MATLAB Example 3: Reproducing Fig. 12(a) in IEEE TCOM "Efficient PHY Layer Abstraction for Fast Simulations in Complex System Environments"

Sian Jin (sianjin@uw.edu)

Prerequisites: MATLAB 2020b (or later version), MATLAB WLAN toolbox in MATLAB 2020b (or later version)

Goal: Learn how to implement and use the EESM-log-SGN-LSC shown in Fig. 11 of our IEEE TCOM paper.

Folders:

MU code LSC/1 Full PHY (for effective SNR)
MU code LSC/1 Full PHY (for effective INR)
MU code LSC/2 EESM parameter optimization
MU code LSC/3 EESM-log-SGN-LSC interference parameter optimization
MU code LSC/4 EESM-log-SGN-LSC method

MU code basic/1 Full PHY (multi interferers)
MU code basic/2 EESM parameter optimization
MU code basic/3 Basic EESM-log-SGN method

Step 1 (Full PHY simulation):

Step 1.1 (Full PHY simulation for a desired TX-desired RX pair)

Goal: full PHY simulation for a desired TX-desired RX pair (single BSS setup, corresponding to effective SNR) as suggested in the input of Fig. 11 in our IEEE TCOM paper.

Folder:
MU code LSC/1 Full PHY (for effective SNR)

Open: MU code LSC/1 Full PHY (for effective SNR)/fullPHY.m

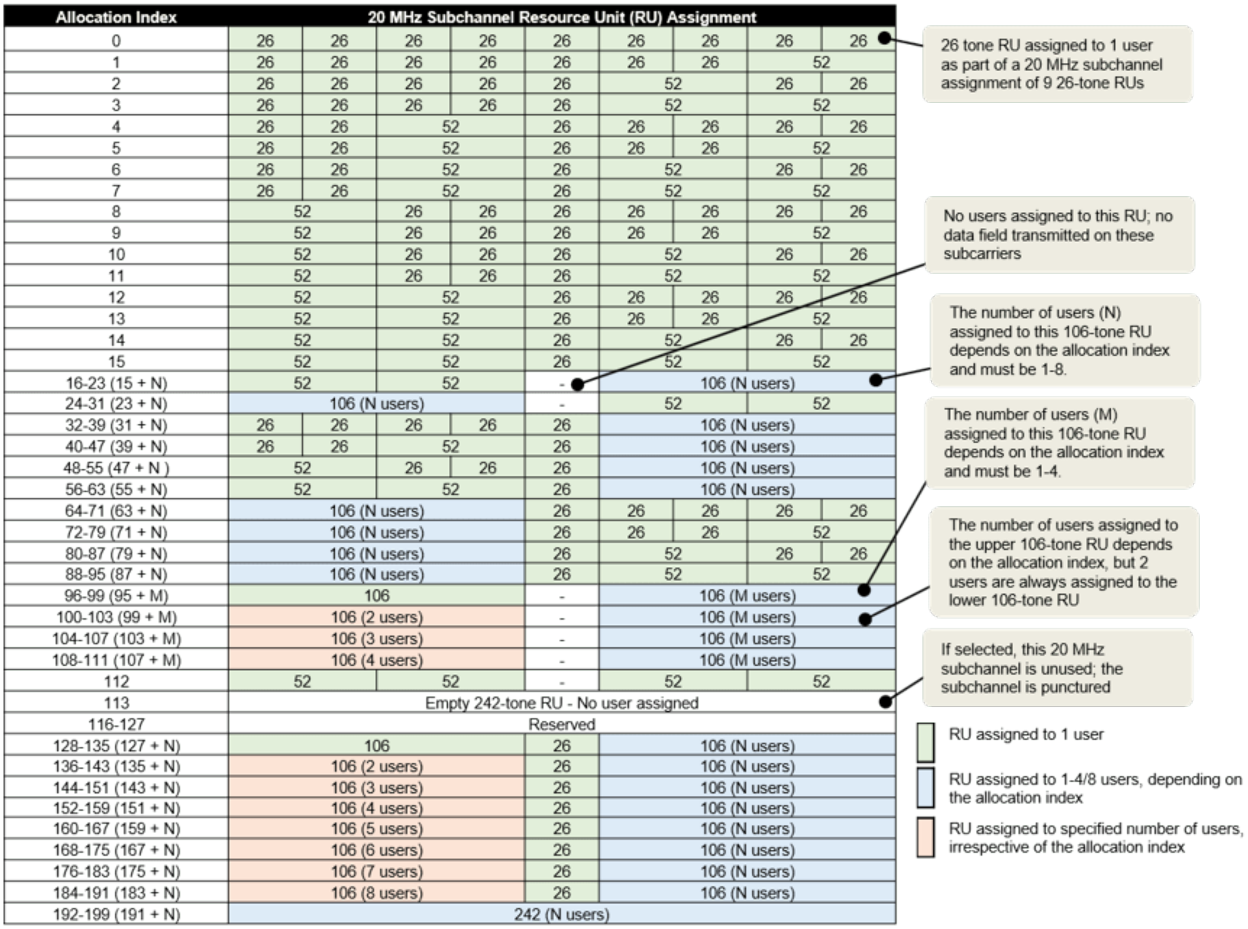
PHY layer configuration:

- OFDM with 242 subcarriers
- 4 × 2 MIMO with 2 streams/user
- TGax channel model-D
- Payload length = 1000 Byte
- MCS4, LDPC coding

In this example, we set the 11ax allocation index to be 192.

Checking the RU assignment figure, for allocation index 192, the only RU contains 242 subcarriers; the only user occupies this RU. So, we set userIdx = 1, and ruIdx = 1.

4×2 MIMO with 2 streams corresponds to numTxRx = [4 2] and Nsts = 2.
MCS4 corresponds to mcs = [4].
TGax channel model-D corresponds to chan = "Model-D".
Payload length = 1000 Byte corresponds to cfgHE.User{userIdxIter}.APEPLength = 1000.
LDPC coding is the default coding for 11ax. There is no need to setup LDPC. Please check cfgHE.User for the LDPC configuration.



Allocation Index	20 MHz Subchannel Resource Unit (RU) Assignment								
0	26	26	26	26	26	26	26	26	26
1	26	26	26	26	26	26	26	52	
2	26	26	26	26	26	52		26	26
3	26	26	26	26	26	52		52	
4	26	26	52		26	26	26	26	26
5	26	26	52		26	26	26	52	
6	26	26	52		26	52		26	26
7	26	26	52		26	52		52	
8	52		26	26	26	26	26	26	26
9	52		26	26	26	26	26	52	
10	52		26	26	26	52		26	26
11	52		26	26	26	52		52	
12	52		52		26	26	26	26	26
13	52		52		26	26	26	52	
14	52		52		26	52		26	26
15	52		52		26	52		52	
16-23 (15 + N)	52		52		-	106 (N users)			
24-31 (23 + N)	106 (N users)				-	52		52	
32-39 (31 + N)	26	26	26	26	26	106 (N users)			
40-47 (39 + N)	26	26	52		26	106 (N users)			
48-55 (47 + N)	52		26	26	26	106 (N users)			
56-63 (55 + N)	52		52		26	106 (N users)			
64-71 (63 + N)	106 (N users)				26	26	26	26	26
72-79 (71 + N)	106 (N users)				26	26	26	52	
80-87 (79 + N)	106 (N users)				26	52		26	26
88-95 (87 + N)	106 (N users)				26	52		52	
96-99 (95 + M)	106				-	106 (M users)			
100-103 (99 + M)	106 (2 users)				-	106 (M users)			
104-107 (103 + M)	106 (3 users)				-	106 (M users)			
108-111 (107 + M)	106 (4 users)				-	106 (M users)			
112	52		52		-	52		52	
113	Empty 242-tone RU - No user assigned								
116-127	Reserved								
128-135 (127 + N)	106				26	106 (N users)			
136-143 (135 + N)	106 (2 users)				26	106 (N users)			
144-151 (143 + N)	106 (3 users)				26	106 (N users)			
152-159 (151 + N)	106 (4 users)				26	106 (N users)			
160-167 (159 + N)	106 (5 users)				26	106 (N users)			
168-175 (167 + N)	106 (6 users)				26	106 (N users)			
176-183 (175 + N)	106 (7 users)				26	106 (N users)			
184-191 (183 + N)	106 (8 users)				26	106 (N users)			
192-199 (191 + N)	242 (N users)								

26 tone RU assigned to 1 user as part of a 20 MHz subchannel assignment of 9 26-tone RUs

No users assigned to this RU; no data field transmitted on these subcarriers

The number of users (N) assigned to this 106-tone RU depends on the allocation index and must be 1-8.

The number of users (M) assigned to this 106-tone RU depends on the allocation index and must be 1-4.

The number of users assigned to the upper 106-tone RU depends on the allocation index, but 2 users are always assigned to the lower 106-tone RU

If selected, this 20 MHz subchannel is unused; the subchannel is punctured

🟩 RU assigned to 1 user

🟦 RU assigned to 1-4/8 users, depending on the allocation index

🟥 RU assigned to specified number of users, irrespective of the allocation index

The configuration code is shown as follows:

```
%% Full PHY simulation setup
clear all
tStart = tic;
mcs = [4]; % Vector of MCS to simulate between 0 and 11
numTxRx = [4 2]; % Matrix of MIMO schemes, each row is [numTx numRx]
chan = "Model-D"; % String array of delay profiles to simulate
userIdx = 1; % User of investigation
ruIdx = 1; % RU of investigation
Nsts = 2; % Number of space-time streams
maxnumberrors = 40*1e3;  % The maximum number of packet errors at an SNR
point
maxNumPackets = 40*1e3; % The maximum number of packets at an SNR point
% maxnumberrors = 5*1e1;  % The maximum number of packet errors at an SNR
point
% maxNumPackets = 5*1e2; % The maximum number of packets at an SNR point

% Fixed PHY configuration for all simulations
cfgHE = wlanHEMUConfig(192); % Input 11ax allocation index
% The full RU assignment and allocation index lookup table is shown
% in the quick start guide
% Example: when allocation index = 24,
% then 1st RU has size 106, 2nd/3rd RU size = 52
for userIdxIter = 1:numel(cfgHE.User)
    cfgHE.User{userIdxIter}.APEPLength = 1000; % Payload length in bytes
end
```

Open: MU code LSC/1 Full PHY (for effective SNR)/getBox0SimParams.m

Consider simulating the PHY layer under RX SNR = 9, 11, 13, 15, 17, 18, 19, 20 dBs. The transmitter is the desired transmitter that transmit signal.
In the variable snr: Model-D, 4x2, MCS4 part, set the snr to be [9 11 13 15 17 18 19 20] (notice the bold part in the following code):

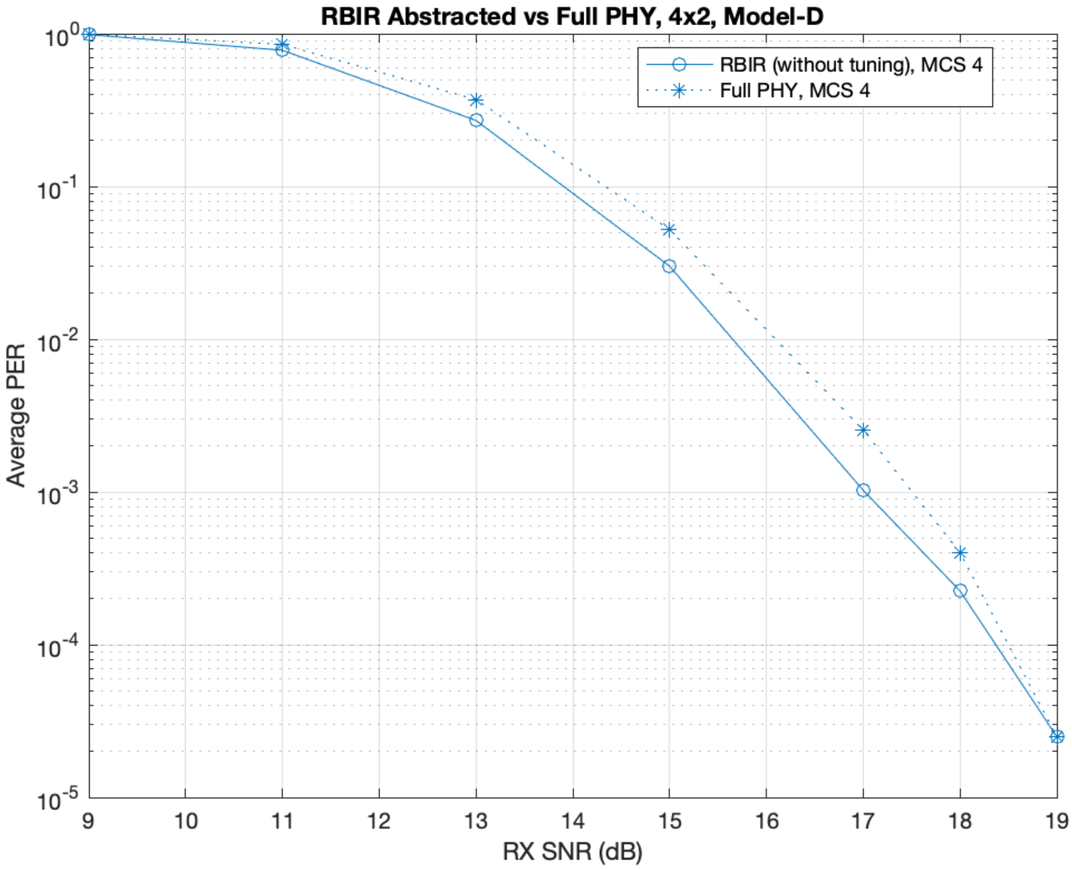
```
snr = {
    % Model-B
    [ ...
        {... % 1x1
    [-3:4:9,11], ...   % MCS 0
    [1:4:13], ...     % MCS 1
    [2:4:18], ...     % MCS 2
    [7:4:19,21], ...     % MCS 3
    [9:4:25], ...     % MCS 4
    [13:4:29], ...    % MCS 5
    [14:4:30], ...    % MCS 6
    [16:4:32,34], ...    % MCS 7
    [18:4:34,36] ... % MCS 8
    [18:4:38] ...     % MCS 9
    }; ...
        {... % 4x1
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
        {... % 4x2
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
         {... % 8x2
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
    ];

% Model-D
    [ ...
        {... % 1x1
    -3:3:9, ...     % MCS 0
    0:4:12, ...     % MCS 1
    2:4:18, ...     % MCS 2
    8:4:20, ...     % MCS 3
    11:4:23, ...    % MCS 4
    [14:4:26,28], ...    % MCS 5
    16:4:28, ...    % MCS 6
    18:4:30, ...    % MCS 7
    21.5:4:37.5 ... % MCS 8
    20:4:36 ...     % MCS 9
    }; ...
        {... % 4x1
    -4:2:2, ...   % MCS 0
    5:2:15, ...   % MCS 1
    9:2:21, ...   % MCS 2
    12:2:24, ...  % MCS 3
    12:2:16, ...  % MCS 4
    20:3:32, ...  % MCS 5
    22:3:34, ...  % MCS 6
    24:3:39, ...  % MCS 7
    27:3:40 ...   % MCS 8
    29:3:44 ...   % MCS 9
    }; ...
        {... % 4x2
    [11,13,14,15], ...   % MCS 0
    5:2:15, ...   % MCS 1
    9:2:21, ...   % MCS 2
    12:2:24, ...  % MCS 3
    [9 11 13 15 17 18 19 20], ...  % MCS 4
    20:3:32, ...  % MCS 5
    22:3:34, ...  % MCS 6
    24:3:39, ...  % MCS 7
    27:3:40 ...   % MCS 8
    29:3:44 ...   % MCS 9
    }; ...
        {... % 8x2
    1:2:10, ...   % MCS 0
    5:2:15, ...   % MCS 1
```

```
9:2:21, ...    % MCS 2
12:2:24, ...   % MCS 3
[15,16,17,18,19,20,21,22], ...  % MCS 4
20:2:30, ...   % MCS 5
22:3:34, ...   % MCS 6
24:3:39, ...   % MCS 7
27:3:40 ...    % MCS 8
29:3:44 ...    % MCS 9
}; ...
] ...
};
```

After the above setup, go back to MU code LSC/1 Full PHY (for effective SNR)/fullPHY.m
This is the main script of full PHY simulation. click on ‘Run’ in MATLAB.
The main script will call MU code LSC/1 Full PHY (for effective SNR)/box0Simulation.m, the key function to realize full PHY simulation. This key function assumes ZF MIMO precoding and MMSE MIMO decoding. Users can comment out the ZF MMO precoding calculation part in box0Simulation.m to enable the default Fourier MIMO precoding.

After running the above full PHY simulation (takes about 3~5 hours), a file named “snrPer_Config192_Model-D_4-by-2_MCS4.mat” will be created.
This file includes 40000 {post-MIMO processing SINR matrix, packet error state} pairs for each RX SNR. The post-MIMO processing SINR matrices are in results{snrIdx}.sinrStore, and the packet error states are in results{snrIdx}.perStore, where snrIdx =1,2,3,4,5,6,7,8.

A figure named “RBIR Abstracted vs Full PHY, 4x2, Model-D” will also pop up.
This figure shows the the average PER vs RX SNR(dB) of the full PHY simulation and RBIR prediction (without tuning). We can observe that the two curves approximately match, indicating that the full PHY simulation works well.



Step 1.2 (Full PHY simulation for an interferer-desired RX pair)
Goal: full PHY simulation for an interferer-desired RX pair (single BSS setup, corresponding to effective INR) as suggested in the input of Fig. 11 in our IEEE TCOM paper.

Folder:
MU code LSC/1 Full PHY (for effective INR)

Open: MU code LSC/1 Full PHY (for effective INR)/fullPHY.m
Config the same setup as in MU code LSC/1 Full PHY (for effective SNR)/fullPHY.m:

```
%% Full PHY simulation setup
clear all
tStart = tic;
mcs = [4]; % Vector of MCS to simulate between 0 and 11
numTxRx = [4 2]; % Matrix of MIMO schemes, each row is [numTx numRx]
chan = "Model-D"; % String array of delay profiles to simulate
userIdx = 1; % User of investigation
ruIdx = 1; % RU of investigation
Nsts = 2; % Number of space-time streams
maxnumberrors = 40*1e3;  % The maximum number of packet errors at an SNR
point
maxNumPackets = 40*1e3; % The maximum number of packets at an SNR point
% maxnumberrors = 5*1e1;  % The maximum number of packet errors at an SNR
point
% maxNumPackets = 5*1e2; % The maximum number of packets at an SNR point

% Fixed PHY configuration for all simulations
cfgHE = wlanHEMUConfig(192); % Input 11ax allocation index
% The full RU assignment and allocation index lookup table is shown
% in the quick start guide
% Example: when allocation index = 24,
% then 1st RU has size 106, 2nd/3rd RU size = 52
for userIdxIter = 1:numel(cfgHE.User)
    cfgHE.User{userIdxIter}.APEPLength = 1000; % Payload length in bytes
end
```

The difference from step 1.1 lies in the following setting.
Open: MU code LSC/1 Full PHY (for effective INR)/getBox0SimParams.m

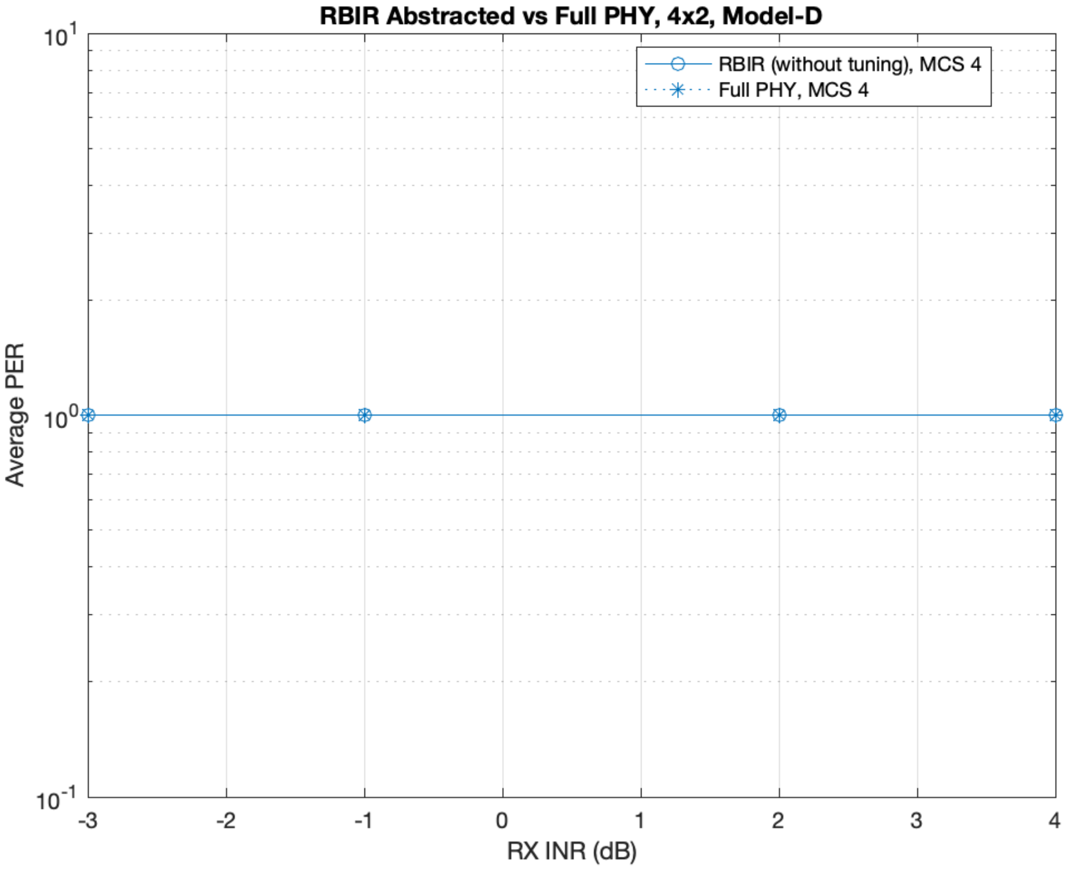
Consider simulating the PHY layer under RX INR = -3, -1, 2, 4 dBs. The transmitter is the interferer that transmit interference.
In the variable inr: Model-D, 4x2, MCS4 part, set the inr to be [-3 -1 2 4] (notice the bold part in the following code):

```
inr = {
    % Model-B
    [ ...
        {... % 1x1
    [-3:4:9,11], ...    % MCS 0
    [1:4:13], ...     % MCS 1
    [2:4:18], ...     % MCS 2
    [7:4:19,21], ...     % MCS 3
    [9:4:25], ...     % MCS 4
    [13:4:29], ...    % MCS 5
    [14:4:30], ...    % MCS 6
    [16:4:32,34], ...    % MCS 7
    [18:4:34,36] ... % MCS 8
    [18:4:38] ...     % MCS 9
    }; ...
        {... % 4x1
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
        {... % 4x2
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
         {... % 8x2
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
    ];

% Model-D
    [ ...
        {... % 1x1
    -3:3:9, ...     % MCS 0
    0:4:12, ...     % MCS 1
    2:4:18, ...     % MCS 2
    8:4:20, ...     % MCS 3
    11:4:23, ...    % MCS 4
    [14:4:26,28], ...    % MCS 5
    16:4:28, ...    % MCS 6
    18:4:30, ...    % MCS 7
```

```
21.5:4:37.5 ... % MCS 8
20:4:36 ...     % MCS 9
}; ...
    {... % 4x1
-4:2:2, ...   % MCS 0
5:2:15, ...   % MCS 1
9:2:21, ...   % MCS 2
12:2:24, ...  % MCS 3
12:2:16, ...  % MCS 4
20:3:32, ...  % MCS 5
22:3:34, ...  % MCS 6
24:3:39, ...  % MCS 7
27:3:40 ...   % MCS 8
29:3:44 ...   % MCS 9
}; ...
    {... % 4x2
[11,13,14,15], ...   % MCS 0
5:2:15, ...   % MCS 1
9:2:21, ...   % MCS 2
12:2:24, ...  % MCS 3
[-3 -1 2 4], ...  % MCS 4
20:3:32, ...  % MCS 5
22:3:34, ...  % MCS 6
24:3:39, ...  % MCS 7
27:3:40 ...   % MCS 8
29:3:44 ...   % MCS 9
}; ...
    {... % 8x2
1:2:10, ...   % MCS 0
5:2:15, ...   % MCS 1
9:2:21, ...   % MCS 2
12:2:24, ...  % MCS 3
[15,16,17,18,19,20,21,22], ...  % MCS 4
20:2:30, ...  % MCS 5
22:3:34, ...  % MCS 6
24:3:39, ...  % MCS 7
27:3:40 ...   % MCS 8
29:3:44 ...   % MCS 9
}; ...
] ...
};
```

After the above setup, go back to MU code LSC/1 Full PHY (for effective INR)/fullPHY.m
This is the main script of full PHY simulation. click on ‘Run’ in MATLAB.
The main script will call MU code LSC/1 Full PHY (for effective INR)/box0Simulation.m, the key function to realize full PHY simulation. This key function assumes ZF MIMO precoding and MMSE MIMO decoding. Users can comment out the ZF MMO precoding calculation part in box0Simulation.m to enable the default Fourier MIMO precoding.

After running the above full PHY simulation (takes about 1~3 hours), a file named “inrPer_Config192_Model-D_4-by-2_MCS4.mat” will be created.
This file includes 40000 {post-MIMO processing SINR matrix, packet error state} pairs for each RX SNR. The post-MIMO processing SINR matrices are in results{snrIdx}.sinrStore, and the packet error states are in results{snrIdx}.perStore, where snrIdx =1,2,3,4.



A figure named “RBIR Abstracted vs Full PHY, 4x2, Model-D” will also pop up.
This figure shows the the average PER vs RX INR(dB) of the full PHY simulation and RBIR prediction (without tuning). We can observe that the two curves approximately match, indicating that the full PHY simulation works well.

Step 1.3 (Full PHY simulation for a desired TX-desired RX pair plus an interferer-desired RX pair)
Goal: full PHY simulation for a desired TX-desired RX pair plus an interferer-desired RX pair (2 BSSs setup, corresponding to effective SINR) for optimizing the interference tuning parameter θ in our IEEE TCOM paper.

Folder:
MU code basic/1 Full PHY (multi interferers)

Open: MU code basic/1 Full PHY (multi interferers)/fullPHY.m
Config the same setup in MU code basic/1 Full PHY (multi interferers)/fullPHY.m as in MU code LSC/1 Full PHY (for effective SNR)/fullPHY.m. The only difference is that here, we need to set interference path loss in dB such that RX INR = RX SNR - 16dB (please notice the bold part of the code below).

```
%% Full PHY simulation setup
```

```
clear all
tStart = tic;
mcs = [4]; % Vector of MCS to simulate between 0 and 11
numTxRx = [4 2]; % Matrix of MIMO schemes, each row is [numTx numRx]
chan = "Model-D"; % String array of delay profiles to simulate
userIdx = 1; % User of investigation
ruIdx = 1; % RU of investigation
Nsts = 2; % Number of space-time streams
intPathlossdB = 16; % Interference path loss in dB scale
maxnumberrors = 40*1e3;  % The maximum number of packet errors at an SNR
point
maxNumPackets = 40*1e3; % The maximum number of packets at an SNR point
% maxnumberrors = 5*1e1;  % The maximum number of packet errors at an SNR
point
% maxNumPackets = 5*1e2; % The maximum number of packets at an SNR point

% Fixed PHY configuration for all simulations
cfgHE = wlanHEMUConfig(192); % Input 11ax allocation index
% The full RU assignment and allocation index lookup table is shown
% in the quick start guide
% Example: when allocation index = 24,
% then 1st RU has size 106, 2nd/3rd RU size = 52
for userIdxIter = 1:numel(cfgHE.User)
    cfgHE.User{userIdxIter}.APEPLength = 1000; % Payload length in bytes
end
```

Open: MU code basic/1 Full PHY (multi interferers)/getBox0SimParams.m

Consider simulating the PHY layer under RX SNR = 13, 16, 18, 20 dBs. That is, the desired receiver receives desired signal with RX SNR 13, 16, 18, 20 dBs. While the RX INR = RX SNR - 16dB, the desired receiver receives interference with RX INR -3, 0, 2, 4 dBs. The RX INR = -3dB and RX INR = 4dB cases are of interest in this example.

In the variable snr: Model-D, 4x2, MCS4 part, set the snr to be [13 16 18 20] (notice the bold part in the following code):

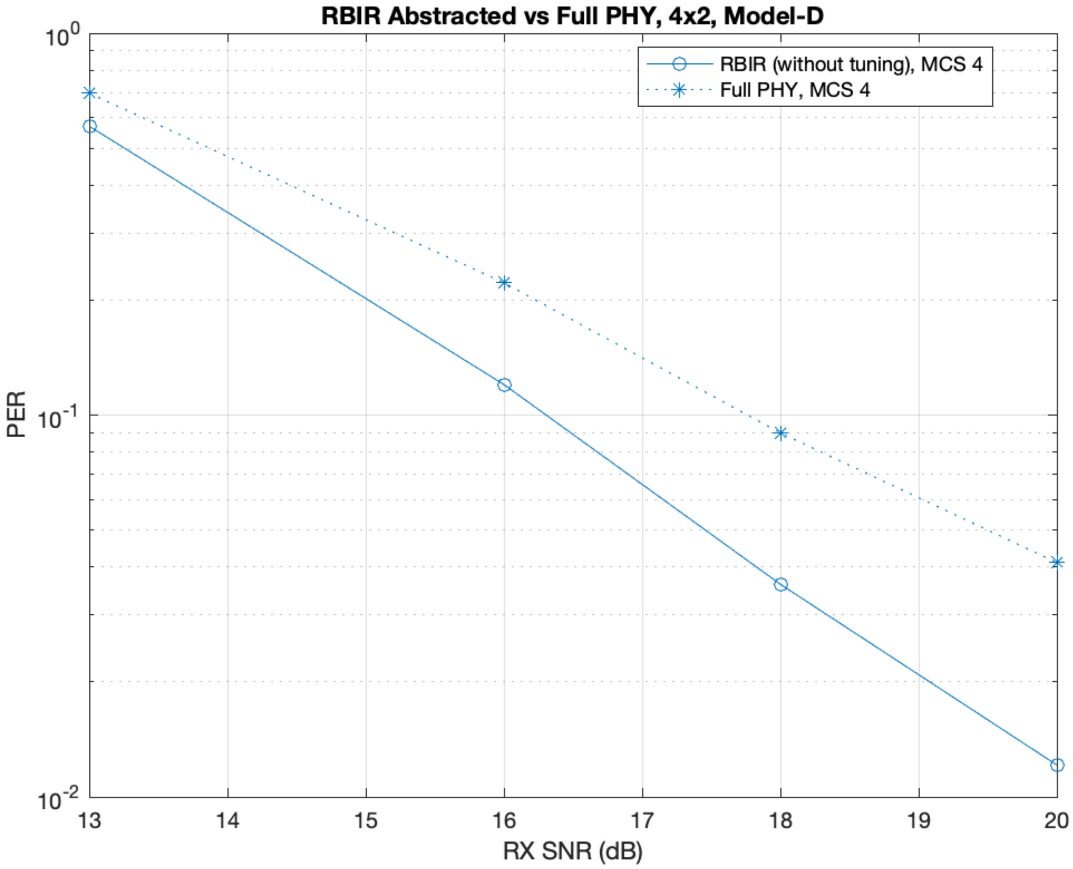
```
snr = {
    % Model-B
    [ ...
        {... % 1x1
    [-3:4:9,11], ...   % MCS 0
    [1:4:13], ...     % MCS 1
    [2:4:18], ...     % MCS 2
    [7:4:19,21], ...     % MCS 3
    [9:4:25], ...     % MCS 4
    [13:4:29], ...    % MCS 5
    [14:4:30], ...    % MCS 6
    [16:4:32,34], ...    % MCS 7
    [18:4:34,36] ... % MCS 8
    [18:4:38] ...     % MCS 9
    }; ...
        {... % 4x1
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
        {... % 4x2
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
         {... % 8x2
    1:3:17, ...  % MCS 0
    5:3:23, ...  % MCS 1
    9:4:30, ...  % MCS 2
    12:4:36, ... % MCS 3
    16:4:40, ... % MCS 4
    20:4:43, ... % MCS 5
    22:4:46, ... % MCS 6
    24:4:48, ... % MCS 7
    26:4:50 ...  % MCS 8
    29:4:54 ...  % MCS 9
    }; ...
    ];

% Model-D
    [ ...
        {... % 1x1
    -3:3:9, ...     % MCS 0
    0:4:12, ...     % MCS 1
    2:4:18, ...     % MCS 2
    8:4:20, ...     % MCS 3
    11:4:23, ...    % MCS 4
    [14:4:26,28], ...    % MCS 5
    16:4:28, ...    % MCS 6
    18:4:30, ...    % MCS 7
    21.5:4:37.5 ... % MCS 8
    20:4:36 ...     % MCS 9
    }; ...
        {... % 4x1
    -4:2:2, ...   % MCS 0
    5:2:15, ...   % MCS 1
    9:2:21, ...   % MCS 2
    12:2:24, ...  % MCS 3
    12:2:16, ...  % MCS 4
    20:3:32, ...  % MCS 5
    22:3:34, ...  % MCS 6
    24:3:39, ...  % MCS 7
    27:3:40 ...   % MCS 8
    29:3:44 ...   % MCS 9
    }; ...
        {... % 4x2
    [11,13,14,15], ...    % MCS 0
    5:2:15, ...   % MCS 1
    9:2:21, ...   % MCS 2
    12:2:24, ...  % MCS 3
    [13 16 18 20], ...   % MCS 4
    20:3:32, ...  % MCS 5
    22:3:34, ...  % MCS 6
    24:3:39, ...  % MCS 7
    27:3:40 ...   % MCS 8
    29:3:44 ...   % MCS 9
    }; ...
        {... % 8x2
    1:2:10, ...   % MCS 0
    5:2:15, ...   % MCS 1
    9:2:21, ...   % MCS 2
    12:2:24, ...  % MCS 3
    [10,14,17,19], ...   % MCS 4
    20:2:30, ...  % MCS 5
    22:3:34, ...  % MCS 6
    24:3:39, ...  % MCS 7
    27:3:40 ...   % MCS 8
    29:3:44 ...   % MCS 9
    }; ...
    ] ...
    };
```

After the above setup, go back to MU code basic/1 Full PHY (multi interferers)/fullPHY.m
This is the main script of full PHY simulation. click on ‘Run’ in MATLAB.
The main script will call MU code basic/1 Full PHY (multi interferers)/box0Simulation.m, the key function to realize full PHY simulation. This key function assumes ZF MIMO precoding and MMSE MIMO decoding. Users can comment out the ZF MMO precoding calculation part in box0Simulation.m to enable the default Fourier MIMO precoding.

After running the above full PHY simulation (takes about 3~5 hours), a file named “sinrPer_Config192_Model-D_4-by-2_MCS4.mat” will be created.

This file includes 40000 {post-MIMO processing SINR matrix, packet error state} pairs for each RX SNR. The post-MIMO processing SINR matrices are in results{snrIdx}.sinrStore, and the packet error states are in results{snrIdx}.perStore, where snrIdx =1,2,3,4.

A figure named “RBIR Abstracted vs Full PHY, 4x2, Model-D” will also pop up.
This figure shows the the average PER vs RX SNR(dB) of the full PHY simulation and RBIR prediction (without tuning). We can observe that the two curves approximately match, indicating that the full PHY simulation works well.



Step 2 (EESM parameter estimation):

Goal: optimizing EESM parameter (beta) using 40000 {post-MIMO processing SINR matrix, packet error state} pairs generated from full PHY simulation. This step is shown in Fig. 3 of our IEEE TCOM paper. The principle follows the traditional PHY layer abstraction flow plotted in Fig. 6 of our IEEE TCOM paper.

Step 2.1 (EESM parameter estimation for a desired TX-desired RX pair or an interferer-desired RX pair)

Folder:
MU code basic/2 EESM parameter optimization

Copy “snrPer_Config192_Model-D_4-by-2_MCS4.mat” generated in Step 1 into MU code LSC/2 EESM parameter optimization

Open: MU code LSC/2 EESM parameter optimization/eesmAbstractionPerVsEffSnr.m
This is the main script in MU code LSC/2 EESM parameter optimization.

Load “snrPer_Config192_Model-D_4-by-2_MCS4.mat”:

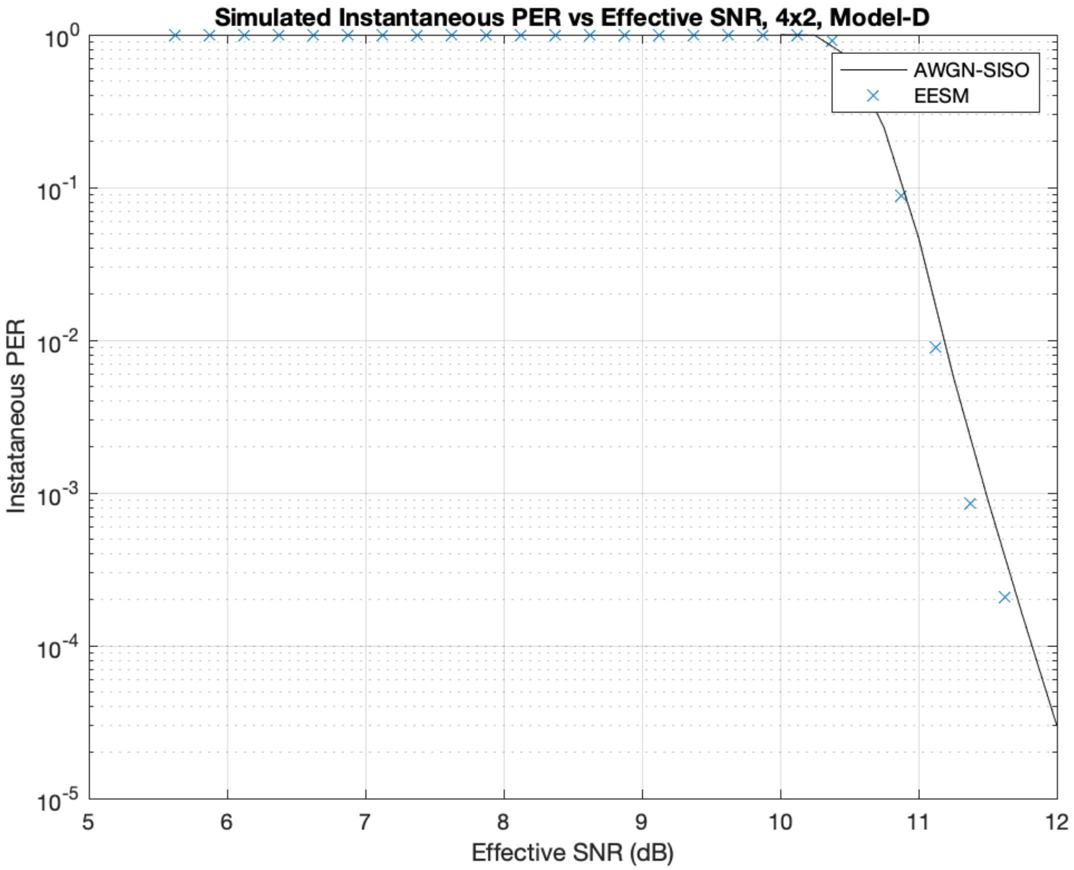
```
load('snrPer_Config24_Model-D_8-by-2_MCS4.mat');
```

For different MCS, set different initial EESM parameter for optimization.
The suggestions for choosing initial EESM parameter are shown in the following code.

```
% Tuning parameter in this function: EESM parameter - beta
% Suggestion: the higher MCS, the larger initialized beta
% Initial values for reference:
%   MCS: 0  -> beta : 1
%   MCS: 1  -> beta : 2
%   MCS: 2  -> beta : 1.5
%   MCS: 3  -> beta : 5
%   MCS: 4  -> beta : 7
%   MCS: 5  -> beta : 26
%   MCS: 6  -> beta : 33
%   MCS: 7  -> beta : 43
%   MCS: 8  -> beta : 111
%   MCS: 9  -> beta : 170
%   MCS: 10  -> beta : 410
%   MCS: 11  -> beta : 650
beta = 7;
```

As we use MCS4, set the initial EESM parameter value to be 7 according the above suggestion.

Opening eesmAbstractionPerVsEffSnr.m, click on ‘Run’ in MATLAB.
This main script minimizes MSE between 1) instantaneous PER VS EESM based effective SNR under the PHY layer configuration in step 1, and 2) instantaneous PER VS SNR under AWGN-SISO channel. The output is the optimized EESM parameter (beta): betaOptSig. This is realized in the awgnPerSnrFittingMse.m function and MATLAB fminsearch function.



After running eesmAbstractionPerVsEffSnr.m (takes about a few mins), a file named "eesmEffSnr_Config192_Model-D_4-by-2_MCS4.mat" will be created.
This file includes the optimized beta value stored in betaOptSig.

A figure named "Simulated Instantaneous PER vs Effective SNR, 4x2, Model-D" will also pop up. This figure shows 1) instantaneous PER VS EESM based effective SNR under the PHY layer configuration in step 1, and 2) instantaneous PER VS SNR under AWGN-SISO channel. If the two curves approximately match, then the traditional EESM L2S mapping works well.

Step 2.2 (EESM parameter estimation for a desired TX-desired RX pair plus an interferer-desired RX pair)

Folder:
MU code basic/2 EESM parameter optimization

Copy "sinrPer_Config192_Model-D_4-by-2_MCS4.mat" generated in Step 1 into MU code basic/2 EESM parameter optimization

Open: MU code basic/2 EESM parameter optimization/eesmAbstractionPerVsEffSinr.m
This is the main script in MU code basic/2 EESM parameter optimization.

Load "sinrPer_Config192_Model-D_4-by-2_MCS4.mat":

```
load('sinrPer_Config192_Model-D_4-by-2_MCS4.mat');
```

For different MCS, set different initial EESM parameter for optimization.
The suggestions for choosing initial EESM parameter are shown in the following code.

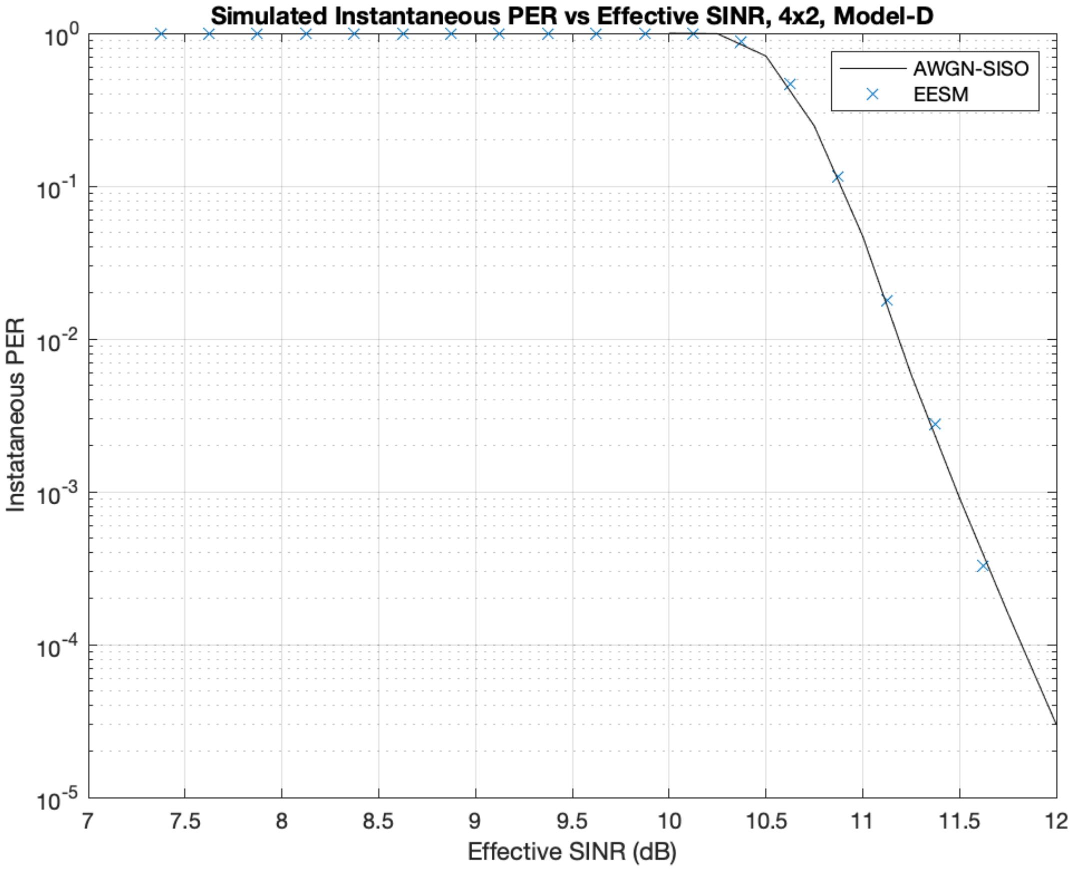
```
% Tuning parameter in this function: EESM parameter - beta
% Suggestion: the higher MCS, the larger initialized beta
% Initial values for reference:
%   MCS: 0  -> beta : 1
%   MCS: 1  -> beta : 2
%   MCS: 2  -> beta : 1.5
%   MCS: 3  -> beta : 5
%   MCS: 4  -> beta : 7
%   MCS: 5  -> beta : 26
%   MCS: 6  -> beta : 33
%   MCS: 7  -> beta : 43
%   MCS: 8  -> beta : 111
%   MCS: 9  -> beta : 170
%   MCS: 10  -> beta : 410
%   MCS: 11  -> beta : 650
beta = 7;
```

As we use MCS4, set the initial EESM parameter value to be 7 according the above suggestion.

Opening eesmAbstractionPerVsEffSinr.m, click on 'Run' in MATLAB.
This main script minimizes MSE between 1) instantaneous PER VS EESM based effective SINR under the PHY layer configuration in step 1, and 2) instantaneous PER VS SNR under AWGN-SISO channel. The output is the optimized EESM parameter (beta): betaOpt. This is realized in the awgnPerSnrFittingMse.m function and MATLAB fminsearch function.

After running eesmAbstractionPerVsEffSinr.m (takes about a few mins), a file named "eesmEffSinr_Config192_Model-D_4-by-2_MCS4.mat" will be created.
This file includes the optimized beta value stored in betaOpt.

A figure named "Simulated Instantaneous PER vs Effective SINR, 4x2, Model-D" will also pop up. This figure shows 1) instantaneous PER VS EESM based effective SINR under the PHY layer configuration in step 1, and 2) instantaneous PER VS SNR under AWGN-SISO channel. If the two curves approximately match, then the traditional EESM L2S mapping works well.



Step 3 (EESM-log-SGN-LSC interference parameter optimization):
Goal: optimize EESM-log-SGN-LSC interference parameter θ that is used in equation (6) of our IEEE TCOM paper. θ is optimized to minimize the MSE between the distribution estimated by equation (6) and the true effective SINR distribution under the PHY layer setup in step 1 with a specified RX SNR and a specified RX INR. Here, we choose such RX SNR to be 20dB and RX INR to be 4dB. For the effective SINR estimation to be a LSC solution, the optimal θ under the PHY layer setup in step 1 with 20dB RX SNR and 4dB RX INRs should provide accurate approximations of SINR distribution under the same PHY layer setup with other RX SNR and RX INRs.

Folder:
MU code LSC/3 EESM-log-SGN-LSC interference parameter optimization

Open: MU code LSC/3 EESM-log-SGN-LSC interference parameter optimization/getOptIntTuningParam.m
This is the main script in MU code LSC/3 EESM-log-SGN-LSC interference parameter optimization.

Copy:
“snrPer_Config192_Model-D_4-by-2_MCS4.mat” generated in step 1.1,
“inrPer_Config192_Model-D_4-by-2_MCS4.mat” generated in step 1.2,
“sinrPer_Config192_Model-D_4-by-2_MCS4.mat” generated in step 1.3,
“eesmEffSnr_Config192_Model-D_4-by-2_MCS4.mat” generated in step 2.1,
“eesmEffSinr_Config192_Model-D_4-by-2_MCS4.mat” generated in step 2.2
into MU code LSC/3 EESM-log-SGN-LSC interference parameter optimization.

Load all these 5 files MU code LSC/3 EESM-log-SGN-LSC interference parameter optimization/getOptIntTuningParam.m:

```
%% Load effective SNR
load('snrPer_Config192_Model-D_4-by-2_MCS4.mat')
load('eesmEffSnr_Config192_Model-D_4-by-2_MCS4.mat’)
...
%% Load effective INR for the 1st interferer
load('inrPer_Config192_Model-D_4-by-2_MCS4.mat')
load('eesmEffSnr_Config192_Model-D_4-by-2_MCS4.mat')
…
%% Load effective SINR for calibration
load('sinrPer_Config192_Model-D_4-by-2_MCS4.mat')
load('eesmEffSinr_Config192_Model-D_4-by-2_MCS4.mat’)
```

Set correct SNR/INR index:
We need effective SNR under RX SNR = 20dB. This corresponds to the 8th RX SNR index in the RX SNRs [9 11 13 15 17 18 19 20] set in step 1.1:

```
snrIdxSig = 8;
```

We need effective INR under RX INR = 4dB. This corresponds to the 8th RX SNR index in the RX SNRs [-3 -1 2 4] set in step 1.2:

```
inrIdx = 4;
```

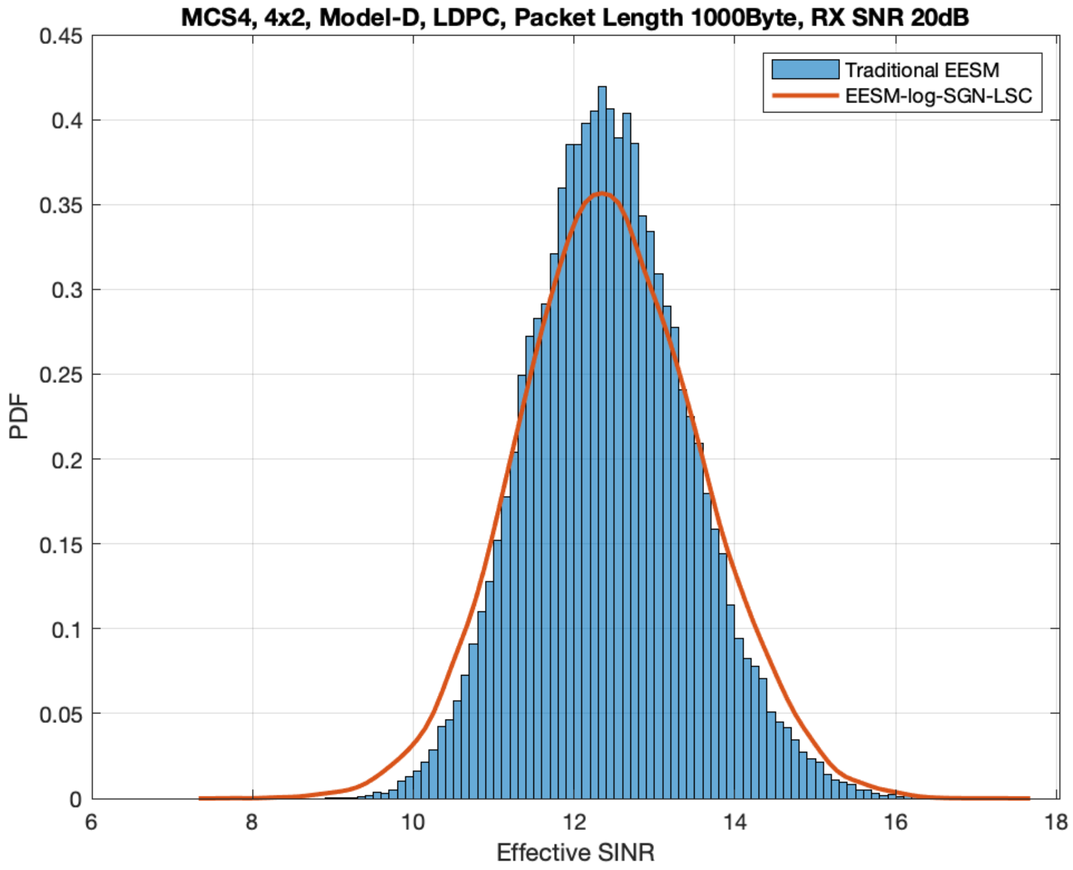
We need effective SINR under RX SNR = 20dB. This corresponds to the 4th RX SNR index in the RX SNRs [13 16 18 20] set in step 1.3:

```
snrIdx = 4;
```

Opening getOptIntTuningParam.m, click on ‘Run’ in MATLAB.
This script minimizes the MSE between the distribution estimated by equation (6) and the true effective SINR distribution under the PHY layer setup in step 1. The key function is called sinrModelFittingMse.m and fminbnd in MATLAB.

After running getOptIntTuningParam.m (takes about a few mins), a figure named “MCS4, 4x2, Model-D, LDPC, Packet Length 1000Byte, RX SNR 20dB” will pop up. This figure shows

1) EESM effective SINR obtained by traditional EESM L2S mapping, and
2) EESM effective SINR obtained by EESM-log-SGN-LSC in equation (6) of our IEEE TCOM paper.



The optimized interference parameter is shown in thetaOpt:

```
thetaOpt =

   0.6434
```

Step 4 (EESM-log-SGN-LSC implementation):

Goal: Implement EESM-log-LSC in Fig. 11 of our IEEE TCOM paper. Validate that the optimal θ under the PHY layer setup in step 1 with 20dB RX SNR and 4dB RX INRs provides accurate approximations of SINR distribution under the same PHY layer setup with other RX SNR and RX INRs.

Folder:
MU code LSC/4 EESM-log-SGN-LSC method
Open: MU code LSC/4 EESM-log-SGN-LSC method/eesmLogSGNLSC.m
This is the main script in MU code LSC/4 EESM-log-SGN-LSC method.

Copy:
“snrPer_Config192_Model-D_4-by-2_MCS4.mat” generated in step 1.1,
“inrPer_Config192_Model-D_4-by-2_MCS4.mat” generated in step 1.2,
“sinrPer_Config192_Model-D_4-by-2_MCS4.mat” generated in step 1.3,
“eesmEffSnr_Config192_Model-D_4-by-2_MCS4.mat” generated in step 2.1,

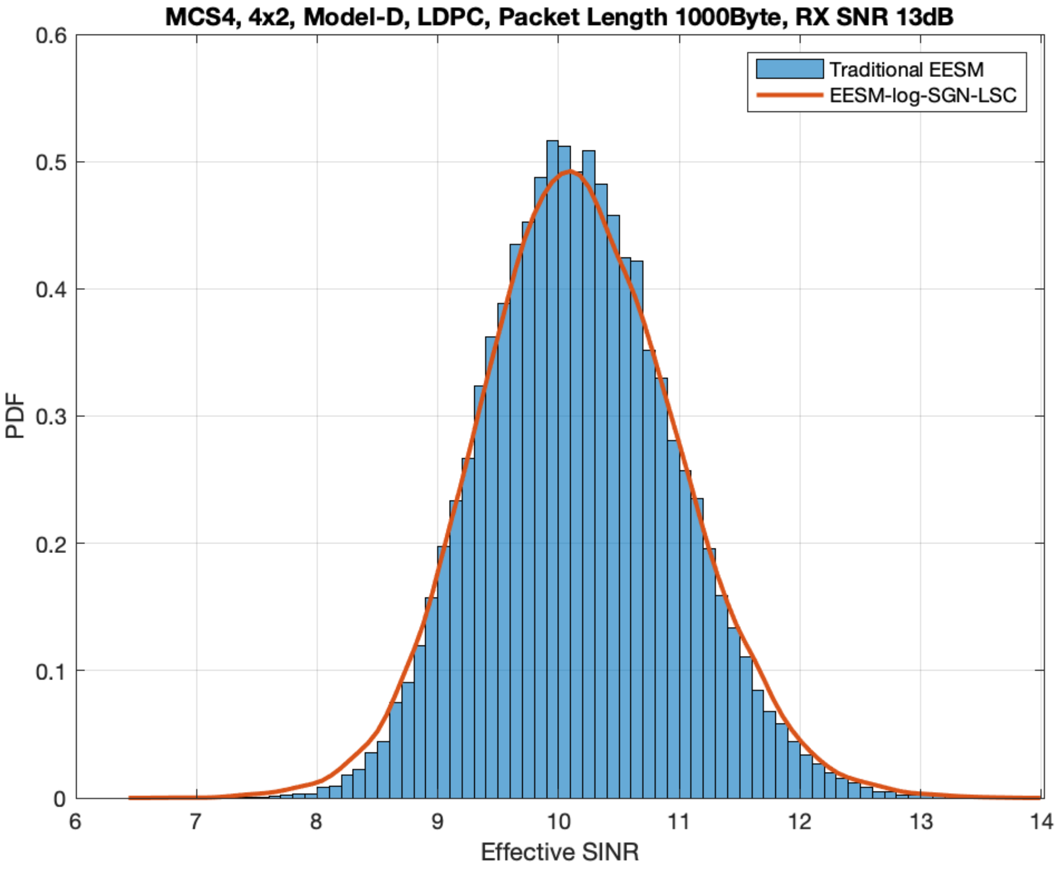
“eesmEffSinr_Config192_Model-D_4-by-2_MCS4.mat” generated in step 2.2 into MU code LSC/4 EESM-log-SGN-LSC method.

Load all these 5 files in MU code MU code LSC/4 EESM-log-SGN-LSC method/ eesmLogSGNLSC.m:

```
%% Load effective SNR
load('snrPer_Config192_Model-D_4-by-2_MCS4.mat')
load('eesmEffSnr_Config192_Model-D_4-by-2_MCS4.mat’)
...
%% Load effective INR for the 1st interferer
load('inrPer_Config192_Model-D_4-by-2_MCS4.mat')
load('eesmEffSnr_Config192_Model-D_4-by-2_MCS4.mat')
…
%% Load effective SINR for calibration
load('sinrPer_Config192_Model-D_4-by-2_MCS4.mat')
load('eesmEffSinr_Config192_Model-D_4-by-2_MCS4.mat’)
```

Set correct SNR/INR index:
We need effective SNR under RX SNR = 13dB. This corresponds to the 3rd RX SNR index in the RX SNRs [9 11 13 15 17 18 19 20] set in step 1.1:

```
snrIdxSig = 3;
```



We need effective INR under RX INR = 4dB. This corresponds to the 8th RX SNR index in the RX SNRs [-3 -1 2 4] set in step 1.2:

```
inrIdx = 1;
```

We need effective SINR under RX SNR = 20dB. This corresponds to the 4th RX SNR index in the RX SNRs [13 16 18 20] set in step 1.3:

```
snrIdx = 1;
```

Load thetaOpt obtained from step 3:

```
thetaOpt = 0.6434;
```

Opening eesmLogSGNLSC.m, click on ‘Run’ in MATLAB.
This script implements EESM-log-SGN-LSC in Fig. 11 of our IEEE TCOM paper.

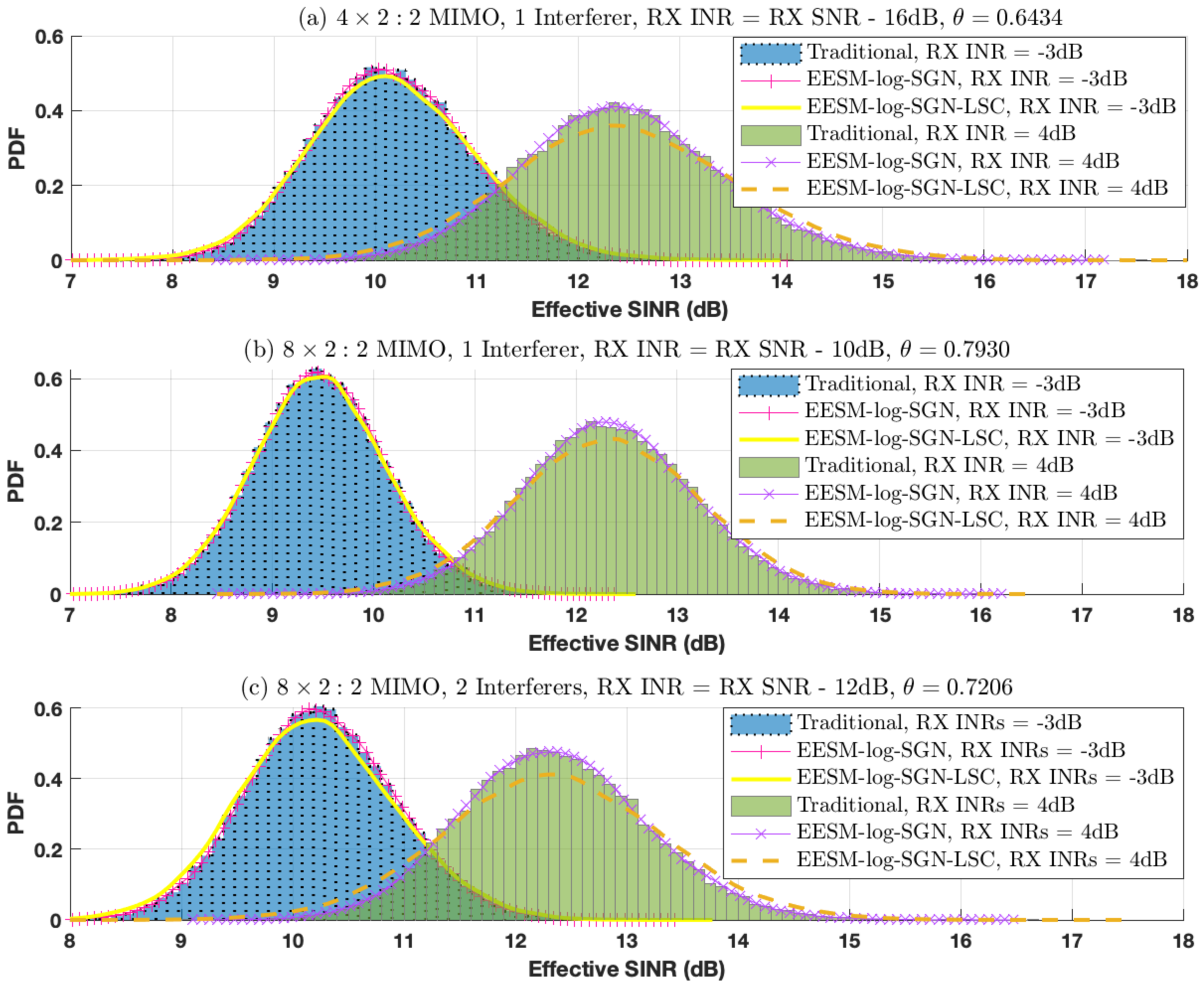
After running eesmLogSGNLSC.m (takes about a few mins), a figure named “MCS4, 4x2, Model-D, LDPC, Packet Length 1000Byte, RX SNR 13dB” will pop up. This figure shows

1) EESM effective SINR obtained by traditional EESM L2S mapping, and
2) EESM effective SINR obtained by EESM-log-SGN-LSC in equation (6) of our IEEE TCOM paper.

From the above figure, we can see that thetaOpt is also very suitable for this case. This validates that the optimal θ under the PHY layer setup in step 1 with 20dB RX SNR and 4dB RX INRs provides accurate approximations of SINR distribution under the same PHY layer setup with other RX SNR and RX INRs.

Final result:

Putting all the effective SINR histogram, EESM-log-SGN-LSC approximated PDF generated in step 4, as well as adding EESM-log-SGN approximated PDF (following the steps in MATLAB example 1), we have the following figure (Fig. 12 in our IEEE TCOM paper).


(a) $4 \times 2 : 2$ MIMO, 1 Interferer, RX INR = RX SNR - 16dB, $\theta = 0.6434$

(b) $8 \times 2 : 2$ MIMO, 1 Interferer, RX INR = RX SNR - 10dB, $\theta = 0.7930$

(c) $8 \times 2 : 2$ MIMO, 2 Interferers, RX INR = RX SNR - 12dB, $\theta = 0.7206$