University of Oulu

Faculty of Information Technology and Electrical Engineering - ITEE

Statistical Signal Processing 1

MATLAB Task #1

Group: 14

Students:

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

Due date: September 20th 2022

1. Showing that the estimator $\hat{\theta}$ is efficient by Monte Carlo (MC) simulation

The data $x\left[n\right]=\theta n+w\left[n\right]$for $n=0,1,2,\ldots N-1$ are observed, where $w\left[n\right]$ is zero-mean WGN with variance σ^2.

The Cramer-Rao lower bound is $CRLB_{\hat{\theta}} = \frac{\sigma^2}{\sum_{n=0}^{N-1} n^2}$.

The code below starts by defining the simulation parameters (MC, θ, N, σ^2), then it calculates the signal part θn, the noise part $w[n]$, and the data samples $x[n]$. After that, it calculates the estimator $\theta = \frac{\sum_{n=0}^{N-1} x[n]n}{\sum_{n=0}^{N-1} n^2}$, its statistics, and the Cramer-Rao lower bound $CRLB_{\hat{\theta}} = \frac{\sigma^2}{\sum_{n=0}^{N-1} n^2}$. Then, the program compares the absolute error of the estimator's statistics ($E\{\hat{\theta}\}$, $\sigma^2_{\hat{\theta}}$) to the CRLB, and it decides that it is an efficient estimator if the error is less than some percentage (3% in this case).

```
close all ; clear all ; clc ;


MC = 10000 ; % Number of Monte Carlo loops
N = 10 ; % Number of samples in the data
n = [0:N-1] ; % row vector to represent the index of each sample
noise_var = 2 ; % Variance of the AWGN
theta = 10 ; % Mean

noise = 0 + randn(MC,N)*sqrt(noise_var) ; % w[n] ==> AWGN with zero mean and variance = noise_var
signal = theta*n ; % theta*n ==> signal of interset
x = signal + noise ; % captured samples with MC rows and N columns

estimate = sum(x.*n,2)/sum(n.^2) ; % (sum of weighted samples / sum of squared indices)

mean_estimate = mean(estimate) % mean of the estimate along MC loops
```

```
mean_estimate =
  10.001013481756980
```

```
var_estimate = var(estimate) % variance of the estimate along MC loops
```

```
var_estimate =
   0.007026093409173
```

```
CRLB = (noise_var)/(sum(n.^2)) % (variance / sum of squared indecies )
```

```
CRLB =
   0.007017543859649
```

```
% The estimate is efficient ("good enough") if the error in its statistics is
% ... less than some percentage (3% in this test)
if abs((mean_estimate-theta)/theta)<.03 && abs((var_estimate-CRLB)/CRLB)<.03
    disp('The estimator is an efficient estimator ')
else
    disp('The estimator is not an efficient estimator ')
end
```

```
The estimator is an efficient estimator
```

After running the code, we notice that $E\{\hat{\theta}\} \approx \theta = 10$ and $\sigma^2_{\hat{\theta}} \approx CRLB_{\hat{\theta}}$ with and absolute error that is less than 3%, hence, the estimator $\hat{\theta}$ is efficient.

2. Generating figure that shows that the simulated and theoretical probability density functions agree for the previous estimator

The code below generates Figure 1, by first obtaining the simulated pdf (in blue stairs) using the *histogram* function, and then taking its x-axis values and using them to plot (in red) the theoretical pdf of the estimator $\hat{\theta}$ which follows a Gaussian distribution with mean θ and variance of $\sigma^2_{\hat{\theta}} = CRLB_{\hat{\theta}}$ (.i.e $\hat{\theta} \sim \mathcal{N}(\theta, CRLB_{\hat{\theta}})$). Also, the code highlights (in green) the true value of the parameter θ used in the simulation.

```
figure

H = histogram(estimate,'Normalization','pdf','DisplayStyle',"stairs") ;
hold on
x2 = linspace(H.BinLimits(1),H.BinLimits(2),100) ; % range of x for theoretical results

theor_PDF = normpdf(x2,theta,sqrt(CRLB)) ;
plot(x2, theor_PDF,'r-','Linewidth',2) ;

xline(theta,'-g','Linewidth',2) ;

grid minor
legend('Simulated','Theoretical','True Value of $\theta$','interpreter','latex','Location','best') ;
xlabel('$\hat{\theta}$','interpreter','latex') ;
ylabel('PDF','interpreter','latex') ;
```

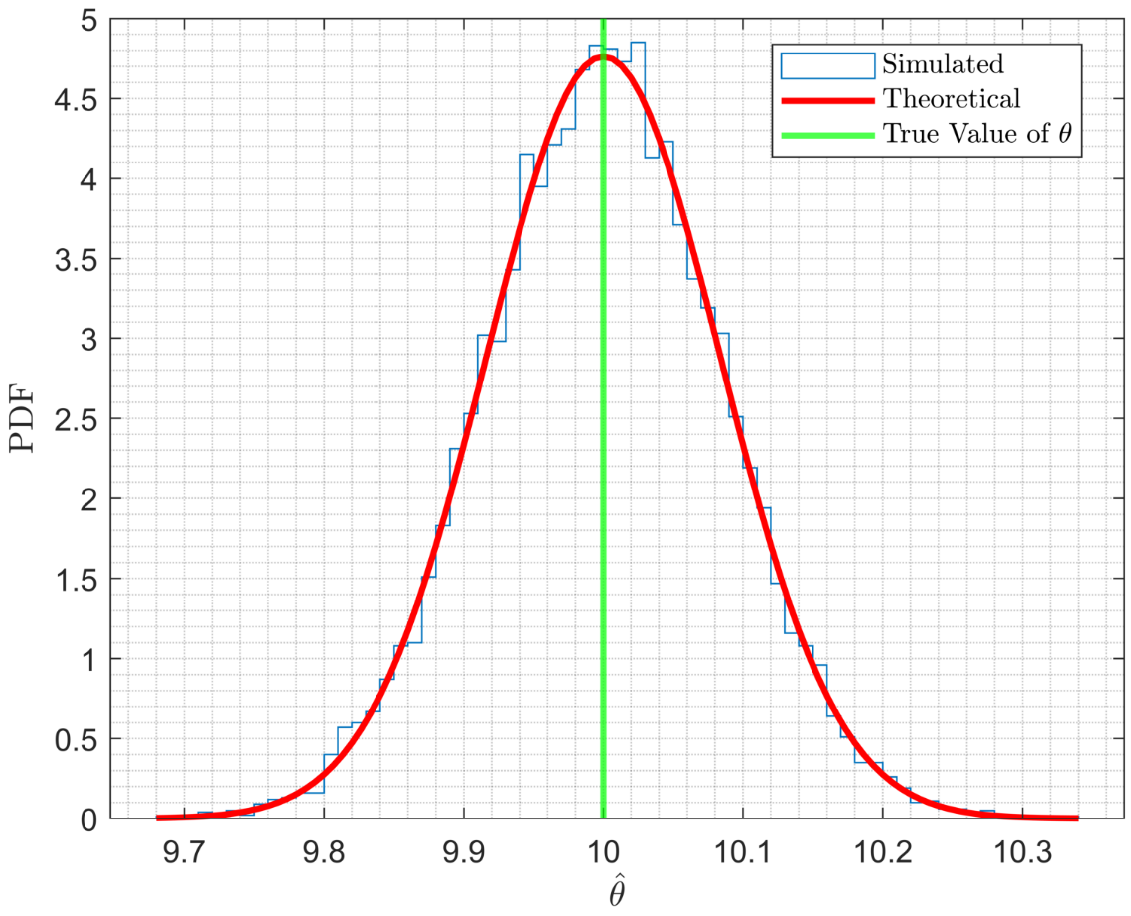

Figure 1: The simulated and theoretical PDFs of the estimator $\hat{\theta}$

3. Determining the effect of one and very large sample count N

When $N = 1$, the estimator gets only one sample $x[0]$ to use, and thus it gives erroneous values due to the n^2 term in the denominator ($\hat{\theta} \to \infty$) and the same happens with its variance and CRLB ($CRLB_{\hat{\theta}}, \sigma^2_{\hat{\theta}} \to \infty$).

When we increase the number of data samples ($N \to \infty$), the estimator gives more accurate results ($E\{\hat{\theta}\} \to \theta$), and its variance and CRLB vanish($CRLB_{\hat{\theta}}, \sigma^2_{\hat{\theta}} \to 0$), and we get less deviations from the true value of θ.

To investigate this effect using Monte Carlo simulation, the code below benchmarks the estimator $\hat{\theta}$ against different values of N. The code starts as previously by defining the simulation parameters ((MC, θ, σ^2)) but it varies the number of samples N from $10^0 = 1$ sample to 10^7 samples with power of 10 steps. Then for each value of N, it calculates the estimator statistics and its CRLB as in the first part and displays these values. Here, we notice that with $N = 1$ (the first value) gives a wrong estimate and an infinite CRLB, and as we increase N, the mean of the estimator approaches the parameter $theta$ and the variance and CRLB approach zero. To better visualize this effect, the code plots two subplots, Figure 2 both on the loglog scale to exaggerate the small values as $N \to \infty$. The subplot to the left, shows the effect of N on the mean of the estimator $E\{\hat{\theta}\}$ and the one to the right shows the effect of N on $CRLB_{\hat{\theta}}$ and the variance of the estimator $\sigma^2_{\hat{\theta}}$.

```
% close all ; clear all ; clc ;
MC = 10 ; % Number of Monte Carlo loops (reduced to speed up the iterations)
N = 10.^[0:7] ; % Number of samples in the data (swept)
noise_var = 2 ; % Variance of the AWGN
theta = 10 ; % Mean

for ii = 1:length(N) % used as an index to sweep N
    n = [0:N(ii)-1] ; % row vector to represent the index of each sample

    x = theta*n + randn(MC,N(ii))*sqrt(noise_var)  ; % captured samples with MC rows and N(ii) columns

    estimate = sum(x.*n,2)/sum(n.^2) ; % (sum of weighted samples / sum of squared indices)

    mean_estimate(ii) = mean(estimate) ; % mean of the estimate along MC loops
    var_estimate(ii) = var(estimate) ; % variance of the estimate along MC loops

    CRLB(ii) = (noise_var)/(sum(n.^2)) ; % (variance / sum of squared indices )
end
format long
mean_estimate
```

```
  mean_estimate = 1x8
      NaN  10.025149688833483   9.998238217515659   9.999998553326240   9.999998585471937
          10.000000018113194   9.999999999526327  10.000000000026077
```

```
var_estimate
```

```
  var_estimate = 1x8
      NaN   0.006266279213217   0.000012346504144   0.000000006148877   0.000000000005927
          0.000000000000008   0.000000000000000   0.000000000000000
```

```
CRLB
```

```
  CRLB = 1x8
      Inf   0.007017543859649   0.000006091061367   0.000000006009011   0.000000000006001
          0.000000000000006   0.000000000000000   0.000000000000000
```

```
% plotting spagetti
subplot(1,2,1)
loglog(N,mean_estimate)
hold on
yline(theta,'k--')
hold off
```

```
ylim([0.9999*min(mean_estimate) 1.0001*max(mean_estimate)])
grid minor
title('$E\{\hat{\theta\}}$ vs $N$','interpreter','latex')
xlabel('$N$','interpreter','latex')
xticks(N)
ylabel('$E\{\hat{\theta\}}$','interpreter','latex')
legend('$E\{\hat{\theta\}}$','True value of $\theta$','interpreter','latex')

subplot(1,2,2)
loglog(N,var_estimate)
hold on
loglog(N,CRLB)
hold off
grid minor
title('$\sigma^2_{\hat{\theta}}$ vs $N$','interpreter','latex')
xlabel('$N$','interpreter','latex')
xticks(N)
ylabel('$CRLB_{\hat{\theta}} ; \sigma^2_{\hat{\theta}}$','interpreter','latex')
legend('$\sigma^2_{\hat{\theta}}$','$CRLB_{\hat{\theta}}$','interpreter','latex','Location','best')
```

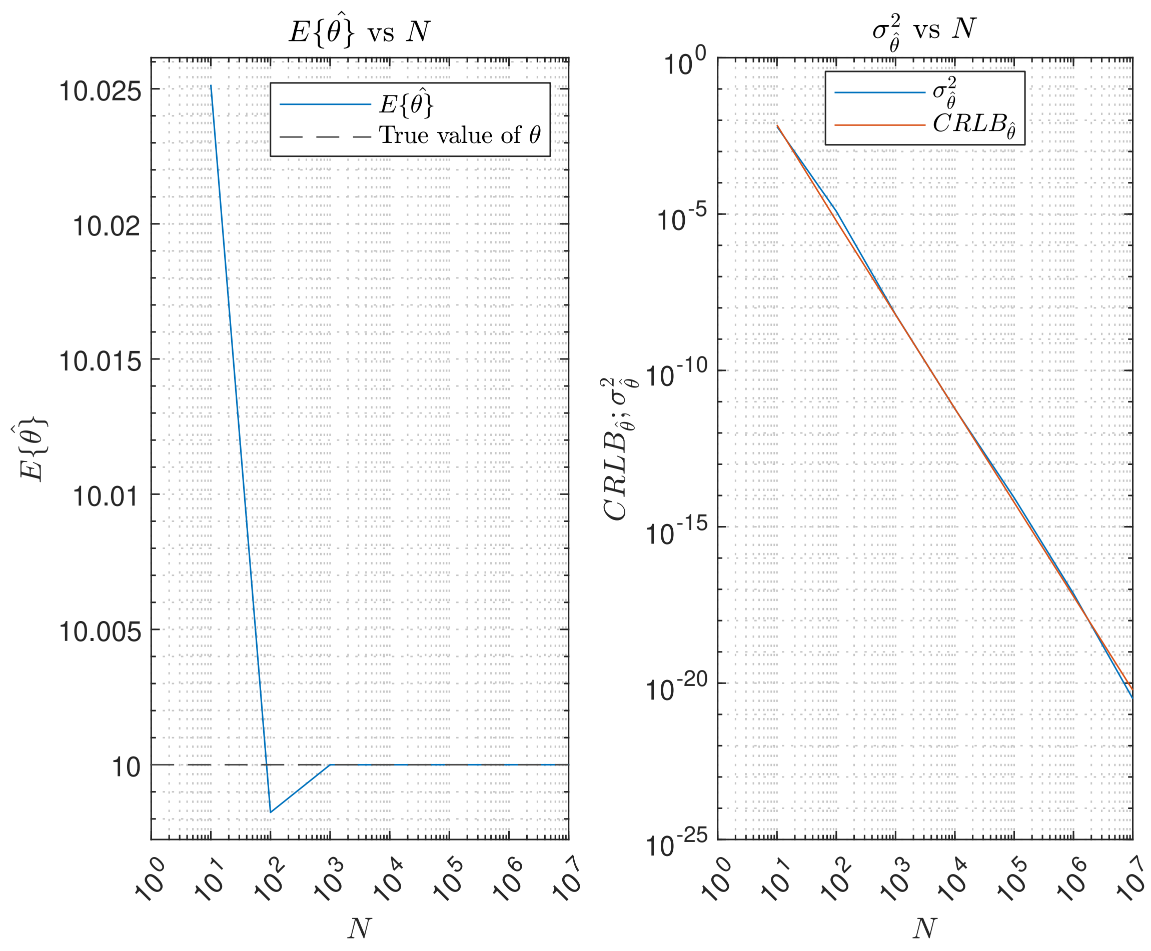

Figure 2: N effect