Ciências ULisboa

Modelação Numérica
Aula 5

Filtros de media móvel

Filtro linear não recursivo (convolução)

$$y_k = \sum_{n=0}^{N} x_n h_{k-n} \Longrightarrow y = x * h$$

Se:

$$\sum_{n=0}^{J} h_j = 1$$

Trata-se de um **filtro de média móvel**. Exemplo:

$$h_j = \frac{1}{J}$$

Teorema da convolução

A transformada de Fourier da convolução de duas séries é igual ao produto das transformadas das séries individuais:

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

Notar que Y, X, H são complexos.

H é designada por função de transferência do filtro h

Série sintética com 5 frequências

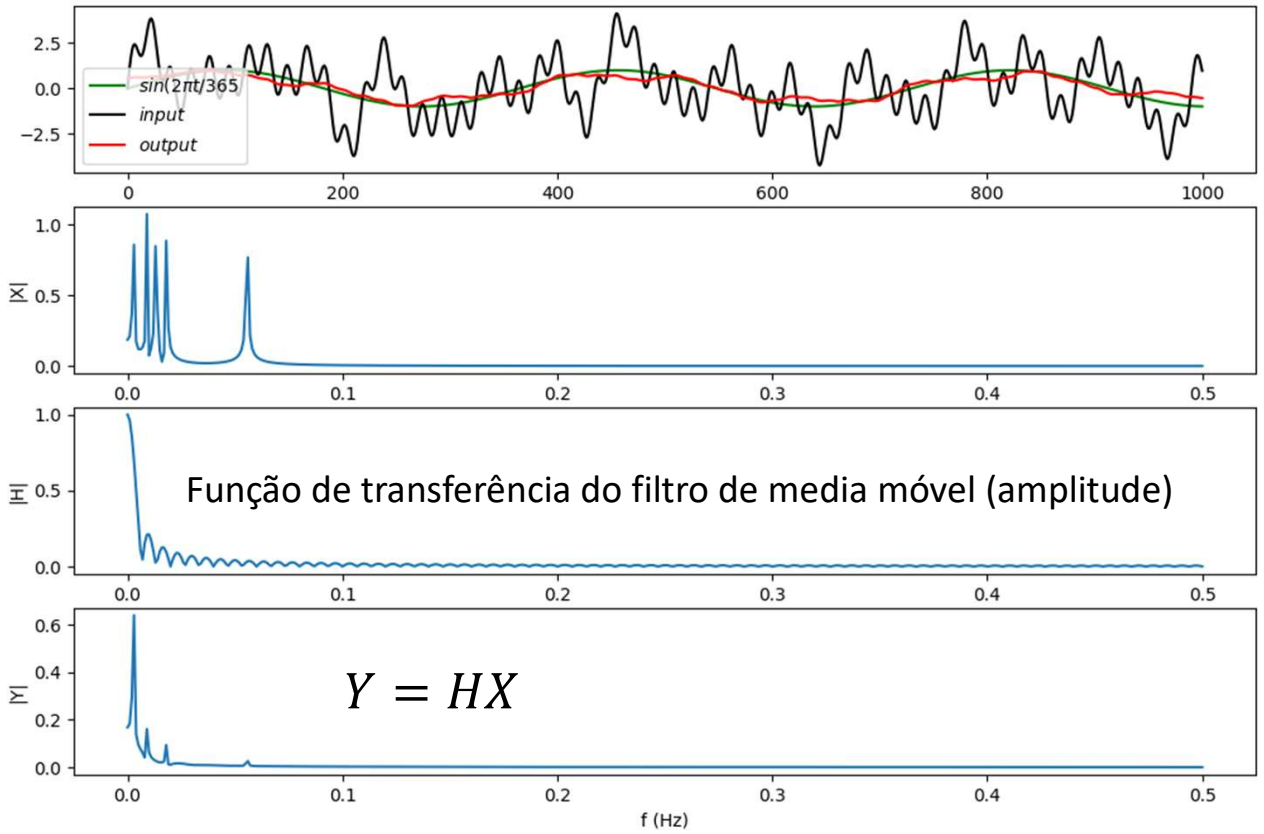
```
import numpy as np
import matplotlib.pyplot as plt
N=1001
dt=1.0
T=365.
t=np.arange(0.,(N-1)*dt+dt,dt)
x=np.sin(2*np.pi*t/T)
plt.subplot(4,1,1)
plt.plot(t,x,label=r"$sin(2\pi t/365$",color='green')
for Tr in [110.,75.,55.,18.]:
    x=x+np.sin(2*np.pi*t/Tr)
```

```
plt.plot(t,x,label=r"$input$",color='black')
nn=150 #comprimento do filtro
h=np.ones((nn))/nn #média móvel
y=np.convolve(x,h,mode='same')
plt.plot(t,y,color='red',label=r"$output$")
plt.legend()
X=np.fft.fft(x)
Y=np.fft.fft(y)
```

```
fNyq=1/(2*dt)
if N%2==0:
    df=2*fNyq/N
else:
   df=2*fNyq/(N-1)
freq=np.arange(0,fNyq+df,df)
plt.subplot(4,1,2)
plt.plot(freq,np.abs(X[0:N//2+1])/(N//2),label='X')
plt.ylabel('|X|') #espectro de amplitude de x
hExt=np.zeros(x.shape)
hExt[0:nn]=h #os outros termos são nulos
H=np.fft.fft(hExt)
plt.subplot(4,1,3)
plt.plot(freq,np.abs(H[0:N//2+1]),label='H')
plt.ylabel('|H|') #espectro de amplitude de h
plt.subplot(4,1,4)
plt.plot(freq,np.abs(Y[0:N//2+1])/(N//2),label='Y')
plt.ylabel('|Y|') #espectro de amplitude de y
plt.xlabel('f (Hz)')
```

Teorema da convolução

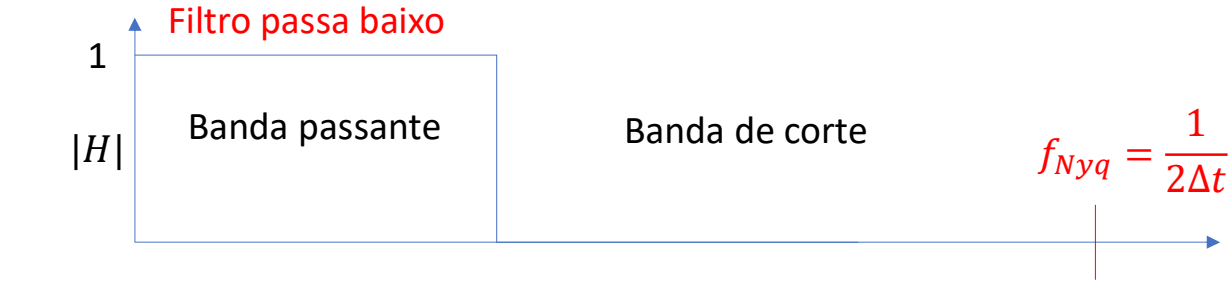
```
x=np.sin(2*np.pi*t/365)
for Tr in [110.,75.,55.,18.]:
    x=x+np.sin(2*np.pi*t/Tr)
```



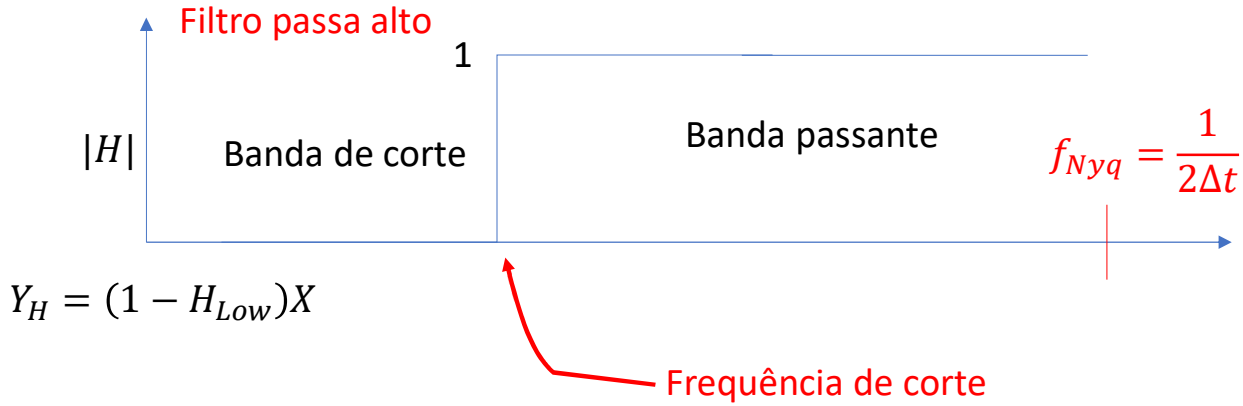
O que seria um filtro perfeito? (amplitude)

$Y_L = H_{Low}X$

$Y = HX$



$Y_H = H_{High}X$



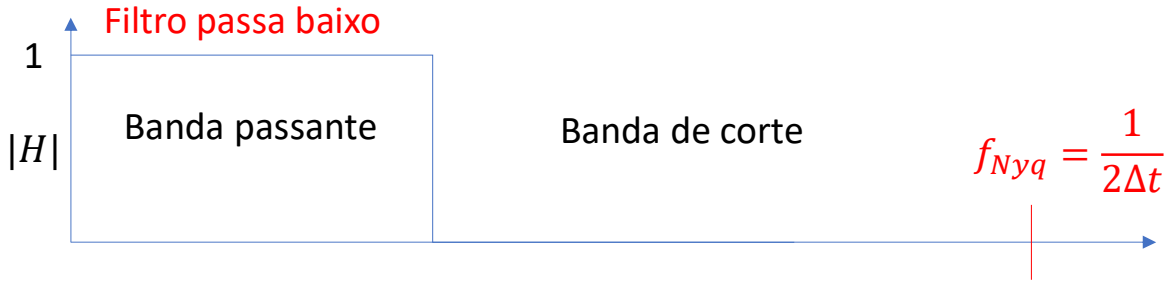
$Y_H = (1 - H_{Low})X$

Frequência de corte

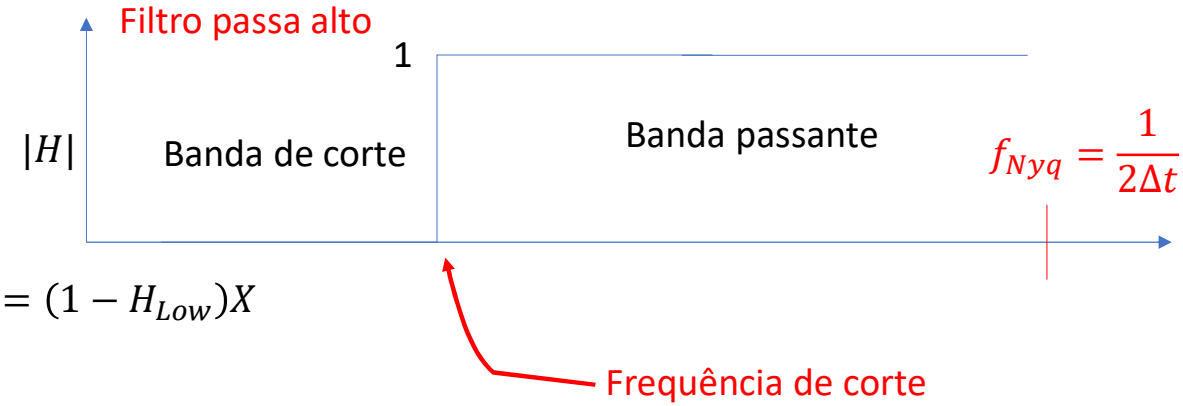
Filtro perfeito (resposta em amplitude)

$Y_L = H_{Low}X$

$Y = HX$



$Y_H = H_{High}X$



$Y_H = (1 - H_{Low})X$

Frequência de corte

Média móvel

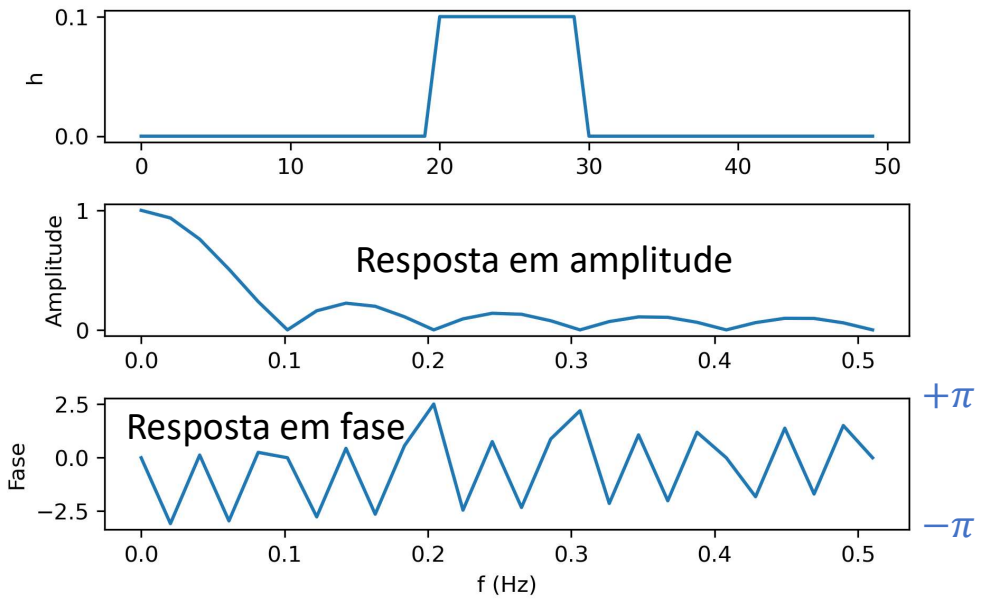
$$y_k = \sum_{n=0}^{N} x_n h_{k-n} \Longrightarrow y = x * h$$

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

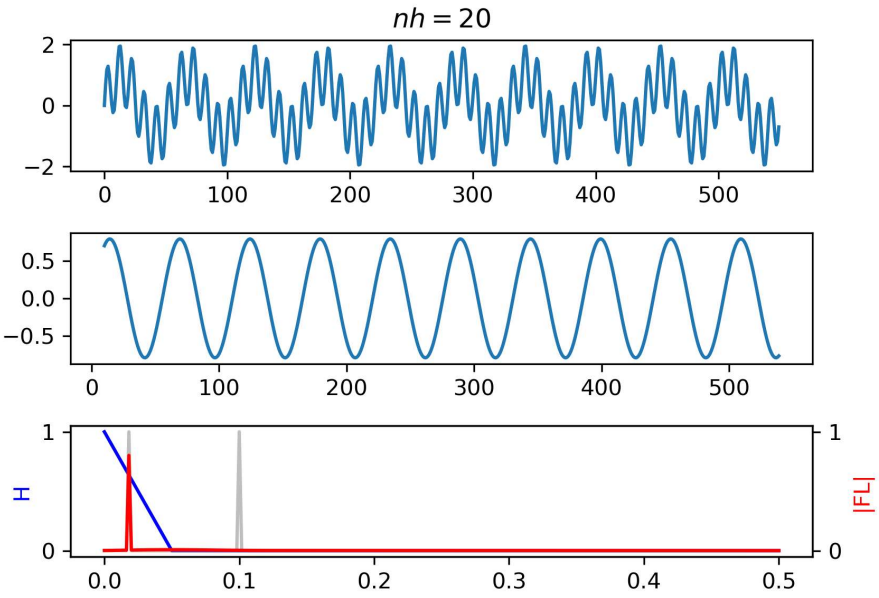
$H = \mathcal{F}(h)$ é a **função de transferência** do filtro

Atenção: $H \in \mathbb{C}$ (complexo)

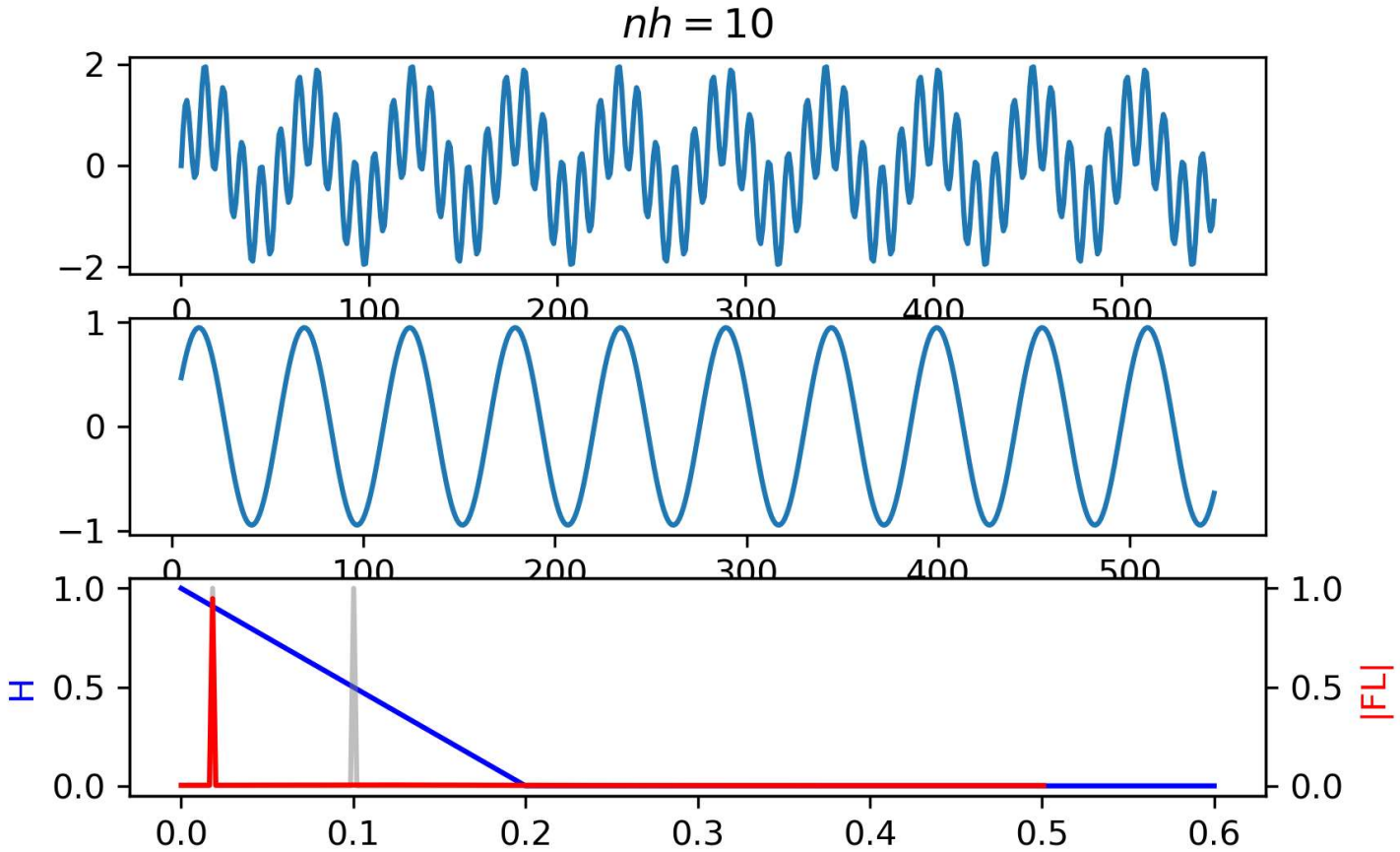
```
import numpy as np;import matplotlib.pyplot as plt
from time import process_time
from scipy import fft
N=50;dt=1
t=np.linspace(0,dt*(N-1),N)
nh=10;k0=20 #só altera a fase
h=np.zeros((N))
h[k0:k0+nh]=1/nh
H=fft.fft(h)
fNyq=1/(2*dt); df=2*fNyq/(N-1)
freq=np.arange(0,fNyq+df,df)
fig,ax=plt.subplots(nrows=3)
ax[0].plot(t,h);ax[0].set_ylabel('h')
H=fft.fft(h)
ax[1].plot(freq,np.abs(H[0:N//2+1]))  #não se escala por N//2!
ax[1].set_ylabel('Amplitude')
Fase=np.arctan2(np.imag(H),np.real(H))
ax[2].plot(freq,Fase[0:N//2+1])
ax[2].set_ylabel('Fase');plt.xlabel('f (Hz)')
fig.tight_layout()
```



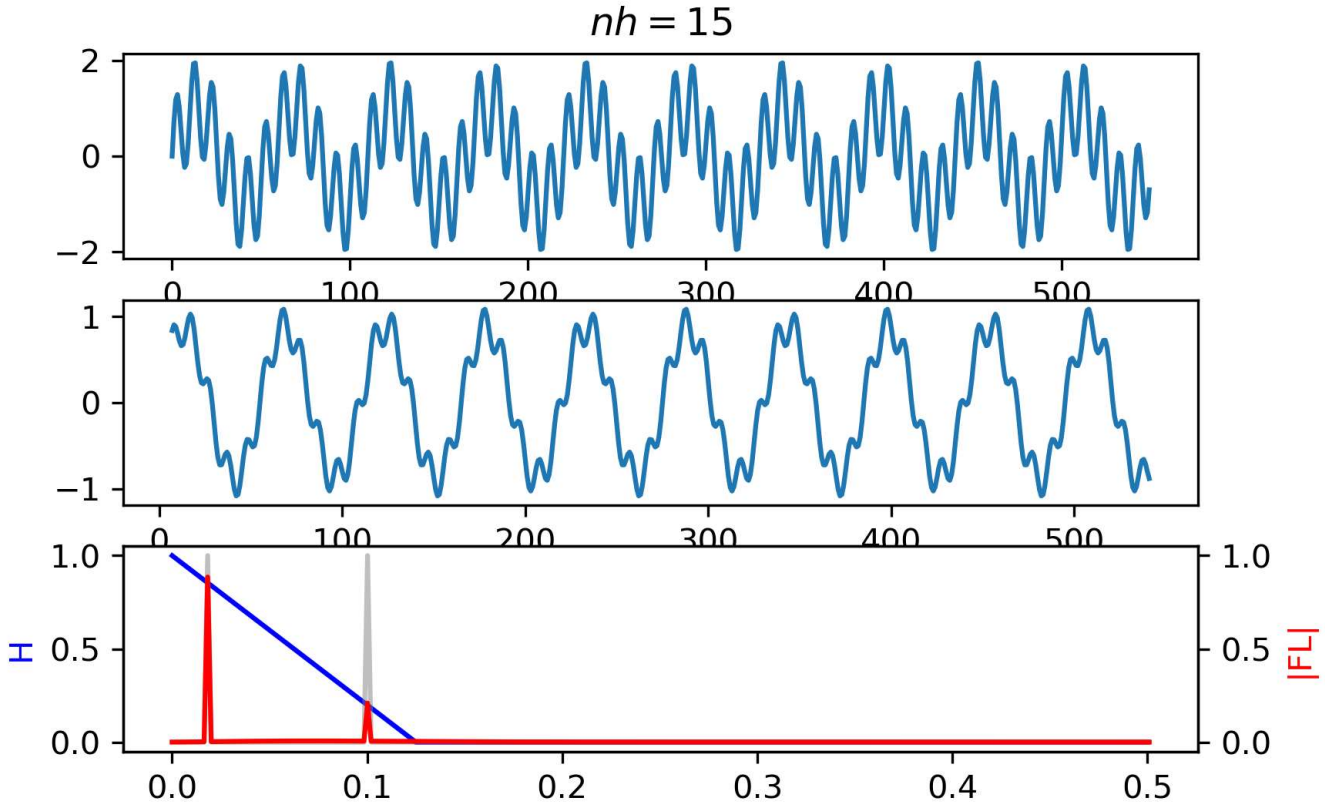
```
N = 550;dt = 1;T1 = 10;T2 = 55  # cf N=T1*T2
t = np.arange(0, dt*N, dt)
f = np.sin(2*np.pi*t/T1)+np.sin(2*np.pi*t/T2)
nh = 20
h = np.ones((nh))/nh
fig, ax = plt.subplots(nrows=3)
fL = np.convolve(f, h, mode='same')
fH=f-fL
ax[0].plot(t, f)
ax[1].plot(t[nh//2:-nh//2], fL[nh//2:-nh//2])
H = fft.fft(h)
fNyq = 1/(2*dt)
df = 2*fNyq/(2*(nh//2))
freqH = np.arange(0, fNyq+df, df)
ax[2].plot(freqH, np.abs(H[0:len(freqH)]), color='blue')
axt = ax[2].twinx()
F = fft.fft(f)
FL = fft.fft(fL)
fNyq = 1/(2*dt)
df = 2*fNyq/(2*(N//2))
freq = np.arange(0, fNyq+df, df)
axt.plot(freq, np.abs(F[0:N//2+1]/(N//2)), color='gray', alpha=0.5)
axt.plot(freq, np.abs(FL[0:N//2+1]/(N//2)), color='red')
axt.set_ylabel('|FL|', color='red')
ax[2].set_ylabel('H', color='blue')
ax[0].set_title(r'$nh=%3i$' % nh)
fig.tight_layout()
```



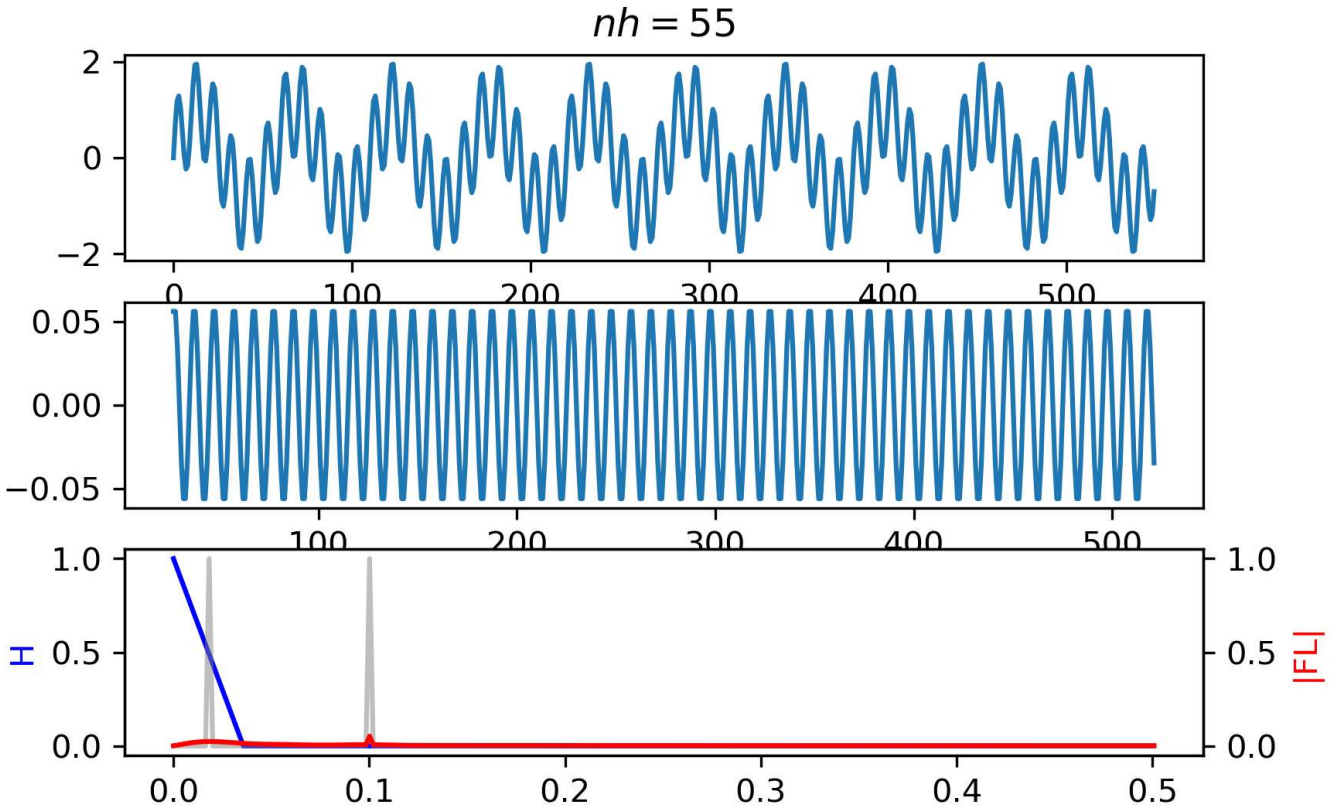
Outros **nh** **(T1=10;T2=55)**



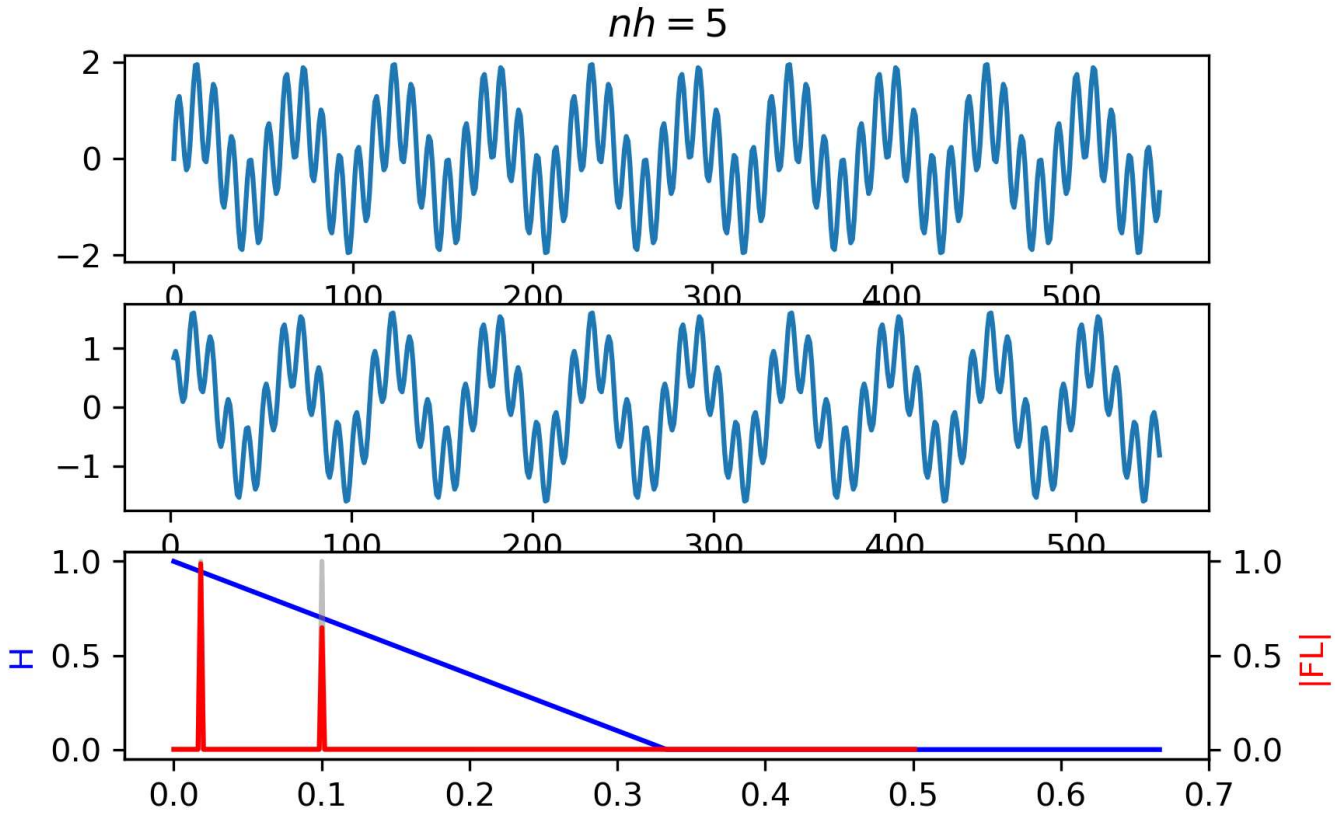
Outros **nh** **(T1=10;T2=55)**



Outros **nh** **(T1=10;T2=55)**



Outros **nh** **(T1=10;T2=55)**



Notem que

```
ax[1].plot(t[nh//2:-nh//2], fL[nh//2:-nh//2])
```

Estamos a esconder a zona de fronteira

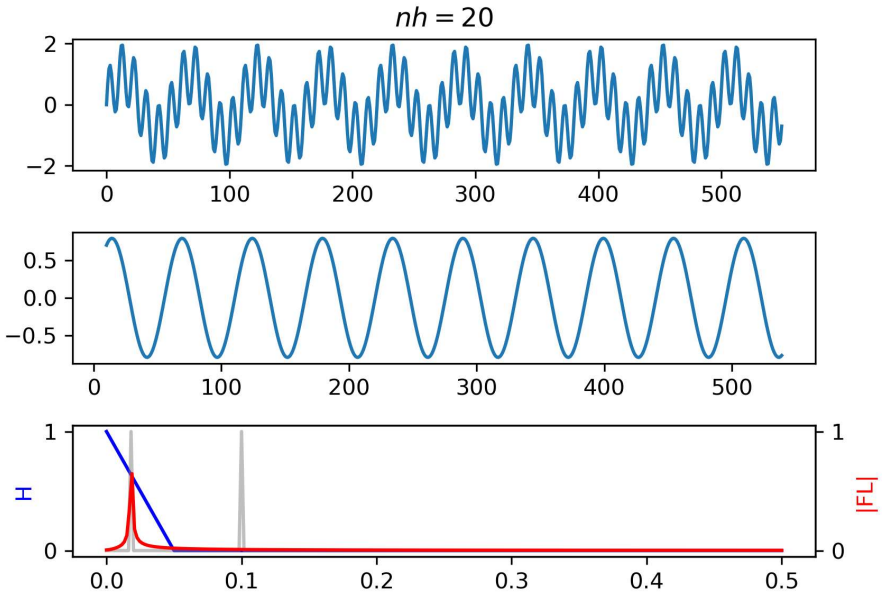
Mas quando fizemos

```
FL = fft.fft(fL)
```

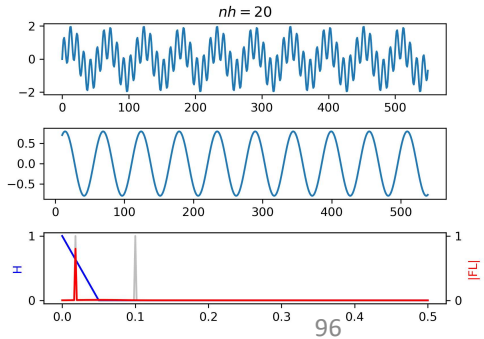
Usámos todos os pontos…

“Limpando” a fronteira

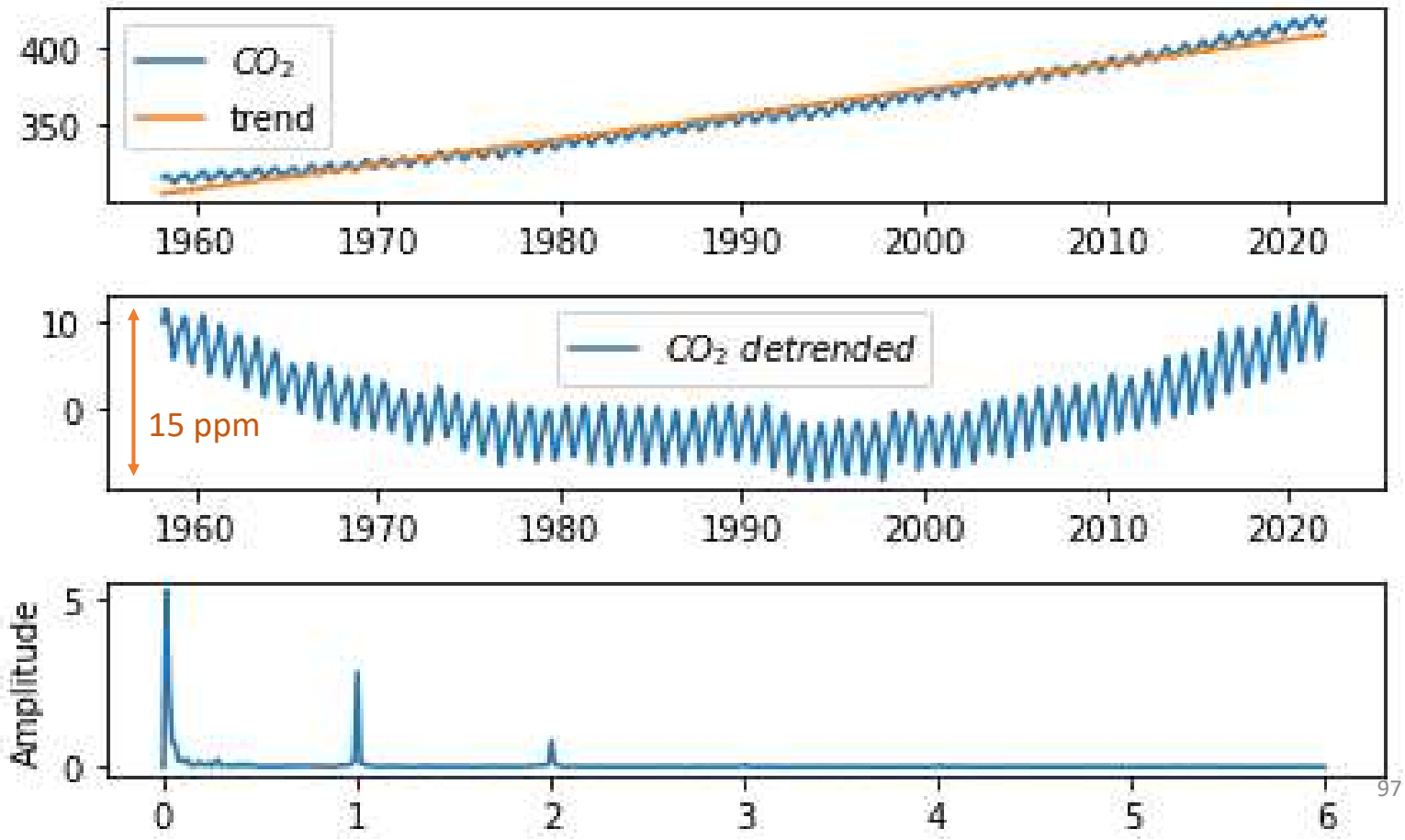
```
N = 550;dt = 1; fNyq = 1/(2*dt)
T1 = 10; T2 = 55  # cf N=T1*T2
t = np.arange(0, dt*N, dt)
f = np.sin(2*np.pi*t/T1)+np.sin(2*np.pi*t/T2)
nh = 20
h = np.ones((nh))/nh
fig, ax = plt.subplots(nrows=3)
fL = np.convolve(f, h, mode='same')
fH=f-fL
ax[0].plot(t, f)
ax[1].plot(t[nh//2:-nh//2],fL[nh//2:-nh//2])
H = fft.fft(h)
df = 2*fNyq/(2*(nh//2));freqH=np.arange(0,fNyq+df,df)
ax[2].plot(freqH,np.abs(H[0:len(freqH)]),color='blue')
axt = ax[2].twinx()
F = fft.fft(f); FL = fft.fft(fL[nh//2:-nh//2])
NFL=len(FL)
dfL = 2*fNyq/(2*(NFL//2));freqL = np.arange(0, fNyq+dfL, dfL)
df = 2*fNyq/(2*(N//2));freq = np.arange(0, fNyq+df, df)
axt.plot(freq, np.abs(F[0:N//2+1]/(N//2)), color='gray', alpha=0.5)
axt.plot(freqL, np.abs(FL[0:NFL//2+1]/(NFL/2)), color='red')
axt.set_ylabel('|FL|', color='red')
ax[2].set_ylabel('H', color='blue')
ax[0].set_title(r'$nh=%3i$' % nh); fig.tight_layout()
```



Análise exata (a menos do erro de aeredondamento)

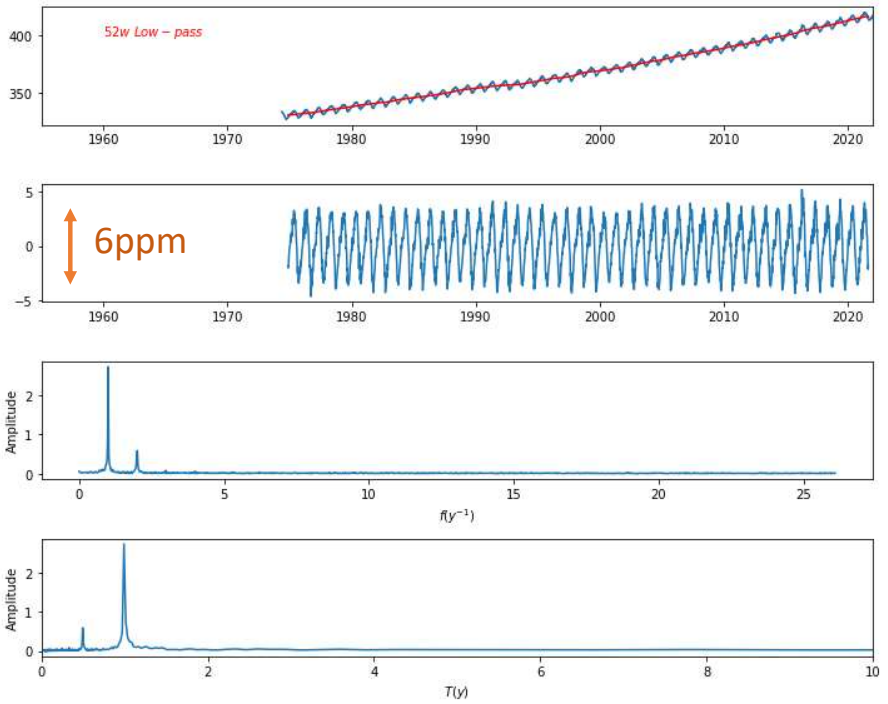


De volta a Mauna Loa



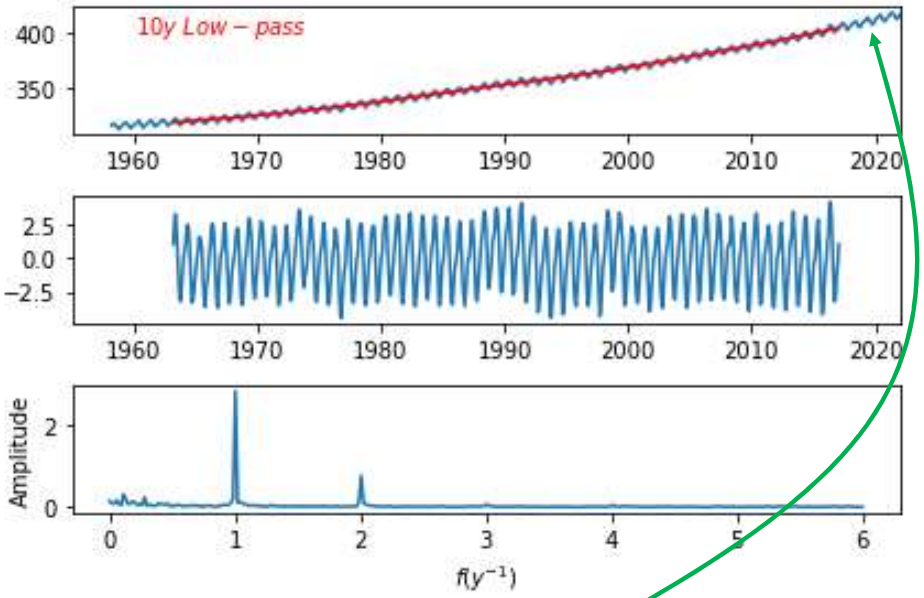
Média móvel (e passa alto)

```
t=np.copy(year); dt=7/365.25
nh=52
h=np.ones((nh))/nh
co2LP=np.convolve(co2I,h,mode='same')
fig,ax=plt.subplots(nrows=4)
ax[0].plot(t,co2I)
ax[0].plot(t[nh//2:-nh//2],\
      co2LP[nh//2:-nh//2])
co2DT=co2I-co2LP
ax[1].plot(t[nh//2:-nh//2],\
      co2DT[nh//2:-nh//2])
f=co2DT[nh//2:-nh//2];N=len(f)
F=fft.fft(f)
fNyq=1/(2*dt); df=2*fNyq/(N-1)
freq=np.arange(0,fNyq+df,df)
ax[2].plot(freq,np.abs(F[0:N//2+1])/(N//2))
ax[2].set_ylabel('Amplitude')
ax[2].set_xlabel(r'$f (y^{-1})$')
ax[3].plot(1/freq,np.abs(F[0:N//2+1])/(N//2))
ax[3].set_ylabel('Amplitude')
ax[3].set_xlabel(r'$T (y)$')
ax[3].set_xlim(0,10)
fig.tight_layout()
```



Eliminando baixas frequências

Voltaremos a este assunto (filtros)




```
ny=10; nh=12*ny
h=np.ones((nh))/nh
co2LP=np.convolve(co2,h,mode='same')
fig,ax=plt.subplots(nrows=3)
ax[0].plot(t,co2)
ax[0].plot(t[nh//2:-nh//2],co2LP[nh//2:-nh//2],color='red')
co2DT=co2-co2LP
ax[1].plot(t[nh//2:-nh//2],co2DT[nh//2:-nh//2])
f=co2DT[nh//2:-nh//2];N=len(f) #efeito de fronteira
ax[0].text(1960,400,r'$%2i y\ Low-pass$' % ny,color='red')
ax[0].set_xlim(1955,2022);ax[1].set_xlim(1955,2022)
F=fft.fft(f)
fNyq=1/(2*dt); df=2*fNyq/(N-1)
freq=np.arange(0,fNyq+df,df)
ax[2].plot(freq,np.abs(F[0:N//2+1])/(N//2))
ax[2].set_ylabel('Amplitude');ax[2].set_xlabel(r'$f (y^{-1})$')
fig.tight_layout()
```

Transformada discreta de Fourier

Transformada discreta de Fourier

$$F_k = \sum_{n=0}^{N-1} f_n e^{-2\pi i n \ /N}$$

transformada discreta inversa de Fourier

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} F_k e^{2\pi i n k/N}$$

$\{f_n\}$ e $\{F_k\}$ têm o mesmo número de termos (N), e a mesma informação! (*a menos do erro de arredondamento*)

A transformada discreta (com N termos)…

Quando a função é discretizada (amostrada a intervalo regular) existe um período mínimo (ou frequência máxima) que pode ser representado. Se ela tem um número finito de termos, também existe um período máximo (frequência mínima, para além de 0). Logo temos uma série discreta e finita (e com erro de arredondamento).

Na prática a análise numérica de dados reais refere-se sempre a esse tipo de série. Nesse caso tanto a representação da função (transformada inversa) como o cálculo dos coeficientes (transformada) envolve somatórios (não integrais) com um número finito de termos.

Propriedades da Transfomada Discreta de Fourier

Linearidade

A transformada de uma combinação linear de funções é a mesma combinação linear de transformadas

$$G = \mathcal{F}(g), H = \mathcal{F}(h) \Longrightarrow \mathcal{F}(ag + bh) = aG + bH$$

$$g = g(t), h = h(t); G = G(f), H = H(f)$$

$$t \equiv \text{tempo}, f \equiv \text{frequência}$$

Translação

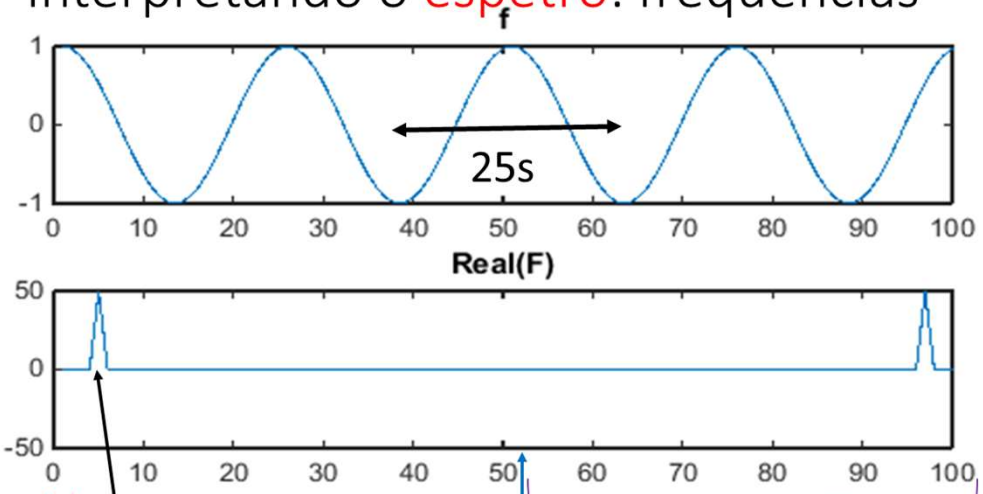
$$S(f) = \mathcal{F}\big(s(t)\big) \Longrightarrow \mathcal{F}\big(s(t-a)\big) = e^{-ifa}S(f)$$

Escalamento

$$S = \mathcal{F}(s(t)) \Longrightarrow \mathcal{F}\big(s(at)\big) = \frac{1}{a}S\left(\frac{f}{a}\right)$$

Interpretando o espetro: frequências

$$= \sum_{k=-N/2}^{N/2} c_k e^{i2\pi kt/T}$$



$$f(n) = \cos\frac{2\pi nt}{25}$$

$$n = 0 \ldots N-1$$

$$N = 100$$

n>0

n=51=(N/2+1)

n<0

$$c_0 = \frac{a_0}{2},\ c_n = \frac{a_n - ib_n}{2},\ c_{-n} = \frac{a_n + ib_n}{2}$$

Média (frequência=0)

$$f = \frac{5-1}{50} f_{Nyq} = \frac{1}{25} s^{-1} f = \frac{1}{2\Delta t} = f_{Nyquist}$$

6

Transformada discreta de Fourier

$$f_n = \frac{1}{N}\sum_{k=0}^{N-1} F_k e^{2\pi i n k/N}$$

$\{f_n\}$ e $\{F_k\}$ têm o mesmo número de termos (N)

F_0 é proporcional à média da série

F_k, F_{-k} representam a harmónica k (frequência $\frac{f_{Nyq}k}{\frac{N}{2}}$)

indicando que existe um número ímpar de termos em F_k. Mas só são calculados N termos e N pode ser par. Se for esse o caso não é calculado o termo correspondente a $-f_{Nyq}$.

Em geral,

Tanto série $\{f_n\}$ com a sua transformada de Fourier $\{F_k\}$ são **séries complexas**.

Mesmo que $\{f_n\}$ seja real, $\{F_k\}$ é complexa.

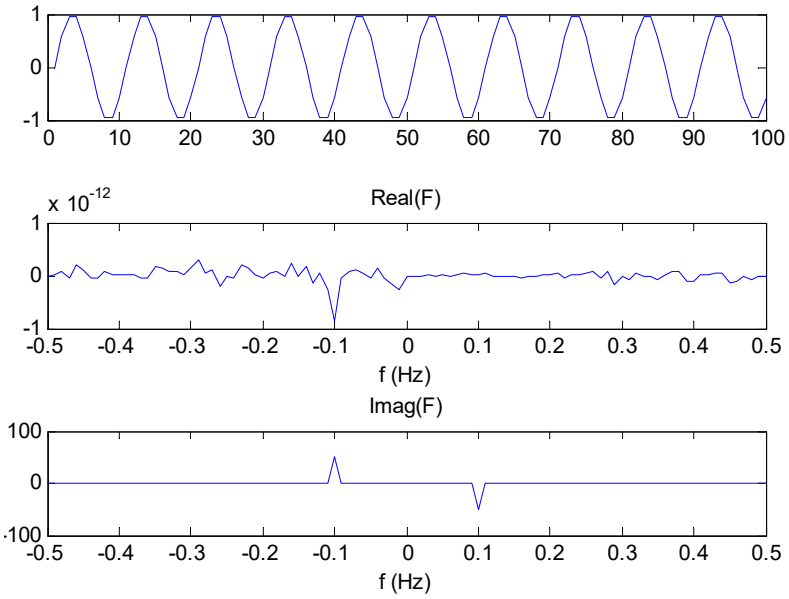
Casos especiais, se $\{f_n\}$ for real e

Par $f_n = f_{-n}$: $\{F_k\}$ é real e simétrica

Ímpar $f_n = -f_{-n}$: $\{F_k\}$ é imaginária e anti-simétrica

Casos especiais

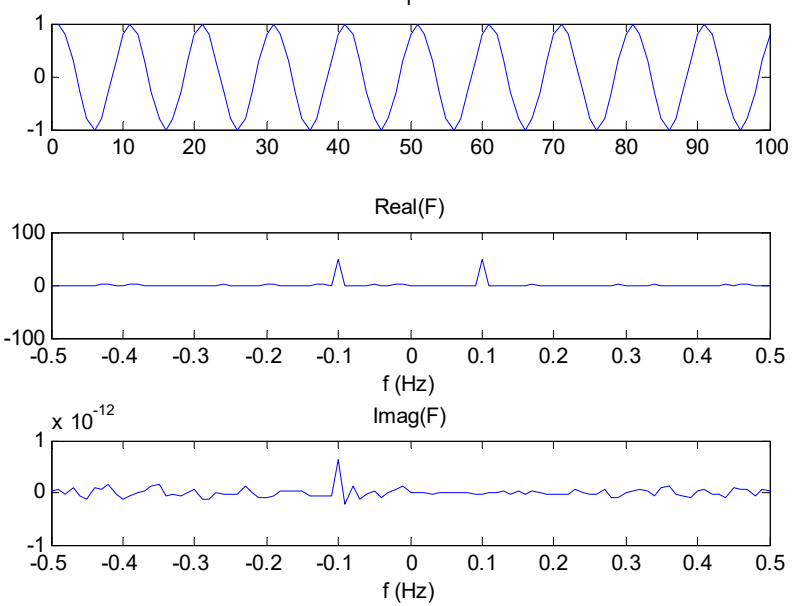
$\sin(x) = -\sin(-x)$



Transformada imaginária, anti-simétrica

$(c_k = ir_k = -ir_{-k})$

$\cos(x) = cos(-x)$



Transformada real, simétrica

$(c_k = r_k = r_{-k})$

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.

Ciências ULisboa

Modelação Numérica
Aula 6

Filtros de Fourier