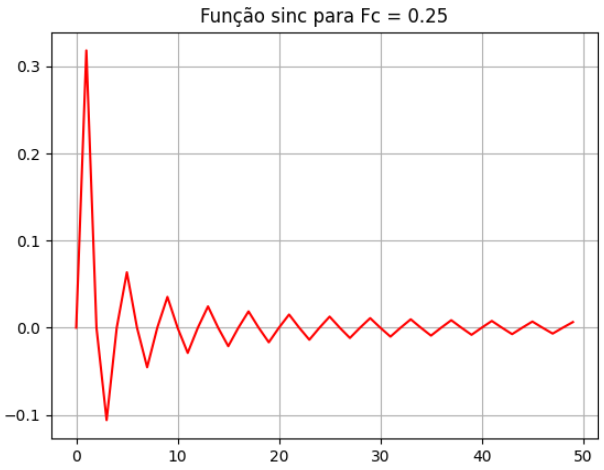
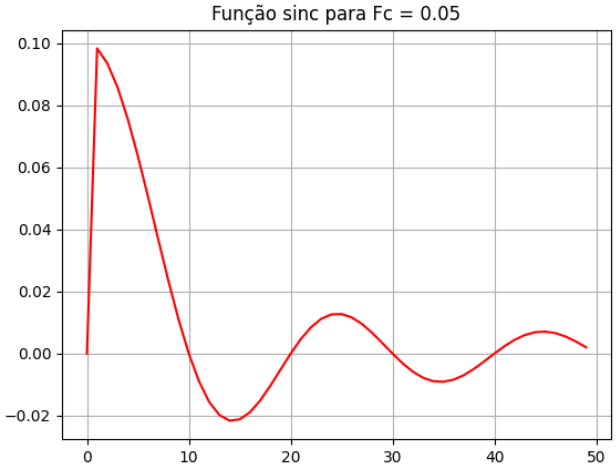
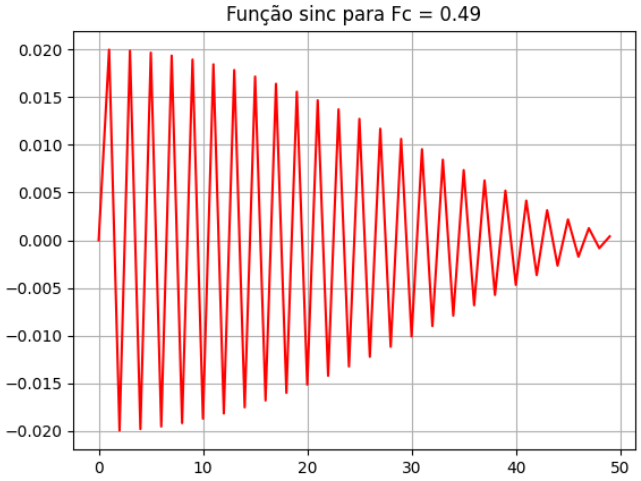
Função Sinc: Para plotar essa função foi utilizado Python, dessa forma foi utilizado 50 amostras para h[i] e diferentes frequências de corte, como podemos ver nas imagens abaixo.

$$h[i] = \frac{\sin(2\pi f_c i)}{i\pi}$$

Os gráficos a seguir são referentes a 0.25, 0.49 e 0.05 Fc respectivamente.







Hamming e Blackman: Para plotar essas janelas foi utilizado Python, além das expressões fornecidas no capítulo 16.

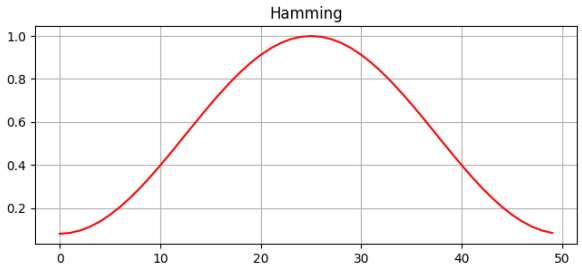
EQUATION 16-1
The Hamming window. These windows run from $i = 0$ to M, for a total of $M + 1$ points.

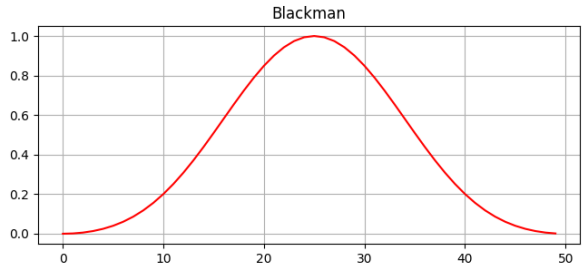
$$w[i] = 0.54 - 0.46\cos(2\pi i/M)$$

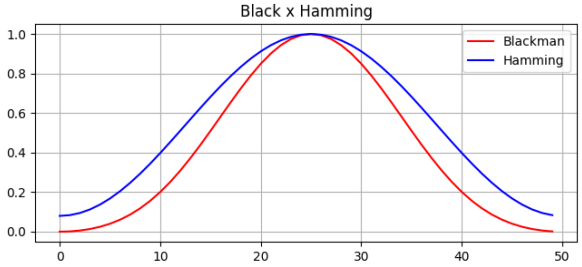
EQUATION 16-2
The Blackman window.

$$w[i] = 0.42 - 0.5\cos(2\pi i/M) + 0.08\cos(4\pi i/M)$$

Dessa forma utilizando M = 50 temos a seguinte resposta do programa:

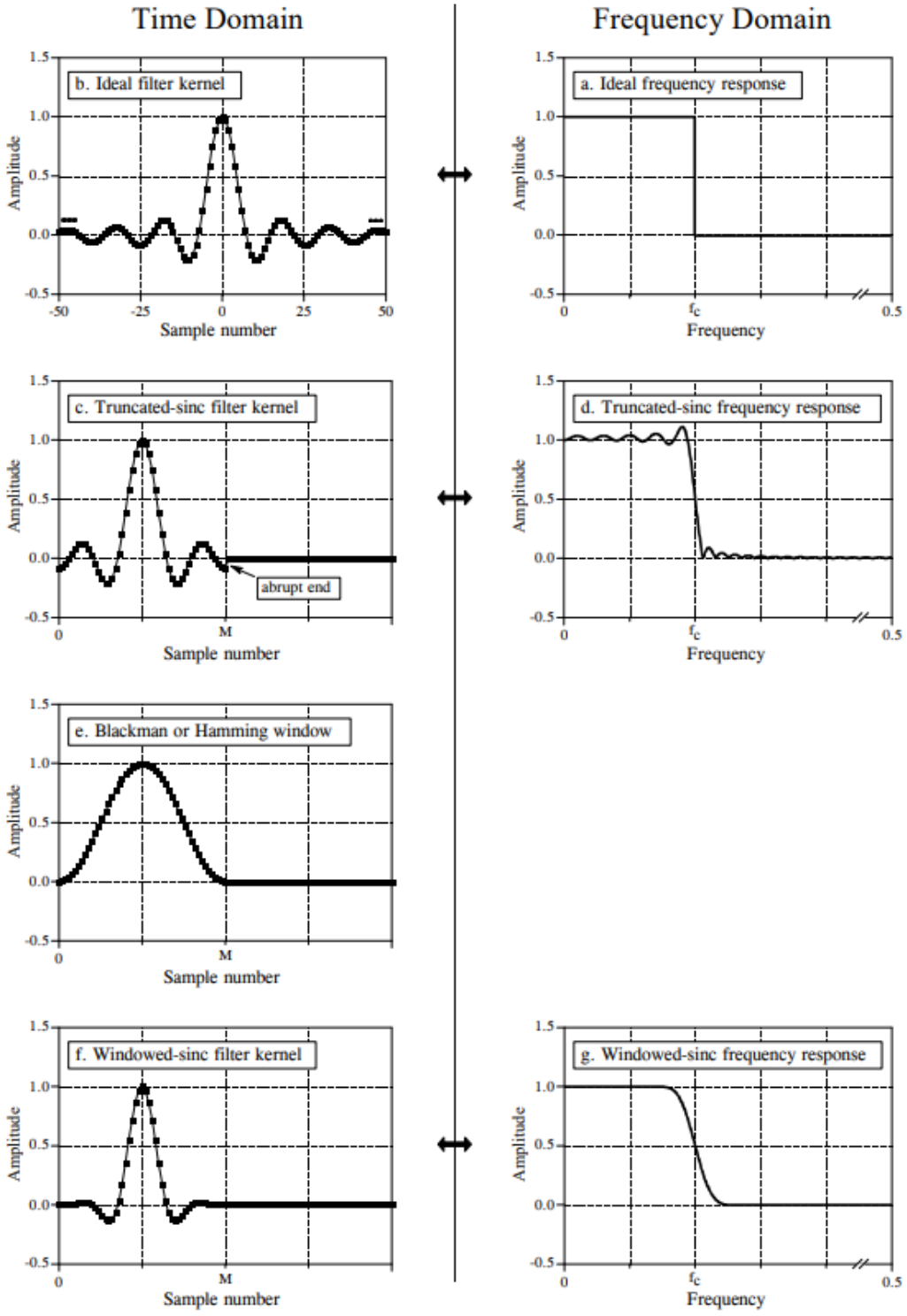






Onde podemos ver o comparativo entre cada janela, que nos indica que a janela Blackman possui uma melhor atenuação de na borda de descida.

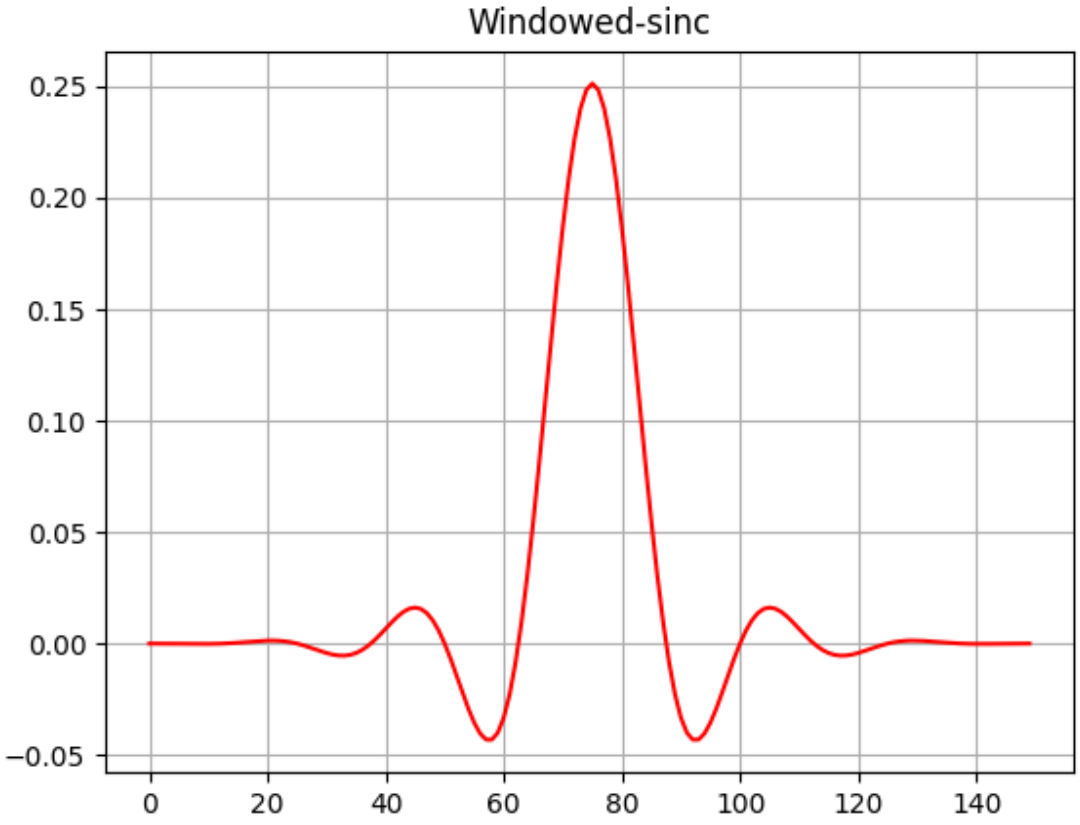
A resposta ideal na frequência é mostrada no item (a), com sua respectiva resposta no domínio do tempo, o que nos exemplifica uma função sinc. Porém como sinc é infinitamente longo, é necessário um truncamento para ser utilizado em computadores, dessa forma após realizar o truncamento em (c), podemos ver quem no domínio da frequência (d) ocorrem alterações indesejáveis. A solução para esse problema é realizar o truncamento com uma função de janela como é mostrado em (f), dessa forma podemos ver que na resposta da frequência a saída (g) se demonstra suave e com um bom comportamento.



Filtro windowed-sinc: Desenvolvido em python, sua função é uma mistura entre a função sinc e a janela Blackman, como podemos ver na expressão abaixo, novamente o valor de M precisa ser um número par e Fc entre 0 à 0.5, além disso quando i for igual a M/2 uma segunda função é utilizada sendo ela h[i] = 2.π.Fc.K, além disso utilizamos da seguinte expressão:

$$h[i] = K \frac{\sin(2\pi f_C (i - M/2))}{i - M/2} \left[0.42 - 0.5\cos\left(\frac{2\pi i}{M}\right) + 0.08\cos\left(\frac{4\pi i}{M}\right) \right]$$

Dessa forma, utilizando de M = 150 (Par) e Fc igual a 0.04, geramos o seguinte gráfico da resposta do filtro, onde podemos analisar que centralizado no valor 75 temos a maior amplitude do filtro, como esperado visto que ele usara a segunda função para evitar problemas de divisão por zero.



Filtro PB: Utilizando da tabela 16-1 do capítulo, um filtro passa baixa foi desenvolvido, dessa forma aplicamos os conceitos de geração de coeficientes da tabela, e em seguida testamos o mesmo com o programa média móvel desenvolvido em C anteriormente.

```
100 'LOW-PASS WINDOWED-SINC FILTER
110 'This program filters 5000 samples with a 101 point windowed-sinc filter,
120 'resulting in 4900 samples of filtered data.
130 '
140 DIM X[4999]                    'X[ ] holds the input signal
150 DIM Y[4999]                    'Y[ ] holds the output signal
160 DIM H[100]                     'H[ ] holds the filter kernel
170 '
180 PI = 3.14159265
190 FC = .14                       'Set the cutoff frequency (between 0 and 0.5)
200 M% = 100                       'Set filter length (101 points)
210 '
220 GOSUB XXXX                     'Mythical subroutine to load X[ ]
230 '
240 '                              'Calculate the low-pass filter kernel via Eq. 16-4
250 FOR I% = 0 TO 100
260   IF (I%-M%/2) = 0 THEN H[I%] = 2*PI*FC
270   IF (I%-M%/2) <> 0  THEN H[I%] = SIN(2*PI*FC * (I%-M%/2)) / (I%-M%/2)
280   H[I%] = H[I%] * (0.54 - 0.46*COS(2*PI*I%/M%) )
290 NEXT I%
300 '
310 SUM = 0                        'Normalize the low-pass filter kernel for
320 FOR I% = 0 TO 100              'unity gain at DC
330   SUM = SUM + H[I%]
340 NEXT I%
350 '
360 FOR I% = 0 TO 100
370   H[I%] = H[I%] / SUM
380 NEXT I%
390 '
400 FOR J% = 100 TO 4999           'Convolve the input signal & filter kernel
410   Y[J%] = 0
420   FOR I% = 0 TO 100
430     Y[J%] = Y[J%] + X[J%-I%] * H[I%]
440   NEXT I%
450 NEXT J%
460 '
470 END
```

TABLE 16-1

Dessa forma, aplicando o filtro para 100 pontos de windowed-sinc, salvando os mesmo em arquivo .dat, para ser lido posteriormente no programa Média móvel desenvolvido na linguagem C. O teste foi feito em cima de sweep, onde após aplicado o filtro ficou da seguinte maneira: