Ciências ULisboa

Modelação Numérica
Aula 6

Filtros de Fourier

Filtros de Fourier

Se desenharmos diretamente $H(\omega)$, podemos fazer:

$$X = \mathcal{F}(x)$$

$$Y = HX$$

$$y = \mathcal{F}^{-1}(Y)$$

Como a fft é **muito** eficiente, o método permite implementar filtros quase-ideais.

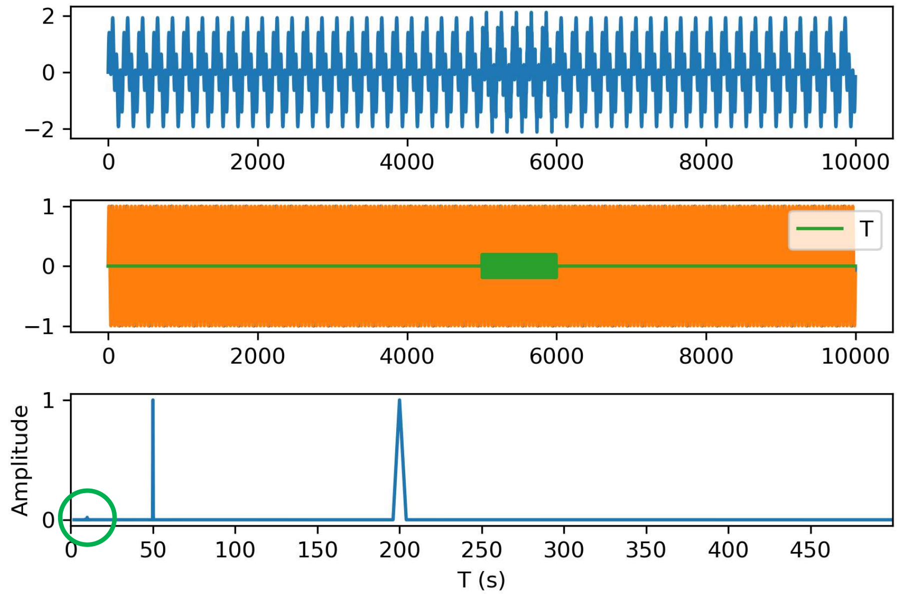
Mas atenção: H (tal com X) é complexo e tem que ser definido em todo o domínio $[-f_{Nyq}, f_{Nyq}]$ com as simetrias adequadas.

Vamos voltar a um sinal sintético

```
import numpy as np;import matplotlib.pyplot as plt
from time import process_time
from scipy import fft

N=10000;dt=1
T=[200,50];A=[1,1]
TT=10;AT=0.2;kT0=5000;kT1=6000
t=np.linspace(0,dt*(N-1),N)
f1=A[0]*np.sin(2*np.pi*t/T[0])
f2=A[1]*np.sin(2*np.pi*t/T[1])
fT=AT*np.sin(2*np.pi*t/TT);fT[0:kT0]=0;fT[kT1:]=0 #transiente
trend=0*t/10000 #sem tendência
f=f1+f2+fT+trend
```

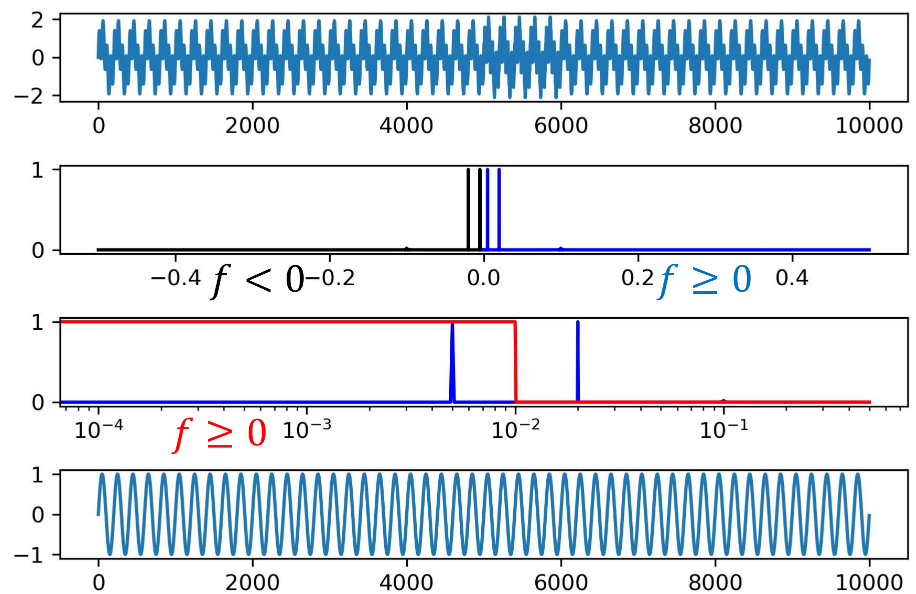
Sem tendência




```
fig,ax=plt.subplots(nrows=3)
ax[0].plot(t,f)
fNyq=1/(2*dt); df=2*fNyq/(N-1) #N par
freq=np.arange(0,fNyq+df,df)
ax[1].plot(t,f1);ax[1].plot(t,f2)
ax[1].plot(t,fT,label=r'T')#transiente
ax[1].legend()
F=fft.fft(f) ax[2].plot(1/freq,np.abs(F[0:N//2+1])/(N//2)) #f ≥ 0
ax[2].set_xlim(0,500)
ax[2].set_xticks(np.arange(0,500,50))
ax[2].set_ylabel('Amplitude');plt.xlabel('T (s)')
fig.tight_layout()
```

Filtro de Fourier

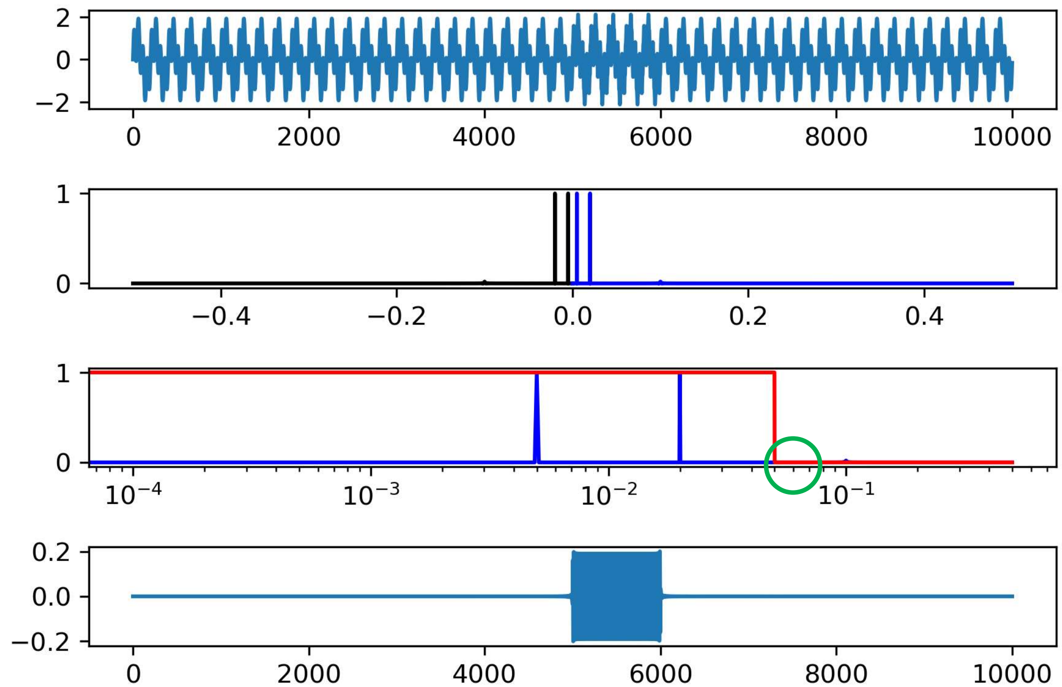
```
if N%2==0: #verifica se N é par
    dfL=df
else:
    dfL=0
freqR=np.arange(0,fNyq+df,df) #f ≥ 0
freqL=np.arange(-fNyq+dfL,0,df) #f < 0
F=fft.fft(f)
FR=F[0:N//2+1];FL=F[N//2+1:]
fig,ax=plt.subplots(nrows=4)
ax[0].plot(t,f)
ax[1].plot(freqL,np.abs(FL)/(N//2),color='black')
ax[1].plot(freqR,np.abs(FR)/(N//2),color='blue')
ax[2].plot(freqR,np.abs(FR)/(N//2),color='blue')
ax[2].set_xscale('log')
H=np.zeros(F.shape)
cut=int(np.argwhere(freqR>0.01)[0]);
H[0:cut+1]=1 #f>=0
H[-cut:]=1 #f<0 (ATENÇÃO [cut,0,cut])
ax[2].plot(freqR,H[0:N//2+1],color='red')
FLow=H*F
fl=fft.ifft(FLow)
ax[3].plot(t,fl)
fig.tight_layout()
```



Passa-baixo

Passa-alto

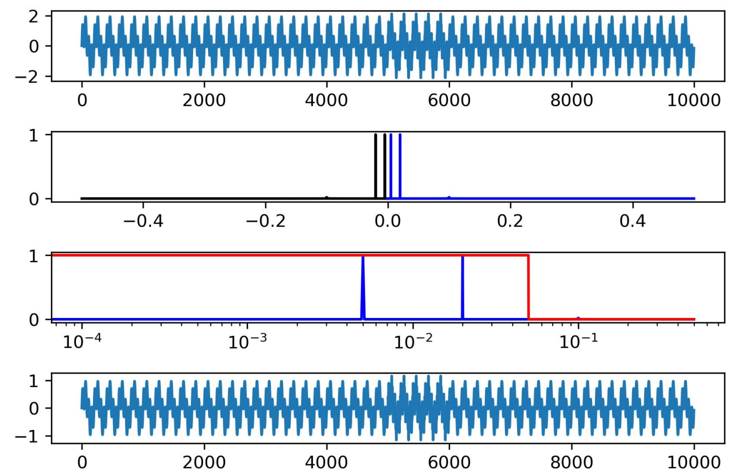
```
fig,ax=plt.subplots(nrows=4)
ax[0].plot(t,f)
ax[1].plot(freqL,np.abs(FL)/(N//2))
ax[1].plot(freqR,np.abs(FR)/(N//2))
ax[2].plot(freqR,np.abs(FR)/(N//2))
ax[2].set_xscale('log')
H=np.zeros(F.shape)
cut=int(np.argwhere(freqR>0.05)[0])
H[0:cut+1]=1;H[-cut:]=1
ax[2].plot(freqR,H[0:N//2+1],color='red')
FLow=H*F #filtro Passa-baixo
fl=fft.ifft(FLow)
fh=f-fl #passa-alto
ax[3].plot(t,fh)
fig.tight_layout()
```



Passa-alto

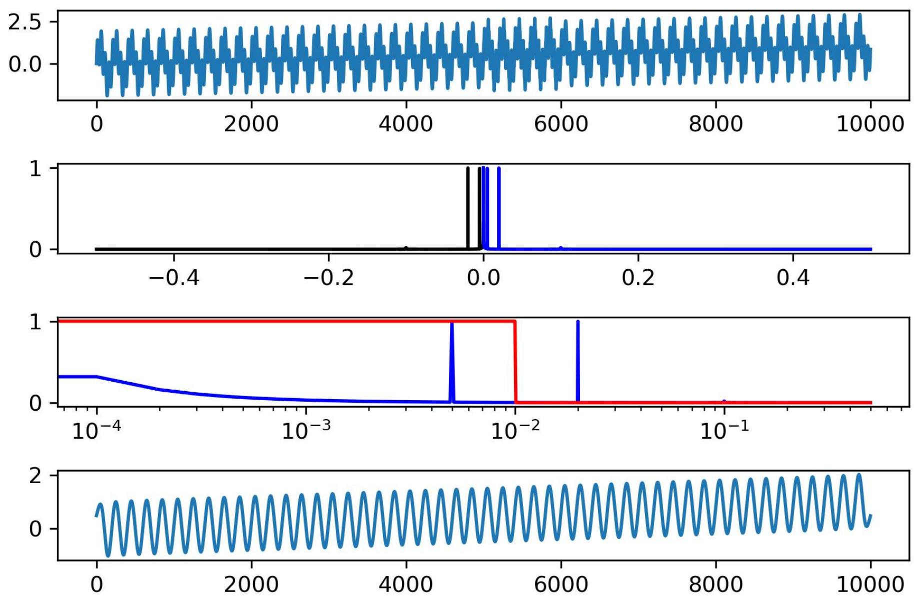
Notem que o filtro tem que atuar também em f<0

```
fig,ax=plt.subplots(nrows=4)
ax[0].plot(t,f)
ax[1].plot(freqL,np.abs(FL)/(N//2),color='black')
ax[1].plot(freqR,np.abs(FR)/(N//2),color='blue')
ax[2].plot(freqR,np.abs(FR)/(N//2),color='blue')
ax[2].set_xscale('log')
H=np.zeros(F.shape)
cut=int(np.argwhere(freqR>0.05)[0])
H[0:cut+1]=1;#H[-cut:]=1
ax[2].plot(freqR,H[0:N//2+1],color='red')
FLow=H*F #passa baixo (mal feito)
fl=fft.ifft(FLow)
fh=f-fl #passa alto
ax[3].plot(t,fh)
fig.tight_layout()
```

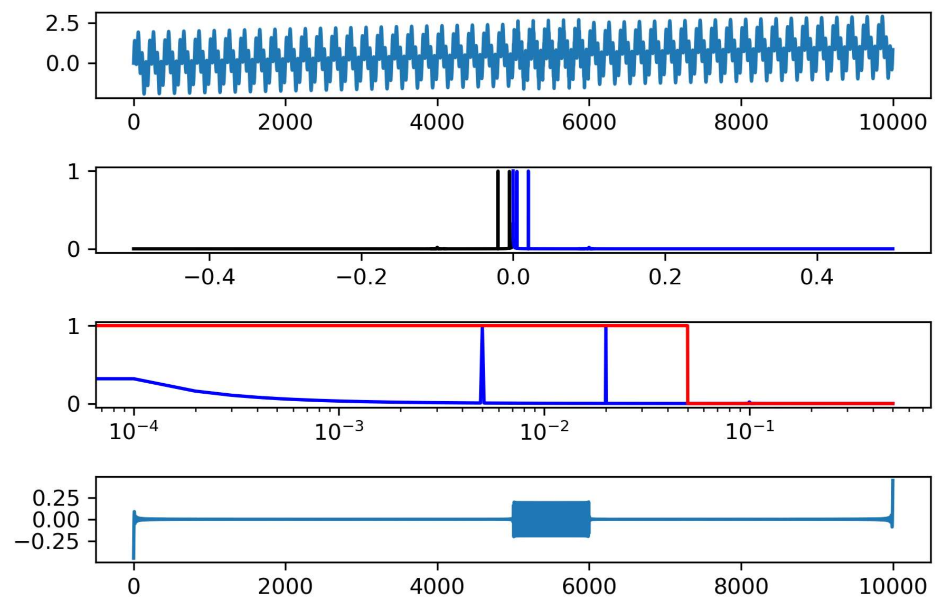


Deixou passas as harmonicas de baixa frequência negativa (metade da energia...)

Com tendência `trend=1*t/10000`



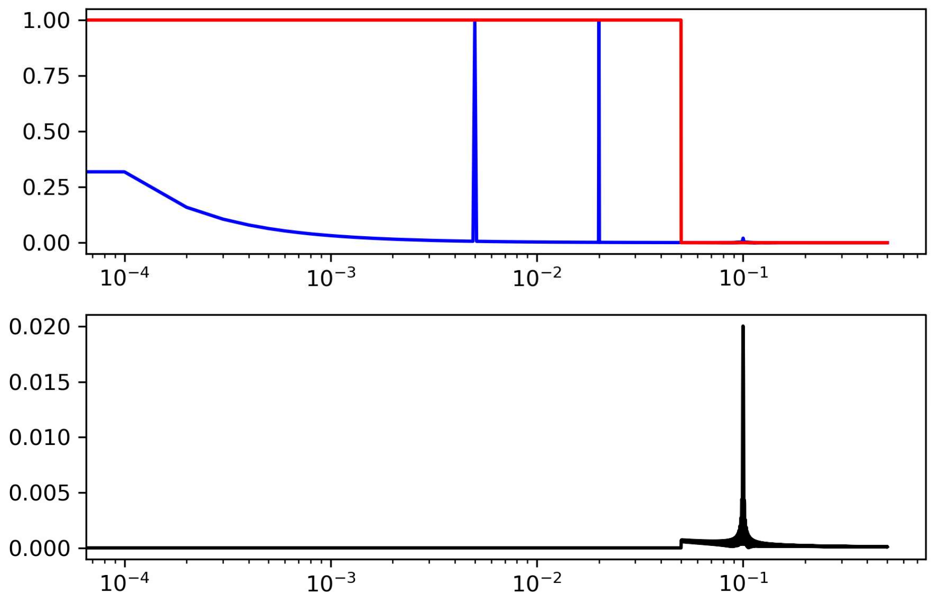
Passa-baixo



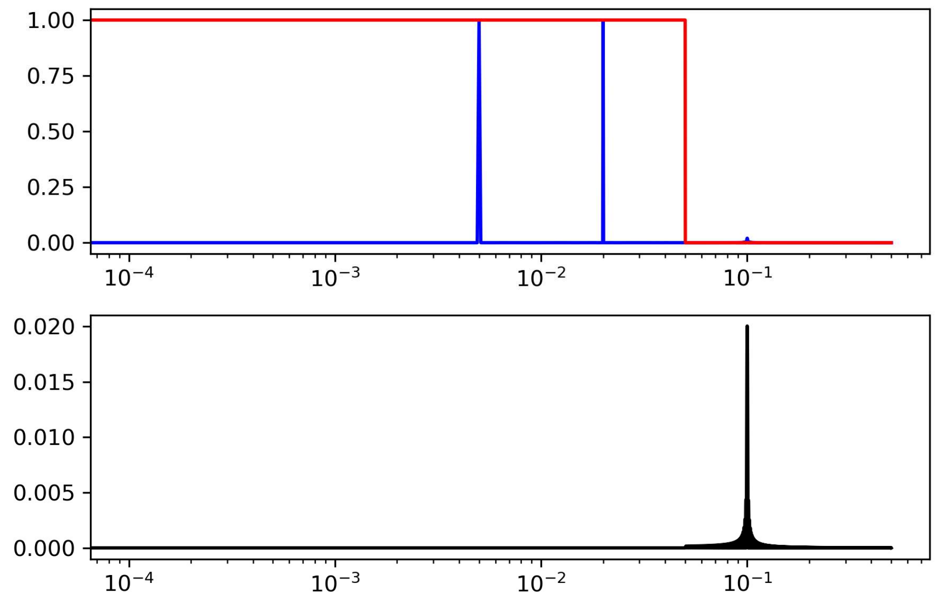
Passa-alto

O degrau (entre o fim e o princípio) deixa problemas na fronteira da série passa alto

Passa alto (diferenças subtis)



Com tendência



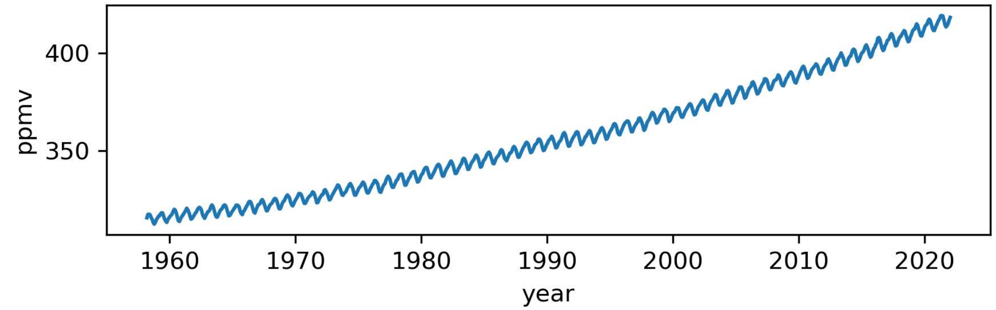
Sem tendência

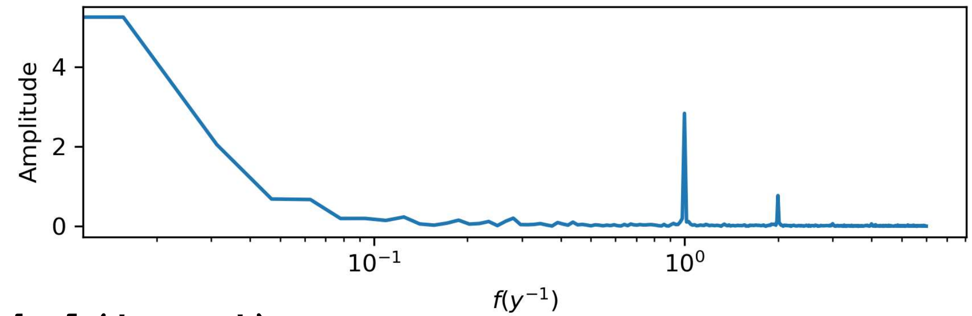
Uma tendência sintética

Seria fácil de resolver (calculando a regressão linear e subtrainda).

As variações reais de longo termo podem ser mais difíceis de caracterizar e de eliminar…

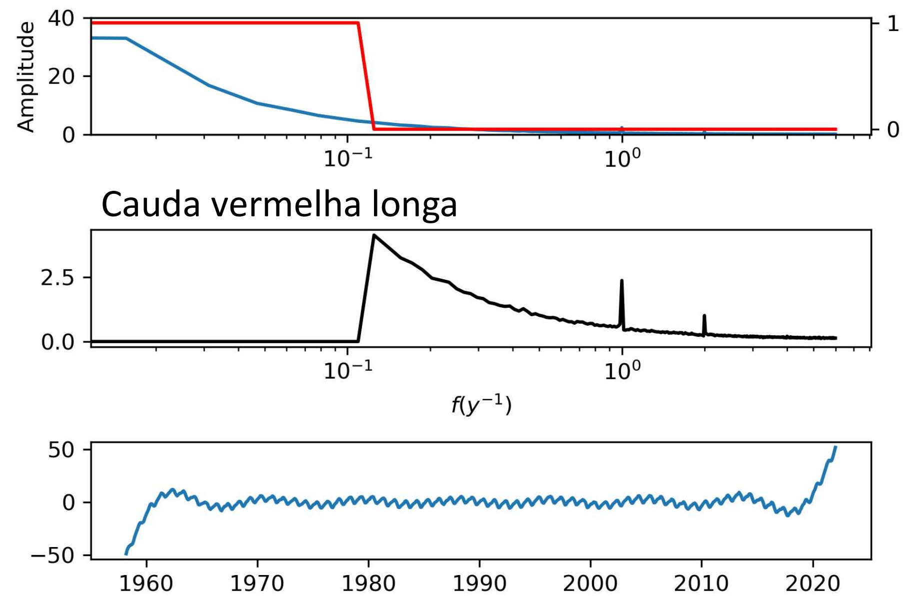
CO_2 Mauna Loa





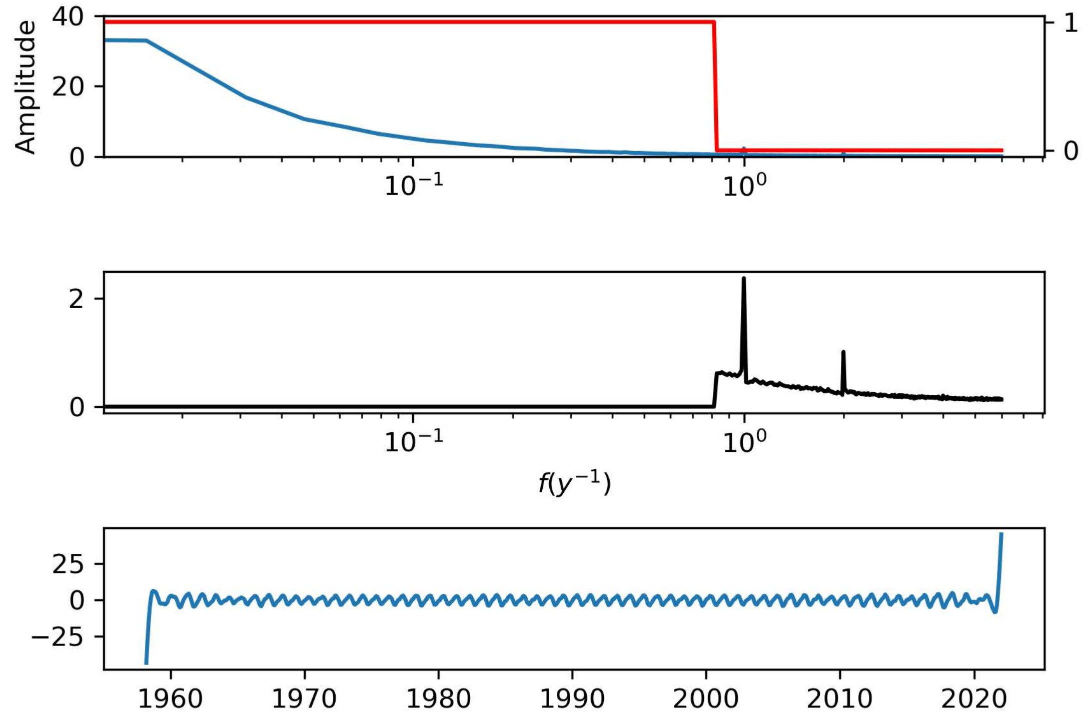

```
ML=np.loadtxt('MaunaLoa_NOAA.txt')
t=ML[:,2]
co2=ML[:,3]
dt=1/12. #ano)
N=len(co2)
F=fft.fft(co2)
fig,ax=plt.subplots(nrows=2)
ax[0].plot(t,co2)
ax[0].set_ylabel('ppmv');ax[0].set_xlabel('year')
fNyq=1/(2*dt); df=fNyq/((N+1)//2)
freq=np.arange(0,fNyq+df,df)
ax[1].plot(freq,np.abs(F[0:(N+1)//2+1])/(N//2))
ax[1].set_ylabel('Amplitude');ax[1].set_xlabel(r'$f (y^{-1})$')
ax[1].set_xscale('log')
fig.tight_layout()
```

```
fig,ax=plt.subplots(nrows=3)
ax[0].plot(freq,np.abs(F[0:(N+1)//2+1])/(N//2))
ax[0].set_ylabel('Amplitude’)
ax[1].set_xlabel(r'$f (y^{-1})$')
ax[0].set_xscale('log');ax[0].set_ylim(0,40)
axtwin=ax[0].twinx()
H=np.zeros(F.shape)
cut=int(np.argwhere(freq>=0.1)[0])
H[0:cut+1]=1;H[-cut:]=1
axtwin.plot(freq,H[0:(N+1)//2+1],color='red')
FLow=H*F #passa baixo (NO DOMÍNIO ESPECTRAL)
fl=fft.ifft(FLow)
fh=co2-fl #passa-alto (NO DOMÍNIO DO TEMPO)
FH=fft.fft(fh)
ax[1].plot(freq,np.abs(FH[0:(N+1)//2+1]/(N//2)))
ax[1].set_xscale('log')
ax[2].plot(t,fh)
fig.tight_layout()
```



Passa-alto

```
cut=int(np.argwhere(freq>=0.8)[0])
```



Comentários

No domínio espectral é fácil desenhar filtros "ideais".

Estes filtros só são aplicáveis ***a posteriori***, i.e. se for conhecida toda a série.

Mesmo estes filtros podem lidar mal com sinais não estacionários, pois a transformada de Fourier assume sempre **continuidade cíclica**, i.e. que a série se prolonga para sempre de forma periódica (repetindo a amostra).

Se a continuidade não se verificar vão aparecer artefactos nas fronteiras.