Modelação Numérica 2017

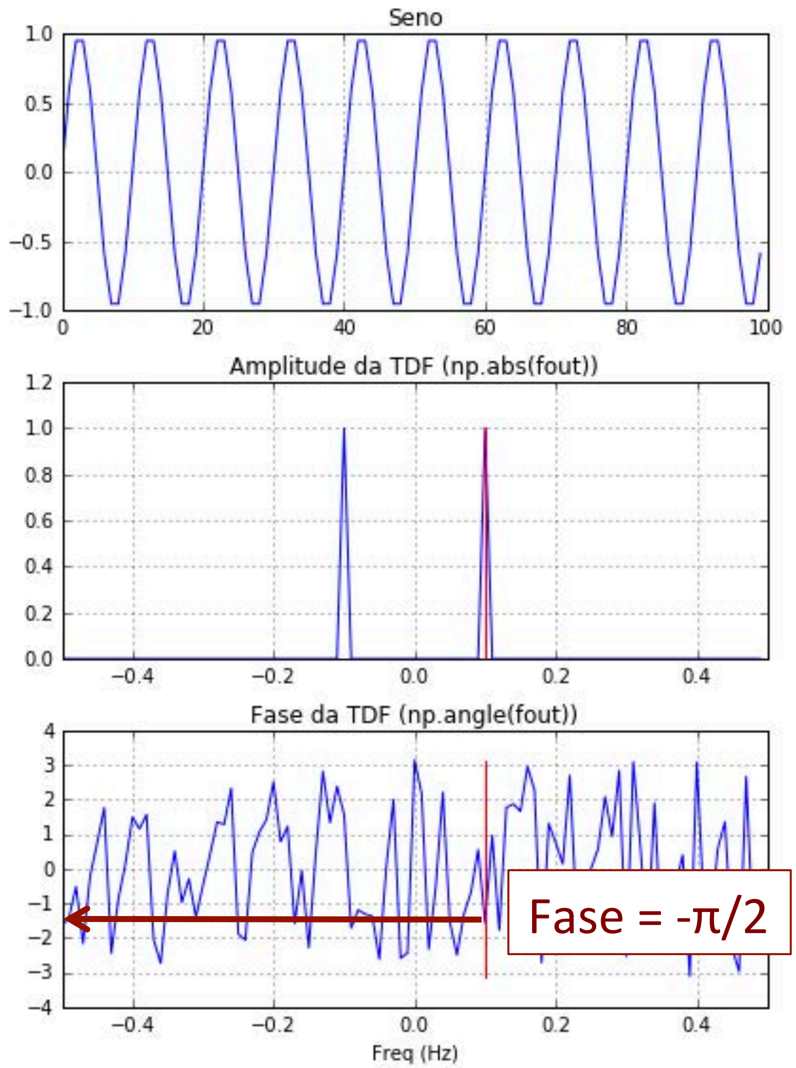
Aula 7, 8/Mar

- Filtros de média móvel
- Filtros de Fourier

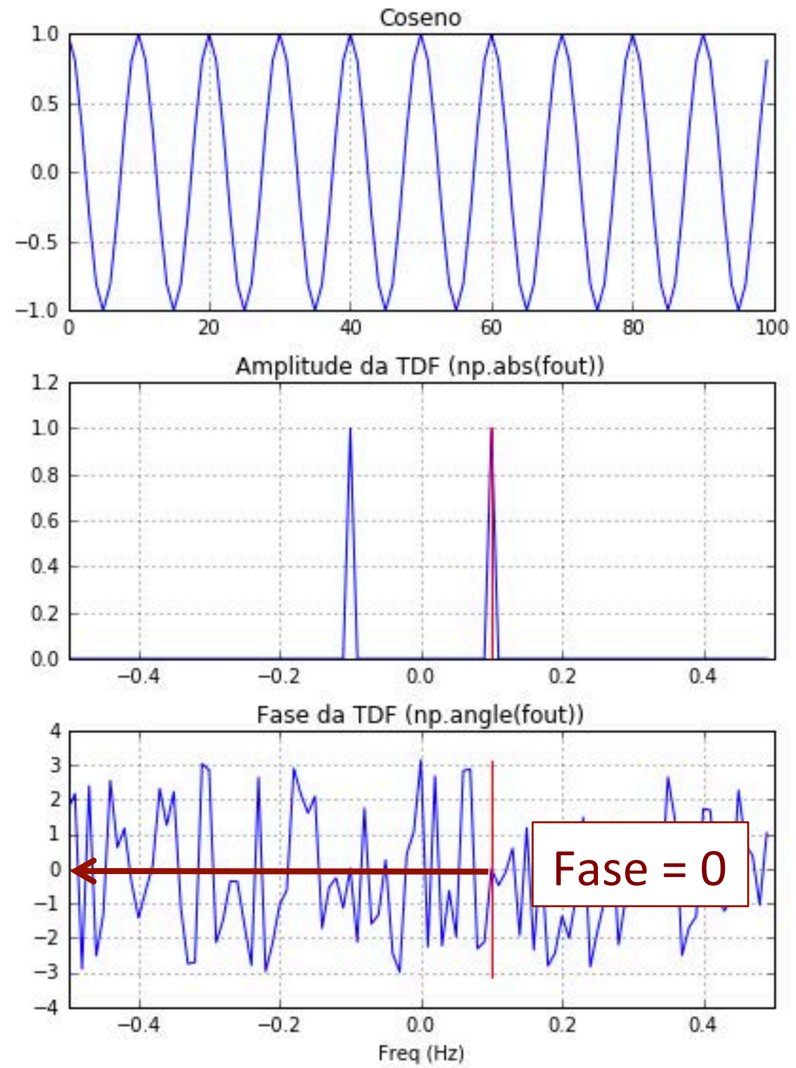
http://modnum.ucs.ciencias.ulisboa.pt

Espectro de amplitude e de fase

sin(x)



cos(x)



Aula passada

Propriedades da TDF

- Linearidade:

$$G = \mathcal{F}(g);\ H = \mathcal{F}(h)$$

$$\downarrow$$

$$\mathcal{F}(ag+bh) = aG + bH$$

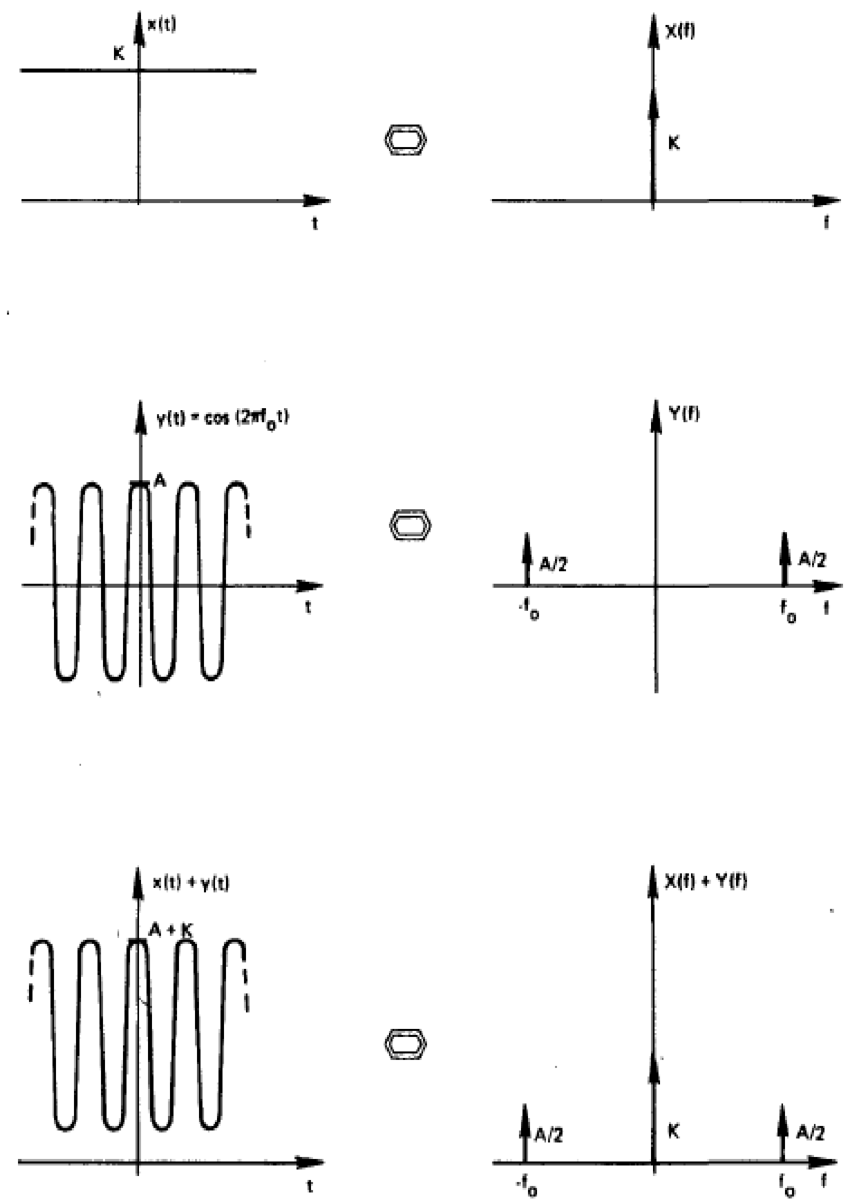


Figure 3-1. The linearity property.

Aula passada

Propriedades da TDF

- Translação:

$$G(f) = \mathcal{F}(g(t))$$

↓

$$\mathcal{F}(g(t-a)) = e^{-ifa}G(f)$$

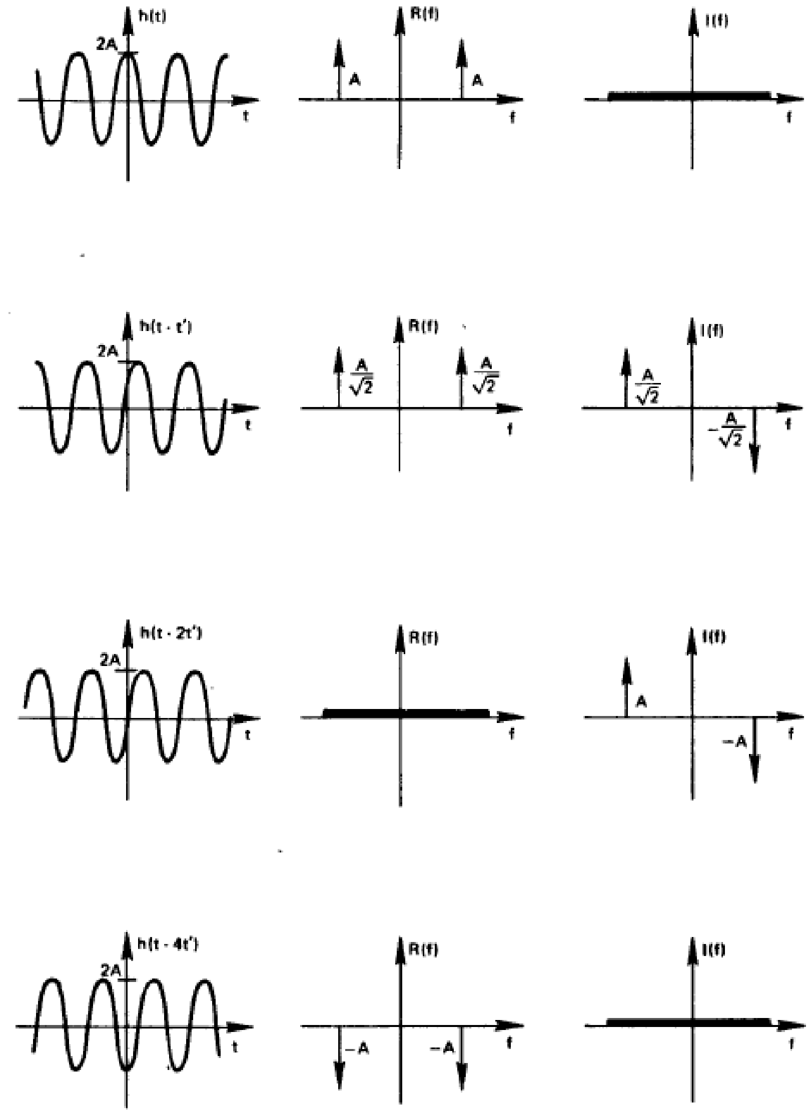


Figure 3-4. Time shifting property.

Aula passada

Propriedades da TDF

- Escalamento:

$$G(f) = \mathcal{F}(g(t))$$

$$\downarrow$$

$$\mathcal{F}(g(at)) = 1/a \; G(f/a)$$

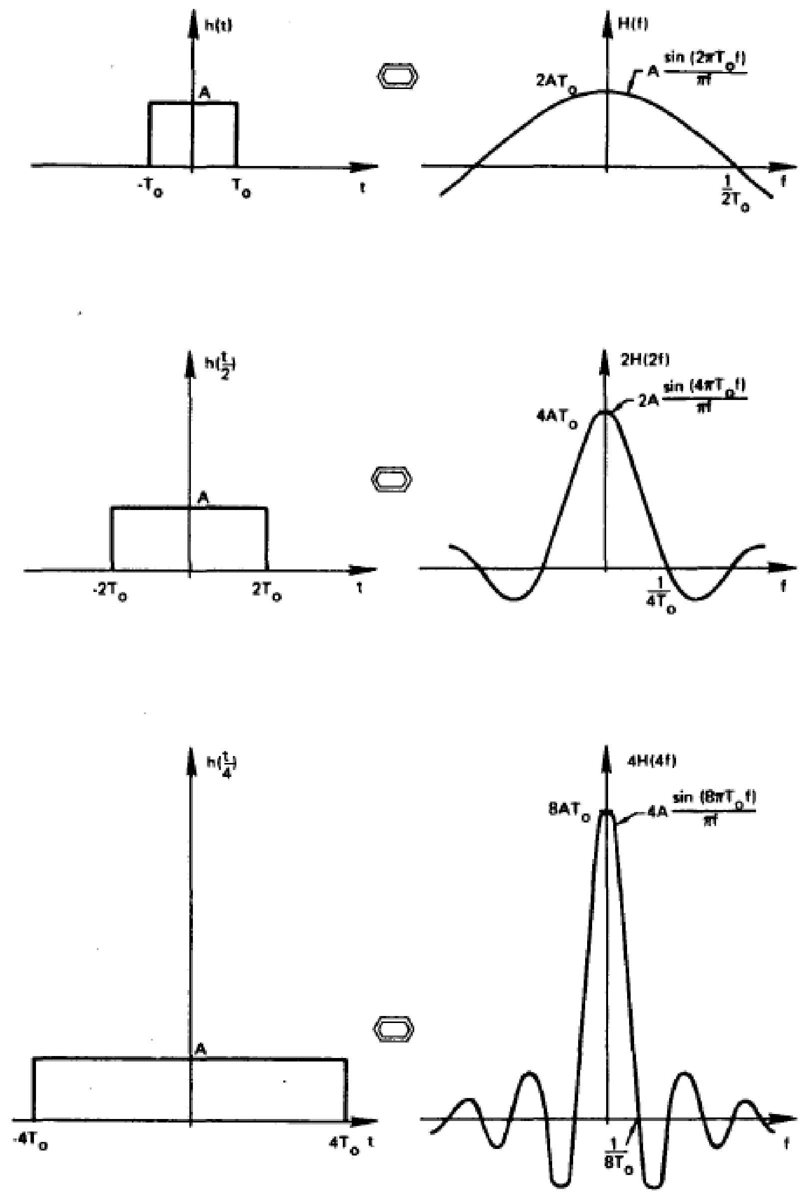
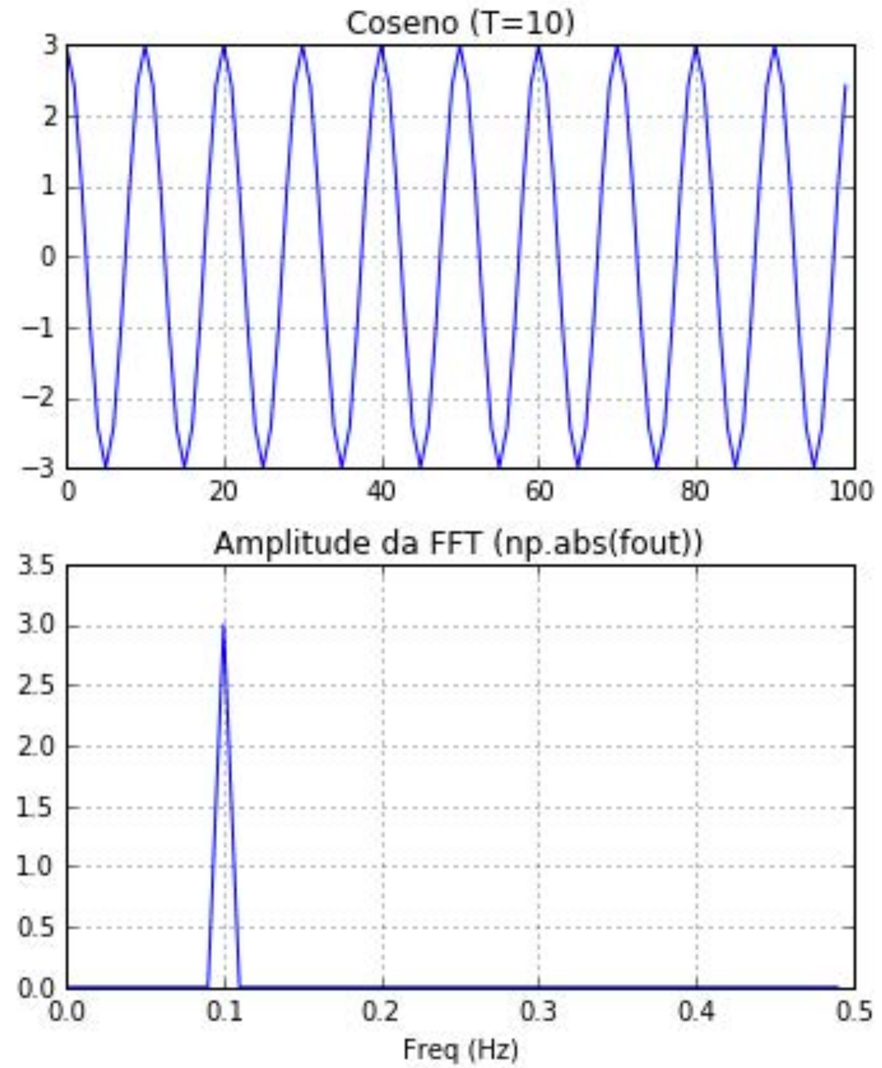


Figure 3-2. Time scaling property.

Aula passada

FFT (iFFT): Fast Fourier Transform

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import pi as pi
import numpy.fft as fft

plt.rcParams['figure.figsize'] = 5, 6

#%%
N=100.; T=10.; dt=1.
t=np.arange(0.,N,dt)
f=3*np.cos(2*pi * t/T)

tf=fft.fft(f)
fout=tf[:N/2]
fNyq=1/(2*dt); df=1/(N*dt);
freq=np.arange(0, fNyq,df)

plt.close();
plt.subplot(2,1,1); plt.plot(t, f)
plt.title('Coseno (T=10)'); plt.grid();

plt.subplot(2,1,2); plt.plot(freq,np.abs(fout)/(N/2))
plt.title('Amplitude da FFT (np.abs(fout))'); plt.grid();
plt.xlabel('Freq (Hz)')

plt.tight_layout()
```



A FFT será eficiente se $N=2^k$.

Aula passada

Convolução e Correlação

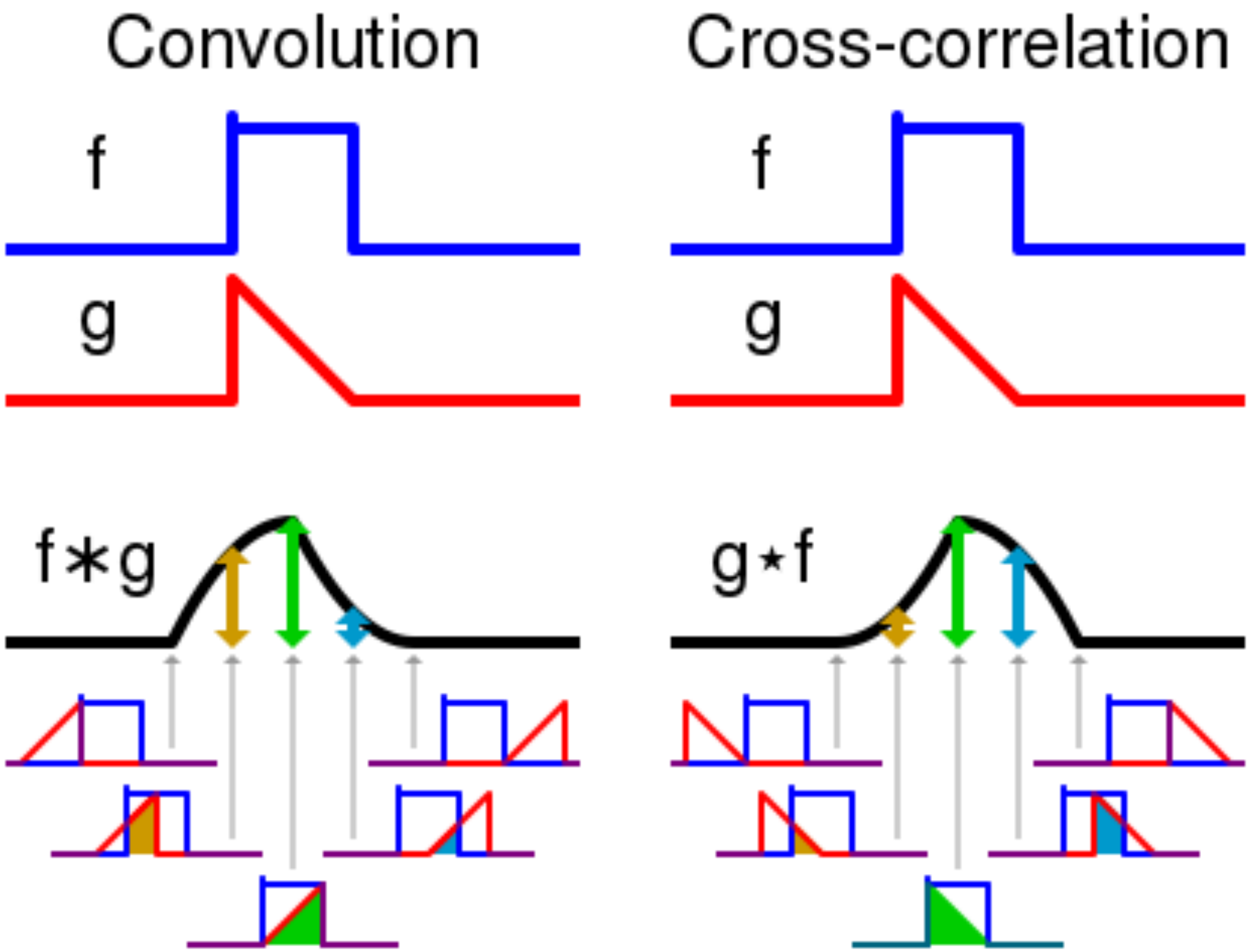
- Convolução:

$$y(k) = x * h = conv(x, h) = \sum_{n=-\infty}^{+\infty} x(n)h(k-n)$$

- Correlação:

$$y(k) = corr(x, h) = \sum_{n=-\infty}^{+\infty} x(n)h(n+k)$$

Convolução e Correlação



Cmglee

Aula passada

Teorema da convolução

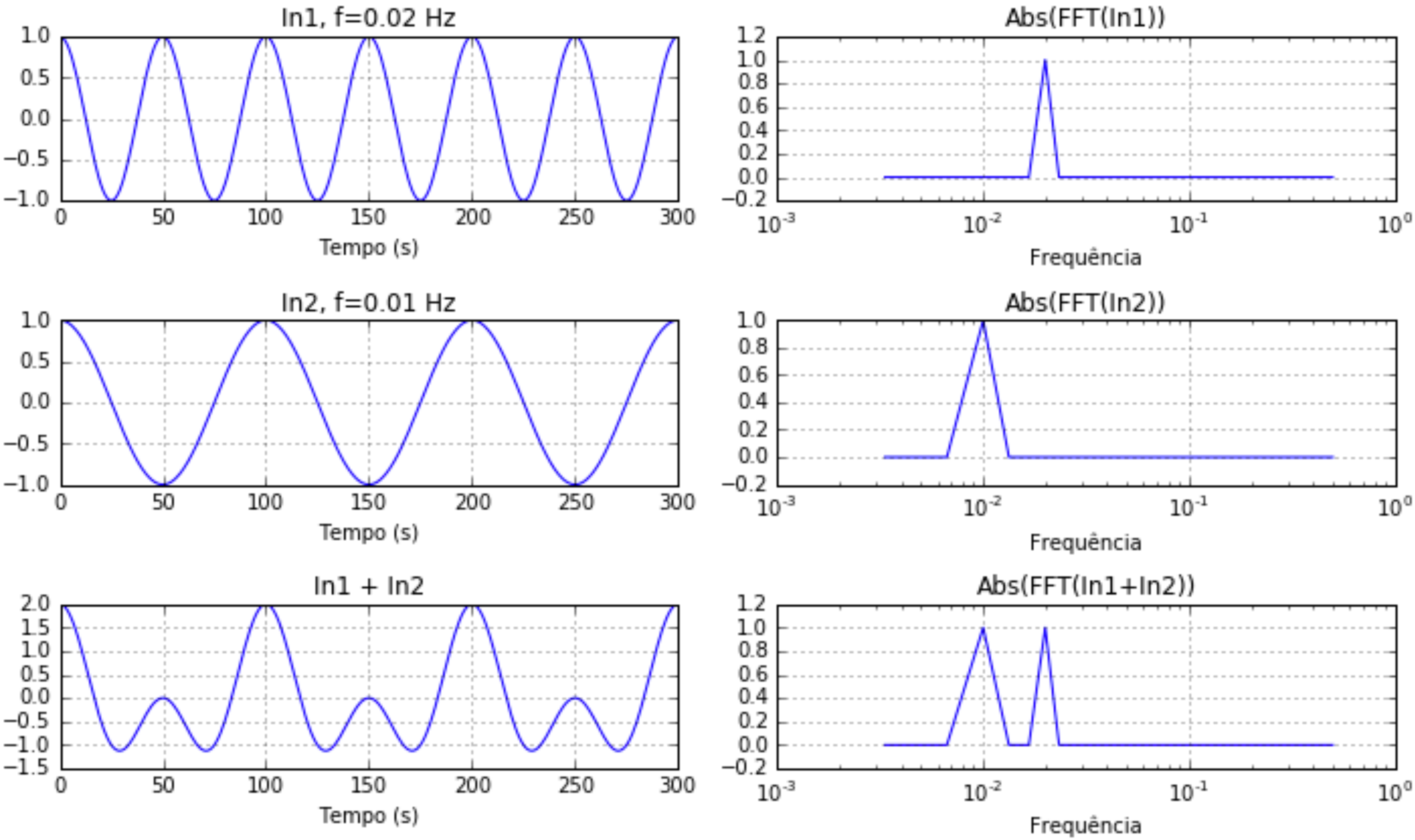
$$\mathcal{F}(xy) = \mathcal{F}(x) * \mathcal{F}(y)$$

$$\mathcal{F}(x*y) = \mathcal{F}(x)\ \mathcal{F}(y)$$

Teorema da correlação

$$\mathcal{F}(corr(x,x)) = \mathcal{F}(x)\ \mathcal{F}(x)^*$$

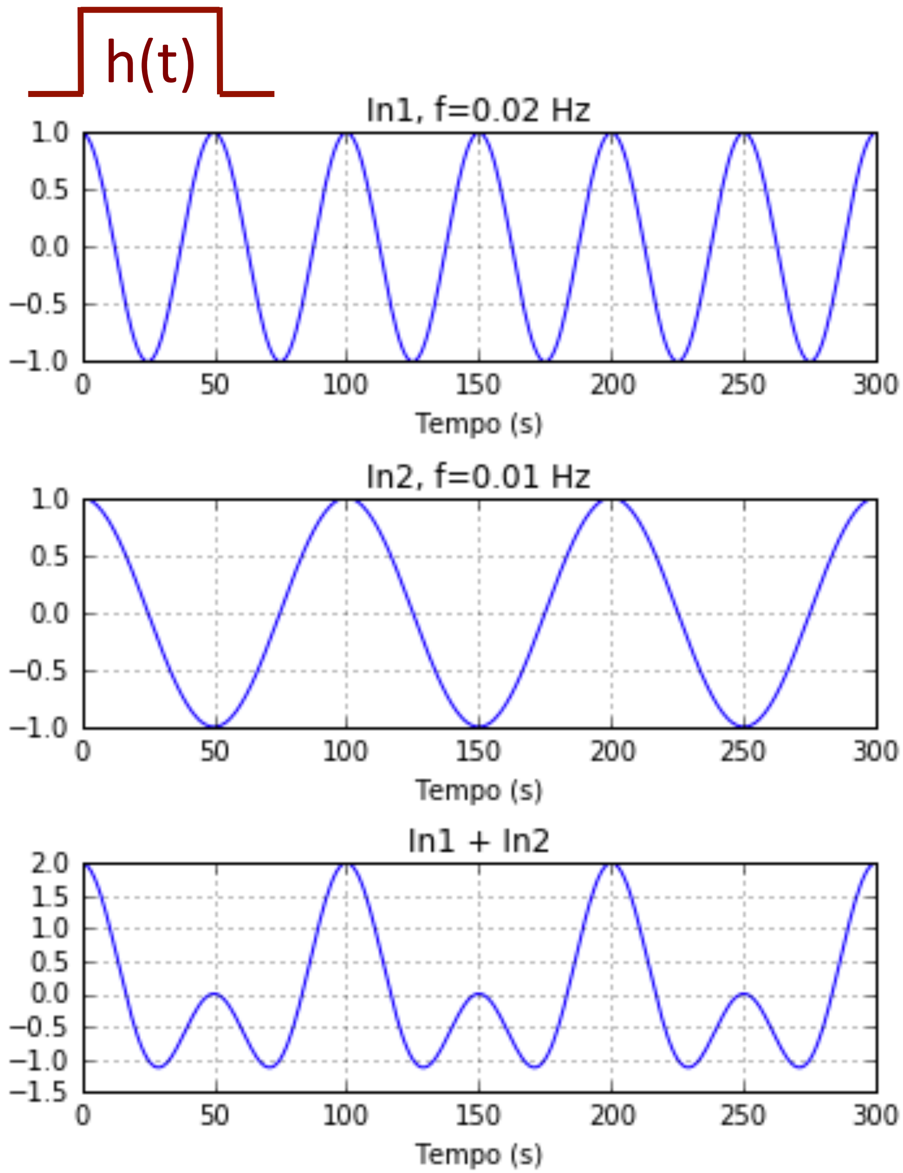
Filtros – Input



```python
import matplotlib.pyplot as plt
import numpy as np
from numpy import pi as pi
import numpy.fft as fft

plt.rcParams['figure.figsize'] = 10, 6

#%%
## Cos 50
N=300; T=50.; dt=1.
t=np.arange(0.,N,dt)
f1=np.cos(2*pi * t/T)

tf1=fft.fft(f1)
fout1=tf1[:N/2]
fNyq=1/(2*dt); df=1/(N*dt);
freq=np.arange(0, fNyq, df)
N1=N

#%%
## Cos 100
T=100.;
f2=np.cos(2*pi * t/T)

tf2=fft.fft(f2)
fout2=tf2[:N/2]

#%%
## Cos 50 + cos 100
fs=f1+f2

tfs=fft.fft(fs)
fouts=tfs[:N/2]
```

```python
#%% Plots
plt.close();
plt.subplot(3,2,1);
plt.plot(t, f1)
plt.title('In1, f=0.02 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,3);
plt.plot(t, f2)
plt.title('In2, f=0.01 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,5);
plt.plot(t, fs)
plt.title('In1 + In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,2);
plt.semilogx(freq,np.abs(fout1)/(N/2))
plt.title('Abs(FFT(In1))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,4);
plt.semilogx(freq,np.abs(fout2)/(N/2))
plt.title('Abs(FFT(In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,6);
plt.semilogx(freq,np.abs(fouts)/(N/2))
plt.title('Abs(FFT(In1+In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.tight_layout()
```

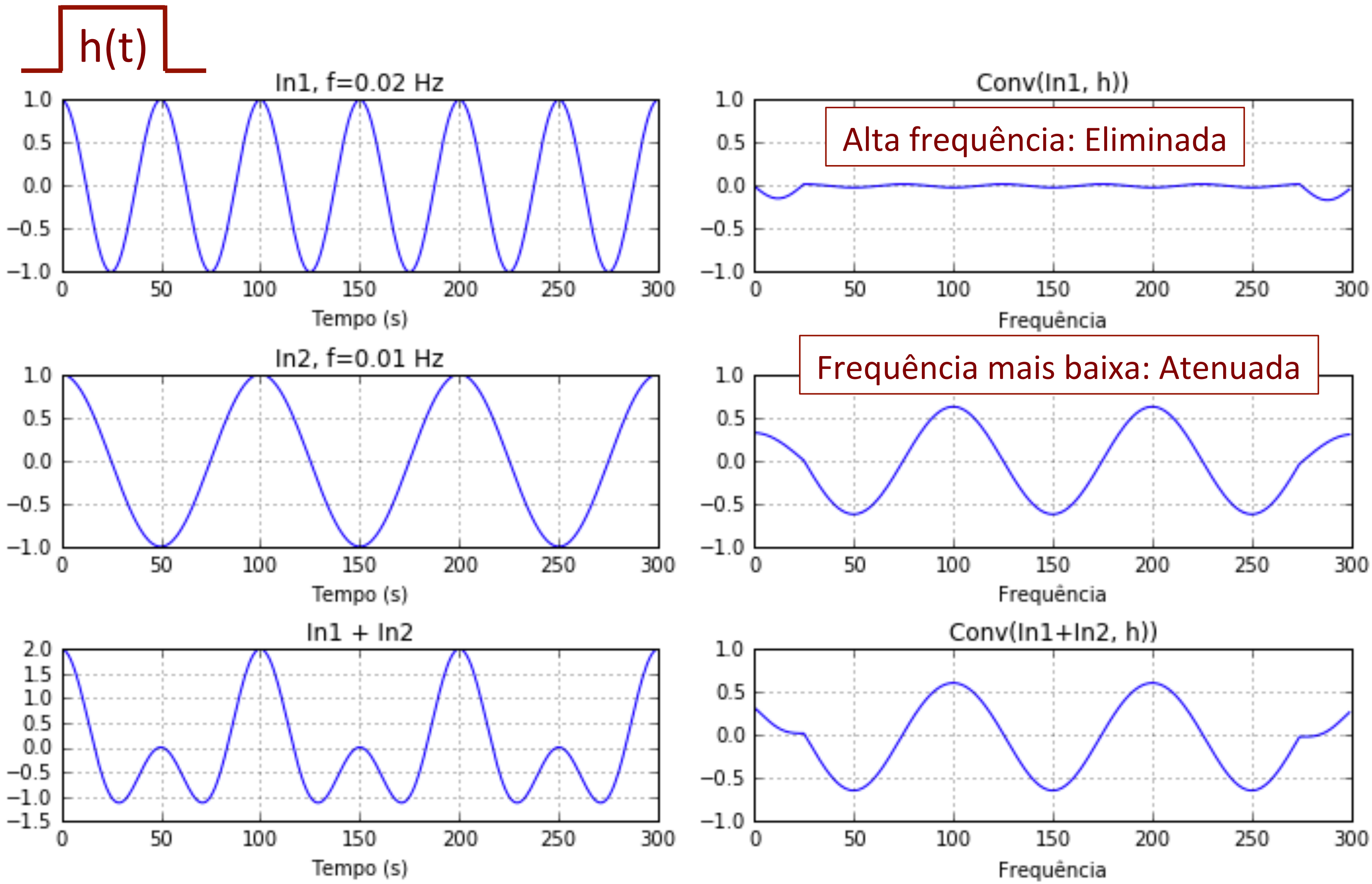
Filtro de média móvel (domínio do tempo)



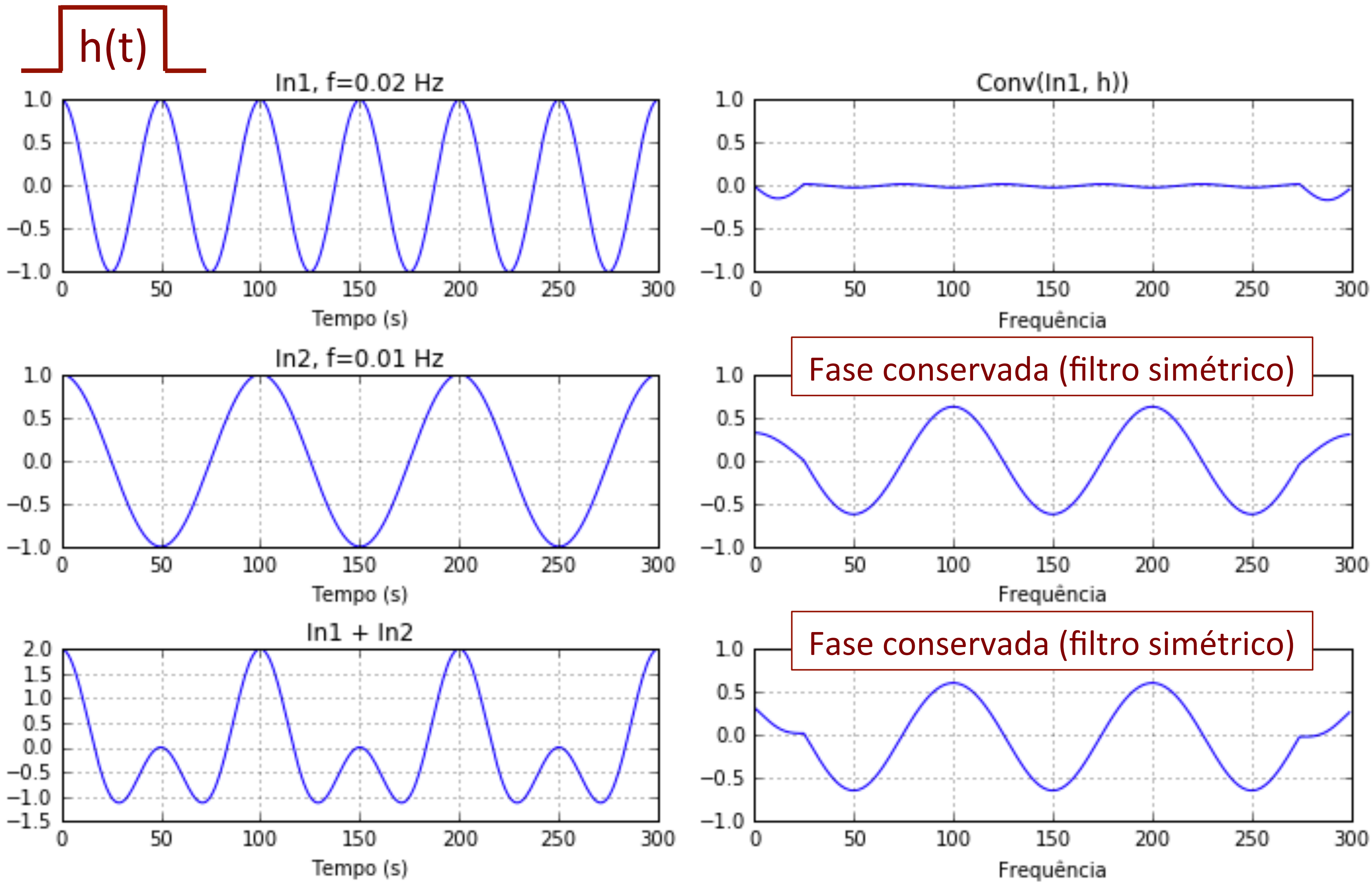

```
#%% Média móvel passa-baixo

NF=51;            # nr de pontos da janela
h=np.ones(NF)/NF;

c1=np.convolve(f1, h, mode='same')
c2=np.convolve(f2, h, mode='same')
cs=np.convolve(fs, h, mode='same')
```

```python
#%% Plots
plt.close();
plt.subplot(3,2,1);
plt.plot(t, f1)
plt.title('In1, f=0.02 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,3);
plt.plot(t, f2)
plt.title('In2, f=0.01 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,5);
plt.plot(t, fs)
plt.title('In1 + In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,2);
plt.plot(t,c1)
plt.title('Conv(In1, h))');
plt.grid(); plt.ylim([-1,1])
plt.xlabel(u'Frequência')

plt.subplot(3,2,4);
plt.plot(t,c2)
plt.title('Conv(In2, h))');
plt.grid();  plt.ylim([-1,1])
plt.xlabel(u'Frequência')

plt.subplot(3,2,6);
plt.plot(t,cs)
plt.title('Conv(In1+In2, h))');
plt.grid();  plt.ylim([-1,1])
plt.xlabel(u'Frequência')

plt.tight_layout()
```

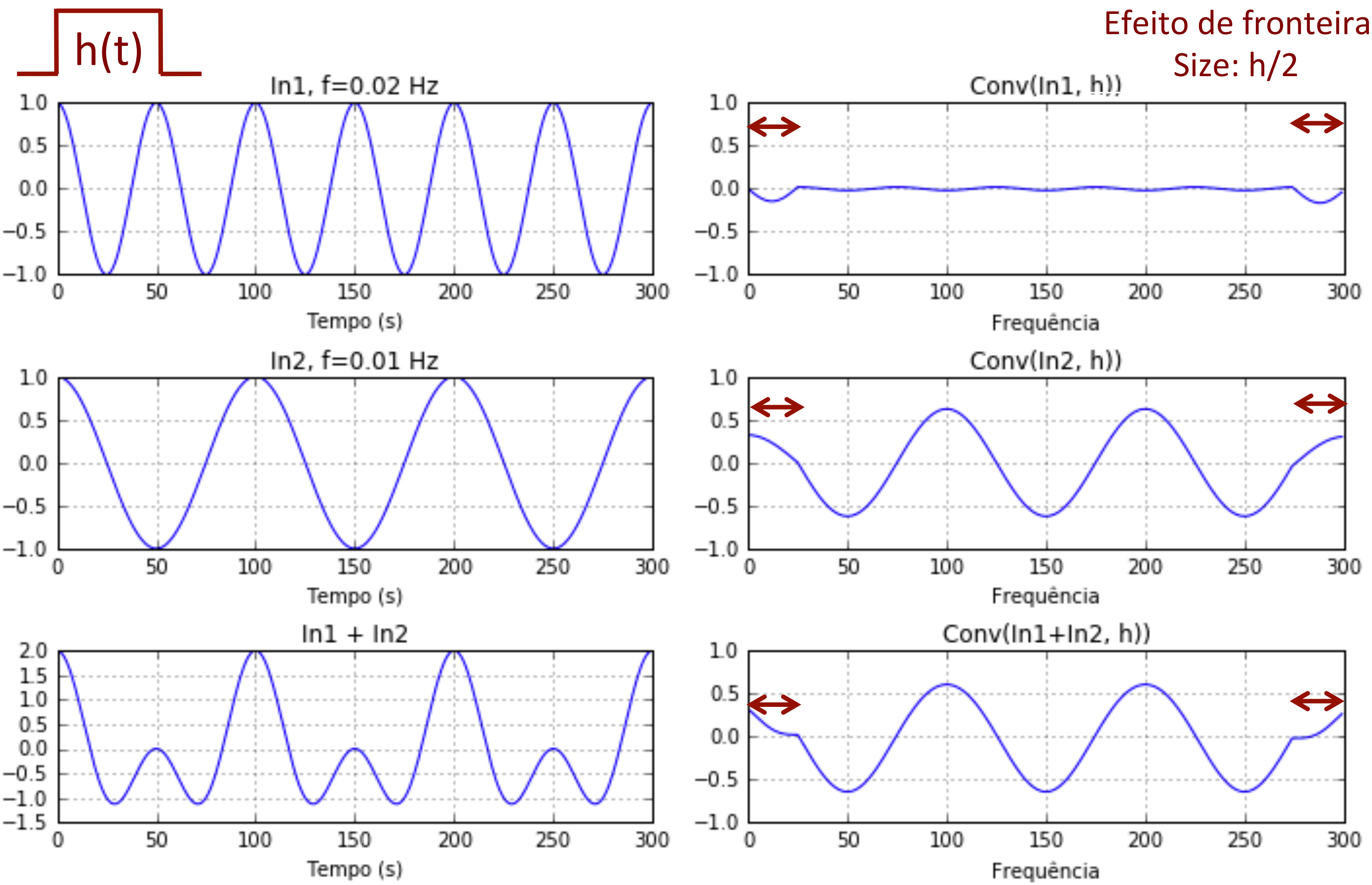
Filtro de média móvel (domínio do tempo)



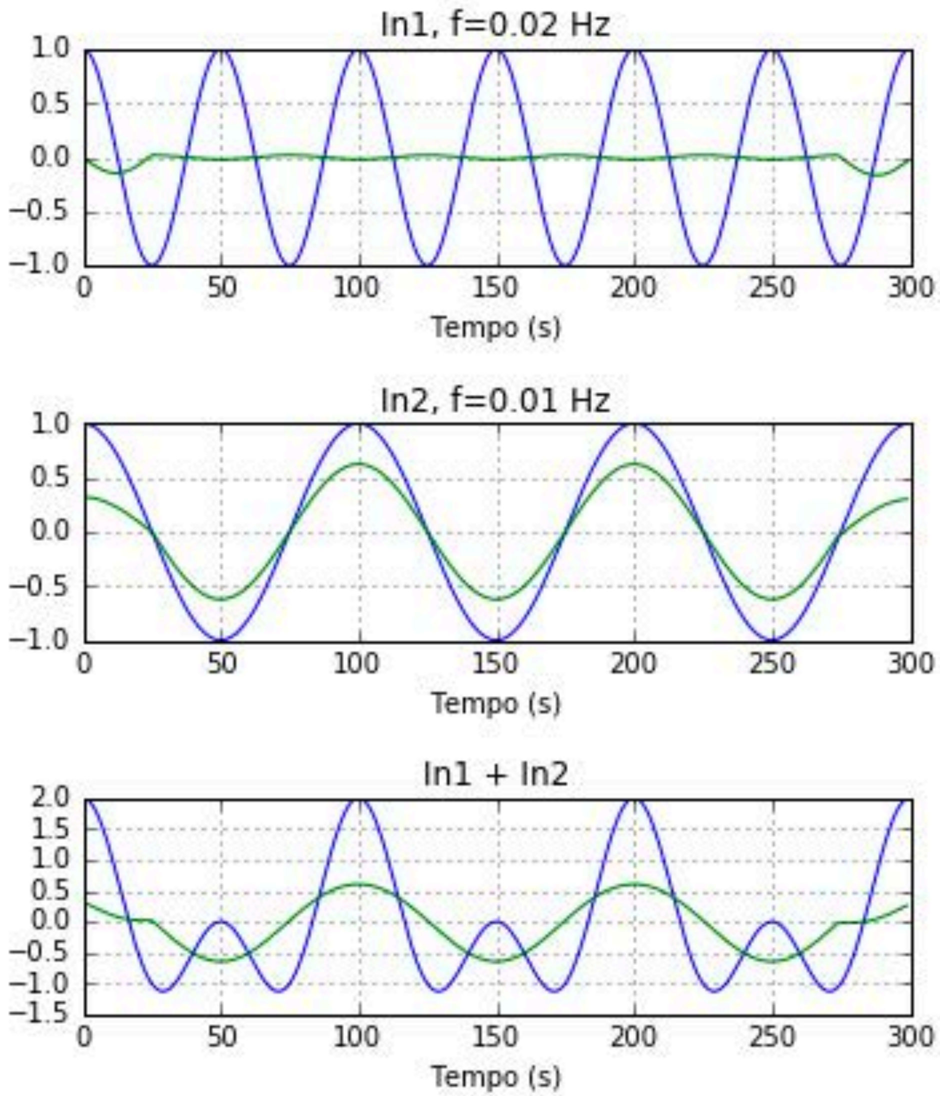
Filtro de média móvel (domínio do tempo)



Filtro de média móvel (domínio do tempo)



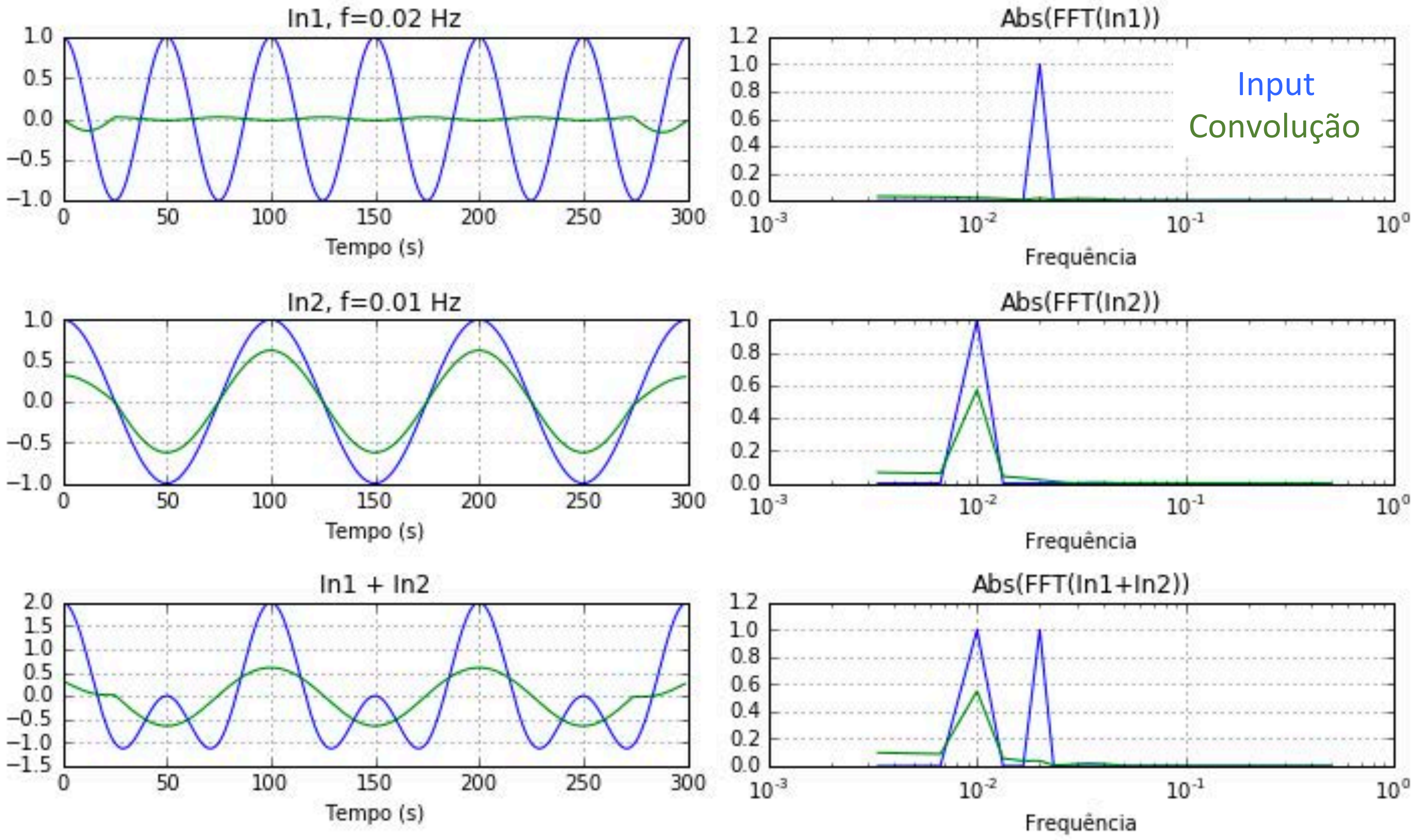
Filtro de média móvel (domínio do tempo)




```
#%% Espectros dos sinais filtrados

tfc1=fft.fft(c1)
foutc1=tfc1[:N/2]
Fc1=np.abs(foutc1)/(N/2)

tfc2=fft.fft(c2)
foutc2=tfc2[:N/2]
Fc2=np.abs(foutc2)/(N/2)

tfcs=fft.fft(cs)
foutcs=tfcs[:N/2]
Fcs=np.abs(foutcs)/(N/2)
```

```python
#%% Plots
plt.close();
plt.subplot(3,2,1);
plt.plot(t, f1, t, c1)
plt.title('In1, f=0.02 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,3);
plt.plot(t, f2, t, c2)
plt.title('In2, f=0.01 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,5);
plt.plot(t, fs, t, cs)
plt.title('In1 + In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,2);
plt.semilogx(freq,np.abs(fout1)/(N/2), freq, Fc1)
plt.title('Abs(FFT(In1))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,4);
plt.semilogx(freq,np.abs(fout2)/(N/2), freq, Fc2)
plt.title('Abs(FFT(In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,6);
plt.semilogx(freq,np.abs(fouts)/(N/2), freq, Fcs)
plt.title('Abs(FFT(In1+In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.tight_layout()
```

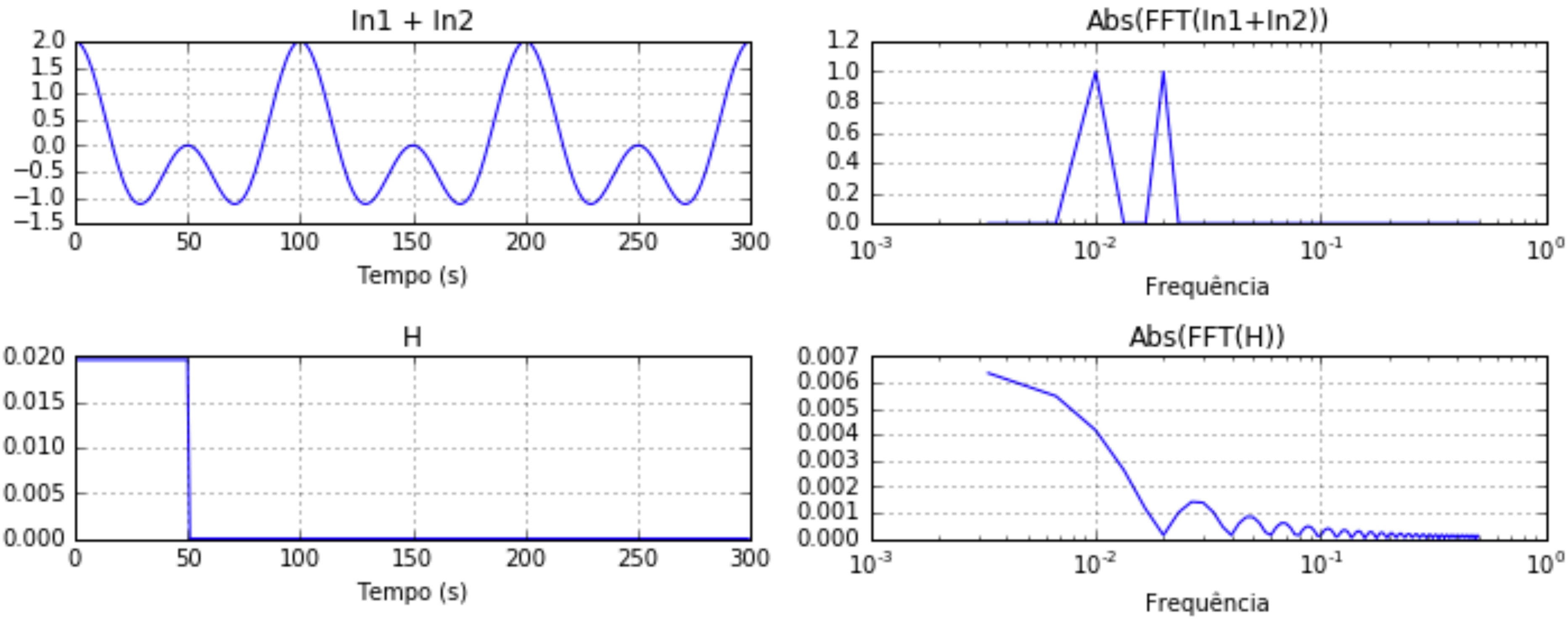
Filtro de média móvel (domínio do tempo)



Filtro de média móvel (domínio espectral)

Domínio do tempo

Domínio espectral




```
hh = np.append(h, np.zeros(len(fs)-len(h)))
tfh=fft.fft(hh)
fouth=tfh[:N/2]
Fh=np.abs(fouth)/(N/2)
```

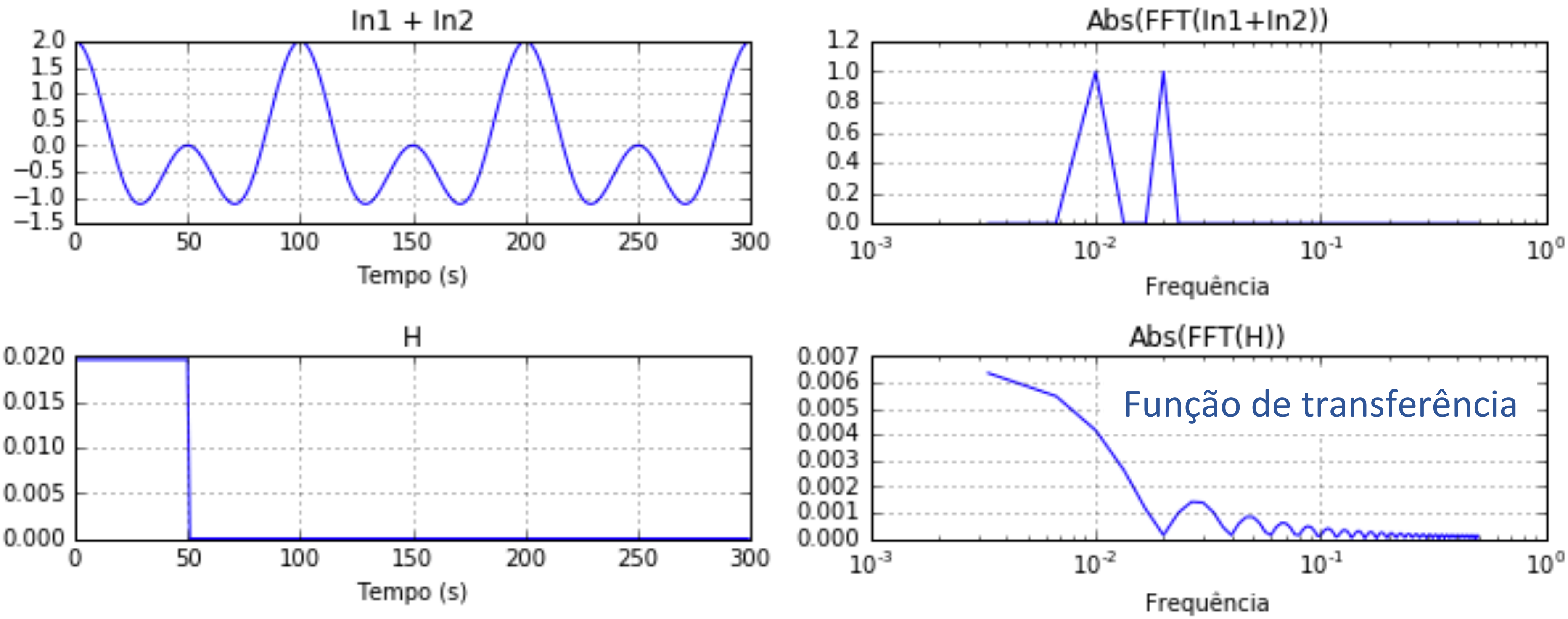
Filtro de média móvel (domínio espectral)

```
#%% Plots
plt.close();
plt.subplot(3,2,1);
plt.plot(t, fs)
plt.title('In1 + In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,3);
plt.plot(t, hh)
plt.title('H'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,2);
plt.semilogx(freq,np.abs(fouts)/(N/2))
plt.title('Abs(FFT(In1+In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,4);
plt.semilogx(freq,np.abs(fouth)/(N/2))
plt.title('Abs(FFT(H))');
plt.grid();
plt.xlabel(u'Frequência')
```

Filtro de média móvel (domínio espectral)

Domínio do tempo

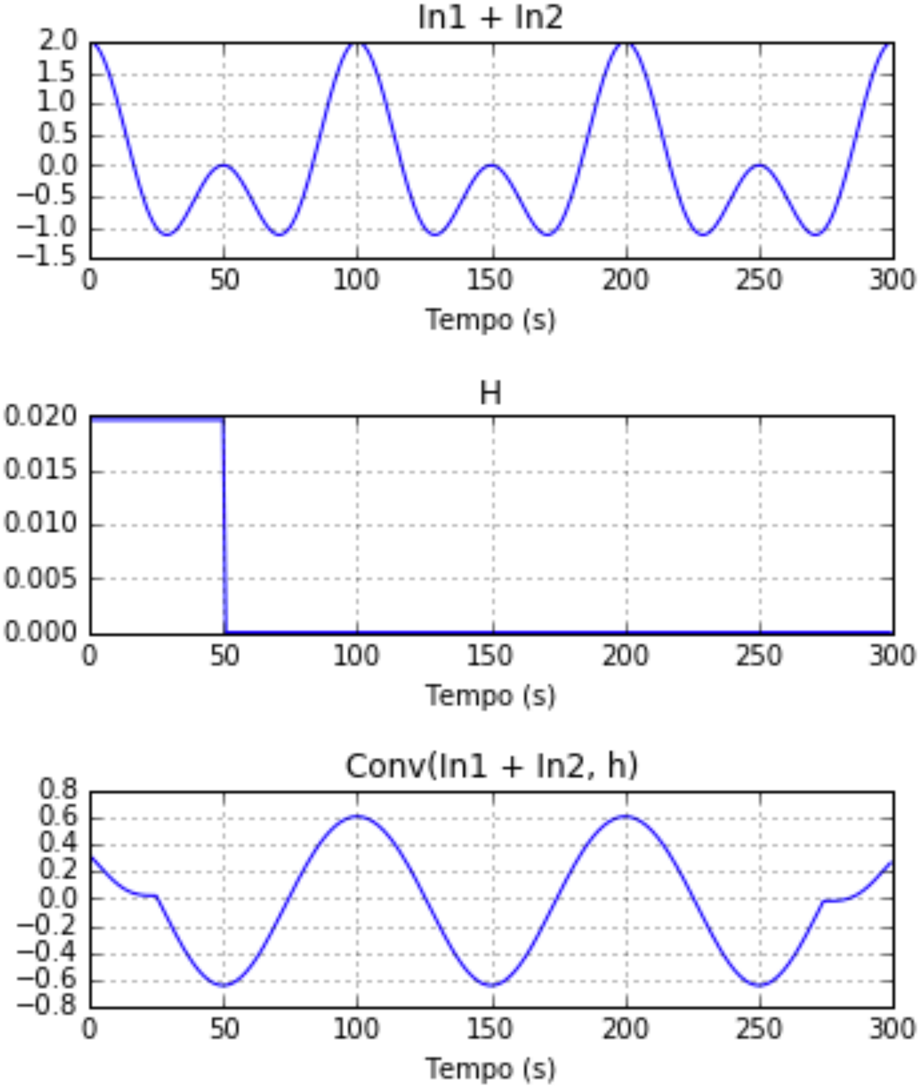
Domínio espectral



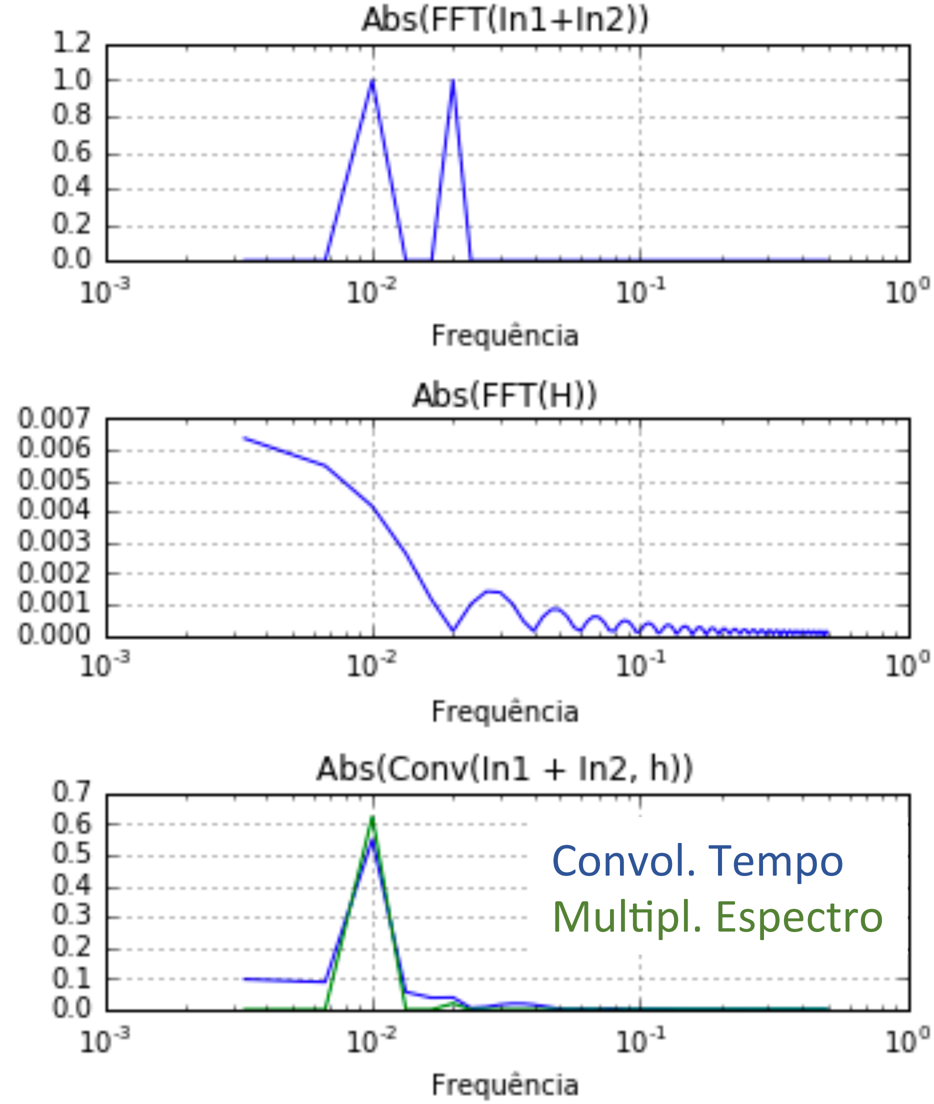
- Altas frequências <u>quase</u> eliminadas.
- Função de transferência com <u>oscilações</u>.
- <u>Atenuação</u> em todo o espectro.

Filtro de média móvel (domínio espectral)

Domínio do tempo: convolução



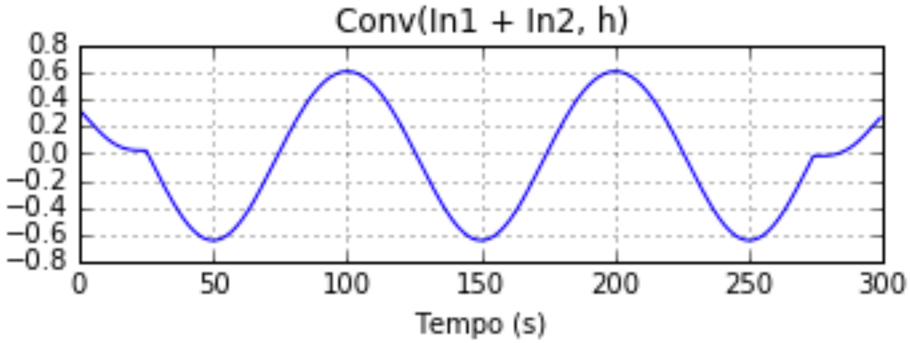
Domínio espectral: multiplicação



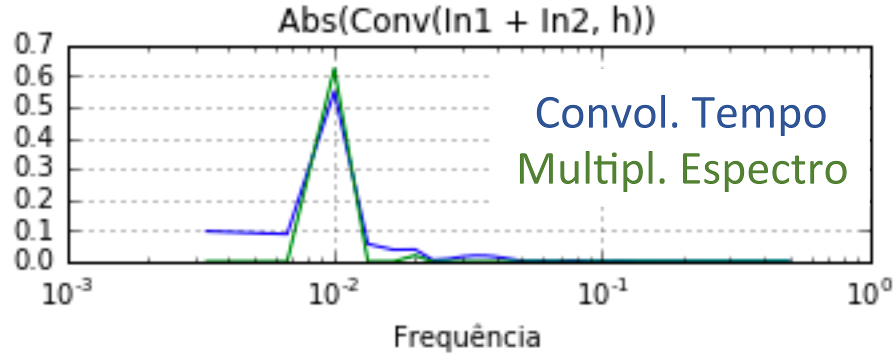
Filtro de média móvel (domínio espectral)

```
plt.subplot(3,2,5);
plt.plot(t, cs)
plt.title('Conv(In1 + In2, h)'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,6);
plt.semilogx(freq, Fcs, freq, np.abs(fouts*fouth)/(N/2))
plt.title('Abs(Conv(In1 + In2, h))');
plt.grid();
plt.xlabel(u'Frequência')
```

Domínio do tempo: convolução



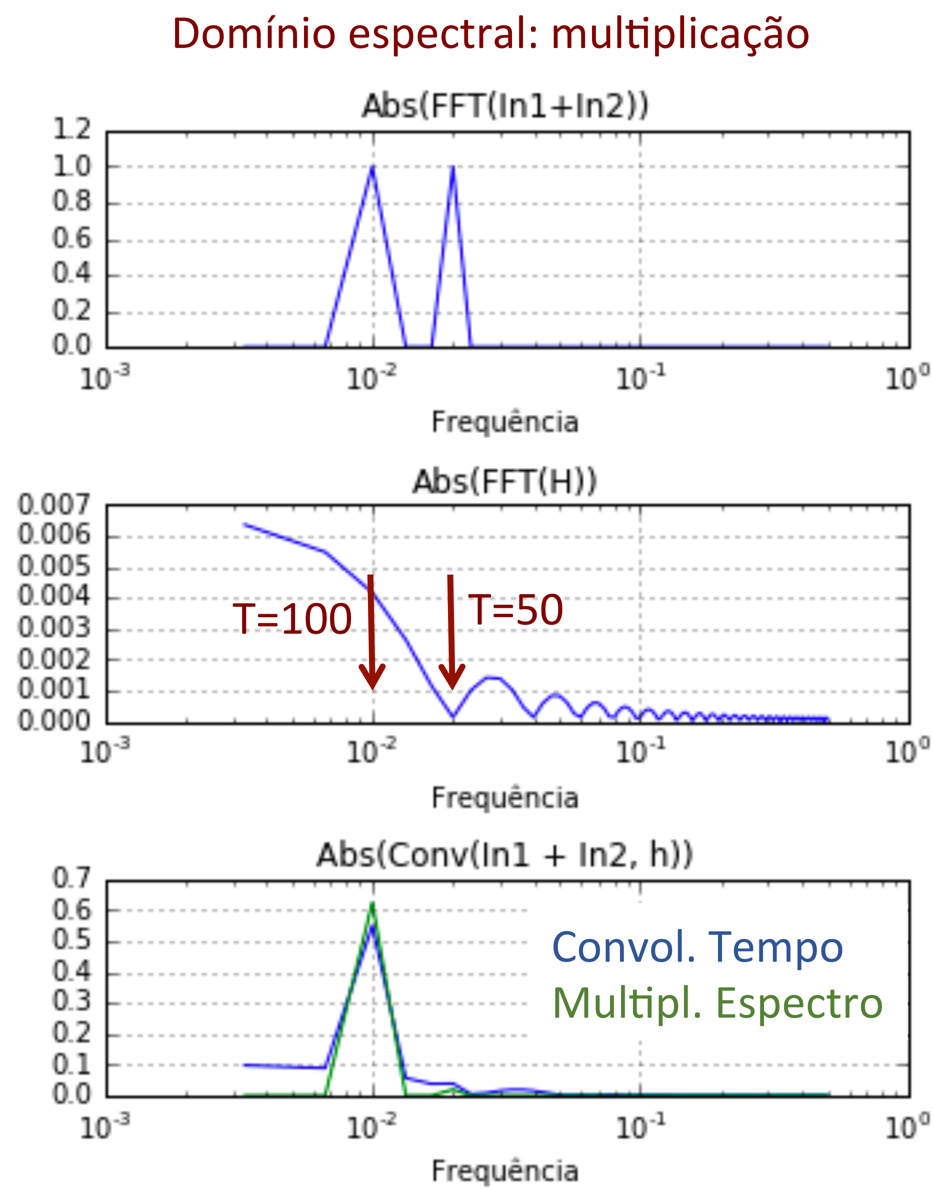
Domínio espectral: multiplicação



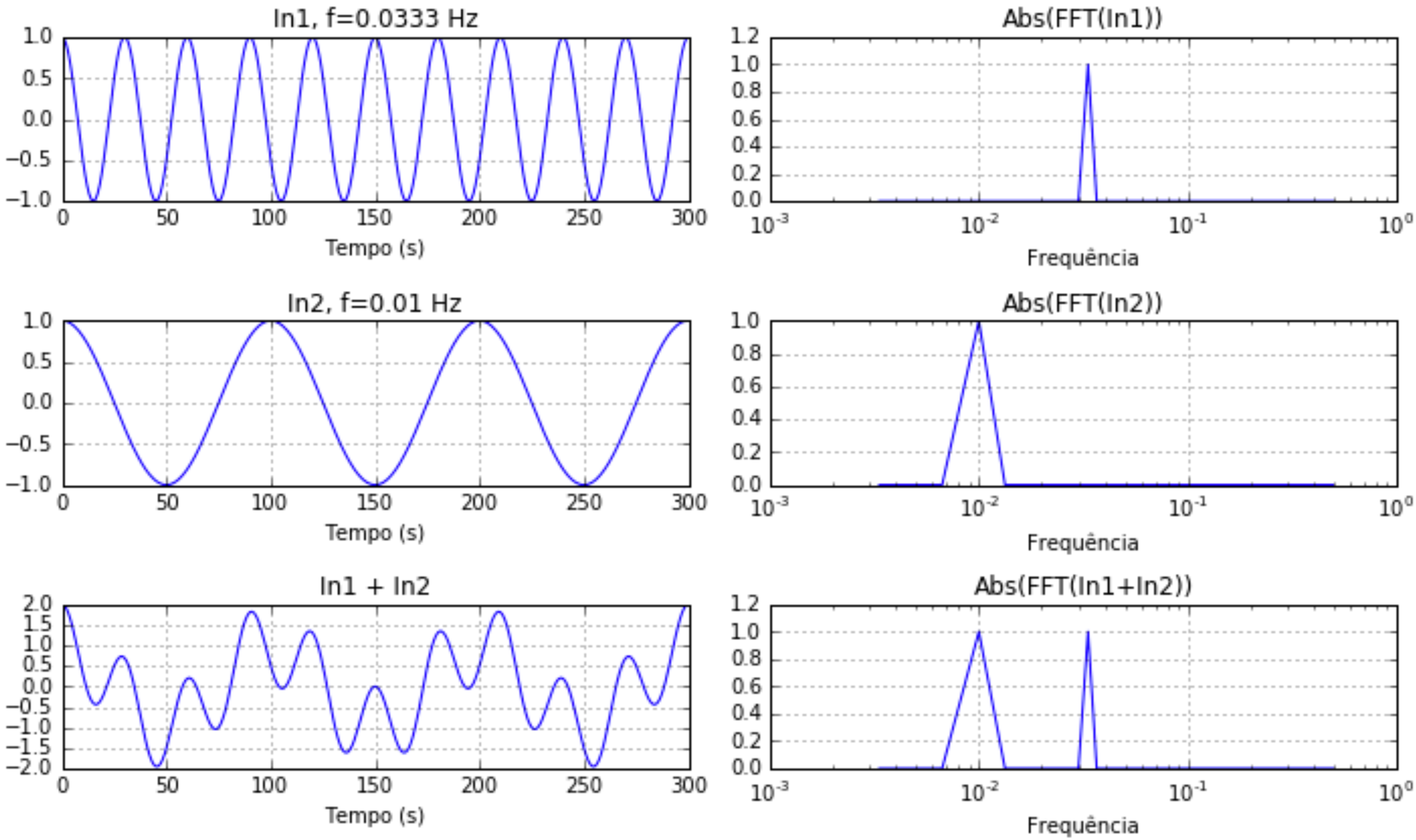
Filtro de média móvel (função de transferência)

O filtro até funcionou bem, mas foi "sorte"...

A frequência a eliminar estava num dos mínimos....



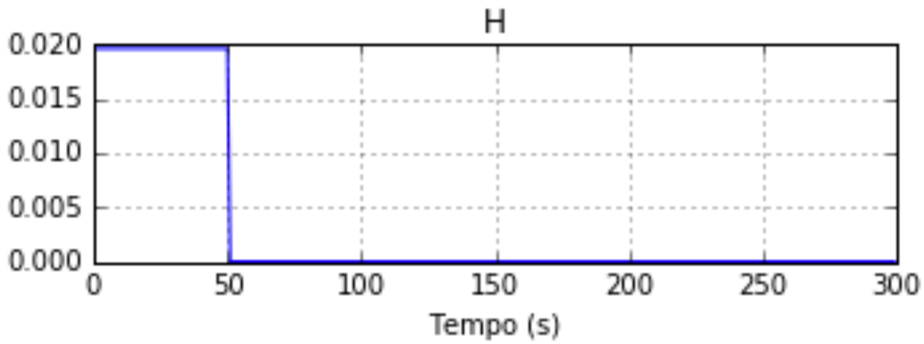
E com uma frequência mais alta? (T=30 s)

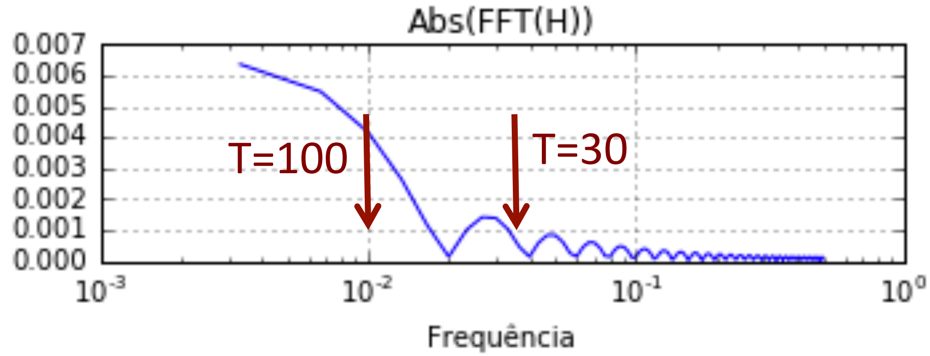


```python
import matplotlib.pyplot as plt
import numpy as np
from numpy import pi as pi
import numpy.fft as fft

#%%
## Cos 50
N=300; T=30.; dt=1.
t=np.arange(0.,N,dt)
f1=np.cos(2*pi * t/T)

tf1=fft.fft(f1)
fout1=tf1[:N/2]
fNyq=1/(2*dt); df=1/(N*dt);
freq=np.arange(0, fNyq, df)
N1=N

#%%
## Cos 100
T=100.;
f2=np.cos(2*pi * t/T)

tf2=fft.fft(f2)
fout2=tf2[:N/2]

#%%
## Cos 50 + cos 100
fs=f1+f2
tfs=fft.fft(fs)
fouts=tfs[:N/2]
```

```python
plt.rcParams['figure.figsize'] = 10, 6

plt.close();
plt.subplot(3,2,1);
plt.plot(t, f1)
plt.title('In1, f=0.02 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,3);
plt.plot(t, f2)
plt.title('In2, f=0.01 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,5);
plt.plot(t, fs)
plt.title('In1 + In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,2);
plt.semilogx(freq,np.abs(fout1)/(N/2))
plt.title('Abs(FFT(In1))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,4);
plt.semilogx(freq,np.abs(fout2)/(N/2))
plt.title('Abs(FFT(In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,6);
plt.semilogx(freq,np.abs(fouts)/(N/2))
plt.title('Abs(FFT(In1+In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.tight_layout()
```

E com uma frequência mais alta? (T=30 s)



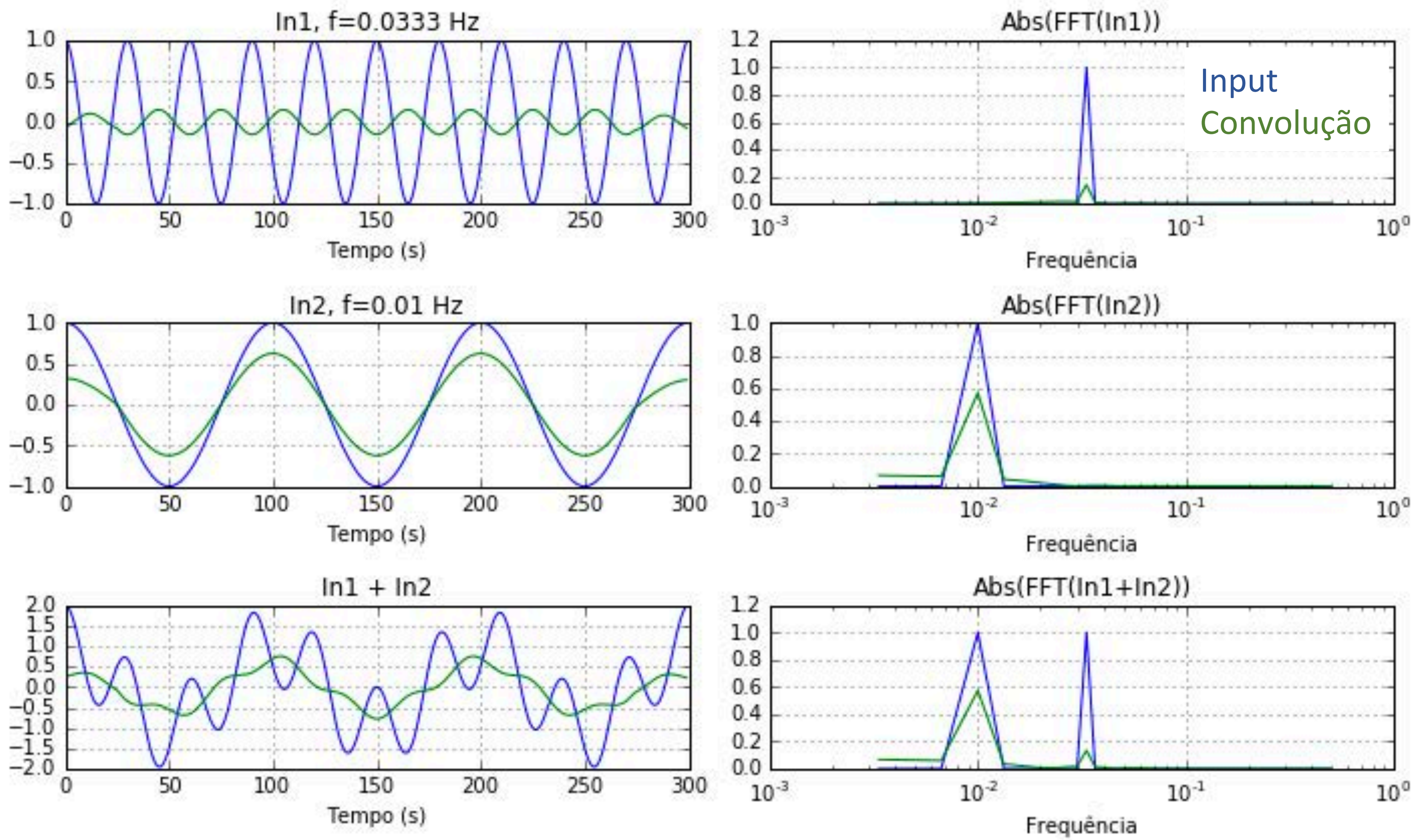


E com uma frequência mais alta? (T=30 s)

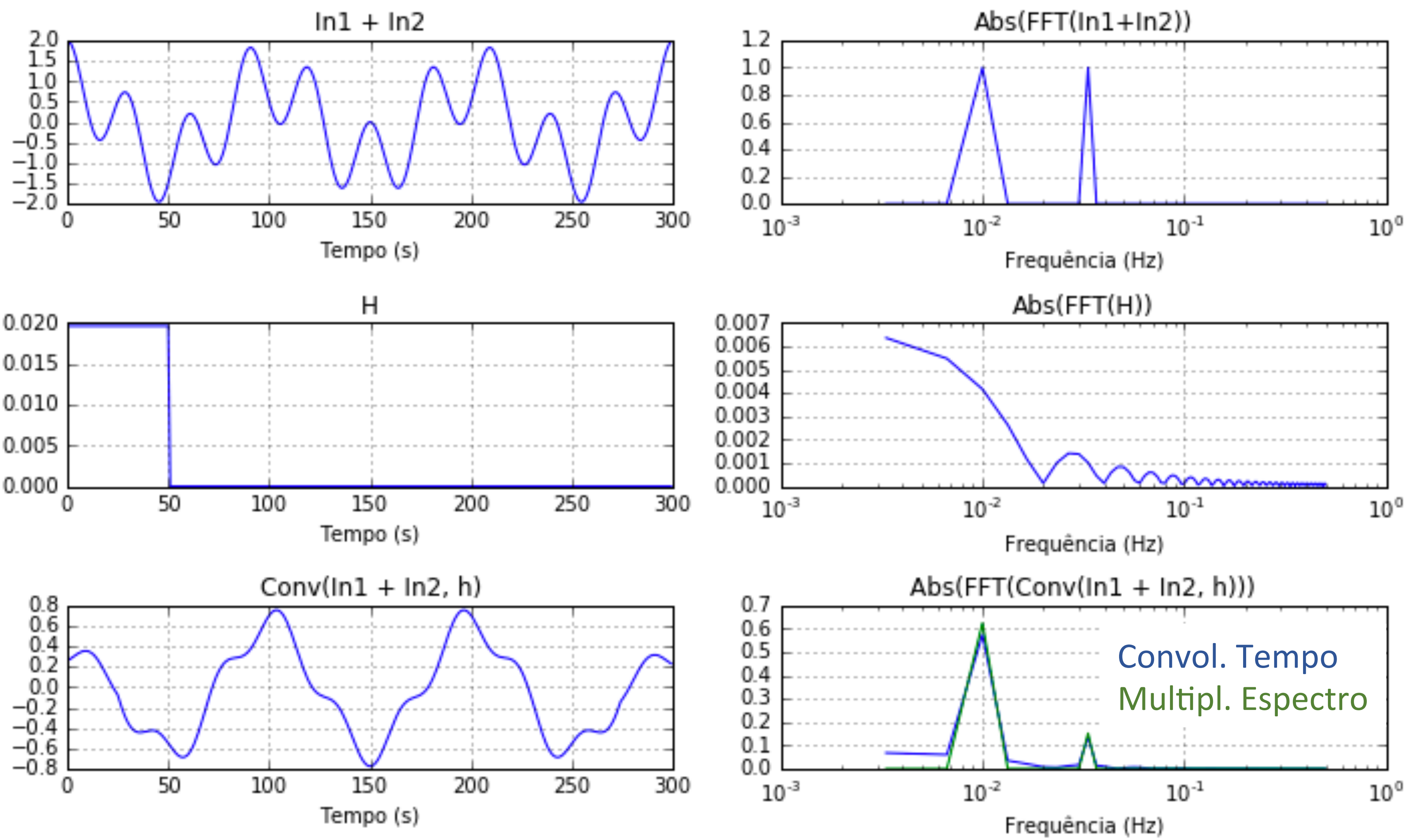
```
#%% Média móvel passa-baixo

NF=51;          # nr de pontos da janela
h=np.ones(NF)/NF;

c1=np.convolve(f1, h, mode='same')
c2=np.convolve(f2, h, mode='same')
cs=np.convolve(fs, h, mode='same')


#%% Espectros dos sinais filtrados

tfc1=fft.fft(c1)
foutc1=tfc1[:N/2]
Fc1=np.abs(foutc1)/(N/2)

tfc2=fft.fft(c2)
foutc2=tfc2[:N/2]
Fc2=np.abs(foutc2)/(N/2)

tfcs=fft.fft(cs)
foutcs=tfcs[:N/2]
Fcs=np.abs(foutcs)/(N/2)
```

```python
#%% Plots
plt.close();
plt.subplot(3,2,1);
plt.plot(t, f1, t, c1)
plt.title('In1, f=0.02 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,3);
plt.plot(t, f2, t, c2)
plt.title('In2, f=0.01 Hz'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,5);
plt.plot(t, fs, t, cs)
plt.title('In1 + In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,2);
plt.semilogx(freq,np.abs(fout1)/(N/2), freq, Fc1)
plt.title('Abs(FFT(In1))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,4);
plt.semilogx(freq,np.abs(fout2)/(N/2), freq, Fc2)
plt.title('Abs(FFT(In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,6);
plt.semilogx(freq,np.abs(fouts)/(N/2), freq, Fcs)
plt.title('Abs(FFT(In1+In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.tight_layout()
```

E com uma frequência mais alta? (T=30 s)
Convolução no domínio do tempo



E com uma frequência mais alta? (T=30 s)
Convolução no domínio do tempo



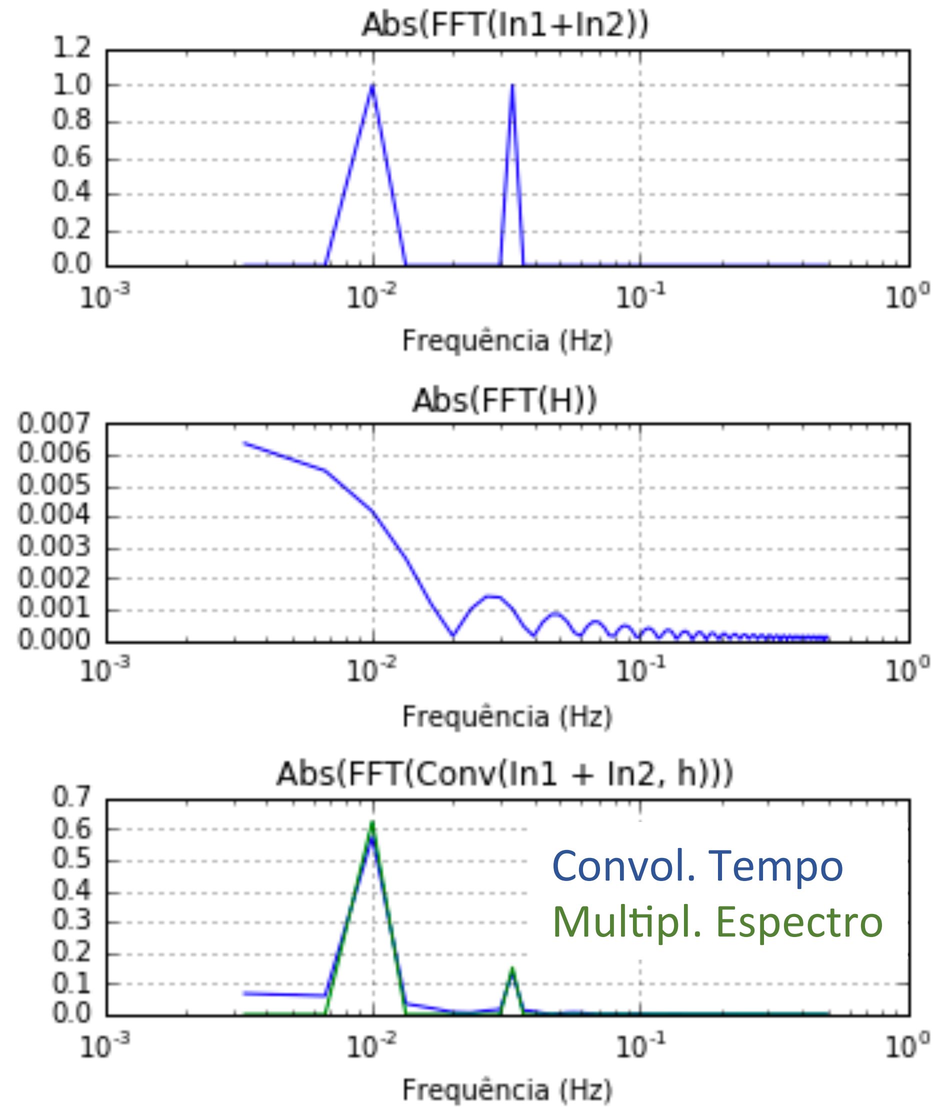
E com uma frequência mais alta? (T=30 s)
Convolução no domínio do tempo

A função de transferência:

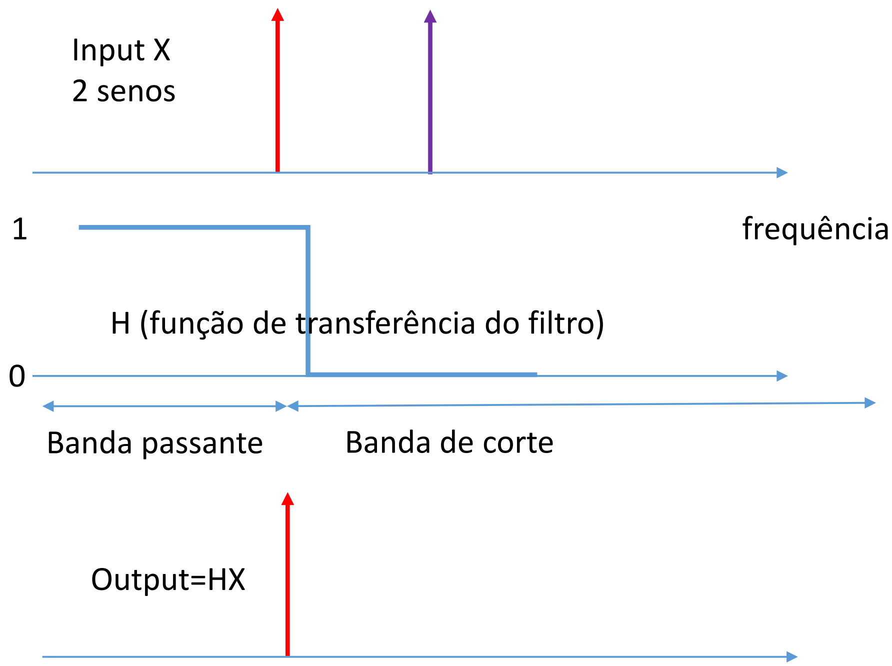
- Oscila
- Decai lentamente

É um mau filtro...

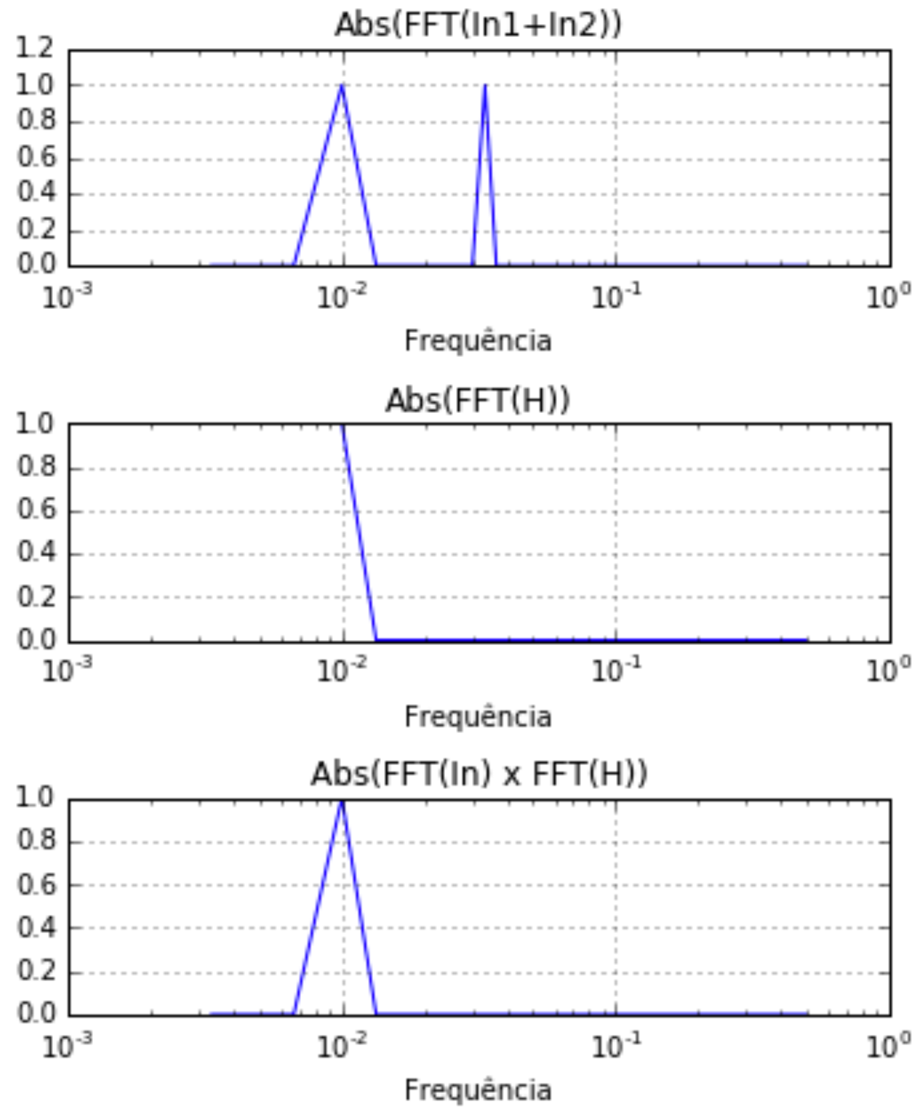
O comportamento do filtro de media móvel depende não só da frequência como também da sua localização exacta em relação às oscilações da função de transferência!



É melhor definir os filtros no domínio espectral



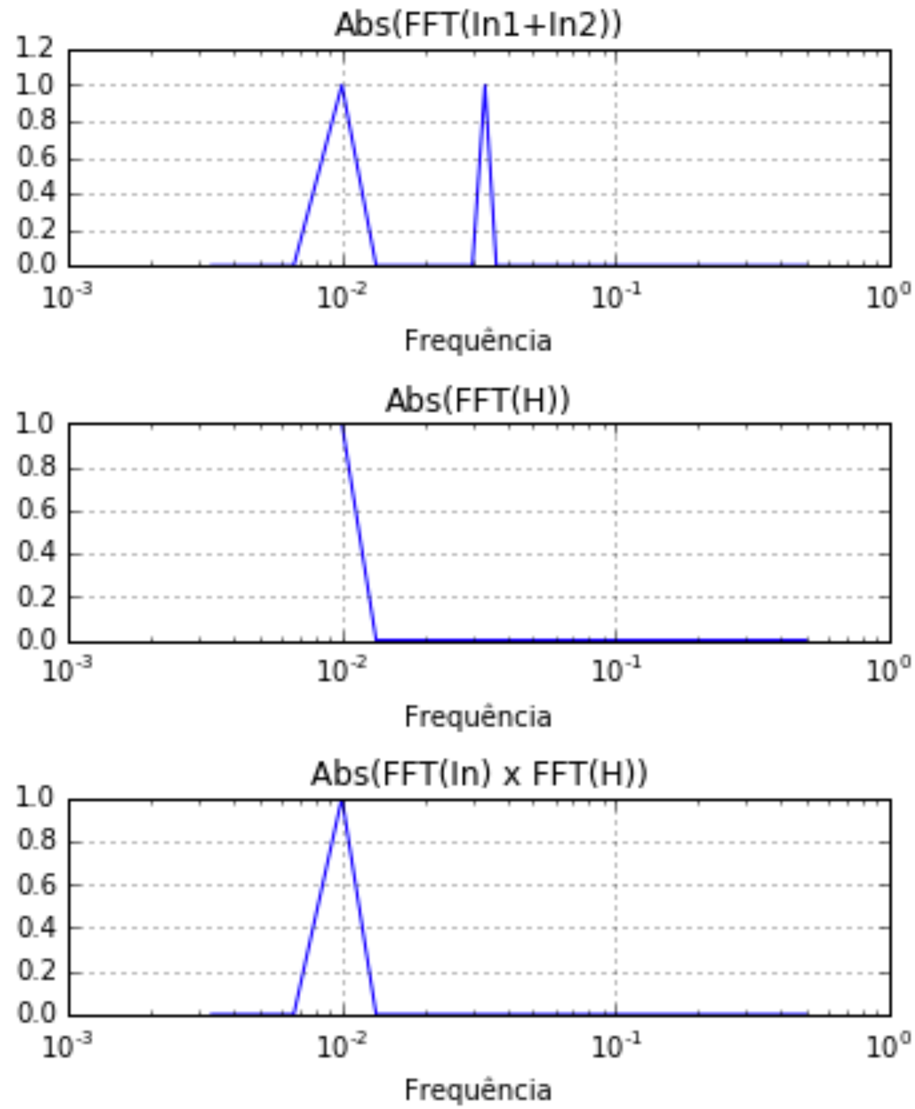
No caso discreto:



É melhor definir os filtros no domínio espectral

```
# Filtro no domínio do tempo
for ifr in range(0,N/2):
    if freq[ifr]<=1/100.:
        ifreq=ifr+1

FFTfilt=np.zeros(N)
FFTfilt[:ifreq]=1;
FFTfilt[-ifreq:]=1;

# Transformadas de Fourier inversas
ff1=fft.ifft(FFTfilt*tf1)
ff2=fft.ifft(FFTfilt*tf2)
ffs=fft.ifft(FFTfilt*tfs)
```

No caso discreto:

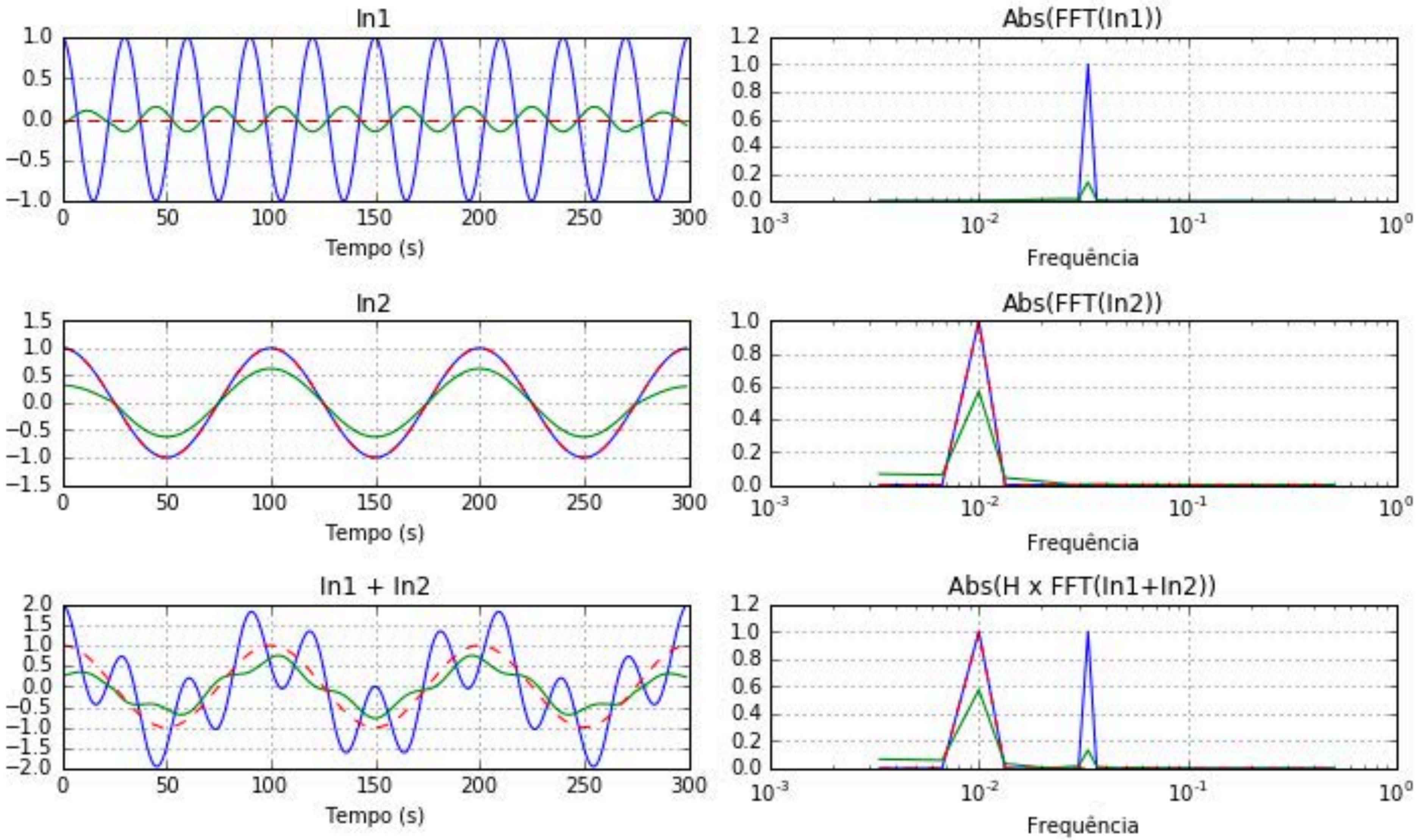


Atenção!

A função de transferência tem de estar definida em todo o domínio espectral (entre f=0 e f=$2f_{Nyq}$).

E com uma frequência mais alta? (T=30 s)
Multiplicação no domínio do espectro

Input
Convol. Tempo
Multipl. Espectro

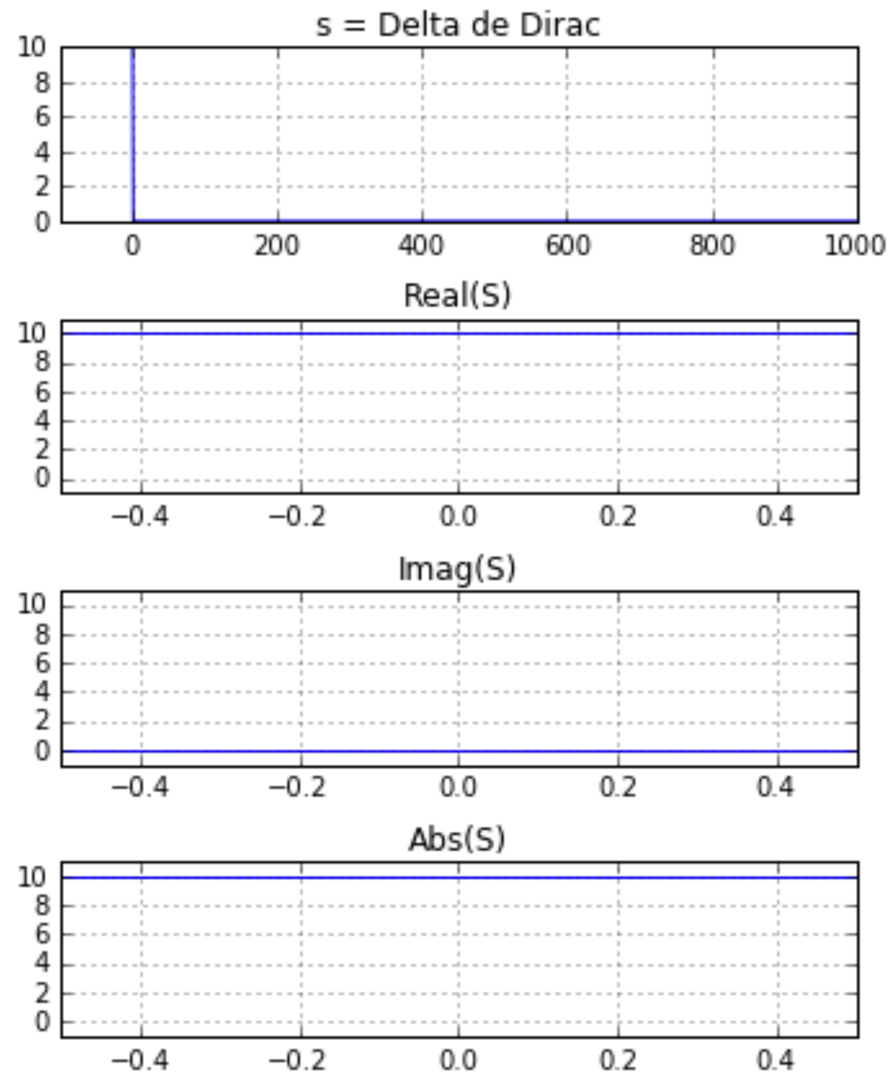


```
#%%
plt.close();
plt.subplot(3,2,1);
plt.plot(t, f1, t, c1, t, ff1, '--')
plt.title('In1'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,3);
plt.plot(t, f2, t, c2, t, ff2, '--')
plt.title('In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,5);
plt.plot(t, fs, t, cs, t, ffs, '--')
plt.title('In1 + In2'); plt.grid();
plt.xlabel(u'Tempo (s)')

plt.subplot(3,2,2);
plt.semilogx(freq,np.abs(tf1[:N/2])/(N/2), freq,np.abs(tfc1[:N/2])/(N/2),
             freq,np.abs((FFTfilt*tf1)[:N/2])/(N/2), '--')
plt.title('Abs(FFT(In1))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,4);
plt.semilogx(freq,np.abs(tf2[:N/2])/(N/2), freq,np.abs(tfc2[:N/2])/(N/2),
             freq,np.abs((FFTfilt*tf2)[:N/2])/(N/2), '--')
plt.title('Abs(FFT(In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.subplot(3,2,6);
plt.semilogx(freq, np.abs(tfs[:N/2])/(N/2), freq,np.abs(tfcs[:N/2])/(N/2),
             freq,np.abs((FFTfilt*tfs)[:N/2])/(N/2), '--')
plt.title('Abs(H x FFT(In1+In2))');
plt.grid();
plt.xlabel(u'Frequência')

plt.tight_layout()
```

Transformadas de Fourier de séries importantes

```
#%% Delta de Dirac 0
N=1000;
s=np.zeros(N)
t=np.arange(N)
dt=1.;
fNyq=1/(2*dt);
df=1/(dt*N);
freq=np.arange(-fNyq,fNyq,df);

#%%
s=np.zeros(N);
s[0]=10;
S=fft.fft(s);
SS=np.concatenate([S[N/2:], S[:N/2]])

plt.rcParams['figure.figsize'] = 5, 6
plt.close()

plt.subplot(4,1,1);
plt.plot(t,s); plt.grid()
plt.title(u's = Delta de Dirac');
plt.xlim([-100,1000])

plt.subplot(4,1,2);
plt.plot(freq,np.real(SS));
plt.title('Real(S)');  plt.grid()
plt.xlim([-.5,.5]); plt.ylim([-1,11])

plt.subplot(4,1,3);
plt.plot(freq,np.imag(SS));
plt.title('Imag(S)');  plt.grid()
plt.xlim([-.5,.5]); plt.ylim([-1,11])

plt.subplot(4,1,4);
plt.plot(freq,np.abs(SS));
```

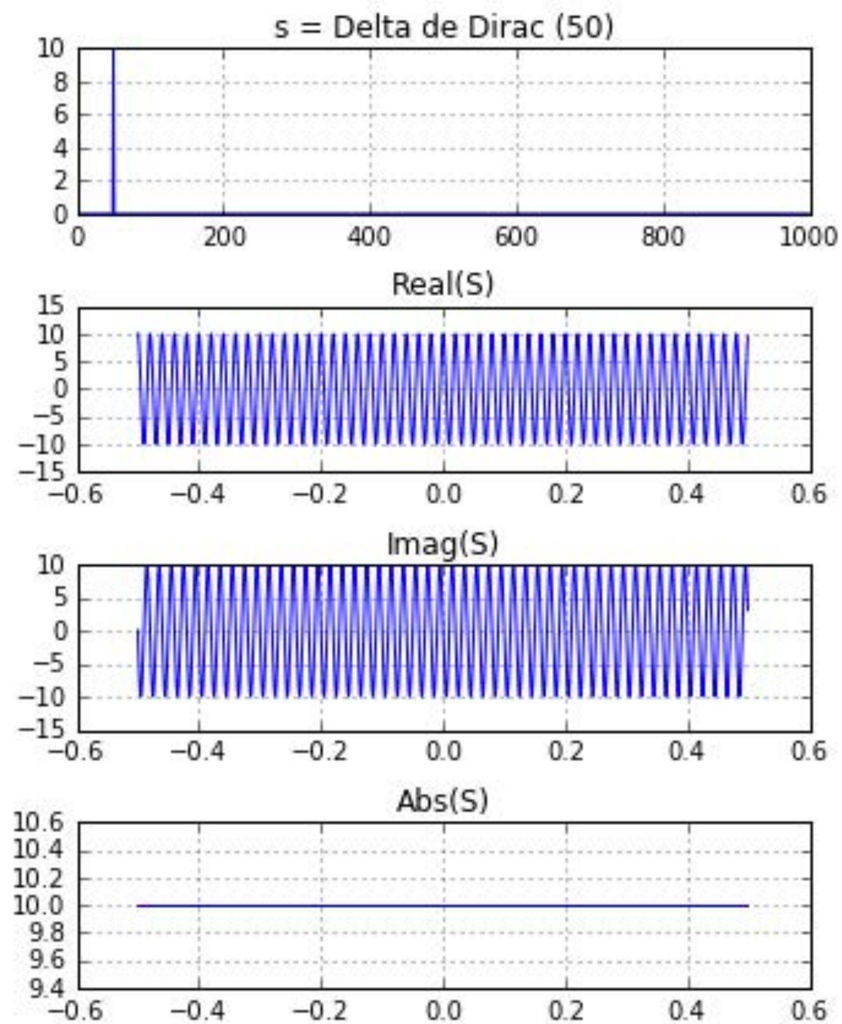
$\mathcal{F}(\delta_0 \text{ de Dirac}) = \text{constante}$



Transformadas de Fourier de séries importantes

```
#%% Delta de Dirac 50
N=1000;
s=np.zeros(N)
t=np.arange(N)
dt=1.;
fNyq=1/(2*dt);
df=1/(dt*N);
freq=np.arange(-fNyq,fNyq,df);

#%%
s[50]=10;
S=fft.fft(s);
SS=np.concatenate([S[N/2:], S[:N/2]])

plt.subplot(4,1,1);
plt.plot(t,s); plt.grid()
plt.title(u's = Delta de Dirac (50)');

plt.subplot(4,1,2);
plt.plot(freq,np.real(SS));
plt.title('Real(S)');  plt.grid()

plt.subplot(4,1,3);
plt.plot(freq,np.imag(SS));
plt.title('Imag(S)');  plt.grid()

plt.subplot(4,1,4);
plt.plot(freq,np.abs(SS));
plt.title('Abs(S)');  plt.grid()

plt.tight_layout()
```

$\mathcal{F}(\delta_{50}$ de Dirac$) =$ constante

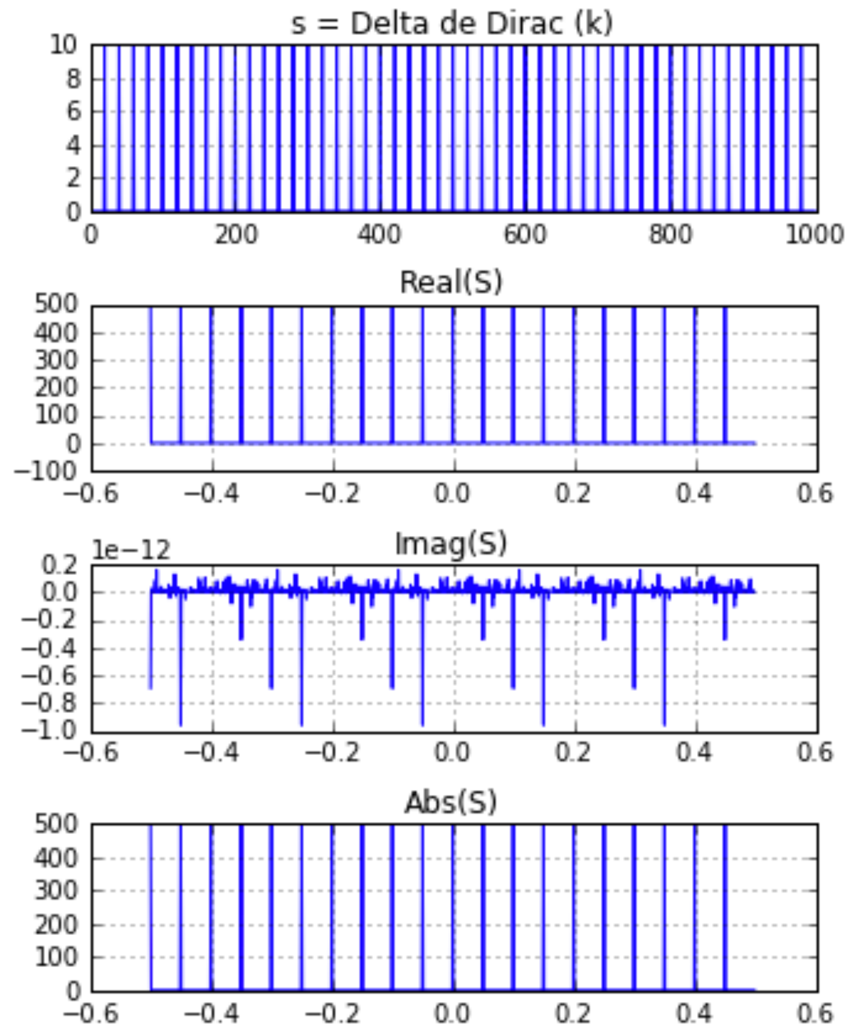


Transformadas de Fourier de séries importantes

```
#%% Pente
N=1000;
s=np.zeros(N)
t=np.arange(N)
dt=1.;
fNyq=1/(2*dt);
df=1/(dt*N);
freq=np.arange(-fNyq,fNyq,df);

#%%
s[range(0,N,20)]=10;
S=fft.fft(s);
SS=np.concatenate([S[N/2:], S[:N/2]])

plt.subplot(4,1,1);
plt.plot(t,s); plt.grid()
plt.title(u's = Delta de Dirac (k)');

plt.subplot(4,1,2);
plt.plot(freq,np.real(SS));
plt.title('Real(S)');  plt.grid()

plt.subplot(4,1,3);
plt.plot(freq,np.imag(SS));
plt.title('Imag(S)');  plt.grid()

plt.subplot(4,1,4);
plt.plot(freq,np.abs(SS));
plt.title('Abs(S)');  plt.grid()

plt.tight_layout()
```

Pente de Dirac: $\mathcal{F}(\delta_k \text{ de Dirac}) = \delta_n$

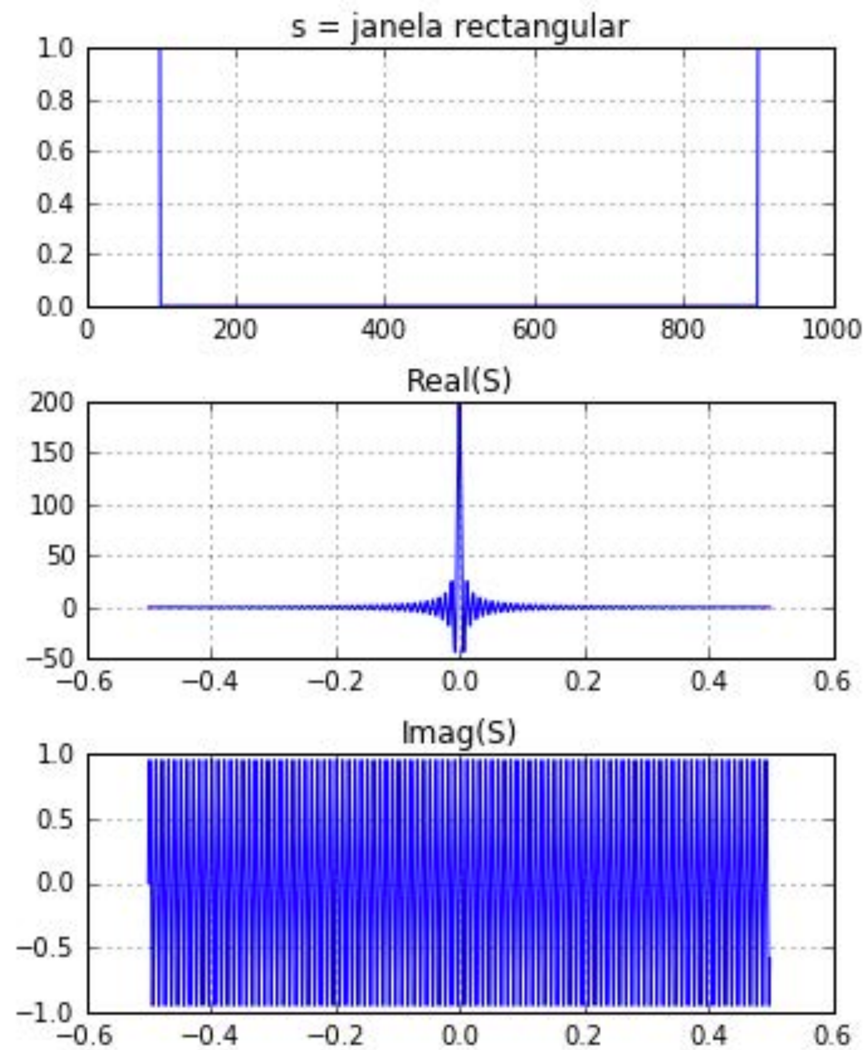
Amostrar regularmente = multiplicar por pente



Transformadas de Fourier de séries importantes

```
#%% Janela rectangular

s=np.zeros(N);
NJ=100
s[:NJ]=1;
s[-NJ:]=1;
S=fft.fft(s);
SS=np.concatenate([S[N/2:], S[:N/2]])

plt.rcParams['figure.figsize'] = 5, 6
plt.close()

plt.subplot(3,1,1);
plt.plot(t,s); plt.grid()
plt.title(u's = janela rectangular');

plt.subplot(3,1,2);
plt.plot(freq,np.real(SS));
plt.title('Real(S)');  plt.grid()

plt.subplot(3,1,3);
plt.plot(freq,np.imag(SS));
plt.title('Imag(S)');  plt.grid()
plt.ylim([-1,1])

plt.tight_layout()
```

Janela rectangular



Transformadas de Fourier de séries importantes

```
#%% Janela de Hann

s=np.zeros(N);
alpha=0.5;beta=1-alpha;
NH=300;NH2=NH/2;
th=np.arange(NH);
hann=alpha-beta*np.cos(2*pi*th/(NH))
hann=hann/np.sum(hann)
for i in range(NH2):
    s[i]=hann[NH2+i];
    s[-i]=hann[NH2-i];
S=fft.fft(s);
SS=np.concatenate([S[N/2:], S[:N/2]])

plt.subplot(3,1,1);
plt.plot(t,s); plt.grid()
plt.title(u's = janela Hann');

plt.subplot(3,1,2);
plt.plot(freq,np.real(SS));
plt.title('Real(S)');  plt.grid()

plt.subplot(3,1,3);
plt.plot(freq,np.imag(SS));
plt.title('Imag(S)');  plt.grid()
plt.ylim([-1,1])

plt.tight_layout()
```

Janela de Hann