UNIVERSIDADE FEDERAL DE SANTA CATARINA
Departamento de Engenharia Elétrica
EEL 7521 – Processamento Digital de Sinais
Professor – José Carlos M. Bermudez
Alunos – Romano Sabetzki Weirich

Projeto de Filtros FIR – Método de Janelas

INTRODUÇÃO

Sinais estão presentes em nossa vida e desempenham um papel importante em nosso cotidiano. Voz, música, imagem e vídeo são exemplos de sinais que estão constantemente ao nosso redor. Sinais, por definição, contêm informação e o objetivo do processamento de sinais é extrair essa informação e torná-la disponível para usuários ou outros sistemas.

Uma ação comumente empregada no processamento digital de sinais é a filtragem. Justamente por isso, o projeto de filtros digitais é o objetivo da disciplina. A proposta deste trabalho é projetar o código em Matlab que seja implementável em DSP e modele o comportamento de um filtro passa baixas do tipo FIR (do inglês *finite impulse response*). Um filtro com resposta infinita ao impulso (*infinite impulse response*, ou IIR) será discutido e projetado em um trabalho futuro.

O projeto dos filtros aproveitou o método de janelas, usando quatro destas: a janela Retangular, a de Hamming, a de Kaiser e a de Blackman. Além disso, utilizou-se precisão finita nas operações realizadas pelos somadores e nos coeficientes de todos os multiplicadores. Três quantizações estão disponíveis, com oito, dez e dezesseis *bits*. Doze filtros foram projetados, compreendendo todas as combinações de janela e quantização

As especificações, informações relacionadas a filtros FIR, ao quantizador, e o projeto, desenvolvido passo a passo, serão apresentados nas próximas seções.

ESPECIFICAÇÕES

Filtros digitais são especificados em termos da atenuação mínima na banda de rejeição (A_S), do *ripple* máximo na banda passante (A_P) e, obviamente, pelas frequências que limitam essas regiões do espectro. A frequência limite da banda de rejeição será chamada F_P e a frequência limite de banda de rejeição será F_S. Usualmente, esses valores são apresentados em dB, mas é possível convertê-los para uma escala linear (A_S se δ_S torna e A_P, δ_P). Essas relações são descritas nas expressões abaixo.

$$A_P = 20\log_{10}\left(\frac{1+\delta_P}{1-\delta_P}\right) \Rightarrow \delta_P = \frac{10^{A_P/20}-1}{10^{A_P/20}+1}$$

$$A_S = -20\log_{10}(\delta_S) \Rightarrow \delta_S = 10^{-A_S/20}$$

A magnitude da resposta em frequência de um filtro digital é comumente especificada de forma gráfica, usando um “gabarito”. Um exemplo genérico para um filtro passa baixas é apresentado na Figura 1. Nesta imagem, se δ_S for nulo, a linha tracejada corresponde à resposta de um filtro ideal. De acordo com essas especificações o valor da magnitude de $H(e^{j\omega})$ deve estar de acordo com as expressões seguintes.

$$1-\delta_P \le \left|H\left(e^{j\omega}\right)\right| \le 1+\delta_P\ ,\ 0 \le \omega \le \omega_P$$

$$\left|H\left(e^{j\omega}\right)\right| \le \delta_S, \omega_S \le \omega \le \pi$$

$$\omega_P = \frac{2\pi F_P}{F_A}\ ,\ \omega_S = \frac{2\pi F_S}{F_A}$$

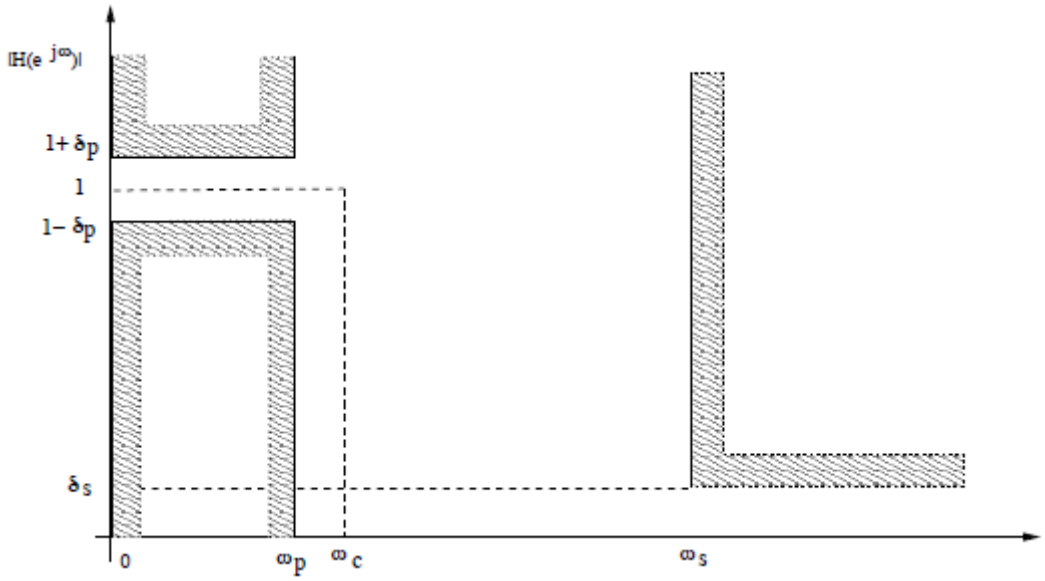


Figura 1 – Gabarito para verificação da resposta em frequência de um filtro digital.

As especificações que os filtros deverão atender estão contidas na Tabela 1. Observar se os valores da magnitude da função de transferência do filtro projetado, avaliada no domínio da frequência, “coube” nas demarcações do gabarito construído usando a Tabela 1 será o método de avaliação dos coeficientes do filtro.

Tabela 1 – Tabela de especificações.

Limite superior de banda passante:	F_P=10 kHz
Limite inferior de banda de rejeição:	F_S=15 kHz
Frequência de amostragem:	F_A=44.1 kHz
Ripple na banda passante:	A_P=0,1 dB
Atenuação mínima na banda de rejeição:	A_S=40 dB

FILTROS FIR

Filtros FIR têm resposta finita ao impulso e são excelentes para muitas aplicações, pois sempre são estáveis, operam com fase linear e são facilmente implementáveis em DSP.

A estrutura de um filtro FIR é composta por: *delays*, blocos multiplicadores (estes multiplicam as amostras por coeficientes definidos e constantes) e por somadores. Um diagrama de blocos é apresentado na Figura 2.

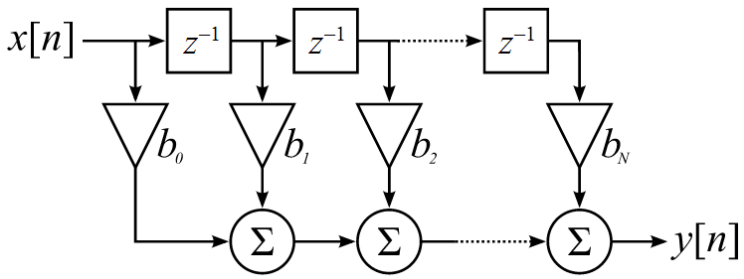


Figura 1 – Estrutura de um filtro FIR.

Os valores das amostras da função de transferência, h[n], do filtro implementado são os coeficientes b_n na Figura 1. O sinal y[n] é a saída do sistema e é igual ao sinal de entrada (x[n]) filtrado. A ordem de um filtro FIR é igual a N. A resposta ao impulso de um filtro comumente tem a forma de um “sinc” e esta característica será explorada para avaliar o filtro. Um impulso será aplicado à sua entrada e espera-se obter um sinal tipo “sinc” em sua saída.

O PROJETO

Inicia-se o projeto determinando qual seria a função de transferência de um filtro ideal, ou seja um filtro capaz de selecionar sem distorção determinada faixa de frequência e rejeitar, absolutamente, o restante do espectro. No caso de um filtro passa baixas digital, o ângulo ω_C associado à frequência de corte F_C define essa faixa. A resposta em frequência de um filtro com essas características será:

$$H\left(e^{j\omega}\right)=\begin{cases}1, banda\ passante\\ 0, banda\ de\ rejeição\end{cases}\Rightarrow H\left(e^{j\omega}\right)=\begin{cases}1\ ,\ |\omega|\le\omega_C\\ 0\ ,\ \omega\le\omega\le\pi\end{cases}\Rightarrow H\left(e^{j\omega}\right)=rect_{\omega_c}[n]$$

Nesse caso, $H(e^{j\omega})$ é nada mais que a função retangular de largura ω_C. A transformada inversa de Fourier para sinais discretos de $H(e^{j\omega})$ é conhecida. Assim, a resposta ao impulsdo do filtro passa baixas ideal será:

$$h[n]=\frac{\omega_C}{\pi}sinc[\omega_C n]\ ,\ -\infty<n<\infty$$
$$h[n]\neq 0\ para\ n<0$$

Enfim, percebe-se que a h[n] é não-causal e tem duração infinita. De forma que não é implementável. Hipoteticamente, se fosse possível criar um filtro FIR com infinitos coeficientes, *delays*, avanços, etc, tal estrutura seria capaz de representar h[n]. Apesar disso parecer uma grande barreira no fluxo de projeto, esses dois problemas são facilmente contornáveis pelo procedimento apresentado a seguir.

Truncando h[n] entre –M e M, e deslocando-a M amostras para a direita é possível criar $h_{causal}[n]$, que é a função de transferência um filtro passa baixas causal e com resposta ao impulso finita. Vale destacar que $h_{causal}[n]$ é facilmente implementável em uma estrutura de filtros FIR. Os valores de $h_{causal}[n]$ para cada n e, consequentemente, o comportamento em frequência desse novo filtro são funções de ω_C e de M. Essas serão as variáveis de projeto que devem ser escolhidas de forma que as especificações sejam atingidas. A expressão que modela $h_{causal}[n]$ será:

$$h_{causal}[n]=\begin{cases}\frac{\omega_C}{\pi}sinc\left[\omega_C(n-M)\right], 0\le n\le 2M\\ 0, demais\ valores\ de\ n\end{cases}$$

Uma vez encontrada $h_{causal}[n]$ que atenda às especificações os valores de suas amostras serão os coeficientes b do filtro FIR que deverá atuar sobre o sinal de entrada. Dessa forma, o filtro terá 2M+1 coeficientes e 2M+1 atrasos, terá ordem N igual a 2M e frequência de corte igual a ω_C.

James Kaiser desenvolveu um método de estimar o valor da ordem de um filtro FIR passa baixas a partir de suas especificações originais. A fórmula, chamada Fórmula de Kaiser é dada abaixo.

$$N\simeq\frac{-20\log_{10}(\sqrt{\delta_P\delta_S})-13}{14.6(\omega_S-\omega_P)/2\pi}$$
$$N\simeq 17.7599\Rightarrow N=18$$

O arquivo “filtro_ideal_mais_filtro_causal.m” implementa e testa as funções de transferência h[n] e $h_{causal}[n]$. Neste teste supôs-se ω_C igual a 0,5 πrad. Para simular a não-causalidade e o comprimento infinito da resposta ao impulso do filtro ideal, considerou-se n entre -1000 e 1000. Já para o filtro com resposta causal, e N igual a 18, obtido a partir da Fórmula de Kaiser.

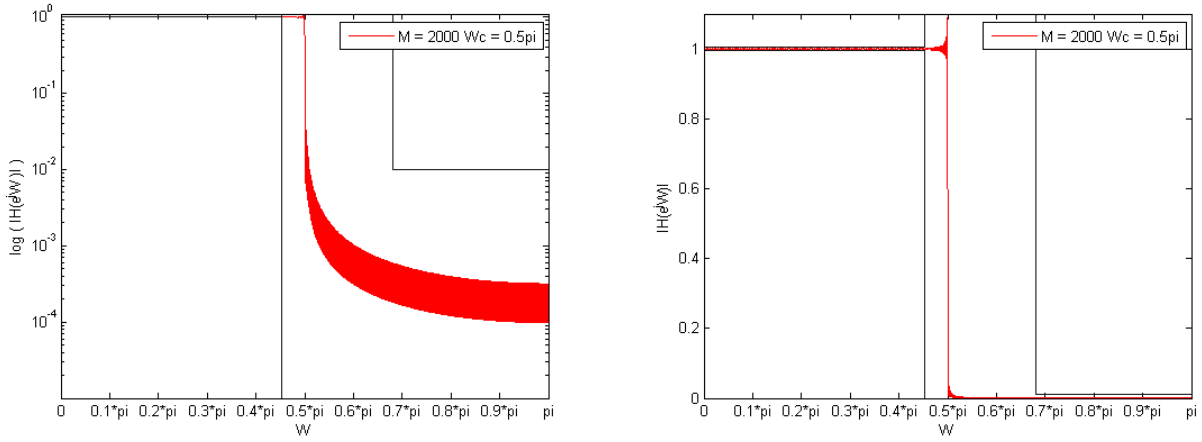
A transformação de domínio (tempo para frequência) foi feita pela instrução nativa da *toolbox* de processamento digital de sinais do Matlab “DTFT”, que retorna a transformada de Fourier para sinais discretos do vetor de entrada (H), e um vetor contendo os pontos do espectro de frequências de -π a π (W). Adicionalmente, a função “freqplot”, criada ao longo deste trabalho, opera em conjunto com “DTFT” e plota a resposta em frequência da função desejada em cima do gabarito de verificação criado pelas especificações. Existem duas opções para a escala do eixo Y no gráfico: a linear e logarítmica. A possibilidade de escolha é interessante visto que a escala logarítmica se mostrou melhor para verificar $H(e^{j\omega})$ na região da banda de rejeição e a linear para fazer o mesmo na banda passante. Os argumentos de entrada se freqplot são: a resposta ao impulso,

o eixo x do gráfico, W_P, W_S, δ_P, δ_S, a escala do eixo Y ('linear' ou 'log'), M e ω_C. Esses dois últimos são usados para criar a legenda da imagem, a fazendo carregar mais informação.

Estas duas funções serão usadas novamente no *script* principal, entretanto só serão citadas, por já terem sido devidamente esclarecidas acima.

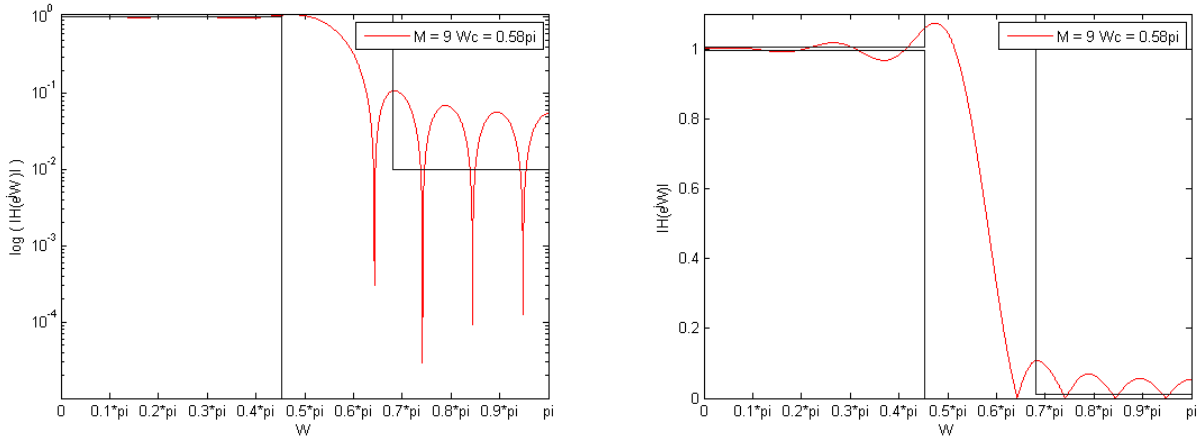
Adicionalmente existe o arquivo "teste_freqplot.m" um *script* para testar essa função separadamente. Além de facilitar o desenvolvimento de "freqplot", esse arquivo de teste possibilita ao leitor alterar a função original para outra aplicação, testando-a isoladamente.

Os resultados obtidos por "filtro_ideal_mais_filtro_causal.m" referentes ao desempenho do filtro ideal aproximado e do filtro causal de ordem estimada pela fórmula de Kaiser são apresentados nas Figuras 3 e 4 e serão discutido na sequência.



(a) (b)

Figura 3 – Resposta em frequência do filtro ideal em escala (a) logarítmica e (b) linear.



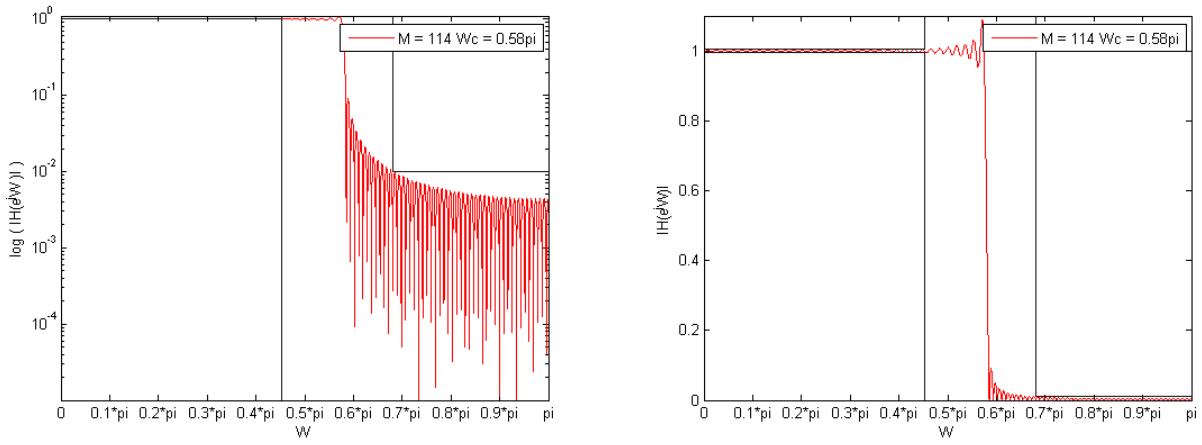
(a) (b)

Figura 4 – Resposta em frequência do filtro causal com M igual a 9 e ω_C igual a 0,5π rad em escala (a) logarítmica e (b) linear.

Analisando a Figura 3b percebe-se que a resposta em frequência obtida se assemelha bastante a de um filtro ideal. E, claramente, h[n] cumpre as especificações iniciais. Por outro lado a Figura 4a mostra que a DTFT de $h_{causal}[n]$ não "coube" no gabarito na região da banda rejeição, e a Figura 4b mostra que, $h_{causal}[n]$ não atende às especificações na banda passante.

O próximo passo foi ir em busca de uma dupla de valores para ω_C e M que fizesse o filtro causal atender as especificações com a menor ordem possível. Este foi um processo iterativo realizado por um laço presente na sessão "SWEEP" do arquivo "filtro_ideal_mais_filtro_causal.m". Se M assumir o valor 114 e ω_C for 0,58π rad, $h_{causal}[n]$ representa um filtro que respeita às especificações, como a Figura 5 mostra.

A ordem do filtro, nessas condições é 228, o que é bem maior do que a estimativa de Kaiser, entretanto as especificações nesse caso são bastante restritas, e a influência de M e ω_C em $H_{causal}(e^{j\omega})$ não é altamente trivial, de forma que é difícil estimar com propriedade a ordem mínima do filtro.



(a) (b)

Figura 5 – Resposta em frequência do filtro causal com M igual a 114 e ω_C igual a 0,58π rad em escala (a) logarítmica e (b) linear.

Finalmente, o filtro causal foi implementado para filtrar um sinal de entrada. Por facilidade, escolheu-se o impulso unitário para ser o sinal de entrada, se forma que a saída deveria ser um "sinc" de duração 229, centrado em 114 e multiplicado por 0,58 (ω_C dividido por π), ou seja $h_{causal}[n]$.A Figura 6 mostra o resultado dessa implementação, que foi bastante satisfatório.

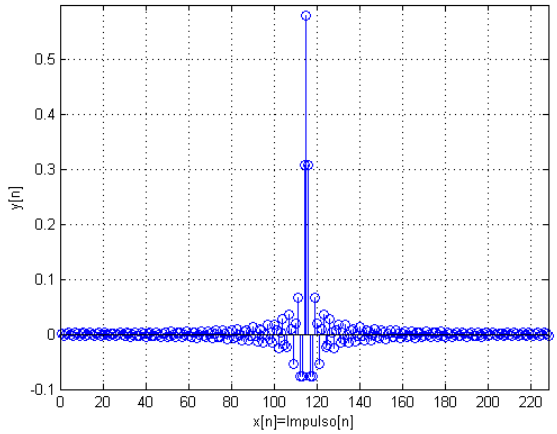


Figura 6 – Sinal de saída para um impulso filtrado por $h_{causal}[n]$.

Vale dizer que para modelar a estrutura de um filtro FIR o arquivo "fitlrodigital.m" foi criado. Usando, nessa ordem, o sinal de entrada (x) e a resposta ao impulso do filtro projetado (h), como argumentos, essa função retorna y, o sinal x filtrado. A implementabilidade em DSP foi levada em conta ao longo do desenvolvimendo de "filtrodigital.m" e seu funcionamento pode ser entendido como a convolução entre x[n] e h[n]. Há também o arquivo de testes "testefiltrodigital.m" que, como o nome sugere , testa "filtrodigital.m", entretanto o teste é realizado supondo $h_{causal}[n]$ com M igual a 114 e ω_C igual a 0,58π rad, portanto a resposta seria igual à presente na Figura 6, de forma que não é interessante repeti-la. Mesmo assim, o leitor pode fazê-lo caso deseje alterar a função para otimizá-la.

Por enquanto os valores das amostras da resposta ao impulso e, portanto, dos coeficientes do filtro, os valores das amostras do sinal de entrada e os cálculos realizados pelos somadores na implementação puderam assumir qualquer valor que o Matlab consiga operar. Entretanto esse trabalho foi desenvolvido pensando na aplicação em DSP, o que exige precisão finita. Por esse motivo o arquivo "quantizafor.m" foi criado. Suas entradas são um vetor (in) e o número de *bits* que

se deseja usar na quantificação (n). A saída é outro vetor (out) de mesmo comprimento de in. As amostras de out são iguais às amostras de in quantizadas com n *bits*. Desses n *bits* um deles é destinado à determinação de sinal e os outros à mantissa. Há saturação caso a entrada seja menor do que -1 ou maior do que 1-Δ, (Δ vale 2^{1-n}). O usuário pode fazer nbits assumir o valor de 128 para que este bloco retorne o sinal de entrada com precisão infinita, o equivalente a não fazer a quantização. Essa opção se mostrou interessante em algumas situações ao longo do projeto, facilitando a reutilização de laços de testes na determinação de parâmetros. As quatro opções principais de nbits são, por tanto, 8; 10; 16 e 128.

O quantizador também foi testado. O arquivo "testequantizador.m" insere uma rampa como entrada, calcula a saída quantizada em 3 *bits*, mas essa característica é facilmente customizável (basta alterar a linha 3) e essas duas curvas em azul e vermelho, respectivamente. A Figura 6 é uma cópia do gráfico gerado em "testequantizador.m".

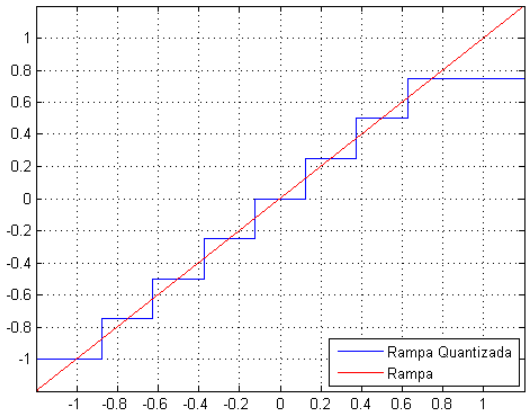
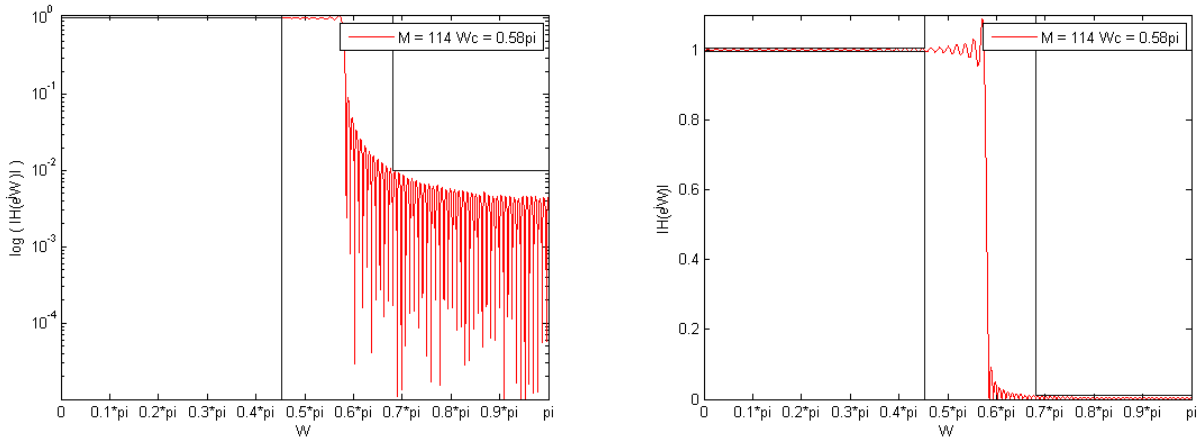


Figura 6 – Quantização de uma rampa em 3 *bits* feita pelo bloco "quantizador.m".

Era necessário gerar $h_{Q16}[n]$, $h_{Q10}[n]$ e $h_{Q8}[n]$. Essas três respostas ao impulso nada mais são que $h_{causal}[n]$ quantizada para 16, 10 e 8 *bits*, respectivamente. Feito isso, o próximo passo era checar se cada resposta em frequência associada cumpria as especificações. Isso é feito pelo arquivo "filtro_quantizado.m". Apenas funções já mencionada ao longo deste documento são usadas nesse arquivo. Na linha 37 o número de *bits* da quantização é setado. A sessão "FILTRO QUANTIZADO" retorna gráficos da resposta em frequência com o eixo Y em escala linear e logarítmica.

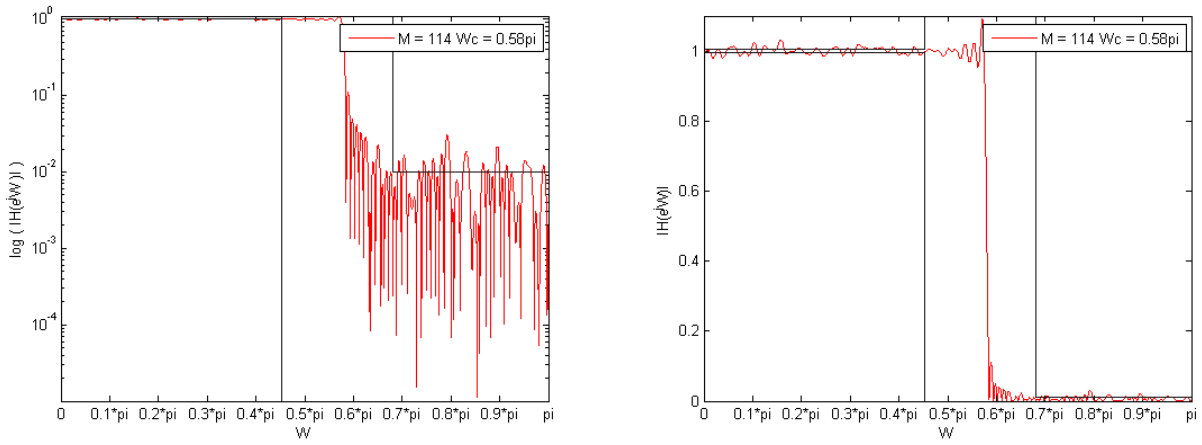
As Figuras 7 a 9 apresentam $H_{Q16}(e^{j\omega})$, $H_{Q10}(e^{j\omega})$ e $H_{Q8}(e^{j\omega})$, nessa ordem, plotadas sobre o gabarito de verificação.



(a) (b)

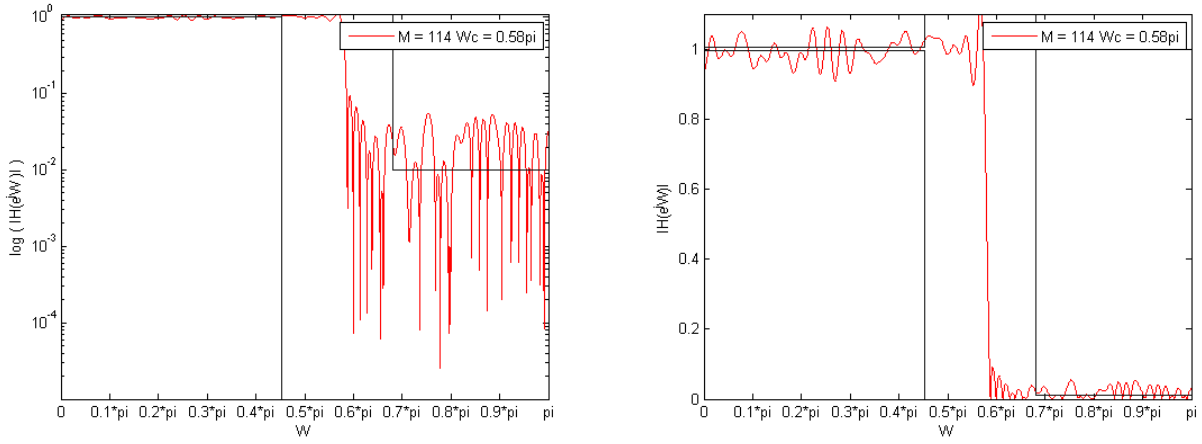
Figura 7 – Resposta em frequência de $h_{causal}[n]$ (M igual a 114 e ω_C igual a 0,58π rad) quantizada com 16 *bits* em escala (a) logarítmica e (b) linear.

A partir da Figura 7 percebe-se que o filtro associado à $h_{Q16}[n]$ cumpre as especificações iniciais com M for 114 e ω_C igual a 0,58π rad. É válido destacar que, nesse caso, um filtro com precisão infinita e um quantizado em 16 *bits* obtiveram desempenhos iguais. O interessante é que este último pode ser implementado em um DSP convencional.



(a) (b)

Figura 8 – Resposta em frequência de $h_{causal}[n]$ (M igual a 114 e ω_C igual a 0,58π rad) quantizada com 10 *bits* em escala (a) logarítmica e (b) linear.



(a) (b)

Figura 9 – Resposta em frequência de $h_{causal}[n]$ (M igual a 114 e ω_C igual a 0,58π rad) quantizada com 8 *bits* em escala (a) logarítmica e (b) linear.

Em contrapartida, as Figuras 8 e 9 mostram que quantizar $h_{causal}[n]$ com 8 e 10 *bits* insere muita distorção em relação às especificações. Assim, outro processo iterativo para determinação de M e ω_C se fará necessário.

O arquivo "filtro_quantizado.m" realiza a tarefa citada acima na sessão "SWEEP" da mesma forma que "filtro_ideal_mais_filtro_causal.m". Mesmo com muitas iterações não se conseguiu achar nenhum M que fizesse com que $h_{Q10}[n]$ ou $h_{Q8}[n]$ atendesse às especificações. Os filtros com ordens muito elevadas (N a partir 600) obtiveram os piores resultados.

Todo o procedimento realizado até agora usou a janela retangular para truncar h[n]. Entretanto, existem outras funções com transições mais suaves e que obtém um desempenho em frequência melhor, possibilitando o projeto de filtros com ordens menores. Este trabalho considerou as janelas de Hamming, Kaiser e Blackman. Veja abaixo a expressão de cada uma delas.

$$w_{Hamming}[n] = 0.54 - 0.46\cos\left(2\pi\frac{n}{N}\right),\ 0 \le n \le N$$

$$w_{Kaiser}[n]=\begin{cases}\dfrac{I_o\pi\beta\sqrt{1-\left(\dfrac{2n}{N-1}-1\right)^2}}{I_o(\pi\beta)}, & 0\le n\le N-1 \;\; \beta=0.1102*(A_S-8.7);\\ 0, \; demais\; valores\; de\; n\end{cases}$$

$$w_{Blackman}[n]=0.42-0.5\cos\left(\frac{2\pi n}{(N-1)}\right)+0.08\cos\left(\frac{4\pi n}{(N-1)}\right),\; 0\le n\le M-1$$

$$M=\frac{N}{2}\;\; se\;\; N\;\; for\;\; par\;\; e\;\; M=\frac{(N+1)}{2}\;\; se\;\; N\;\; for\;\; ímpar$$

A rotina presente em “filtro_outras_janelas.m” repete o processo de projeto e teste usando as expressões acima. As expressões abaixo são usadas por esse *script* para estimar o valor de M a partir das especificações e do tipo de janela escolhida.

$$M_{Hamming}=\frac{3.32\pi}{\omega_S-\omega_P}\simeq 15$$

$$M_{Kaiser}=\frac{32}{2.285(\omega_S-\omega_P)}\simeq 20$$

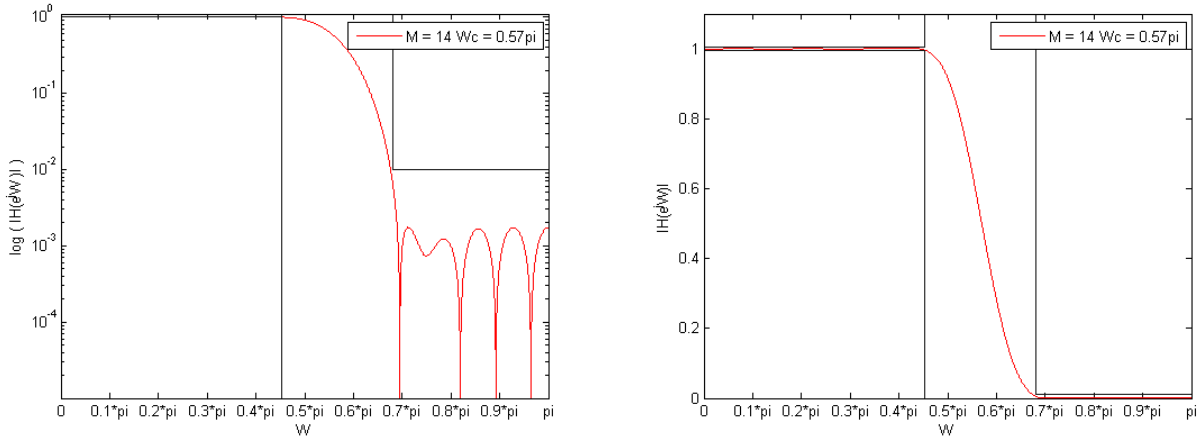
$$M_{Blackman}=\frac{5.56\pi}{\omega_S-\omega_P}\simeq 25$$

Ainda em relação a “filtro_outras_janelas.m”: é possível escolher o número de *bits* usado na quantização do sinal (lembrando que nbits igual a 128 equivale a precisão infinita). Então, outros três filtros foram criados, estes, referentes a h[n] truncada pelas janelas de Hamming, Kaiser e Blackman. Novamente, foi-se em busca de valores para M e ω_C que criassem funções de transferência que cumprem as especificações, minimizando o valor de M. O mesmo processo iterativo foi usado novamente, tendo como início M definido pelas expressões acima.

Neste momento considerou-se precisão infinita. Foi possível fazer os três filtros atenderem as especificações. As Figuras 10 a 12 trazem os resultados. Nas próprias imagens, no canto superior direito, estão os valores de M e ω_C que foram usados.

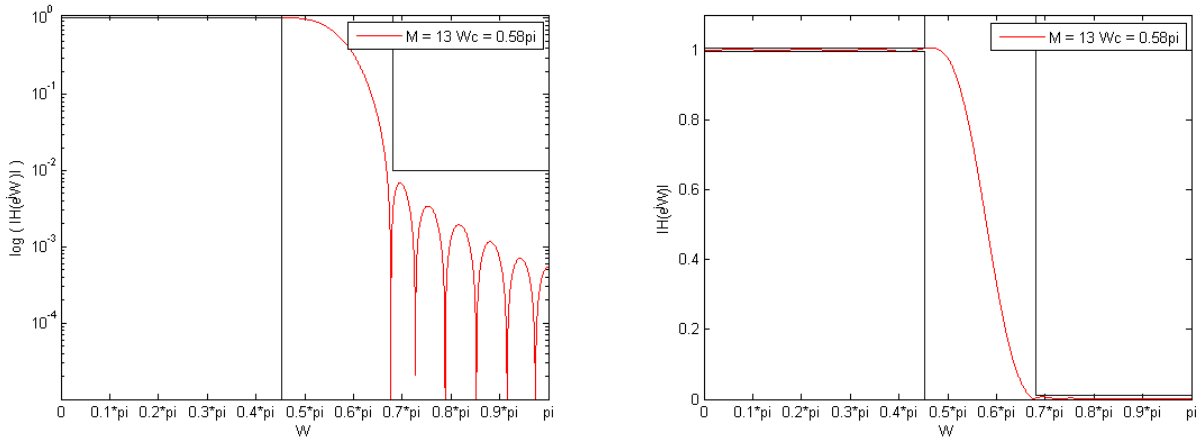
Aparentemente, a janela de Kaiser obteve o melhor desempenho, pois o filtro associado a ela pode ser implementado com a menor ordem, entre as três. Além disso, os valores de ω_C foram bastante próximos em todos os filtros.

Os resultados são bons, entretanto, a implementação em DSP não pode ser esquecida. Assim sendo, o próximo passo será quantizar os coeficientes de h[n] truncada por cada janela.



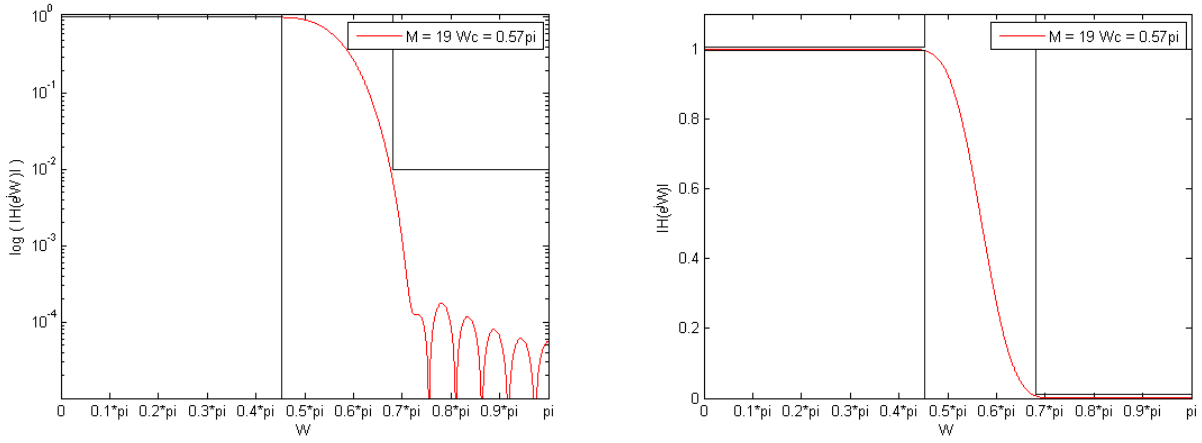
(a) (b)

Figura 10 – Resposta em frequência de h[n] truncada pela janela de Hamming com precisão infinita em escala (a) logarítmica e (b) linear.



(a) (b)

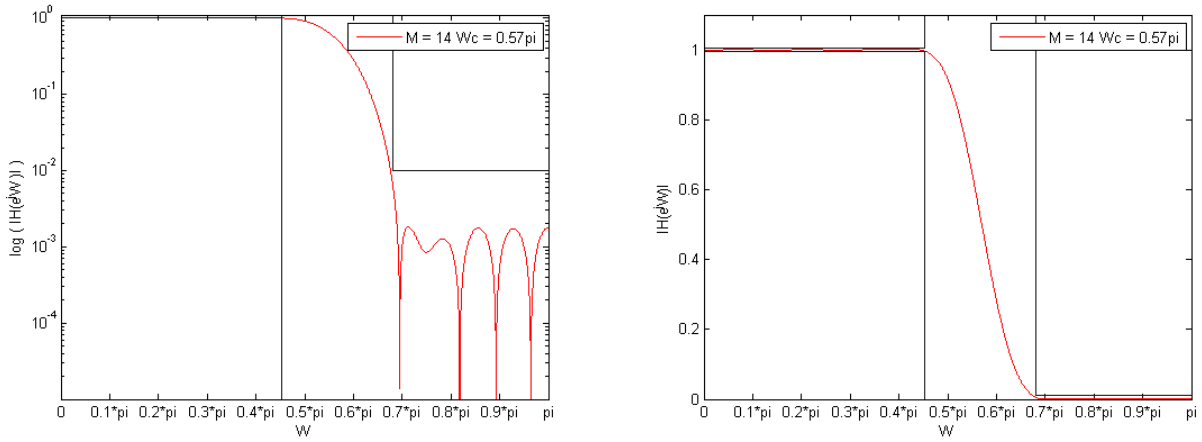
Figura 11 – Resposta em frequência de h[n] truncada pela janela de Kaiser com precisão infinita em escala (a) logarítmica e (b) linear.



(a) (b)

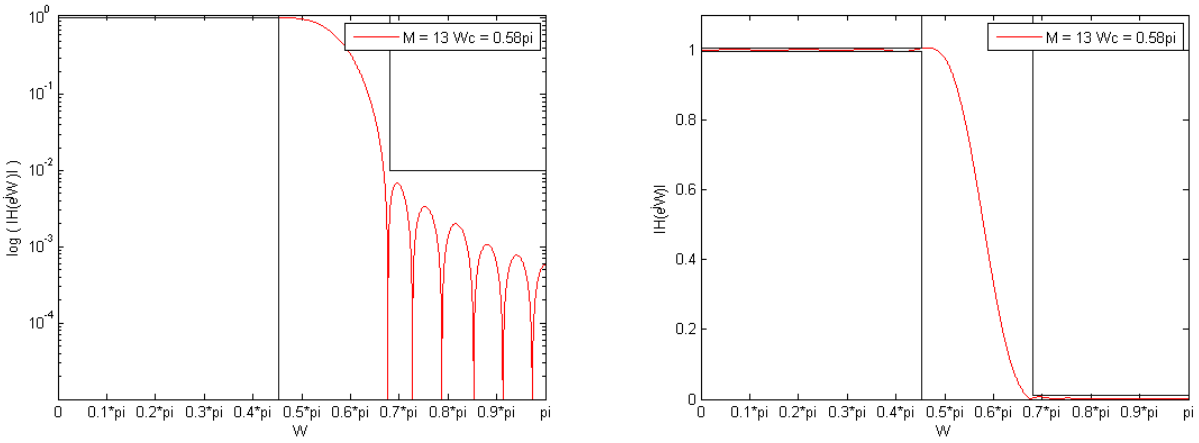
Figura 12 – Resposta em frequência de h[n] truncada pela janela de Blackman com precisão infinita em escala (a) logarítmica e (b) linear.

Considerou-se, por fim, precisão finita de forma equivalente ao caso da janela retangular. Os resultados foram interessantes. A quantização em 16 bits não inseriu distorção suficiente para tornar necessário a alteração das variáveis de projeto, ou seja, M e ω_C foram iguais considerando precisão finita ou infinita. Por outro lado, se fosse necessário operar com 8 bits, nenhum filtro atenderia as especificações.. As Figuras 13 a 18 mostram os resultados dos filtros que obtiveram sucesso.



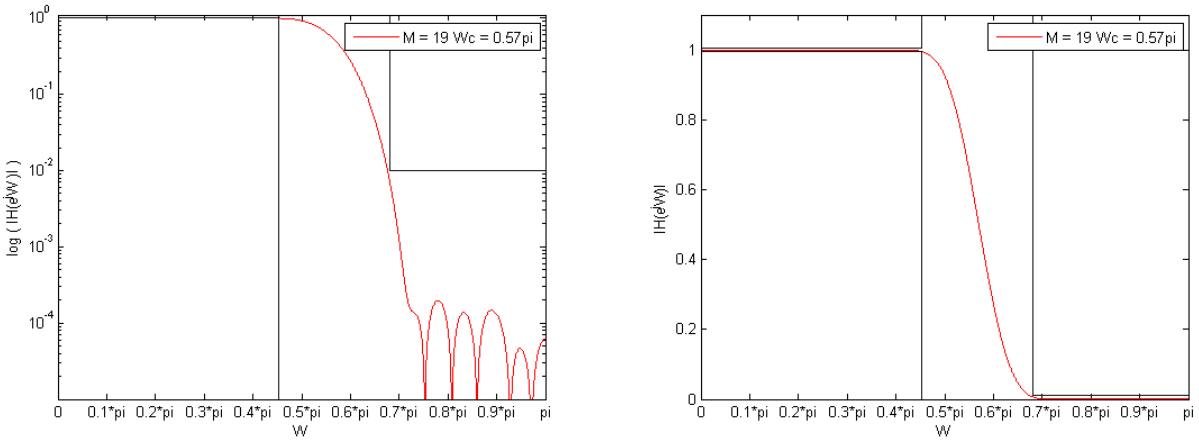
(a) (b)

Figura 13 – Janela de Hamming 16 bits em escala (a) logarítmica e (b) linear.



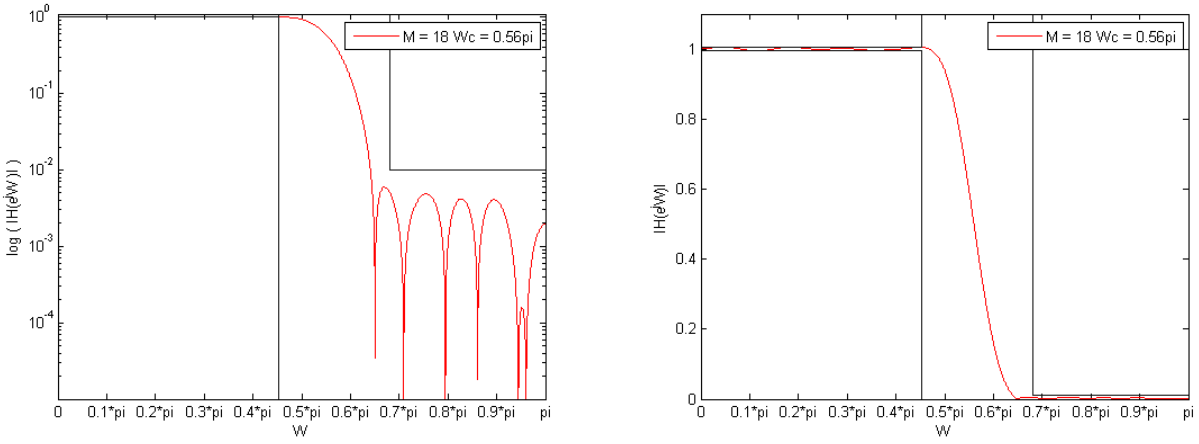
(a) (b)

Figura 14 – Janela de Kaiser 16 bits em escala (a) logarítmica e (b) linear.



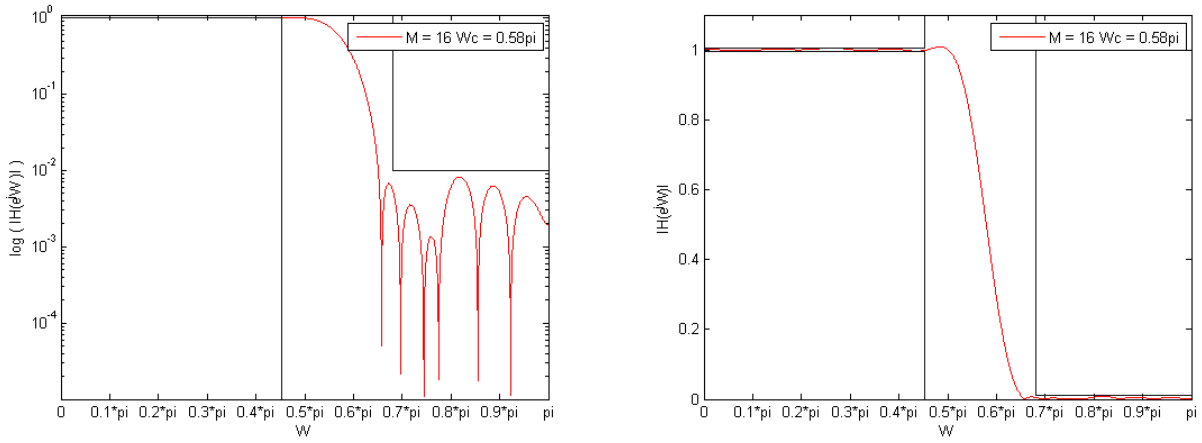
(a) (b)

Figura 15 – Janela de Blackman 16 bits em escala (a) logarítmica e (b) linear.



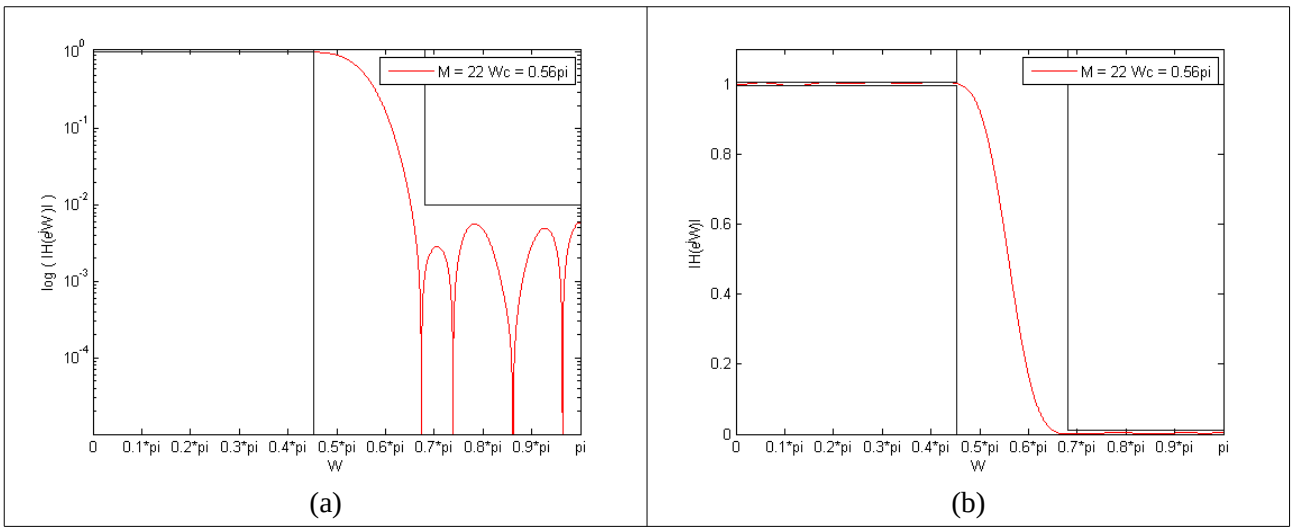
(a) (b)

Figura 16 – Janela de Hamming 10 bits em escala (a) logarítmica e (b) linear.



(a) (b)

Figura 17 – Janela de Kaiser 10 bits em escala (a) logarítmica e (b) linear.



(a) (b)

Figura 18 – Janela de Blackman 10 bits em escala (a) logarítmica e (b) linear.

O arquivo “script_v2.m” foi desenvolvido para possibilitar a rápida recuperação dos dados aqui apresentados. Nas primeiras linhas o usuário insere o tipo de janela e o número de bits desejados. A partir dessas informações cada resultado apresentado acima pode ser visto isoladamente.

A Tabela 2 resume os resultados obtidos. Finalmente, conclui-se que a escolha de uma janela adequada pode mudar drasticamente a ordem de um filtro. Cada parâmetro deve escolhido cuidadosamente analisando, sempre, a aplicação. Dentre as quatro possibilidades, analisando como melhor janela, aquela que permite a implementação de um filtro que atenda às especificações com a menor ordem, a janela de Hamming seria a vitoriosa.

Tabela 2 – Resultados obtidos.

Janela	**Precisão**	**M**	**ω_C**
Retangular	Infinita	114	0,58π
	16 *bits*	114	0,58π
	10 *bits*	-	-
	8 *bits*	-	-
Hamming	Infinita	14	0,57π
	16 *bits*	14	0,57π
	10 *bits*	18	0,56π
	8 *bits*	-	-
Kaiser	Infinita	13	0,58π
	16 *bits*	13	0,58π
	10 *bits*	16	0,58π
	8 *bits*	-	-
Blackman	Infinita	19	0,57π
	16 *bits*	19	0,57π
	10 *bits*	22	0,56π
	8 *bits*	-	-