MATLAB Simulation Exercise

Statistical Signal Processing II

Wiener Filter

Group 13

Athmajan Vivekananthan

University of Oulu
Finland

January 25, 2024

1 Theoretical Background

Wiener filter is a linear optimum filters for complex valued discrete time series. The filter is specified with its impulse response. In the context of estimation theory, a Wiener filter is useful to minimize the mean squared error (MSE) between the estimated and the true signal in a wide-sense stationary (WSS) process in the face of noise [1].

The exercise task uses a third-order auto regressive (AR) process with input signal $d(n)$ with zero mean and variance $\frac{11}{18}$, symbolized by, $d(n) \sim N(0, \frac{11}{18})$. The corresponding channel given can be formulated using the difference equation given by equation 1. In the end of the task the whole exercise is repeated for a AR process of fourth order.

$$x(n) = \frac{1}{9}x(n-1) + \frac{1}{9}x(n-2) + \frac{4}{9}x(n-3) + d(n). \tag{1}$$

The diagram for the AR process is presented in figure 1.

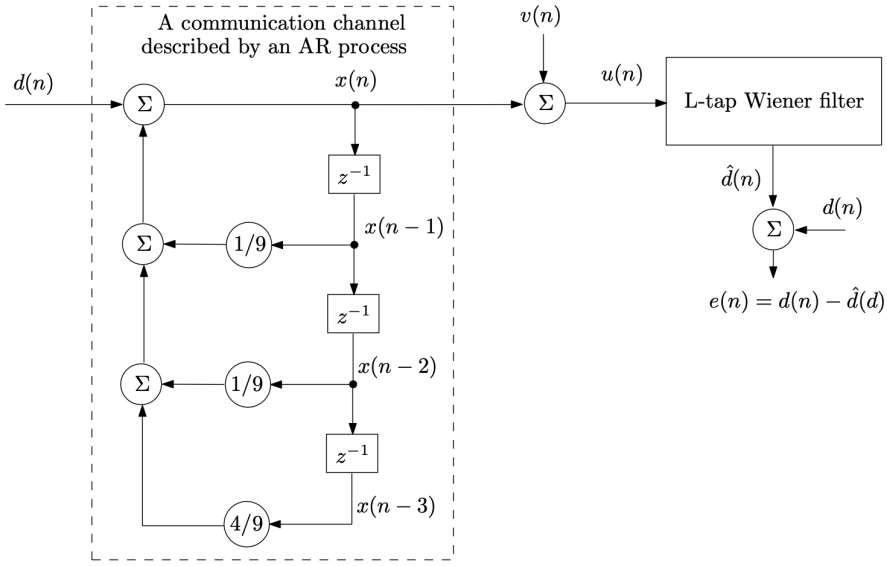

Figure 1: AR process diagram

The parameters of the random processes related to the task are summarized in table 1.

Table 1: Random Variables and Their Distributions

Random Variable	**Distribution**
$d(n)$	$N(0, \frac{11}{18})$
$v(n)$	$N(0, \sigma^2)$

1.1 Formalization of the communication channel

The input to the communication channel and the output from the channel is formulated as shown in the difference equation in 1. However this difference equation can be converted to Z-transform to make analysis easier.

The Z-transform is defined as:

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}$$

Applying the Z-transform to the difference equation, we get:

$$X(z) = \frac{1}{9} z^{-1} X(z) + \frac{1}{9} z^{-2} X(z) + \frac{4}{9} z^{-3} X(z) + D(z)$$

where $D(z)$ is the Z-transform of the input sequence $d(n)$.
Now, let's solve for $X(z)$:

$$X(z) - \frac{1}{9} z^{-1} X(z) - \frac{1}{9} z^{-2} X(z) - \frac{4}{9} z^{-3} X(z) = D(z)$$

Combine terms with $X(z)$ on one side:

$$X(z) \left(1 - \frac{1}{9} z^{-1} - \frac{1}{9} z^{-2} - \frac{4}{9} z^{-3}\right) = D(z)$$

Factor out $X(z)$:

$$X(z) \left(1 - \frac{z^{-1}}{9} - \frac{z^{-2}}{9} - \frac{4z^{-3}}{9}\right) = D(z)$$

Now, solve for $X(z)$:

$$X(z) = \frac{D(z)}{1 - \frac{z^{-1}}{9} - \frac{z^{-2}}{9} - \frac{4z^{-3}}{9}}$$

So, the Z-transform of the given difference equation is:

$$X(z) = \frac{D(z)}{1 - \frac{z^{-1}}{9} - \frac{z^{-2}}{9} - \frac{4z^{-3}}{9}} \tag{2}$$

1.2 Derivation of optimal Wiener filter

Since all intermediate signals present in the communication system is formalized now, the derivation of the optimum Wiener filter for the system given can be done.

The filter input is characterized by the following,

$$u(n) = v(n) + x(n) \tag{3}$$

The estimation error is characterized by the following

$$e(n) = d(n) - \hat{d}(n) \tag{4}$$

The criterion for optimality is the minimization of the cost function driven by the mean squared error values of the estimation error.

The impulse response of the Wiener Filter is characterized by the filter coefficients $w(0), w(1), w(2), \ldots$. Given that all input sequence and the filter impulse response are infinite duration the estimation of the signal can be derived as shown in equation 5. **From here onwards in the calculations, the complex conjugate is ignored since the problem in hand does not involve complex signals.**

$$\hat{d}(n) = \sum_{k=0}^{\infty} w_k \cdot u(n-k) \tag{5}$$

The cost function can be formulated as shown in equation 6.

$$J = \mathbb{E}\left[|e(n)^2|\right] \tag{6}$$

When the Wiener Filter is optimum the cost function in terms of the filter coefficients will have stationery points. This can be formulated by taking partial derivatives of the cost function in terms of all the Wiener Filter Coefficients and equating them to zero as shown in equation 7. Since the error performance nature is quadratic this local minimization using the zero gradient will assure the global optimality [2].

$$\nabla_k J = \frac{\partial J}{\partial w_k} = 0 \quad \text{Where } k = 0, 1, 2, 3, \ldots \tag{7}$$

The following shows the calculation for minimization.

$$\nabla_k J = \mathbb{E}\left[2\frac{\partial e(n)}{\partial w_k} e(n)\right]$$

$$\frac{\partial e(n)}{\partial w_k} = -u(n-k)$$

$$\nabla_k J = -2\mathbb{E}\left[u(n-k) \cdot e(n)\right]$$

At optimal Wiener Filter

$$\nabla_k J = 0$$

$$\mathbb{E}\left[u(n-k) \cdot e(n)\right] = 0$$

This implies that the estimation error samples are orthogonal to the input sample that enters in to the estimation at each epoch. Using this principal of orthogonality following derivations are done. **It is notable that all Wiener Filter coefficients from this point onwards are treated as optimal values even though the same notation has been continued for easiness of reporting**.

$$e(n) = d(n) - \sum_{i=0}^{\infty} w_i u(n-i)$$

$$\mathbb{E}\left[u(n-k) \cdot e(n)\right] = 0$$

$$\mathbb{E}\left[u(n-k) \cdot \left[d(n) - \sum_{i=0}^{\infty} w_i u(n-i)\right]\right] = 0$$

$$\mathbb{E}\left[u(n-k) \cdot d(n)\right] = \sum_{i=0}^{\infty} w_i \mathbb{E}\left[u(n-k) \cdot u(n-i)\right]$$

Here, LHS is the covariance between the signal and the desired values at $lag = -k$ and RHS is the auto correlation function of the signal at $lag = i-k$. LHR and RHS together when written in the matrix form is called the Wiener hopf equation which will be used to compute the tap coefficients in section 2.2.

$$p(-k) = \sum_{i} w_i r(i-k) \tag{8}$$

cross correlation vector of the input signal and the desired signal is p. And the cross correlation matrix between the inputs R gives the matrix formulation of the Wiener hopf equations as shown in equation 9.

$$\mathbf{Rw} = \mathbf{p} \tag{9}$$

$$\mathbf{R} = \mathbf{R} + \mathbf{R_2} \tag{10}$$

Where $\mathbf{R_2}$is the covariance matrix of the noise process $v(n)$

1.3 Derivation of minimum MSE

By expanding the definition of the MSE cost function given in 6, the relationship between the cost function, cross correlation vector of the input signal and the desired signal and the cross correlation matrix between the inputs can be derived as shown below.

$$\begin{aligned} J &= \mathbb{E}\left[e(n)^2\right] \\ &= \mathbb{E}\left[d(n)^2\right] - 2\sum_{k=0}^{M-1} w_k \mathbb{E}\left[u(n-k)\cdot d(n)\right] \\ &\quad + \sum_{k=0}^{M-1}\sum_{i=0}^{M-1} w_k w_i \mathbb{E}\left[u(n-k)\cdot u(n-i)\right] \\ &= \sigma_d^2 - 2\mathbf{w}^T\mathbf{p} + \mathbf{w}^T\mathbf{R}\mathbf{w} \end{aligned}$$

At optimum Wiener Filter based on equation 9, $\mathbf{w} = \mathbf{R}^{-1}\cdot\mathbf{p}$.

Substituting above result to the cost function gives the minimum cost as shown below.

$$\begin{aligned} \hat{J} &= \sigma_d^2 - 2\hat{\mathbf{w}}^T\mathbf{p} + \hat{\mathbf{w}}^T\mathbf{R}\hat{\mathbf{w}} \\ &= \sigma_d^2 - 2\left(\mathbf{R}^{-1}\cdot\mathbf{p}\right)^T\mathbf{p} + \left(\mathbf{R}^{-1}\cdot\mathbf{p}\right)^T\mathbf{R}\left(\mathbf{R}^{-1}\cdot\mathbf{p}\right) \\ &= \sigma_d^2 - \mathbf{p}^T\mathbf{R}^{-1}\mathbf{p} \end{aligned}$$

2 Results and Discussions

The initial task is to compute the intermediate signals of the system and plot them. Subsection 2.1 shows the results and discussions on that.

Listing 1:]Matlab snippet for generating x[n]

```
% input signal of the channel
d = randn(1, N) * sqrt(11/18);
% channel impulse response
channel_impulse = [1 -1/9 -1/9 -4/9];
% filtering operation to generate x[n] from d[n]
x = filter(1, channel_impulse, d);
```

2.1 Calculation of $x(n)$ and $u(n)$

Using the Z-transform equations that we obtained earlier 1.1, the equivalent rational transfer function for the MATLAB filter function can be derived as shown in equations 12 [3],

$$\text{Numerator coefficients} = 1 \tag{11}$$

$$\text{Denominator coefficients} = \left[1, -\frac{1}{9}, -\frac{1}{9}, -\frac{4}{9}\right] \tag{12}$$

Therefore the computation of $x(n)$ can be done using Matlab as shown in Listing 1.

2.2 Computation of optimal Wiener filter coefficients

Using the Wiener hopf equation derived in the theoretical section, the matrix for 3-tap Wiener Filter can be formulated as shown in equation 13.

$$\begin{bmatrix} r(0) & r(1) & r(2) \\ r(1) & r(0) & r(1) \\ r(2) & r(1) & r(0) \end{bmatrix} \begin{bmatrix} 1/9 \\ 1/9 \\ 4/9 \end{bmatrix} = \begin{bmatrix} r(1) \\ r(2) \\ r(3) \end{bmatrix} \tag{13}$$

By expanding the matrix multiplication and using the weighted summation of the auto correlation functions we have the following four simultaneous equations.

$$
\begin{aligned}
r(0) - 8r(1) + 4r(2) + 0r(3) &= 0 \\
r(0) + 5r(1) - 9r(2) + 0r(3) &= 0 \\
-4r(0) - r(1) - r(2) + 9r(3) &= 0 \\
18r(0) - 2r(1) - 2r(2) - 8r(3) &= 11
\end{aligned}
$$

Using Matlab to solve these four equations as shown in Listing 2, provides these coefficients for the optimum 3-tap Wiener Filter.

Listing 2: MATLAB Code for Solving a System of Equations

```
% Define the coefficients matrix A and the right-hand side ...
    vector B
A = [1, -8, 4, 0; 1, 5, -9, 0; -4, -1, -1, 9; 18, -2, -2, -8];
%B = [2, -1, 3, 1; 1, 3, 1, 2; 4, 2, -1, 3; 3, 4, 2, 1];
B = [0; 0; 0; 11];
%B = [8; 12; 3; 5];

% Solve the system of equations using linsolve
X = linsolve(A, B);

% Alternatively, you can use the backslash operator
% X = A \ B;
X = rats(X);
disp('Solution:');
disp(X);
```

$$
\begin{aligned}
r(0) &= \frac{11}{13} \\
r(1) &= \frac{11}{52} \\
r(2) &= \frac{11}{52} \\
r(3) &= \frac{11}{26}
\end{aligned}
$$

2.3 Computation of corresponding MSE

$$
\mathbf{p} = \begin{bmatrix} p(0) & p(-1) & p(-2) & \end{bmatrix}^T
$$

Using equation 8 all elements of the cross correlation vector can be calculated as shown below:

$$p(0) = r(0) - \frac{1}{9}r(1) - \frac{1}{9}r(2) - \frac{4}{9}r(3)$$

$$p(-1) = r(1) - \frac{1}{9}r(0) - \frac{1}{9}r(-1) - \frac{4}{9}r(-2)$$

$$p(-2) = r(2) - \frac{1}{9}r(-1) - \frac{1}{9}r(0) - \frac{4}{9}r(-1)$$

$$p(-3) = r(3) - \frac{1}{9}r(2) - \frac{1}{9}r(1) - \frac{4}{9}r(0)$$

$$\mathbf{p} = \begin{bmatrix} \frac{11}{18} & 0 & 0 \end{bmatrix}^T$$

Since all covariance have been found the optimum filter coefficients and the minimum MSE can be found. The Matlab code for the calculation of those are shown in Listing 3. The filter coefficients computed are shown below.

$$\mathbf{w} = [0.7034 \quad -0.1285 \quad -0.1285 \quad]^T$$

Listing 3: MATLAB Code for Covariance Calculation

```
r0 = 11/13;
r1 = 11/52;
r2 = 11/52;
r3 = 11/26;

r = [r0; r1; r2; r3];
% Covariance of the input x(n)
Rx = toeplitz(r(1:3));

% Covariance of the noise process v(n)
R2 = sigma_sqr*eye(3);

% Covariance of input signal u(n)
R = Rx + R2;
% Covariance vector
p = [11/18; 0; 0];

% Optimum filter coefficients
```

```
w = inv(R)*p;
% Minimum MSE
MSE = 11/18 - p'*inv(R)*p;
```

2.4 Estimating the desired response using Wiener Filter

Estimation of the desired signal can be now done using the calculated Optimum Wiener Filter. The Matlab snippet shown in Listing 4 provides the estimated desired signal.

Listing 4: MATLAB Snippet

```
r0 = 11/13;
r1 = 11/52;
r2 = 11/52;
r3 = 11/26;

r = [r0; r1; r2; r3];
Rx = toeplitz(r(1:3));
%Rx = [r0 r1 r2; r1 r0 r1; r2 r1 r0];
R2 = sigma_sqr*eye(3);
R = Rx + R2;
p = [11/18; 0; 0];

w0 = inv(R)*p; % Wiener filter coefficients obtained by ...
    calculation
dn_hat = filter(w0, 1, u); % filtering operation
```

All the figures of the results section has been shown for a length of 10 vectors in Figure 2.

2.5 Comparison of d(n) and $\hat{\mathbf{d}}$(n)

For the realization with $N = 10$, the element-wise comparison can be found in table 2.5 where the error for each discrete time and the total mean squared error has also been numerically calculated for the example of seed $rng(42)$.

n	1	2	2	4	5	6	7	8	9	10
$d(n)$	-0.42	0.68	0.76	0.26	-0.78	-0.41	-1.01	0.72	0.14	0.59
$\hat{d}(n)$	-0.43	0.93	0.88	-0.20	-0.72	-0.14	-0.54	0.45	-0.05	0.19
$e(n)$	0.01	-0.25	-0.12	0.46	-0.06	-0.27	-0.47	0.27	0.18	0.40

MSE of data presented in table 2.5 is calculated as below where it can be seen the similarity between $d(n)$ and $\hat{d}(n)$ is very high by observing the MSE.

$$
\begin{aligned}
MSE &= (0.01)^2 + (-0.25)^2 + (-0.12)^2 + (0.46)^2 + (-0.06)^2 \\
&\quad + (-0.27)^2 + (-0.47)^2 + (0.27)^2 + (0.18)^2 + (0.4)^2 \\
&= 0.0857
\end{aligned}
$$

The comparison with the theoretical MSE and the simulated MSE for the case of $N = 10$ and $N = 10^8$ can be found in table2. It can be seen for infinitely long discrete time sequences the **simulation results tally with theoretical calculations**.

Table 2: Comparison of Theoretical and Simulated MSE

Method	MSE
Theoretical	0.1813
Simulation ($N = 10$)	0.0857
Simulation ($N = 10^8$)	0.1812

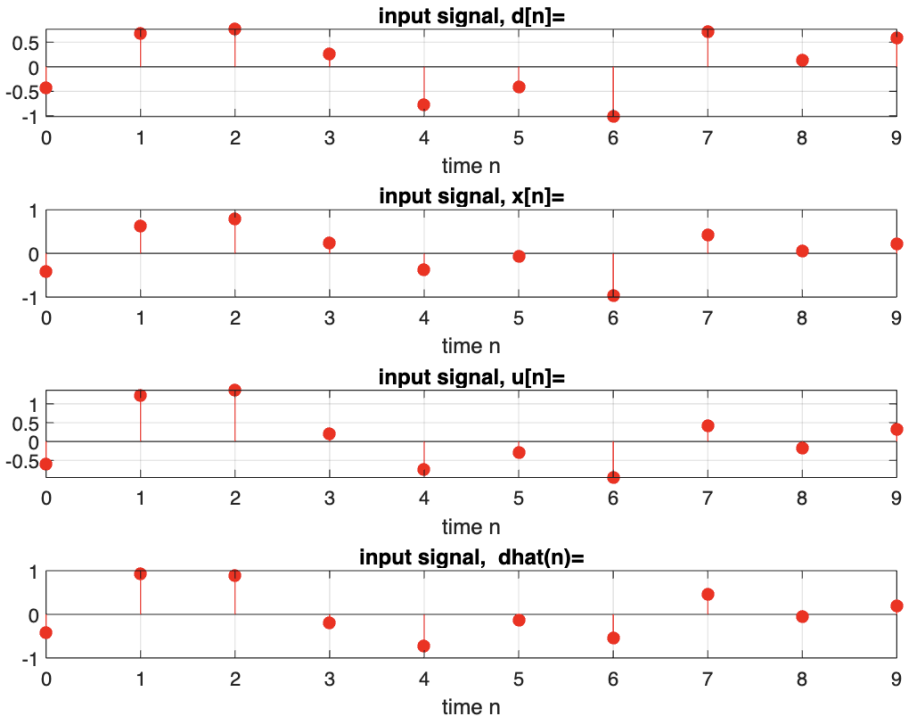

Figure 2: Realizations of d(n), x(n), u(n) and $\hat{d}(n)$ for N =10

2.6 Calculation of MSE between d(n) and $\hat{\mathbf{d}}$(n)

The Matlab code snippet to calculate the MSE for this section has been given in Listing 5.

Listing 5: MATLAB code for calculating Mean Square Error (MSE)

```
calculateMSE(dn_hat, d)

function mse = calculateMSE(dhat, d)
    % Calculate Mean Square Error (MSE)
    mse = mean((dhat - d).^2);
end
```

2.7 Simulation for different noise variances

Simulation results obtained by changing σ^2 values are presented in table 3.

2.8 Simulation for $L = 4$

When the number of tap for the Wiener fiter was increased to 4, the cross correlation vector and matrix can be written based on the previous calculations done for $L = 3$.

$$\mathbf{p} = \begin{bmatrix} \frac{11}{18} & 0 & 0 & 0 \end{bmatrix}^T$$

The cross correlation matrix $\mathbf{R}$ can be derived as

$$R = \begin{bmatrix} r(0) & r(1) & r(2) & r(3) \\ r(1) & r(0) & r(1) & r(2) \\ r(2) & r(1) & r(0) & r(1) \\ r(3) & r(2) & r(1) & r(0) \end{bmatrix} \tag{14}$$

Using the above two newly defined matrix and vector the new filter coefficients can be computed. Using these new coefficients the MSE for various noise variance values were simulated and the results are presented together for both $L = 3$ and $L = 4$ in table 3.

$$\mathbf{w} = [\quad 0.8357 \quad -0.0919 \quad -0.0919 \quad -0.3326 \quad]^T$$

Table 3: Theoretical Minimum MSE and Simulated MSE results for $N = 10^8$

MSE	L=3			L=4		
	$\sigma^2 = 0.1$	$\sigma^2 = 0.01$	$\sigma^2 = 0.5$	$\sigma^2 = 0.1$	$\sigma^2 = 0.01$	$\sigma^2 = 0.5$
Theoretical	0.1813	0.1276	0.3213	0.1004	0.0120	0.2959
Simulated	0.1812	0.1275	0.3212	0.1004	0.0120	0.2958

Comparing the results presented in Table 3 shows that when the number of taps are increased the theoretical Minimum MSE and the simulated test MSEs reduce drastically. However the MSE largely depends on the noise variance. It can be observed that for larger noise variances the MSE is higher for both cases of $L = 3$ and $L = 4$.

3 Conclusion

After completing the exercise of designing an optimum Wiener Filter for a third order AR communication system a clear understanding on the theoretical calculations and their applications was obtained.

Based on the simulation observations several conclusions can be made.

1. **Importance of Infinite Sequence Length:**
   Theoretical concepts emphasize the consideration of infinitely long sequences of discrete-time data. This significance becomes apparent through a comparison of simulation cases with different sequence lengths, such as $N = 10$ and $N = 10^8$. In the case of $N = 10$, the simulated Mean Squared Error (MSE) was found to be smaller than the theoretical minimum. However, this deviation was rectified by significantly increasing the length of the input sequences.

2. **Reducing the Effect of Noise:**

   The simulation was conducted for various noise variance scenarios, revealing the substantial impact of noise on signal estimation. The optimal Wiener filter demonstrated its effectiveness in diminishing the influence of noise, showcasing its robustness across different noise levels.

3. **Effect of the Number of Taps:**

   The simulation was executed for two different tap counts, providing insights into the influence of the filter's order. It was observed that the number of taps has a profound effect on Wiener filter performance. Specifically, increasing the tap count from 3 to 4 resulted in a significant reduction in MSE for each test case, highlighting the importance of choosing an appropriate filter length.

4 Appendix - Matlab Code

```
%% SSP II - Matlab exercise 1 - Wiener Filter

clear;clc;
N = 10^8; % no of sample - Change this
L = 4; % no of Wiener filter taps - Change this
sigma_sqr = 0.5; %- Change this
%% generate input signal

% Set the seed for reproducibility
rng(42);

d = randn(1, N) * sqrt(11/18);
%% output signal of the channel
channel_impulse = [1 -1/9 -1/9 -4/9]; % channel impulse response
x = filter(1, channel_impulse, d);  % filtering operation to generate x[n] from d[n]
%%
v = randn(1, N) * sqrt(sigma_sqr);
u = x +v;
%% Applying Wiener filter

r0 = 11/13;
r1 = 11/52;
r2 = 11/52;
r3 = 11/26;
% Defining the r vector to create cross correlation matrix
r = [r0; r1; r2; r3];

% Define cross correlation matrix
Rx = toeplitz(r(1:L));
% Define correlation matrix of noise process v(n)
R2 = sigma_sqr*eye(L);
% Correlation matrix of the input signal u(n)
R = Rx + R2;

%p = [11/18;0;0]; % L=3
%p = [11/18;0;0;0]; % L=4
% Defining p for the general case for both L=3 and L=4
p = zeros([L,1]);
```

```
p(1) = 11/18;
w0 = inv(R)*p; % Wiener filter coefficients obtained by calculation
dn_hat = filter(w0, 1, u); % filtering operation
%%
% Plot figures only for the case of N=10
if N==10

    subplot(4,1,1)
    stem([1:N]-1,d,'fill','r')
    grid on;
    title('input signal, d[n]=')
    xlabel('time n')

    subplot(4,1,2)
    stem([1:N]-1,x,'fill','r')
    grid on;
    title('input signal, x[n]=')
    xlabel('time n')

    subplot(4,1,3)
    stem([1:N]-1,u,'fill','r')
    grid on;
    title('input signal, u[n]=')
    xlabel('time n')

    subplot(4,1,4)
    stem([1:N]-1,dn_hat,'fill','r')
    grid on;
    title('input signal,  dhat(n)=');
    xlabel('time n')
end

%% calculate MSE
mse = mean((dn_hat - d).^2);
disp(mse)
```

References

[1] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall International, 1993.

[2] S. Haykin, *Adaptive Filter Theory*. Pearson, 2014. [Online]. Available: https://books.google.fi/books?id=J4GRKQEACAAJ

[3] T. M. Inc., "Matlab version: 9.13.0 (r2022b)," Natick, Massachusetts, United States, 2022. [Online]. Available: https://www.mathworks.com/help/matlab/ref/filter.html