# Fourier Transform

May 18, 2022

[1]:
```
import matplotlib.pyplot as plt
import scipy.integrate as integrate
from matplotlib import style
import numpy as np
```

[2]:
```
style.use('ggplot')
t = np.arange(-15, 15, 0.01)
```

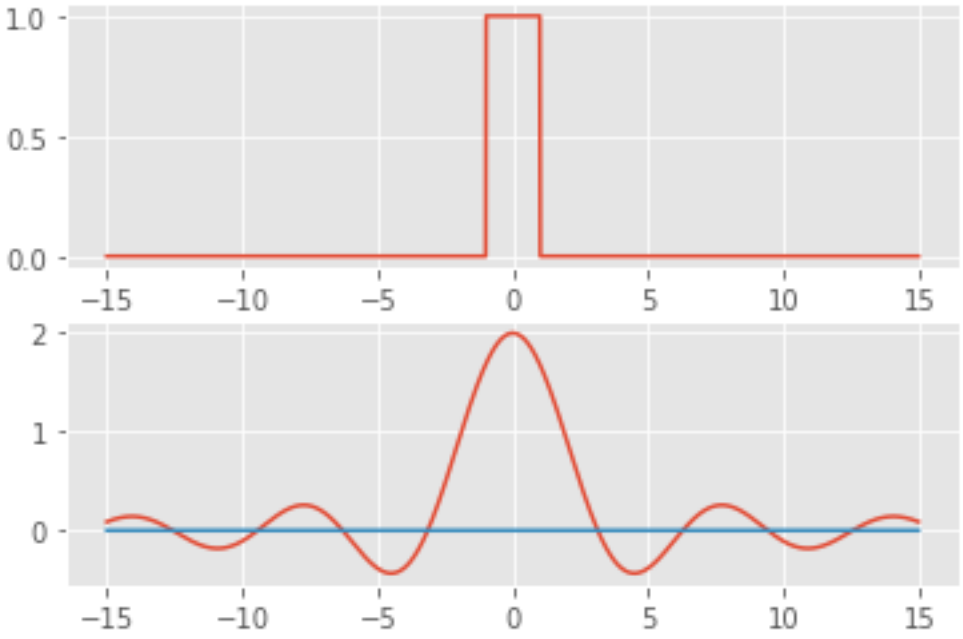
[3]:
```
def fourier_transform(t, x, w):
    return (x(t) * np.exp(-1j * w * t)).real
```

[4]:
```
def signal_ft (x):
    return [integrate.quad(fourier_transform, -15, 15, args=(x, w)) for w in t]
```

[5]:
```
def plot_s(x):
    figure, axis = plt.subplots(2)
    axis[0].plot(t, x(t))
    axis[1].plot(t, signal_ft(x))
    plt.show()
```

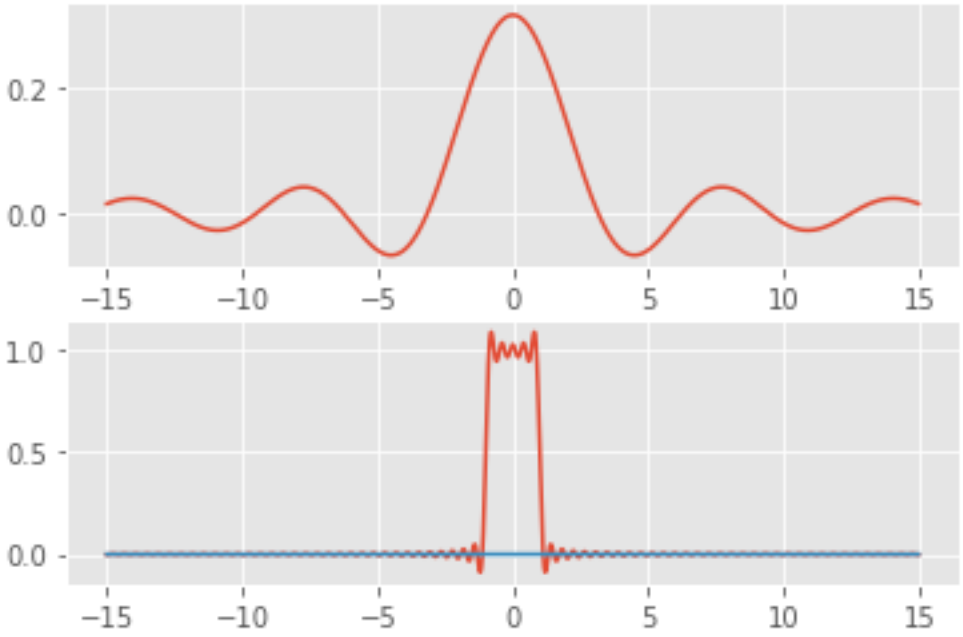
[6]:
```
def A(t):
    return - np.heaviside(t-1, 1) - np.heaviside(-t - 1, 1) + 1
```

[7]:
```
plot_s(A)
```




[8]:
```
def B(t):
    return (1 / (np.pi)) * np.sinc(t / np.pi)
```

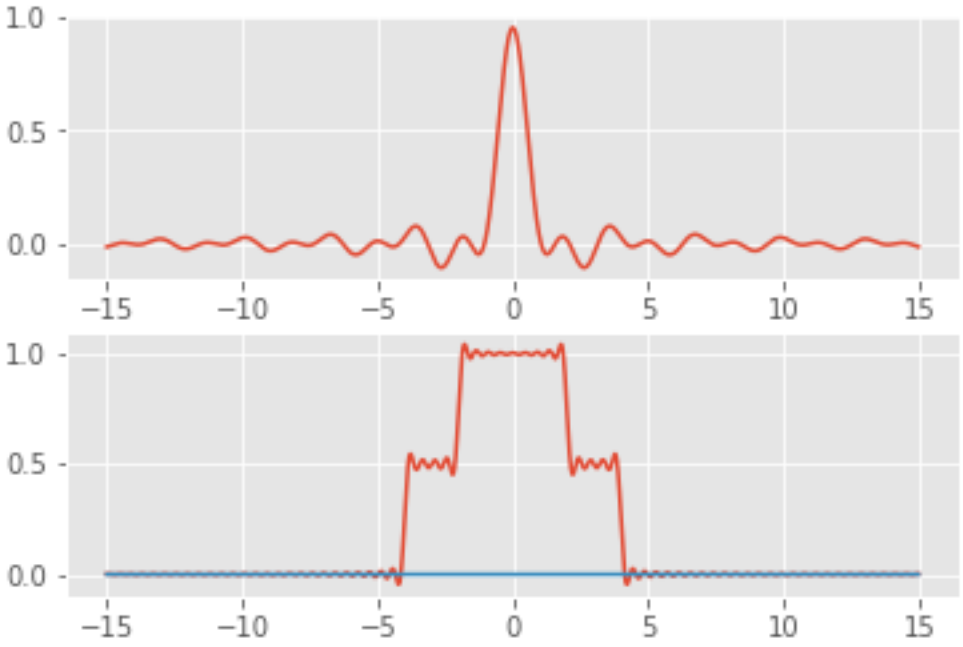
[9]:
```
plot_s(B)
```



[12]:
```
def C(t):
    return np.cos(t) * ((3 / (np.pi)) * np.sinc((3*t)/np.pi))
```

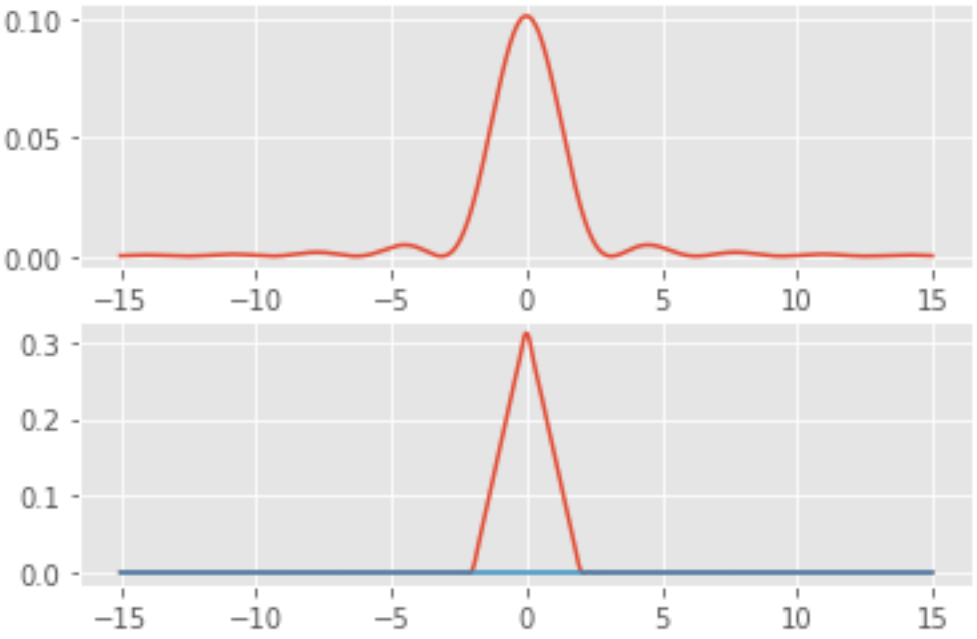
[13]:
```
plot_s(C)
```

```
/var/folders/_4/sv16xs3n1w32wvsk4tn5k5k00000gn/T/ipykernel_50873/3152425190.py:2
: IntegrationWarning: The maximum number of subdivisions (50) has been achieved.
  If increasing the limit yields no improvement it is advised to analyze
  the integrand in order to determine the difficulties.  If the position of a
  local difficulty can be determined (singularity, discontinuity) one will
  probably gain from splitting up the interval and calling the integrator
  on the subranges.  Perhaps a special-purpose integrator should be used.
  return [integrate.quad(fourier_transform, -15, 15, args=(x, w)) for w in t]
```




[14]:
```
def D(t):
    return B(t) ** 2
```

[15]:
```
plot_s(D)
```


0.10
0.05
0.00
−15
−10
−5
0
5
10
15
0.3
0.2
0.1
0.0
−15
−10
−5
0
5
10
15