Projeto de Graduação

30 de Junho de 2015

DESENVOLVIMENTO DE UMA INTERFACE GRÁFICA PARA PROJETO DE FILTROS DIGITAIS

Isabela Cunha Maia Nobre

www.ele.puc-rio.br

Projeto de Graduação

DESENVOLVIMENTO DE UMA INTERFACE GRÁFICA PARA PROJETO DE FILTROS DIGITAIS

Aluna: Isabela Cunha Maia Nobre

Orientador: Rodrigo de Lamare

Trabalho apresentado com requisito parcial à conclusão do curso de Engenharia Elétrica na Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil.

Agradecimentos

Agradeço a Deus, por ter me dado a oportunidade de estudar nessa universidade e forças para concluir essa etapa da minha vida. Agradeço também aos meus pais e à minha família pelo carinho e incentivo, além dos exemplos de dedicação e perseverança. Finalmente, gostaria de agradecer ao professor Rodrigo de Lamare, que aceitou me orientar nesse projeto, por todo apoio e ensinamento sem os quais esse trabalho não seria possível.

Resumo

Um filtro digital é um conjunto de operações matemáticas que manipula um sinal discreto. Possui várias aplicações em biomedicina, instrumentação e diversas outras áreas, além de ser um elemento essencial de eletrônicos do dia a dia como rádios e celulares. Projetar filtros digitais é uma tarefa complexa, já que cada especificação de projeto contribui para uma função de erro que deve ser minimizada. Um filtro digital pode ser usado como um protótipo para a implementação de um banco de filtros, arranjo que permite decompor o sinal de entrada em diversas sub-bandas de frequência que podem ser processadas separadamente.

O projeto consiste em usar a ferramenta computacional Matlab para o desenvolvimento de diversos tipos de filtros digitais e os agrupar em uma interface gráfica. O seu usuário poderá, além de selecionar parâmetros para construir o filtro, obter uma visualização gráfica e gerar uma saída a partir de uma entrada desejada. Além disso, um banco de filtros com custos computacionais reduzidos e número arbitrário de canais é implementado a partir da *Fast Fourier Transform* (FFT).

Palavras-chave: processamento digital de sinais; filtros digitais; banco de filtros

Development of a Graphical interface for Digital Filters Design

Abstract

A digital filter is a set of mathematical operations that manipulates a discrete signal. It has many applications in biomedicine, in instrumentation and several other areas, besides being an essential element of day to day electronics like radios and cell phones. Each design specification contributes to an error function that must be minimized, thereby making filter construction a complex task. A digital filter can be used as a prototype for the implementation of a filter bank, an array that allows the decomposition of the input signal into several sub-bands of frequencies that can be processed separately.

The project consists of using the computational tool Matlab for the design of several types of digital filters and gathering them in a graphical interface. Its user will be able to, in addition to selecting parameters to build the filter, obtain a graphical preview and generate an output from a desired input. Moreover, a filter bank with reduced computational costs and an arbitrary number of channels is implemented with the Fast Fourier Transform algorithm.

Keywords: digital signal processing; digital filters; filter banks

Sumário

1	Introdução	1
2	Projeto de Filtros FIR : Janelamento	3
a	Retangular	3
b	Triangular/ Bartlett	4
c	Hann	5
d	Hamming	6
e	Blackman	7
3	Projeto de Filtros IIR: Aproximações a Partir de Filtros Analógicos	9
a	Butterworth	10
b	Chebyshev	11
c	Elíptico	12
4	Transformações em Frequência	14
a	Transformações em Frequência para Filtros FIR	14
b	Transformações em Frequência para Filtros IIR	16
5	Bancos de Filtros	21
a	Alteração da Taxa de Amostragem	21
b	Decomposição Polifásica	22
c	Bancos de Filtros Digitais	22
1	Bancos de Filtros sem Decomposição Polifásica	23
2	Bancos de Filtros com Decomposição Polifásica	26
6	Interface Gráfica no Matlab	30
a	Painel de Entrada	30
b	Painel do Filtro	32
c	Painel de Saída	34
7	Conclusão	36

1 Introdução

Processamento de sinais é uma tecnologia que usa representações e técnicas matemáticas e computacionais para modelagem, interpretação e manipulação de sinais [1]. Um sinal pode ser definido como uma função que transmite informação sobre o comportamento de algum fenômeno (que pode ser físico ou não). É uma área diversa com várias aplicações: telecomunicações, sistemas eletrônicos, sistemas de controle, radares, diagnósticos médicos, sismologia, instrumentação científica, entre outros [2]. Devido à emergência das técnicas digitais nos anos 60, as técnicas de processamento de sinais se tornaram cada vez mais baratas e rápidas. Assim, apesar de processamento de sinais englobar técnicas digitais e analógicas, a área é dominada pela primeira [2].

O objetivo principal do processamento digital de sinais é a manipulação de sinais contínuos e analógicos, convertendo-os para o domínio discreto, processando-os e os convertendo de volta para o domínio contínuo. Hoje em dia, algumas das tecnologias usadas para o processamento digital de sinais incluem microprocessadores, FPGAs, controladores digitais de sinais, entre outros [3].

Um filtro digital é um sistema que manipula um sinal discreto através de operações matemáticas. Filtros digitais não estão sujeitos à não-linearidades de componentes presentes em filtros analógicos [4]. Filtros analógicos são compostos de componentes eletrônicos sujeitos a imprecisões e que também possuem seus valores alterados com a temperatura e com o tempo. Os valores dos coeficientes de um filtro digital são armazenados em memória de computador, o que os deixa mais estáveis e previsíveis [4]. Os coeficientes do filtro digital podem ser também mais facilmente manipulados, tornando possíveis projetos que são impraticáveis para filtros analógicos. Além disso, essa característica torna mais fácil a construção de filtros adaptativos e filtros com parâmetros definidos por usuários [4]. Suas aplicações são muitas, sendo um elemento essencial de sistemas eletrônicos do dia a dia como rádios e celulares, mas que também são usados na área biomédica e na instrumentação [3]. Exemplos do uso de filtros digitais em áudio incluem amplificadores, equalizadores, reverberadores, dispositivos de eco e defasadores [5].

Um filtro digital pode ser entendido como um cálculo em uma sequência de números (sinal de entrada) que produz uma nova sequência de números (sinal de saída filtrado). Ele pode ser caracterizado pela sua função de transferência, transformada Z da sua resposta impulsional (ou pela sua resposta em frequência, transformada de Fourier da sua resposta impulsional). Apesar do filtro digital ser conceitualmente fácil de entender, seu projeto e implementação são desafiadores e são objetos de muita pesquisa.

O projeto de filtros pode ser descrito como um problema de otimização onde cada especificação contribui para uma função de erro que deve ser minimizada [6]. Exemplos de requisitos típicos: o filtro deve ter uma resposta em frequência e um retardo de grupo específicos, o filtro deve ter uma resposta ao impulso específica, o filtro deve ser causal, o filtro deve ser estável, o filtro deve ter uma complexidade computacional pequena (proporcional a sua ordem), etc...

Os tipos de filtros projetados nesse trabalho são:

- Filtro passa-baixas: filtro que rejeita componentes de alta frequência do sinal
- Filtro passa-altas: filtro que rejeita componentes de baixa frequência do sinal
- Filtro passa-banda: filtro que passa uma faixa limitada de frequências
- Filtro rejeita-banda: filtro que passa somente frequências abaixo e acima de uma faixa limitada, rejeitando as componentes pertencentes à faixa.

De uma forma generalizada, a banda de passagem é o conjunto de frequências que o filtro não altera consideravelmente, e a banda de rejeição o conjunto de frequências que o filtro rejeita. A frequência de corte pode ser definida como aquela em que a energia do sinal é reduzida à metade. Ela está presente na faixa denominada “zona de transição”, entre a banda de passagem e a banda de rejeição.

Para os tipos de filtros acima, é geralmente desejado que na banda de passagem a resposta em frequência seja o mais próximo possível da unidade, enquanto que na banda de rejeição a resposta seja mais próxima de zero. Também é desejado que a transição seja curta. As oscilações em toda o espectro deve ser minimizada e, por último, a ordem do filtro (que implica em sua complexidade computacional) deve ser a menor possível. É impossível que todos esses requisitos sejam atendidos, portanto ao se projetar um filtro, procura-se determinar quais as tolerâncias para cada especificação e quais são as prioridades (o que muda com a aplicação).

Os filtros podem ser do tipo FIR (*Finite Impulse Response*), filtros não recursivos (não requerem realimentação) cuja resposta impulsional tem tamanho finito, ou do tipo IIR (*Infinite Impulse Response*), filtros recursivos (usam realimentação) cuja resposta impulsional tem tamanho infinito. Os filtros IIR projetados com auxílio de mapeamento matemático entre uma função analógica e uma função discreta são o equivalente digital dos filtros analógicos [3].

Uma vez projetado um filtro passa-baixas com as especificações requeridas, é possível utilizá-lo como protótipo para a implementação de um conjunto de filtros passa-banda. A esse conjunto se dá o nome de banco de filtros, um arranjo que permite decompor o sinal de entrada em diversas sub-bandas de frequências. Dessa forma, as diferentes componentes de frequência do sinal são isoladas e podem ser processadas separadamente. Um banco de filtro de análise permite essa decomposição, tendo uma entrada em comum e várias saídas, enquanto que um banco de filtros de síntese faz a recombinação de sub-bandas em um sinal único, apresentando assim uma saída em comum e várias entradas.

Existem diversas aplicações para bancos de filtros. Uma delas é a compressão de sinais, quando algumas frequências são mais importantes do que outras. Em áudio por exemplo, pode-se aproveitar do fato de que algumas frequências são menos perceptíveis do que outras para o ouvido humano e assim frequências mais importantes podem ser codificadas com uma resolução melhor [7]. Ao se usar uma codificação com menor resolução para frequências menos importantes, o tamanho do sinal é reduzido, e apesar de detalhes serem perdidos nestas frequências, eles não são igualmente necessários. A referência [8] descreve um método para o projeto de um banco de filtros para cancelamento de eco acústico. O eco degrada muito a qualidade da voz e sua eliminação é portanto desejada. Um outro exemplo é o *vocoder*, que usa um banco de filtros para determinar as amplitudes das sub-bandas do sinal de entrada (geralmente uma voz), gerando uma série de sinais representando o seu espectro [9]. Dessa forma, a quantidade de informação necessária para armazenar o sinal de voz é dramaticamente reduzida, pois ao invés de uma gravação inteira, somente uma série de números é transmitida.

O objetivo deste projeto é desenvolver uma interface no Matlab que permita usuários projetar filtros digitais. A interface permitirá que os projetistas de filtros digitais interativamente sintetizem o filtro desejado. O programa contará com a visualização gráfica dos sinais e do filtro, facilitando uma boa verificação. Assim, o usuário poderá decidir visualmente o melhor filtro para cada projeto, determinando sua ordem e verificando se suas especificações são atendidas. Os sinais de entrada e saída contarão com gráficos tanto no domínio do tempo quanto no domínio da frequência. Além disso, um banco de filtros com um número arbitrário de bandas (especificado pelo usuário) será implementado.

Primeiramente, será abordado o projeto de filtros passa-baixas do tipo FIR, cuja técnica explorada é a de janelamento. Em seguida o mesmo é feito com filtros IIR, sendo obtidos com aproximações a partir de filtros analógicos. No capítulo 4 são exploradas técnicas de obtenção dos outros tipos de filtros (passa-altas, passa-banda e rejeita-banda) a partir de um protótipo passa-baixas. A implementação de banco de filtros é discutida no capítulo 5. No último capítulo, a interface final no Matlab será apresentada.

2 Projeto de Filtros FIR : Janelamento

Um filtro FIR (*Finite Impulse Response*) possui resposta impulsional de tamanho finito (L coeficientes). Ou seja, a resposta ao impulso retorna a zero no tempo $n = L$. A sua ordem é dada por $M = L-1$.

A função de transferência desse tipo de filtro tem o seguinte formato:

$$H(z) = B(z) = \sum_{k=0}^M b_k z^{-k}, \quad (1)$$

em que $B(z)$ é um polinômio em z de grau M e b_k são os coeficientes do filtro. O filtro FIR não possui pólos. Neste trabalho foi abordado o método do janelamento para esse tipo de filtro, discutido a seguir.

O método do janelamento consiste em truncar a resposta impulsional (em $L = M+1$ amostras) de um filtro ideal passa-baixas e depois multiplicá-la por uma função janela.

A resposta em frequência de um filtro passa-baixas ideal com frequência de corte ω_c e atraso α é dada por:

$$H_d(e^{j\omega}) = \begin{cases} e^{-j\alpha\omega} & |\omega| \leq \omega_c \\ 0 & \omega_c \leq |\omega| \leq \pi. \end{cases} \quad (2)$$

A resposta impulsional desse filtro é dada por

$$h_d[n] = \frac{\sin(\omega_c(n - \alpha))}{\pi(n - \alpha)}. \quad (3)$$

Este sinal é truncado em L amostras. Um filtro com resposta finita pode apresentar fase linear, ou seja, a resposta em fase do filtro é uma função linear da frequência. Dessa forma não há distorção de fase já que todas as componentes de frequência serão atrasadas pelo mesmo valor. Para que este filtro apresente esta propriedade, $\alpha = \frac{M}{2}$, para M par e $\alpha = \frac{M-1}{2}$ para M ímpar.

O filtro será obtido através do seguinte produto:

$$h[n] = h_d[n] \cdot w[n] \quad (4)$$

em que $w[n]$ é a função janela.

Ao fazer essa multiplicação no domínio do tempo, o efeito no domínio da frequência é a convolução da resposta do filtro ideal com a transformada de Fourier da função janela. Assim, quanto mais estreito o lóbulo principal da janela, mas acentuada é a transição no filtro. Da mesma forma, quanto menor for a amplitude dos lóbulos secundários, menos ondulações ocorrerão.

a Retangular

A função janela mais simples é a janela retangular. Ela é definida como:

$$w[n] = \begin{cases} 1 & 0 \leq n \leq M \\ 0 & \text{diferente.} \end{cases} \quad (5)$$

Essa função possui uma transição abrupta para o zero, e isso faz com que haja um comportamento oscilatório no filtro resultante. Os outros tipos de janela buscam amenizar essa descontinuidade.

Este filtro foi implementado da seguinte forma no Matlab:

```
1 %Passa baixas ideal
2 a = M/2;
3 for n = 1:M
4     hd1(n) = sinc(wc1*(n-floor(a)))*wc1;
5 end
6 hd1 = hd1';
7
8 %Grafico do filtro com a janela retangular
9 [h,w] = freqz(hd);
10 figure
11 plot(w/pi, 20*log10(abs(h)));
12 title('Janela retangular')
```

```
13 y1=-130; y2=20;
14 ylim([y1 y2])
15 xlim([0 1])
```

O código acima gera o seguinte filtro, com frequência de corte normalizada $\omega_c = 0.2$ e ordem $M = 30$:

Figura 1: Filtro FIR passa-baixas com janela retangular

b Triangular/ Bartlett

A janela triangular é definida como:

$$w[n] = \begin{cases} \frac{2n}{M} & 0 \leq n \leq \frac{M}{2} \\ 2 - \frac{2n}{M} & \frac{M}{2} \leq n \leq M \\ 0 & \text{diferente.} \end{cases} \quad (6)$$

A janela pode ser entendida como a convolução de duas janelas retangulares. A transição para o zero é linear, não mais descontínua. Dessa forma, o comportamento oscilatório da janela retangular é reduzido.

Utilizando o mesmo filtro ideal mostrado no código da janela retangular (hd), o código abaixo implementa a janela triangular e constrói o filtro digital:

```
1 % Funcao Janela
2 for n = 1:M/2
3     w_bart(n) = 2*n/M;
4 end
5 for n = floor(M/2)+1:M
6     w_bart(n) = 2-2*n/M;
7 end
8 w_bart = w_bart';
9
10 % Filtro
11 hd = hd.*w_bart;
12 [h,w] = freqz(hd);
13 figure
14 plot(w/pi, 20*log10(abs(h)));
15 title('Janela de Bartlett / triangular')
16 ylim([y1 y2])
17 xlim([0 1])
```

O código acima gera o seguinte filtro, com frequência de corte normalizada $\omega_c = 0.2$ e ordem $M = 30$:

Figura 2: Filtro FIR passa-baixas com janela triangular

c Hann

A janela de Hann é definida como:

$$w[n] = \begin{cases} 0.5 - 0.5\cos\left(\frac{2\pi n}{M}\right), & 0 \leq n \leq M. \\ 0, & \text{diferente.} \end{cases} \quad (7)$$

Esta janela apresenta uma boa atenuação na banda de rejeição, se comparada às janelas anteriores.

Utilizando o mesmo filtro ideal mostrado no código da janela retangular (hd), o código abaixo implementa a janela de Hann e constrói o filtro digital:

```
1 % Funcao Janela
2 w_hann= 0.5*(1- cos(2*pi*(1:M)/M));
3 w_hann = w_hann';
4
5 % Filtro
6 hd = hd.*w_hann;
7 [h,w] = freqz(hd);
8 figure
9 plot(w/pi,20*log10(abs(h)));
10 title('Janela de Hann')
11 ylim([y1 y2])
12 xlim([0 1])
```

O código acima gera o seguinte filtro, com frequência de corte normalizada $\omega_c = 0.2$ e ordem $M = 30$:

Figura 3: Filtro FIR passa-baixas com janela de Hann

d Hamming

A janela de Hamming é definida como:

$$w[n] = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi n}{M}\right), & 0 \leq n \leq M. \\ 0, & \text{diferente.} \end{cases} \quad (8)$$

A janela foi projetada de forma a minimizar o maior lóbulo secundário da janela de Hann, distribuindo sua energia para os demais lóbulos secundários.

O código abaixo implementa a janela de Hamming e constrói o filtro digital:

```
1 % Funcao Janela
2 w_hamming= 0.54 - 0.46*cos(2*pi*(1:M)/M);
3 w_hamming = w_hamming';
4
5 % Filtro
6 hd = hd.*w_hamming;
7 [h,w] = freqz(hd);
8 figure
9 plot(w/pi,20*log10(abs(h)));
10 title('Janela de Hamming')
11 ylim([y1 y2])
12 xlim([0 1])
```

O código acima gera o seguinte filtro, com frequência de corte normalizada $\omega_c = 0.2$ e ordem $M = 30$:

Figura 4: Filtro FIR passa-baixas com janela de Hamming

e Blackman

A janela de Blackman é definida como:

$$w[n] = \begin{cases} 0.42 - 0.5\cos\left(\frac{2\pi n}{M-1}\right) + 0.08\cos\left(\frac{4\pi n}{M-1}\right), & 0 \leq n \leq M. \\ 0, & \text{diferente.} \end{cases} \quad (9)$$

Essa janela possui uma atenuação maior na banda de rejeição, se comparado com todas as janelas anteriores. Por outro lado, a zona de transição se alarga.

O código abaixo implementa a janela de Blackman e constrói o filtro digital:

```
1 % Funcao Janela
2 w_black= 0.42 - 0.5* cos(2*pi*(1:M)/(M-1))+0.08*cos(4*pi*(1:M)/(M-1));
3 w_black=w_black';
4
5 % Filtro
6 hd = hd.* w_black;
7 [h,w] = freqz(hd);
8 figure
9 plot(w/pi,20*log10(abs(h)));
10 title('Janela de Blackman')
11 ylim([y1 y2])
12 xlim([0 1])
```

O código acima gera o seguinte filtro, com frequência de corte normalizada $\omega_c = 0.2$ e ordem $M = 30$:

Figura 5: Filtro FIR passa-baixas com Janela de Blackman

O formato no tempo de todas as janelas descritas acima é mostrado na figura abaixo (para $M = 20$):

Figura 6: Janelas comumente usadas

3 Projeto de Filtros IIR: Aproximações a Partir de Filtros Analógicos

Esse tipo de filtro possui uma resposta impulsional que não retorna a zero depois de algum tempo. Um filtro IIR (*Infinite Impulse Response*) possui resposta impulsional de tamanho infinito. Geralmente um sistema com realimentação apresenta uma resposta IIR.

A função de transferência desse tipo de filtro tem o seguinte formato:

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^N b_k z^{-k}}{\sum_{k=1}^M a_k z^{-k}}, \quad (10)$$

em que $B(z)$ e $A(z)$ são polinômios em z . $A(z)$ é de ordem M , enquanto $B(z)$ tem ordem igual ou inferior a M (filtros causais). A maior vantagem do filtro IIR é a necessidade de uma ordem menor para atender às mesmas restrições, se comparado com a de um filtro FIR. A maior desvantagem é a dificuldade em garantir fase linear e uma possível instabilidade do filtro.

Neste trabalho, os filtros IIR foram construídos com o método de aproximações a partir de três filtros analógicos, apresentados abaixo.

Filtros IIR analógicos foram extensivamente estudados e desenvolvidos [6]. Por isso já existem inúmeras opções de filtros analógicos com características bem conhecidas. Além disso, muitos desses filtros contínuos possuem fórmulas fechadas. Assim, é vantajoso aproveitar a existência deles e construir filtros digitais através de algum mapeamento.

Um dos mapeamentos mais utilizados é a transformação bilinear. Nela, cada ponto da resposta em frequência do filtro analógico a um ponto correspondente na resposta em frequência do filtro digital. A estabilidade é preservada, ou seja, o semi-plano esquerdo do plano S (Laplace) é mapeado no interior do círculo unitário no plano Z . Além disso, esse mapeamento não produz distorção devido ao fenômeno de *aliasing*.

O filtro digital é obtido a partir da seguinte mudança de variáveis:

$$z = \frac{1 + \frac{s}{2f_s}}{1 - \frac{s}{2f_s}}, \quad (11)$$

onde f_s é um parâmetro que pode variar.

O mapeamento entre as frequências digital (ω) e analógica (Ω) resultado da transformação acima é dado por:

$$\Omega = 2f_s \cdot \tan\left(\frac{\omega}{2}\right) \quad (12)$$

ou

$$\omega = 2 \cdot \arctan\left(\frac{\Omega}{2f_s}\right) \quad (13)$$

O seguinte código do Matlab realiza essa transformação. Ele recebe os pólos, zeros e ganho (z, p, k) do filtro analógico e a frequência discreta (ω) na qual a frequência continua $\Omega = 1$ será mapeada, calculando os pólos, zeros e ganho (z_d, p_d, k_d) do filtro digital.

```
1 fs= 1/(2*tan(w/2));
2 z = z(isfinite(z)); %retira zeros no infinito
3 pd = (1+p/(2fs))./(1-p/(2fs));
4 zd= (1+z/(2fs))./(1-z/(2fs));
5 kd= real(k*prod((2fs)-z)./prod((2fs)-p));
6
7 %transforma os zeros no infinito em zeros em -1:
8 for i=size(zd)+1:size(pd)
9     zd(i)=-1;
10 end
11 zd=zd';
```

A seguir, três filtros analógicos passa-baixas são explorados, e é mostrada a implementação de seus respectivos filtros digitais.

a Butterworth

O filtro Butterworth é otimizado para apresentar a resposta mais plana na banda de passagem. Um filtro Butterworth passa-baixas é sempre decrescente (monotônico). Ele não possui oscilações, mas apresenta uma taxa de decaimento menor, se comparado com os filtros seguintes (Chebyshev e Elíptico), assim demandando uma ordem maior para que as mesmas especificações sejam atendidas. O filtro também apresenta uma resposta em fase mais linear.

O quadrado do módulo da sua resposta em frequência possui o seguinte formato:

$$|H(j\Omega)|^2 = \frac{1}{1 + \left(\frac{\Omega}{\Omega_c}\right)^{2M}}, \quad (14)$$

em que Ω_c é a frequência de corte do filtro e M sua ordem. Assim, tem-se:

$$H(s)H(-s) = \frac{(j\Omega_c)^{2M}}{s^{2M} + (j\Omega_c)^{2M}} \quad (15)$$

As raízes do denominador de $H(s)H(-s)$ (pólos) são dadas por:

$$p_k = \Omega_c e^{j\frac{\pi}{2M}(2k+M+1)} \quad k = 0, 1, \dots, 2N - 1 \quad (16)$$

Os 2M pólos de $H(s)H(-s)$ estão distribuídos no círculo de raio Ω_c (separados por $\frac{\pi}{M}$ radianos). Os M pólos de $H(s)$ são aqueles localizados no lado esquerdo do eixo imaginário (condição para filtro estável e causal). O filtro não possui zeros.

Os seguinte comandos projetam um filtro Butterworth analógico normalizado ($\Omega_c = 1$ rad), calculando seus pólos, zeros e ganho a partir da ordem do filtro.

```
1 z = []; %zeros
2 p = exp(1i*(pi*(1:2:2*M-1)/(2*M)+pi/2)).'; % polos
3 k = real(prod(-p)); %ganho
```

Usando o código que faz a transformação bilinear, obtém-se o filtro digital. Com os pólos, zeros e ganho do filtro digital, o seguinte comando produz o gráfico do filtro desejado:

```
1 sos = zp2sos(zd,pd,kd);
2 [h,w] = freqz(sos);
3 plot(w/pi, 20*log10(abs(h)));
4 title('Butterworth')
```

Para $M = 15$ e frequência de corte do filtro digital $\omega_c = 0.2$:

Figura 7: Filtro Butterworth

b Chebyshev

Filtros Chebyshev (tipo I) possui oscilações que mudam uniformemente dentro da tolerância definida (*equiripple*) na faixa de passagem. Além disso, ele é monotônico na banda de rejeição. Por possuírem maior oscilação do que o Butterworth, a transição é mais íngreme. Assim como no filtro anterior, os pólos estão localizados na parte esquerda de uma elipse centrada na origem do plano S, e são igualmente espaçados nela.

O quadrado do módulo da sua resposta em frequência possui o seguinte formato:

$$|H(j\Omega)|^2 = \frac{1}{1 + \epsilon^2 T_M^2\left(\frac{\Omega}{\Omega_c}\right)} \quad (17)$$

Sendo M sua ordem, ϵ o fator de oscilação da banda de passagem e $T_M(x)$ o polinômio Chebyshev de ordem M. O filtro não possui zeros, e os pólos podem ser calculados com o seguinte procedimento (explicado em detalhes em [10]):

$$p_k = \sigma_k + j\Omega_k \quad k = 0, \dots, M-1, \quad \text{onde} \quad (18)$$

$$\sigma_k = a\Omega_c \cos\left(\frac{\pi}{2} + \frac{(2k+1)\pi}{2M}\right) \quad k = 0, \dots, M-1 \quad (19)$$

$$\Omega_k = b\Omega_c \sin\left(\frac{\pi}{2} + \frac{(2k+1)\pi}{2M}\right) \quad k = 0, \dots, M-1 \quad (20)$$

$$\text{sendo } a = \frac{1}{2}(\sqrt[M]{\alpha} - \sqrt[M]{1/\alpha}), \quad b = \frac{1}{2}(\sqrt[M]{\alpha} + \sqrt[M]{1/\alpha}) \quad (21)$$

$$\text{e } \alpha = \frac{1}{\epsilon} + \sqrt{1 + \frac{1}{\epsilon^2}} \quad (22)$$

$$\text{onde } \epsilon = \sqrt{10^{0.1R_p} - 1} \quad (23)$$

Sendo Ω_c a frequência de corte e R_p a máxima oscilação permitida em dB na banda de passagem.

O ganho é escolhido de tal forma que:

$$H(j0) = \begin{cases} 1 & \text{M ímpar} \\ \frac{1}{\sqrt{1+\epsilon^2}} & \text{M par} \end{cases} \quad (24)$$

Os seguinte comandos projetam um filtro Chebyshev analógico normalizado ($\Omega_c = 1$ rad), calculando seus pólos, zeros e ganho a partir de M e R_p :

```

1  z = []; %zeros
2  epi=sqrt(10^(0.1*Rp)-1);
3  alpha = 1/epi+sqrt(1+1/epi^2);
4  a=1/2*(alpha^(1/M)-(1/alpha)^(1/M));
5  b=1/2*(alpha^(1/M)+(1/alpha)^(1/M));
6
7  sigma = a*cos(pi/2+pi*(1:2:2*M-1)/(2*M));
8  omega = b*sin(pi/2+pi*(1:2:2*M-1)/(2*M));
9
10 p = (sigma+sqrt(-1)*omega).'; %polos
11
12 if (mod(M,2) == 0) %ganho
13     k = real(1/(sqrt(1+epi^2))*prod(-p));
14 else
15     k = real(prod(-p));
16 end

```

Usando o mesmo procedimento do filtro Butterworth para obter o filtro digital e gerar o gráfico da sua resposta em frequência, obtém-se a seguinte imagem para $M = 15$, $\omega_c = 0.2$ e $R_p = 10$:

Figura 8: Filtro Chebyshev

c Elíptico

Filtros Elípticos são *equiripple* tanto na banda de passagem quanto na banda de rejeição, assim sua banda de transição é a mais estreita, se comparada com os filtros Chebyshev e Butterworth. Do mesmo modo, para atender às mesmas especificações, é necessária uma ordem menor para o filtro Elíptico. Além do mais, as oscilações na banda de passagem e banda de rejeição podem ser independentemente controladas.

O quadrado do módulo da sua resposta em frequência possui o seguinte formato [10]:

$$|H(j\Omega)|^2 = \frac{1}{1 + \epsilon^2 U_M^2\left(\xi, \frac{\Omega}{\Omega_c}\right)} \quad (25)$$

Sendo M sua ordem, ϵ o fator de oscilação da banda de passagem (relacionado com R_p), ξ o fator de seleção (relacionado com R_p e R_s) e $U_M(x)$ a função Jacobiana elíptica de ordem M . A análise dessa função é bem complexa e a construção do filtro foi feita com base na referência [11].

Os seguinte comandos projetam um filtro Elíptico analógico normalizado ($\Omega_c = 1$ rad), calculando seus pólos, zeros e ganho (aqui z, p e g) a partir de M , R_p e R_s :

```

1  ep = sqrt(10^(Rp/10) - 1); % fatores de oscilacao
2  es = sqrt(10^(Rs/10) - 1);
3  k1 = ep/es;
4  k = ellipdeg(M,k1);
5  L = floor(M/2);

```

```

6  r = mod(M,2);
7  i = (1:L)';
8  ui = (2*i-1)/M;
9  zeta_i = cde(ui,k);
10 z = j./(k*zeta_i);
11 v0 = -j*asne(j/ep, k1)/M;
12 p = j*cde(ui-j*v0, k);
13 p0 = j*sne(j*v0, k);
14
15 z = cplxpair([z; conj(z)]); %zeros
16 p = cplxpair([p; conj(p)]); %polos
17 if r==1, p = [p; p0]; end
18
19 Gp = 10^(-Rp/20);
20 H0 = Gp^(1-r);
21 g = abs(H0*prod(p)/prod(z)); %ganho

```

Usando o mesmo procedimento do filtro Butterworth para obter o filtro digital e gerar o gráfico da sua resposta em frequência, obtém-se a seguinte imagem para $M = 15$, $\omega_c = 0.2$, $R_p = 10$ e $R_s = 40$:

Figura 9: Filtro Elíptico

4 Transformações em Frequência

Todos os exemplos de construção de filtros mostrados anteriormente são do tipo passa-baixas. No entanto, outros tipos de filtros (passa-banda, passa-altas e rejeita-banda) também são importantes. Eles podem ser obtidos a partir dos filtros passa-baixas através de dois métodos, um para os tipos FIR e outro para os tipos IIR. Eles serão detalhados a seguir.

a Transformações em Frequência para Filtros FIR

Os filtros do tipo FIR desenvolvidos anteriormente apresentam fase linear. Além disso, o atraso na fase depende somente da ordem do filtro. O que significa que, ao se subtrair dois filtros de mesma ordem, como as fases deles serão iguais, o efeito será somente na magnitude, conforme o seguinte:

$$H_2(e^{j\omega}) - H_1(e^{j\omega}) = |H_2(e^{j\omega})|e^{j\alpha\omega} - |H_1(e^{j\omega})|e^{j\alpha\omega} \quad (26)$$

$$= (|H_2(e^{j\omega})| - |H_1(e^{j\omega})|)e^{j\alpha\omega} \quad (27)$$

Para construir um filtro passa-banda, a propriedade acima é muito útil. Isso é possível usando dois filtros passa-baixas de mesma ordem, mas com diferentes frequências de corte. A figura abaixo ilustra o efeito da operação com os filtros:

Figura 10: Filtro ideal passa-banda a partir de dois filtros passa-baixas

Por causa da propriedade da linearidade da Transformada de Fourier, fazer essa operação no domínio da frequência equivale a fazê-la com os filtros $h_1[n]$ e $h_2[n]$ no domínio do tempo.

Para um filtro passa-altas, a mesma operação deve ser feita, no entanto com $|H_2(e^{j\omega})| = 1$ (Ou seja, $H_2(e^{j\omega}) = e^{j\alpha\omega}$). Isso significa um filtro passa-baixas com frequência de corte $\omega_c = \pi$.

Para o filtro rejeita-banda, basta somar um filtro passa-baixas com um filtro passa-altas.

Sendo ω_{c1} e ω_{c2} as frequências de corte (o passa-altas é definido por ω_{c1}), o seguinte código realiza as transformações:

```
1 a = M/2;
2 for n = 1:M
3     hd1(n) = sinc(wc1*(n-floor(a)))*wc1;
4 end
5 for n = 1:M
6     hd2(n) = sinc(wc2*(n-floor(a)))*wc2;
7 end
8 for n = 1:M
9     hd3(n) = sinc(1*(n-floor(a)));
10 end
11
12 if(strcmp(tipo,'baixa') )
13     hd = hd1;
14 elseif (strcmp(tipo,'pbanda') )
```

```

15     hd = hd2-hd1;
16     elseif(strcmp(tipo,'alta') )
17         hd = hd3-hd1;
18     elseif(strcmp(tipo,'rbanda') )
19         hd = hd1+hd3-hd2;
20 end
    
```

As janelas são aplicadas ao filtro resultante h_d depois da transformação em frequência. O procedimento é exatamente o mesmo discutido na seção 2. Alguns resultados abaixo, para $M = 30$, $\omega_{c1} = 0.2$ e $\omega_{c2} = 0.5$ são mostrados:

Figura 11: Janela de Blackman passa-altas

Figura 12: Janela de Hamming rejeita-banda

Figura 13: Janela de Hann passa-banda

b Transformações em Frequência para Filtros IIR

O método consiste em aplicar uma transformação algébrica no filtro passa-baixas $H(Z)$ para obter o filtro desejado $H_d(z)$. Sendo a variável Z associada ao filtro passa-baixas e a variável z associada ao filtro desejado, a transformação é definida como:

$$Z^{-1} = G(z^{-1}) \quad (28)$$

Assim:

$$H_d(z) = H(Z)|_{Z^{-1}=G(z^{-1})} \quad (29)$$

A referência [6] mostra as seguintes transformações:

Filter Type	Transformations	Associated Design Formulas
Lowpass	$Z^{-1} = \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}}$	$\alpha = \frac{\sin\left(\frac{\theta_p - \omega_p}{2}\right)}{\sin\left(\frac{\theta_p + \omega_p}{2}\right)}$ $\omega_p = \text{desired cutoff frequency}$
Highpass	$Z^{-1} = -\frac{z^{-1} + \alpha}{1 + \alpha z^{-1}}$	$\alpha = -\frac{\cos\left(\frac{\theta_p + \omega_p}{2}\right)}{\cos\left(\frac{\theta_p - \omega_p}{2}\right)}$ $\omega_p = \text{desired cutoff frequency}$
Bandpass	$Z^{-1} = -\frac{z^{-2} - \frac{2\alpha k}{k+1}z^{-1} + \frac{k-1}{k+1}}{\frac{k-1}{k+1}z^{-2} - \frac{2\alpha k}{k+1}z^{-1} + 1}$	$\alpha = \frac{\cos\left(\frac{\omega_{p2} + \omega_{p1}}{2}\right)}{\cos\left(\frac{\omega_{p2} - \omega_{p1}}{2}\right)}$ $k = \cot\left(\frac{\omega_{p2} - \omega_{p1}}{2}\right) \tan\left(\frac{\theta_p}{2}\right)$ $\omega_{p1} = \text{desired lower cutoff frequency}$ $\omega_{p2} = \text{desired upper cutoff frequency}$
Bandstop	$Z^{-1} = \frac{z^{-2} - \frac{2\alpha}{1+k}z^{-1} + \frac{1-k}{1+k}}{\frac{1-k}{1+k}z^{-2} - \frac{2\alpha}{1+k}z^{-1} + 1}$	$\alpha = \frac{\cos\left(\frac{\omega_{p2} + \omega_{p1}}{2}\right)}{\cos\left(\frac{\omega_{p2} - \omega_{p1}}{2}\right)}$ $k = \tan\left(\frac{\omega_{p2} - \omega_{p1}}{2}\right) \tan\left(\frac{\theta_p}{2}\right)$ $\omega_{p1} = \text{desired lower cutoff frequency}$ $\omega_{p2} = \text{desired upper cutoff frequency}$

Figura 14: Transformações a partir de um filtro passa-baixas de frequência de corte θ_p .

Para que o mapeamento possa ser feito, é necessário antes construir uma função que implementa a substituição de funções racionais. Ou seja, que receba os polinômios do numerador ($b(Z)$) e denominador ($a(Z)$) de $H(Z)$ e os polinômios do numerador ($N(z)$) e denominador ($D(z)$) de $G(z)$, retornando os polinômios do numerador (num) e denominador (den) de $H_d(z)$. A operação algébrica é desenvolvida da seguinte forma:

$$H(Z) = \frac{b(Z)}{a(Z)} = \frac{b_1 Z^{n-1} + b_2 Z^{n-2} + \dots + b_n}{a_1 Z^{m-1} + a_2 Z^{m-2} + \dots + a_m} \quad (30)$$

$$= \frac{b_1 \left(\frac{N(z)}{D(z)}\right)^{n-1} + b_2 \left(\frac{N(z)}{D(z)}\right)^{n-2} + \dots + b_n}{a_1 \left(\frac{N(z)}{D(z)}\right)^{m-1} + a_2 \left(\frac{N(z)}{D(z)}\right)^{m-2} + \dots + a_m} \times \left(\frac{D(z)^{m-1}}{D(z)^{m-1}}\right) \quad (31)$$

$$= \frac{b_1 N(z)^{n-1} D(z)^{m-n} + b_2 N(z)^{n-2} D(z)^{m-n+1} + \dots + b_n D(z)^{m-1}}{a_1 N(z)^{m-1} + a_2 N(z)^{m-2} D(z) + \dots + a_m D(z)^{m-1}} \quad (32)$$

$$= \frac{\text{num}}{\text{den}} \quad (33)$$

Para que a transformação acima possa ser implementada no Matlab, as seguintes operações com polinômios devem ser utilizadas:

1. Soma

A seguinte função realiza essa operação:

```
1 function p = SomaPoly(p1,p2)
2 p=[zeros(1, size(p1,2)-size(p2,2)) p2] + [zeros(1, size(p2,2)-size(p1,2)) p1];
```

2. Produto

O comando *conv* do Matlab, que realiza a convolução entre dois vetores, também serve para realizar o produto entre dois polinômios.

3. Potenciação

A seguinte função realiza essa operação (N é o polinômio e m a potência):

```
1 function p = ElevarPoly(N,m)
2 if m==0
3     p=1;
4     return
5 end
6 p=1;
7 for i = 1:m
8     p = conv(p,N);
9 end
```

Com as três funções acima, é possível fazer a função que faz o mapeamento:

```
1 function [num,den] = ZMAPA(b,a,N,D)
2 n=size(b,2);
3 m=size(a,2);
4 num=0;
5 den=0;
6 for i= 1:n
7     num = SomaPoly(num, b(i)*conv(ElevarPoly(N,n-i),ElevarPoly(D,m-n+i-1)));
8 end
9 for i=1:m
10    den = SomaPoly(den, a(i)*conv(ElevarPoly(N,m-i), ElevarPoly(D, i-1)));
11 end
```

Assim, a transformação em frequência se torna um procedimento simples, com base na figura 14 e na função ZMAPA:

```
1 switch tipo
2     case 'alta'
3         alpha = -cos(wc1*pi);
4         Nz = [1,alpha];
5         Dz = [alpha,1];
6         [b,a] = zp2tf(zd,pd,kd);
7         [b,a] = ZMAPA(b,a,Nz,Dz);
8         [zd,pd,kd] = tf2zp(b,a);
9     case 'pbanda'
10        alpha = cos((wc1+wc2)*pi/2)/cos((wc2-wc1)*pi/2);
11        K = cot((wc2-wc1)*pi/2)*tan(wc1*pi/2);
12        Dz = - [(K-1)/(K+1), -2*alpha*K/(K+1),1];
13        Nz = [1, -2*alpha*K/(K+1), (K-1)/(K+1)];
14        [b,a] = zp2tf(zd,pd,kd);
15        [b,a] = ZMAPA(b,a,Nz,Dz);
16        [zd,pd,kd] = tf2zp(b,a);
17    case 'rbanda'
18        alpha = cos((wc1+wc2)*pi/2)/cos((wc2-wc1)*pi/2);
19        K = tan((wc2-wc1)*pi/2)*tan(wc1*pi/2);
20        Dz = [(1-K)/(K+1), -2*alpha/(K+1),1];
21        Nz = [1, -2*alpha/(K+1), (1-K)/(K+1)];
22        [b,a] = zp2tf(zd,pd,kd);
23        [b,a] = ZMAPA(b,a,Nz,Dz);
24        [zd,pd,kd] = tf2zp(b,a);
25
26 end
```

Abaixo podem ser visualizados alguns resultados, para $M = 10$, $\omega_{c1} = 0.2$, $\omega_{c2} = 0.5$, $R_p = 10$ e $R_s = 60$:

Figura 15: Filtro Butterworth passa-banda

Figura 16: Filtro Chebyshev passa-altas

Figura 17: Filtro elíptico rejeita-banda

5 Bancos de Filtros

Às vezes é necessário decompor o sinal em diversas sub-bandas de frequência, ou seja, isolar as diferentes componentes de frequência do sinal. Um banco de filtros é um arranjo que, além de poder fazer essa decomposição (análise), permite a recombinação dessas sub-bandas para formar o sinal original (síntese). O filtro digital é um conjunto de filtros passa-banda que ora possui uma entrada em comum (análise) ora uma saída em comum (síntese).

a Alteração da Taxa de Amostragem

Para implementação de um banco de filtros é necessária a realização da mudança da taxa de amostragem de um sinal. Abaixo estão exemplificadas duas maneiras simples de se fazer essa alteração.

A sobreamostragem (fator inteiro L) de um sinal digital pode ser realizada por:

$$y[n] = \begin{cases} x[n/L] & n = 0, \pm L, \pm 2L, \dots, \\ 0 & \text{diferente} \end{cases} \quad (34)$$

Isso significa que L-1 amostras nulas são inseridas entre duas amostras consecutivas do sinal original, e assim o sinal resultante fica L vezes maior. No Matlab a operação é feita como mostrada abaixo:

```
1 y = zeros(1, L*length(x));
2 y([1:L:length(y)]) = x;
```

Pode ser demonstrado que a transformada z do sinal sofre a seguinte mudança:

$$Y(z) = X(z^L) \quad (35)$$

No círculo unitário isso significa:

$$Y(e^{j\omega}) = X(e^{j\omega L}) \quad (36)$$

Indicando que a transformada de Fourier é comprimida pelo fator L.

Do mesmo modo, a subamostragem (fator inteiro M) de um sinal pode ser realizada por:

$$y[n] = x[nM] \quad (37)$$

Dessa forma, cada M-ésima amostra de $x[n]$ é mantida, porém M-1 amostras são descartadas. A taxa de amostragem do sinal $y[n]$ é então $\frac{1}{M}$ da de $x[n]$. No Matlab isso é feito da seguinte forma:

```
1 y = x([1:M:length(x)]);
```

Pode-se demonstrar [4] que a relação entre as transformadas de Fourier é dada por:

$$Y(e^{j\omega}) = \frac{1}{M} \sum_{k=0}^{M-1} X(e^{j(\frac{\omega - 2\pi k}{M})}) \quad (38)$$

Se o sinal $x[n]$ tiver banda limitada em $\pm\pi/M$, não ocorrerá *aliasing* e a expressão acima pode ser simplificada para:

$$Y(e^{j\omega}) = \frac{1}{M} X(e^{j\frac{\omega}{M}}) \quad (39)$$

b Decomposição Polifásica

Para reduzir a complexidade computacional de filtros digitais, utiliza-se o método da representação em decomposição polifásica [4].

Podemos representar a transformada z do sinal $x[n]$ a partir da seguinte expressão:

$$X(z) = \sum_{k=0}^{M-1} z^{-k} X_k(z^M) \quad (40)$$

em que

$$X_k(z) = \sum_{n=-\infty}^{\infty} x_k[n] z^{-n} \quad (41)$$

As sequências $x_k[n]$ são as M componentes polifásicas de $x[n]$, dadas por:

$$x_k[n] = x[Mn + k], \quad k = 0, 1, \dots, M-1 \quad (42)$$

c Bancos de Filtros Digitais

Seja $h_0[n]$ a resposta impulsional de um filtro protótipo passa-baixas causal com frequência de corte em torno de π/M (M inteiro qualquer, representando número de filtros do banco) e $H_0(z)$ sua função de transferência. Definindo a resposta impulsional do k-ésimo filtro do banco como:

$$h_k[n] = h_0[n] e^{j\frac{2\pi kn}{M}} \quad k = 0, 1, \dots, M-1 \quad (43)$$

É simples demonstrar que a resposta em frequência do k-ésimo filtro será dada por:

$$H_k(e^{j\omega}) = H_0(e^{j(\omega - \frac{2\pi k}{M})}) \quad k = 0, 1, \dots, M-1 \quad (44)$$

Ou seja, a resposta de $H_0(z)$ é deslocada para direita pela quantidade $\frac{2\pi k}{M}$.

Acima foi descrito o método de realizar cada filtro do banco de forma separada. Existe outra forma de se obter o mesmo resultado, mas com uma eficiência computacional maior. Essa forma usa a decomposição polifásica do filtro protótipo.

Sendo $E_l(z)$ a l-ésima componente polifásica de $H_0(z)$, a referência [4] desenvolve a seguinte implementação do banco de filtros de análise uniforme usando a IDFT (inversa da *Discrete-Time Fourier Transform*):

Figura 18: Implementação polifásica do banco de filtro de análise uniforme

Da mesma forma, a estrutura para um banco de filtros de síntese uniforme usando a DFT (*Discrete-Time Fourier Transform*) pode ser realizada da seguinte maneira [4] :

Figura 19: Implementação polifásica do banco de filtro de síntese uniforme

Tanto a DFT quanto a IDFT podem ser implementadas a partir do algoritmo da FFT (*Fast Fourier Transform*), o que reduz os custos computacionais dos bancos.

Se nenhuma modificação for feita nas saídas do banco de análise e estas se transformarem nas entradas do banco de síntese, é de se esperar que a saída do banco de síntese seja igual ao sinal original deslocado no tempo.

1 Bancos de Filtros sem Decomposição Polifásica

Abaixo é feita a implementação em Matlab de um banco de filtros de 4 canais, ao se fazer o produto do filtro protótipo por exponenciais. O filtro protótipo é um filtro FIR feito a partir da janela de Blackman de ordem 32.

```
1 N = 32; wc=0.25;a = N/2;
2 %Gerando H0 com janela Blackman
3 for n = 1:N
4     hd(n) = sinc(wc*(n-floor(a)))*wc;
5 end
6 w_black= 0.42 - 0.5* cos(2*pi*(1:N)/(N-1))+0.08*cos(4*pi*(1:N)/(N-1));
7 h0 = hd.* w_black;
8 %Gerando outros filtros do banco
9 for n = 1:length(h0)
10     h1(n) = exp(1i*pi*n/2)*h0(n);
11     h2(n) = (-1)^n*h0(n);
12     h3(n) = exp(1i*pi*n*3/2)*h0(n);
13 end
14 %Resposta em frequencia dos filtros
15 figure
16 subplot(4,1,1)
17 [H0,w] = freqz(h0,1,'whole',2001);
18 plot(w/pi,20*log10(abs(H0)));
19 title('H0')
20 subplot(4,1,2)
21 [H1,w] = freqz(h1,1,'whole',2001);
22 plot(w/pi,20*log10(abs(H1)));
23 title('H1')
24 subplot(4,1,3)
25 [H2,w] = freqz(h2,1,'whole',2001);
26 plot(w/pi,20*log10(abs(H2)));
27 title('H2')
28 subplot(4,1,4)
29 [H3,w] = freqz(h3,1,'whole',2001);
30 plot(w/pi,20*log10(abs(H3)));
31 title('H3')
```

Os filtros do banco são mostrados abaixo:

Figura 20: Banco de filtros de 4 canais - janela de Blackman

Para um filtro IIR, foi necessária a utilização da função ZMAPA, descrita na página 18, para realizar as substituições das exponenciais. A implementação do banco de 4 canais com um protótipo Butterworth de ordem 10 é mostrada a seguir:

```
1 [z,p,k] = Butterworth(10, 0.25, 0.7, 'baixa');
2 [b0,a0] = zp2tf(z,p,k);
3 %Gerando outros filtros do banco
4 [b1,a1] = ZMAPA(b0,a0,[exp(-1i*pi/2) 0],1);
5 [b2,a2] = ZMAPA(b0,a0,[1 0],1);
6 [b3,a3] = ZMAPA(b0,a0,[exp(1i*pi/2) 0],1);
7 %Resposta em frequencia dos filtros
8 subplot(4,1,1)
9 [H0,w] = freqz(b0,a0,'whole',2001);
10 plot(w/pi,20*log10(abs(H0)));
11 title('H0')
12 subplot(4,1,2)
13 [H1,w] = freqz(b1,a1,'whole',2001);
14 plot(w/pi,20*log10(abs(H1)));
15 title('H1')
16 subplot(4,1,3)
17 [H2,w] = freqz(b2,a2,'whole',2001);
18 plot(w/pi,20*log10(abs(H2)));
19 title('H2')
20 subplot(4,1,4)
21 [H3,w] = freqz(b3,a3,'whole',2001);
22 plot(w/pi,20*log10(abs(H3)));
23 title('H3')
```

Os filtros do banco são mostrados abaixo:

Figura 21: Banco de filtros de 4 canais - filtro Butterworth

Para a seguinte entrada:

```
1 x = cos(pi*0.1*(1:200)) + cos(pi*0.9*(1:200));
```

Figura 22: Entrada 1

São geradas as saídas:

Figura 23: Saídas banco de filtros FIR (Blackman)

Figura 24: Saídas banco de filtros IIR (Butterworth)

E para esse segunda entrada:

```
1 x = cos(pi*0.5*(1:200));
```

Figura 25: Entrada 2

São geradas as saídas:

Figura 26: Saídas banco de filtros FIR (Blackman)

Figura 27: Saídas banco de filtros IIR (Butterworth)

2 Bancos de Filtros com Decomposição Polifásica

Abaixo a implementação do banco usando decomposição polifásica e a FFT é mostrada. O número de canais é arbitrário, mas foram utilizados 8 canais como forma de exemplificação:

```
1 N_FIL = 8; %numero de canais
2 N = 23; % Ordem do filtro
3 h0 = fir1(N,0.3,'low'); % filtro prototipo
4 w = linspace(0,2*pi,128); % pontos de frequencia
5 % Completando o filtro com zeros
6 if (mod(N,N_FIL)~=0)
```



```

7   h0=[h0, zeros(1,N_FIL-mod(N+1,N_FIL))];
8   end
9   % Decomposição Polifásica
10  for i=1:N_FIL
11      e(i,:) = h0(i:N_FIL:length(h0));
12      eM(i,:)= zeros(1,N_FIL*size(e,2)); %sobreamostragem
13      eM(i,[1:N_FIL:size(eM,2)])= e(i,:);
14  end
15  % Atrasos
16  eM = [eM, zeros(N_FIL,N_FIL-1)];
17  for k = 1:N_FIL
18      eM(k,:) = circshift(eM(k,:),[2,k-1]);
19  end
20  % Matriz FFT
21  W = ifft(eye(N_FIL));
22  % Banco de filtros
23  hp = zeros(N_FIL,N+1);
24  hpf = zeros(N_FIL,length(w));
25  hp = N_FIL*W*eM;
26  %Gráficos
27  for k = 1:N_FIL
28      hpf(k,:) = freqz(hp(k,:),1,w);
29      subplot(N_FIL,1,k);
30      plot(w/pi,20*log10(abs(hpf(k,:))));
31      axis([0 2 -70 10]);
32      title(['H',num2str(k-1)])
33  end

```

O código acima gera o seguinte gráfico:

Figura 28: Banco de filtros com decomposição polifásica

Pode-se aplicar a seguinte entrada ao banco:

```

1   x = cos(pi*0.1*(1:100)) ; % entrada
2   figure
3   for i=1: N_FIL
4       subplot(N_FIL,1,i)
5       v(i,:) = filter(hp(i,:), 1, x);
6       stem(v(i,:));
7       title(['Saída H',num2str(i-1)])
8       ylim([-1 1])

```

9 end

As saídas do banco são:

Figura 29: Saídas do banco de filtros

Se nenhuma manipulação for feita a essas saídas, é de se esperar que, ao se implementar a etapa de síntese, retornaremos ao sinal original (com algum retardo). Dessa forma:

```
1 %síntese
2 %FFT
3 for i=1:size(v,2)
4     u(:,i) = fft(v(:,i));
5 end
6
7 %Filtrando e deslocando
8 y=0;
9 for i = 1: N_FIL
10     Y(i,:) = filter(eM(i,:),1,u(i,:));
11     Y(i,:) = [ zeros(1,i-1), Y(i,1:size(Y,2)-(i-1))];
12     y = y+Y(i,:);
13 end
14
15 %Comparando sinal original e reconstruído
16 figure
17 subplot(2,1,1)
18 stem(x);
19 title('Sinal original')
20 subplot(2,1,2)
21 stem(real(y));
22 title('Sinal reconstruído')
23 ylim([-1 1])
```

Figura 30: Sinais original e reconstruído

De fato, foi possível recuperar o sinal original após algum retardo.

6 Interface Gráfica no Matlab

Os filtros e estruturas desenvolvidos nas seções anteriores foram agrupados em uma interface gráfica implementada no Matlab. Assim, o usuário desta interface poderá especificar uma entrada digital desejada, selecionar o filtro adequado e observar sua saída.

Abaixo está a visão geral do programa:

Figura 31: Visão Geral da Interface

A interface se subdivide em três painéis: Entrada (à esquerda), Filtro (central) e Saída (a direita).

a Painel de Entrada

No painel de entrada, podemos observar dois gráficos (sinal no tempo e sinal na frequência) e uma caixa de texto editável, onde será inserido pelo usuário o sinal de entrada.

Figura 32: Inserção do sinal de entrada

Após a inserção do texto, o usuário deverá apertar o botão “Gerar Entrada”, que efetivamente transforma o texto digitado na entrada do sistema e gera os gráficos de suas respostas no tempo e na frequência.

Figura 33: Geração do sinal de entrada

b Painel do Filtro

Dentro do painel do filtro há três painéis e um gráfico. O painel superior (Especificações) permite selecionar as propriedades gerais do filtro desejado (tanto para filtros FIR quanto para os filtros IIR):

Figura 34: Especificações gerais do filtro

Acima, na caixa de texto editável, pode-se inserir a ordem do filtro. Logo abaixo, há um menu *pop-up* onde o usuário pode escolher o tipo de filtro (passa-baixas, passa-altas, passa-banda ou rejeita-banda). Em seguida, pode-se especificar a(s) frequência(s) de corte (normalizadas) ω_{c1} (e ω_{c2}). Para os filtros passa-baixas e passa-altas, só é necessário especificar ω_{c1} , portanto o programa desabilita ω_{c2} quando um desses filtros é selecionado, e habilita quando o filtro rejeita-banda ou passa-banda é selecionado.

O painel inferior esquerdo permite a seleção do tipo de janela desejada para projeto de filtros FIR:

Figura 35: Especificações do filtro FIR

Ele é constituído por um menu *pop-up* onde a janela pode ser escolhida, e um botão “Gerar Filtro” que efetivamente gera o filtro no sistema (considerando as especificações escolhidas no painel “Especificações”) e plota o gráfico da sua resposta em frequência.

Por último, o painel inferior direito (IIR) permite escolher o tipo de filtro IIR e especificar a oscilação em dB da banda de passagem (R_p) e a atenuação em dB na banda de rejeição (R_s):

Figura 36: Especificações do filtro IIR

Para o filtro Butterworth, R_p e R_s são desabilitados, para o filtro Chebyshev somente é possível especificar a oscilação na banda de passagem e no filtro Elíptico ambos são habilitados.

Da mesma forma, o botão “Gerar Filtro” efetivamente gera o filtro no sistema (considerando as especificações escolhidas no painel “Especificações”) e plota o gráfico da sua resposta em frequência.

Abaixo pode ser visualizado o painel de filtro depois que um filtro Elíptico passa-baixas de ordem 8, frequência de corte normalizada 0.3, com oscilação na banda de passagem de 10dB e atenuação na banda de rejeição de 60dB foi gerado:

Figura 37: Painel Filtro

c Painel de Saída

No painel de saída há dois gráficos e um botão “Gerar Saída” que, após gerados a entrada e o filtro, ao ser pressionado, gera a saída do sistema e os gráficos de suas respostas no tempo e na frequência. Abaixo pode-se ver a saída gerada a partir da entrada da figura 33 e do filtro da figura 37:

Figura 38: Painel de Saída

Por fim, abaixo é apresentado o estado da interface depois que os passos anteriores foram executados:

Figura 39: Visão geral da interface

7 Conclusão

Neste trabalho foram estudadas diversas técnicas de projeto de filtros digitais. Para o caso de filtros do tipo FIR, a técnica de janelamento foi abordada; e para filtros IIR, foi feito um estudo de projeto de filtros analógicos e de uma técnica de mapeamento para o domínio digital: a transformação bilinear. Técnicas de transformação em frequência foram usadas de forma a obter filtros que não são passa-baixas. Com esses conhecimentos, foi possível o desenvolvimento de rotinas em Matlab para execução de cada tipo específico de filtro. Também foi realizado um estudo de mudança da taxa de amostragem de um sinal, além de sua representação em decomposição polifásica. Dessa forma se realizou o desenvolvimento de rotinas em Matlab de bancos de filtros com um número arbitrário de canais e com o uso da transformada rápida de Fourier, reduzindo os gastos computacionais.

A partir da pesquisa dessas diversas técnicas e do desenvolvimento das rotinas, uma interface gráfica do usuário reunindo todos os elementos foi implementada com ajuda do GUIDE (*Graphical User Interface Design Environment*), uma ferramenta do Matlab que possibilita o desenvolvimento de interfaces gráficas. A interface construída apresenta uma visualização simples e intuitiva para que seu usuário possa interagir. Com ela, o projeto de filtros digitais é facilitado.

Referências

- [1] J. Moura, "What is signal processing? [president's message]," *Signal Processing Magazine, IEEE*, vol. 26, pp. 6–6, Nov 2009.
- [2] "Fifty years on signal processing," *The IEEE Signal Processing Society*, 1998.
- [3] D. Stranneby, *Digital signal processing and applications*. Elsevier, 2004.
- [4] S. K. Mitra and Y. Kuo, *Digital signal processing: a computer-based approach*, vol. 2. McGraw-Hill New York, 2006.
- [5] J. O. Smith, *Introduction to digital filters: with audio applications*, vol. 2. Julius Smith, 2007.
- [6] A. V. Oppenheim, R. W. Schaffer, J. R. Buck, et al., *Discrete-time signal processing*, vol. 2. Prentice-hall Englewood Cliffs, 1989.
- [7] F. Yu, Z. Lu, H. Luo, and P. Wang, *Three-dimensional model analysis and processing*. Springer Science & Business Media, 2011.
- [8] Q. Li, W.-G. Chen, C. He, and H. Malvar, "Design of oversampled dft modulated filter banks optimized for acoustic echo cancellation," in *International Conference on Acoustics, Speech and Signal Processing*, Institute of Electrical and Electronics Engineers, Inc., April 2009.
- [9] H. Bode, "History of electronic sound modification," *Journal of the Audio Engineering Society*, vol. 32, no. 10, pp. 730–739, 1984.
- [10] V. Ingle and J. Proakis, *Digital signal processing using MATLAB*. Cengage Learning, 2011.
- [11] S. J. Orfanidis, "Lecture notes on elliptic filter design," 2006.