Trabalho Prático 3

Processamento Digital de Sinais

Curso de Licenciatura Informática e Multimédia (LEIM)

Ano Letivo 2017/2018

Data: 17/06/2018

Turma LEIM 23D

Docentes:

Eng. André Lourenço

Eng. Isabel Rodrigues

Alunos:

Luis Fonseca (A45125)

Philipp Al-Badavi (A45138)

Índice

Introdução

O objetivo deste trabalho consistia na representação de sinais FIR/IIR no python usando as bibliotecas: numpy, scipy e matplotlib. Este trabalho está divido em 2 grupos, no qual o primeiro consistia em estudar a forma e o comportamento dos sinais quando chamada a função *scipy.signal.freqz*. E de seguida nas alíneas a seguir, pedia-se o cálculo analiticamente para comprovar os resultados

O segundo grupo consistia em construir o desenho dos filtros, dando uma frequência de corte, usando o: *scipy.signal.firwin* , e de estudar o comportamento dos sinais quando aplicado no python um ficheiro wav.

Exercício 1

- ## Alínea a)

  Neste exercício pedia-se a construção de gráficos dando as expressões fornecidas, foi-nos também fornecido os valores dos coeficientes para a visualização de diferentes gráficos. Colocando os valores desses coeficientes dentro de um array, neste caso os arrays bk representavam os coeficientes da variavel “x” e os ak representavam os coeficientes da variavel “y”. Usou-se a função *freqz* da biblioteca de “scipy.signal”, que vai ter como argumentos as variáveis dos arrays colocados,os tais bk e ak. No final para a representação e visualização dos gráficos usou-se o plot.

Fig.1 – Código Python

```
import numpy as np
import scipy.signal as ss
import matplotlib.pyplot as plt
import scipy.io.wavfile as wav

#----------------------------alinea a)-------------------------
#-------------------------------------Y1----------------------------
#--------Y1 c = 1 r =-0.9----------
plt.figure()
ak1 = [1.]
bk1 = [1.,-0.9]
plt.subplot(3,1,1)
w,h = ss.freqz(bk1,ak1)
plt.grid()
plt.title("alinea a) Filtro y1[n] = x[n] + 0.9x[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))

#--------Y1 c = 1 r =0.1----------
ak2 = [1.]
bk2 = [1.,0.1]
plt.subplot(3,1,2)
w,h = ss.freqz(bk2,ak2)
plt.grid()
plt.title("alinea a) Filtro y1[n] = x[n] - 0.1x[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))

#--------Y1 c = 1 r =0.9----------
ak3 = [1.]
bk3 = [1.,0.9]
plt.subplot(3,1,3)
w,h = ss.freqz(bk3,ak3)
plt.grid()
plt.title("alinea a) Filtro y1[n] = x[n] - 0.9x[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))
plt.show()
```

Fig.2 – Output

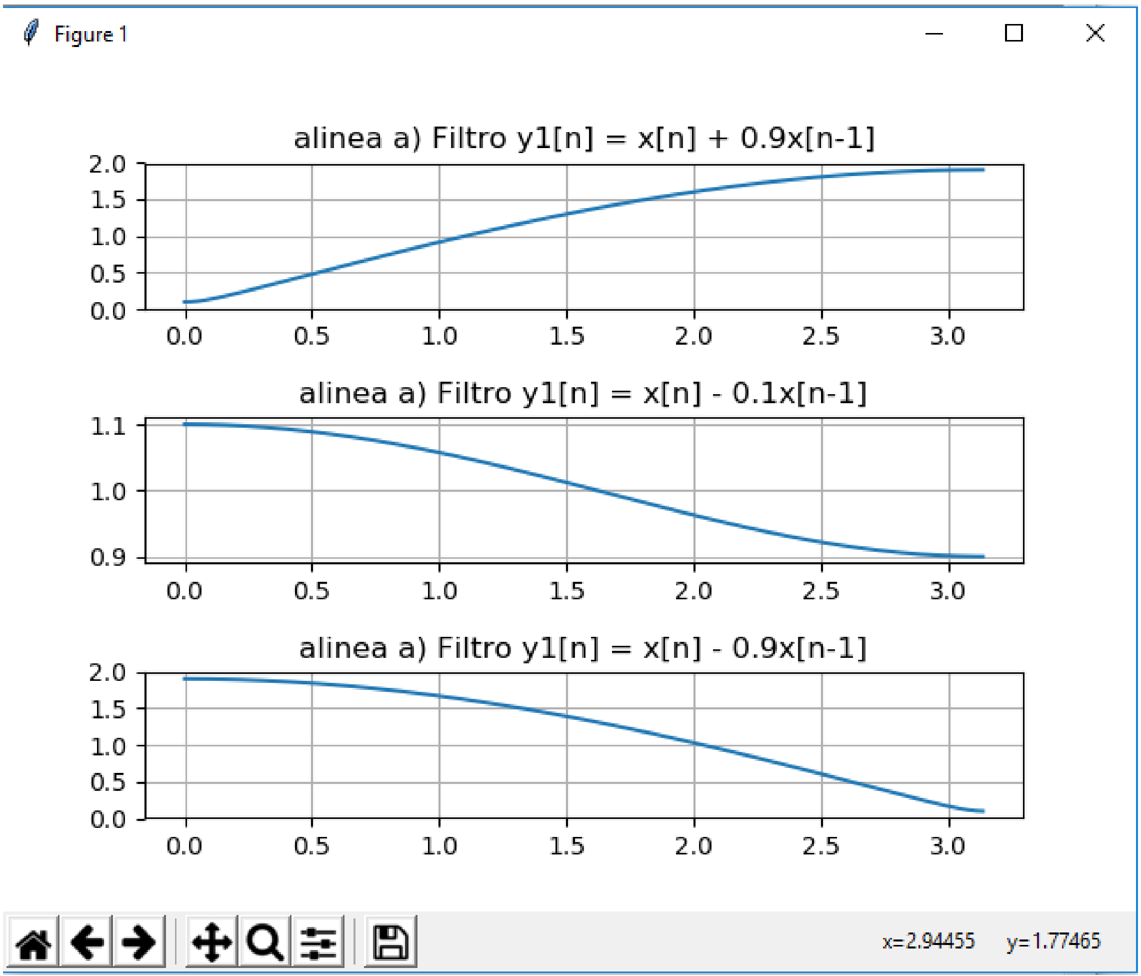


Figure 1
alinea a) Filtro y1[n] = x[n] + 0.9x[n-1]
alinea a) Filtro y1[n] = x[n] - 0.1x[n-1]
alinea a) Filtro y1[n] = x[n] - 0.9x[n-1]
x=2.94455 y=1.77465

Fig.3 – Código Python

```
#--------Y1 c = 2 r =-0.9----------
plt.figure()
ak4 = [1.]
bk4 = [1.,0.,-0.9]
plt.subplot(3,1,1)
w,h = ss.freqz(bk4,ak4)
plt.grid()
plt.title("alinea a) Filtro y1[n] = x[n] + 0.9x[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))

#--------Y1 c = 2 r =0.1----------
ak5 = [1.]
bk5 = [1.,0.,0.1]
plt.subplot(3,1,2)
w,h = ss.freqz(bk5,ak5)
plt.grid()
plt.title("alinea a) Filtro y1[n] = x[n] - 0.1x[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))

#--------Y1 c = 2 r =0.9----------
ak6 = [1.]
bk6 = [1.,0.,0.9]
plt.subplot(3,1,3)
w,h = ss.freqz(bk6,ak6)
plt.grid()
plt.title("alinea a) Filtro y1[n] = x[n] - 0.9x[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))
plt.show()
```

Fig.4 – Output

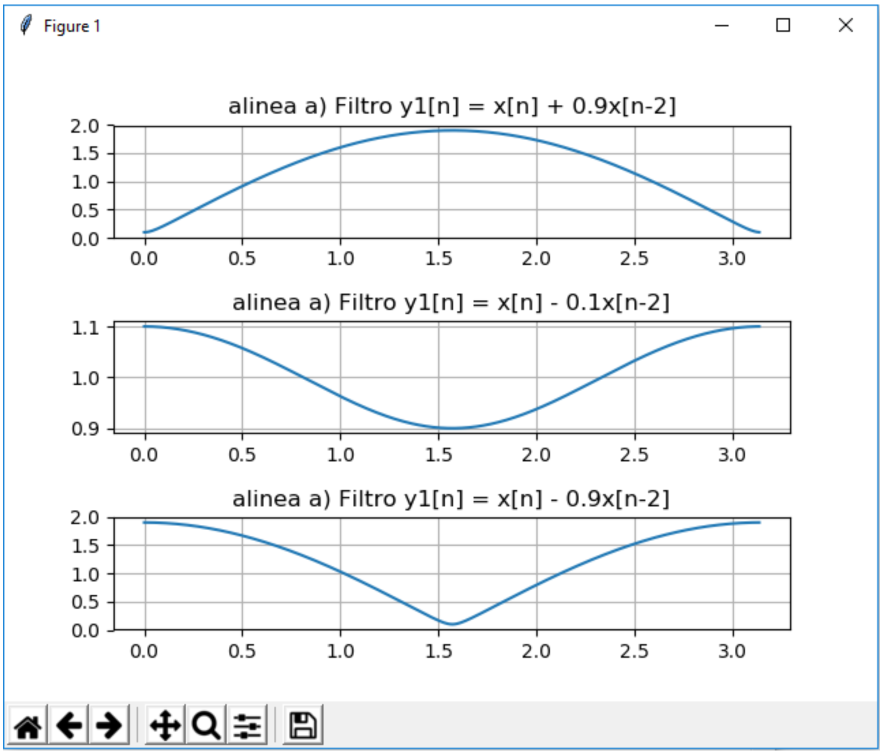


Fig.5 – Código Python

```
#---------------------------------------Y2--------------------------
#--------Y2 c = 1 r =-0.9----------
plt.figure()
ak7 = [1.,-0.9]
bk7 = [1.]
plt.subplot(3,1,1)
w,h = ss.freqz(bk7,ak7)
plt.grid()
plt.title("alinea a) Filtro y2[n] = x[n] + 0.9y2[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))

#--------Y2 c = 1 r =0.1----------
ak8 = [1.,0.1]
bk8 = [1.]
plt.subplot(3,1,2)
w,h = ss.freqz(bk8,ak8)
plt.grid()
plt.title("alinea a) Filtro y2[n] = x[n] - 0.1y2[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))

#--------Y2 c = 1 r =0.9----------
ak9 = [1.,0.9]
bk9 = [1.]
plt.subplot(3,1,3)
w,h = ss.freqz(bk9,ak9)
plt.grid()
plt.plot(w,np.abs(h))
plt.title("alinea a) Filtro y2[n] = x[n] - 0.9y2[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.show()
```

Fig.6 – Output

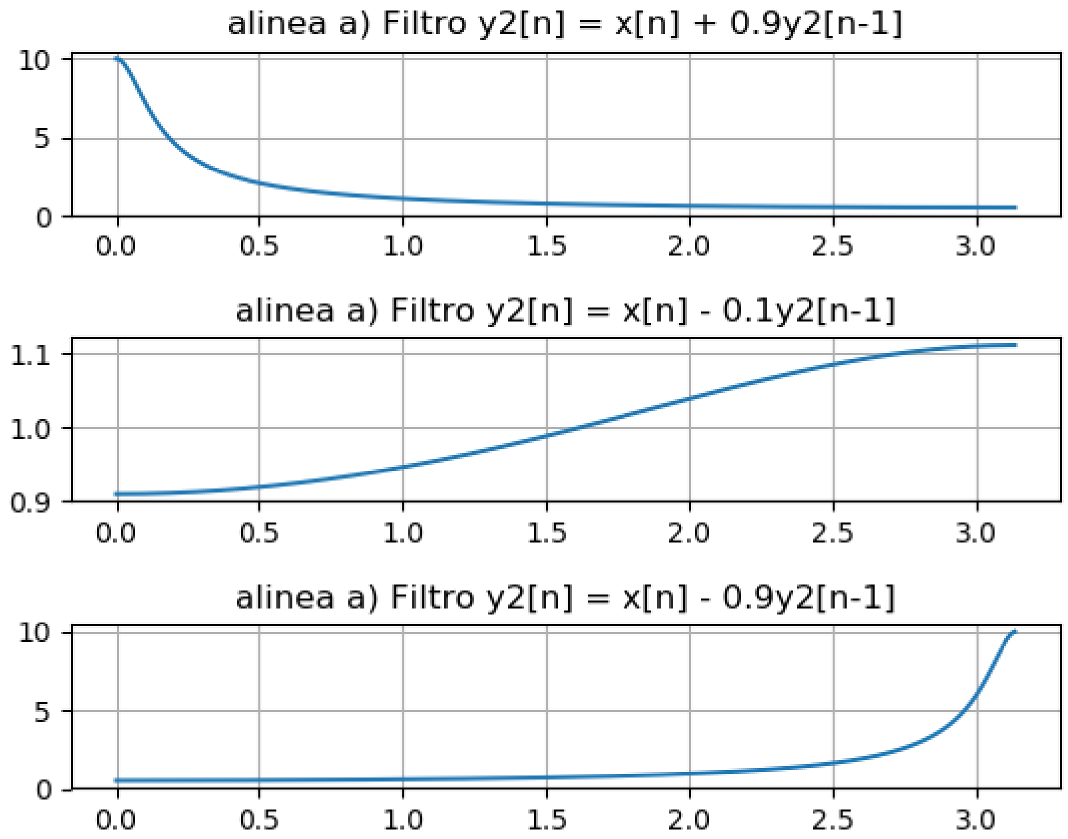
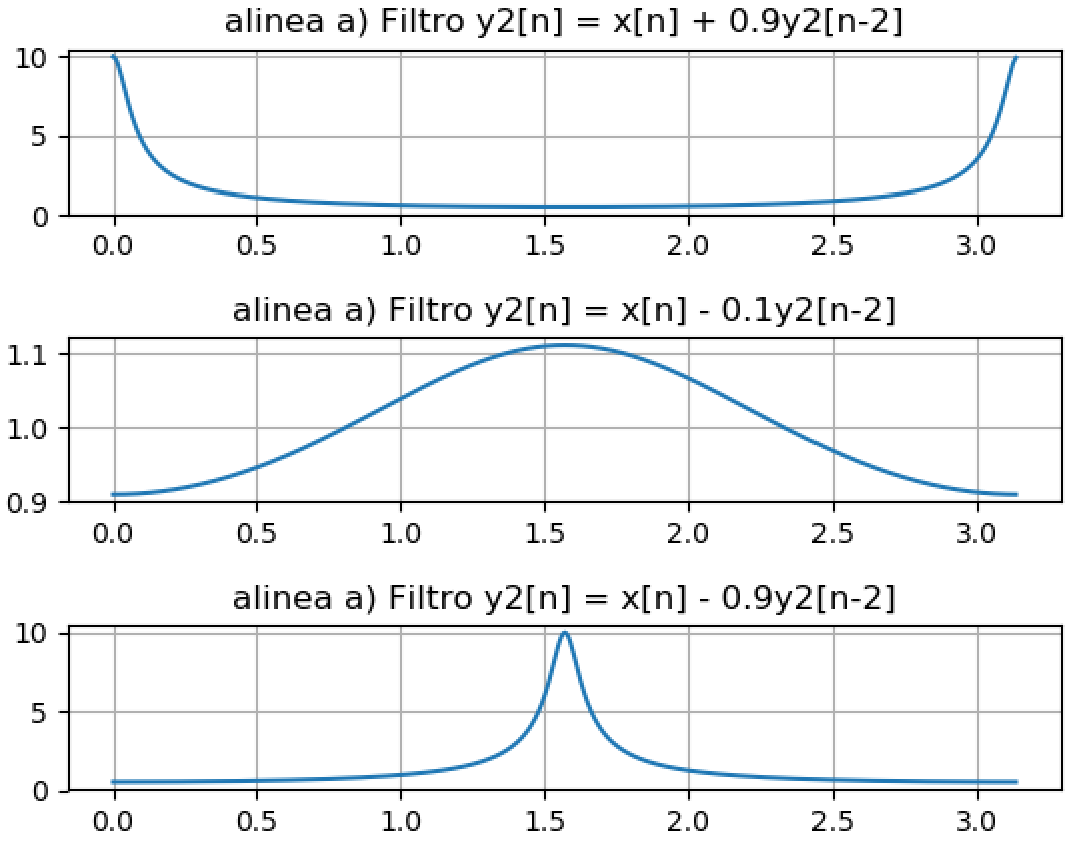


Fig.7 – Código Python

```
#--------Y2 c = 2 r =-0.9----------
plt.figure()
ak10 = [1.,0.,-0.9]
bk10 = [1.]
plt.subplot(3,1,1)
w,h = ss.freqz(bk10,ak10)
plt.grid()
plt.title("alinea a) Filtro y2[n] = x[n] + 0.9y2[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))

#--------Y2 c = 2 r =0.1----------
ak11 = [1.,0.,0.1]
bk11 = [1.]
plt.subplot(3,1,2)
w,h = ss.freqz(bk11,ak11)
plt.grid()
plt.title("alinea a) Filtro y2[n] = x[n] - 0.1y2[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(w,np.abs(h))
#--------Y2 c = 2 r =0.9----------
ak12 = [1.,0.,0.9]
bk12 = [1.]
plt.subplot(3,1,3)
w,h = ss.freqz(bk12,ak12)
plt.grid()
plt.plot(w,np.abs(h))
plt.title("alinea a) Filtro y2[n] = x[n] - 0.9y2[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.show()
```

Fig.8 – Output



- **Alínea b)**

1. $y1[n] = x[n] + 0,9x[n - 1]$
   Filtro passa alto.
2. $y1[n] = x[n] - 0,9x[n - 1]$
   Filtro passa baixo.
3. $y1[n] = x[n] - 0,1x[n - 1]$
   Filtro passa baixo.
4. $y1[n] = x[n] + 0,9x[n - 2]$
   Filtro corta banda.
5. $y1[n] = x[n] - 0,9x[n - 2]$
   Filtro corta banda.
6. $y1[n] = x[n] - 0,1x[n - 2]$
   Filtro corta banda.
7. $y2[n] = x[n] + 0.9y2[n - 1]$
   Filtro passa baixo.
8. $y2[n] = x[n] - 0.1y2[n - 1]$
   Filtro passa altro.
9. $y2[n] = x[n] - 0.9y2[n - 1]$
   Filtro passa alto.
10. $y2[n] = x[n] + 0.9y2[n - 2]$
    Filtro corta banda.
11. $y2[n] = x[n] - 0.1y2[n - 2]$
    Filtro corta banda.
12. $y2[n] = x[n] - 0.9y2[n - 2]$
    Filtro corta banda.

- **Alínea c)**

Para esta alínea era pedido a saída do sinal quando passada a função $x[n] = 10 + 2\cos(\frac{\pi}{6} n) + 10\cos(\frac{\pi}{3} n)$ usando a função *scipy.signal.lfilter()*, e foi também feito o cálculo teóricamente para verificar que os valores que nos deram.

Fig.9– Função de entrada, feito em Python

```
#-----------------------------alinea c)--------------------------
t = np.arange(-5,5,1./1000)

x = 10 + 2*np.cos((np.pi/6)*t)+10*np.cos((np.pi/3)*t)
```

Fig.10 – Código Python

```
#----Y1--c =1 R = -0.9-------------------------------------
plt.figure()
y = ss.lfilter(bk1,ak1,x)
plt.subplot(3,1,1)
plt.grid()
plt.title("alinea c) Filtro y1[n] = x[n] + 0.9x[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
#----Y1--c =1 R = 0.1--------------------------------------
y = ss.lfilter(bk2,ak2,x)
plt.subplot(3,1,2)
plt.grid()
plt.title("alinea c) Filtro y1[n] = x[n] - 0.1x[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
#----Y1--c =1R = 0.9 ----------------------------------
y = ss.lfilter(bk3,ak3,x)
plt.subplot(3,1,3)
plt.grid()
plt.title("alinea c) Filtro y1[n] = x[n] - 0.9x[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
plt.show()
```

Fig.11 – Output

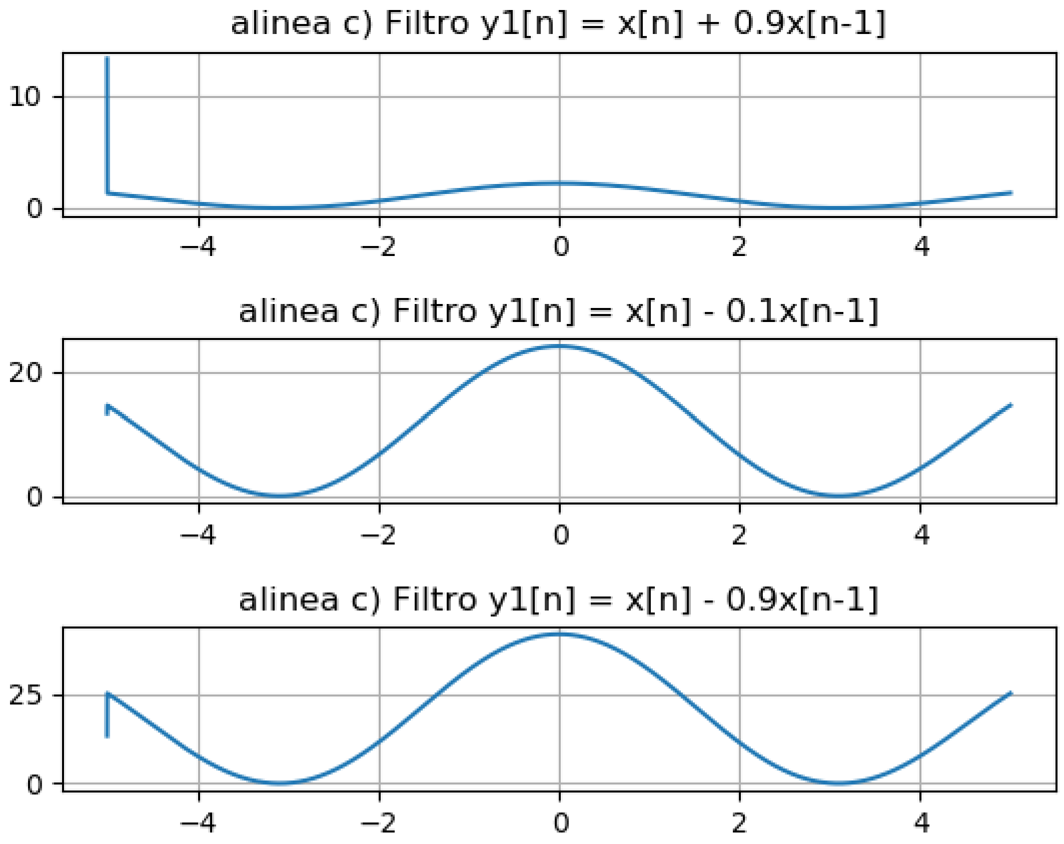


Fig.12– Código Python

```
#----Y1--c =2 R = -0.9--------------------------------------
plt.figure()
y = ss.lfilter(bk4,ak4,x)
plt.subplot(3,1,1)
plt.grid()
plt.title("alinea c) Filtro y1[n] = x[n] + 0.9x[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
#----Y1--c =2 R = 0.1--------------------------------------
y = ss.lfilter(bk5,ak5,x)
plt.subplot(3,1,2)
plt.grid()
plt.title("alinea c) Filtro y1[n] = x[n] - 0.1x[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
#----Y1--c =2 R = 0.9 ---------------------------------
y = ss.lfilter(bk6,ak6,x)
plt.subplot(3,1,3)
plt.grid()
plt.title("alinea c) Filtro y1[n] = x[n] - 0.9x[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
plt.show()
```

Fig.13 – Output

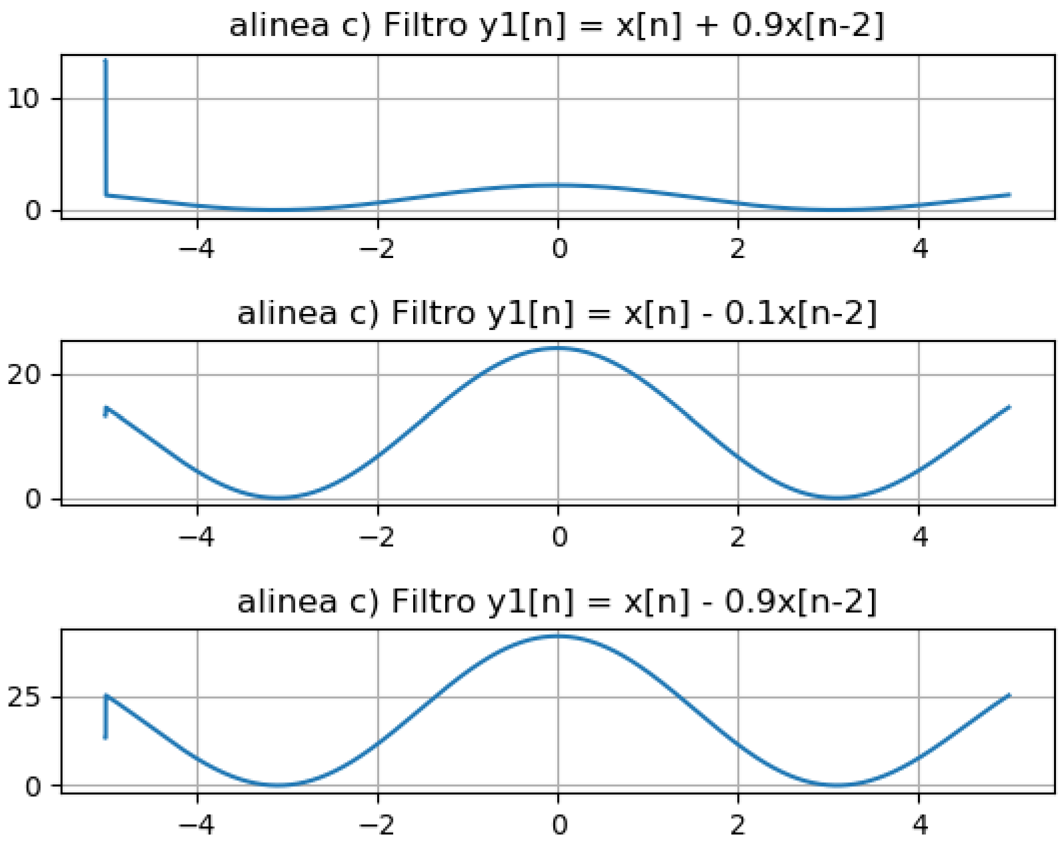


Fig.14 – Código Python

```
#----Y2--c=1 R = -0.9 -----------------------------
plt.figure()
y = ss.lfilter(bk7,ak7,x)
plt.subplot(3,1,1)
plt.title("alinea c) Filtro y2[n] = x[n] + 0.9y2[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.grid()
plt.plot(t,y)
#----Y2--c =1 R = 0.1---------------------------------------
y = ss.lfilter(bk8,ak8,x)
plt.subplot(3,1,2)
plt.grid()
plt.title("alinea c) Filtro y2[n] = x[n] - 0.1y2[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)

#----Y2--c =1 R = 0.9 ---------------------------------
y = ss.lfilter(bk9,ak9,x)
plt.subplot(3,1,3)
plt.grid()
plt.title("alinea c) Filtro y2[n] = x[n] - 0.9y2[n-1]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
plt.show()
```

Fig.15 – Output

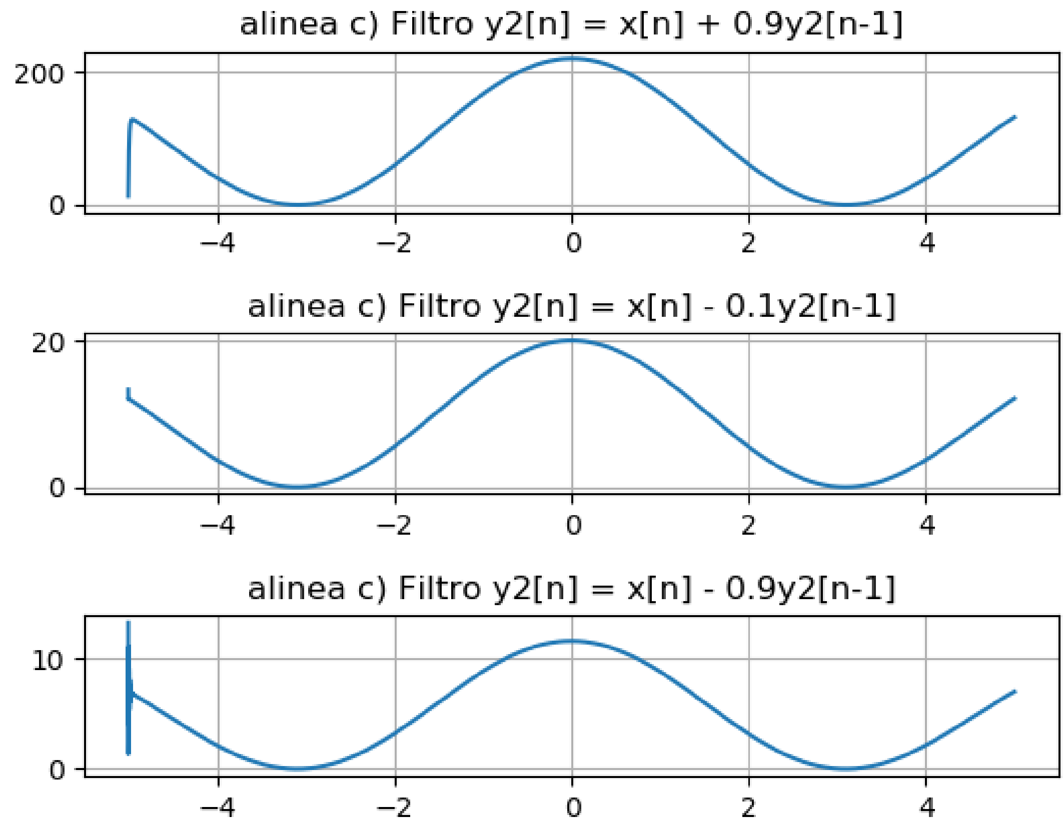


Fig.16 – Código Python

```
#----Y2--c =2 R =-0.9 ---------------------------------
plt.figure()
y = ss.lfilter(bk10,ak10,x)
plt.subplot(3,1,1)
plt.grid()
plt.title("alinea c) Filtro y2[n] = x[n] + 0.9y2[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)

#----Y2--c =2 R = 0.1--------------------------------------
y = ss.lfilter(bk11,ak11,x)
plt.subplot(3,1,2)
plt.grid()
plt.title("alinea c) Filtro y2[n] = x[n] - 0.1y2[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)

#----Y2--c =2 R = 0.9 ---------------------------------
y = ss.lfilter(bk12,ak12,x)
plt.subplot(3,1,3)
plt.grid()
plt.title("alinea c) Filtro y2[n] = x[n] - 0.9y2[n-2]")
plt.subplots_adjust(hspace = 0.75)
plt.plot(t,y)
plt.show()

plt.figure()
```

Fig.17 – Output

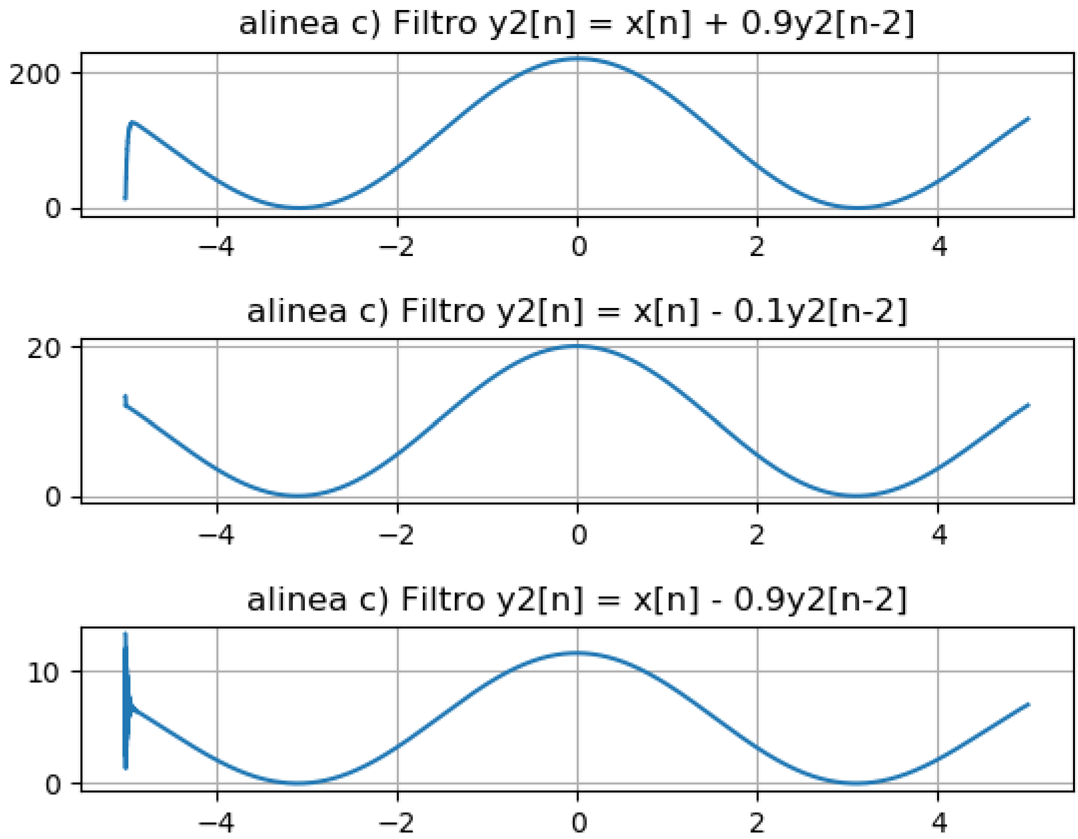
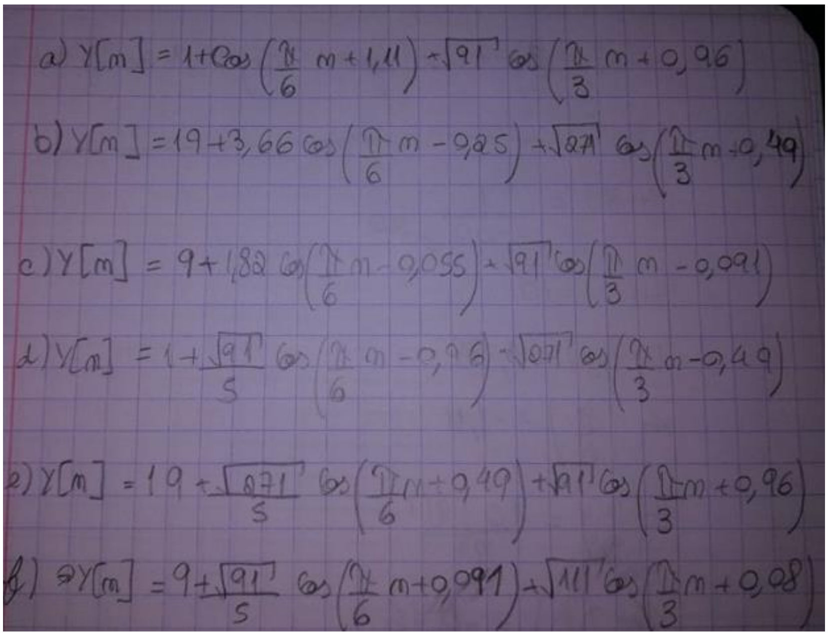


Fig.18– cálculo dos valores de entrada, para a verificação

- Cálculo efetuado para o sistema FIR(para c =1 e 2, r = 0.9, 0.1 e -0.9)



a) $y[n] = 1+\cos\left(\frac{\pi}{6}n + 1{,}11\right) + \sqrt{91}\cos\left(\frac{\pi}{3}n + 0{,}96\right)$

b) $y[n] = 19+3{,}66\cos\left(\frac{\pi}{6}n - 0{,}25\right) + \sqrt{271}\cos\left(\frac{\pi}{3}n + 0{,}49\right)$

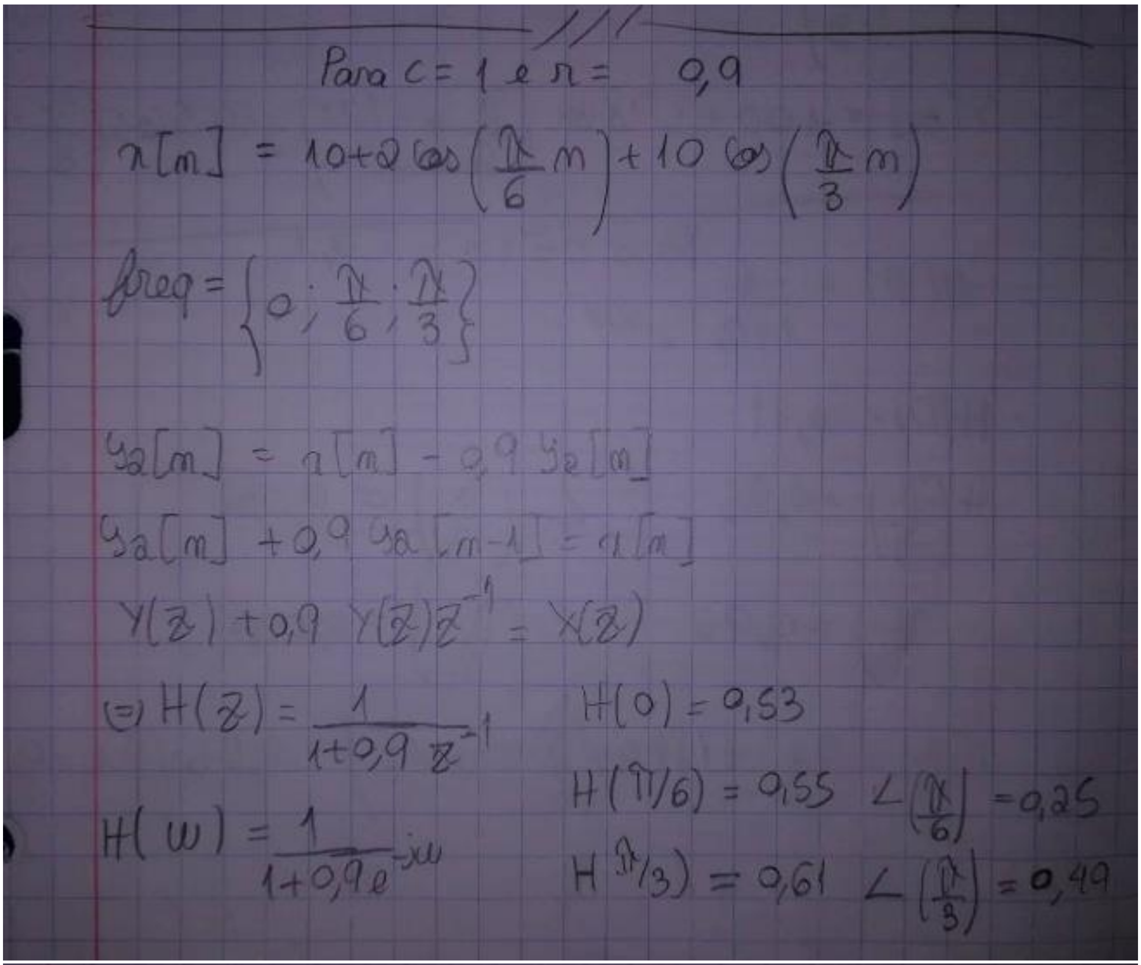
c) $y[n] = 9+1{,}82\cos\left(\frac{\pi}{6}n - 0{,}055\right) + \sqrt{91}\cos\left(\frac{\pi}{3}n - 0{,}091\right)$

d) $y[n] = 1+\frac{\sqrt{91}}{5}\cos\left(\frac{\pi}{6}n - 0{,}96\right) + \sqrt{271}\cos\left(\frac{\pi}{3}n - 0{,}49\right)$

e) $y[n] = 19+\frac{\sqrt{271}}{5}\cos\left(\frac{\pi}{6}n + 0{,}49\right) + \sqrt{91}\cos\left(\frac{\pi}{3}n + 0{,}96\right)$

f) $y[n] = 9+\frac{\sqrt{91}}{5}\cos\left(\frac{\pi}{6}n + 0{,}091\right) + \sqrt{111}\cos\left(\frac{\pi}{3}n + 0{,}08\right)$

- Cálculo efetuado para o sistema IIR(para c =1, r = 0.9, 0.1 e -0.9)



Para $c = 1$ e $r = 0,9$

$$x[n] = 10 + 2\cos\left(\frac{\pi}{6}n\right) + 10\cos\left(\frac{\pi}{3}n\right)$$

$$freq = \left\{0;\ \frac{\pi}{6};\ \frac{\pi}{3}\right\}$$

$$y_a[n] = x[n] - 0,9\, y_a[n]$$

$$y_a[n] + 0,9\, y_a[n-1] = x[n]$$

$$Y(z) + 0,9\, Y(z) z^{-1} = X(z)$$

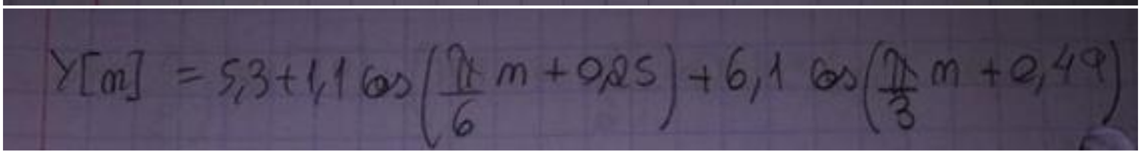
$$\Leftrightarrow H(z) = \frac{1}{1 + 0,9 z^{-1}}$$

$$H(\omega) = \frac{1}{1 + 0,9 e^{-j\omega}}$$

$H(0) = 0,53$

$H(\pi/6) = 0,55 \quad \angle\left(\frac{\pi}{6}\right) = -0,25$

$H(\pi/3) = 0,61 \quad \angle\left(\frac{\pi}{3}\right) = 0,49$



$$Y[n] = 5,3 + 1,1\cos\left(\frac{\pi}{6}n + 0,25\right) + 6,1\cos\left(\frac{\pi}{3}n + 0,49\right)$$

Para $C=1$ e $r=-0{,}9$

$$H(z) = \frac{1}{1-0{,}9\,e^{-j\omega}}$$

$H(0)=10$

$H(\pi/6) = 1{,}99$

$\angle\left(\frac{\pi}{6}\right) = -1{,}11$

$H\left(\frac{\pi}{3}\right) = \frac{10\sqrt{91}}{91}$

$\angle\left(\frac{\pi}{3}\right) = -0{,}96$

$$y[n] = 100 + 3{,}98\cos\left(\frac{\pi}{6}n - 1{,}11\right) + 10{,}5\cos\left(\frac{\pi}{3}n - 0{,}96\right.$$

Para $C=1$ e $r=0{,}1$

$$H(z) = \frac{1}{1+0{,}1\,e^{-j\omega}}$$

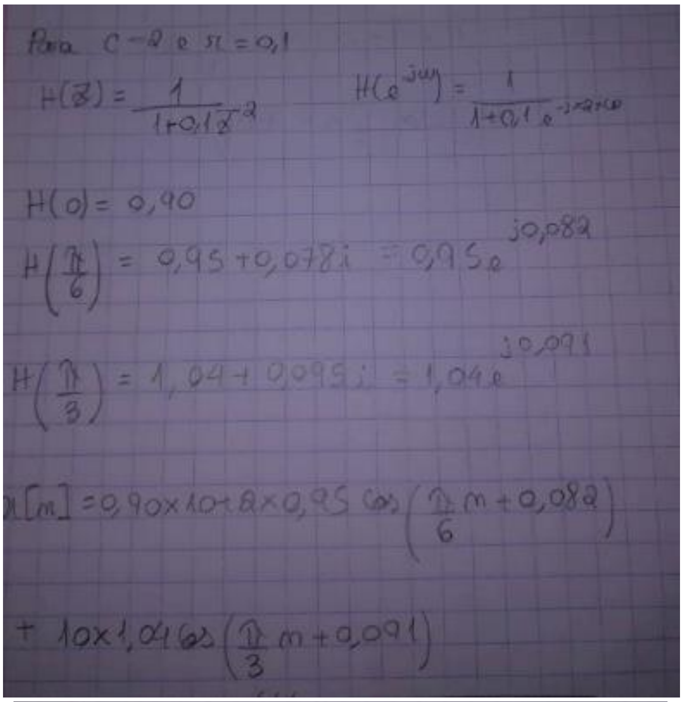
$H(0) = 0{,}91$

$H\left(\frac{\pi}{3}\right) = 0{,}95$ $\quad\angle\left(\frac{\pi}{3}\right) = 0{,}082$

$H\left(\frac{\pi}{6}\right) = 0{,}92$ $\quad\angle\left(\frac{\pi}{6}\right) = 0{,}046$

$$y[n] = 9{,}1 + 1{,}84\cos\left(\frac{\pi}{6}n + 0{,}046\right) + 9{,}5\cos\left(\frac{\pi}{3}n + 0{,}082\right)$$

- Cálculo efetuado para o sistema IIR(para c =2, r = 0.9, 0.1 e -0.9)



Para $c=2$ e $r=0,1$

$H(z) = \frac{1}{1+0,1z^{-2}}$ $\quad H(e^{jw}) = \frac{1}{1+0,1e^{-j2w}}$

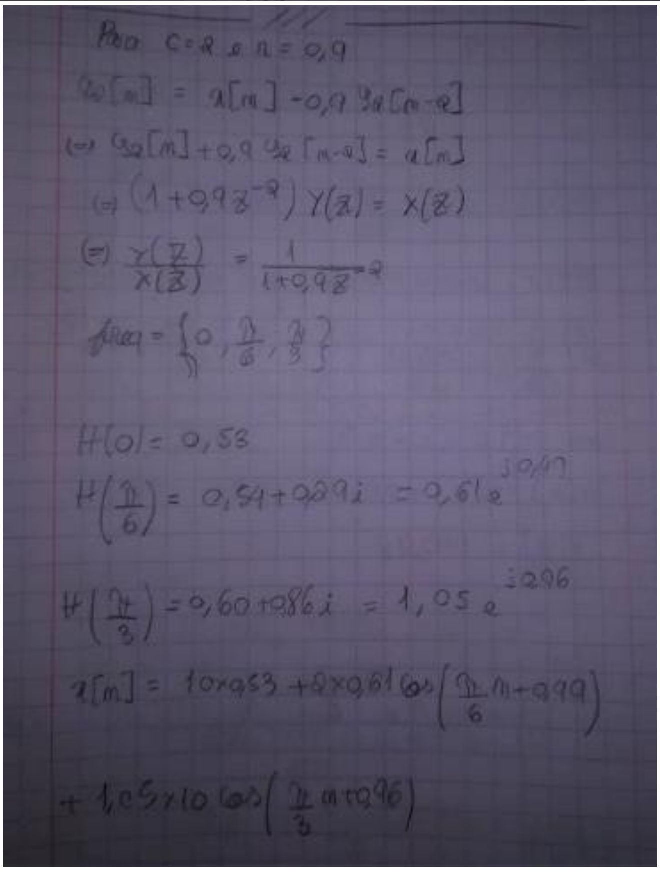
$H(0) = 0,90$

$H\left(\frac{\pi}{6}\right) = 0,95+0,078i = 0,95e^{j0,082}$

$H\left(\frac{\pi}{3}\right) = 1,04+0,095i = 1,04e^{j0,091}$

$x[n] = 0,90\times10 + 2\times0,95\cos\left(\frac{\pi}{6}n+0,082\right)$

$+\ 10\times1,04\cos\left(\frac{\pi}{3}n+0,091\right)$



Para $c=2$ e $r=0,9$

$y_2[n] = x[n] - 0,9\,y_2[n-2]$

$\Leftrightarrow y_2[n] + 0,9\,y_2[n-2] = x[n]$

$\Leftrightarrow (1+0,9z^{-2})\,Y(z) = X(z)$

$\Leftrightarrow \frac{Y(z)}{X(z)} = \frac{1}{1+0,9z^{-2}}$

$freq = \left\{0, \frac{\pi}{6}, \frac{\pi}{3}\right\}$

$H(0) = 0,53$

$H\left(\frac{\pi}{6}\right) = 0,54+0,29i = 0,61e^{j0,47}$

$H\left(\frac{\pi}{3}\right) = 0,60+0,86i = 1,05e^{j0,96}$

$x[n] = 10\times0,53 + 2\times0,61\cos\left(\frac{\pi}{6}n+0,49\right)$

$+\ 1,05\times10\cos\left(\frac{\pi}{3}n+0,96\right)$

Para $c=2$ e $r=-0,9$

$$y_2[n] = x[n] + 0,9\, y_2[n-2]$$
$$\Leftrightarrow y_2[n] - 0,9\, y_2[n-2] = x[n]$$
$$\Leftrightarrow (1-0,9z^{-2})Y(z) = X(z)$$
$$\Leftrightarrow \frac{Y(z)}{X(z)} = \frac{1}{1-0,9z^{-2}}$$

$$H(0) = 10$$
$$H\left(\frac{\pi}{6}\right) = 0,60 - 0,86i = 1,05\, e^{j(-0,61)}$$
$$H\left(\frac{\pi}{3}\right) = 0,54 - 0,29i = 0,61\, e^{j(-0,49)}$$

$$x[n] = 10\times10 + 2\times 1,05 \cos\left(\frac{\pi}{6}n - 0,61\right) + 0,61\times 10 \cos\left(\frac{\pi}{3}n - 0,49\right)$$

Com os cálculos efetuados analíticamente foi possível chegar a conclusão de que os resultados não apresentam semelhança.

Exercício 2

- **Alíneas a), b) e Alínea c)**

Como já referido foi usado a função *scipy.signal.firwin*, com os seguintes parâmetros *(numtaps,cutoff,pass zero,nyq)* onde *numtaps* define a ordem do filtro, para este exercício usou-se 1001, pois se este valor fosse mais pequeno, o gráfico não ia originar o resultado pretendido (originalmente o gráfico origina uma onde curva, mas o resultado que se queria obter era uma de onde quadrada), *cutoff*(ou frequência de corte) é uma lista que define as frequências de corte, neste caso recorreu se por fazer um cálculo, no qual divide-se entre a frequência dada por 2 e a frequência de corte que o enunciado nos fornece, no caso da representação do filtro passa-banda teve-se que definir um array entre essas duas frequências de corte e *pass zero* (ou passa zero) corresponde a uma variável booleana, e como pretendíamos estudar cada tipo de filtro, foi definido como True para filtro passa-alto e

False para filtro passa-alto e passa-banda, e o *nyq*(frequência de nyquist) no qual tem que se dividir a frequência que foi definida por metade.

Fig.18 – Código Python

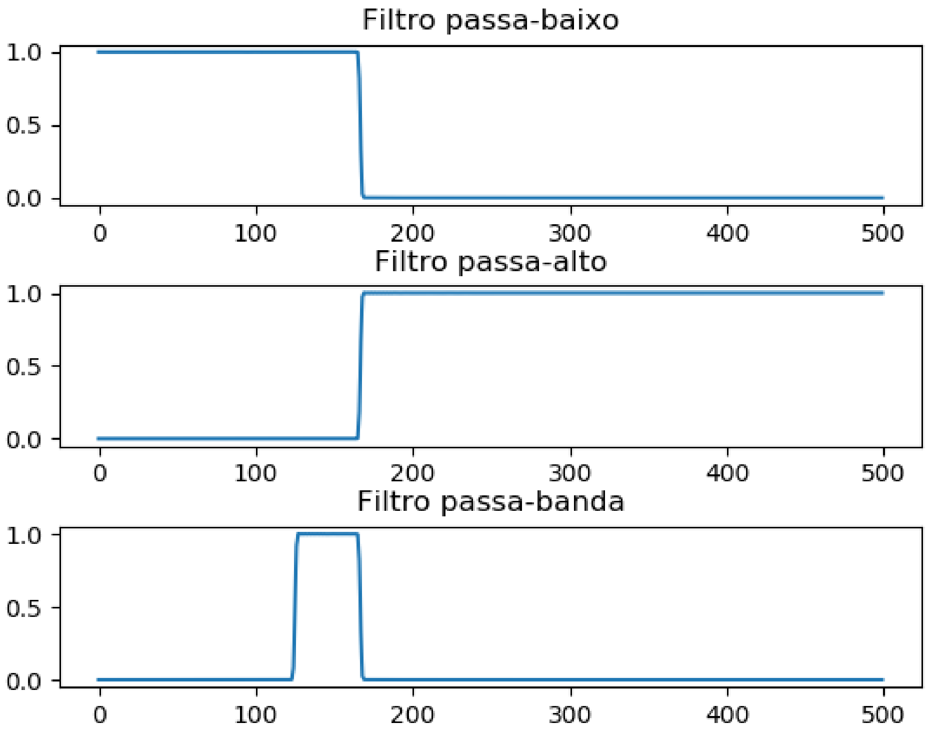
```
plt.figure()
Fs = 1000.0

#a) filtro passa-baixo
bka = ss.firwin(1001, (Fs/2.)/3., pass_zero = True, nyq = Fs/2.0)
w,h = ss.freqz(bka, [1.0])
plt.subplot(311)
plt.subplots_adjust(hspace = 0.5)
plt.title("Filtro passa-baixo")
plt.plot(w / np.pi * Fs/2.0, np.abs(h))

#b) filtro  passa-alto
bkb = ss.firwin(1001, (Fs/2.)/3., pass_zero = False, nyq = Fs/2.0)
w,h = ss.freqz(bkb, [1.0])
plt.subplot(312)
plt.subplots_adjust(hspace = 0.5)
plt.title("Filtro passa-alto")
plt.plot(w / np.pi * Fs/2.0, np.abs(h))

#c) filtro passa-banda entre pi/4 e pi/3
bkc = ss.firwin(1001,[(Fs/2.)/4.,(Fs/2.)/3.], pass_zero = False, nyq = Fs/2.0)
w,h = ss.freqz(bkc, [1.0])
plt.subplot(313)
plt.subplots_adjust(hspace = 0.5)
plt.title("Filtro passa-banda")
plt.plot(w / np.pi * Fs/2.0, np.abs(h))
plt.show()
```

Fig.19-Output



- **Alínea d)**

Para esta alínea era pedido que se construísse gráficos a partir da leitura dos ficheiros wave fornecidos, foram usados dois ficheiros para comparar o seu output. Comparando os gráficos podemos concluir que a frequência de corte vai ser bastante diferente devido ao facto do tipo de frequências que se apresenta.

Fig.20-Código do Python

```
#------------filtro a) do wav Fs,som-----------------------
ya = ss.lfilter(bka,ak,som)
plt.subplot(3,1,1)
plt.subplots_adjust(hspace = 0.5)
plt.plot(ya)


#------------filtro b) do wav Fs,som -----------------------
yb = ss.lfilter(bkb,ak,som)
plt.subplot(3,1,2)
plt.subplots_adjust(hspace = 0.5)
plt.plot(yb)


#------------filtro c)do wav Fs,som -----------------------
yc = ss.lfilter(bkc,ak,som)
plt.subplot(3,1,3)
plt.subplots_adjust(hspace = 0.5)
plt.plot(yc)
plt.show()
```

Fig.21-Output

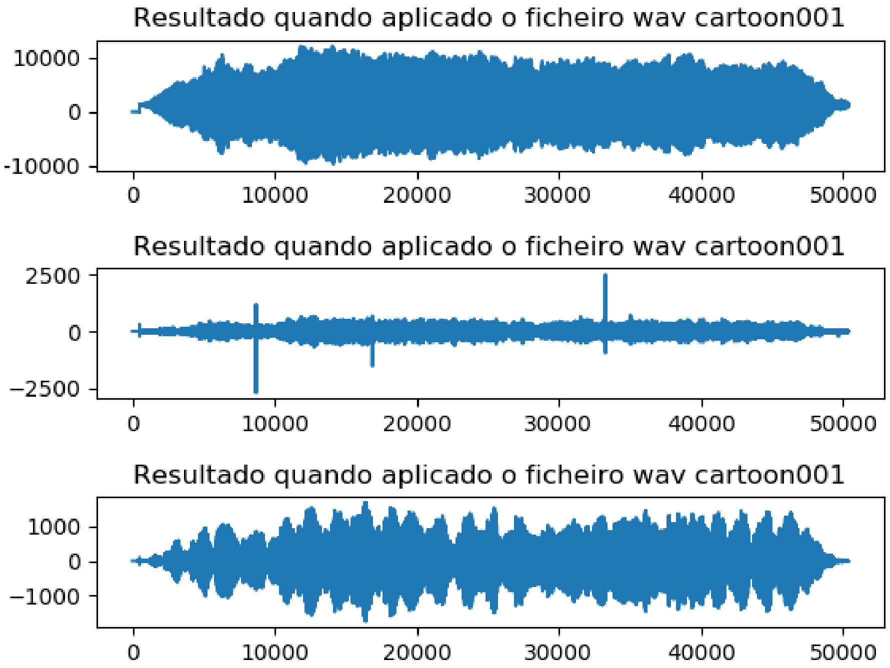
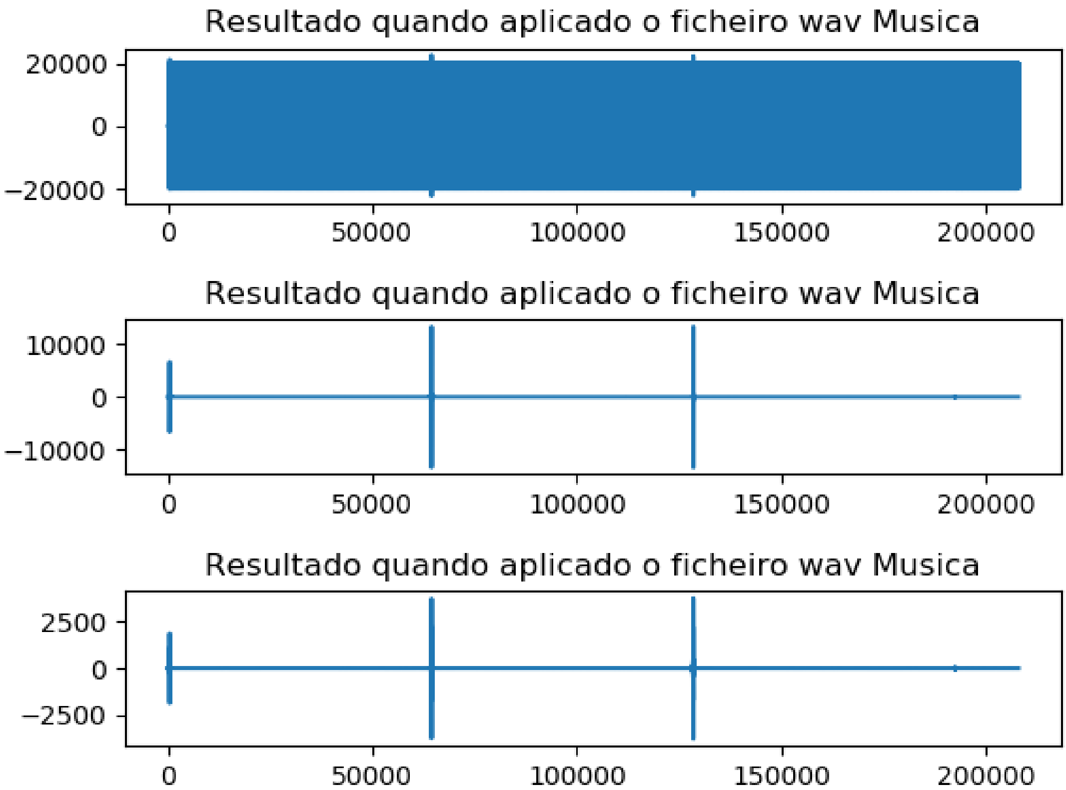


Fig.22-Código do Python

```
#-------------filtro a) do wav Fs2,som2------------------------
ya2 = ss.lfilter(bka,ak,som2)
plt.subplot(3,1,1)
plt.subplots_adjust(hspace = 0.5)
plt.plot(ya2)
#-------------filtro b) do wav Fs2,som2------------------------
yb2 = ss.lfilter(bkb,ak,som2)
plt.subplot(3,1,2)
plt.subplots_adjust(hspace = 0.5)
plt.plot(yb2)
#-------------filtro c) do wav Fs2,som2------------------------
yc2 = ss.lfilter(bkc,ak,som2)
plt.subplot(3,1,3)
plt.subplots_adjust(hspace = 0.5)
plt.plot(yc2)
plt.show()
```

Fig.23-Output



Conclusões

Com este trabalho conseguimos ter uma sólida perceção de como os sinais FIR/IIR funcionam no python, como os podemos calcular e como é possível obter os gráficos dos mesmos. Retirando os coeficientes de cada expressão e usando métodos adequados com as bibliotecas fornecidas, foi possível a representação dos gráficos e o estudo do seu comportamento quando usadas várias frequências.

Apesar de tudo não foi possível confirmar a semelhança entre os resultados dos calculos analíticos e calculos feitos em Python. Pelo que os gráficos obtidos em Python eram diferentes dos gráficos obtidos nas calculadoras através das expressões analíticas. De qualquer maneira os resultados deviam apresentar semelhança com os calculos feitos em Python para os quais foram utilizadas bibliotecas como “numpy”, “scipy.signal” , “matplotlib” , scipy.signal.firwin” , entre outros.

Bibliografia

PowerPoint, III – Amostragem, p. 18

PowerPoint, IV – Sistemas e Filtros IIR, pp.45-47