



Ciências
ULisboa

Modelação Numérica

Aula 4

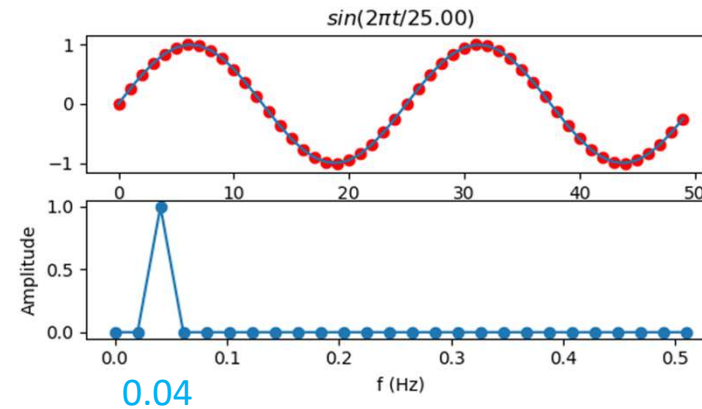
Gestão de falhas em dados, espectrogramas, filtros de média móvel

Problema do domínio

Exatamente dois períodos
fundamentais.

Espectro OK

Pico em $0.04\text{Hz}=1/T$

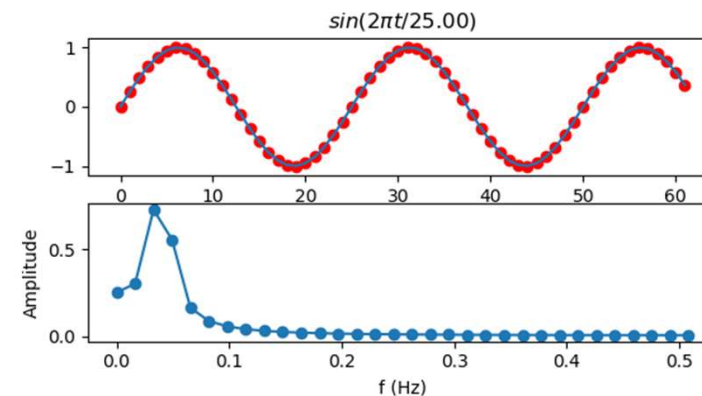


2.5 períodos fundamentais.

Espectro poluído

Mas, em ambos os casos

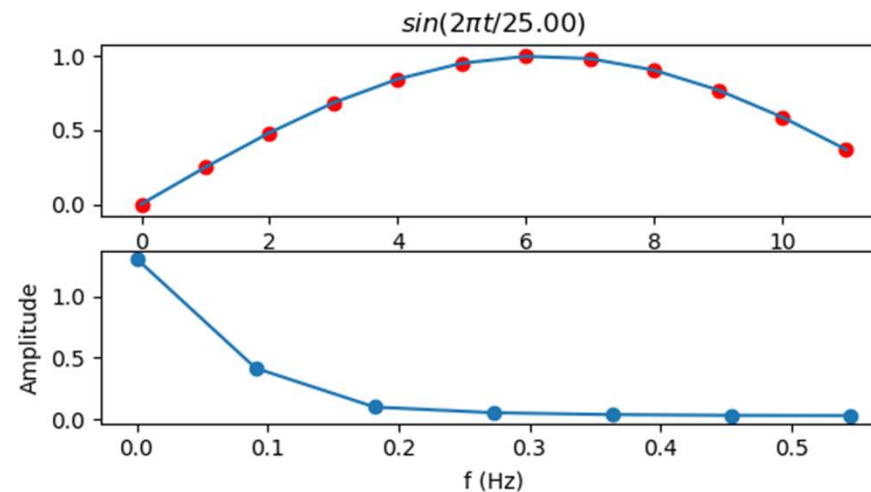
$$\mathcal{F}^{-1}(\mathcal{F}(f)) = f$$



Domínio inferior ao período fundamental

Mas,

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



Pico em 0Hz (média não nula)

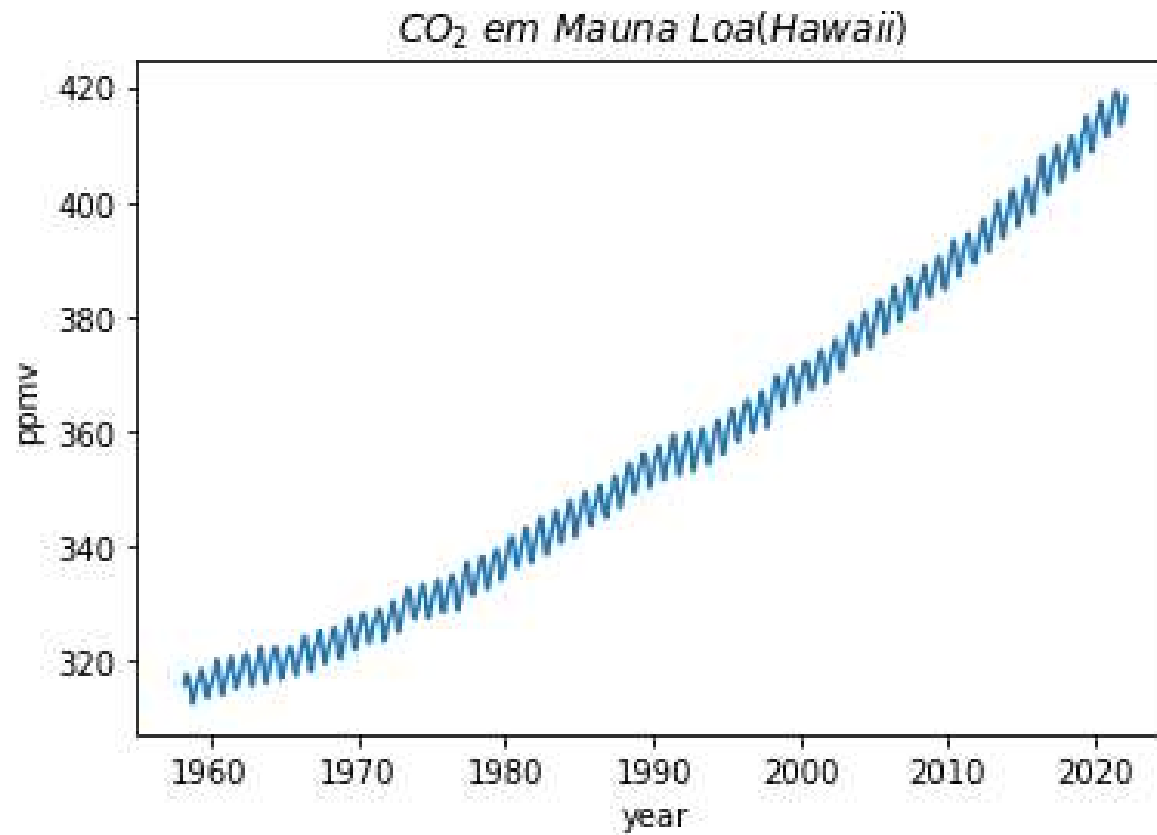
Com dados reais

Existem muitas harmónicas e podemos não saber qual a **harmónica fundamental**.

É preciso ter bom senso e ensaiar **diferentes aproximações**.

Por exemplo, se se tratar de uma série climática (em que devem existir ciclos anuais e diurnos) deve analisar-se um período que corresponda a anos completos, ou a dias completos.

E agora, um caso real



2 ficheiros NOAA

MaunaLoa_Week_NOAA - Notepad

<https://gml.noaa.gov/>

#	Start of week			CO2 molfrac	(-999.99 = no data)		increase	
#	(yr,	mon,	day,	(ppm)	#days	1 yr ago	10 yr ago	since 1800
1974	5	19	1974.3795	333.37	5	-999.99	-999.99	50.40
1974	5	26	1974.3986	332.95	6	-999.99	-999.99	50.06
1974	6	2	1974.4178	332.35	5	-999.99	-999.99	49.60
1974	6	9	1974.4370	332.20	7	-999.99	-999.99	49.65
1974	6	16	1974.4562	332.37	7	-999.99	-999.99	50.06
1974	6	23	1974.4753	331.73	5	-999.99	-999.99	49.72
1974	6	30	1974.4945	331.68	6	-999.99	-999.99	50.02
1974	7	7	1974.5137	331.46	6	-999.99	-999.99	50.20
1974	7	14	1974.5329	330.83	5	-999.99	-999.99	50.01
1974	7	21	1974.5521	330.76	7	-999.99	-999.99	50.41
1974	7	28	1974.5712	329.80	4	-999.99	-999.99	49.97
1974	8	4	1974.5904	329.85	5	-999.99	-999.99	50.54
1974	8	11	1974.6096	329.15	5	-999.99	-999.99	50.37
1974	8	18	1974.6288	329.06	6	-999.99	-999.99	50.80
1974	8	25	1974.6479	328.33	7	-999.99	-999.99	50.54
1974	9	1	1974.6671	328.06	5	-999.99	-999.99	50.68
1974	9	8	1974.6863	327.56	4	-999.99	-999.99	50.53
1974	9	15	1974.7055	326.72	6	-999.99	-999.99	49.95
1974	9	22	1974.7247	326.99	5	-999.99	-999.99	50.37

MaunaLoa_NOAA - Notepad

File	Edit	View					
1958	3	1958.2027	315.70	314.43	-1	-9.99	-0
1958	4	1958.2877	317.45	315.16	-1	-9.99	-0
1958	5	1958.3699	317.51	314.71	-1	-9.99	-0
1958	6	1958.4548	317.24	315.14	-1	-9.99	-0
1958	7	1958.5370	315.86	315.18	-1	-9.99	-0
1958	8	1958.6219	314.93	316.18	-1	-9.99	-0
1958	9	1958.7068	313.20	316.08	-1	-9.99	-0
1958	10	1958.7890	312.43	315.41	-1	-9.99	-0
1958	11	1958.8740	313.33	315.20	-1	-9.99	-0
1958	12	1958.9562	314.67	315.43	-1	-9.99	-0
1959	1	1959.0411	315.58	315.55	-1	-9.99	-0
1959	2	1959.1260	316.48	315.86	-1	-9.99	-0
1959	3	1959.2027	316.65	315.38	-1	-9.99	-0
1959	4	1959.2877	317.72	315.41	-1	-9.99	-0
1959	5	1959.3699	318.29	315.49	-1	-9.99	-0
1959	6	1959.4548	318.15	316.03	-1	-9.99	-0
1959	7	1959.5370	316.54	315.86	-1	-9.99	-0
1959	8	1959.6219	314.80	316.06	-1	-9.99	-0
1959	9	1959.7068	313.84	316.73	-1	-9.99	-0
1959	10	1959.7890	313.33	316.33	-1	-9.99	-0
1959	11	1959.8740	314.81	316.68	-1	-9.99	-0
1959	12	1959.9562	315.58	316.35	-1	-9.99	-0
1960	1	1960.0410	316.43	316.40	-1	-9.99	-0
1960	2	1960.1257	316.98	316.36	-1	-9.99	-0
1960	3	1960.2049	317.58	316.28	-1	-9.99	-0
1960	4	1960.2896	319.03	316.70	-1	-9.99	-0
1960	5	1960.3716	320.04	317.22	-1	-9.99	-0
1960	6	1960.4563	319.59	317.47	-1	-9.99	-0
1960	7	1960.5383	318.18	317.52	-1	-9.99	-0
1960	8	1960.6230	315.90	317.19	-1	-9.99	-0
1960	9	1960.7077	314.17	317.08	-1	-9.99	-0
1960	10	1960.7896	313.83	316.83	-1	-9.99	-0
1960	11	1960.8743	315.00	316.88	-1	-9.99	-0

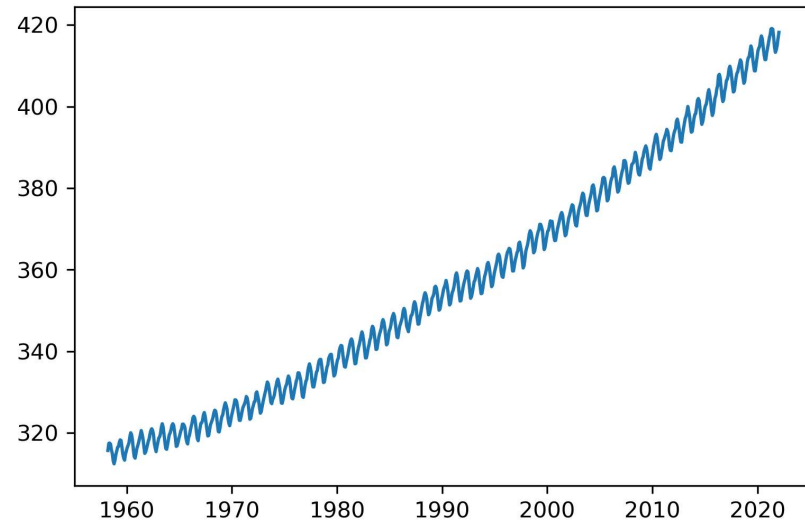
Ln 1, Col 1

50

Dados mensais

```
import numpy as np
import matplotlib.pyplot as plt

ML=np.loadtxt('MaunaLoa_NOAA.txt')
year=ML[:,2]
co2=ML[:,3]
plt.plot(year,co2)
```



```

ML=np.loadtxt('MaunaLoa_NOAA.txt')
t=ML[:,2]
co2=ML[:,3]
dt=1/12. #ano

```

```

fig,ax=plt.subplots(nrows=2)
N=len(co2)
F=fft.fft(co2)

```

```

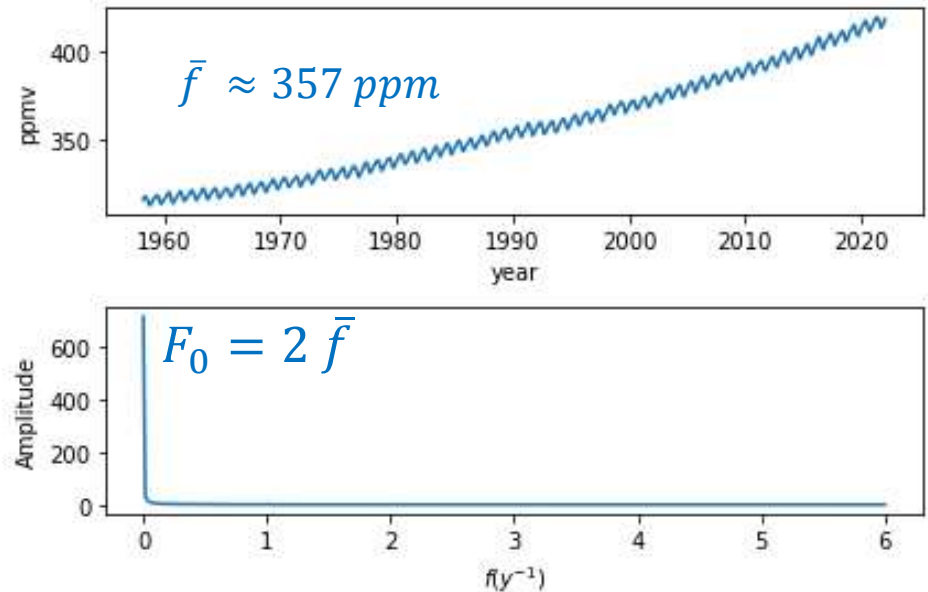
ax[0].plot(t,co2)
ax[0].set_ylabel('ppmv');ax[0].set_xlabel('year')

```

```

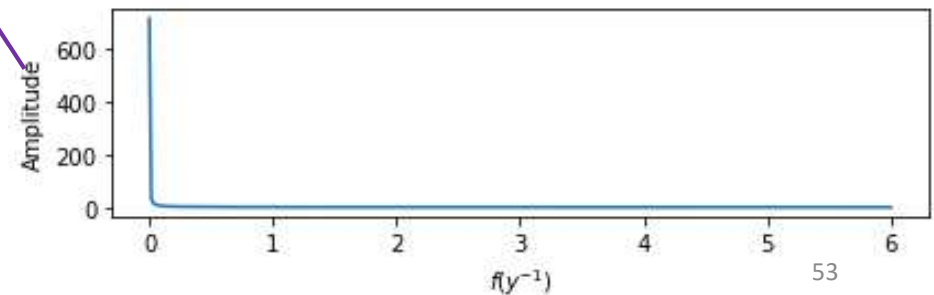
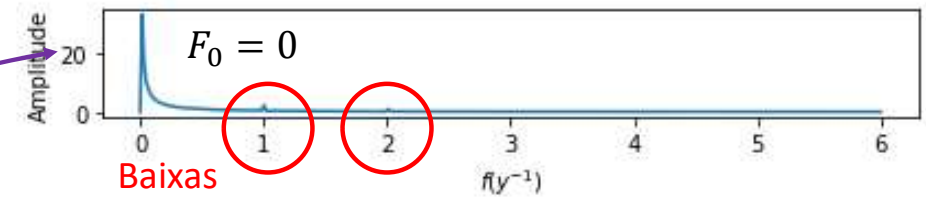
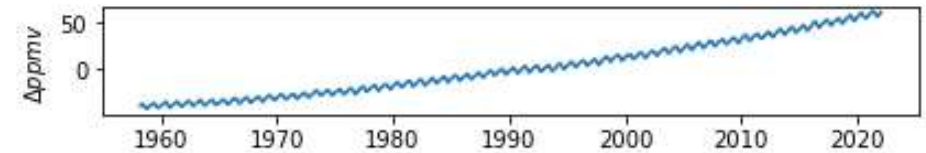
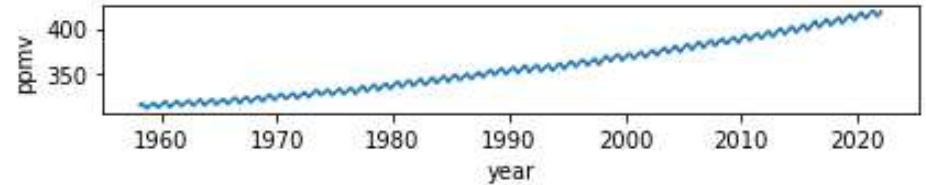
fNyq=1/(2*dt); df=2*fNyq/(N-1)
freq=np.arange(0,fNyq+df,df)
ax[1].plot(freq,np.abs(F[0:N//2+1])/(N//2))
ax[1].set_ylabel('Amplitude');ax[1].set_xlabel(r'$f$ (y-1)$')
fig.tight_layout()

```



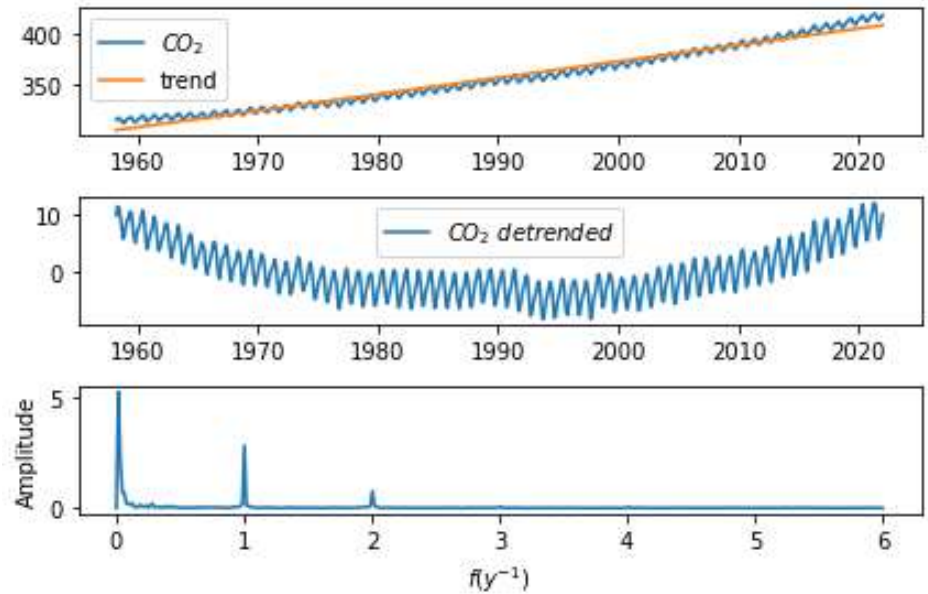
Retirando a média

```
fig, ax=plt.subplots(nrows=3)
f=co2-np.mean(co2)
F=fft.fft(f)
ax[0].plot(t,co2)
ax[0].set_ylabel('ppmv')
ax[0].set_xlabel('year')
ax[1].plot(t,f)
ax[1].set_ylabel(r'$\Delta$ ppmv$')
fNyq=1/(2*dt); df=2*fNyq/(N-1)
freq=np.arange(0,fNyq+df,df)
ax[2].plot(freq,np.abs(F[0:N//2+1])/(N//2))
ax[2].set_ylabel('Amplitude');ax[2].set_xlabel(r'$f$ (y$^{-1}$)')
fig.tight_layout()
fig.show()
```



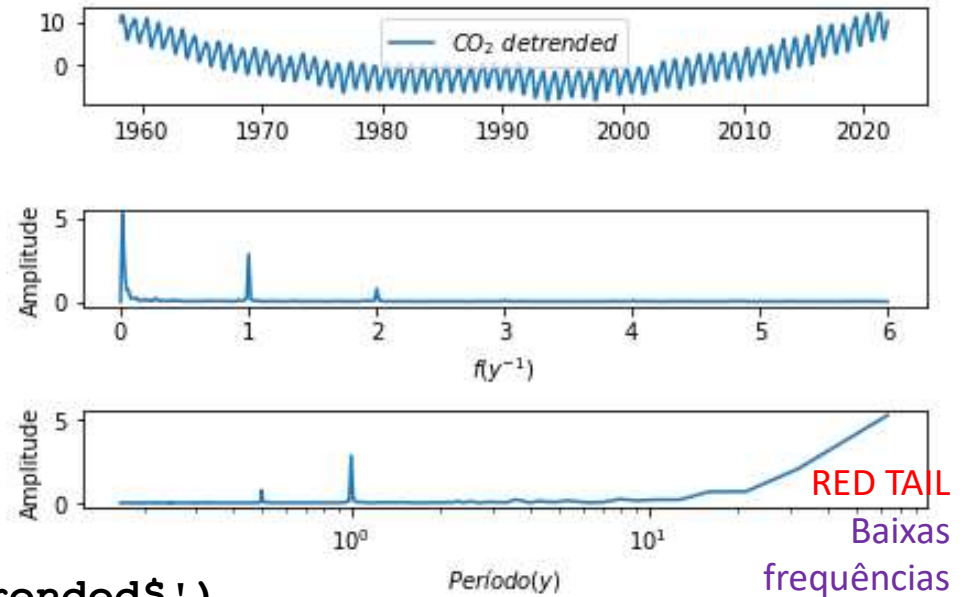
Retirando a tendência

```
pol=np.polyfit(t,co2,1)
co2trend=pol[1]+pol[0]*t
fig,ax=plt.subplots(nrows=3)
ax[0].plot(t,co2,label=r'$CO_2$')
ax[0].plot(t,co2trend,label='trend')
ax[0].legend()
f=co2-co2trend
ax[1].plot(t,f,label=r'$CO_2$ detrended$')
ax[1].legend()
F=fft.fft(f)
ax[2].plot(freq,np.abs(F[0:N//2+1])/(N//2))
ax[2].set_ylabel('Amplitude');ax[2].set_xlabel(r'$f$ (y-1)$')
plt.tight_layout()
```



De novo mas mudando o gráfico

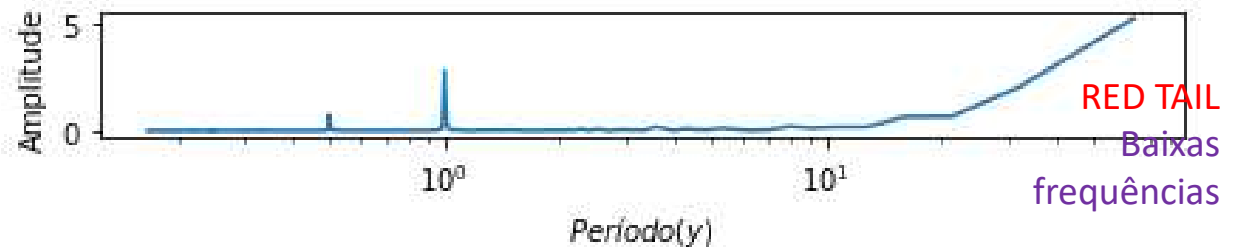
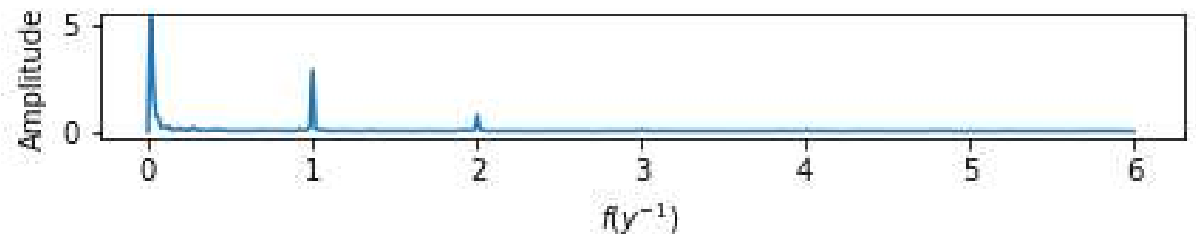
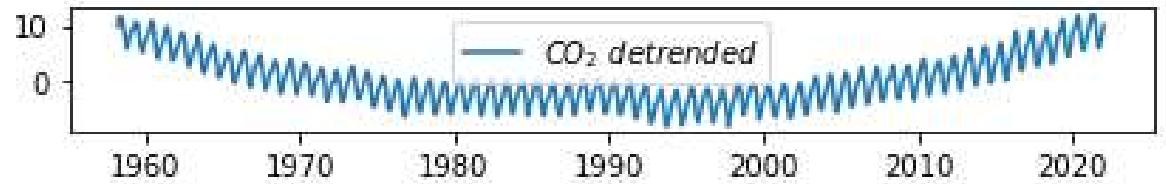
```
fig,ax=plt.subplots(nrows=3)
f=co2-co2trend
ax[0].plot(t,f,label=r'$CO_2$ \ detrended$')
ax[0].legend()
F=fft.fft(f)
ax[1].plot(freq,np.abs(F[0:N//2+1])/(N//2))
ax[1].set_ylabel('Amplitude')
ax[1].set_xlabel(r'$f$ (y-1)$')
ax[2].plot(1/freq,np.abs(F[0:N//2+1])/(N//2))
ax[2].set_xscale('log')
ax[2].set_ylabel('Amplitude');ax[2].set_xlabel(r'$Período$ (y)$')
plt.tight_layout()
```



A subtração da média impõe $F[0] = 0$.

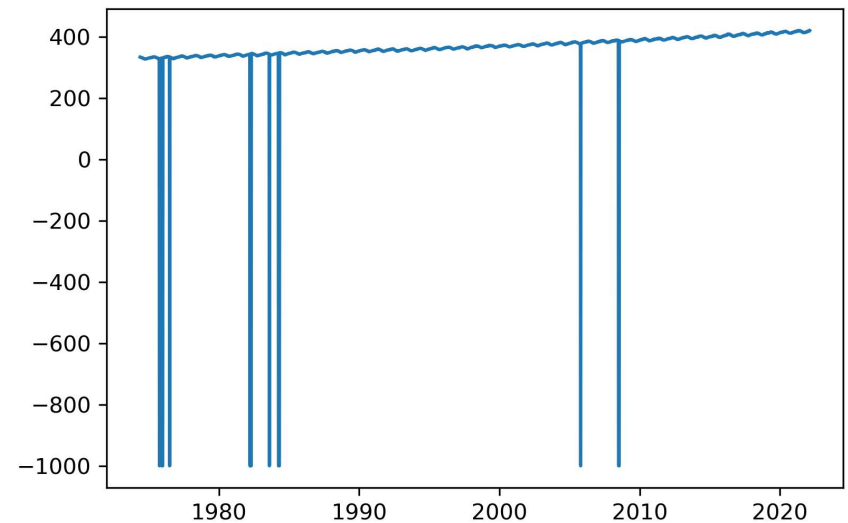
A subtração da linha de tendência impõe $F[0] = 0$ e modifica as muito baixas frequências mas não as elimina.

Precisamos de utilizar um **filtro passa-alto**.



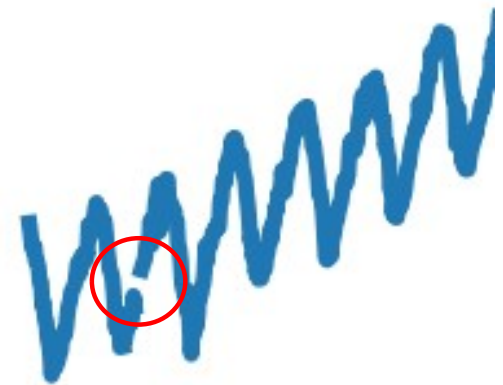
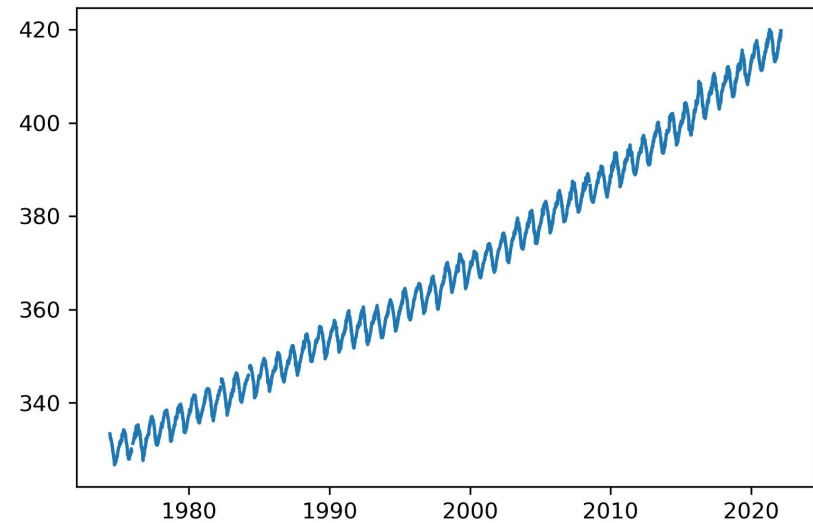
Antes disso, dados semanais (falha=-999)

```
ML=np.loadtxt('MaunaLoa_Week_NOAA.txt')
year=ML[:,3]
co2=ML[:,4]
plt.plot(year,co2)
```



nan nas falhas (só para plt)

```
ML=np.loadtxt('MaunaLoa_Week_NOAA.txt')
year=ML[:,3]
co2=ML[:,4]
N=len(co2)
for k in range(N):
    if co2[k]<0:
        co2[k]=np.float('nan')
plt.plot(year,co2)
```



A utilização de nan “só” funciona nos gráficos... Preencher falhas

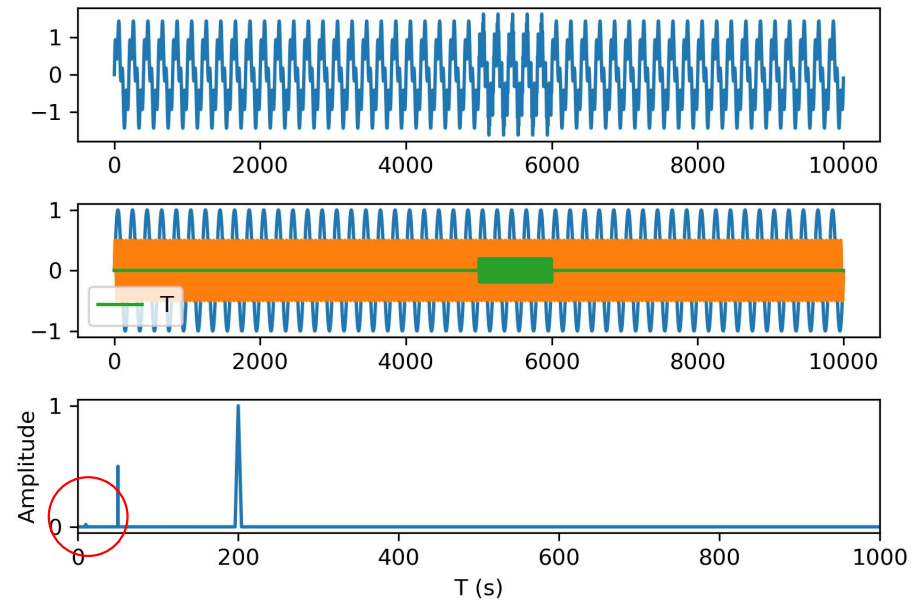
```
ML=np.loadtxt('MaunaLoa_Week_NOAA.txt')
dt=7/365.25
year=ML[:,3];co2=ML[:,4];N=len(co2)
co2V=[];yearV=[]
for k in range(N):
    if co2[k]>0:
        co2V.append(co2[k])
        yearV.append(year[k])
co2V=np.array(co2V) #dados válidos
yearV=np.array(yearV)
co2I=np.interp(year,yearV,co2V)
```

Sinais **transientes** e espectrogramas

Sinais de curta duração podem ser impossíveis de detetar quando se analisa espectro.

No exemplo sintético estão sobrepostas 3 sinusoides, uma delas só durante uma janela temporal. No espectro de amplitude o sinal transiente ($T=10s$) é muito fraco e estaria ausente se existir um pequeno nível de ruído.

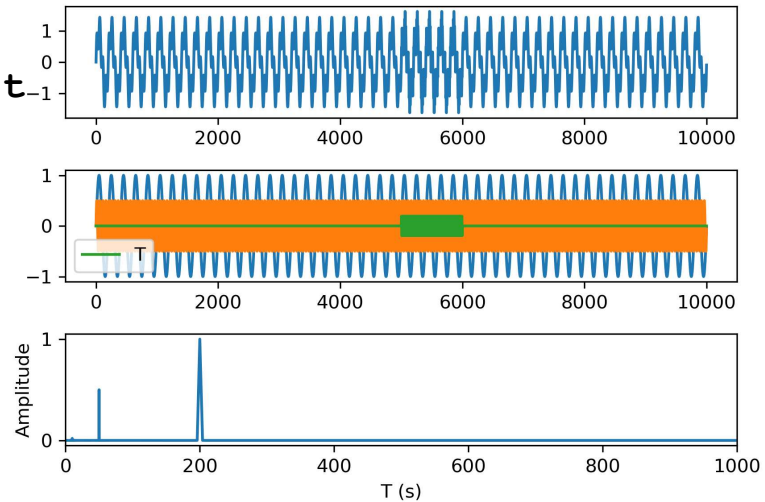
Se analisarmos por janelas podemos dar relevância a estes sinais transientes.




```

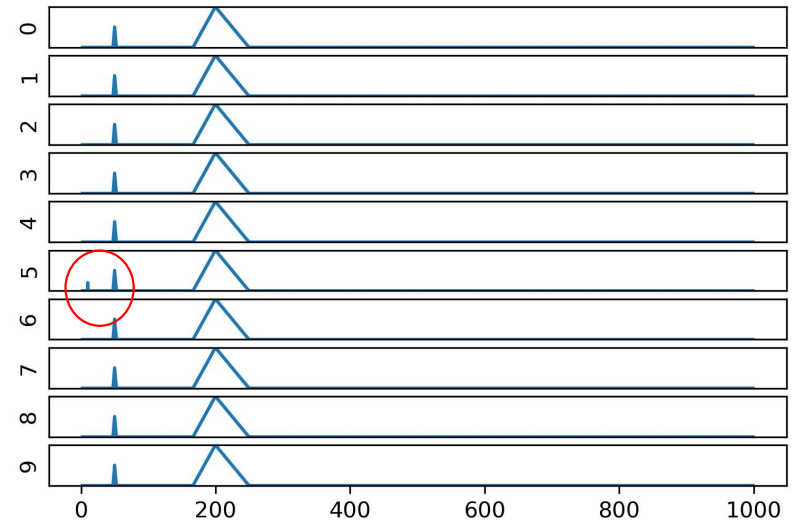
import numpy as np;import matplotlib.pyplot as plt
from scipy import fft
N=10000;dt=1; t=np.linspace(0,dt*(N-1),N)
T=[200,50];A=[1,0.5]
TT=10;AT=0.2;kT0=5000;kT1=6000;
fT=AT*np.sin(2*np.pi*t/TT);fT[0:kT0]=0;fT[kT1:]=0
f1=A[0]*np.sin(2*np.pi*t/T[0]);
f2=A[1]*np.sin(2*np.pi*t/T[1])
f=f1+f2+fT
fig,ax=plt.subplots(nrows=3)
ax[0].plot(t,f)
fNyq=1/(2*dt);df=2*fNyq/(N-1);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')
ax[1].legend()
F=fft.fft(f)
ax[2].plot(1/freq,np.abs(F[0:N//2+1])/(N//2))
ax[2].set_xlim(0,1000)
ax[2].set_ylabel('Amplitude');plt.xlabel('T (s)')

```



Espectrograma

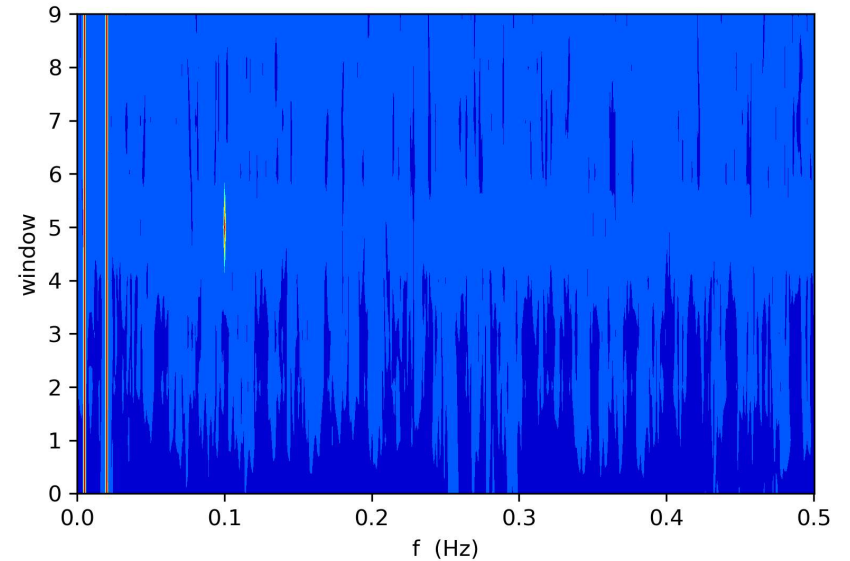
```
fig,ax=plt.subplots(nrows=10)
window=1000
kw=-1
for k in range(0,N,window):
    kw+=1
    fw=f[k:k+window]
    Nw=len(fw)
    fNyq=1/(2*dt); df=2*fNyq/(Nw-1)
    freq=np.arange(0,fNyq+df,df)
    Fw=fft.fft(fw)
    ax[kw].plot(1/freq,np.abs(Fw[0:Nw//2+1])/(Nw//2))
    ax[kw].set_ylim(0,1)
    ax[kw].set_yticks([])
    if kw<9:
        ax[kw].set_xticks([])
    ax[kw].set_ylabel(str(k//1000))
```



```

#versão 2
window=1000
FF=np.zeros((window//2+1,N//window))
xis=np.copy(FF);yps=np.copy(FF)
kw=-1
for k in range(0,N,window):
    kw+=1
    fw=f[k:k+window];Nw=len(fw)
    fNyq=1/(2*dt); df=2*fNyq/(Nw-1)
    freq=np.arange(0,fNyq+df,df)
    xis[:,kw]=freq;yps[:,kw]=kw
    Fw=fft.fft(fw)
    FF[:,kw]=np.abs(Fw[0:Nw//2+1])/(Nw//2)
plt.figure()
plt.contourf(xis,yps,np.log(FF),cmap='jet')
plt.xlabel('f (Hz)')
plt.ylabel('window')

```



Filtro digital



Sinal e ruído

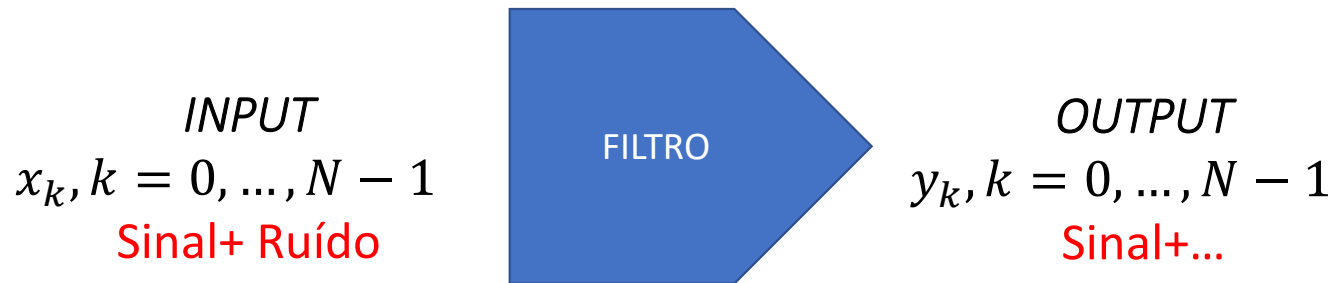
As séries de dados reais contém em geral informação proveniente de diferentes **processos**.

Dependendo do objetivo do analista só parte dessa informação é relevante, constituindo o **sinal**, sendo as outras componentes designadas por **ruído**.

Separar o sinal do ruído é o objetivo dos **filtros**.

No caso dos **filtros digitais** a separação é feita de forma algébrica, tipicamente por operações lineares sobre os dados.

Filtro



Filtro linear **não recursivo** (o output depende linearmente do input)

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

convolução

Coefficientes do filtro $h_j, j = 0, \dots, J - 1$

Filtro recursivo

O output num dado instante depende do input, e também do próprio output em instantes anteriores:

$$y_k = \sum_{n=0}^N x_n h_{k-n} + \sum_{m=0}^{k-1} y_m b_{k-m}$$

Depende de dois conjuntos de coeficientes (h, b)

Média móvel

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

Se:

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

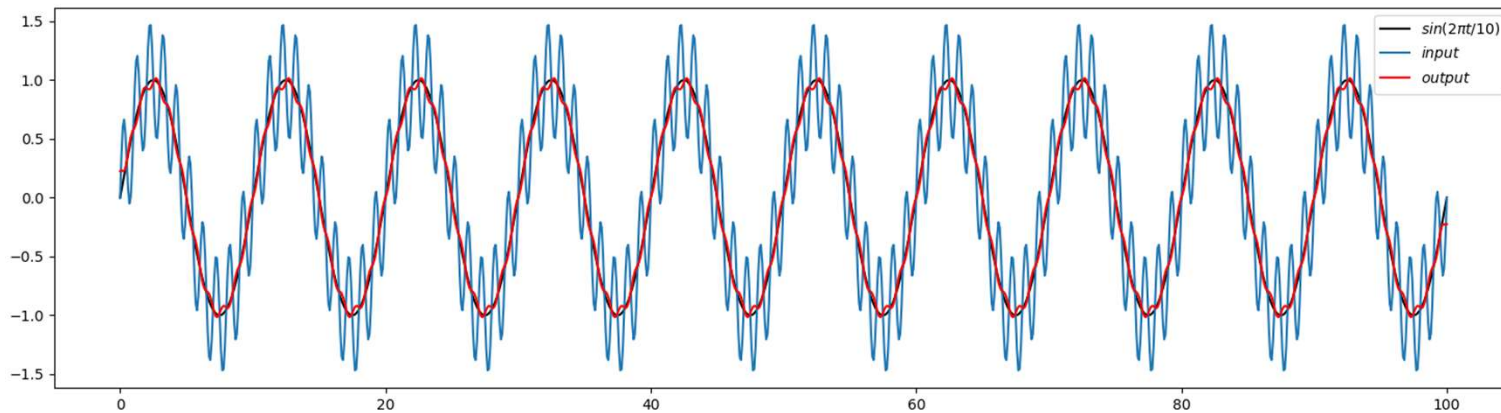
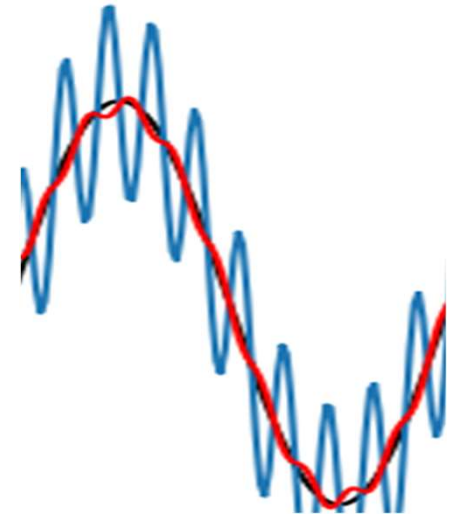
Trata-se de um filtro de média móvel. Exemplo:

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

Média móvel

```
import numpy as np; import matplotlib.pyplot as plt
N=1001;dt=0.1;T=10.
t=np.arange(0.,(N-1)*dt+dt,dt)
x1=np.sin(2*np.pi*t/T) # "sinal"
x2=0.5*np.sin(2*np.pi*t/(T/10)) # "ruído"
x=x1+x2 # série a filtrar input
plt.plot(t,x1,color='black',label=r"$\sin(2\pi t/10)$")
plt.plot(t,x,label=r"$input$")
nn=11 # comprimento do filtro
h=np.ones((nn))/nn
y=np.convolve(x,h,mode='same') # output
plt.plot(t,y,color='red',label=r"$output$")
plt.legend()
```

```
plt.plot(t,x1,color='black',\
         label=r"$\sin(2\pi t/10)$")
plt.plot(t,x,label=r"$input$")
plt.plot(t,y,color='red',label=r"$output$")
```



A media móvel aplicada...

Diminuiu a amplitude das altas frequências.

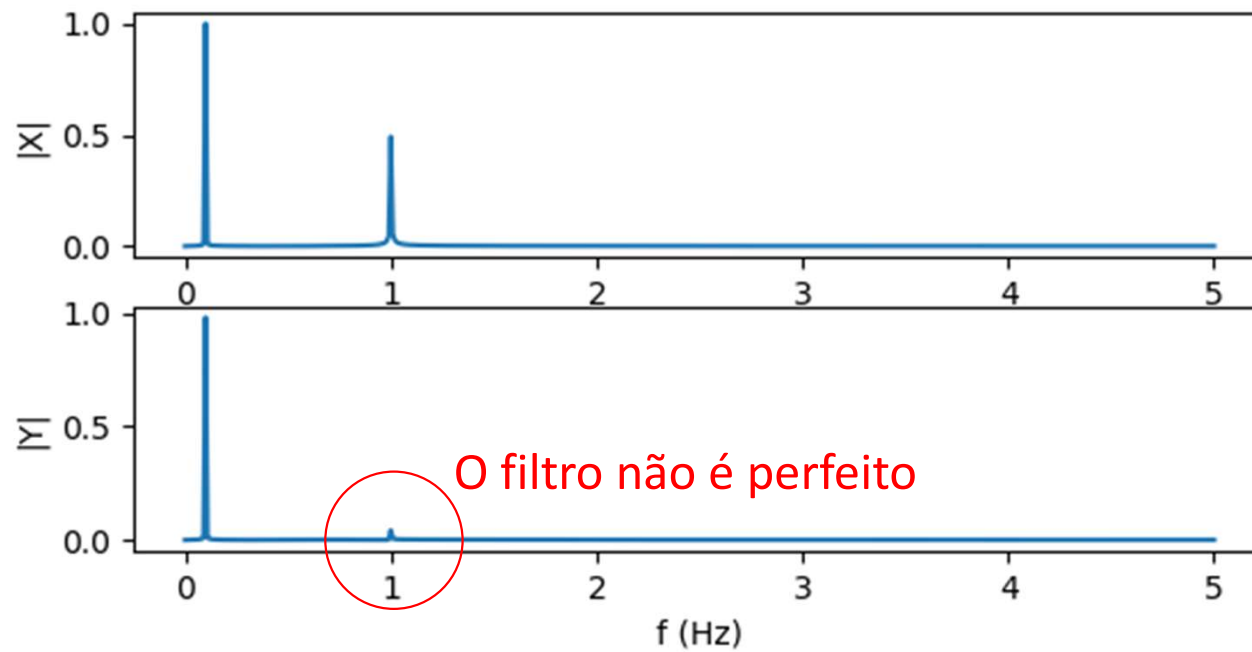
Constitui um filtro **passa-baixo**: deixa passar as baixas frequências.

```
x1=np.sin(2*np.pi*t/T) # "sinal" (100 pontos/T)
x2=0.5*np.sin(2*np.pi*t/(T/10)) # "ruído" (10 pontos/T=)
x=x1+x2 # série a filtrar input
nn=11 # comprimento do filtro
h=np.ones((nn))/nn
y=np.convolve(x,h,mode='same') # output
```

Com a mesma série e filtro

```
plt.figure()
X=np.fft.fft(x) #Espectro do input
Y=np.fft.fft(y) #Espectro do output
fNyq=1/(2*dt)
df=2*fNyq/(N-1)
freq=np.arange(0,fNyq+df,df)
plt.subplot(3,1,1)
plt.plot(freq,np.abs(X[0:N//2+1])/(N//2),label='X')
plt.ylabel('|X|') #espectro de amplitude de x
plt.subplot(3,1,2)
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)')
```

```
x1=np.sin(2*np.pi*t/T) #"sinal"  
x2=0.5*np.sin(2*np.pi*t/(T/10)) #"ruído"  
x=x1+x2  
y=np.convolve(x,h,mode='same')
```

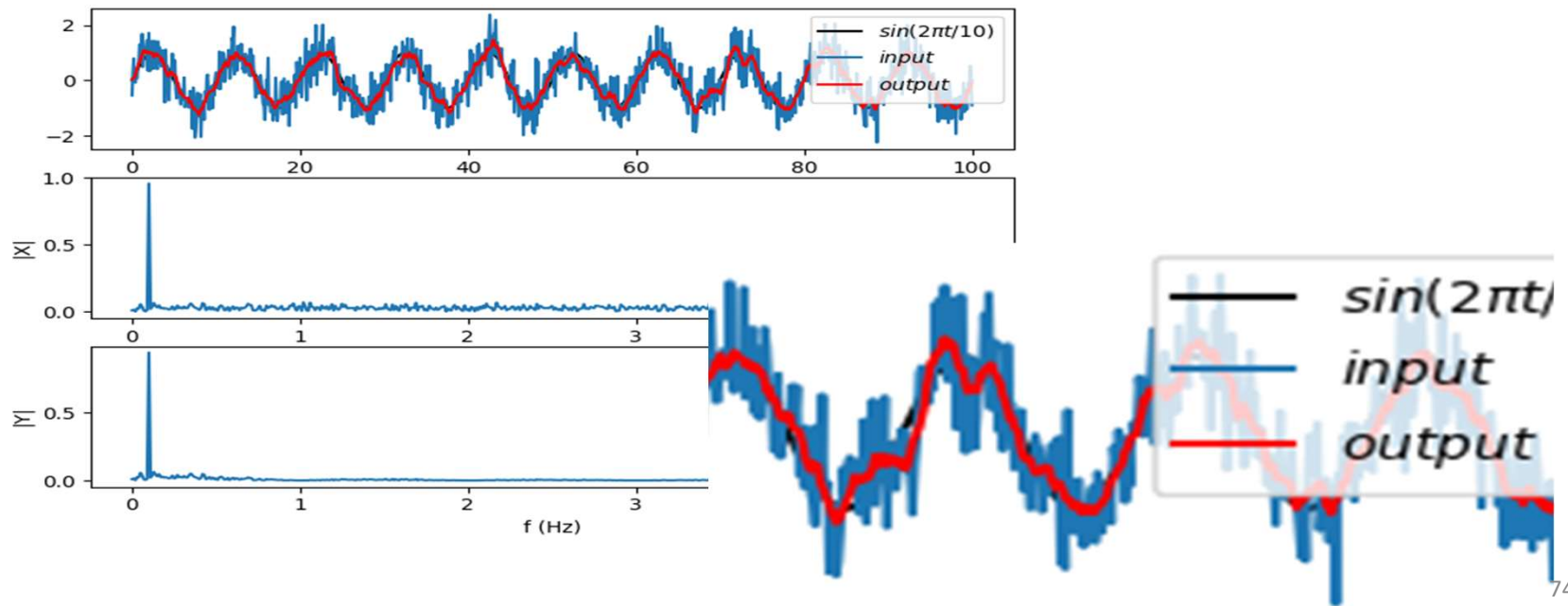


É claro que o ruído não costuma ser um seno de alta frequência...

```
x1=np.sin(2*np.pi*t/T)
```

```
r=0.5*np.random.randn(t.shape[0])
```

```
x=x1+r
```



Efeitos de fronteira

`y=np.convolve(x,h,mode='same')`
'same' prolonga a série x
Produzindo y com os mesmos pontos

Série com n pontos, filtro com J pontos

$$\{x_k\} = x_0, x_1, \dots, x_{N-1}$$

$$\{h_j\} = h_0, h_1, \dots, h_{J-1}$$

Filtragem (convolução):

$$y_k = \sum_{n=0}^N x_n h_{k-n}, (k-n) \in [0, J-1]$$

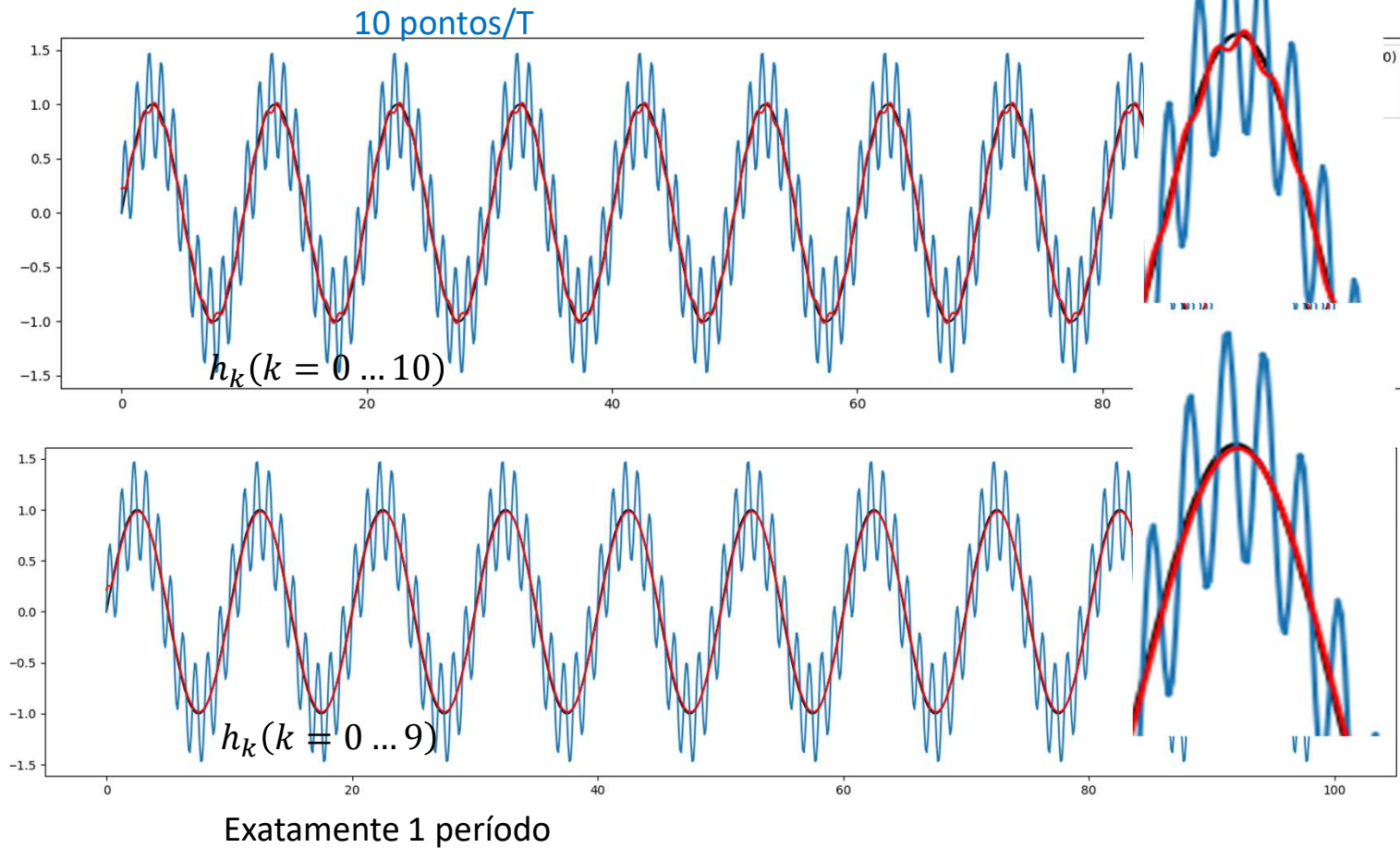
Exemplo $J = 10$:

$$y_9 = x_9 h_0 + x_8 h_1 + \dots + x_0 h_9$$

$$y_8 = x_8 h_0 + x_7 h_1 + \dots + x_1 h_7 + x_0 h_8 + x_{-1} h_9$$

Não existe

O que faz a media móvel ao espectro?



Teorema da convolução

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

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

Notar que Y, X, H são **complexos**.

H é designada por **função de transferência** do filtro h

Série sintética com 4 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)
```

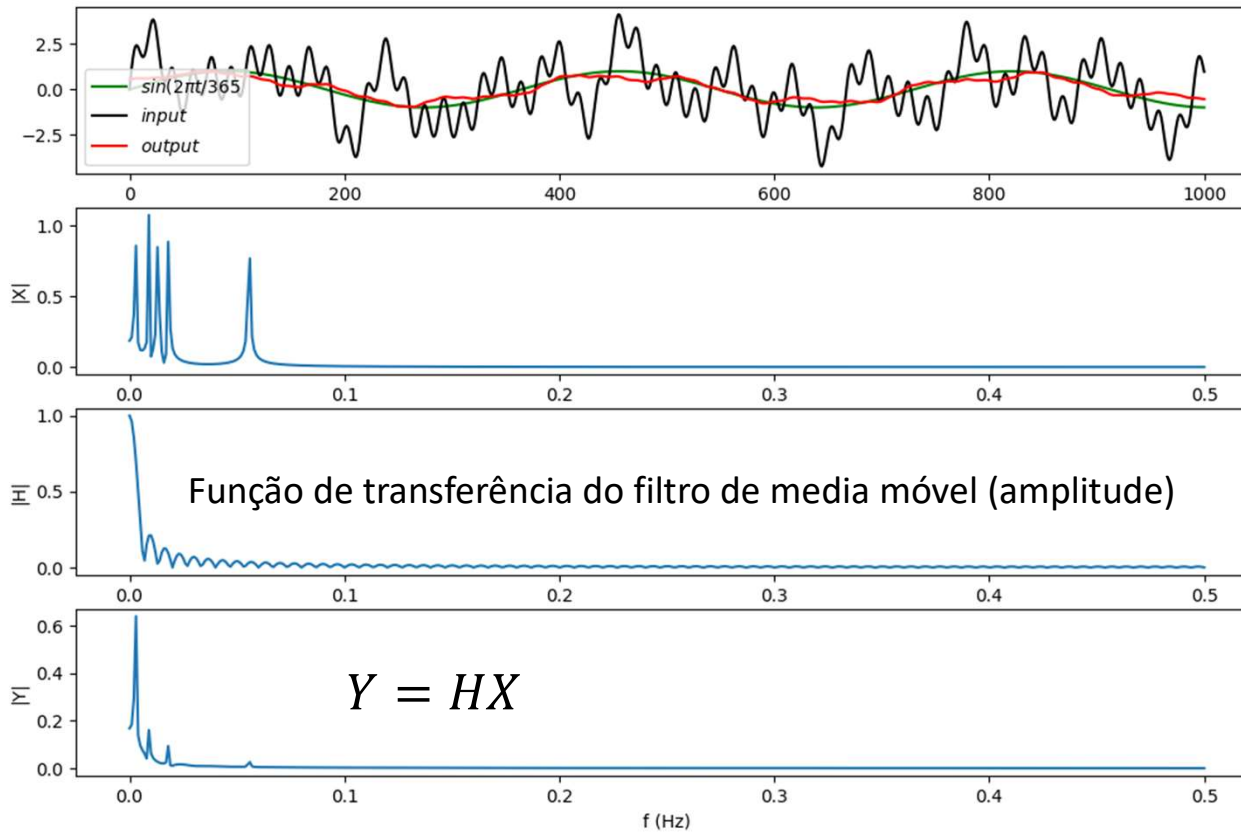
```

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)

