University of Oulu

Faculty of Information Technology and Electrical Engineering - ITEE

Statistical Signal Processing 1

MATLAB Task #4

Group: 14

Students:

- AZZAZ Aissa (Number: 2207335)
- BOULFRAD Mourad (Number: 2207592)

Due date: October 20th 2022

Bayesian estimation error comparison

The captured data follows the model $X = H\theta + W$ where $\theta = \{-1, 1\}$ and $H \sim \mathcal{CN}(0, \sigma^2 I)$ with $\sigma^2 = 1$. The SNR (signal to noise ratio) of the samples is given by

$$\text{EbN0} = SNR = \frac{var\{X\}}{var\{W\}} = \frac{\sigma^2}{\sigma_W^2}$$

Thus, noise variance is

$$\sigma_W^2 = \frac{\sigma^2}{SNR} = \frac{\sigma^2}{\text{EbN0}}$$

For H, it's a zero mean complex random distribution consisting of two joint complex random variables represent the real (in phase "I") and imaginary (quadrature "Q") components both having

$$\mu_I = \mu_Q = 0\ ;\ \ \sigma_I^2 = \sigma_Q^2 = \frac{\sigma^2}{2}$$

The parameter to be estimated is θ (in the code "A"), which represent the output of a signum function applied to a standard Gaussian distribution (.i.e with mean zero and unity variance). Thus, $\theta \sim \mathcal{N}(0, 1I)$ (.i.e the priori pdf of $\theta = A$ follows a standard Gaussian distribution).

From equations (15.65) and (15.67), the estimators are given by

$$\hat{\theta}_{LS} = \left(H^H H\right)^{-1} H^H X$$

$$\hat{\theta}_{LMMSE} = \mu_\theta + \left(C_{\theta\theta}^{-1} + H^H C_W^{-1} H\right)^{-1} C_W^{-1} \left(X - H\mu_\theta\right) = \left(C_{\theta\theta}^{-1} + H^H C_W^{-1} H\right)^{-1} C_W^{-1} X$$

$$\hat{\theta}_{MSE} = diag\left\{\left(C_{\theta\theta}^{-1} + H^H C_W^{-1} H\right)^{-1}\right\}$$

The code below calculates the stated estimated values fro the various channel realizations, and then it calculates the average square error for each estimator. The code generates Figure 1, which presents the changes in estimation error for different SNR values. In contrast, to the Bayesian estimators, the LS estimator doesn't depend on a priori pdf of the parameter θ and thus it has a constant output and constant error that is independent from the SNR values. The LMMSE estimator takes into account the priori knowledge about θ and properly weighs its estimation with the estimation obtained from the captured samples. An increase in SNR results in decrease of noise variance and thus the LMMSE would have better samples to use in the estimation leading to a small error and vice versa, as clearly shown in Figure 1.

```
close all ; clear ; clc ;
rng(0) ; % reset the random number generator (for reproducibility)
% format long eng

MC = 1000;   % number of Monte Carlo runs for data
MC_channel = 1000; % number of Monte Carlo runs for channel
N_TX = 2; % number of TX antennas
N_RX = 2; % number of RX antennas

EbN0dB = 0:0.1:10;  % SNR vector in dB
NoSNRPoints = length(EbN0dB); % length of SNR vector
EbN0 = 10.^(EbN0dB./10);  % SNR in linear scale
C = ( 1./EbN0 )./2 ; % Find noise variance per I and Q component
sig = sqrt(C) ; % Find noise standard deviation per I and Q component

A = sign(randn(N_TX,MC));  % random data for all runs, the same for all SNR points
C_A = 1*eye(N_RX, N_TX) ;  % Find data variance (variance of A)
ASE_LS = zeros(NoSNRPoints,MC_channel);  % initialize the LS average square error (ASE)
ASE_LMMSE = zeros(NoSNRPoints,MC_channel);  % LMMSEE ASE
MSE_LMMSE = zeros(NoSNRPoints,MC_channel);  % theoretical MSE for LMMSE

for i = 1:NoSNRPoints     % generate noise realization for SNR values using
```

```
    for c = 1:MC_channel  % generate channel realization

        % generate complex channel response with total variance 1
        H= sqrt(1/2)*( randn(N_RX, N_TX) + 1j*randn(N_RX, N_TX) ) ;

        % generate complex noise with different sig for different SNR
        w= sig(i)*( randn(N_RX, 1) + 1j*randn(N_RX, 1) ) ;
        X = H*A + w ;

        C_W = (C(i)*eye(N_RX, N_TX)) ;

        % Implement LS estimator and take only real part which contain the data
        LS = real( (inv(C_A)+H'*inv(C_W)*H)*H'*C_W*X  )  ;

        % Implement LMMSE estimator and take only real part which contain the data
        LMMSE = real( inv(H'*H)*H'*X  )  ;

        % Calculate average square error for LS
        ASE_LS(i,c) = mean((abs(LS-A)).^2,'all');

        % Calculate average square error for LMMSE
        ASE_LMMSE(i,c) = mean((abs(LMMSE-A)).^2,'all');

        % Implement theoretical MSE for LMMSE by taking real parts of diagonal elements
        MSE(i,c) = sum(abs(diag(real( inv(C_A)+H'*inv(C_W)*H ))));
    end
end

MSE_C = mean(MSE,2) ; % Take average over different channel realization for MSE
ASE_LMMSE_C = mean(ASE_LMMSE,2) ; % Take average over different channel realization for ASE_LMMSE
ASE_LS_C = mean(ASE_LS,2) ; % Take average over different channel relaization for ASE_LS

fig1 = figure ;
semilogy(EbN0dB,ASE_LMMSE_C,'r-');
hold on
semilogy(EbN0dB,ASE_LS_C,'g--');
semilogy(EbN0dB,MSE_C,'b-.');
hold off
grid minor
xlabel('EbN0dB (dB)') ;
ylabel('Estimation Error') ;
legend('ASE_{LMMSE}','ASE_{LS}','MSE_{C}','Location','best') ;
```

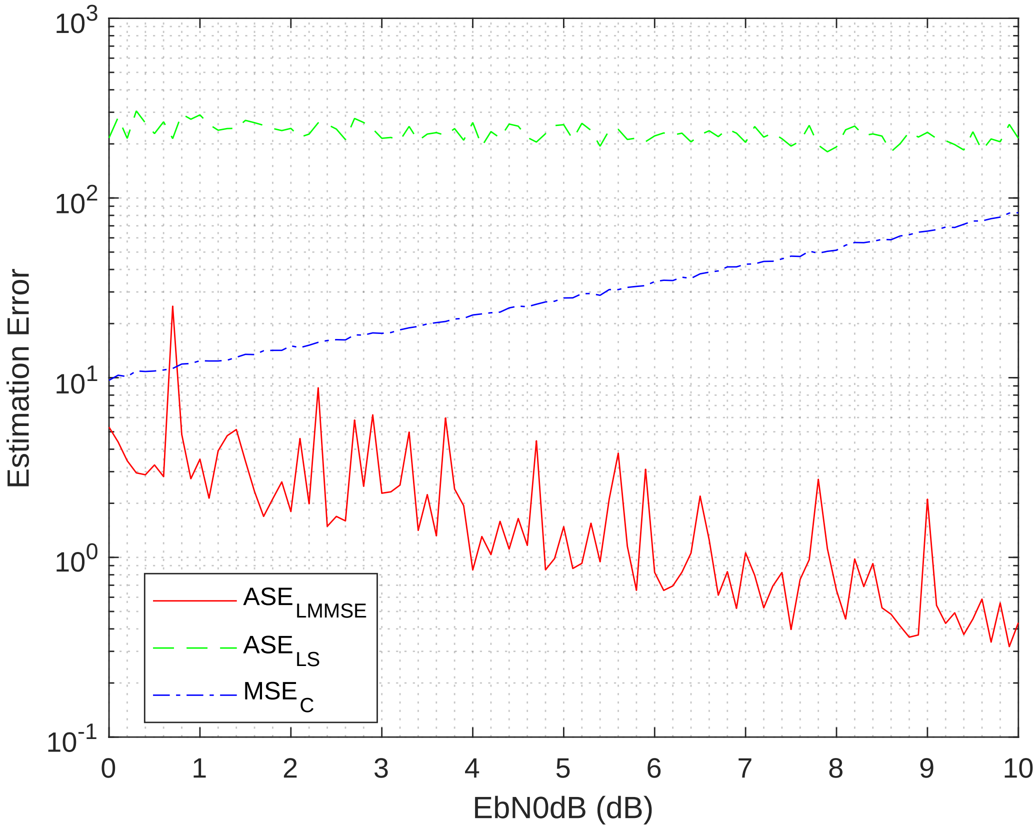

Figure 1: Estimation error for the various estimators