Advanced Digital Signal Processing (ADSP) Lab - Python Lab Manual

Course Code: EEE G613 **Instructer in Charge:** Dr. Rajesh Kumar Tripathy **Teaching Assistant:** Shaswati Dash **Lab Technician:** Ramesh Pokanati

Experiment No. - 8

Weiner Filter for Filtering Application

Let d(n) be an AR(1) process with an autocorrelation sequence

$$r_d(k)=\alpha^{|k|}$$

With $0<\alpha<1$, and suppose that d(n) is observed in the presence of uncorrelated white noise, *v(n),* that has a variance of $\sigma_v^2 = 1$

$$x(n)=d(n)+v(n)$$

Note: Wherever required perform mathematical derivations in your observation book using those expressions write Matlab code.

a) Consider for $\alpha=0.8$, 1st order, 2nd order and 5th order filter. Write a Matlab Program to find w(z) (using which $w(e^{j\omega})$). Plot the magnitude vs frequency response for these filter orders. Calculate the SNR for these filter orders. Write down your observations

b) For a 3rd order filter consider $\alpha = 0.1, 0.2, 0.3 \ldots.0.9$. Calculate mean-square error for each α value. Plot mean square error vs α. Write down your observations.

Hint: Check example 7.2.1 from Monsoon H. Hayes book.

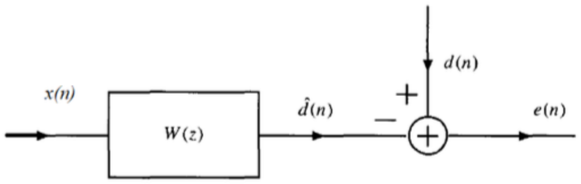
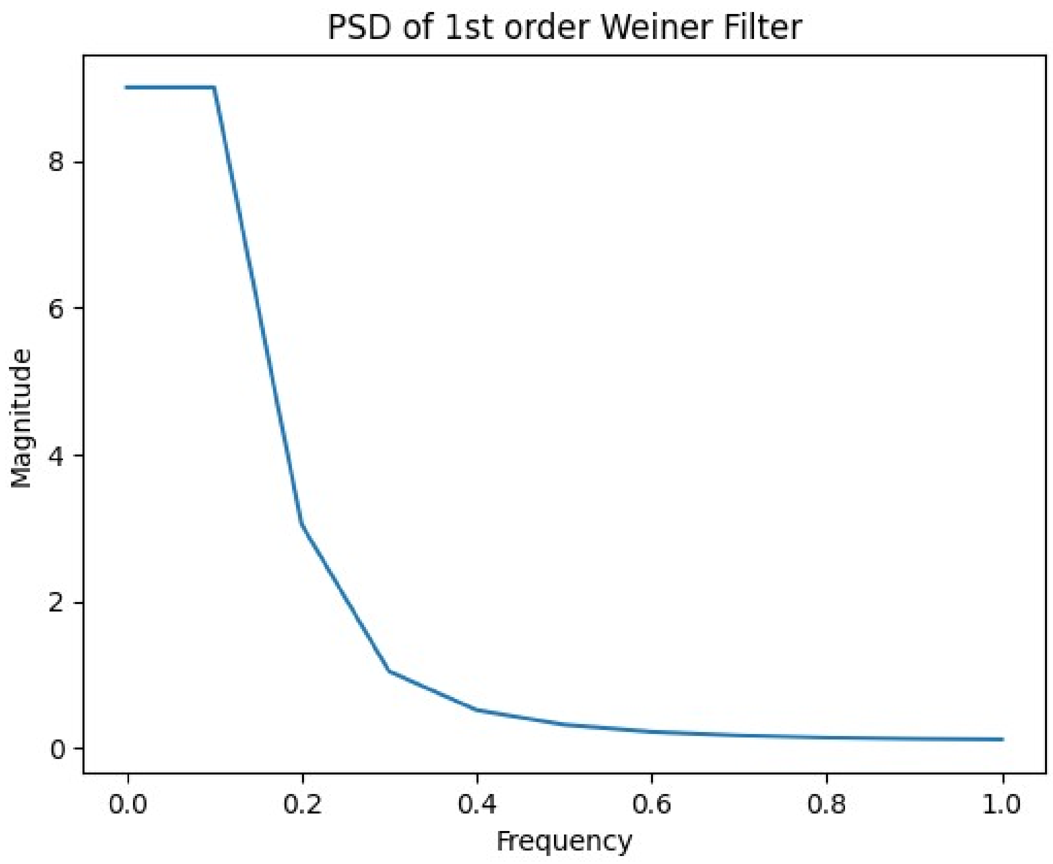


Fig.1 Illustration of the general wiener filtering problem.

Python Code-

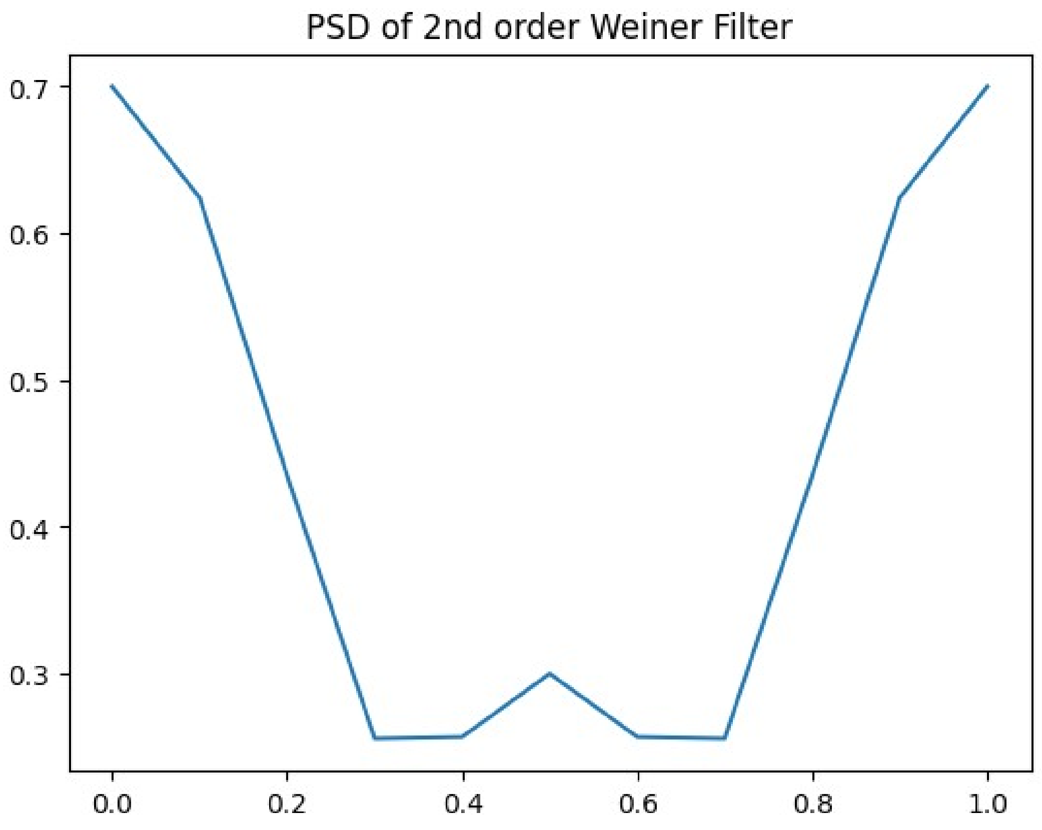
```python
#import libraries
import numpy as np
import matplotlib.pyplot as plt

# For 1st order
a = 0.8
s = 1
f = np.arange(0, 1.1, 0.1)
w = 2 * np.pi * f
Rx = np.array([[1 + s ** 2, a], [a, 1 + s ** 2]])
Rdx = np.array([1, a])
W1 = np.linalg.inv(Rx).dot(Rdx)
signalp = np.dot(W1.T, np.dot(np.array([[1, a], [a, 1]]), W1))
noisep = np.dot(W1.T, W1)
SNR1 = 10 * np.log10(signalp / noisep)
psd = [(1 - a ** 2) / ((1 + a ** 2) - 2 * a * np.cos(0))]
for i in range(1, 11):
    psd.append((1 - a ** 2) / ((1 + a ** 2) - 2 * a * np.cos((i - 1) /
10 * np.pi)))
print('Weiner Coefficients for 1st order filter = ',W1)
print('SNR for 1st order filter =',SNR1)
plt.figure()
plt.plot(f, psd)
plt.title('PSD of 1st order Weiner Filter')
plt.xlabel('Frequency')
plt.ylabel('Magnitude')
```

```
Weiner Coefficients for 1st order filter =  [0.4047619  0.23809524]
SNR for 1st order filter = 2.3025185815993248

Text(0, 0.5, 'Magnitude')
```



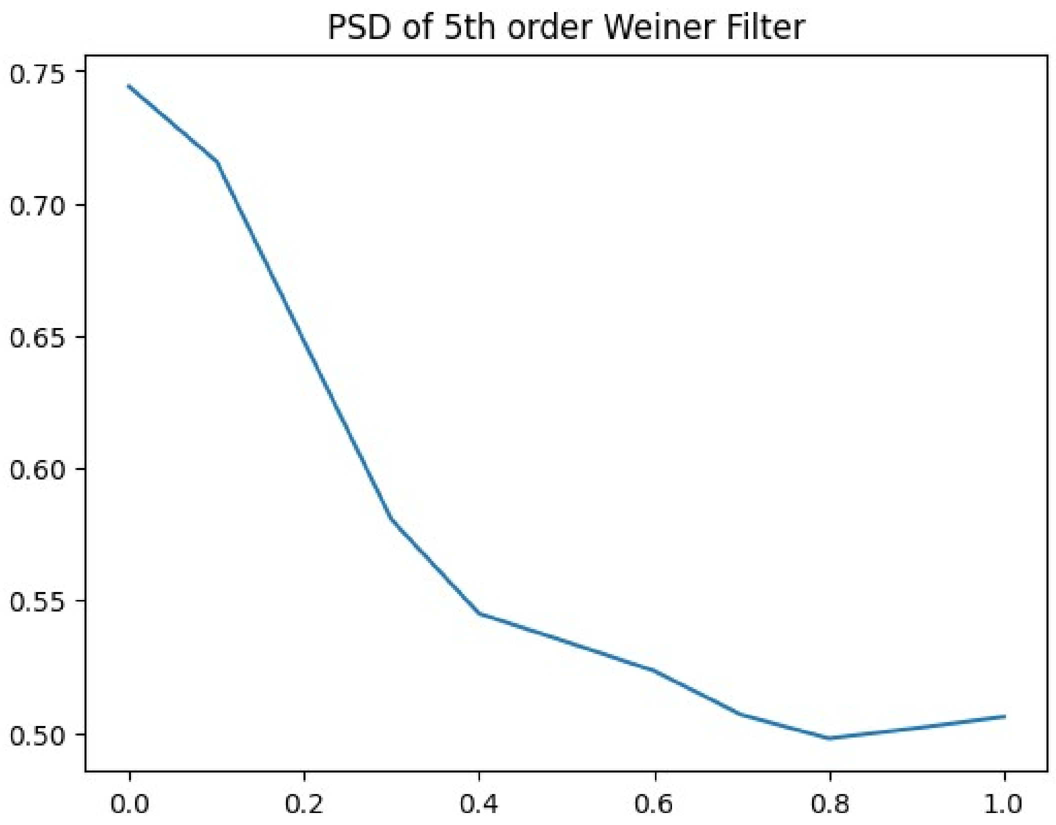
```
# For 2nd order
Rx = np.array([[1 + s ** 2, a, a ** 2], [a, 1 + s ** 2, a], [a ** 2,
a, 1 + s ** 2]])
Rdx = np.array([1, a, a ** 2])
W2 = np.linalg.inv(Rx).dot(Rdx)
signalp = np.dot(W2.T, np.dot(np.array([[1, a, a ** 2], [a, 1, a], [a
** 2, a, 1]]), W2))
noisep = np.dot(W2.T, W2)
SNR2 = 10 * np.log10(signalp / noisep)
Wequ2 = 0.3824 + (0.2 * np.exp(-1j * w)) + (0.1176 * (np.exp(-1j * w)
** 2))
print('Weiner Coefficients for 2nd order filter = ',W2)
print('SNR for 2nd order filter =', SNR2)
plt.figure()
plt.plot(f, np.abs(Wequ2))
plt.title('PSD of 2nd order Weiner Filter')
```

```
Weiner Coefficients for 2nd order filter =  [0.38235294 0.2
0.11764706]
SNR for 2nd order filter = 3.196683179158083

Text(0.5, 1.0, 'PSD of 2nd order Weiner Filter')
```



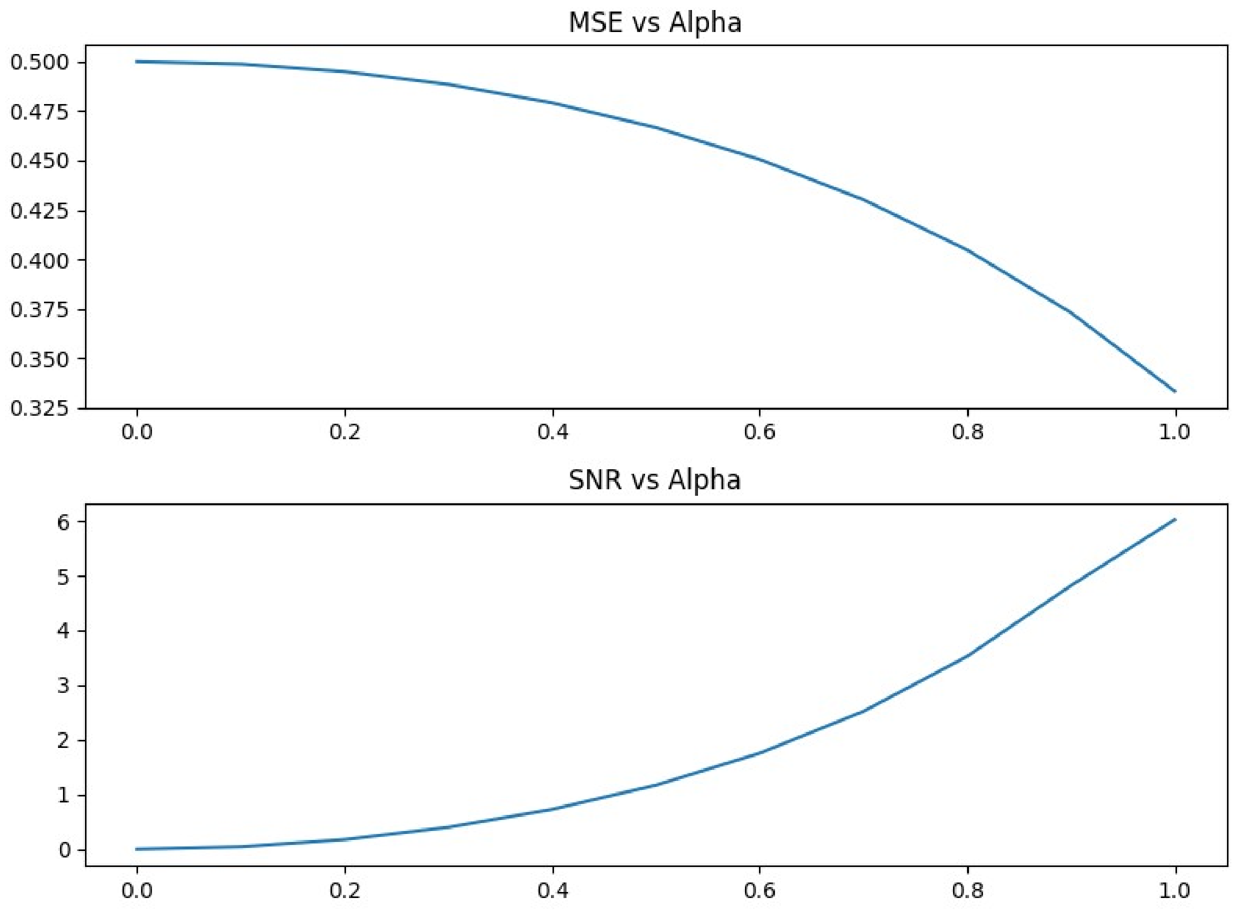
```
# For 5th order
n = 5
k = np.arange(1, n + 2)
a = 0.8
sigma = 1
Rx = np.array([[1 + s ** 2, a, a ** 2, a ** 3, a ** 4, a ** 5],
               [a, 1 + s ** 2, a, a ** 2, a ** 3, a ** 4],
               [a ** 2, a, 1 + s ** 2, a, a ** 2, a ** 3],
               [a ** 3, a ** 2, a, 1 + s ** 2, a, a ** 2],
               [a ** 4, a ** 3, a ** 2, a, 1 + s ** 2, a],
               [a ** 5, a ** 4, a ** 3, a ** 2, a, 1 + s ** 2]])
Rdx = np.array([1, a, a ** 2, a ** 3, a ** 4, a ** 5])
W5 = np.linalg.inv(Rx).dot(Rdx)
print('Weiner Coefficients for 5th order filter=',W5)
signalp = np.dot(W5.T, np.dot(np.array([[1, a, a ** 2, a ** 3, a ** 4,
a ** 5],
                                       [a, 1, a, a ** 2, a ** 3, a **
4],
                                       [a ** 2, a, 1, a, a ** 2, a **
3],
                                       [a ** 3, a ** 2, a, 1, a, a **
2],
                                       [a ** 4, a ** 3, a ** 2, a, 1,
a],
                                                       [a ** 5, a ** 4, a ** 3, a **
2, a, 1]]), W5))
noisep = np.dot(W5.T, W5)
SNR5 = 10 * np.log10(signalp / noisep)
print('SNR for 5th order filter =',SNR5)
f = np.arange(0, 1.1, 0.1)
rd = a ** (k - 1)
rv = np.concatenate(([sigma], np.zeros(n)))
rx = rd + rv
R = np.zeros((n + 1, n + 1))
for i in range(n + 1):
    for j in range(n + 1):
        R[i, j] = rx[np.abs(i - j)]
w = np.linalg.inv(R).dot(rd)
winereqn = w[0]
for b in range(n):
    winereqn += w[b + 1] * np.exp((-b * 1j) * f * np.pi)
plt.figure()
plt.plot(f, np.abs(winereqn))
plt.title('PSD of 5th order Weiner Filter')
plt.show()
```

```
Weiner Coefficients for 5th order filter [0.37511445 0.18769456
0.09412196 0.04761033 0.02490386 0.01464933]
SNR for 5th order filter = 3.665383527068688
```



```
# For a 3rd order filter consider α = 0.1, 0.2, 0.3 ….0.9. Calculate
mean-square error for each α value

#import libraries
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import toeplitz

n = 3
a = np.arange(0, 1.1, 0.1)
mse = []
SNR = []

for alpha in a:
    k = np.arange(1, n + 2)
    sigma = 1
    f = np.arange(0, 1.1, 0.1)
    rd = alpha ** (k - 1)
    rv = np.concatenate(([sigma], np.zeros(n)))
    rx = rd + rv
    R = np.zeros((n + 1, n + 1))
    for i in range(n + 1):
        for j in range(n + 1):
            R[i, j] = rx[np.abs(i - j)]
    w = np.linalg.inv(R).dot(rd)
    winereqn = w[0]
    for b in range(n):
        winereqn += w[b + 1] * np.exp((-b * 1j) * f * np.pi)

    mse_i = sigma**2 * ((1 + sigma) - alpha**2) / ((1 + sigma**2)**2 -
alpha**2)
    Sp = np.dot(np.dot(w.T, toeplitz(rd)), w)
    Np = np.dot(w.T, w)
    SNR_i = 10 * np.log10(Sp / Np)

    mse.append(mse_i)
    SNR.append(SNR_i)

print("rd =", rd)
print("Weiner Coefficients for 3rd order filter =", w)
print("Weiner matrix:\n", winereqn)

plt.figure(figsize=(8, 6))
plt.subplot(2, 1, 1)
plt.plot(a, mse)
plt.title("MSE vs Alpha")
plt.subplot(2, 1, 2)
plt.plot(a, SNR)
plt.title("SNR vs Alpha")
plt.tight_layout()
plt.show()
```

```
rd:
 [1. 1. 1. 1.]
Weiner Coefficients for 3rd order filter
 [0.2 0.2 0.2 0.2]
Weiner matrix:
 [0.8       +0.00000000e+00j 0.7520147 -1.79360449e-01j
 0.6236068 -3.07768354e-01j 0.45575365-3.52014702e-01j
 0.3       -3.07768354e-01j 0.2       -2.00000000e-01j
 0.1763932 -7.26542528e-02j 0.22063955+2.84079044e-02j
 0.3       +7.26542528e-02j 0.3715921 +5.57536516e-02j
 0.4       +2.44929360e-17j]
```


MSE vs Alpha
0.500
0.475
0.450
0.425
0.400
0.375
0.350
0.325
0.0
0.2
0.4
0.6
0.8
1.0
SNR vs Alpha
6
5
4
3
2
1
0
0.0
0.2
0.4
0.6
0.8
1.0