A quick start of the MATLAB code of EESM-log-AR

Author: Sian Jin
sianjin@uw.edu

This MATLAB codes consist of 4 folders. The steps to obtain full PHY simulation results, traditional EESM L2S mapping results and EESM-log-AR results are shown below.
The code is labeled in orange color, the folder is labeled in green color and the file is labeled in blue color.

Step 1:
Open folder 1: uncorrelatedFullPHY

Open fullPHY.m

Set the MCS, MIMO dimension, channel type in the following codes:

```
mcs = [4]; % Vector of MCS to simulate between 0 and 9
numTxRx = [4 2]; % Matrix of MIMO schemes, each row is [numTx numRx]
chan = "Model-D"; % String array of delay profiles to simulate
```

Set RX SNR and number of space time streams in getBox0SimParams.m:

```
snr = {…
}
…
sp.Config.NumSpaceTimeStreams = 2;
```

Then, run the fullPHY.m

Obtain the saved file:
snrPer_CBW20_Model-D_4-by-2_MCS4.mat

Step 2:
Open folder 2: eesmParameterGeneration

Open eesmAbstractionPerVsEffSnr.m

Load the file obtained from step 1:

```
load('snrPer_CBW20_Model-D_4-by-2_MCS4.mat');
```

Run eesmAbstractionPerVsEffSnr.m

Obtain the saved file:
eesmEffSnr_CBW20_Model-D_2-by-1_MCS4.mat

This file provides the optimized EESM tuning parameter: betaOpt.

Step 3:
Open folder 3: correlatedFullPHY and EESM

Open validation.m

Set the MCS, MIMO dimension, channel type in the following codes:

mcs = [4]; % Vector of MCS to simulate between 0 and 9
numTxRx = [4 2]; % Matrix of MIMO schemes, each row is [numTx numRx]
chan = "Model-D"; % String array of delay profiles to simulate

Set beta equal to the betaOpt obtained from the output of step 2:
eesmEffSnr_CBW20_Model-D_2-by-1_MCS4.mat

Set RX SNR and number of space time streams in getBox0SimParams.m:

snr = {…
}
…
sp.Config.NumSpaceTimeStreams = 2;

Run validation.m

Obtain the saved file:
snrPer_CBW20_Model-D_4-by-2_MCS4.mat

This file contains the output of traditional EESM L2S mapping and full PHY result under correlated channel.

Step 4:
Open folder 4: EESM-log-AR

Open logArModel.m to obtain EESM-log-AR under ML parameters

Load the file obtained from step 3:

load('snrPer_CBW20_Model-D_4-by-2_MCS4.mat')

Set the index of RX SNR for simulation:

snrIdx = 2;

Run logArModel.m

The simulation results and validations of EESM-log-AR under ML parameters pop up in figures.

Open logARLinearInterpolation.m to obtain EESM-log-AR under LI parameters

Load the two dataset from two different RX SNRs:

%% Load parameters at gamma_1 and gamma_2
arLags = 10;

```
innovdist = struct('Name',"Gaussian");
load('snrPer_CBW20_Model-D_4-by-2_MCS4.mat')
% Load data at gamma_1
snrIdx = 1;
effSnrVec1 = results{snrIdx}.effSnrVec;
effSnrVecdB1 = effSnrVec1';
effSnrVecLinear1 = 10.^(effSnrVecdB1/10); % Transfer dB into linear
effSnrVecLog1 = log(effSnrVecLinear1); % Transfer linear into log domain
Mdl1 = arima('ARLags',1:arLags,'Distribution',innovdist);
EstMdl1 = estimate(Mdl1,effSnrVecLog1)
constLI1 = EstMdl1.Constant;
varLI1 = EstMdl1.Variance;
arLI1 = EstMdl1.AR;
% Load data at gamma_2
snrIdx = 3;
effSnrVec2 = results{snrIdx}.effSnrVec;
effSnrVecdB2 = effSnrVec2';
effSnrVecLinear2 = 10.^(effSnrVecdB2/10); % Transfer dB into linear
effSnrVecLog2 = log(effSnrVecLinear2); % Transfer linear into log domain
Mdl2 = arima('ARLags',1:arLags,'Distribution',innovdist);
EstMdl2 = estimate(Mdl2,effSnrVecLog2)
constLI2 = EstMdl2.Constant;
varLI2 = EstMdl2.Variance;
arLI2 = EstMdl2.AR;
```

Load target data set and set the index of target RX SNR for simulation:

```
%% Loading effective SNR in dB
load('snrPer_CBW20_Model-D_4-by-2_MCS4.mat')
snrIdx = 2;
```

Run logARLinearInterpolation.m

The simulation results and validations of EESM-log-AR under LI parameters pop up in figures.