}(u,n)|}$

$$b_u = \sum_n \log_2 \left(1 + \frac{\mathrm{SNR}(u,n)}{\Gamma}\right)$$

- Partition energy range or use discrete integer bit quanta for scalar case:
  - These are integer-programming part.

$$M = \frac{\max_u \boldsymbol{\mathcal{E}}(u)}{\Delta\mathcal{E}} \quad \text{or} \quad b = 0,1,\dots,bitcap$$

- **Energy or bit step:** For each tone, search M^U possible energies (or $2^{U \cdot bitcap}$ bit quanta) to minimize tonal Lagrangian and add these tonals.
  - The users' energies are weighted, and the weights are optimized (compute $\mathcal{L}$ update to make small),
  - to ensure that the energy constraint is met.
  - Or equivalently the $\boldsymbol{\theta}$ **is similarly adjusted if** $\boldsymbol{w}$ **is given.**

- **Constraint: External to energy-bit step,** Use a descent method to update the $\boldsymbol{\theta}$ ***or*** $\boldsymbol{w}$ Lagrange multiplier for rate constraints:

$$\mathcal{L} = \sum_{u=1}^{U} \sum_{n=0}^{\bar{N}-1} w_u \cdot \mathcal{E}_{u,n} - \theta_u \cdot b_{u,n}$$

$$\boldsymbol{w} = \boldsymbol{w} + \epsilon \cdot \Delta\boldsymbol{\mathcal{E}} \text{ or}$$
$$w_u \leftarrow {}^{w_u}/_2 \text{ if} < \mathcal{E}_{u,max}$$
$$w_u \leftarrow 2 \cdot w_u \text{ if} > \mathcal{E}_{u,max}$$

$\boldsymbol{w}$ update is for admissibility.

OSB.m (this maximizes weighted rate-sum)

```
 function [S1, S2, b1, b2] = osb(Hmag_sq, No, E, theta, mask, ...
  gap, bitcap, cb)

  osb and also finds w1 energy weight for USER 1
  A. Chowdhery ~2010 ; Updated by J. Cioffi in 2024.  It presently
handles only 2 users, so U=2.

  Inputs

  Hmag_sq is a N x 2 x 2 where N is FFT size. N inferred from this.
  No      is a 1 x U white-noise power spectra density matrix.
     If Hmag_sq is complex BB, then No should be the one-sided PSD.
  E       is a 1 x U energy vector.
  theta   is a 1 x U user-rate weighting vector.
  mask    is an N x U PSD maximum allowed.
  gap     is the (non-dB) linear gap (so 1 if 0 dB gap).
  bitcap  is a 1 x U maximum number of bits allowed per tone.
  cb      is 2 for real baseband and 1 for cplex bband

  Outputs

  S1      is user 1's Nx1 PSD
  S2      is user 2's Nx1 PSD
  b1      is user 1's Nx1 bit distribution
  b2      is user 2's Nx1 bit distribution

calls optimize_l2.m, which calls optimize_s.m
User order is reversed with respect to class convention.
```

Subroutines adjust *w* Values.

```
>> H2=zeros(1,2,2);
H2(:,:,1) = [ 0.6400    0.2500 ]; % note this is squared mag each term
H2(:,:,2) = [ 0.4900    0.3600 ];
>> Noise =  1.0e-04 * [    1.0000    1.0000];
>> Ex =[    1     1];
>> mask =[    1     1];
>> gap =    1;
>> bitcap =[   15    15];
>> [S1, S2, b1, b2] = osb(H2, Noise, Ex, [0.5 .5], mask, gap, bitcap,2)

S1 =    0.6398
S2 =    0
b1 =    6  % note < 6.3 for the GDFE based IC's maximum L11:16
b2 =    0
>> [S1, S2, b1, b2] = osb(H2, Noise, Ex, [0.01 .99], mask, gap, bitcap,2)
S1 =    0
S2 =    0.1419
b1 =    0
b2 =    4.5000 <5.9 for L11:16
```

- The OSB search can be very complex for $U > 3$.
- OSB also can have severe numerical issues (cause it to diverge), even in matlab double precision.

Multitone OSB Example

```
h = cat(3, [1  .8 ;  -1   1], [-.9  -.7 ;  0   1]  )*10;
He = fft(h, 8, 3);
>> H3=zeros(8,2,2);
>> H3(:,1,1)=He(1,1,:); % tone index moves
>> H3(:,2,1)=He(2,1,:); % yes, you could use permute also.
>> H3(:,2,2)=He(2,2,:);
>> H3(:,1,2)=He(1,2,:);
>> Noise=ones(8,2);
>> mask=ones(8,2);
>> Ex=8*Ex;
>> theta = [ .5 .5];
>> [S1, S2, b1, b2] = osb(H3.*conj(H3), Noise, Ex, theta, mask, gap,
bitcap,1);
>> S1' =     0          0        0.7017   0.8272    0   0.8272   0.7017      0
>> S2' =   0.6375   0.7469       0        0         0      0         0        0.7469

>> b1' =    0    0    7    8    0    8    7    0
>> b2' =    8    8    0    0    0    0    0    8
>> sum(b1) =    30
>> sum(b2) =    24
sum(b1+b2) =    54 % < ~116 that MAC, BC, single had for this channel
```

Same 2x2 channel as in L16, except now an IC.

OSB solution is often FDM

- GDFE's cancellation of crosstalk makes a large difference.

- L18 later compares this to IW (like SWF, Except for IC – see last section today).
- We could similar have a minPOSB,
- Or even admOSB.

Searching for more Software on OSB and related.

CIC Optimization

allow multiple-user decoding

minPIC = more "optimum"

- minPIC concept allows for each receiver u to cancel $i \in \mathcal{D}_u(\boldsymbol{\Pi}, p_{\boldsymbol{xy}}, \boldsymbol{b})$; the decodable set.
- Order has been restored 😀.
- The optimization is
  - $(i,u) = (RCVR, USER)$.

Same format, but the energy-to-information relation changes

$$\min_{\{R_{\boldsymbol{xx}}(i,u,n)\}} \sum_{n=0}^{\overline{N}-1} \sum_{u=1}^{U} w_u \cdot \text{trace} \underbrace{\left\{ \sum_{i=1}^{U} R_{\boldsymbol{xx}}(i,u,n) \right\}}_{\mathcal{E}_{u,n}}$$

$$ST: \quad b_u \geq b_{min,u}$$

$$R_{\boldsymbol{xx}}(i,u,n) \succeq \mathbf{0} \ .$$

- $\boldsymbol{\theta}$ still has U terms, and they determine the U! "**sensible" orders $\boldsymbol{\Pi}$**.
- The achievable-region constraint remains convex already (like minPMAC).
- Each receiver may use a GDFE, but precoders are not possible (on IC).

3-User Order example

- Given a $\boldsymbol{\theta}$, say for example with $\theta_3 > \theta_1 > \theta_2$, they determine all receivers' order:

$FOR: \quad \theta_3 > \theta_1 > \theta_2 > 0$

$(RCVR, USER)$	Receiver 3	Receiver 1	Receiver 2
	(1,1),(2,1),(1,2),(2,2)	(2,2),(3,2),(2,3),(3,3)	(1,1),(3,1),(1,3),(3,3)
	(3,3)	(1,3)	(2,3)
	(1,3)	**(3,1)**	(2,1)
	(2,3)	**(1,1)**	**(3,2)**
	(3,1)	**(2,1)**	**(1,2)**
	(3,2)	(1,2)	**(2,2)**

$THESE\ ARE\ constant\ XTALK,\ AND\ CAN\ BE\ 0$

Π

$$\left\{\sum_{u=1}^{3} \theta_u \cdot b_u\right\}_{RCVR3opt} = (\theta_3 - \theta_1)\cdot \log_2(B) + (\theta_1 - \theta_2)\cdot \log_2(C) + \theta_2 \cdot \log_2(D)$$

- Any other order is inconsistent with the Lagrangian multipliers' interpretation.

$$\begin{aligned} A &\triangleq |H_{3,1}|^2 \cdot (\mathcal{E}_{1,1} + \mathcal{E}_{2,1}) + |H_{3,2}|^2 \cdot (\mathcal{E}_{1,2} + \mathcal{E}_{2,2}) + I \\ B &\triangleq |H_{3,3}|^2 \cdot \mathcal{E}_3 + A \\ C &\triangleq |H_{3,1}|^2 \cdot \mathcal{E}_{3,1} + B \\ D &\triangleq |H_{3,2}|^2 \cdot \mathcal{E}_{3,2} + C \end{aligned}$$

RCVR 3

$$\begin{aligned} b_3 &= \log_2(B) - \log_2(A) \\ b_{3,1} &= \log_2(C) - \log_2(B) \\ b_{3,2} &= \log_2(D) - \log_2(C) \end{aligned}$$

Do same for other 2 receivers

- RCVR 1 optimization of rate sum

$$
\begin{aligned}
A &\triangleq |H_{1,3}|^2 \cdot (\mathcal{E}_{2,3} + \mathcal{E}_{3,3}) + |H_{1,2}|^2 \cdot (\mathcal{E}_{1,2} + \mathcal{E}_{2,2}) + I \\
B &\triangleq |H_{1,3}|^2 \cdot \mathcal{E}_{1,3} + A \\
C &\triangleq |H_{1,1}|^2 \cdot \mathcal{E}_{1} + B \\
D &\triangleq |H_{1,2}|^2 \cdot \mathcal{E}_{1,2} + C
\end{aligned}
$$

RCVR 1

$$
\begin{aligned}
b_{1,3} &= \log_2(B) - \log_2(A) \\
b_1 &= \log_2(C) - \log_2(B) \\
b_{1,2} &= \log_2(D) - \log_2(C) \quad .
\end{aligned}
$$

$$\left\{\sum_{u=1}^{3} \theta_u \cdot b_u\right\}_{RCVR1opt} = (\theta_3 - \theta_1) \cdot \log_2(B) + (\theta_1 - \theta_2) \cdot \log_2(C) + \theta_2 \cdot \log_2(D)$$

- RCVR 2 optimization of rate sum

$$
\begin{aligned}
A &\triangleq |H_{2,3}|^2 \cdot (\mathcal{E}_{1,3} + \mathcal{E}_{3,3}) + |H_{2,1}|^2 \cdot (\mathcal{E}_{1,1} + \mathcal{E}_{3,1}) + I \\
B &\triangleq |H_{2,3}|^2 \cdot \mathcal{E}_{2,3} + A \\
C &\triangleq |H_{2,1}|^2 \cdot \mathcal{E}_{2,1} + B \\
D &\triangleq |H_{1,1}|^2 \cdot \mathcal{E}_{2} + C
\end{aligned}
$$

RCVR 2

$$
\begin{aligned}
b_{2,3} &= \log_2(B) - \log_2(A) \\
b_{2,1} &= \log_2(C) - \log_2(B) \\
b_2 &= \log_2(D) - \log_2(C) \quad .
\end{aligned}
$$

$$\left\{\sum_{u=1}^{3} \theta_u \cdot b_u\right\}_{RCVR2opt} = (\theta_3 - \theta_1) \cdot \log_2(B) + (\theta_1 - \theta_2) \cdot \log_2(C) + \theta_2 \cdot \log_2(D)$$

- Six energies repeat – select from each such energy pair, that which has corresponding lowest rate/info.
- Outer $\boldsymbol{\theta}$ loop (e.g., Ellipsoid) remains the same as minPMAC.

Generalize – first order them to simplify

- Create order of users for each of (reordered) users

θ_U	$\dots$	θ_u	$\dots$	θ_1
$U^2 \setminus \{(1:U,U),(U,1:U-1)\}$	$\dots$	$U^2 \setminus \{(1:U,u),(u,1:U-1)\}$	$\dots$	$U^2 \setminus \{(U,1:U),(1,1:U-1)\}$
(U,U)	$\dots$	(u,U)	$\dots$	1,U-1
		$\vdots$		$\vdots$
$\vdots$	$\dots$	$(u,U-u+1)$	$\dots$	$(1,1)$
$(1,U)$	$\dots$	(U,u)	$\dots$	$(U,1)$
$(U,U-1)$	$\dots$	$\vdots$	$\dots$	$\vdots$
$\vdots$	$\dots$	$(1,u)$	$\dots$	$\vdots$
		$(u,U-u-1)$		
$\vdots$	$\dots$	$\vdots$	$\dots$	$\vdots$
$(U,1)$	$\dots$	$(u,1)$	$\dots$	(U,U)

Table 5.2: **Generalized of overall decoding order pairs given descending-order θ.**

Generalize A,B,C, D

$$
\begin{array}{lll}
K_{1,u} & \triangleq \sum_{i \neq u} H_{u,i} \cdot \left(\sum_{j \neq i} R_{\boldsymbol{xx}}(i,j) \right) \cdot H^*_{u,i} + I & \text{for } \; b_{u,U} \\
K_{2,u} & \triangleq H_u(2) \cdot R_{\boldsymbol{xx}}(u, 2U-3, 2) \cdot H^*_u(2^{nd}) + K_{1,u} & \text{for } \; b_{u,U-} \\
\vdots & \vdots & \\
K_{u,u} & \triangleq H_{u,2U-u+1}(2) \cdot R_{\boldsymbol{xx}}(u, 2U-u+1(2^{nd})) \cdot H^*_{u,2U-u+1}(2) + K_{u-1,u} & \text{for } \; b_u \\
\vdots & \vdots & \\
K_{2U-2,u} & \triangleq H_{u,1}(2) \cdot R_{\boldsymbol{xx}}(u, 1(2^{nd})) \cdot H^*_{u,1}(2) + K_{2U-3,u} & \text{for } \; b_{u,1}
\end{array}
$$

$$
\left\{ \sum_{u=1}^{U} \theta_u \cdot b_u \right\}_{RCV\ Ruopt} = (\theta_U - \theta_{U-1}) \cdot \log_2(K_{1,u}) + \ldots + (\theta_2 - \theta_1) \cdot \log_2(K_{2U-3,u}) + \theta_2 \cdot \log_2(K_{2U-2,u})
$$

- This is convex in those quantities optimized
- Need the outer subgradient loop on theta to drive IC rate vector to bmin.

Software awaits writing.

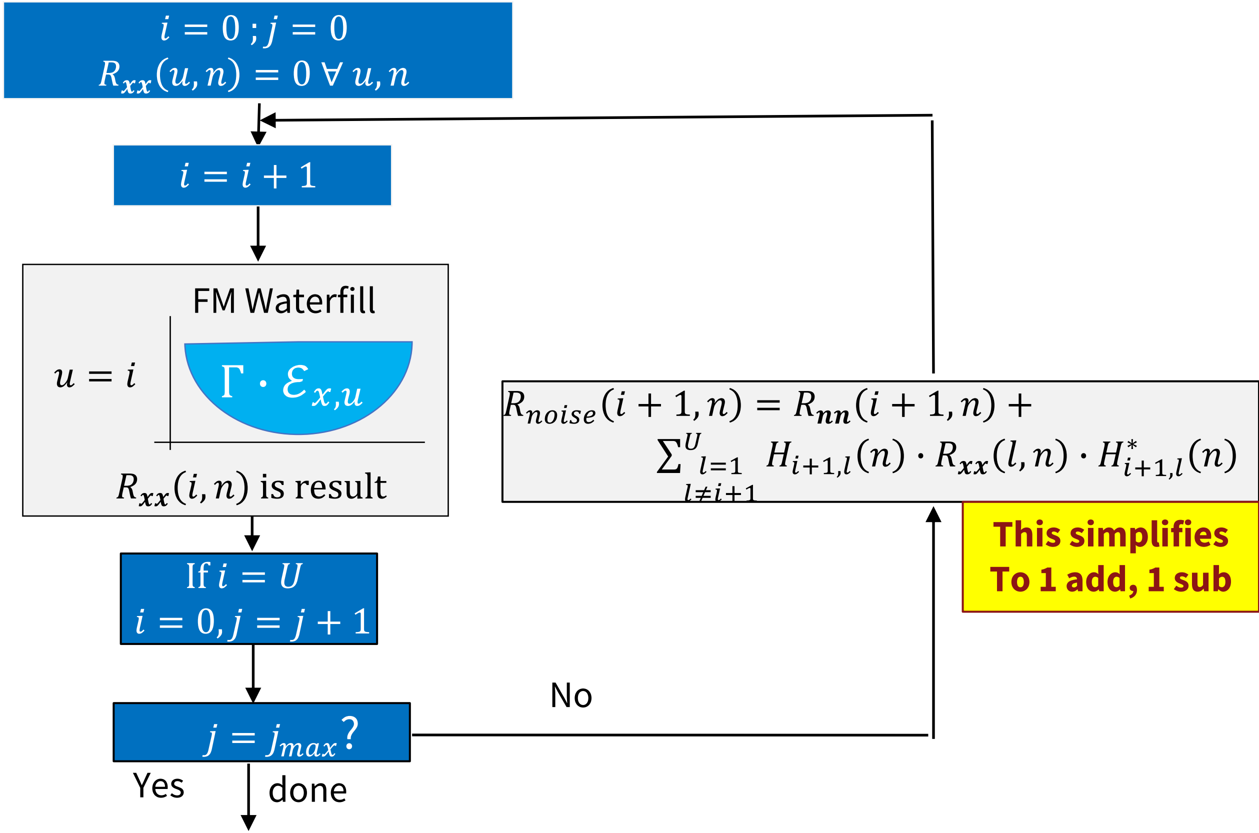
LIC Design

Iterative WF borrows MAC's SWF for the IC

- Rate-sum partial derivatives yield:

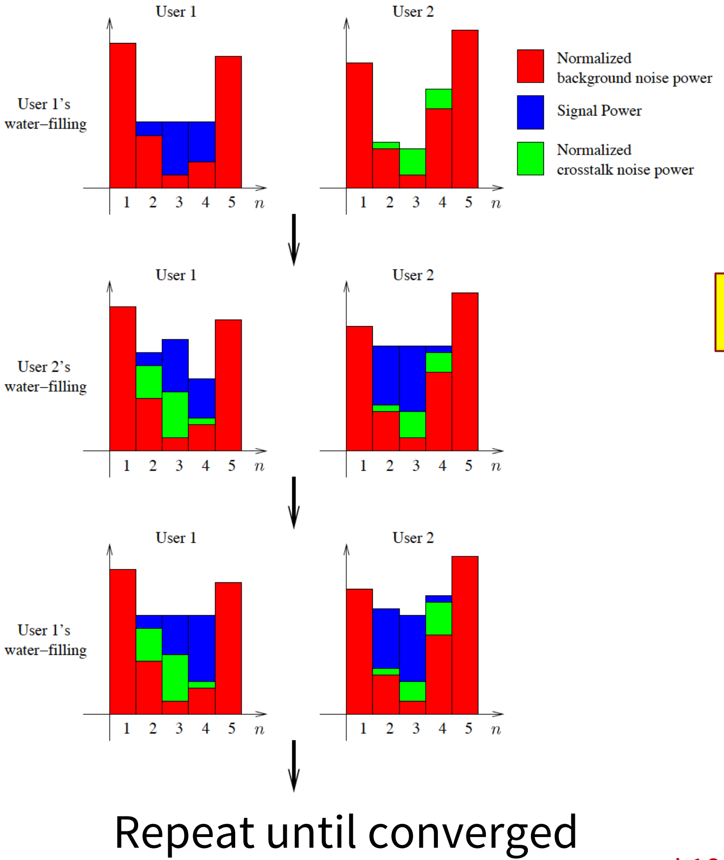
$$\lambda_u = \mathcal{E}_{u,n} + \frac{\Gamma\sigma_n^2 + \sum_{i\neq u} |H_{ui}|^2 \cdot \mathcal{E}_{i,n}}{|H_{uu,n}|^2}$$

- IW converges in practice, and theory,
  - under mild channel conditions (Leshem –Xtalk not outrageous)



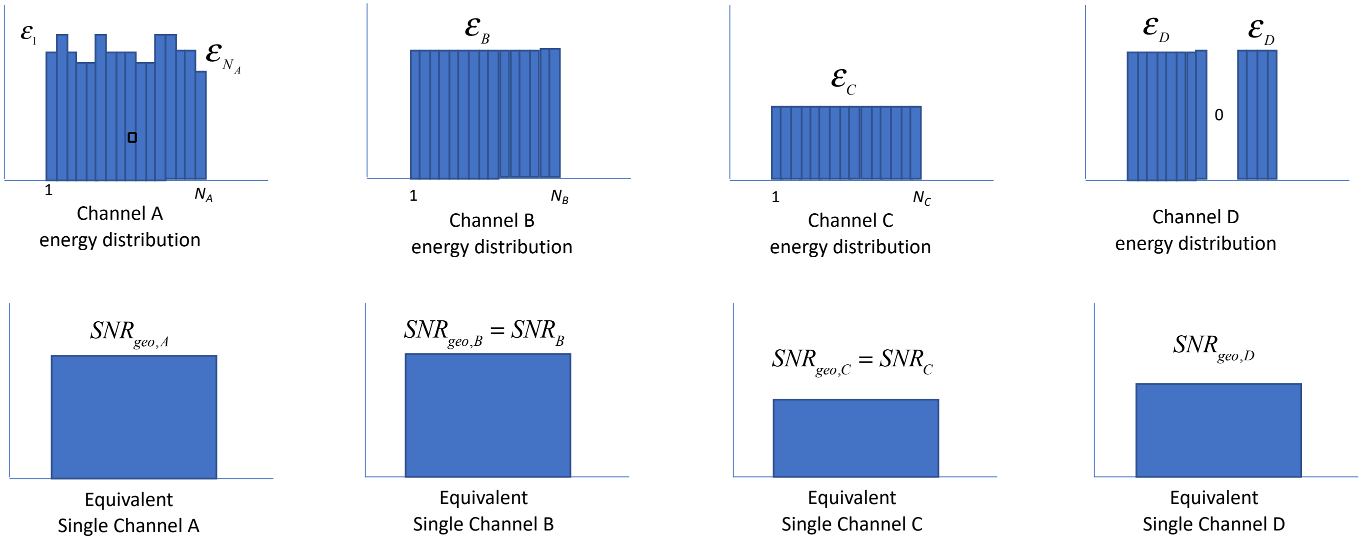
IW Illustrated for the LIC

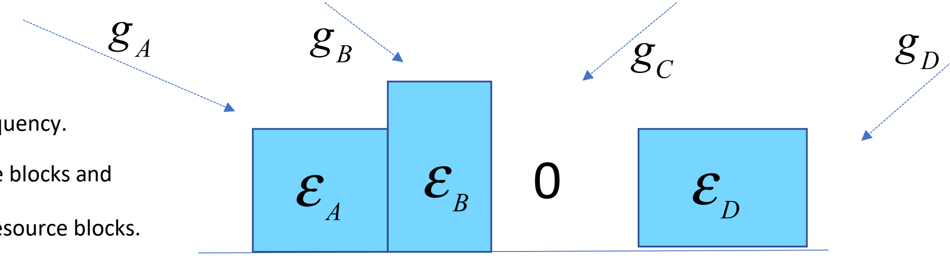
- Each user "reacts" to others.
- Others sense the new xtalk.
- They eventually converge.



"Nash Equilibrium"

Wireless Potential Use (Resource Blocks)





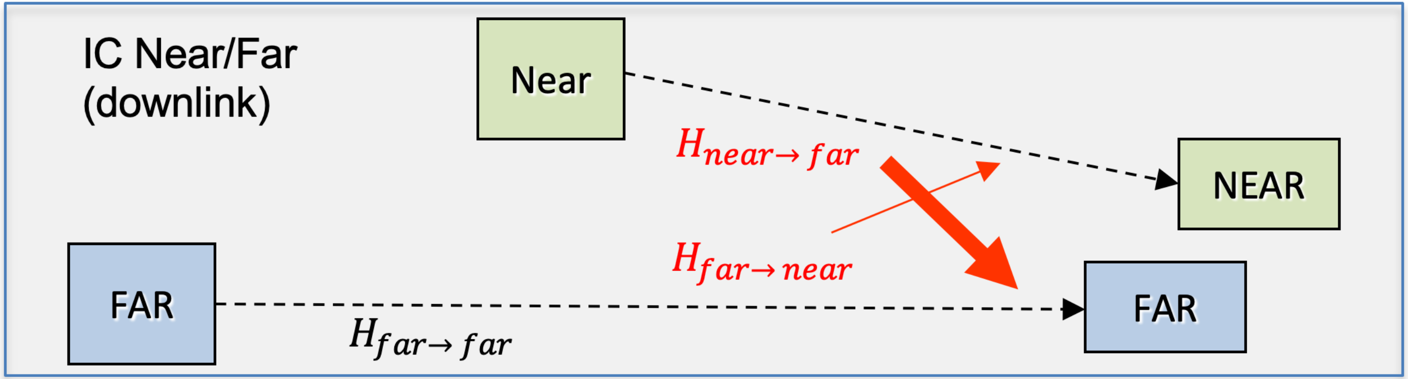
- This could be space and/or frequency.
- It uses C-OFDM within resource blocks and
- Vector-Coding among spatial resource blocks.

Nested loading

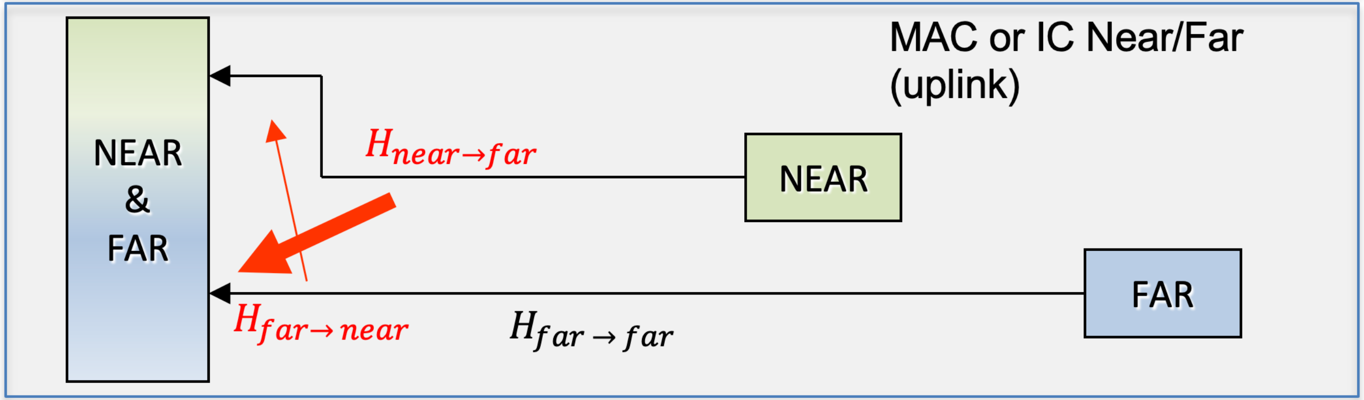
$$\mathcal{E}_X = N_X \cdot \bar{\mathcal{E}}_X \quad \text{for } X = A, B, C, D$$

Near-Far Example

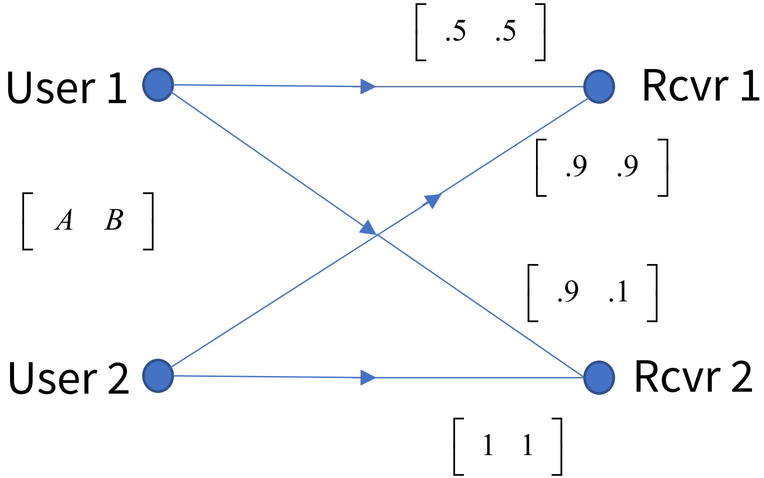
- Downlink has another transmitter for another IC user closer (the "near" user).



- Uplink has another transmitter for another IC user closer (the "near" user).



Example IC with IWF (near-far)



$$\begin{bmatrix} y_{1A} \\ y_{1B} \end{bmatrix} = .5 \cdot \begin{bmatrix} x_{1A} \\ x_{1B} \end{bmatrix} + .9 \cdot \begin{bmatrix} x_{2A} \\ x_{2B} \end{bmatrix} + \begin{bmatrix} n_{1A} \\ n_{1B} \end{bmatrix}$$

$$\begin{bmatrix} y_{2A} \\ y_{2B} \end{bmatrix} = \begin{bmatrix} x_{2A} \\ x_{2B} \end{bmatrix} + [.9 \quad .1] \cdot \begin{bmatrix} x_{1A} \\ x_{1B} \end{bmatrix} + \begin{bmatrix} n_{1A} \\ n_{1B} \end{bmatrix}$$

$$\boxed{\sigma_1^2 = \sigma_2^2 = 0.1 \text{ (noises independent)}}$$

Energy 1A = 2.0

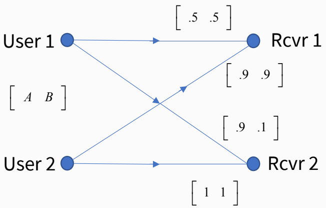
Energy 2B = 2.0

- Which receiver has near-far issue?
  - RCVR 1 in both bands
- Who is near user?
  - User 2

Tabular Tracking of IW

- User 2 reacts to user 1 crosstalk.
- User 1 then counter acts.
- Further reduction of energy on user 2 band B.
- IW here converges in 2 cycles and
  - solution looks like FDM.
- This is better than equal energy on both users in both bands.

```
>> He1
He1(:,:,1) =
   0.2500   0.8100
   0.2500   0.8100
He1(:,:,2) =
   0.8100   1.0000
   0.0100   1.0000
>> [b, E] = iw(2, 2, He1, .1*ones(2,2), [2 2], 0, [0 0], [10 10],1)
b =            N , U          noise      energy    gap
   1.5701   0.1512
        0   4.2555
E =
   2.0000   0.1900
        0   1.8100
```



Same result

Table 2 – Simple IW Example		
	Band A	Band B
User 1	$\mathcal{E}_{1A}=1$	$\mathcal{E}_{1B}=1$
User 2	$\frac{1}{g_{2A}}=.1+(.9)^2=.91$	$\frac{1}{g_{2B}}=.1+(.1)^2=.11$
	$\mathcal{E}_{2A}+.91=\mathcal{E}_{2B}+.11$ $\mathcal{E}_{2A}+\mathcal{E}_{2B}=2$ $\mathcal{E}_{2A}=.6 \quad \mathcal{E}_{2B}=1.4$	
User 1	$\frac{1}{g_{1A}}=\frac{.1+.6\cdot(.9)^2}{(.5)^2}=2.344$	$\frac{1}{g_{1B}}=\frac{.1+1.4\cdot(.9)^2}{(.5)^2}=4.936$
	$\mathcal{E}_{1A}+2.344=\mathcal{E}_{1B}+4.936$ $\mathcal{E}_{1A}+\mathcal{E}_{1B}=2$ $\mathcal{E}_{1A}=2 \quad \mathcal{E}_{2B}=0$	
User 2	$\frac{1}{g_{2A}}=.1+2\cdot(.9)^2=1.72$	$\frac{1}{g_{2B}}=.1+0\cdot(.1)^2=.1$
	$\mathcal{E}_{2A}+1.72=\mathcal{E}_{2B}+.1$ $\mathcal{E}_{2A}+\mathcal{E}_{2B}=2$ $\mathcal{E}_{2A}=.19 \quad \mathcal{E}_{2B}=1.81$	
User 1	Remains $\mathcal{E}_{1A}=2 \quad \mathcal{E}_{2B}=0$ → IW has converged $\frac{1}{g_{1A}}=\frac{.1+.19\cdot.81}{.25}=1.0156$	
Data rates User 1	$\log_2(1+2/1.0156)=1.5701$	0
Total User 1	1.6 bits	
Data rates User 2	$\log_2(1+.19/1.72)=.15$	$\log_2(1+1.81/.1)=4.26$
Total User 2	4.4	
Rate Sum	6.0 bits	

IW_polite.m (integer bits like osb.m)

```
% function [b, E] = iw_polite(N, df, U, Hmag, No, Ex, mask, gap, mode,
b_target, bitcap,cb)
%
% Calculates data rates of M users and corresponding bit distributions and
Energy distributions
% using iterative waterfilling
%
% Inputs
% ------
% N: number of sub-channels
% M: number of users
% Hmag: squared channel transfer and crosstalk matrix (N x U x U matrix)
% Hmag(n,i,j) is the crosstalk transfer function from loop i to j at the
nth bin.
% No: noise energy/sample
% Ex: signal energy/SYMBOL
% mask: PSD mask - largest value N x U
% gap: gap (not in dB)
% mode: (U x 1 vector) each value is one of the followings
% 0 - rate adaptive
% 1 - fixed margin (power minimization)
% 2 - margin adaptive
% b_target: target bits on 1 DMT symbol for modes 1 and 2
% bitcap: maximum possible number of bits at each frequency bin
% cb =1 for cplx BB and =2 for real BB
%
% Outputs
% -------
% b: bit distribution (N x U matrix)
% E: energy distribution (N x U matrix)
%
% Remarks
% Iterate waterfiling for each user 10 times
% Youngjae(Sean) Kim - modified J. Cioffi, April 2024
```

```
>> [b, E] = iw_polite(8, 2, H3.*conj(H3), Noise, [8 8],
mask, gap, zeros(8,1), [5 5], bitcap,1)
b =
        0      8.0000
        0      8.0000
   2.0264      1.0000
   8.0000           0
   8.0000           0
   8.0000           0
   2.0264      1.0000
        0      8.0000
```

```
>> [b1 b2] %(osb)
   0   8
   0   8
   7   0
   8   0
   0   0
   8   0
   7   0
   0   8
```

```
E =
        0      0.6375
        0      0.7469
   0.6300      0.3609
   0.8272           0
   0.7064           0
   0.8272           0
   0.6300      0.3609
        0      0.7469
>> sum(b) =   28.0529   26.0000
>> sum([b1 b2]) =   30    24 % both OSB/IW add to 54
```

```
>> [S1 S2] = %(osb)
        0      0.6375
        0      0.7469
   0.7017           0
   0.8272           0
        0           0
   0.8272           0
   0.7017           0
        0      0.7469
```

- Sum is same, user 1 is better in osb.
- With continuous bit distribution, osb would be slightly better.

Energies < 8 because iw calls campello.m, which allows only integer bits (like osb.m).

IW.m (non-integer) – not in text, but at website

```
function function [b, E] = iw(N, U, Hmag, No, Ex, gap, mode, b_target,cb)

  Calculates data rates of M users and corresponding bit distributions and
Energy distributions
     using iterative waterfilling.

  Inputs
  ------
  N: number of sub-channels
  U: number of users
  Hmag: squared channel transfer and crosstalk matrix (N x U x U matrix)
         Hmag(n,i,j) is the crosstalk transfer function from loop i to j
at the nth bin.
  No: noise power spectrum per tone (N x U)
  Ex: signal energy/SYMBOL
  mask: PSD mask - largest value N x U
  gap: gap in dB
  mode: (U x 1 vector) each value is one of the followings
        0 - rate adaptive
        1 - fixed margin (power minimization)
        2 - margin adaptive
  b_target: target bits on 1 DMT symbol for modes 1 and 2
  bitcap: maximum possible number of bits at each frequency bin
  cb =1 for cplx BB and =2 for real BB

  Outputs
  -------
  b: bit distribution (N x U matrix)
  E: energy distribution (N x U matrix)

  Remarks
  Iterate waterfiling for each user 10 times
  Youngjae(Sean) Kim - modified J. Cioffi, April 2024
```

```
>> [b, E] = iw(8, 2, H3.*conj(H3), Noise, [8 8], gap, zeros(8,1), [5 5],1)

b =
        0    9.5897
        0    9.3612
   1.4972    1.4479
   9.2050         0
   9.4329         0
   9.2050         0
   1.4972    1.4479
        0    9.3612
E =
        0    1.9238
        0    1.9233
   1.1329    1.1148
   1.9112         0
   1.9118         0
   1.9112         0
   1.1329    1.1148
        0    1.9233

>> sum(b) % =   30.8374   31.2079
>> sum(E) % =    8    8
```

(62>54!!)

```
>> [b1 b2] %(osb)
   0    8
   0    8
   7    0
   8    0
   0    0
   8    0
   7    0
   0    8
```

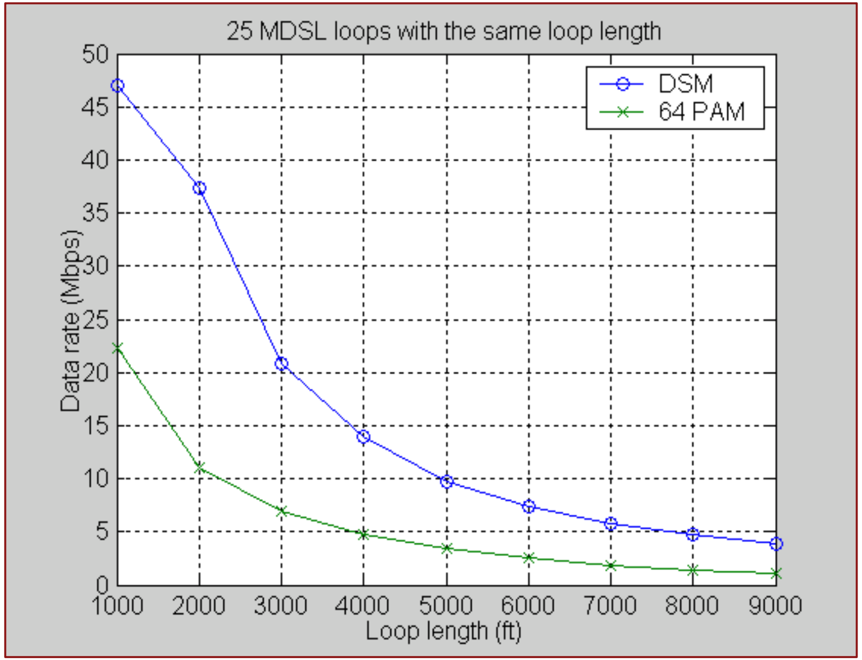
```
>> [S1 S2] = %(osb)
        0    0.6375
        0    0.7469
   0.7017         0
   0.8272         0
        0         0
   0.8272         0
   0.7017         0
        0    0.7469
```

Energies = 8 now with fractional bits

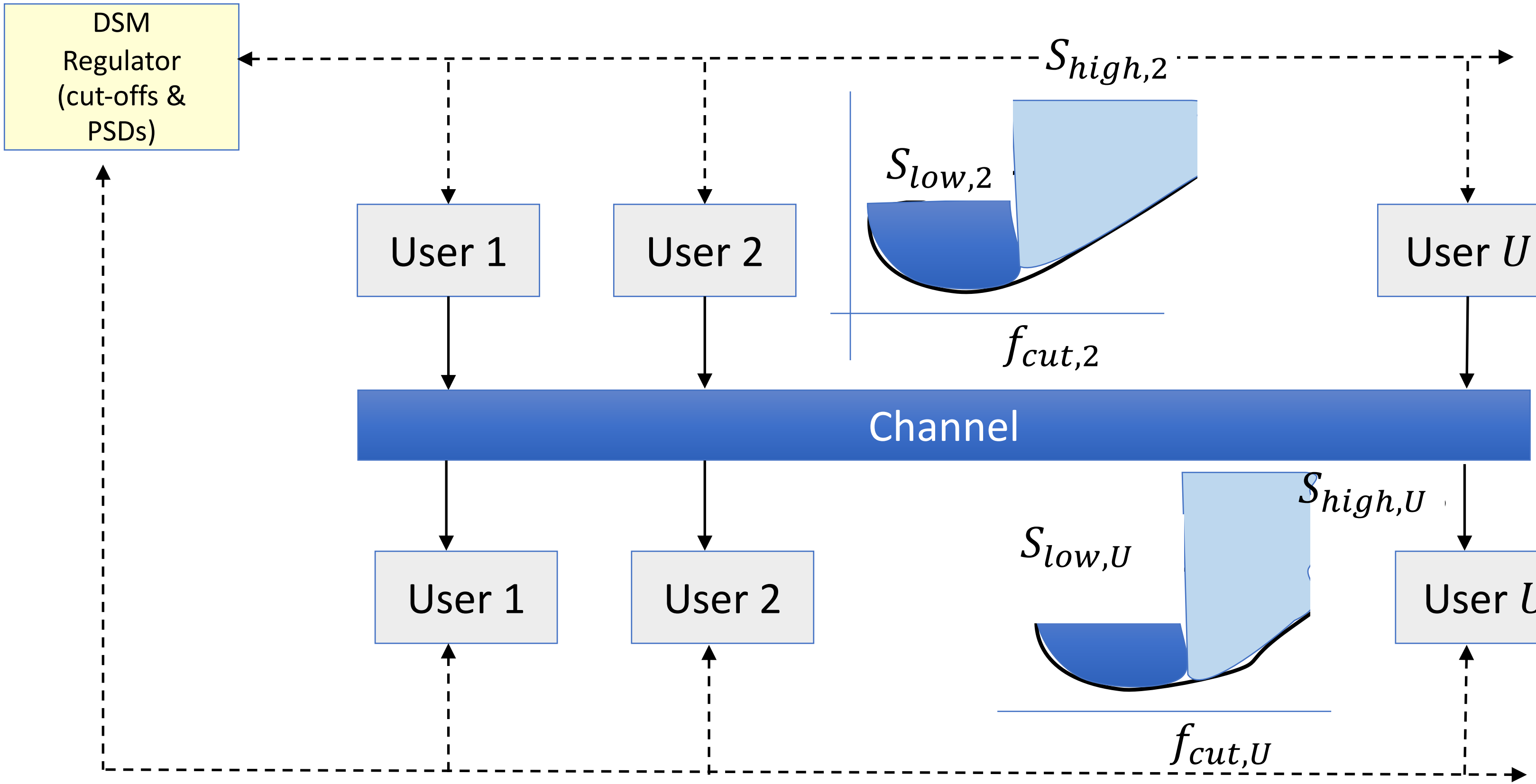
More sophisticated situation (IC of MAC & BC)

- 25 bi-directional users (so really 50 users if they all share same band – each echo cancels itself only.)
  - Otherwise, there is no GDFE xtalk cancellation in this simulation, only IW.

- Turn on MA WF for them all and let them run versus fixed spectrum with (PAM) $b = 6$ bits/Hz,
  - which was state of art prior to IW

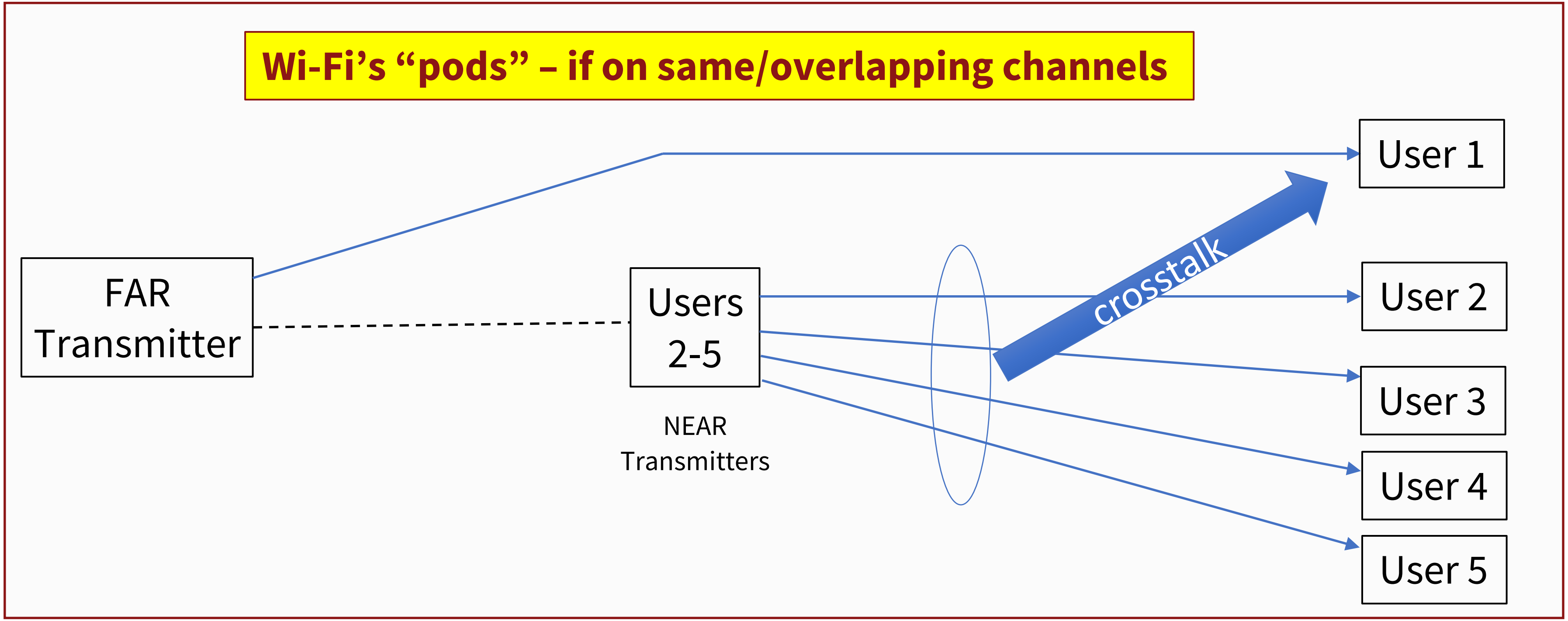


Multi-Level Water-fill Illustration



- MLIW runs IW, but with different water-levels (a bit like SWF) – must find cut-off frequency(ies).
- Very low complexity (same as IW), but central control distributes (learns) users' cut-off frequencies $f_{cut,u}$.

Near-Far Example



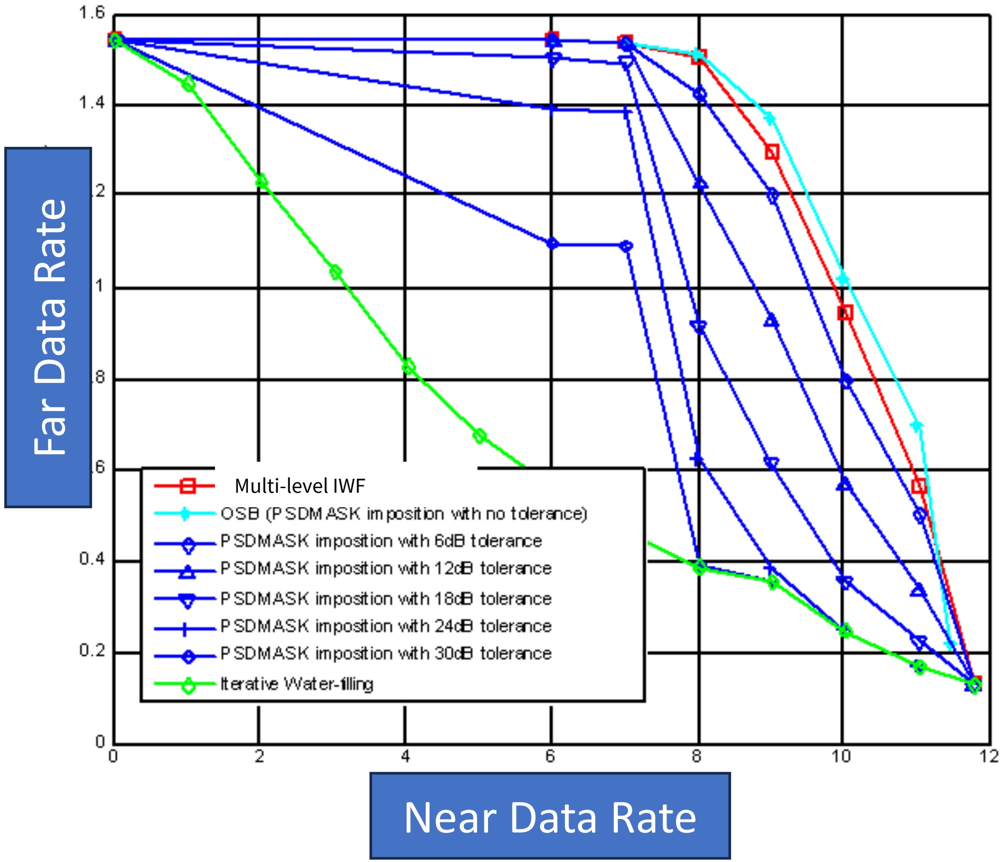
- Near-Far can arise with RIS (reflective intelligent surfaces) for adjacent bands of RIS.
- Near-Far can occur in wireline also – "remote terminals" or "distribution units."
- Adjacent cells in cellular (or Wi-Fi).

Achievable Region Comparison

- IW better than fixed, but not so good
- This plots 25 users with ML IW and with C
  - See upper right
  - Blue curves allow for margin on ML IW.
- ML IW is pretty close to OSB.

Working to locate better IW and ML-IW, OSB.

$U = 25, N = 4096$



STANFORD

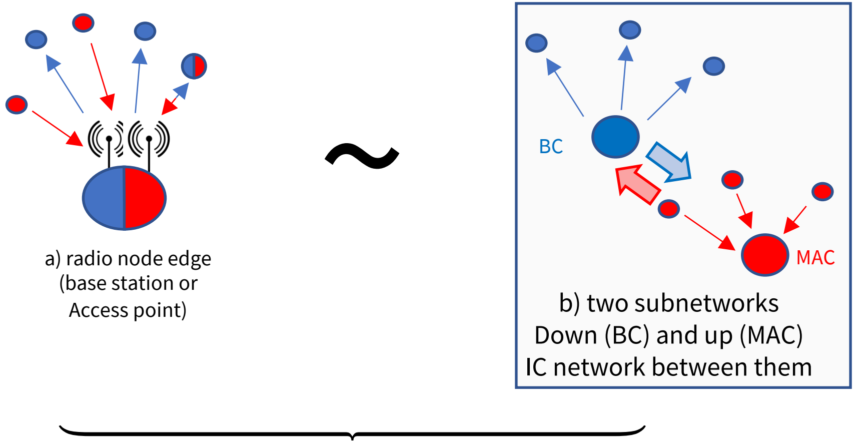
End Lecture 18
&
EE 379- 2024

THANK YOU !!

ML & Challenges

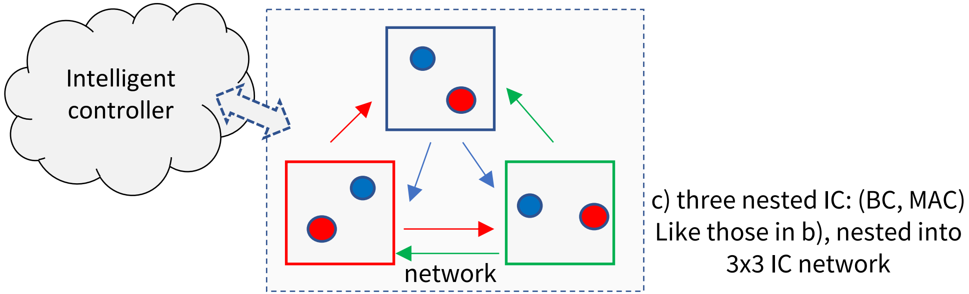
Nesting

- Use minPmac/bc on node channels

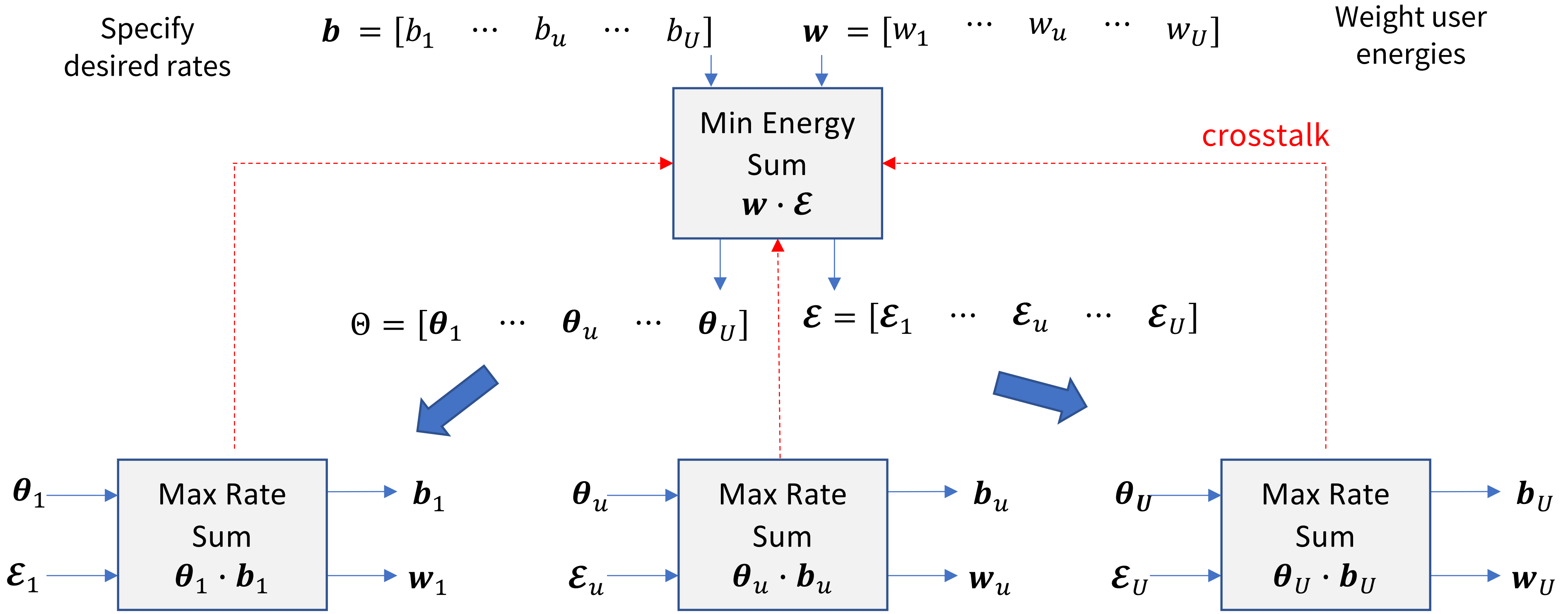


- Use ML Water-Fill between nodes

- Efficient Algorithms?
  - How/where to update?



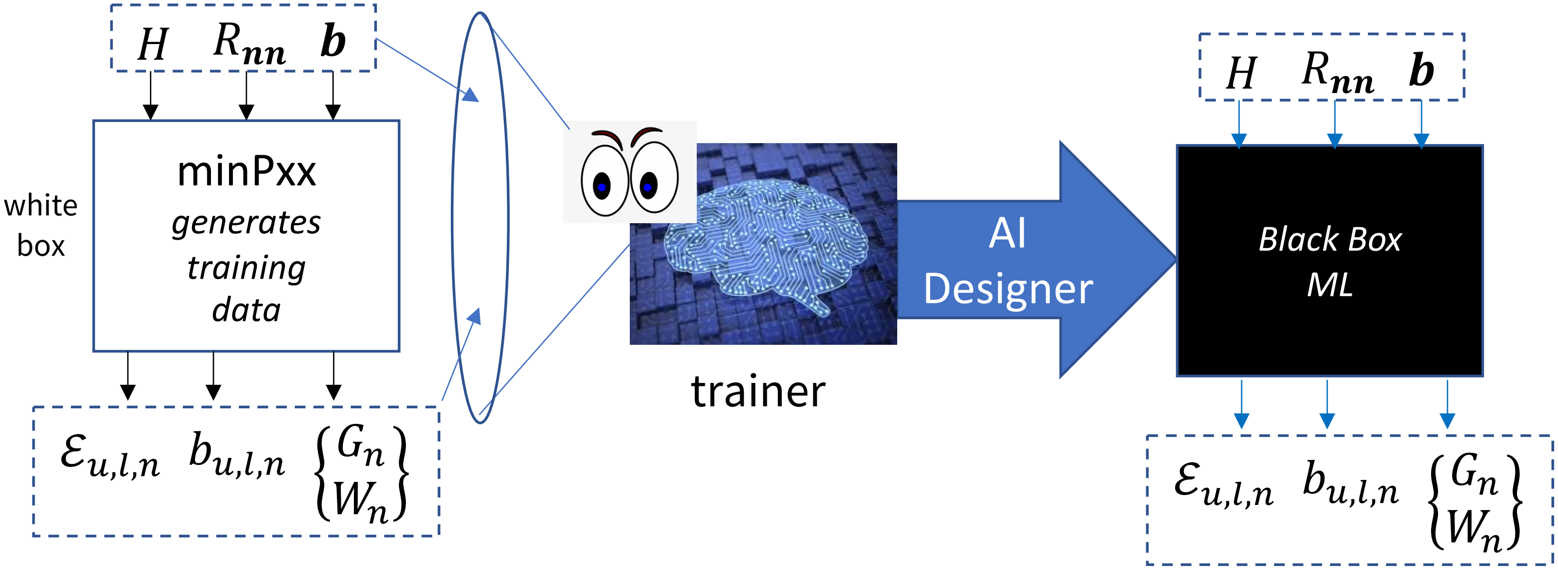
AI ("machine learned") Approximations?



- Each of these "boxes" (subnetworks) can be intense calculation
- The overall recursive cycling is actually then more intense

Machine learned “minPxx”

- minPMAC (and minPIC) optimize, but may have long run-times and numerical issues
  - They accept channel+noise and data rates   (and maybe energy in admxx)

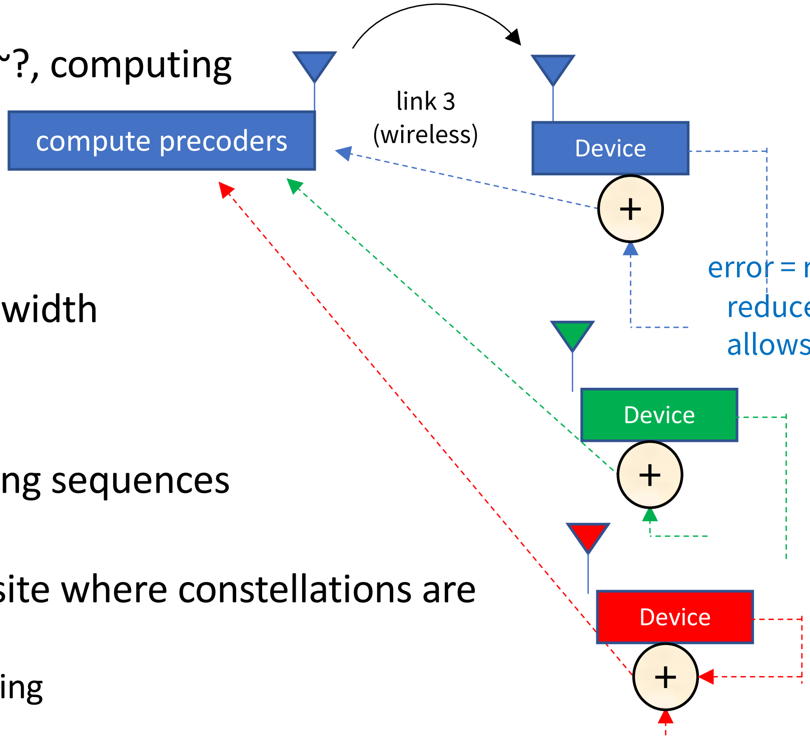


probably reccurrent and/or convolutional neural net with a few layers
With also Reinforcement-learning Over time

- Extended to nesting, complex networks

Where, and from what, to compute precoders?

- Receivers estimate channels today ~?, computing
  - filters/matrices
  - bit distributions
  - energy distributions
- This generates large overhead bandwidth
  - (even with ”indexing” schemes)
- Return only errors for pilots/sounding sequences
- Compute instead at edge or at the site where constellations are generated
  - Need error signals from pilots/sounding

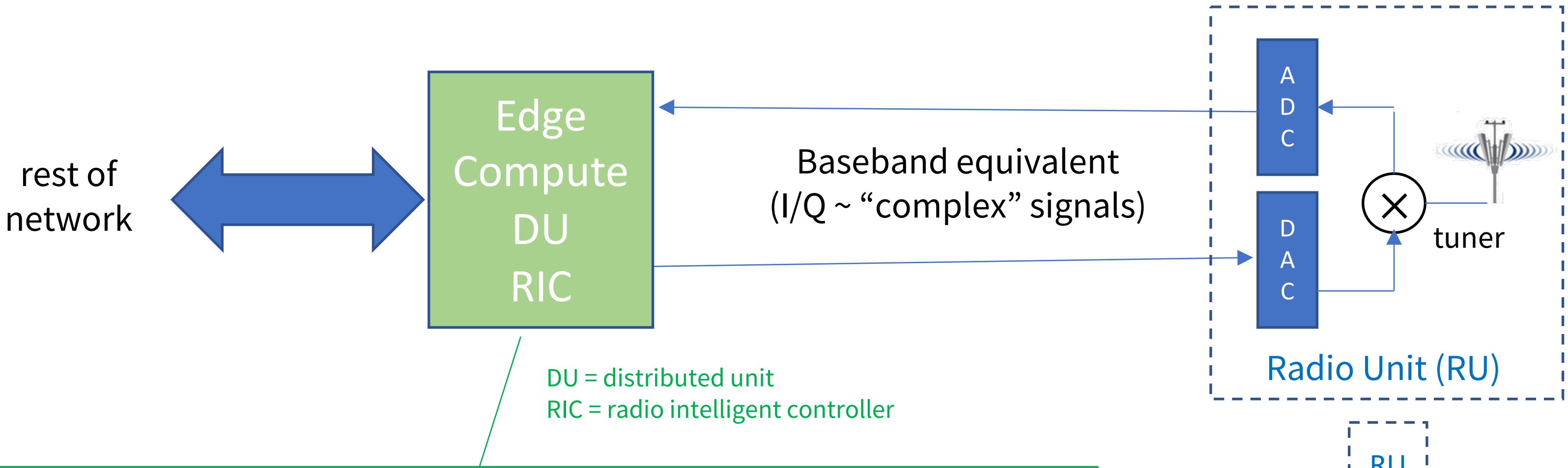


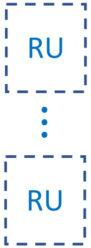
compute only errors here

error = received pilot - known pilot
reduces feedback bandwith
allows precoder calculation at edge

Algorithms (ML/AI) based on digital twin of this to update precoders ?

O-RAN/Xhaul split 7.2 (over) simplified



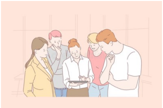


- Share servers for calculations when needed
- Coordinate the messages/signals , space, time, frequency
- This where the AI will become really helpful

Correlate Design choices with User Reaction

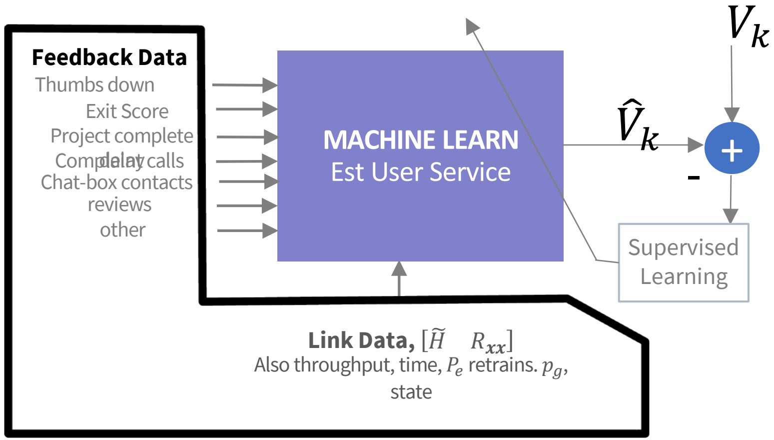
- Thumbs down 
- Exit score

- Calls to IT/ISP 
- Group success rate



- Repair/Intervention counts 

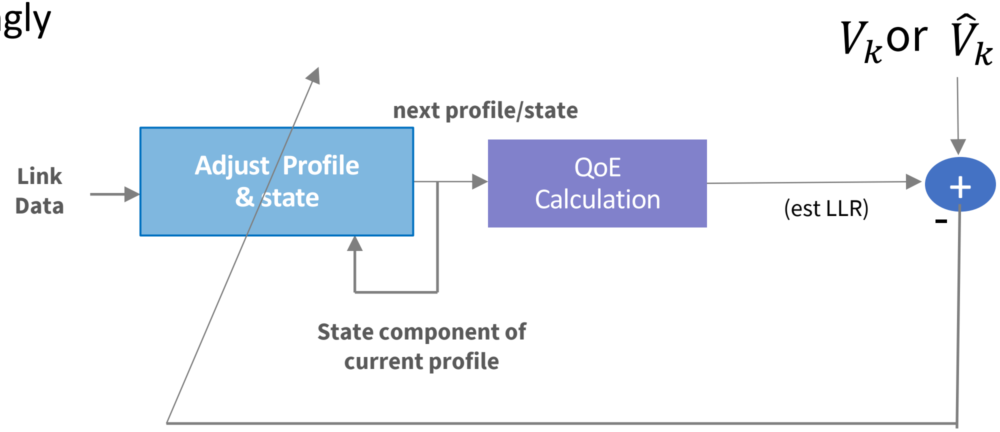
Diagnostics & Analytics
Learn the reward function V
employee feedback and link data



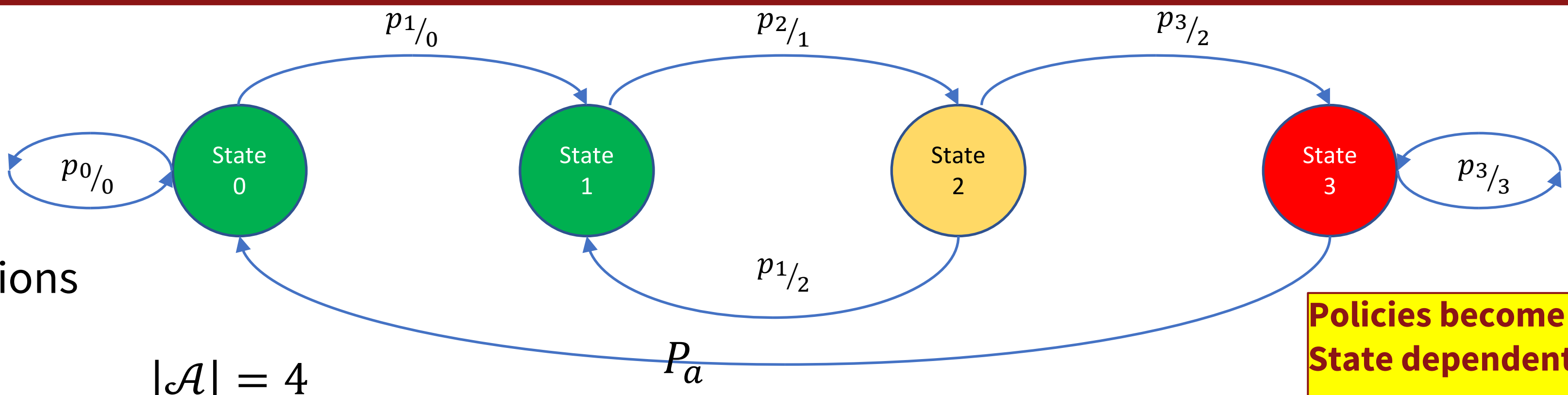
Optimize QoE (value)

- Use analytics to derive
  - Priorities (orders, weights)
- Optimize accordingly

Optimization (management)
Objective is to improve service to "green"



A Network State Machine: Reinforcement Learning



Variations
time
load

$|\mathcal{A}| = 4$

Policies become State dependent

Learn the states and P_a

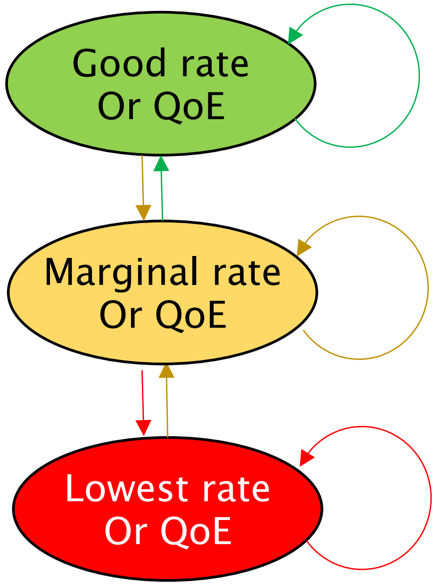
- Network user/link may be in a state or profile
  - Some are ok (user happy or green) ; amber on the edge ;
  - Red – very likely unhappy
- Markov (state-machine) models
- Learn the profile, apply appropriate design for each state
  - Objective is move to green state with profile change

$$P_a = \begin{bmatrix} p_3/_3 & p_3/_2 & 0 & 0 \\ 0 & 0 & p_2/_1 & 0 \\ 0 & p_1/_2 & 0 & p_1/_0 \\ p_0/_3 & 0 & 0 & p_0/_0 \end{bmatrix}$$

$\pi = P_a \cdot \pi$ Markov (stationary) distribution

A Network State Machine: Reinforcement Learning

- Determines Next Action (State)
  - Profile $\{R_{\boldsymbol{xx}}(u), b_u\ , [G_u \quad W_u]\}\ , u = 1, \ldots, U'$
- GYR
  - Try to get to better states
  - But this depends on cost of doing so
- Markov (state-machine) models



Can include, MCS, number of spatial streams, channel, spectrum, priority (weights $[\boldsymbol{w} \quad \boldsymbol{\theta}]$), etc

Estimating the probabilities and States

- Better on-line/real-time "fading" distributions

- While all the "Raleigh, Ricean, log-normal, angle-spread, delay-spread ……… " models create simulation environments that range through many situations, they're not specific to situation

- Each channel/user may need to estimate probability distributions for "fading/xtalk"
  - How do do this well
  - Ergodic state machines (Markov models) or slowing varying
  - Digital Twins?
  - Know the settings for each in advance?

- Then identify which state and associated pre-computed design?
  - Would this save a lot of computing energy?

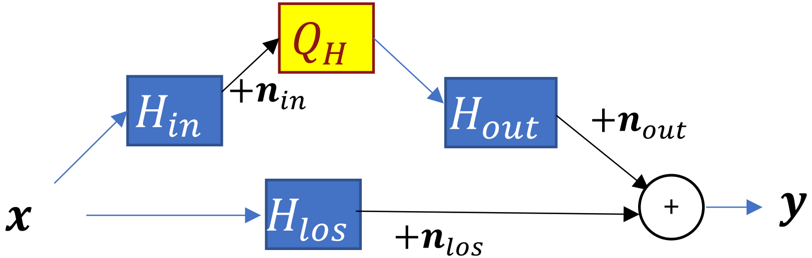
- Are Pe and data-rates the right measures? → Quality of Experience (QoE)
  - Learned from user "feedback"👍 or 👎

- Reinforcement Learning? (Recurrent Neural Net as base?)

Reflective Intelligent Surfaces (RIS)

Posed Project/Research
"maxRIS" or "minRIS"

$$\boldsymbol{y} = \underbrace{\begin{bmatrix} H_{los} \\ H_{out} \cdot Q_H \cdot H_{in} \end{bmatrix}}_{H_{RIS}} \cdot \boldsymbol{x} + \underbrace{\begin{bmatrix} \boldsymbol{n}_{los} \\ \boldsymbol{n}_{out} + Q_H \cdot \boldsymbol{n}_{in} \end{bmatrix}}_{\boldsymbol{n}_{RIS}}$$



- The RIS matrix Q_H satisfies $\|Q_H\|_F^2 \leq G_H$, the RIS gain – it may also satisfy
  - Q_H is unitary matrix (preserves energy)
  - Q_H is diagonal, and usually also unitary, to be phase/gain-only adjustment on each antenna port (in-to-out)
  - Q_H has individual elements restricted

- For a given $R_{\boldsymbol{xx}}$, maximize over Q_H

$$\mathcal{I}(\boldsymbol{y};\boldsymbol{x}) = \log_2 \left| R_{n,RIS} + H_{RIS} \cdot R_{\boldsymbol{xx}} \cdot H_{RIS}^* \right|$$

- For a given Q_H , maximize the same over $R_{\boldsymbol{xx}}$

$$R_{\boldsymbol{nn},RIS} = \begin{bmatrix} R_{\boldsymbol{nn}} & 0 \\ 0 & R_{\boldsymbol{nn},out} + Q_H \cdot R_{\boldsymbol{nn}in} \cdot Q_H^* \end{bmatrix}$$

- Iterate