Supplementary Lecture 15B

Duality Example Commands

May 20, 2026

JOHN M. CIOFFI

Hitachi Professor Emeritus of Engineering

Instructor EE379B – Spring 2026

Announcements & Agenda

- Announcements
  - Show detailes of BC duality

- Agenda
  - MAC/BC Duality Basics
    - Input deflection
    - Mappings
  - Vector MAC/BC Duality
  - MAC-dual Design

Some Matlab Design-Process Help (PS7) & $\mathcal{C}(b)$ construction

PS 7: $N > 1$, constant $L_x = L_{x,u} \equiv \mathfrak{L}_x/U$

- **Generate Hbc** that can be used with mu_bc program.
  - Variable $L_{x,u}$: set $L_x = \max_u L_{x,u}$ and expand each $\tilde{H}_u$ to have L_x- $L_{x,u}$ dummy zero columns.

```
Hmac=reshape(Hmac,Ly,Lxu,U,N);   % 4D tensor from 3D tensor FFT output, which was Ly x Lx x N
Hmac=Hmac(:,:,order,:)
Hbc=zeros(Lxu,Ly,U,N);   % use MAC's Lxu and Ly !
 for n=1:N   % note N>1
   for i=1:U
   Hbc(:,:,i,n)=Hmac(:,:,U+1-i,n)';
   end
 end
Hbc1=permute(reshape(Hbc, [Ly , Lx, N]), [ 2 1 3]); % returns to 3D tensor – reorder because Ly , Lx are from MAC
```

- **Generate Rxxm** that can be used with mac2bc program.

```
Info.Rxxs  % from slower minPMACMIMO is already (Lxu, Lxu, U, N) if Lxu is constant
Etemp=celltomat(info.Rxx) % so only need this if Lxu=1 because faster minPMAC was used. % Rxxm=cell2mat(info.Rxxs)
Rxxm=zeros(1,1,U,N); % for minPMAC
 for n=1:N E=diag(Etemp(:,:,n));
 Rxxm(1,1,:,n)=E(u, order,n);    end  % Since Lxu=1, conversion to mac2bc format (per tone) % any u=1,…,U works
% now find dual Rxxb
Rxxb=zeros(Ly,Ly,U,N);
bbc=zeros(U,N); % for use on next slide
 for n=1:N   Rxxb(:,:,:,n)=mac2bc(Rxxm(:,:,:,n), Hmac(:,:,:,n));  end % per tone use of mac2bc 3D tensor to 3D tensor
```

HWH 7: BC Design Completion

- Data calculation below is a check, shown here for $U = 3$.

```
bbc(1,n)=real(log2(1+Hbc(:,:,1,n)*Rxxb(:,:,1,n)*Hbc(:,:,1,n)'));
bbc(2,n)=real(log2((1+Hbc(:,:,2,n)*(Rxxb(:,:,2,n)+Rxxb(:,:,1,n))*Hbc(:,:,2,n)')/(1+Hbc(:,:,2,n)*Rxxb(:,:,1,n)*Hbc(:,:,2,n)')));
bbc(3,n)=real(log2((1+Hbc(:,:,3,n)*(Rxxb(:,:,3,n)+Rxxb(:,:,2,n)+Rxxb(:,:,1,n))*Hbc(:,:,3,n)')/(1+Hbc(:,:,3,n)*(Rxxb(:,:,2,n)+Rxxb(:,:,1,n))*Hbc(:,:,3,n)')));
end
bvec=sum(bbc')
bsum=sum(bvec)
```

- Prepare for BC Design – stack the A matrices horizontally.

```
A=zeros(Ly, Ly*U, N); % 3D tensor to match Hbc1 and mu_bc.m program
 for n=1:N  A(:,:,n)=[ sqrtm(Rxxb(:,:,1,n))  ….. sqrtm(Rxxb(:,:,U,n)) ];  end

[Bu, Gunb, S0, MSWMFunb, B] = mu_bc(Hbc1, A, [Lyu] , cb);
```

- **mu_bc outputs cell arrays**, so allows variable (BC) $L_{y,u}$; when constant $L_{y,u} = L_y$ can translate for display:

```
GU=zeros(U · 𝔏_y, U · 𝔏_y, N);  % 𝔏_y corresponds to BC here
for n=1:N   GU(:,:,n)=[Gunb{1,n} ; …, Gunb{U,n}];  end
MSWMFU=zeros(U · 𝔏_y, L_y, N);  for n=1:N  MSWMFU(:,:,n)=[MSWMFunb{1,n}; … ; MSWMFunb{U,n}];  end
```

- More generally GU cells are **each** $L_{y,u} \times L_{y,u}$, while MSWMFU are each $\mathfrak{L}_y \times L_{y,u}$.

64-tone - one vertex

```
[Rxx,Eun, theta, bun, FEAS_FLAG, bu_a, info]=minPMAC(H64, [445 412 132]', [1 1 1]',1);
FEAS_FLAG =    2
bu_a = 445.0000      412.0000      132.0000
        %  bu_v                    Eun          bun         theta    order   frac    cID
  445.81   411.19      132  {1x3x64 }  {1x3x64 }  {1x3}    {1x3}   0.99    1
  425.68   431.32      132  {1x3x64 }  {1x3x64}  {1x3}     {1x3}    0.01    1
>> info.order{:} =
  3   2   1
  3   1   2
>> info.frac'*info.bu_v = 445.0000  412.0000  131.9999
>> info.frac' =    0.9904    0.0096
>> sum(Eun') =   65.9553   60.2757   40.4453
>> sum(Eun,'all') =  166.6763 < 3 x 64
```

Small < 1% ;Might just use vertex 1
Large/small → numerical issues

```
H64mac =reshape(H64,2,1,3,64); % add extra tensor dimension for transpose
H64bc=zeros(1,2,3,64); % note reversal of first two dimensionalities
for n=1:64
     for i=1:3   H64bc(:,:,i,n) = H64mac(:,:,4-i,n)';  end
     H64bc1=permute(reshape(H64bc, [2 , 3, 64]), [ 2 1 3]);
end
```

```
Rxxm=zeros(1,1,3,64);
Etemp=cell2mat(info.Rxx);
   for n=1:64
   E=diag(Etemp(:,:,n));
   Rxxm(1,1,:,n)=E;    end
Rxxb=zeros(2,2,3,64);
bbc=zeros(3,64);
  for n=1:64   Rxxb(:,:,:,n)=mac2bc(Rxxm(:,:,:,n), H64mac(:,:,:,n)); end
A=zeros(2, 6, 64);
  for n=1:64  A(:,:,n)=[ sqrtm(Rxxb(:,:,1,n)) , sqrtm(Rxxb(:,:,2,n)),  sqrtm(Rxxb(:,:,3,n)) ];  end
```

```
>> [Bu, Gunb, S0, MSWMFunb, B] = mu_bc(H64bc1, A, [1 1 1], 1);
>> Bu =  131.9999  411.7251  445.2751
Buvertex1=Bu; % save for next slide
```

checks with reverse order for vertex 1

```
GU=zeros(6, 6, 64);
MSWMFU=zeros(6,1,64);
for n=1:64   GU(:,:,n)=[Gunb{1,n} ; Gunb{2,n} ; Gunb{3,n}];
MSWMFU(:,:,n)=[MSWMFunb{1,n}; MSWMFunb{2,n}  ; MSWMFunb{3,n}];  end

>> GU(:,:,23) =     % 6 x 6
  1.0000 + 0.0000i   0.0920 - 0.3464i   8.7367 + 1.1630i  10.8139 -15.1709i  -0.6748 - 4.2623i   1.9688 + 0.6639i
  0.0000 + 0.0000i   1.0000 + 0.0000i   3.1234 +24.3936i  48.6591 +18.2925i  11.0106 - 4.8735i  -0.3797 + 5.7850i
  0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.9891 - 1.8681i  -1.3553 - 0.3648i   0.4582 - 0.4967i
  0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i  -0.1475 - 0.6474i   0.3091 + 0.0816i
  0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i  -0.2233 + 0.4266i
  0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i

>> MSWMFU(:,:,23) =        % 6 x 1
  0.8840 - 0.3319i
  1.5285 + 2.1461i
--------------------------
  0.0553 - 0.0808i
  0.0460 + 0.0052i
--------------------------
  0.0535 - 0.0801i
 -0.1988 - 0.0213i
```

Note the mu_bc match is not quite perfect on the 99% point

```
>> A(:,:,23) =
  0.0805 - 0.0000i   0.0074 - 0.0279i   0.2114 + 0.0000i   0.2091 - 0.3950i   0.9368 - 0.0000i  -0.2092 + 0.3996i
  0.0074 + 0.0279i   0.0103 - 0.0000i   0.2091 + 0.3950i   0.9447 + 0.0000i  -0.2092 - 0.3996i   0.2172 - 0.0000i
```

check other vertex

- Small error in 99% on vertex share rate can require large compensation on the 1% rate.

```
H64mac =reshape(H64,2,1,3,64);
H64mac=H64mac(:,:,[2 1 3],:);  % set order for other vertex
H64bc=zeros(1,2,3,64);
for n=1:64
for i=1:3   H64bc(:,:,i,n) = H64mac(:,:,4-i,n)'; end
end
H64bc1=permute(reshape(H64bc, [2 , 3, 64]), [ 2 1 3]);

Rxxm=zeros(1,1,3,64);
E=cell2mat(info.Eun);

for n=1:64
Rxxm(1,1,:,n)=E(2,[2 1 3],n);    end
Rxxb=zeros(2,2,3,64);
bbc=zeros(3,64);
for n=1:64   Rxxb(:,:,:,n)=mac2bc(Rxxm(:,:,:,n), H64mac(:,:,:,n)); end

A=zeros(2, 6, 64);
for n=1:64  A(:,:,n)=[ sqrtm(Rxxb(:,:,1,n)) , sqrtm(Rxxb(:,:,2,n)),  sqrtm(Rxxb(:,:,3,n)) ];
end

[Bu, Gunb, S0, MSWMFunb, B] = mu_bc(H64bc1, A, [1 1 1], 1);

>> Bu =  131.9999  416.7071  440.2931
>> rate = [Buvertex1 ; Bu]
  131.9999  411.7251  445.2751
  131.9999  416.7071  440.2931
>> info.frac'*rate  =
  131.9999  411.7730  445.2270   versus 412 and 445
```

```
Check with:
for n=1:64
bbc(1,n)=real(log2(1+H64bc(:,:,1,n)*Rxxb(:,:,1,n)*H64bc(:,:,1,n)'));
bbc(2,n)=real(log2((1+H64bc(:,:,2,n)*(Rxxb(:,:,2,n)+Rxxb(:,:,1,n))*H64bc(:,:,2,n)')/(1+H64bc(:
,:,2,n)*Rxxb(:,:,1,n)*H64bc(:,:,2,n)')));
bbc(3,n)=real(log2((1+H64bc(:,:,3,n)*(Rxxb(:,:,3,n)+Rxxb(:,:,2,n)+Rxxb(:,:,1,n))*H64bc(:,:,3,n)
')/(1+H64bc(:,:,3,n)*(Rxxb(:,:,2,n)+Rxxb(:,:,1,n))*H64bc(:,:,3,n)')));
end
bvec=sum(bbc') =  131.9999  411.7251  445.2751
bsum=sum(bvec) =  989.0000

>> newfrac=inv(rate(1:2,2:3)')*[412 ; 445]  =
   0.9448
   0.0552
```

Better design is:
19 symbols mac2bc vertex 1 &
1 symbol mac2bc vertex 2.

```
> E(:,:,21:25)
   1.1044   1.1722   0.4032
   1.1044   1.1722   0.4032

   1.0237   1.1371   0.5177
   1.0237   1.1371   0.5177

   0.9394   1.1115   0.6258
   0.9394   1.1115   0.6258

   0.8528   1.0964   0.7249
   0.8528   1.0964   0.7249

   0.7653   1.0920   0.8132
   0.7653   1.0920   0.8132
```

Energies are from minPMAC, but then used in mac2bc.

Note equal energies both vertices' of each tone

Capacity Region

5.5.5 Generation of the Vector BC Capacity Rate Region

The steps for tracing the vector BC Capacity Region are:

1. Create a dual vector MAC channel (with coefficients $\widetilde{H}$ and noise autocorrelation I).
2. for each $\boldsymbol{b}'$ with $b'_1 = 0, ..., b_{1,max}$, ... $b'_U = 0, ..., b_{U,max}$ with increments selected appropriately and maximums chosen sufficiently large to be outside the rate region (i.e., equal to the single user capacity for all other users zeroed)
   (a) Find the energy vector $\boldsymbol{\mathcal{E}}$ for a given $\boldsymbol{b}$ on the dual vector MAC using the minPMAC program of Section 5.4.
   (b) if $\sum_u \mathcal{E}_u \leq \mathcal{E}$, then the point is in the region, so $c_{new}(\boldsymbol{b}) = \{\boldsymbol{b}' \bigcup c_{old}(\boldsymbol{b})\}$.
3. Trace the boundary for of $\boldsymbol{b}$ in Step 2 for which $\sum_u \mathcal{E}_u = \mathcal{E}$.

- Check any point's admissibility on the dual MAC with admMAC

End Supplementary Lecture 15B