EE2703: Assignment 10

Yogesh Agarwala
EE19B130

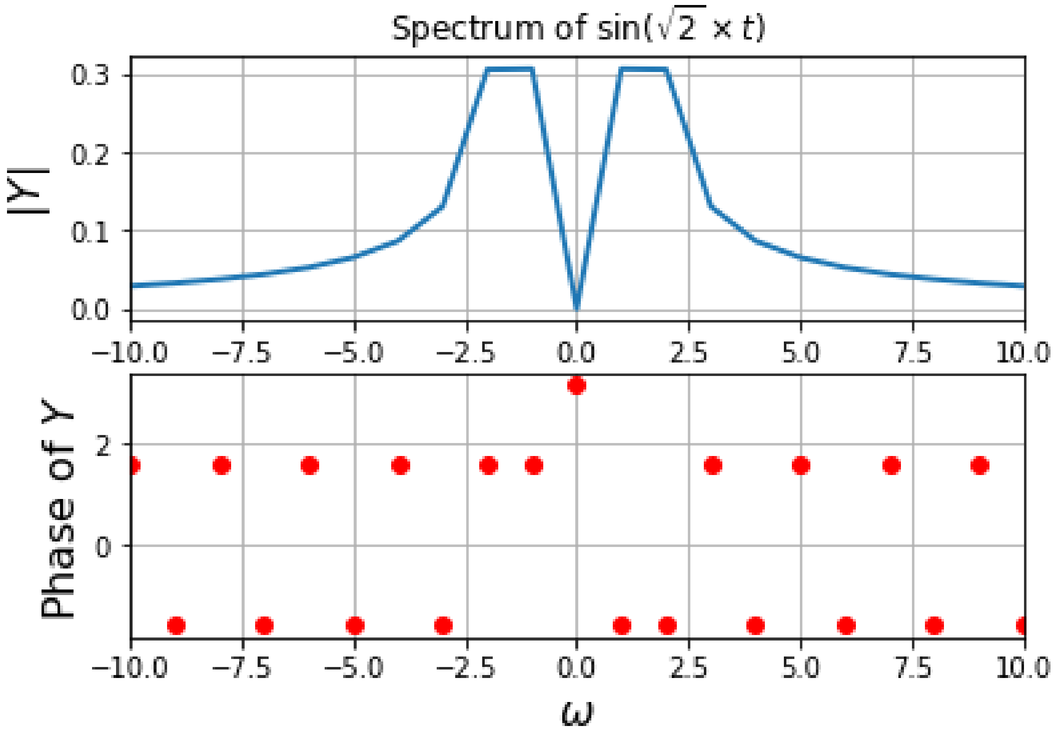
June 3, 2021

[1]:
```
from pylab import*
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
```

1 1. Examples

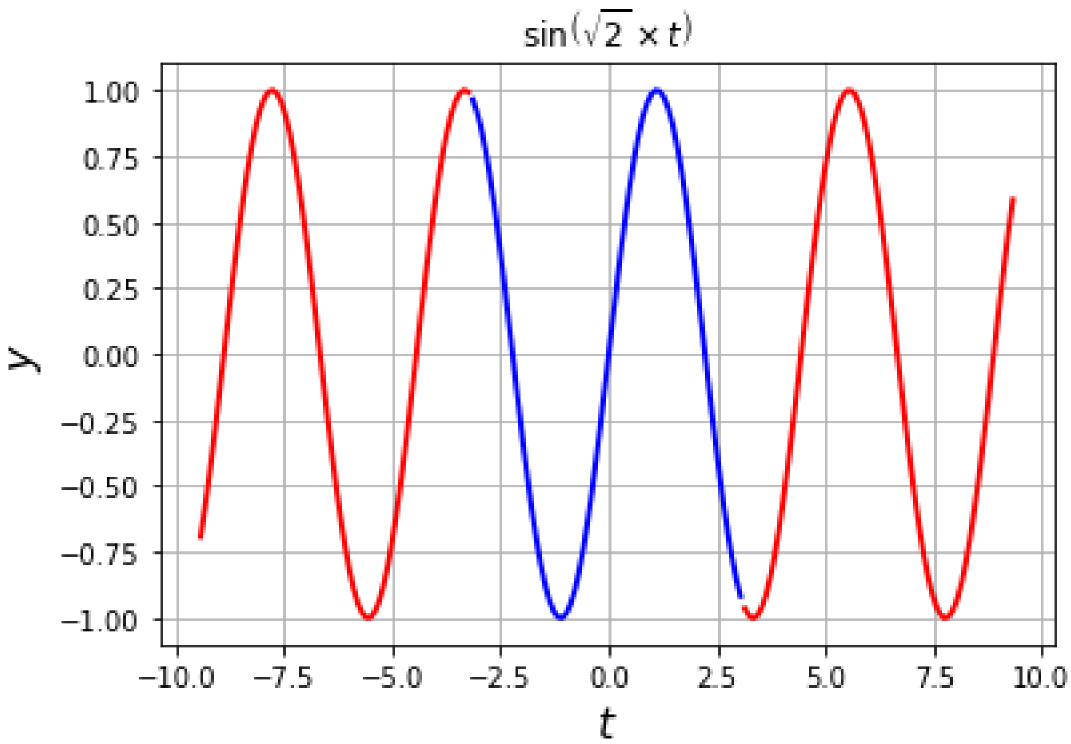
1.1 $y = \sin\left(\sqrt{2} \times t\right)$ over $-\pi$ to $+\pi$ with 64 samples

[2]:
```
t=linspace(-pi,pi,65)[:-1]
dt=t[1]-t[0];fmax=1/dt
y=sin(sqrt(2)*t)
y[0]=0 # the sample corresponding to -tmax should be set zero
y=fftshift(y) # make y start with y(t=0)
Y=fftshift(fft(y))/64.0
w=linspace(-pi*fmax,pi*fmax,65)[:-1]

figure()
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-10,10])
ylabel(r"$|Y|$",size=16)
title(r"Spectrum of $\sin(\sqrt{2}\times t)$")
grid(True)

subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-10,10])
ylabel(r"Phase of $Y$",size=16)
xlabel(r"$\omega$",size=16)
grid(True)
show()
```



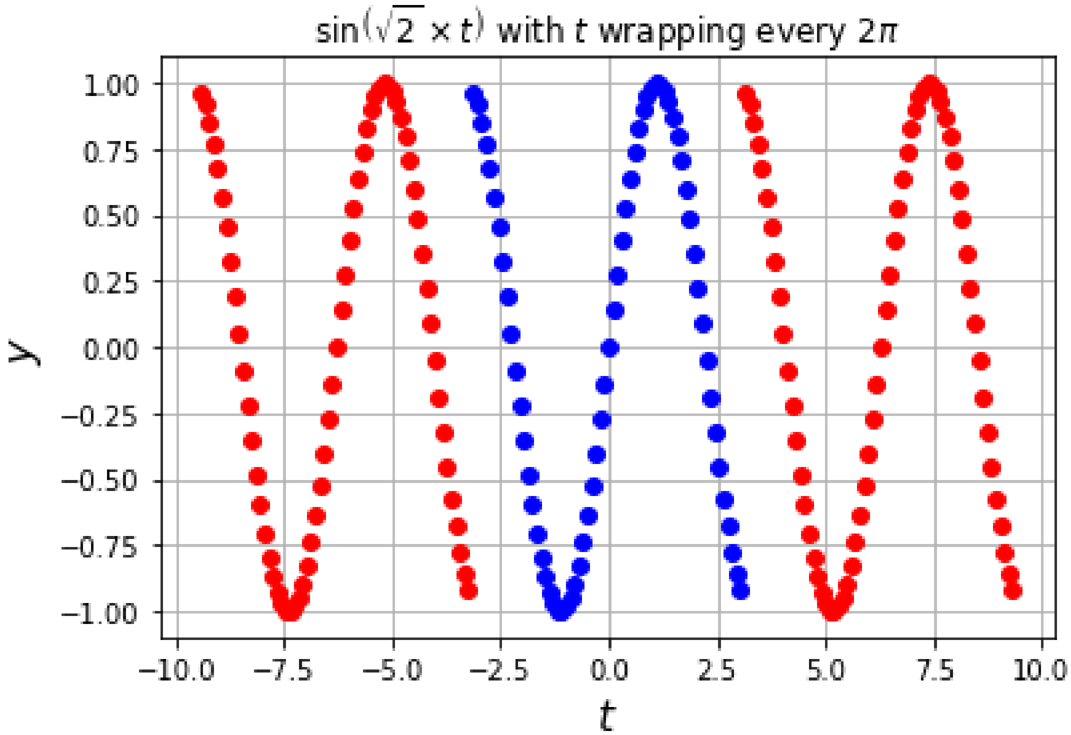
1.2 $y = \sin\left(\sqrt{2} \times t\right)$ over several time periods b/w -3π to $+3\pi$

[3]:
```
t1=linspace(-pi,pi,65)[:-1]
t2=linspace(-3*pi,-pi,65)[:-1]
t3=linspace(pi,3*pi,65)[:-1]
# y=sin(sqrt(2)*t)

figure(2)
plot(t1,sin(sqrt(2)*t1),'b',lw=2)
plot(t2,sin(sqrt(2)*t2),'r',lw=2)
plot(t3,sin(sqrt(2)*t3),'r',lw=2)
ylabel(r"$y$",size=16)
xlabel(r"$t$",size=16)
title(r"$\sin\left(\sqrt{2}\times t\right)$")
grid(True)
show()
```



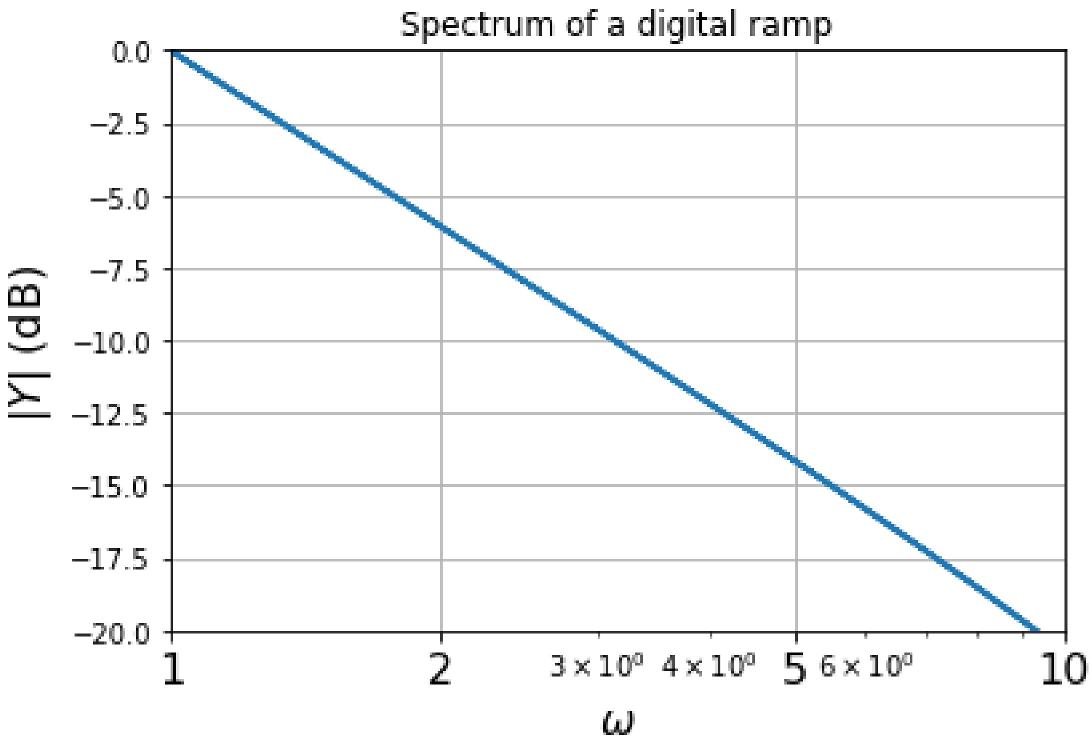
1.3 $y = \sin\left(\sqrt{2} \times t\right)$ with t wrapping every 2π

[4]:
```
t1=linspace(-pi,pi,65)[:-1]
t2=linspace(-3*pi,-pi,65)[:-1]
t3=linspace(pi,3*pi,65)[:-1]
y=sin(sqrt(2)*t1)

figure(3)
plot(t1,y,'bo')
plot(t2,y,'ro')
plot(t3,y,'ro')
ylabel(r"$y$",size=16)
xlabel(r"$t$",size=16)
title(r"$\sin\left(\sqrt{2}\times t\right)$ with $t$ wrapping every $2\pi$ ")
grid(True)
show()
```



1.4 $y = t$

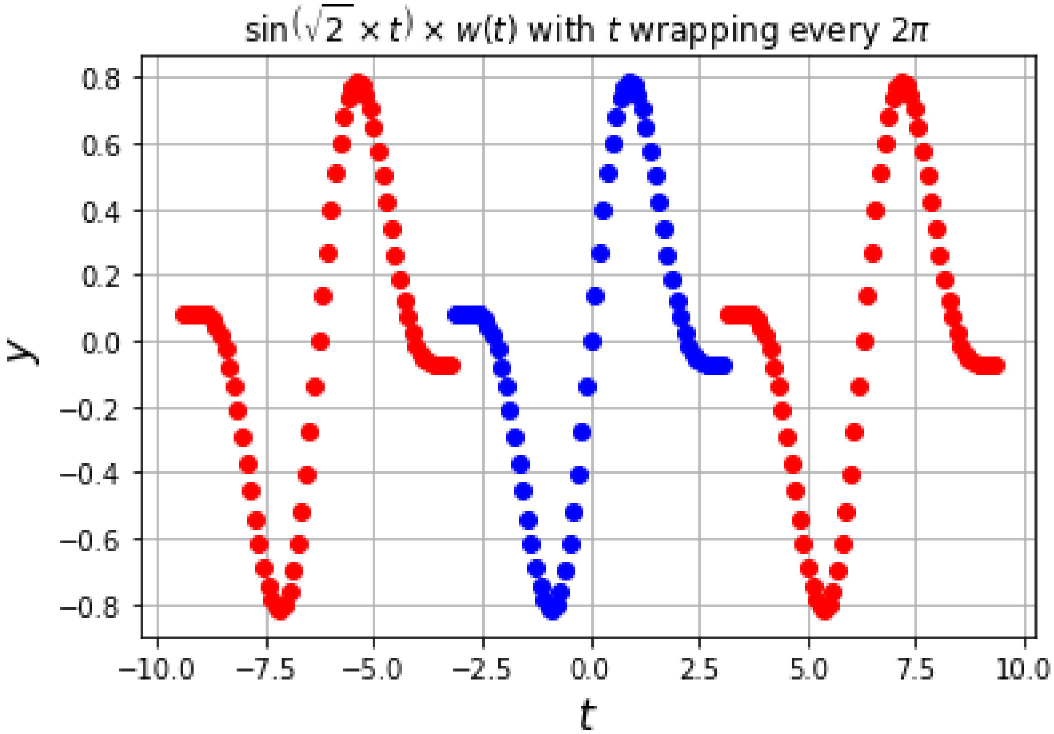
[5]:
```
t=linspace(-pi,pi,65)[:-1]
dt=t[1]-t[0]
fmax=1/dt
y=t
y[0]=0 # the sample corresponding to -tmax should be set zeroo
y=fftshift(y) # make y start with y(t=0)
Y=fftshift(fft(y))/64.0 #noramlisation
w=linspace(-pi*fmax,pi*fmax,65)[:-1]

figure()
semilogx(abs(w),20*log10(abs(Y)),lw=2)
xlim([1,10])
ylim([-20,0])
xticks([1,2,5,10],["1","2","5","10"],size=16)
ylabel(r"$|Y|$ (dB)",size=16)
title(r"Spectrum of a digital ramp")
xlabel(r"$\omega$",size=16)
grid(True)
show()
```



1.5 $y = \sin\left(\sqrt{2} \times t\right) \times w(t)$ with t wrapping every 2π

[6]:

```
t1=linspace(-pi,pi,65)[:-1]
t2=linspace(-3*pi,-pi,65)[:-1]
t3=linspace(pi,3*pi,65)[:-1]
n=arange(64)
wnd=fftshift(0.54+0.46*cos(2*pi*n/63))
y=sin(sqrt(2)*t1)*wnd

figure(3)
plot(t1,y,'bo')
plot(t2,y,'ro')
plot(t3,y,'ro')
ylabel(r"$y$",size=16)
xlabel(r"$t$",size=16)
title(r"$\sin\left(\sqrt{2}\times t\right)\times w(t)$ with $t$ wrapping every␣
↪$2\pi$ ")
grid(True)
show()
```

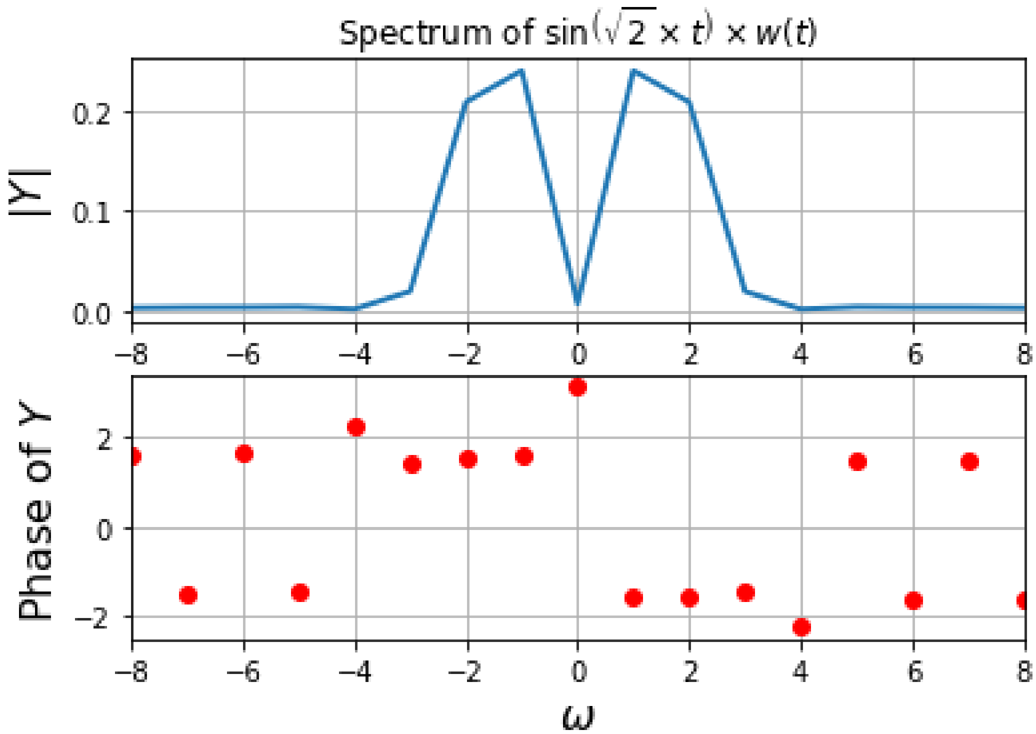


1.6 Spectrum of $y = \sin\left(\sqrt{2} \times t\right) \times w(t)$ with 64 samples

[7]:

```
t=linspace(-pi,pi,65)[:-1]
dt=t[1]-t[0];
fmax=1/dt
n=arange(64)
wnd=fftshift(0.54+0.46*cos(2*pi*n/63))
y=sin(sqrt(2)*t)*wnd
y[0]=0 # the sample corresponding to -tmax should be set zeroo
y=fftshift(y) # make y start with y(t=0)
Y=fftshift(fft(y))/64.0 #normalisation
w=linspace(-pi*fmax,pi*fmax,65)[:-1]

figure()
subplot(2,1,1)
plot(w,abs(Y),lw=2)
xlim([-8,8])
ylabel(r"$|Y|$",size=16)
title(r"Spectrum of $\sin\left(\sqrt{2}\times t\right)\times w(t)$")
grid(True)

subplot(2,1,2)
```

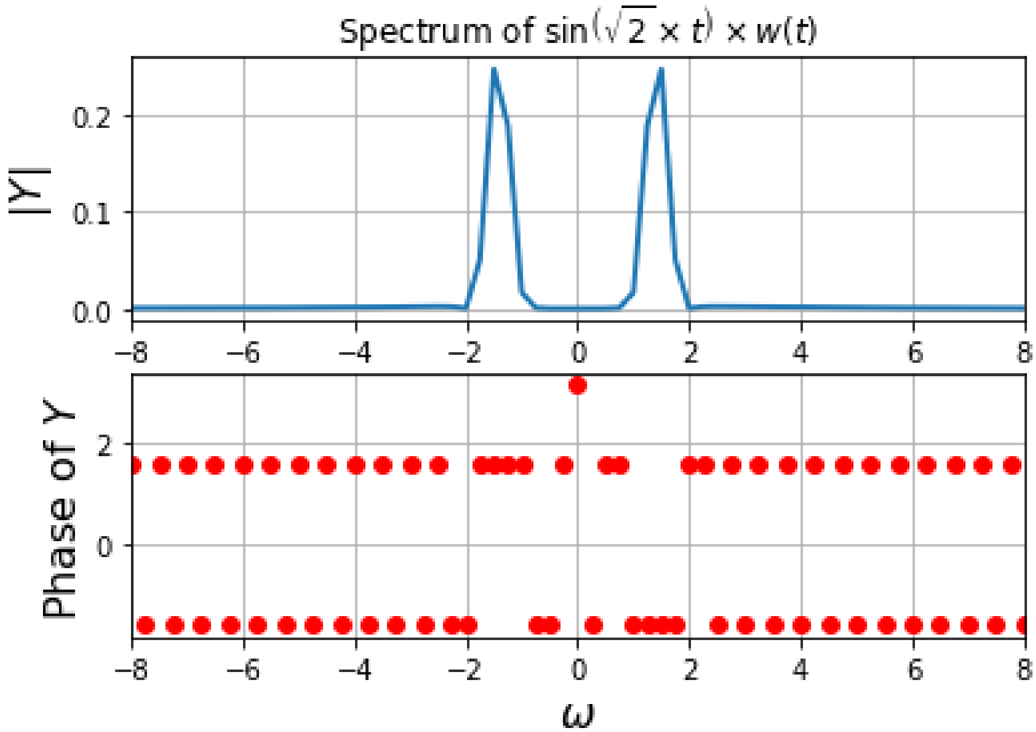
```
plot(w,angle(Y),'ro',lw=2)
xlim([-8,8])
ylabel(r"Phase of $Y$",size=16)
xlabel(r"$\omega$",size=16)
grid(True)
show()
```



1.7 Spectrum of $y = \sin\left(\sqrt{2} \times t\right) \times w(t)$ with 256 samples

[8]:
```
t=linspace(-4*pi,4*pi,257)[:-1]
dt=t[1]-t[0];fmax=1/dt
n=arange(256)
wnd=fftshift(0.54+0.46*cos(2*pi*n/256))
y=sin(sqrt(2)*t)
y=y*wnd
y[0]=0 # the sample corresponding to -tmax should be set zeroo
y=fftshift(y) # make y start with y(t=0)
Y=fftshift(fft(y))/256.0
w=linspace(-pi*fmax,pi*fmax,257)[:-1]

figure()
subplot(2,1,1)
```

```
plot(w,abs(Y),lw=2)
xlim([-8,8])
ylabel(r"$|Y|$",size=16)
title(r"Spectrum of $\sin\left(\sqrt{2}\times t\right)\times w(t)$")
grid(True)

subplot(2,1,2)
plot(w,angle(Y),'ro',lw=2)
xlim([-8,8])
ylabel(r"Phase of $Y$",size=16)
xlabel(r"$\omega$",size=16)
grid(True)
show()
```

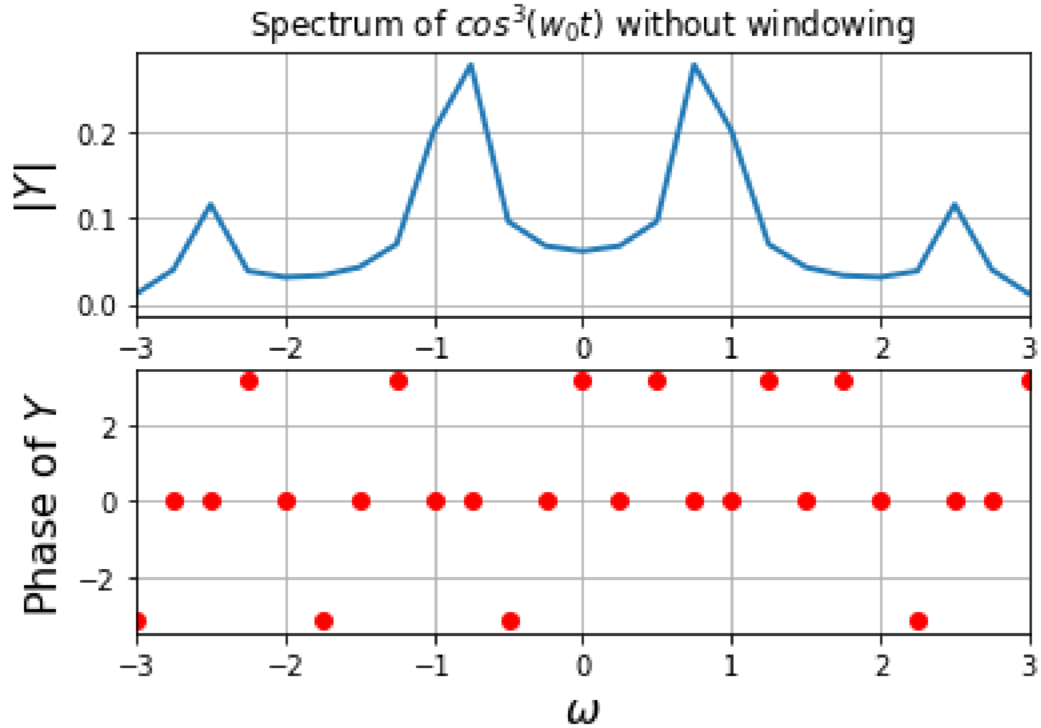


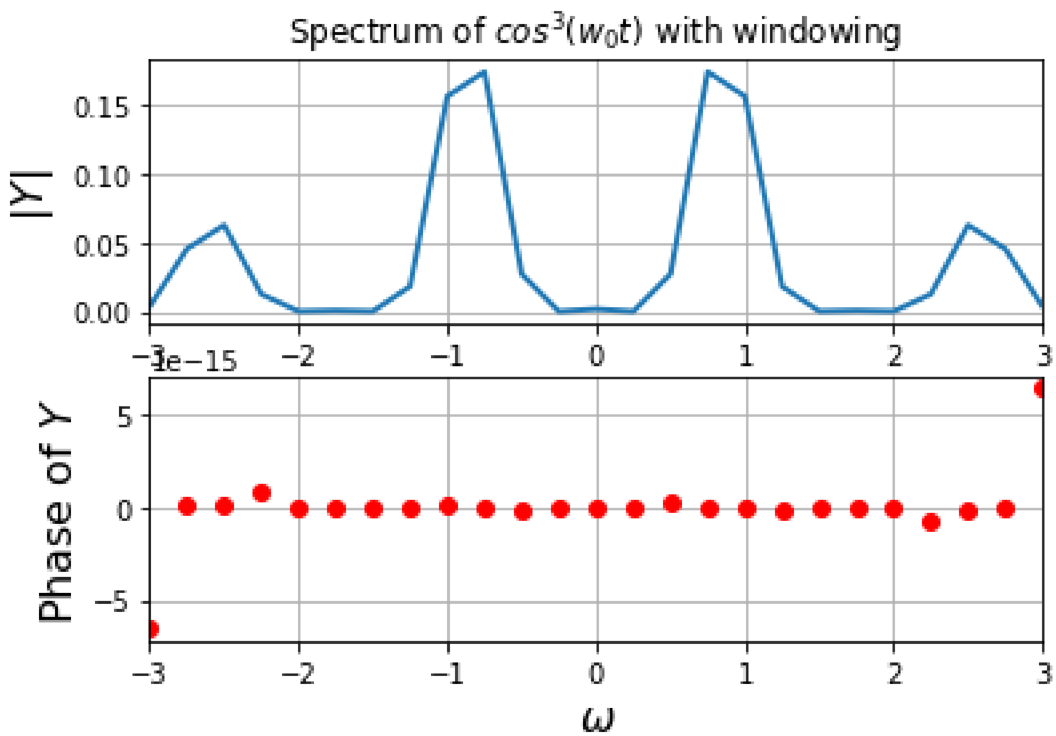
2 2. $y = cos^3(w_0 t)$

[54]:
```
def Plotting_Spectrum(lim,n,f,windowing=False,xlim1=10,title1 = "Spectrum"):
    t=linspace(-lim,lim,n+1)[:-1]
    dt=t[1]-t[0];
    fmax=1/dt
    y = f(t)
```

```
    if (windowing):
        m=arange(n)
        wnd=fftshift(0.54+0.46*cos(2*pi*m/n))
        y = y*wnd

    y[0]=0 # the sample corresponding to -tmax should be set zeroo
    y=fftshift(y) # make y start with y(t=0)
    Y=fftshift(fft(y))/float(n) #normalisation
    w=linspace(-pi*fmax,pi*fmax,n+1)[:-1]

    mag = abs(Y)
    phase = angle(Y)

    figure()
    subplot(2,1,1)
    plot(w,mag,lw=2)
    xlim([-xlim1,xlim1])
    ylabel(r"$|Y|$",size=16)
    title(title1)
    grid(True)

    subplot(2,1,2)
    phase[where(mag<3e-3)] = 0
    plot(w,phase,'ro')
    xlim([-xlim1,xlim1])
    ylabel(r"Phase of $Y$",size=16)
    xlabel(r"$\omega$",size=16)
    grid(True)
    show()

    return w,Y
```

[55]:
```
def cos3(t,w0=0.86):
    return (cos(w0*t))**3

"""
FFT without windowing
"""
w,Y = Plotting_Spectrum(4*pi,64*4,cos3,xlim1= 3,windowing=False, title1 =␣
 ↪r"Spectrum of $cos^3(w_0t)$ without windowing")

"""
FFT with windowing
"""
w,Y = Plotting_Spectrum(4*pi,64*4,cos3,xlim1= 3,windowing=True, title1 =␣
 ↪r"Spectrum of $cos^3(w_0t)$ with windowing")
```

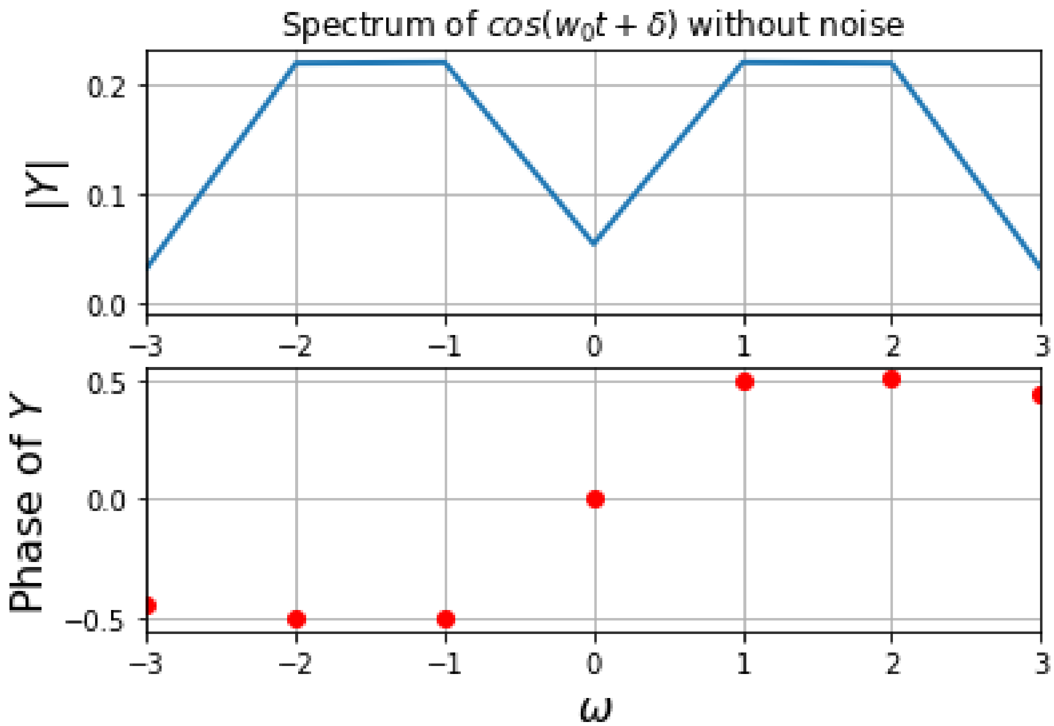

Spectrum of $cos^3(w_0t)$ without windowing
|Y|
Phase of Y
ω


Spectrum of $cos^3(w_0t)$ with windowing
|Y|
1e−15
Phase of Y
ω

3 3. $y = cos(w_0 t + \delta)$ without noise

[56]:
```
def Estimate_omega(w,Y):
    ii = where(w>0)
    omega = sum(abs(Y[ii])**2*w[ii])/sum(abs(Y[ii])**2)
    return omega

def Estimate_delta(w,Y):
    sup = 1e-4
    ii_1=np.where(np.logical_and(np.abs(Y)>sup, w>0))[0]
    np.sort(ii_1)
    window = 1
    points=ii_1[1:window+1]
    delta = np.sum(np.angle(Y[points]))/len(points)
    return delta
```

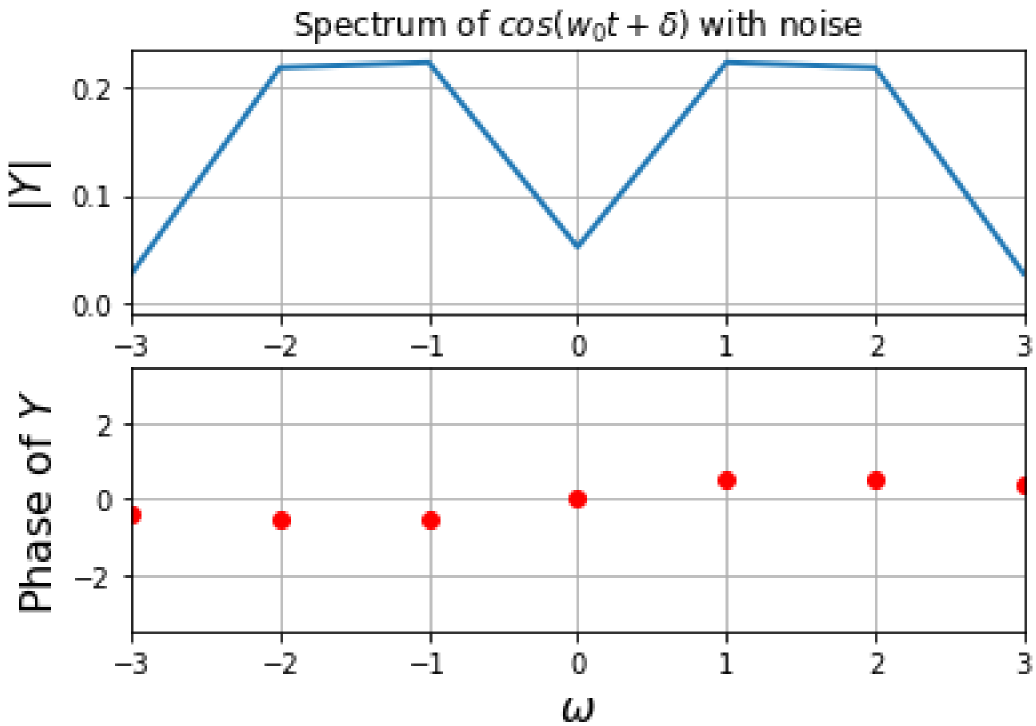
[57]:
```
#FFT of cos(wt+delta) windowed to estimate w, delta
def cos_without_noise(t,w0=1.5,delta=0.5):
    return cos(w0*t + delta)

w,Y = Plotting_Spectrum(pi,128,cos_without_noise,xlim1= 3,windowing=True, title1␣
 ↪= r"Spectrum of $cos(w_0t + \delta)$ without noise")

print(f"Estimates for w and delta (without noise) are : {Estimate_omega(w,Y)},␣
 ↪{Estimate_delta(w,Y)}")
```




```
Estimates for w and delta (without noise) are : 1.5163179648582412,
0.506776265719626
```

4 4. $y = cos(w_0t + \delta)$ with noise

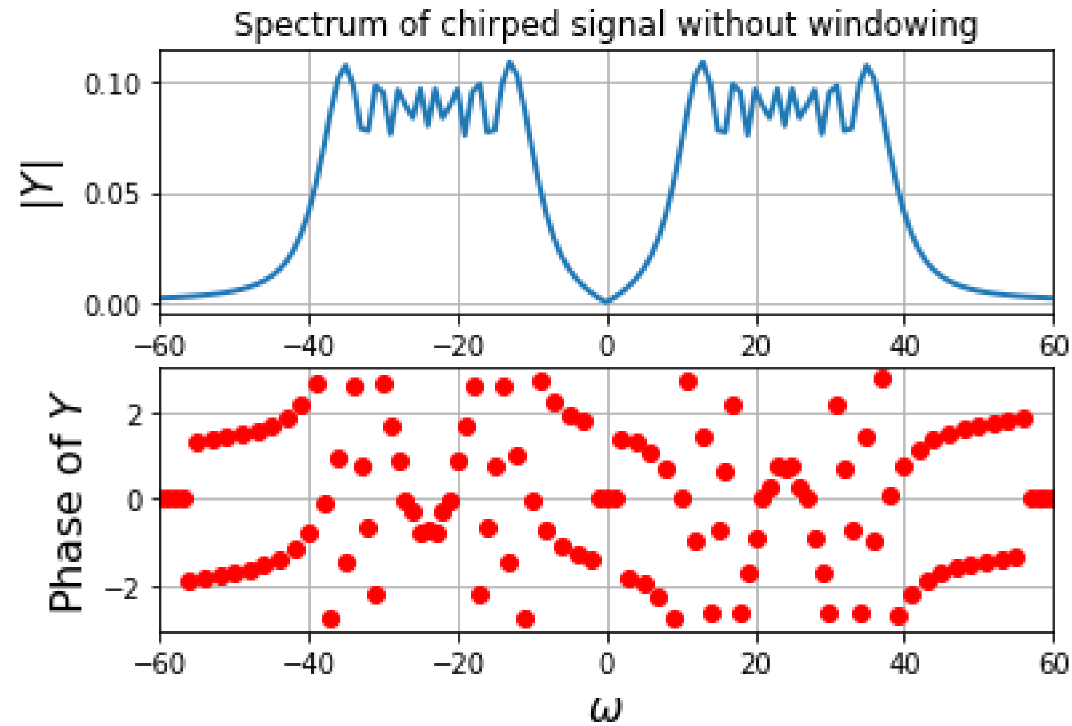
[58]:
```
def cos_with_noise(t,w0=1.5,delta=0.5):
    return cos(w0*t + delta) + 0.1*np.random.randn(128)

w,Y = Plotting_Spectrum(pi,128,cos_with_noise,xlim1= 3,windowing=True, title1 =␣
↪r"Spectrum of $cos(w_0t + \delta)$ with noise")

print(f"Estimates for w and delta (with noise) are : {Estimate_omega(w,Y)},␣
↪{Estimate_delta(w,Y)}")
```

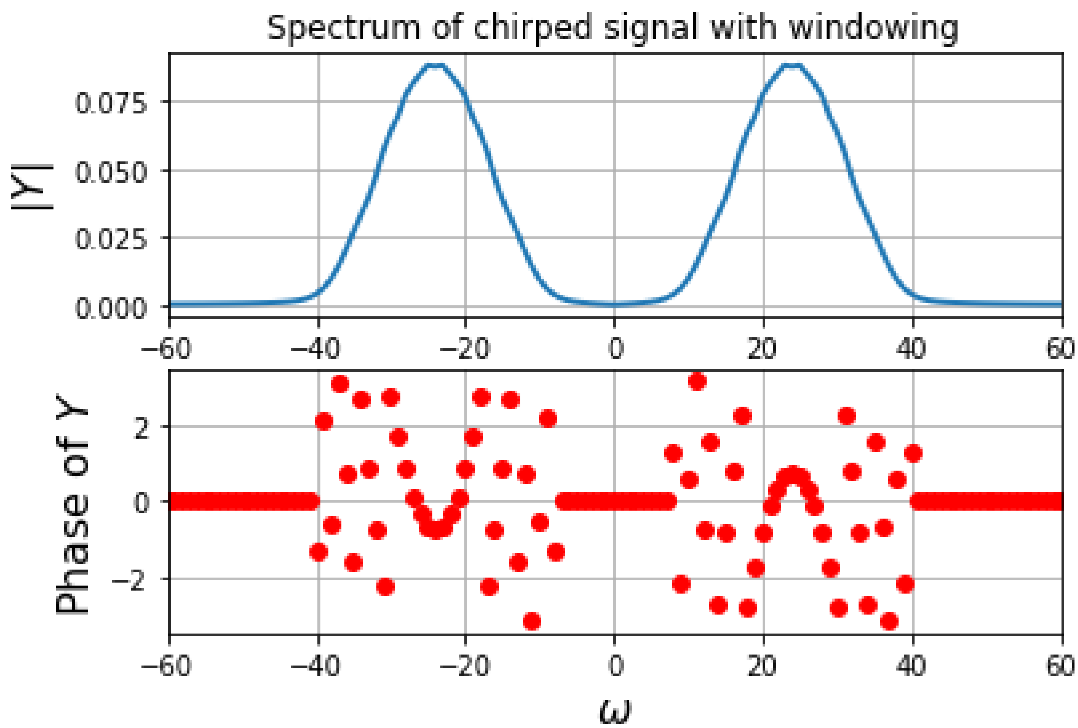



```
Estimates for w and delta (with noise) are : 2.0504716417047724,
0.5069474181511943
```

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

[59]:
```
def chirped_signal(t):
    return cos(16*(1.5 + t/(2*pi))*t)

"""
FFT without windowing
"""
w,Y = Plotting_Spectrum(pi,1024,chirped_signal,xlim1= 60,windowing=False, title1␣
 ↪= r"Spectrum of chirped signal without windowing")

"""
FFT with windowing
"""
w,Y = Plotting_Spectrum(pi,1024,chirped_signal,xlim1= 60,windowing=True, title1␣
 ↪= r"Spectrum of chirped signal with windowing")
```


Spectrum of chirped signal without windowing
|Y|
Phase of Y
ω


Spectrum of chirped signal with windowing
|Y|
Phase of Y
ω

6 6. Time-Frequency surface plot of $y = \cos\left(16\left(1.5 + \frac{t}{2\pi}\right)t\right)$

[62]:
```
t=np.linspace(-np.pi,np.pi,1025);t=t[:-1]
t_arrays=np.split(t,16)

Y_mags=np.zeros((16,64))
Y_angles=np.zeros((16,64))

for i in range(len(t_arrays)):
    t = t_arrays[i]

    dt=t[1]-t[0];
    fmax=1/dt
    y = chirped_signal(t)

    y[0]=0 # the sample corresponding to -tmax should be set zeroo
    y=fftshift(y) # make y start with y(t=0)
    Y=fftshift(fft(y))/float(64) #normalisation
    w=linspace(-pi*fmax,pi*fmax,64+1)[:-1]

    Y_mags[i] =  abs(Y)
    Y_angles[i] =  angle(Y)
```

[110]:
```
t=np.linspace(-np.pi,np.pi,1025);t=t[:-1]
fmax = 1/(t[1]-t[0])
t=t[::64]
w=np.linspace(-fmax*np.pi,fmax*np.pi,64+1);w=w[:-1]
t,w=np.meshgrid(t,w)

"""
|Y| Plot
"""
fig = plt.figure(figsize=plt.figaspect(0.2))
ax = fig.add_subplot(1, 2, 1, projection='3d')
surf=ax.plot_surface(w,t,Y_mags.T, rstride=1, cstride=1, cmap=cm.coolwarm,␣
 ↪linewidth=0, antialiased=False)
fig.colorbar(surf, shrink=0.5, aspect=10)
plt.ylabel(r'$\omega\rightarrow$')
plt.xlabel(r'$t\rightarrow$')
ax.set_zlabel(r'$|Y|$')
title("Time-Frequency surface plot")

"""
Phase of Y Plot
"""
ax = fig.add_subplot(1, 2, 2, projection='3d')
```

```
surf=ax.plot_surface(w,t,Y_angles.T,rstride=1, cstride=1, cmap=cm.
 ↪coolwarm,linewidth=0, antialiased=False)
fig.colorbar(surf, shrink=0.5, aspect=10)
plt.ylabel(r'$\omega\rightarrow$')
plt.xlabel(r'$t\rightarrow$')
ax.set_zlabel("Phase of Y")
plt.show()
```