Assignment 9: Spectra of Non-Periodic Signals

Manognya Param Koolath

April 16, 2019

The task

In this assignment, we learn to find frequency spectrums of non-periodic signals.

Question 1

We'll first look at $y = \sin(\sqrt{2}t)$. Obtained over 0 to 2π with 64 samples, the spectrum obtained is as shown in Figure 1. We expected two spikes, but

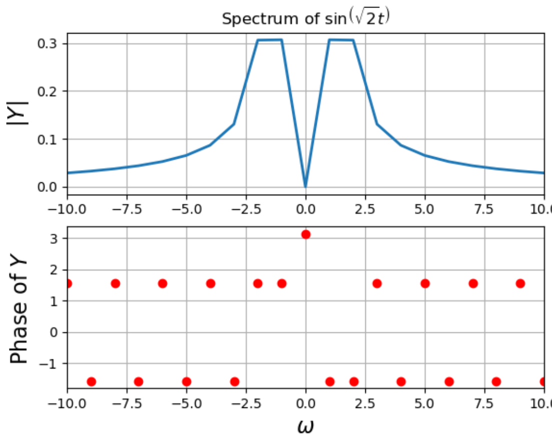

Figure 1: (Question 1)

what we got were two peaks each with two values and a gradually decaying magnitude. The phase is correct though. To understand what went wrong, we plot the time function over several time periods in Figure 2.

The blue line connects the points whose DFT we took. The red lines show the continuation of the function. Quite clearly $-\pi$ to π is not the part thatcan be replicated to get the function. What we just tried replicating is the function shown in Figure 3. We try to observe the Gibbs phenomena and hence have obtained and plotted the spectrum of a digital ramp as shown

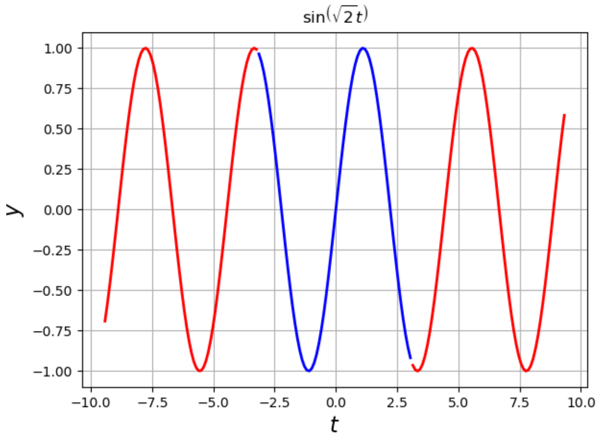

Figure 2: (Question 1)

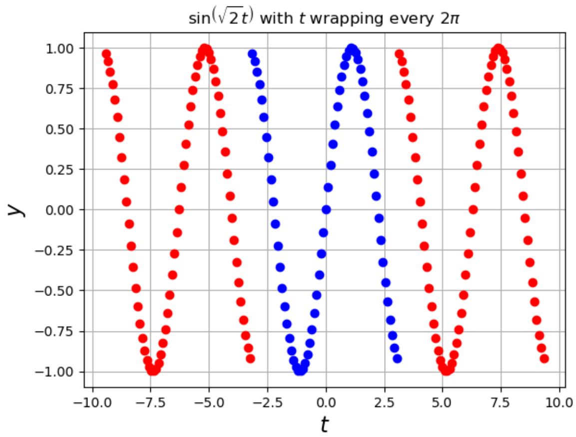

Figure 3: (Question 1)

in Figure 4. To remove the spikes at the end of the periodic interval, we window the function. The function we obtain after windowing is as shown in Figure 5. We have plotted the spectrum of this signal in Figure 6 to find that the magnitude is greatly improved. To improve it further we increase the number of points by four times. This one is plotted in Figure 7.

Question 2

Consider the following function:

$$y(t) = \cos^3(\omega_0 t)$$

We have to obtain the spectrum of this function. The obtained spectrum

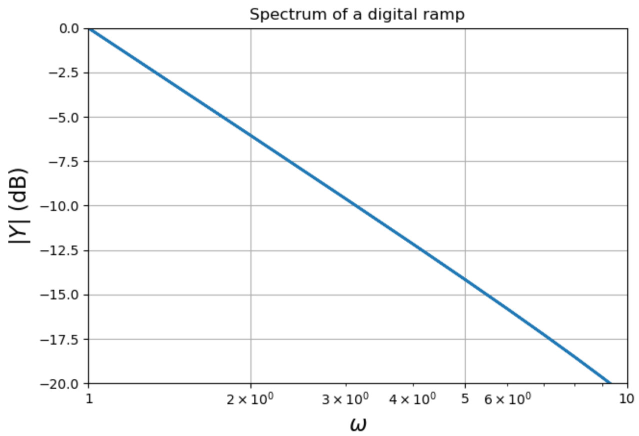

Figure 4: (Question 1)

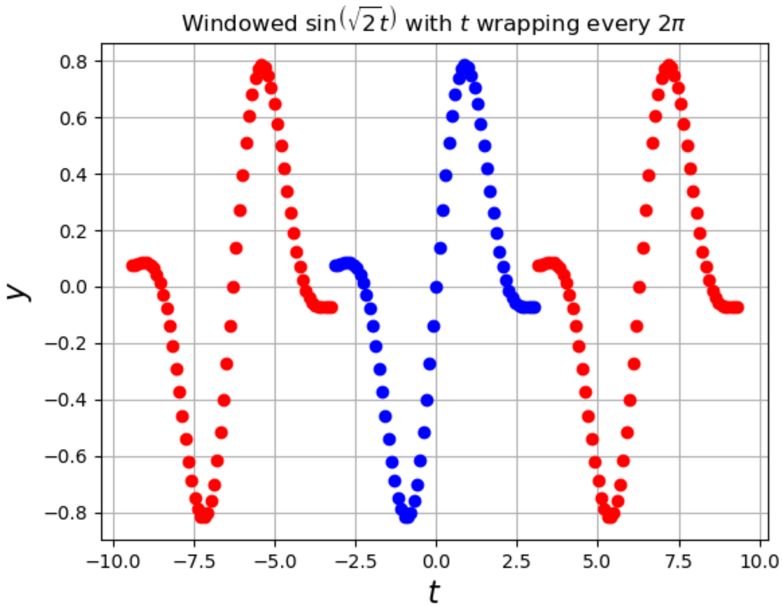

Figure 5: (Question 1)

is as shown in Figure 8.

Question 3

Now we have to generate the spectrum of the following for a given values of the following function and estimate ω_0 and δ.

$$y(t) = \cos(\omega_0 t + \delta)$$

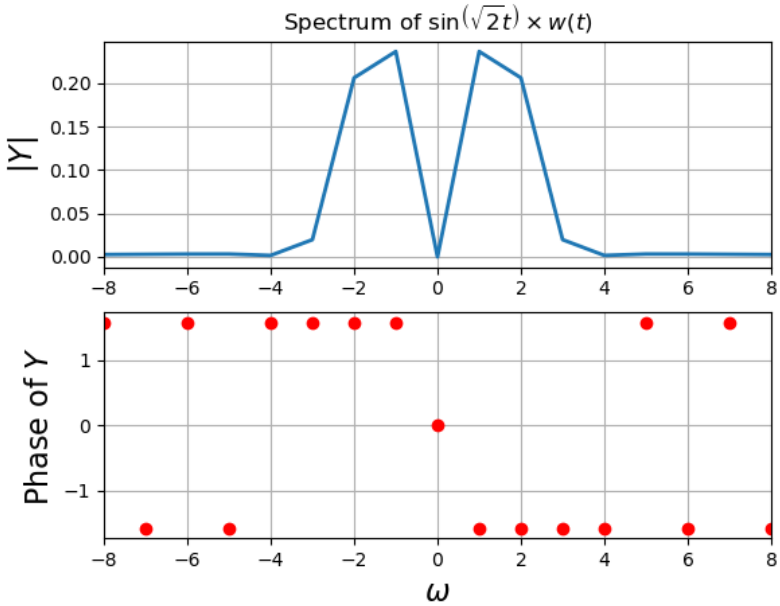

Figure 6: (Question 1)

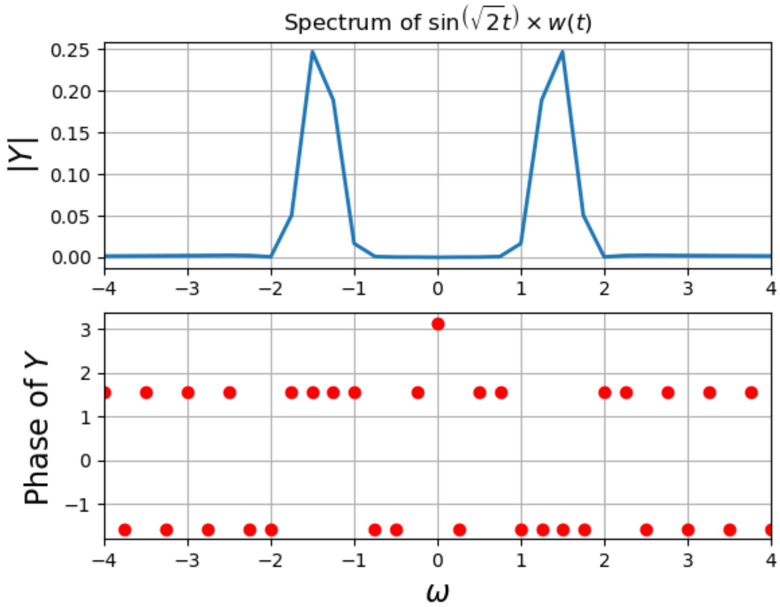

Figure 7: (Question 1)

On generating the spectrum, for particular values, we notice that the peak is obtained at ω_0 and the phase at this peak gives δ. But since we notice

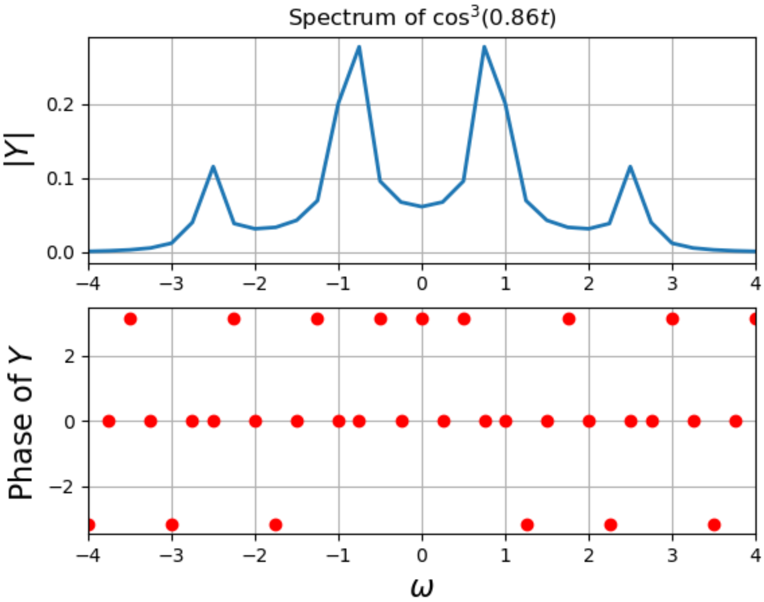

Figure 8: (Question 2)

that the precision is too low, we take a weighted average of the values at the peaks to get a better average. The plots are shown in Figure 9.

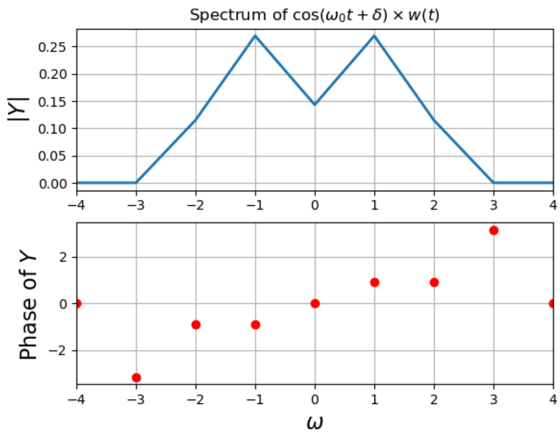

Figure 9: (Question 3)

Question 4

We have to repeat Question 3 with noisy values. The method used is the same and the plot is as shown in Figure 10.

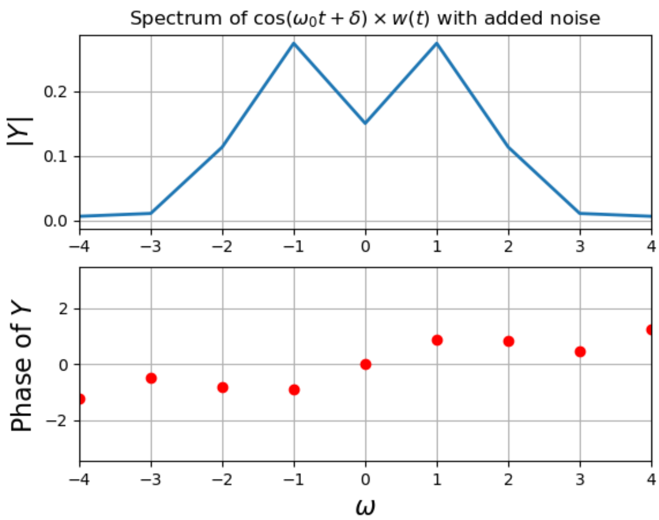

Figure 10: (Question 4)

Question 5

In this, we have to obtain and plot the DFT of the chirped signal:

$$y(t) = \cos(16(1.5 + \frac{t}{2\pi}))$$

We do this for t going from $-\pi$ to π in 1024 steps. The plot is shown in Figure 11. we observe that the frequency continuously changes from 16 to 32 radians per second. This also means that the period is 64 samples near $-\pi$ and is 32 near $+\pi$.

Question 6

For the same chirped signal, we break the 1024 vector into pieces that are 64 samples wide. The DFT of each is extracted and stored in a 2D array. Then a surface plot is plotted to observe how the frequency of the signal changes with time. The plots from different angles are shown in Figures 12-16.

Python code

The code is properly commented and completely vectorised.

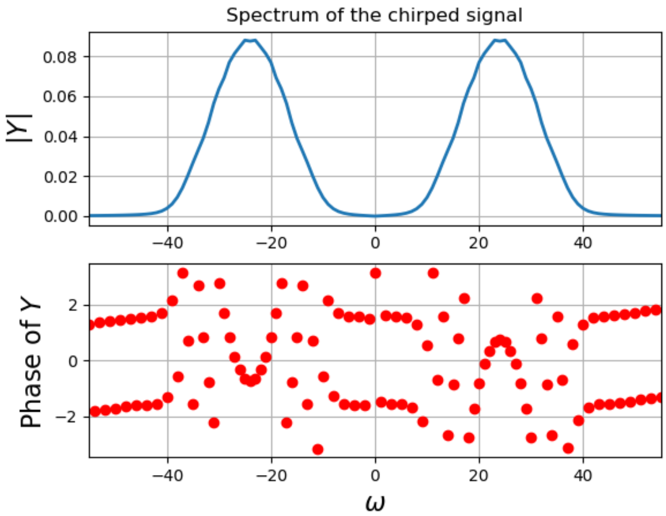

Figure 11: (Question 5)

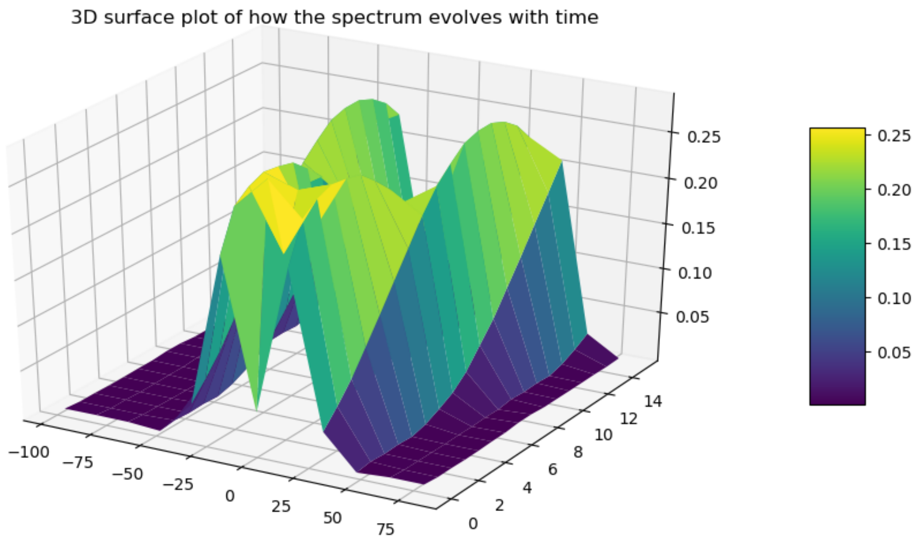

Figure 12: 3D surface plot (Question 6)

Assignment 9 Code

```
#Assignment 9
import numpy as np
import matplotlib.pyplot as plt
import scipy.signal as sp
from scipy.linalg import lstsq
```

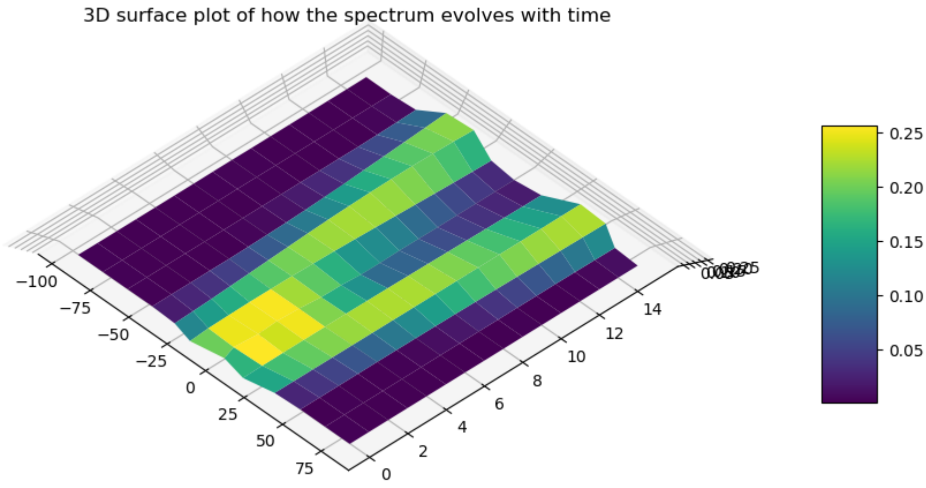

Figure 13: 3D surface plot (Question 6)

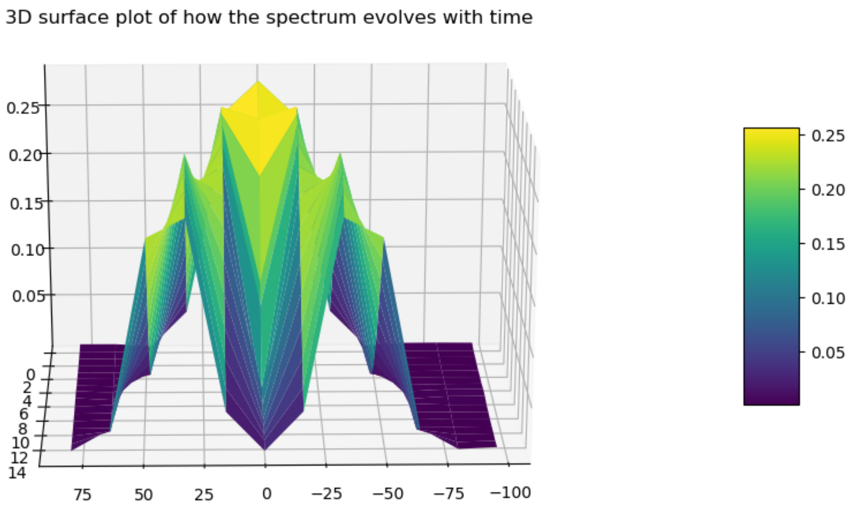

Figure 14: 3D surface plot (Question 6)

```
import scipy.integrate as spint
import mpl_toolkits.mplot3d.axes3d as p3
from scipy import ndimage
import sympy as spy

# Q1
```

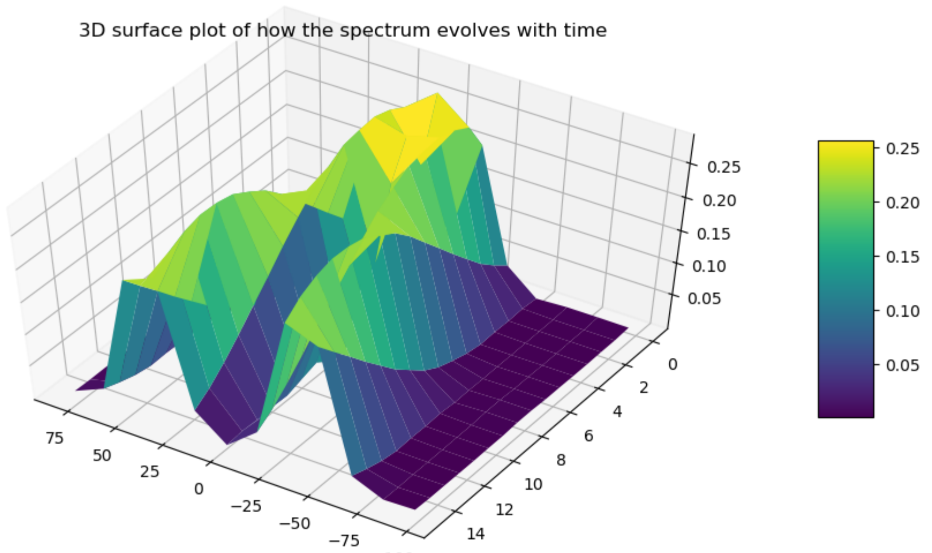

Figure 15: 3D surface plot (Question 6)

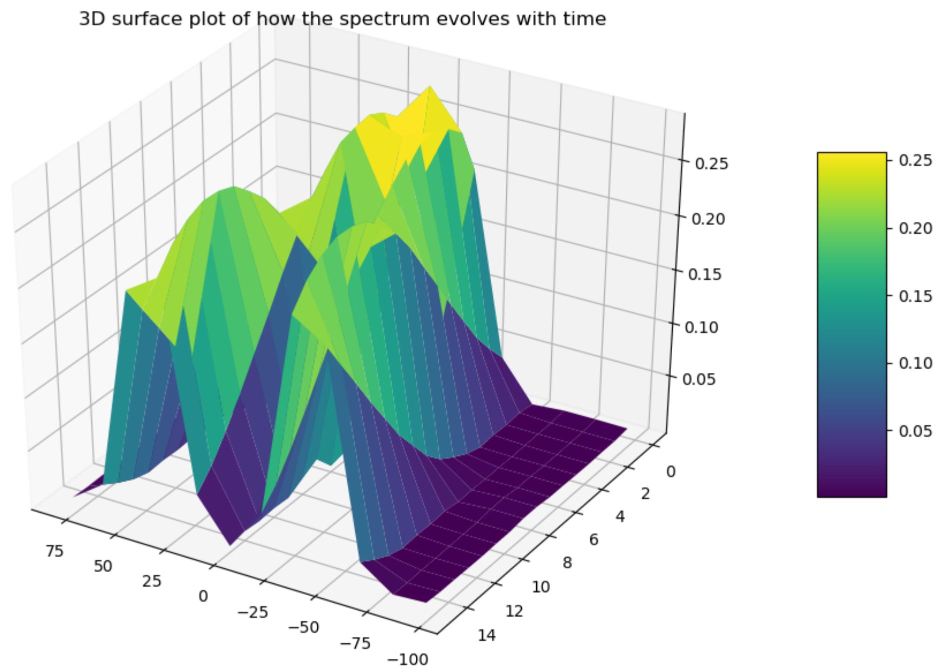

Figure 16: 3D surface plot (Question 6)

```
t=np.linspace(-1*np.pi,np.pi,65)
t=t[:-1]
dt=t[1]-t[0] ; fmax=1/dt #Why??
```

```
y=np.sin(np.sqrt(2)*t)
y[0]=0 #Why should the sample corresponding to -tmax be set to zero?
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/64.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,65)
w=w[:-1]
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,np.abs(Y),lw=2)
plt.xlim([-10,10])
plt.ylabel(r"$|Y|$",size=16)
plt.title(r"Spectrum of $\sin\left(\sqrt{2}t\right)$")
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(w, (np.angle(Y)),'ro',lw=2)#np.unwrap
plt.xlim([-10,10])
plt.ylabel(r"Phase of $Y$",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.grid(True)
# savefig("fig9-1.png")
plt.show()

t1 = np.linspace(-1*np.pi,np.pi,65)
t1 = t1[:-1]
t2 = np.linspace(-3*np.pi,-1*np.pi,65)
t2 = t2[:-1]
t3 = np.linspace(np.pi,3*np.pi,65)
t3 = t3[:-1]
plt.plot(t1, np.sin(np.sqrt(2)*t1),'b',lw=2)
plt.plot(t2,np.sin(np.sqrt(2)*t2),'r',lw=2)
plt.plot(t3,np.sin(np.sqrt(2)*t3),'r',lw=2)
plt.ylabel(r"$y$",size=16)
plt.xlabel(r"$t$",size=16)
plt.title(r"$\sin\left(\sqrt{2}t\right)$")
plt.grid(True)
plt.show()

# Use the last t1, t2, t3
y=np.sin(np.sqrt(2)*t1)
plt.plot(t1,y,'bo',lw=2)
plt.plot(t2,y,'ro',lw=2)
plt.plot(t3,y,'ro',lw=2)
plt.ylabel(r"$y$",size=16)
plt.xlabel(r"$t$",size=16)
```

```
plt.title(r"$\sin\left(\sqrt{2}t\right)$ with $t$ wrapping every $2\pi$")
plt.grid(True)
plt.show()

t=np.linspace(-1*np.pi,np.pi,65)
t=t[:-1]
dt=t[1]-t[0] ; fmax=1/dt
y=t
y[0]=0 # the sample corresponding to -tmax should be set zero
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/64.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,65)
w=w[:-1]
plt.figure()
plt.semilogx(np.abs(w), 20*np.log10(np.abs(Y)),lw=2)
plt.xlim([1,10])
plt.ylim([-20,0])
plt.xticks([1,2,5,10],["1"," ","5","10"],size=10)
plt.ylabel(r"$|Y|$ (dB)",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.title(r"Spectrum of a digital ramp")
plt.grid(True)
plt.show()

t1 = np.linspace(-1*np.pi,np.pi,65)
t1 = t1[:-1]
t2 = np.linspace(-3*np.pi,-1*np.pi,65)
t2 = t2[:-1]
t3 = np.linspace(np.pi,3*np.pi,65)
t3 = t3[:-1]
n = np.arange(64)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/63))
y = np.sin(np.sqrt(2)*t1) * wnd
plt.figure()
plt.plot(t1,y,'bo',lw=2)
plt.plot(t2,y,'ro',lw=2)
plt.plot(t3,y,'ro',lw=2)
plt.ylabel(r"$y$",size=16)
plt.xlabel(r"$t$",size=16)
plt.title(r"Windowed $\sin\left(\sqrt{2}t\right)$ with $t$ wrapping every
plt.grid(True)
plt.show()

#DFT of this sequence:
```

```
t=np.linspace(-1*np.pi,np.pi,65)
t=t[:-1]
dt=t[1]-t[0] ; fmax=1/dt
n = np.arange(64)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/64))
y = np.sin(np.sqrt(2)*t) * wnd
y[0] = 0 #the sample corresponding to -tmax should be set zero
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/64.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,65)
w=w[:-1]
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,np.abs(Y),lw=2)
plt.xlim([-8,8])
plt.ylabel(r"$|Y|$",size=16)
plt.title(r"Spectrum of $\sin\left(\sqrt{2}t\right)\times w(t)$")
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(w, (np.angle(Y)),'ro',lw=2)#np.unwrap
plt.xlim([-8,8])
plt.ylabel(r"Phase of $Y$",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.grid(True)
# savefig("fig9-1.png")
plt.show()

t=np.linspace(-4*np.pi,4*np.pi,257)
t=t[:-1]
dt=t[1]-t[0] ; fmax=1/dt
n = np.arange(256)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/256))
y = np.sin(np.sqrt(2)*t) * wnd
y[0] = 0 #the sample corresponding to -tmax should be set zero
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/256.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,257)
w=w[:-1]
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,np.abs(Y),lw=2)
plt.xlim([-4,4])
plt.ylabel(r"$|Y|$",size=16)
plt.title(r"Spectrum of $\sin\left(\sqrt{2}t\right)\times w(t)$")
```

```
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(w, (np.angle(Y)),'ro',lw=2)#np.unwrap
plt.xlim([-4,4])
plt.ylabel(r"Phase of $Y$",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.grid(True)
# savefig("fig9-1.png")
plt.show()

#Q2

t=np.linspace(-4*np.pi,4*np.pi,257)
t=t[:-1]
dt=t[1]-t[0] ; fmax=1/dt
n = np.arange(256)
# wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/256))
w0 = 0.86
y = np.power(np.cos(w0*t),3) #* wnd
y[0] = 0 #the sample corresponding to -tmax should be set zero
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/256.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,257)
w=w[:-1]
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,np.abs(Y),lw=2)
plt.xlim([-4,4])
plt.ylabel(r"$|Y|$",size=16)
plt.title(r"Spectrum of $\cos^3\left(0.86t\right)$")
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(w, (np.angle(Y)),'ro',lw=2)#np.unwrap
plt.xlim([-4,4])
plt.ylabel(r"Phase of $Y$",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.grid(True)
# savefig("fig9-1.png")
plt.show()

#Q3:

t=np.linspace(-1*np.pi,np.pi,129)
t=t[:-1]
```

```
dt=t[1]-t[0] ; fmax=1/dt
n = np.arange(128)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/128))
w0=1.2
delta=0.9
y = np.cos(w0*t + delta) * wnd
y[0] = 0 #the sample corresponding to -tmax should be set zero
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/128.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,129)
w=w[:-1]
# print(w.shape, Y.shape)
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,np.abs(Y),lw=2)
# print(Y)
l=int(len(Y)/2)
# print(l)
y1=np.abs(Y[l:])
# print(y1)
freq = (1*y1[1]+2*y1[2])/(y1[1]+y1[2])
print("Frequency = ", freq)
# print("Frequency = ", np.abs(w[np.argmax(np.abs(Y))]))
print("Phase difference = ", np.abs(np.angle(Y[np.argmax(np.abs(Y))])))
plt.xlim([-4,4])
plt.ylabel(r"$|Y|$",size=16)
plt.title(r"Spectrum of $\cos\left(\omega_0t + \delta\right)\times w(t)$"
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(w, (np.angle(Y)),'ro',lw=2)#np.unwrap
plt.xlim([-4,4])
plt.ylabel(r"Phase of $Y$",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.grid(True)
# savefig("fig9-1.png")
plt.show()

#Q4:

t=np.linspace(-1*np.pi,np.pi,129)
t=t[:-1]
dt=t[1]-t[0] ; fmax=1/dt
n = np.arange(128)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/128))
```

```
w0=1.0
delta=0.9
y = np.cos(w0*t + delta) * wnd
y = y + 0.1*np.random.randn(128) #Only this line is different from Q3.
y[0] = 0 #the sample corresponding to -tmax should be set zero
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/128.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,129)
w=w[:-1]
# print(w.shape, Y.shape)
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,np.abs(Y),lw=2)
# print(Y)
l=int(len(Y)/2)
# print(l)
y1=np.abs(Y[l:])
# print(y1)
freq = (1*y1[1]+2*y1[2])/(y1[1]+y1[2])
print("Frequency = ", freq)
# print("Frequency = ", np.abs(w[np.argmax(np.abs(Y))]))
print("Phase difference = ", np.abs(np.angle(Y[np.argmax(np.abs(Y))])))
plt.xlim([-4,4])
plt.ylabel(r"$|Y|$",size=16)
plt.title(r"Spectrum of $\cos\left(\omega_0t + \delta\right)\times w(t)$ 
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(w, (np.angle(Y)),'ro',lw=2)#np.unwrap
plt.xlim([-4,4])
plt.ylabel(r"Phase of $Y$",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.grid(True)
# savefig("fig9-1.png")
plt.show()

#Q5:

t=np.linspace(-1*np.pi,np.pi,1025)
t=t[:-1]
dt=t[1]-t[0] ; fmax=1/dt
n = np.arange(1024)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/1024))
y = np.cos(16*np.multiply((1.5 + (t/(2*np.pi))),t)) * wnd
y[0] = 0 #the sample corresponding to -tmax should be set zero
```

```
y=np.fft.fftshift(y)
Y=np.fft.fftshift(np.fft.fft(y))/1024.0
w=np.linspace(-np.pi*fmax,np.pi*fmax,1025)
w=w[:-1]
plt.figure()
plt.subplot(2,1,1)
plt.plot(w,np.abs(Y),lw=2)
plt.xlim([-55,55])
plt.ylabel(r"$|Y|$",size=16)
plt.title(r"Spectrum of the chirped signal")
plt.grid(True)
plt.subplot(2,1,2)
plt.plot(w, (np.angle(Y)),'ro',lw=2)#np.unwrap
plt.xlim([-55,55])
plt.ylabel(r"Phase of $Y$",size=16)
plt.xlabel(r"$\omega$",size=16)
plt.grid(True)
# savefig("fig9-1.png")
plt.show()

#Q6:

x = np.split(t,16)
# print(x)
# We use fmax, dt the same as Q5.
w=np.linspace(-np.pi*fmax,np.pi*fmax,65)
w=w[:-1]
n = np.arange(64)
wnd = np.fft.fftshift(0.54+0.46*np.cos(2*np.pi*n/64))
A=np.full((64,0),0.0)
for i in range(16):
        t=x[i]
        y = np.cos(16*np.multiply((1.5 + (t/(2*np.pi))),t)) * wnd
        y[0] = 0
        y=np.fft.fftshift(y)
        Y=np.fft.fftshift(np.fft.fft(y))/64.0
        # print(Y[:,None].shape, A.shape)
        A = np.concatenate((A,Y[:,None]),axis=1)
A=np.abs(A)
# print(A.shape, w.shape)
time = np.arange(16)
# x-axis is w, y-axis is time.

w=w[26:-26]
```

```python
F,T = np.meshgrid(w,time)
A=A[26:-26]
# print(w)
print(F.shape, T.shape, A.T.shape)
# Y, X = np.meshgrid(y,x)
fig1 = plt.figure()#figure(4)
ax=p3.Axes3D(fig1)
plt.title('3D surface plot of how the spectrum evolves with time')
surf = ax.plot_surface(F, T, A.T, rstride=1, cstride=1, cmap='viridis',li
# ax.set_xlim3d(-100, 100)
# ax.axis('equal')
fig1.colorbar(surf, shrink=0.5, aspect=5)
plt.show()
```