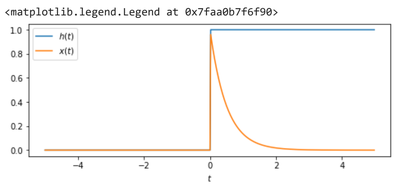
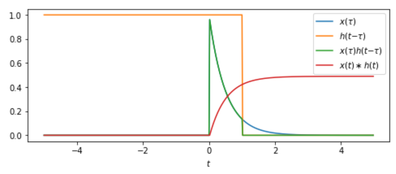
```
from scipy import integrate
import numpy as np
import matplotlib.pyplot as plt
h = lambda t: (t > 0)*1.0
x = lambda t: (t > 0) * np.exp(-2*t)
Fs = 50
T = 5
t = np.arange(-T, T, 1/Fs)
plt.figure(figsize=(8,3))
plt.plot(t, h(t), label='$h(t)$')
plt.plot(t, x(t), label='$x(t)$')
plt.xlabel(r'$t$')
plt.legend()

t_ = 1
flipped = lambda tau: h(t_ - tau)
product = lambda tau: x(tau)*h(t_ - tau)
plt.figure(figsize=(8,3))
plt.plot(t, x(t), label=r'$x(\tau)$')
plt.plot(t, flipped(t), label=r'$h(t - \tau)$')
plt.plot(t, product(t), label=r'$x(\tau)h(t -\tau)$')

y = np.zeros(len(t))
for n, t_ in enumerate(t):
  product = lambda tau: x(tau) * h(t_ - tau)
  y[n] = integrate.simps(product(t), t)

plt.plot(t, y, label=r'$x(t)\ast h(t)$')
plt.xlabel(r'$t$')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7faa0b7f6f90>
```

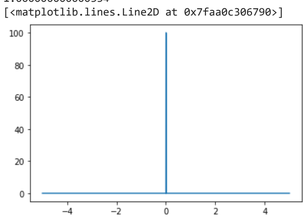




Double-click (or enter) to edit

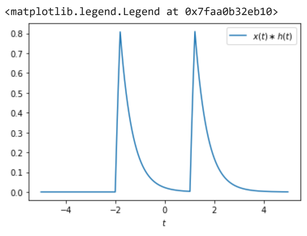
```
fs = 1000
delta = lambda t: np.array([fs/10 if 0 < t_ and t_ < 1/(fs/10) else 0.0 for t_ in t])
t=np.arange(-T,T,1/fs)
y=integrate.simps(delta(t),t)
print(y)
plt.plot(t,delta(t))
```

```
1.000000000000334
[<matplotlib.lines.Line2D at 0x7faa0c306790>]
```



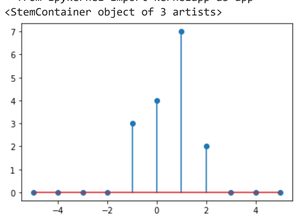
```
h = lambda t: (delta(t+2) + delta(t-1))
x = lambda t: (t > 0) * np.exp(-2*t)
fs = 50
T = 5
t = np.arange(-T, T, 1/Fs)

t_ = 1
flipped = lambda tau: h(t_ - tau)
product = lambda tau: x(tau)*h(t_ - tau)


y = np.zeros(len(t))
for n, t_ in enumerate(t):
  product = lambda tau: x(tau) * h(t_ - tau)
  y[n] = integrate.simps(product(t), t)

plt.plot(t, y, label=r'$x(t)\ast h(t)$')
plt.xlabel(r'$t$')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7faa0b32eb10>
```




```
x = np.array([0, 1, 1, 2, 0])
h = np.array([0, 0, 0, 3, 1, 0, 0])
hr = np.flip(h)
xo = 2
ho = 4
y = np.zeros(len(x) + len(h) - 1)
for n in range(len(y)):
  xkmin = max(0, n - len(h) + 1)
  xkmax = min(len(x), n + 1)
  hkmin = max(0, len(h) - n -1)
  hkmax = min(len(h), len(x) + len(h) -n -1)
  y[n] = np.sum(x[xkmin:xkmax]*hr[hkmin:hkmax])
  print("y[{0}] = x[{1}:{2}]*h[{3}:{4}] = {5}".format(n, xkmin, xkmax, hkmin, hkmax, y[n]))

N=np.arange(-5,6,1)
plt.stem(N, y)
```

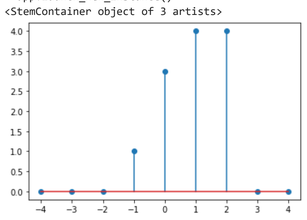
```
y[0] = x[0:1]*h[6:7] = 0.0
y[1] = x[0:2]*h[5:7] = 0.0
y[2] = x[0:3]*h[4:7] = 0.0
y[3] = x[0:4]*h[3:7] = 0.0
y[4] = x[0:5]*h[2:7] = 3.0
y[5] = x[0:5]*h[1:6] = 4.0
y[6] = x[0:5]*h[0:5] = 7.0
y[7] = x[1:5]*h[0:4] = 2.0
y[8] = x[2:5]*h[0:3] = 0.0
y[9] = x[3:5]*h[0:2] = 0.0
y[10] = x[4:5]*h[0:1] = 0.0
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:15: UserWarning: In Matplotlib 3.3 individual lines on a stem plot will be added as a LineCollection instead of individual lines. This significantly improves the performance of a stem plot. To remove this warning and switc
  from ipykernel import kernelapp as app
<StemContainer object of 3 artists>
```



Double-click (or enter) to edit

```
x = np.array([0, 0, 0, 1, 1, 2, 0, 0, 0])
h = np.array([0, 0, 0, 0, 1, 2, 0, 0, 0])
hr = np.flip(h)
xo = 2
ho = 4
y = np.zeros(len(x) + len(h) - 1)
for n in range(len(y)):
    xkmin = max(0, n - len(h) + 1)
    xkmax = min(len(x), n + 1)
    hkmin = max(0, len(h) - n -1)
    hkmax = min(len(h), len(x) + len(h) -n -1)
    y[n] = np.sum(x[xkmin:xkmax]*hr[hkmin:hkmax])
    print("y[{0}] = x[{1}:{2}]*h[{3}:{4}] = {5}".format(n, xkmin, xkmax, hkmin, hkmax, y[n]))

N=np.arange(-4,5,1)
plt.stem(N,y[4:13])
```

```
y[0] = x[0:1]*h[8:9] = 0.0
y[1] = x[0:2]*h[7:9] = 0.0
y[2] = x[0:3]*h[6:9] = 0.0
y[3] = x[0:4]*h[5:9] = 0.0
y[4] = x[0:5]*h[4:9] = 0.0
y[5] = x[0:6]*h[3:9] = 0.0
y[6] = x[0:7]*h[2:9] = 0.0
y[7] = x[0:8]*h[1:9] = 1.0
y[8] = x[0:9]*h[0:9] = 3.0
y[9] = x[1:9]*h[0:8] = 4.0
y[10] = x[2:9]*h[0:7] = 4.0
y[11] = x[3:9]*h[0:6] = 0.0
y[12] = x[4:9]*h[0:5] = 0.0
y[13] = x[5:9]*h[0:4] = 0.0
y[14] = x[6:9]*h[0:3] = 0.0
y[15] = x[7:9]*h[0:2] = 0.0
y[16] = x[8:9]*h[0:1] = 0.0
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:16: UserWarning: In Matplotlib 3.3 individual lines on a stem plot will be added as a LineCollection instead of individual lines. This significantly improves the performance of a stem plot. To remove this warning and switc
  app.launch_new_instance()
<StemContainer object of 3 artists>
```



```
from scipy import signal
import numpy as np
import matplotlib.pyplot as plt
data, samplerate = sf.read('power_of_love.wav')
nyquist = samplerate//2
fc = 2000/nyquist
n = 121
b = signal.firwin(n, fc, pass_zero=True)
w, h = signal.freqz(b)
import matplotlib.pyplot as plt
fig, ax1 = plt.subplots()
```

```
ax1.set_title('Digital filter frequency response')
ax1.plot(w, 20 * np.log10(abs(h)), 'b')
ax1.set_ylabel('Amplitude [dB]', color='b')
ax1.set_xlabel('Frequency [rad/sample]')
ax2 = ax1.twinx()
angles = np.unwrap(np.angle(h))

ax2.plot(w, angles, 'g')
ax2.set_ylabel('Angle (radians)', color='g')
ax2.grid()
ax2.axis('tight')
plt.show()



sf.write('.wav', np.vstack((ch1, ch2)).T + data, samplerate)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-18-ecd39b454b95> in <module>()
      2 import numpy as np
      3 import matplotlib.pyplot as plt
----> 4 data, samplerate = sf.read('power_of_love.wav')
      5 nyquist = samplerate//2
      6 fc = 2000/nyquist

NameError: name 'sf' is not defined

SEARCH STACK OVERFLOW
```

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
x = mpimg.imread('allenkeys.png')
fig, ax = plt.subplots(1,2)
ax[0].imshow(x, cmap='gray')
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-1-eaa4e3da309c> in <module>()
      1 import matplotlib.pyplot as plt
      2 import matplotlib.image as mpimg
----> 3 x = mpimg.imread('allenkeys.png')
      4 fig, ax = plt.subplots(1,2)
      5 ax[0].imshow(x, cmap='gray')

3 frames
/usr/local/lib/python3.7/dist-packages/matplotlib/cbook/__init__.py in to_filehandle(fname, flag, return_opened, encoding)
    401             fh = bz2.BZ2File(fname, flag)
    402         else:
--> 403             fh = open(fname, flag, encoding=encoding)
    404         opened = True
    405     elif hasattr(fname, 'seek'):

FileNotFoundError: [Errno 2] No such file or directory: 'allenkeys.png'

SEARCH STACK OVERFLOW
```

✓ 0s completed at 5:23 PM