Modelação Numérica 2017

Aula 9, 15/Mar

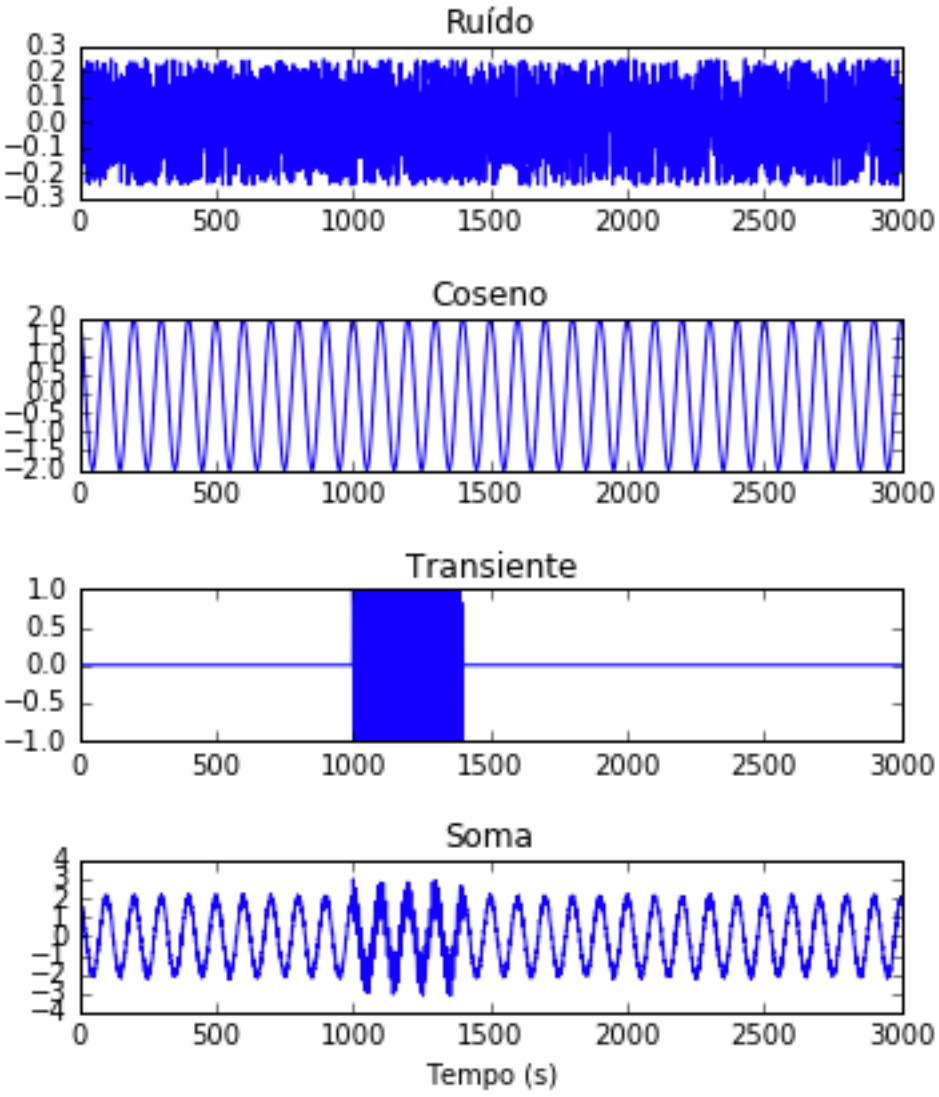
- Tendências e filtros
- Correlação cruzada

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

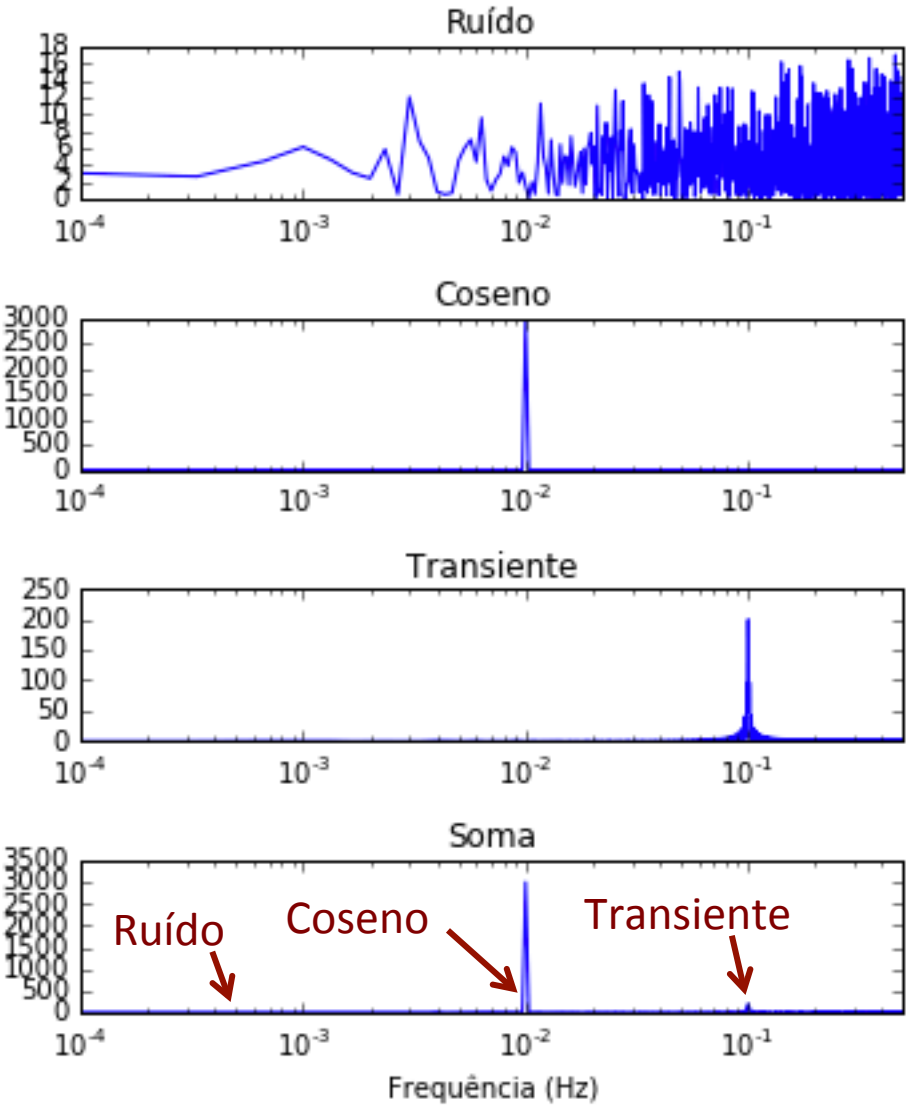
Aula passada

Espectrogramas

Domínio do tempo



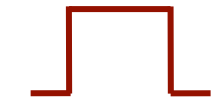
Domínio espectral

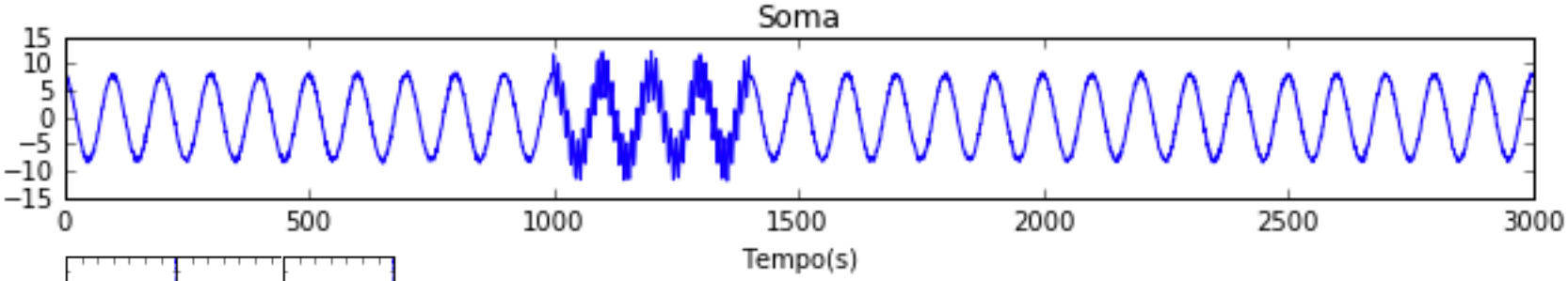


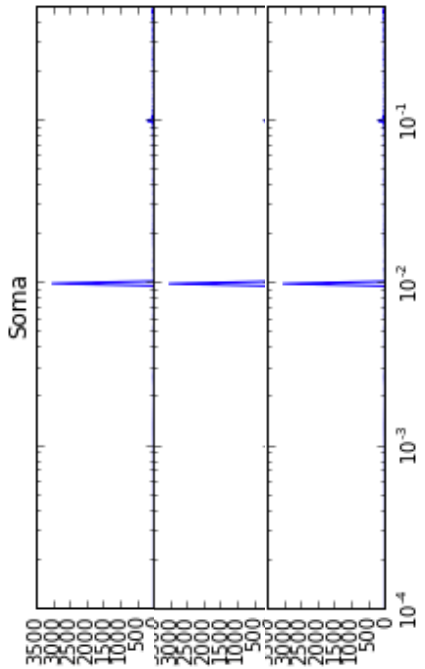
Aula passada

Espectrograma

Janela móvel (NJ = 200)



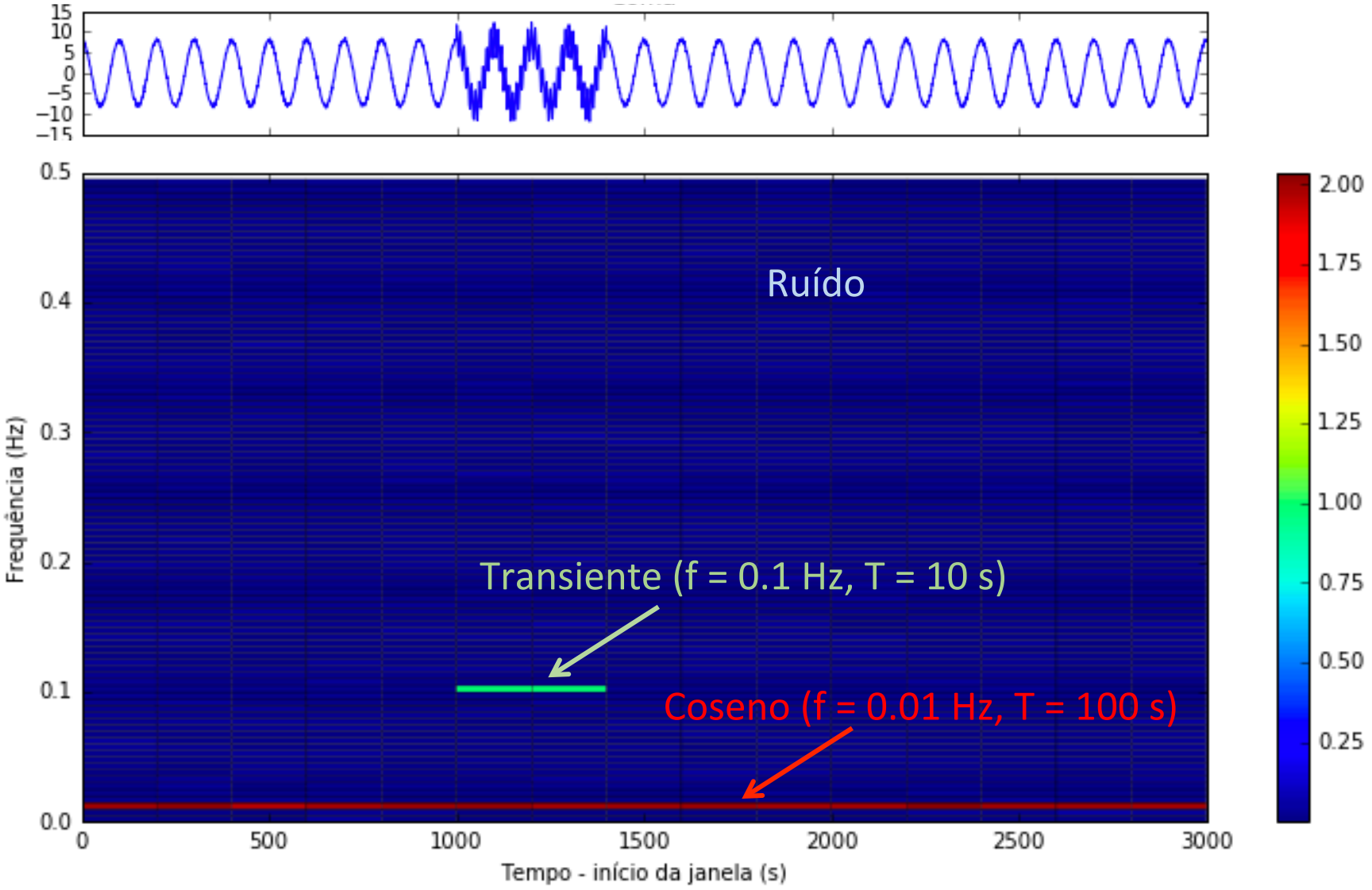




… => Para cada janela, calculamos um espectro.

Aula passada

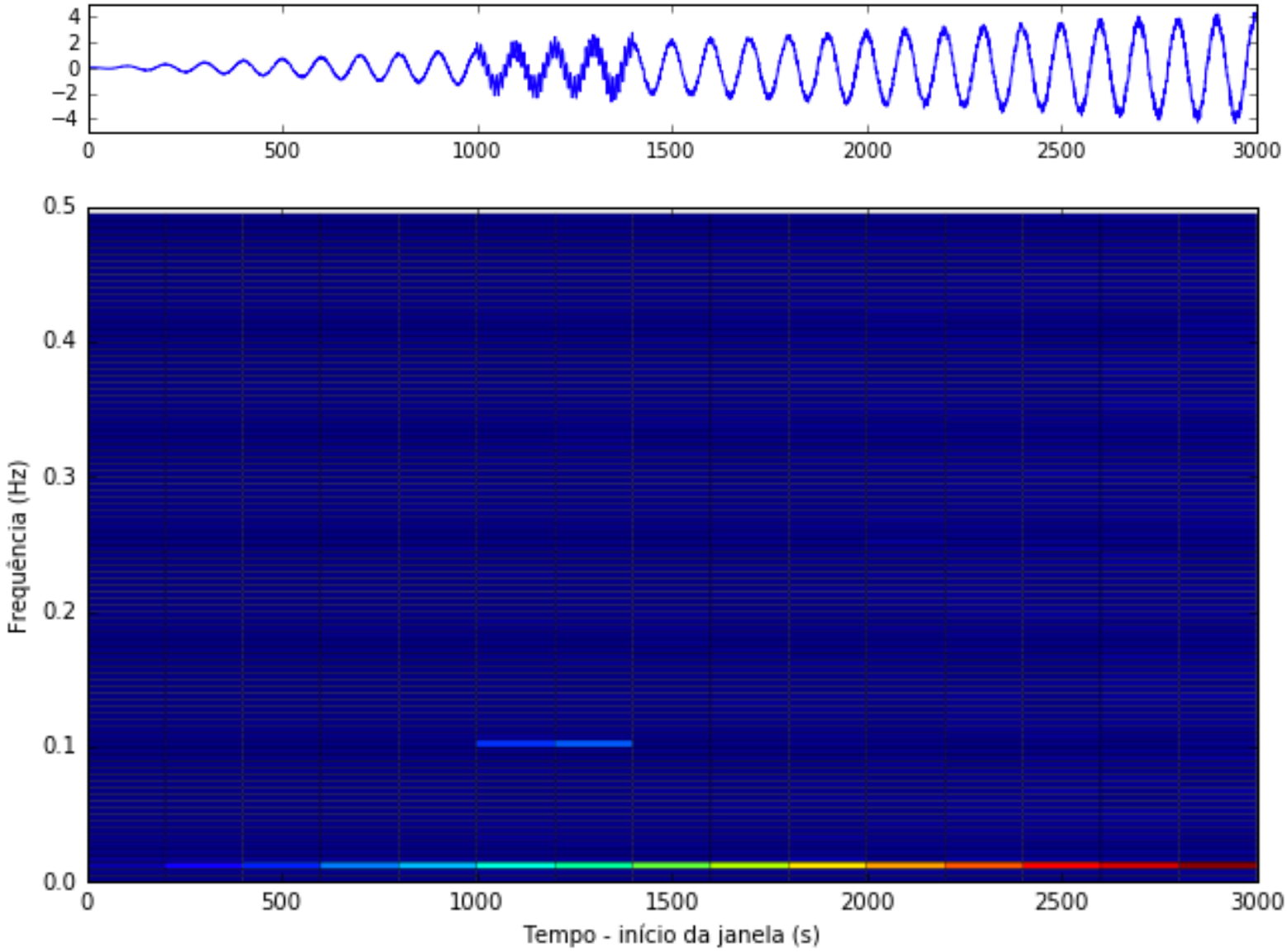
Espectrograma



Aula passada

Espectrograma de sinal variável

```
sT = sT * np.arange(0,2,2./N)
```



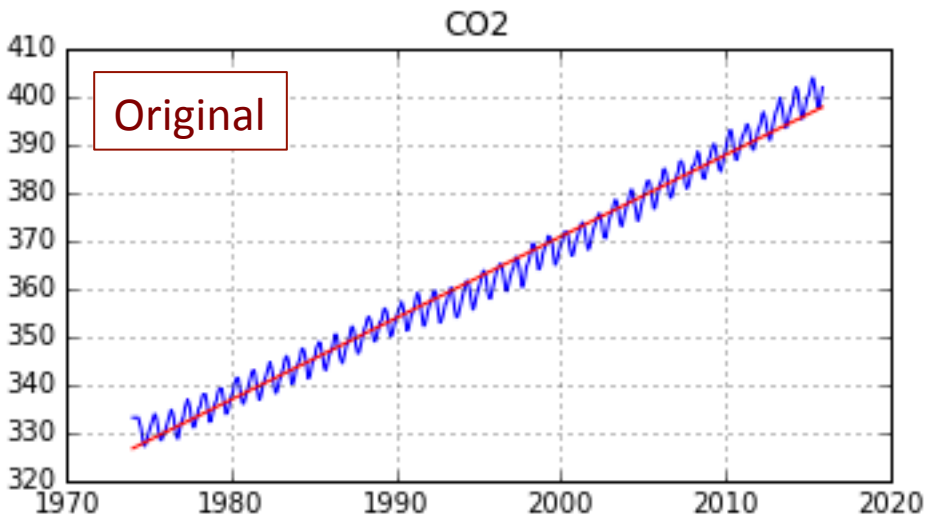
Voltando às tendências…

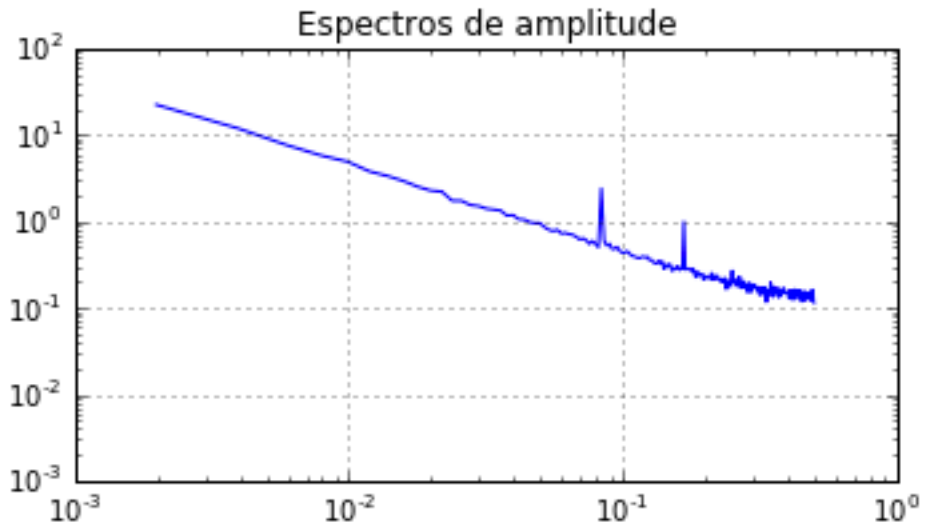
Muitas vezes a tendência não é linear, incluindo variações lentas, i.e. com períodos longos comparados com a dimensão da série.

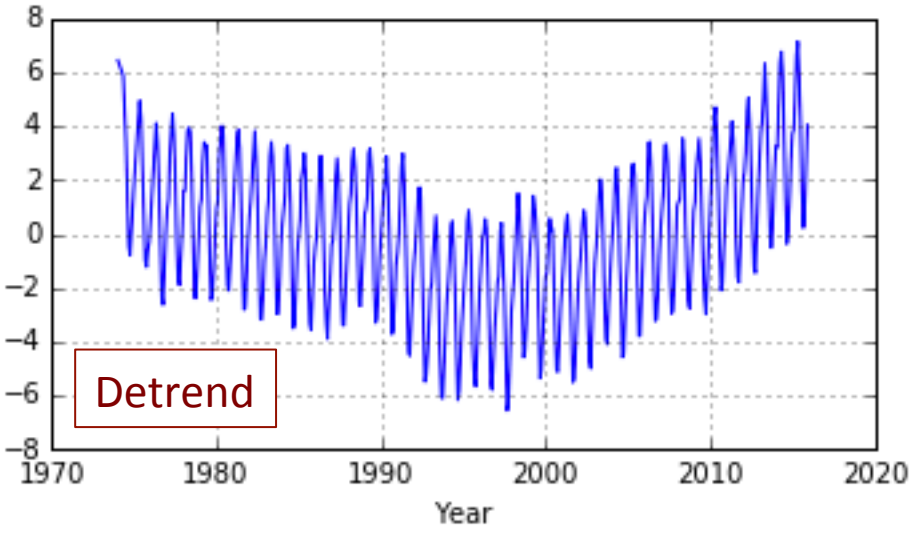
Métodos:

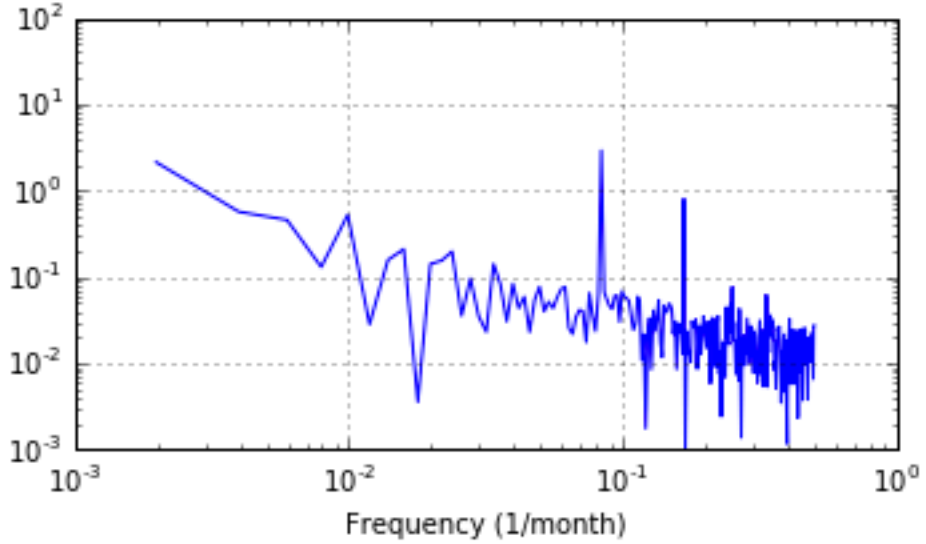
- Estimativa da tendência por regressão linear (já vimos)
- Filtro passa-alto no espectro
- Remoção de média móvel

1. Regressão linear







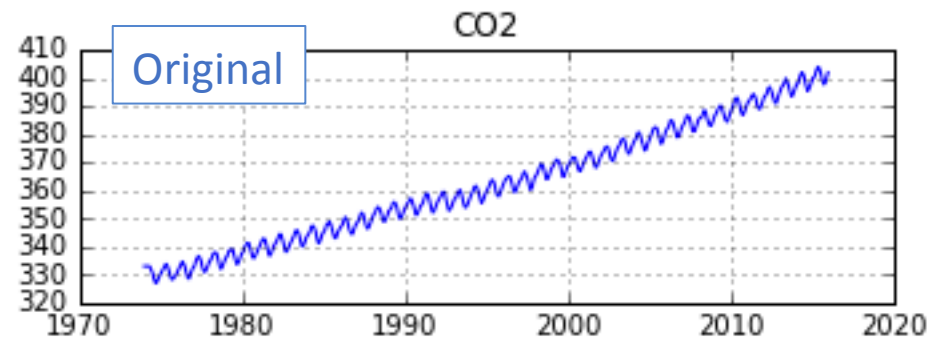


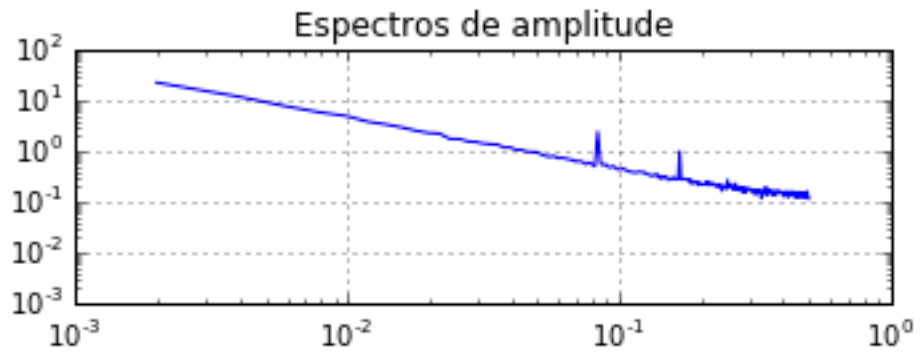
```
import matplotlib.pyplot as plt
import numpy as np
import datetime

#%% Importar dados

fin='0_co2_mlo_surface-insitu_1_ccgg_MonthlyData-susana.txt'        # ficheiro de input
data = np.loadtxt(fin, usecols=range(1,8), skiprows=145)            # ler ficheiro txt

# arrumar as datas num vector datetime
datet=[datetime.datetime(int(data[i,0]), int(data[i,1]), int(data[i,2]),
                         int(data[i,3]), int(data[i,4]), int(data[i,5]))
                         for i in range(len(data))]
juld=[dt.timetuple().tm_yday for dt in datet]                       # dia juliano

# colocar as datas num vector numérico (anos)
date = [data[i,0] + float(juld[i])/365. for i in range(len(datet))]
date = np.array(date)       # transformar em numpy array
co2=data[:,6]               # vector com valores de co2

#%% Verifica número de valores inválidos
### Constrói vector de valores válidos

datev=np.array([]); co2v=np.array([])       # matrizes de valores válidos
inval=0                                     # número de valores inválidos, inicializado a zero

for i in range(len(co2)):
    if co2[i] < 0.:
        inval = inval+1
    else:
        datev=np.append(datev,date[i])      # datas válidas
        co2v=np.append(co2v,co2[i])         # valores de CO2 válidos

print(inval)

#%% Interpola

datei=date                                  # vector com novo array de datas
co2i=np.interp(datei, datev, co2v, left=None, right=None, period=None)      # interpolação
```

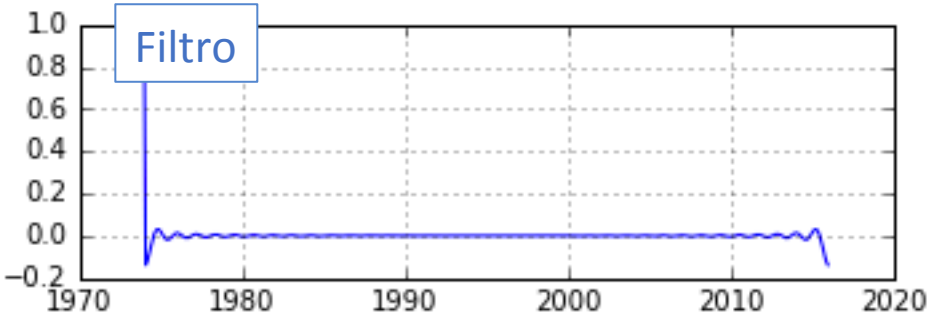
```
#%% Calcular espectro
import numpy.fft as fft

dt=1.                        # espaçamento (em meses)
N=len(co2i)                  # comprimento do vector de dados
N2=N/2                       # metade do comprimento do vector de dados
fNyq=1./(2.*dt);             # frequência de Nyquist
df=1./(dt*N);                # espaçamento do espectro
freq=np.arange(0,fNyq,df);   # vector de frequências
Fout=fft.fft(co2i);          # calcular o espectro com FFT
FoutA=np.abs(Fout[:N2])/N2; # amplitudes do espectro

plt.close()
plt.loglog(freq,FoutA)
plt.axis('tight')
plt.grid()
plt.xlabel('Frequency (1/month)')
plt.ylabel('Amplitude')


#%% Remover a tendência: ajuste a uma recta

P = np.polyfit(datei, co2i, 1)      # ajusta os dados a uma recta
co2trend=P[0]                       # declive da recta
co20=P[1]                           # ordenada na origem da recta

co2fit=co2trend*datei + co20        # recta que ajusta os dados
#co2fit=P[0]*datei + P[1]           # recta que ajusta os dados


#%% Remover a tendência: subtracção da tendência

co2_detrend = co2i - co2fit

Fout=fft.fft(co2_detrend);          # calcular o espectro com FFT
FoutAdetrend=np.abs(Fout[:N2])/N2;         # amplitudes do espectro
```

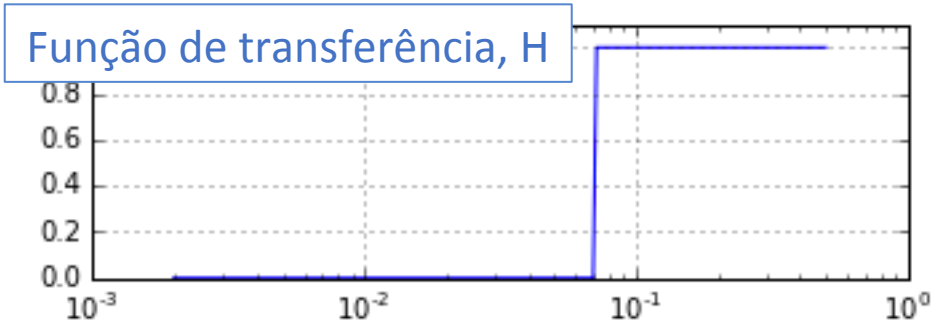
```
#%% Plot

plt.close(); plt.rcParams['figure.figsize'] = 10, 6

plt.subplot(2,2,1)
plt.plot(datei, co2i, 'b', datei, co2fit, 'r')
plt.grid()
plt.title('CO2')

plt.subplot(2,2,2)
plt.loglog(freq,FoutA);
plt.grid();
#plt.axis('tight')
plt.ylim([1e-3,1e2])
plt.title(u'Espectros de amplitude')

plt.subplot(2,2,3)
plt.plot(datei, co2_detrend, 'b')
plt.xlabel('Year')
plt.grid()

plt.subplot(2,2,4)
plt.loglog(freq,FoutAdetrend);
plt.xlabel('Frequency (1/month)')
plt.grid();
#plt.axis('tight')
plt.ylim([1e-3,1e2])

plt.tight_layout()
```

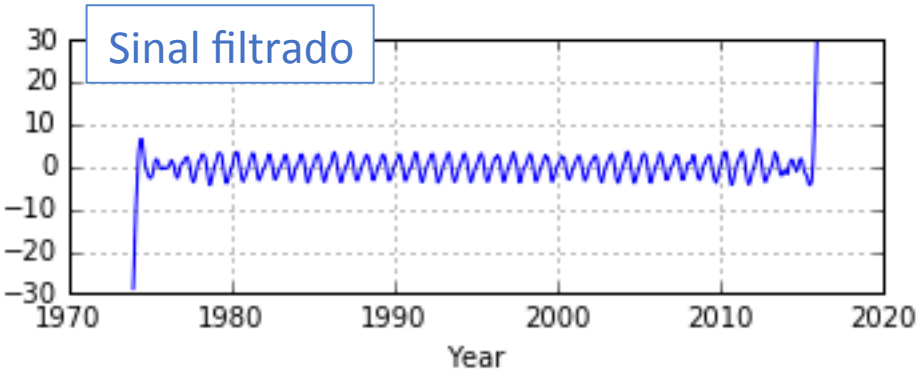
2. Filtro passa-alto (domínio espectral)

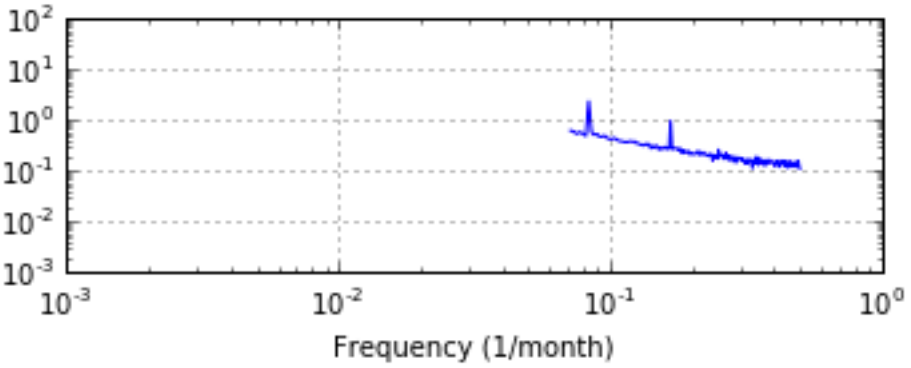










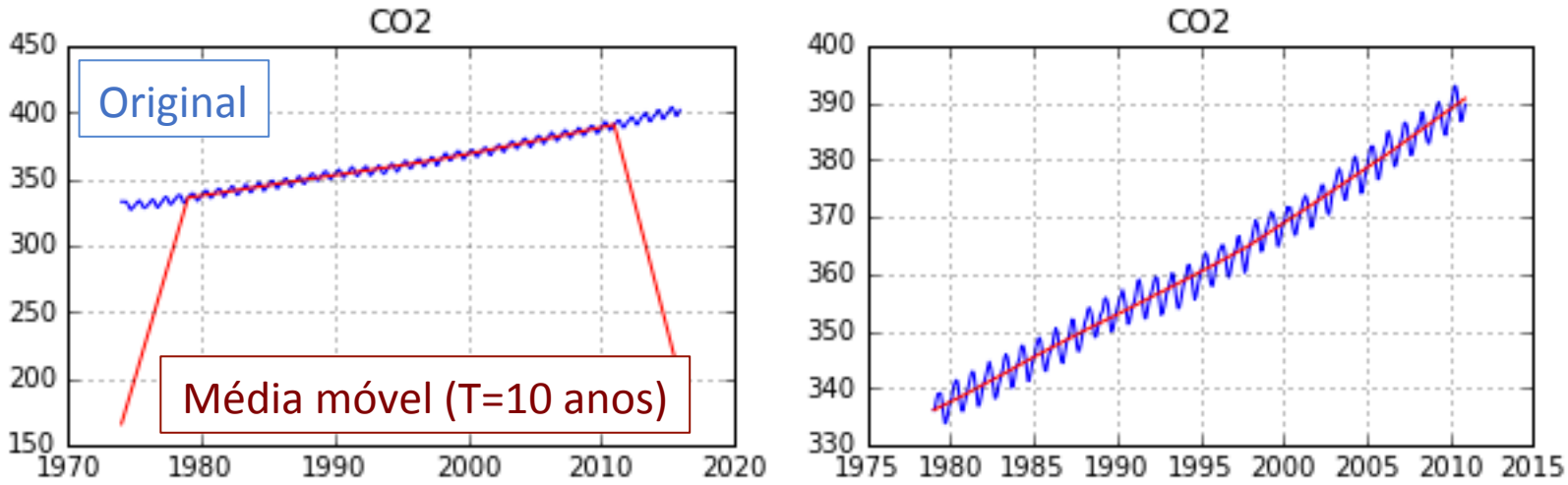


```
#%% Filtro passa alto, domínio espectral

nH = 36       # número de pontos da banda a filtrar na função de transferência
H = np.ones(len(Fout))          # função de transferência do filtro, inicializada a 1
H[:nH] = 0                       # banda a cortar
H[-nH:] = 0                      # banda a cortar (o espectro é simétrico!)

FF = Fout*H
CO2filt1 = fft.ifft(FF)

#%% Plot

plt.close(); plt.rcParams['figure.figsize'] = 10, 6

plt.subplot(3,2,1)
plt.plot(datei, co2i)
plt.grid()
plt.title('CO2')

plt.subplot(3,2,2)
plt.loglog(freq,FoutA);
plt.grid();
#plt.axis('tight')
plt.ylim([1e-3,1e2])
plt.title(u'Espectros de amplitude')
```

Continuação…

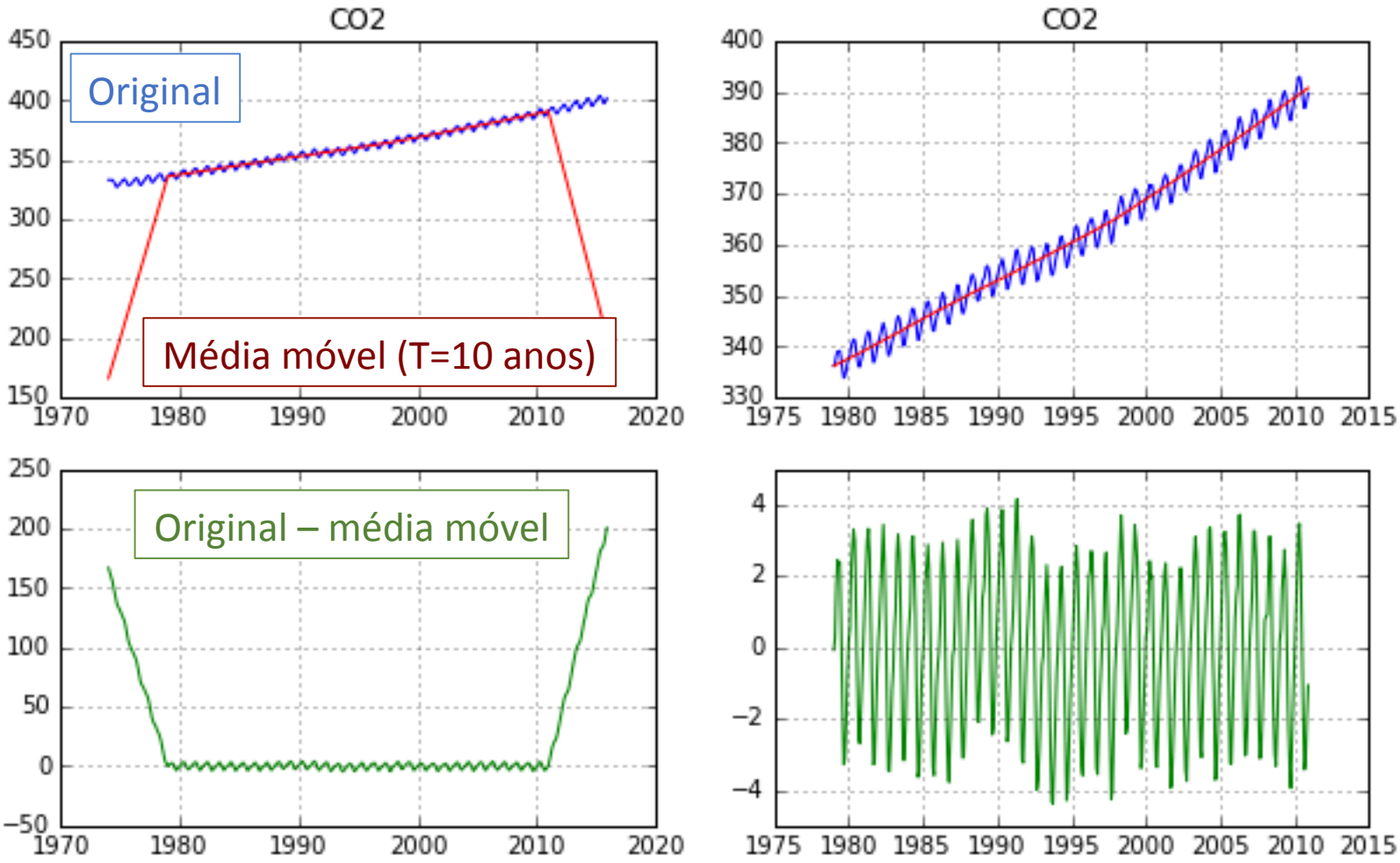
```
plt.subplot(3,2,3)
plt.plot(datei, fft.ifft(H))
plt.grid()

plt.subplot(3,2,4)
plt.semilogx(freq,H[:N2]);
plt.grid();
plt.ylim([0,1.1])

plt.subplot(3,2,5)
plt.plot(datei, CO2filt1)
plt.xlabel('Year')
plt.grid()

plt.subplot(3,2,6)
plt.loglog(freq,np.abs(FF[:N2])/N2);
plt.xlabel('Frequency (1/month)')
plt.grid();
plt.ylim([1e-3,1e2])

plt.tight_layout()
```

3. Filtro passa-alto (domínio do tempo, média móvel)




```
#%% Filtro passa alto, média móvel
nA=120.          # número de pontos da janela: 10 anos = 120 meses
nA2=nA/2
h=np.ones(nA)/nA     # janela, =1
hh=np.append(h,np.zeros(N-nA))       # série da janela, com o mesmo número de pontos que a série or
co2low = np.convolve(co2i, h, 'same')       # média móvel, convolução com h (sinal filtrado passa-
co2high = co2i-co2low                       # sinal filtrado (passa-alto = original - passa-baixo)
```

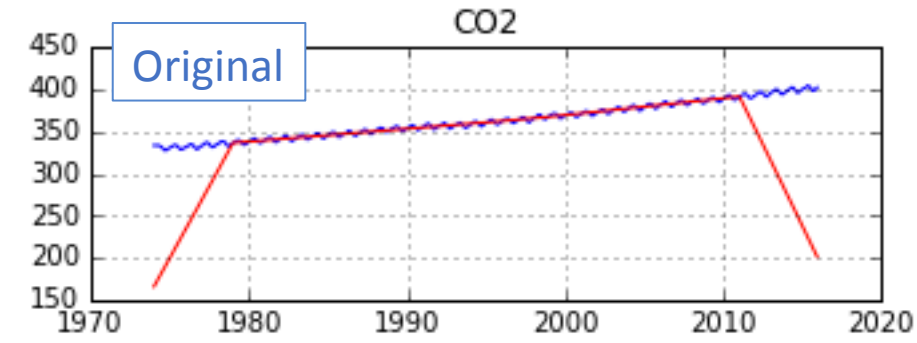
3. Filtro passa-alto (domínio do tempo, média móvel)

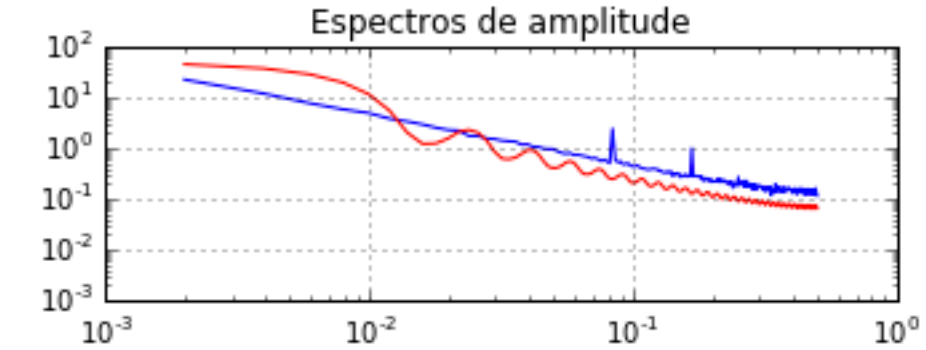


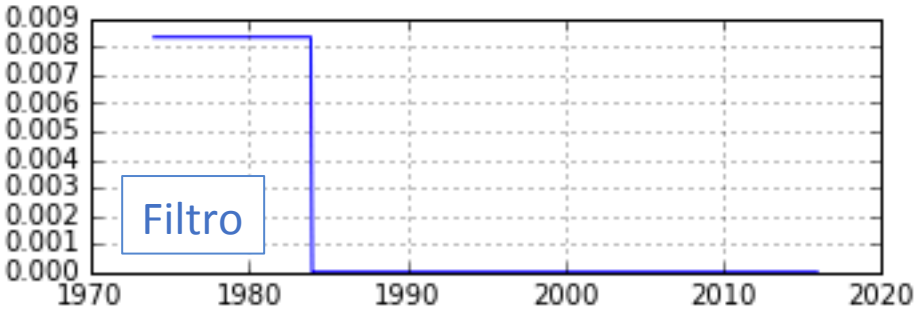
3. Filtro passa-alto (domínio do tempo, média móvel)

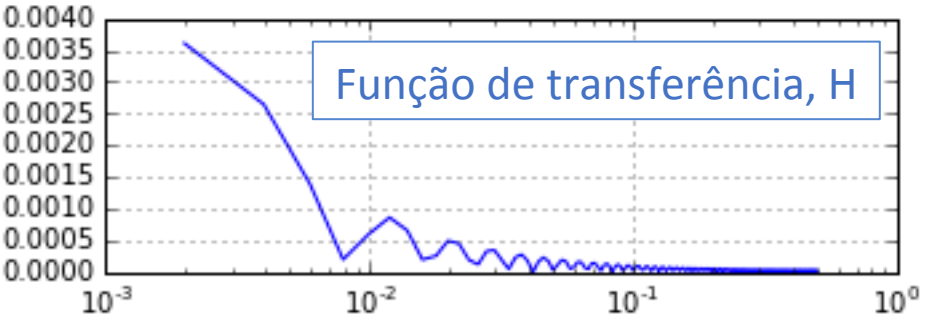
```
#%% Filtro passa alto, média móvel
nA=120.         # número de pontos da janela: 10 anos = 120 meses
nA2=nA/2
h=np.ones(nA)/nA    # janela, =1
hh=np.append(h,np.zeros(N-nA))      # série da janela, com o mesmo número de pontos que a série or
co2low = np.convolve(co2i, h, 'same')       # média móvel, convolução com h (sinal filtrado passa-
co2high = co2i-co2low                       # sinal filtrado (passa-alto = original - passa-baixo)

#%% Plot

plt.close(); plt.rcParams['figure.figsize'] = 10, 6

plt.subplot(2,2,1)
plt.plot(datei, co2i, 'b', datei, co2low, 'r')
plt.grid()
plt.title('CO2')

plt.subplot(2,2,3)
plt.plot(datei, co2high, 'g');
plt.grid();

plt.subplot(2,2,2)
plt.plot(datei[nA2:-nA2], co2i[nA2:-nA2], 'b', datei[nA2:-nA2], co2low[nA2:-nA2], 'r')
plt.grid()
plt.title('CO2')

plt.subplot(2,2,4)
plt.plot(datei[nA2:-nA2], co2high[nA2:-nA2], 'g');
plt.grid();
```

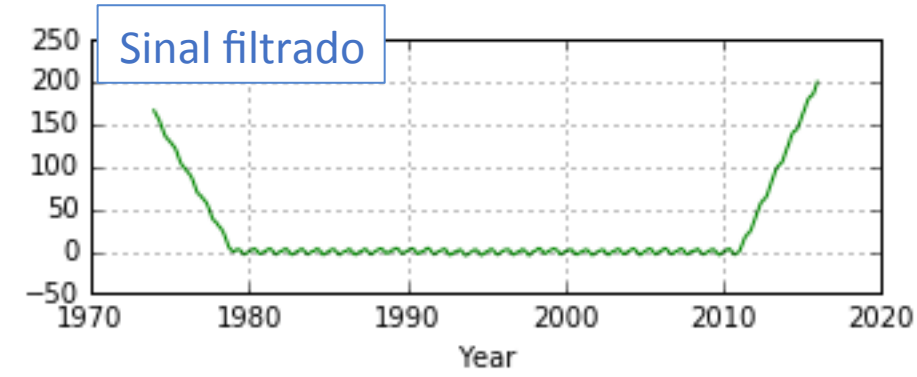
3. Filtro passa-alto (domínio do tempo, média móvel)

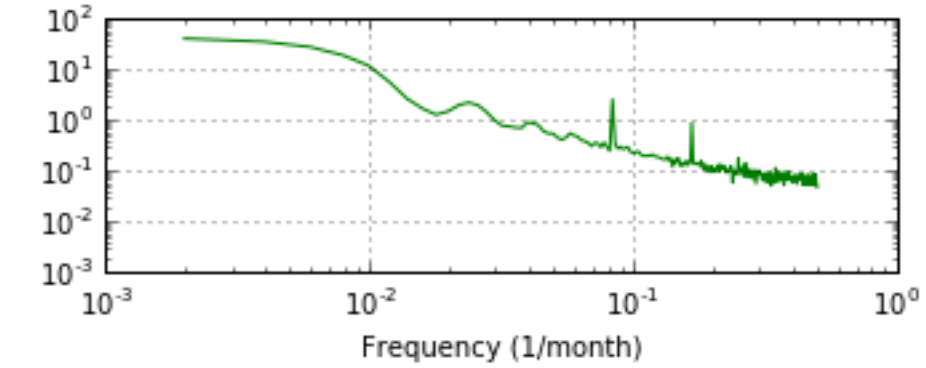








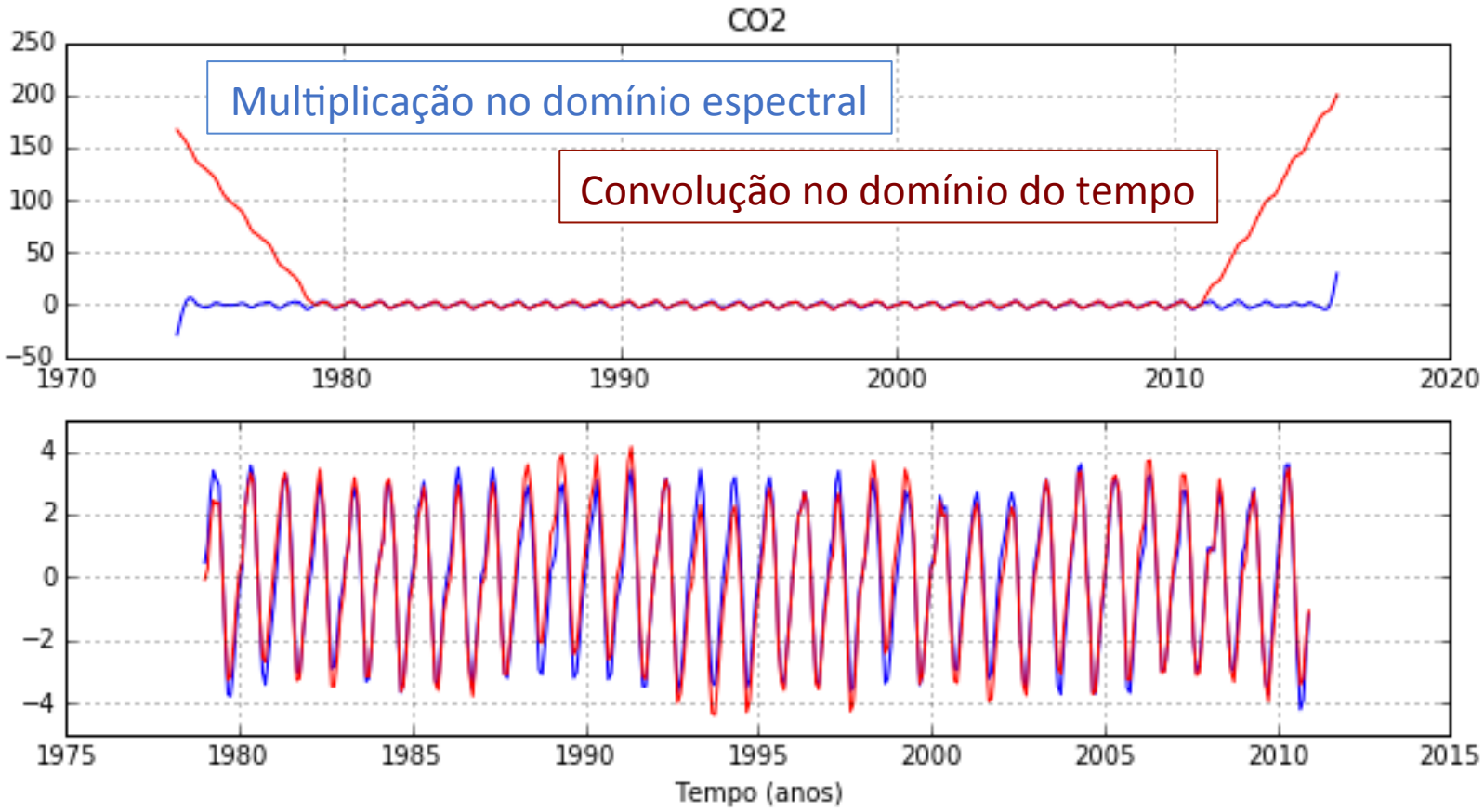




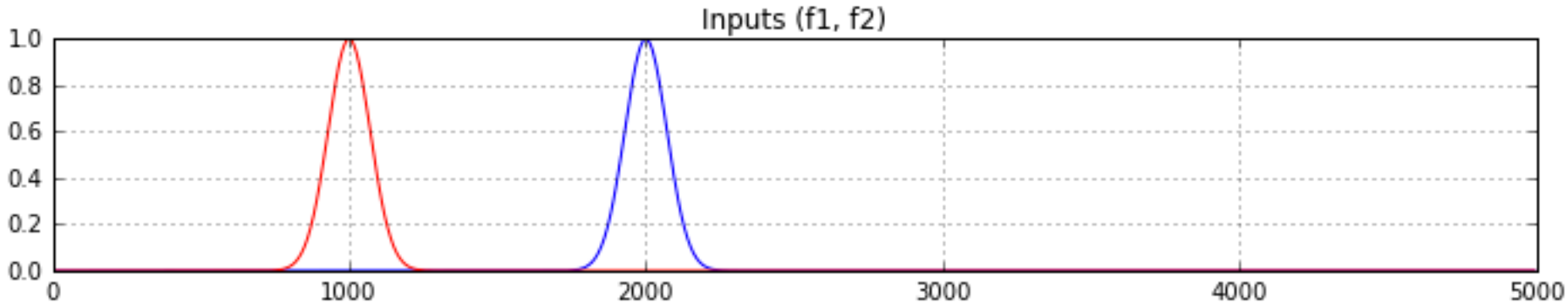
```
#%% Plot

plt.close(); plt.rcParams['figure.figsize'] = 10, 6

plt.subplot(3,2,1)
plt.plot(datei, co2i, 'b', datei, co2low, 'r')
plt.grid()
plt.title('CO2')

plt.subplot(3,2,2)
plt.loglog(freq,FoutA, 'b', freq, np.abs(fft.fft(co2low)[:N2])/N2, 'r');
plt.grid();
plt.ylim([1e-3,1e2])
plt.title(u'Espectros de amplitude')

plt.subplot(3,2,3)
plt.plot(datei, hh)
plt.grid()

plt.subplot(3,2,4)
plt.semilogx(freq, np.abs(fft.fft(hh)[:N2])/N2);
plt.grid();

plt.subplot(3,2,5)
plt.plot(datei, co2high)
plt.xlabel('Year')
plt.grid()

plt.subplot(3,2,6)
plt.loglog(freq, np.abs(fft.fft(co2high)[:N2])/N2);
plt.xlabel('Frequency (1/month)')
plt.grid();
plt.ylim([1e-3,1e2])

plt.tight_layout()
```

Comparação entre filtros passa-alto (tempo vs espectro)

```
plt.subplot(2,1,1)
plt.plot(datei, CO2filt1, 'b', datei, co2high, 'r')
plt.grid()
plt.title('CO2')

plt.subplot(2,1,2)
plt.plot(datei[nA2:-nA2], CO2filt1[nA2:-nA2], 'b', datei[nA2:-nA2], co2high[nA2:-nA2], 'r')
plt.grid()
plt.xlabel('Tempo (anos)')
```



Correlação entre séries

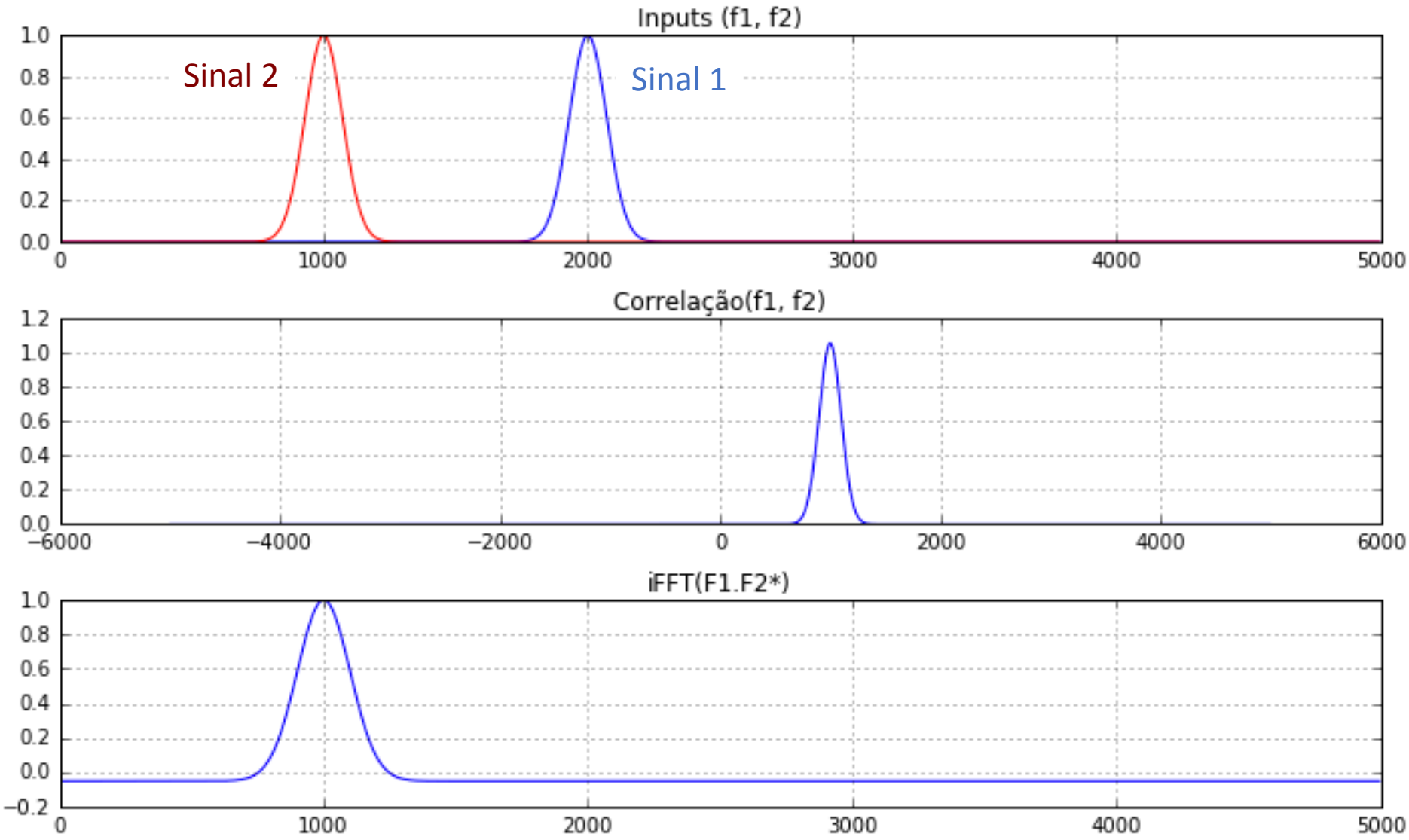


```
import matplotlib.pyplot as plt
import numpy as np
import numpy.fft as fft

#%%

nx=1000.                # número de pontos da série
dx=5.                   # espaçamento temporal da amostragem
x=np.arange(0., nx*dx, dx)          # vector de tempo
Ls=np.array([100., 100., 100.])     # dimensão dos sinais
x1s=np.array([2000., 2000., 2000.]) # lag do sinal 1
x2s=np.array([1000., 2000., 2500.]) # lag do sinal 2
A2s=np.array([1., 2., -5.])         # amplitude do sinal 2

for i in range(len(Ls))[-1:]:
    L=Ls[i]
    x1=x1s[i]
    x2=x2s[i]
    A2=A2s[i]

    f1=np.exp(-((x-x1)/L)**2)       # sinal 1
    f2=A2*np.exp(-((x-x2)/L)**2)    # sinal 2

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

    # sinais de input
    plt.subplot(3,1,1)
    plt.plot(x,f1,'b', x,f2,'r')
    plt.title(u'Inputs (f1, f2)')
    plt.grid()

    # correlação no domínio do tempo
    corr=np.correlate(f1,f2, 'full')
    corr=corr/(np.std(f1)*np.std(f2)*nx)
    lags=np.arange(-nx+1, nx)*dx
    plt.subplot(3,1,2)
    plt.plot(lags, corr, 'b')
    plt.title(u'Correlação(f1, f1)')
    plt.grid()
```

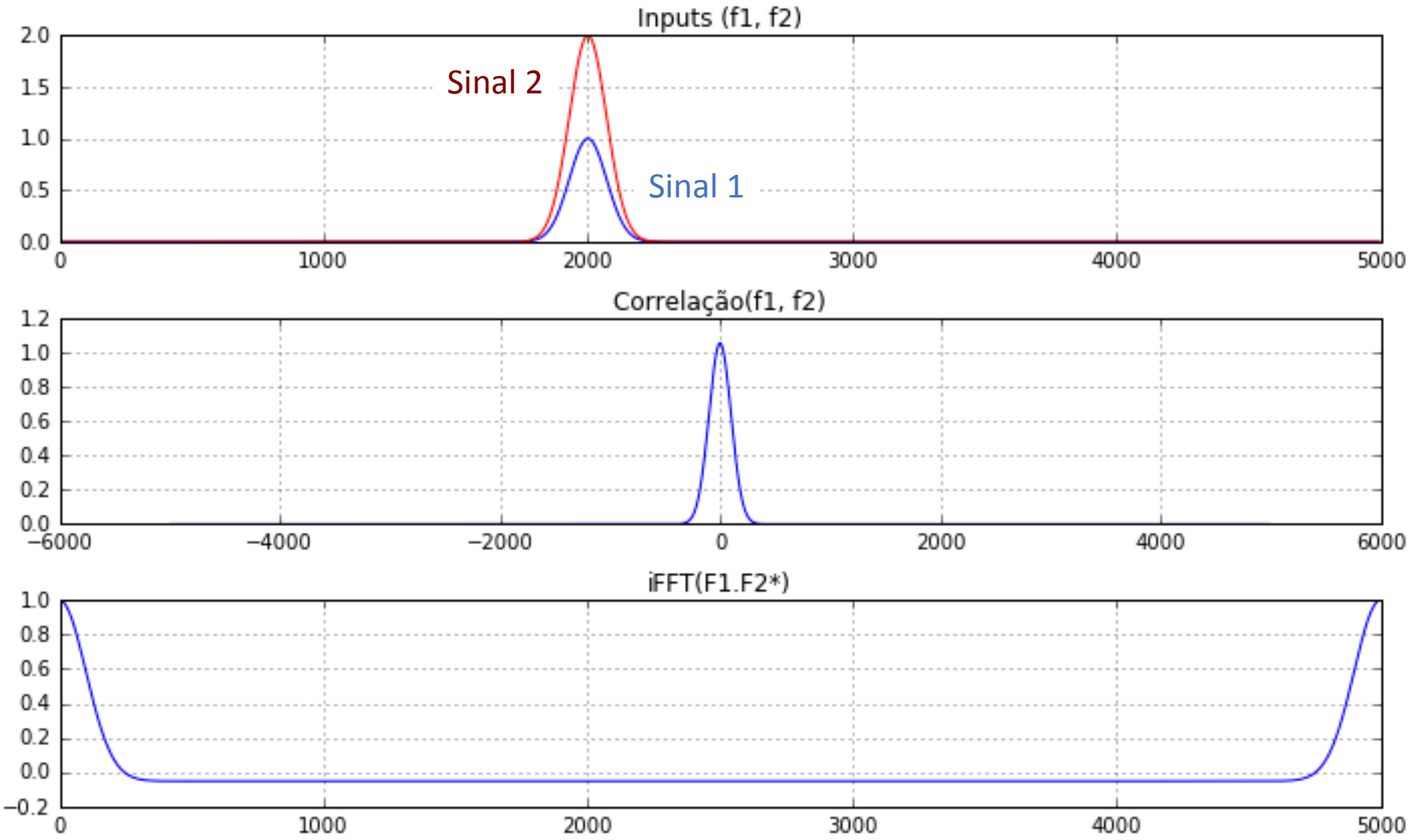
Continua...

```
for i in range(len(Ls))[-1:]:
    L=Ls[i]
    x1=x1s[i]
    x2=x2s[i]
    A2=A2s[i]

    f1=np.exp(-((x-x1)/L)**2)       # sinal 1
    f2=A2*np.exp(-((x-x2)/L)**2)    # sinal 2

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

    # sinais de input
    plt.subplot(3,1,1)
    plt.plot(x,f1,'b', x,f2,'r')
    plt.title(u'Inputs (f1, f2)')
    plt.grid()

    # correlação no domínio do tempo
    corr=np.correlate(f1,f2, 'full')
    corr=corr/(np.std(f1)*np.std(f2)*nx)
    lags=np.arange(-nx+1, nx)*dx
    plt.subplot(3,1,2)
    plt.plot(lags, corr, 'b')
    plt.title(u'Correlação(f1, f1)')
    plt.grid()

    # correlação no domínio do espectro
    F1=fft.fft(f1-np.mean(f1))
    F2=fft.fft(f2-np.mean(f2))
    FF=F1*np.conjugate(F2)
    cc=fft.ifft(FF)
    cc = cc/(np.std(f1)*np.std(f2)*nx)
    plt.subplot(3,1,3)
    plt.plot(x,cc, 'b')
    plt.title(u'iFFT(F1.F2*)')
    plt.grid()

    plt.tight_layout()
```

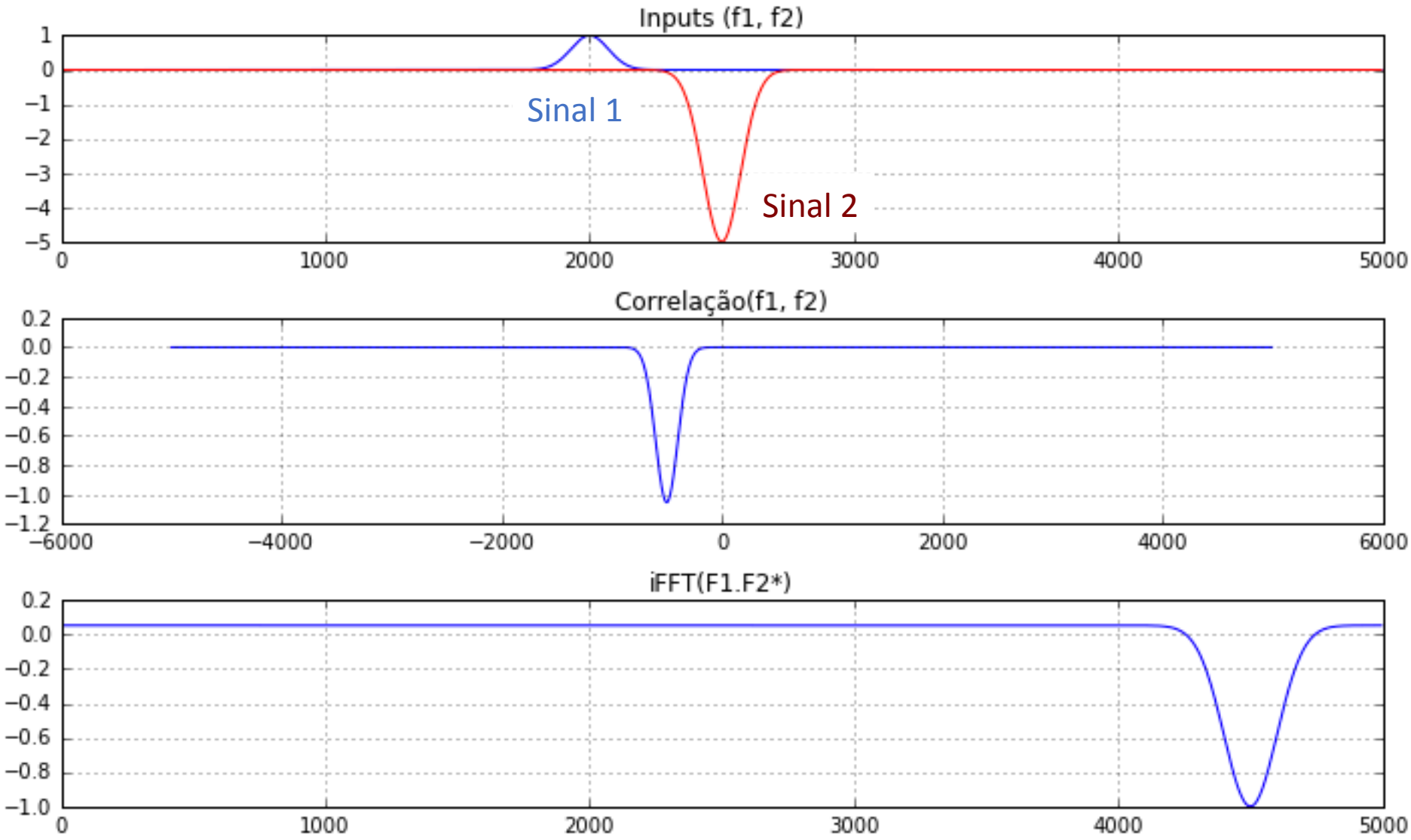
1. Séries positivamente correlacionadas com desfasamento



2. Séries positivamente correlacionadas sem desfasamento



3. Séries nagativamente correlacionadas com desfasamento



Correlação entre séries

- O método de Fourier de cálculo da correlação é muito rápido e preciso (se o comprimento da série permitir a FFT, e.g. $N = 2k$). O método assume continuidade cícilica.

- O cáculo explícito (no domínio físico) assume continuidade nula das series e é, em geral, muito demorado.