ECE63700-Lab2

Sriranga R

January 26, 2023

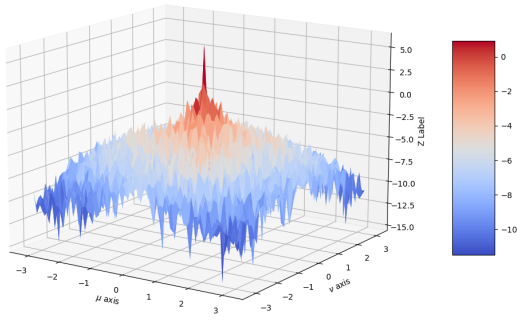
1 Power Spectral Density of an Image

1.1 The gray scale image img04.tif

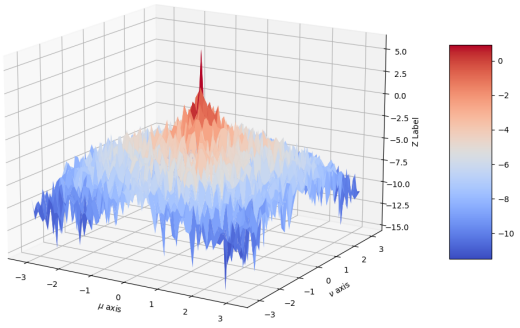
Figure 1: img04.tif

1.2 The power spectral density plots for block sizes

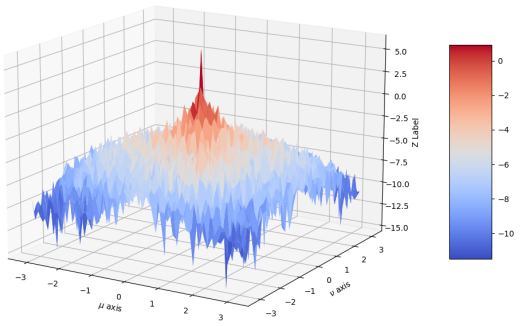
1.2.1 mesh size : 64 * 64



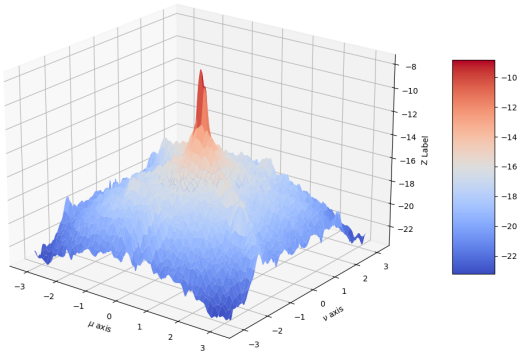
1.2.2 mesh size : 128 * 128



1.2.3 mesh size : 256 * 256



1.3 The improved power spectral density estimate



The above figure in section 1.3 looks to be more uniform and less noisy compared to the other filters of size 64, 128 and 256 mentioned in section 1.2.

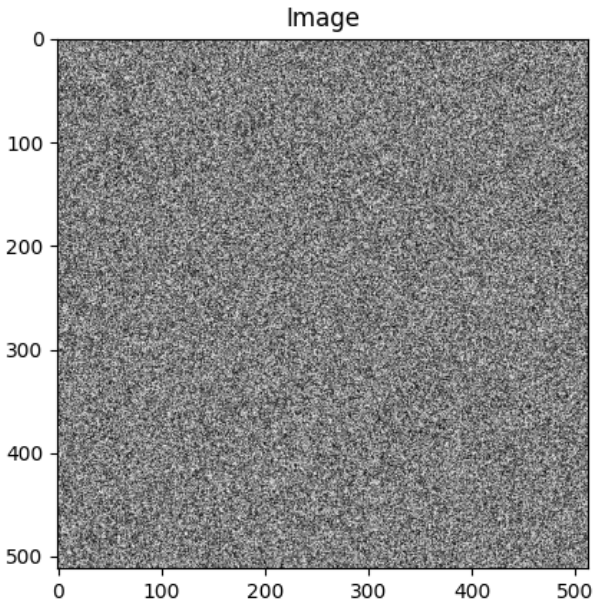
1.4 Code for BetterSpecAnal(x) function.

```
import numpy as np                  # Numpy is a library support computation of large, multi-dimensional arrays and matrices.
from PIL import Image               # Python Imaging Library (abbreviated as PIL) is a free and open-source additional library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.
import matplotlib.pyplot as plt     # Matplotlib is a plotting library for the Python programming language.



def BetterSpecAnalyzer(image):
    [h, w] = image.shape
    print("h = " + str(h) + ", w = " + str(w))
    midh = int(h / 2)
    midw = int(w / 2)
    #25 filters to be used. 5 * 5
    N = 64 #size of each filter
    print(midh, midw)
    H = np.outer(np.hamming(64), np.hamming(64))
    print(H.shape)
    X = np.double(image)/255

    #overlapping
    start_h = int(midh - ((N*5)/2))
    start_w = int(midw - ((N*5)/2))

    Z = np.zeros((512, 512))

    for i in range(0,5):
        for j in range(0,5):
            h_start = start_h + i*N
            h_end = start_h + (i+1)*N
            w_start = start_w + j*N
            w_end = start_w + (j+1)*N
            print('h_start = ' , h_start , ' , h_end = ', h_end, ' , w_start = ', w_start, ' , w_end = ', w_end)
            z = X[h_start : h_end, w_start : w_end]
            z_hamm = z*H
            Z = Z + (1/N**2) * abs(np.fft.fftshift(np.fft.fft2(z_hamm, s=[512, 512]))**2)

    Z = Z / 25
    # Compute the logarithm of the Power Spectrum.
    Zabs = np.log(Z)
    # Plot the result using a 3-D mesh plot and label the x and y axises properly.
    fig = plt.figure()
    ax = plt.axes(projection='3d')
    a = b = np.linspace(-np.pi, np.pi, num = 512)
    X, Y = np.meshgrid(a, b)

    surf = ax.plot_surface(X, Y, Zabs, cmap=plt.cm.coolwarm)
    #ax = plt.axes(projection='3d')
    #ax.contour3D(X, Y, Zabs, 50, cmap='viridis')

```

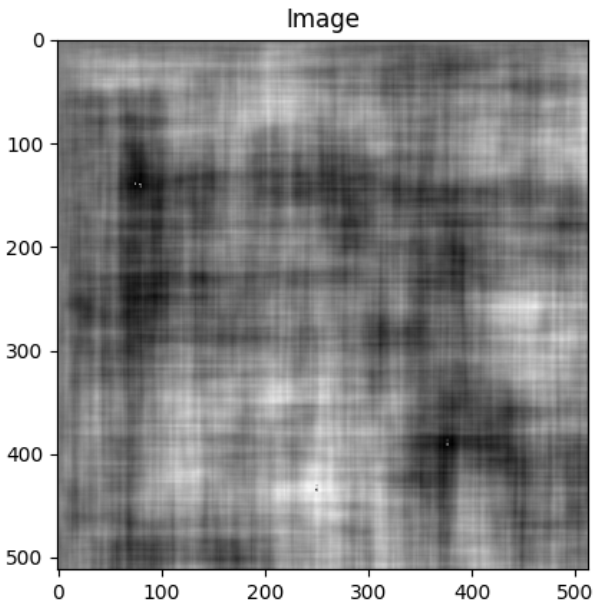
```
    ax.set_xlabel('$\mu$ axis')
    ax.set_ylabel('$\\nu$ axis')
    ax.set_zlabel('Z Label')

    fig.colorbar(surf, shrink=0.5, aspect=5)

    plt.show()

# Read in a gray scale TIFF image.
im = Image.open('img04g.tif')
# Import Image Data into Numpy array.
# The matrix x contains a 2-D array of 8-bit gray scale values.
x = np.array(im)
print('Data type: ', x.dtype)
x = np.double(x)/255.0
BetterSpecAnalyzer(x)
```

2 Power Spectral Density of a 2-D AR Process

2.1 The image 255 * (x + 0:5)

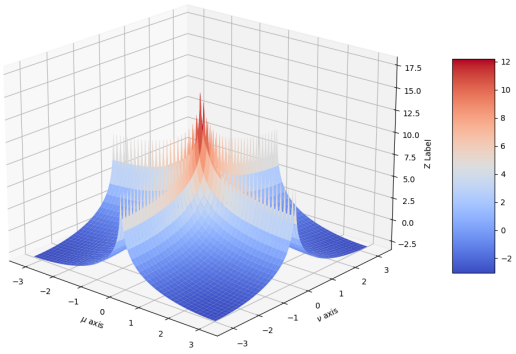


2.2 The image y + 127.

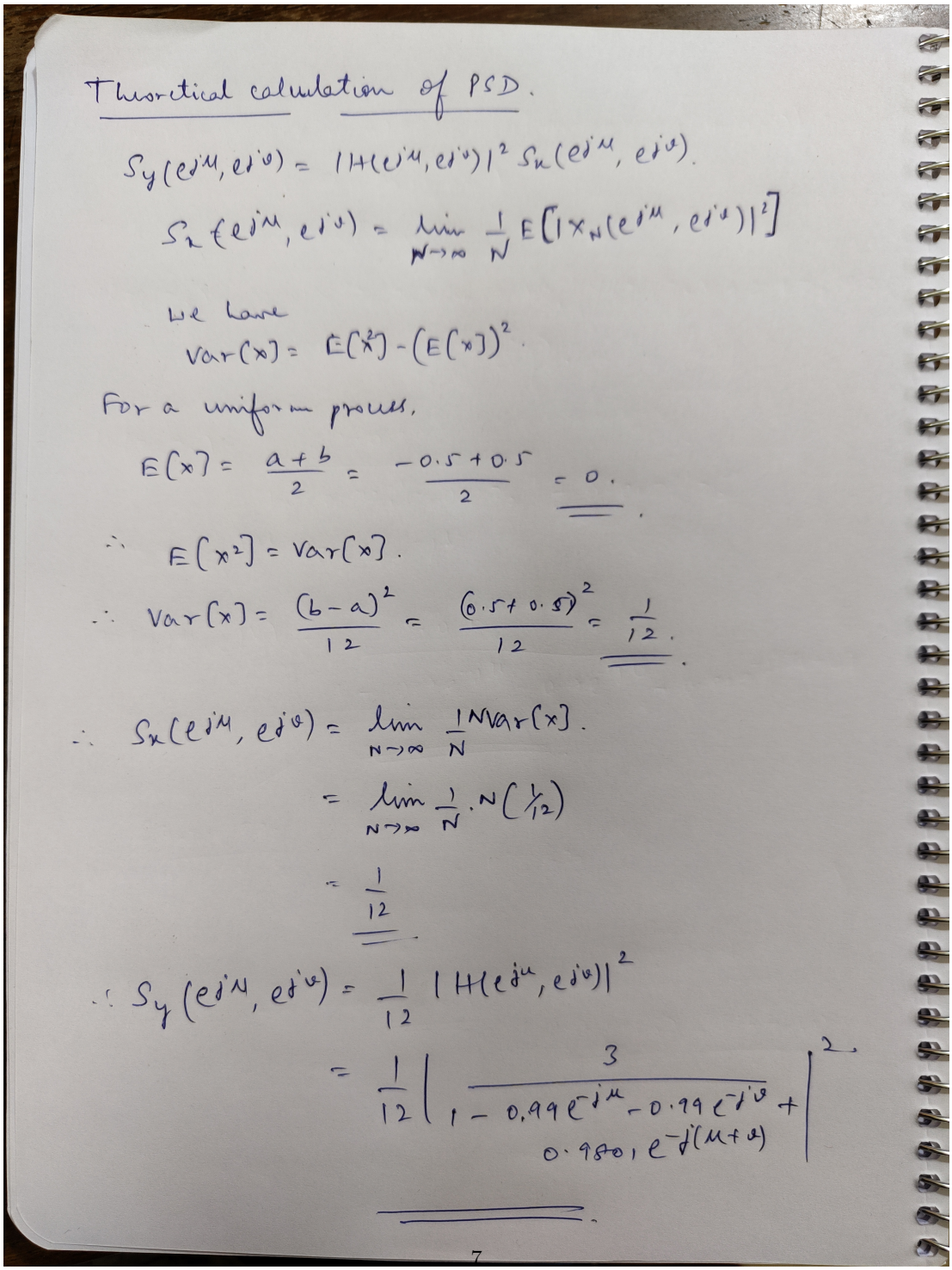


2.3 A mesh plot of the function $log(S_y(e^{j\mu}, e^{j\nu})$.

2.3.1 The Mesh plot of the PSD



2.3.2 Derivation of the theoretical PSD



Theoretical calculation of PSD.

$$S_y(e^{j\mu}, e^{j\upsilon}) = |H(e^{j\mu}, e^{j\upsilon})|^2 \, S_x(e^{j\mu}, e^{j\upsilon})$$

$$S_x(e^{j\mu}, e^{j\upsilon}) = \lim_{N\to\infty} \frac{1}{N} E\left[|X_N(e^{j\mu}, e^{j\upsilon})|^2\right]$$

We have

$$\text{Var}(x) = E(x^2) - (E(x))^2.$$

For a uniform process,

$$E(x) = \frac{a+b}{2} = \frac{-0.5+0.5}{2} = 0.$$

$$\therefore E(x^2) = \text{Var}(x).$$

$$\therefore \text{Var}(x) = \frac{(b-a)^2}{12} = \frac{(0.5+0.5)^2}{12} = \frac{1}{12}.$$

$$\therefore S_x(e^{j\mu}, e^{j\upsilon}) = \lim_{N\to\infty} \frac{1}{N} N \text{Var}(x).$$

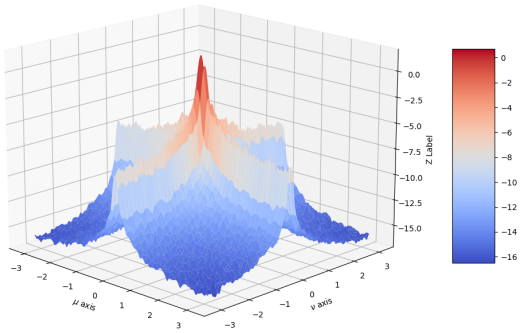
$$= \lim_{N\to\infty} \frac{1}{N} \cdot N\left(\frac{1}{12}\right)$$

$$= \frac{1}{12}$$

$$\therefore S_y(e^{j\mu}, e^{j\upsilon}) = \frac{1}{12} |H(e^{j\mu}, e^{j\upsilon})|^2$$

$$= \frac{1}{12} \left| \frac{3}{1 - 0.99e^{-j\mu} - 0.99e^{-j\upsilon} + 0.980\, e^{-j(\mu+\upsilon)}} \right|^2.$$

2.4 A mesh plot of the log of the estimated power spectral density of y using Better- SpecAnal(y).



2.5 Additional Code listings

2.5.1 Problem2.py

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import cmath
import BetterSpectrumAnalyzer

image = np.random.uniform(low=-0.5, high=0.5, size=(512,512))
image_scaled = (image + 0.5) * 255
plt.imshow(np.uint8(image_scaled), cmap=plt.cm.gray)
plt.title('Image')
plt.show()
# Import Image Data into Numpy array.
# The matrix x contains a 2-D array of 8-bit gray scale values.

x_scaled = np.array(image_scaled)
img_out = Image.fromarray(x_scaled.astype(np.uint8))

img_out.save('random.tif')
x = np.array(image)
print('Data type: ', x.dtype)

#The filter difference equation
#y(m,n) = 3x(m,n) + 0.99y(m-1,n) + 0.99y(m,n-1) - 0.9801y(m-1,n-1)

m,n = x.shape
print(x.shape)
```

```
y = np.zeros((m,n))
for i in range(0,m):
    for j in range(0,n):
        y[i][j] = 3 * x[i][j]
        if(i > 0):
            y[i][j] = y[i][j] + (0.99 * y[i-1][j])
        if(j > 0):
            y[i][j] = y[i][j] + (0.99 * y[i][j-1])
        if(i > 0 and j > 0):
            y[i][j] = y[i][j] - (0.9801 * y[i-1][j-1])

out_img = y
y = np.uint8(y + 127)
print('after uint8 = ')
print(y)
plt.imshow(y, cmap=plt.cm.gray)
plt.title('Image')
plt.show()

img_out_2 = Image.fromarray(y.astype(np.uint8))
img_out_2.save('filter_random.tif')


x = np.linspace(-np.pi, np.pi, 1024)
y = np.linspace(-np.pi, np.pi, 1024)

def genComplexArray(m,n):
    return (np.arange(m) * 0j)[:, None] + np.arange(n)
z = genComplexArray(1024, 1024)
X, Y = np.meshgrid(x,y)



for m in range(0,len(x)) :
    for n in range(0,len(y)) :
        z[m][n] = 3 / (1 - (0.99 * cmath.exp(complex(0,-x[m]))) -
    (0.99 * cmath.exp(complex(0,-y[n]))) + 0.9801 * (cmath.exp(
    complex(0,-(x[m] + y[n])))))
Z = np.log((abs(z)**2)/12)
fig = plt.figure(1)
ax = plt.axes(projection='3d')
surf = ax.plot_surface(X, Y, Z, cmap=plt.cm.coolwarm)

ax.set_xlabel('$\mu$ axis')
ax.set_ylabel('$\\nu$ axis')
ax.set_zlabel('Z Label')

fig.colorbar(surf, shrink=0.5, aspect=5)

plt.show()

BetterSpectrumAnalyzer.BetterSpecAnalyzer(out_img)
```