UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA
DEPARTAMENTO DE COMUNICAÇÕES

Um Sistema para o Projeto de Filtros Digitais Recursivos Descritos por Variáveis de Estado

Narcizo Sabbatini Junior

Orientador: Prof. Dr. Amauri Lopes

Este exemplar corresponde à redação final da tese defendida por Amauri Lopes e aprovada pela Comissão Julgadora em 11/07/90.

Orientador

Dissertação submetida como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica.

julho de 1990

FEC/FEE
1 2 0 11

C m 0000 81866

Sumário

Este trabalho descreve os aspectos teóricos e práticos envolvidos na elaboração do programa de computador FOREST, que sintetiza e analisa filtros digitais recursivos. São sintetizados filtros passa-baixa, passa-alta, passa-faixa e corta-faixa, utilizando-se as aproximações de Butterworth, Chebychev e elíptica. Os efeitos não lineares advindos da utilização de registros de comprimento finito para a representação de coeficientes e variáveis são analisados em detalhe. Apresenta-se a teoria de otimização dos filtros digitais com relação ao ruído de quantização do sinal, baseada na descrição por variáveis de estado, e o programa incorpora essa teoria gerando filtros descritos por variáveis de estado com reduzidos efeitos de quantização.

Agradecimentos

Gostaria de agradecer a todos que de alguma forma tenham me ajudado na realização deste trabalho, seja através de discussões, sugestões ou críticas. Minha maior dívida de gratidão porém, é para com o meu orientador, Prof. Amauri Lopes, cujo apoio e incentivo foram fundamentais para a realização deste trabalho. De fato, sem sua orientação segura e eficiente, sempre pautada pela cordialidade e pela amizade, e suas meticulosas revisões editoriais, muito provavelmente este trabalho não teria sido concluído com sucesso.

Agradeço também à Coordenação de Áreas de Circuitos Integrados do Centro de Pesquisas e Desenvolvimento da TELEBRÁS, onde trabalho, pelo apoio e pelo incentivo à realização deste trabalho, fornecendo infraestrutura e suporte computacional.

Finalmente, agradeço à minha esposa Carmen, pelo seu incentivo e pela compreensão nas horas em que deixei de lhe fazer companhia. A ela dedico este trabalho.

para *Isabela*

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Dos filtros analógicos aos filtros digitais

Um dos processos fundamentais em sistemas eletrônicos é o da filtragem de sinais. A maneira pela qual este problema é resolvido tem evoluído ao longo das últimas décadas, reflexo da grande evolução tecnológica observada na área dos componentes eletrônicos: das válvulas a vácuo aos circuitos integrados em menos de quatro décadas. No início (e ainda hoje!) os filtros eram construídos com elementos passivos: resistores, capacitores e indutores. Sinais elétricos de tensão e corrente aplicados a esses componentes se relacionam através de operações matemáticas de multiplicação por escalar, integração e diferenciação, exatamente as operações necessárias para se implementar as equações diferenciais que descrevem o filtro. Este método foi, e ainda é, empregado com sucesso, porém ele apresenta muitas limitações. Uma delas é a imprecisão nos valores característicos dos componentes (da ordem de alguns por cento), porém pior que isto é a impossibilidade que se tem, na prática, de se obter componentes que sejam somente resistores, somente capacitores ou somente indutores. Normalmente eles apresentam efeitos parasitas que são muito difíceis de se levar em conta no projeto do filtro. Para frequências de até algumas dezenas de MHz, os resistores e capacitores são praticamente "ideais", mas os indutores apresentam uma forte componente resistiva muito difícil de se eliminar. Essa componente resistiva provoca dois efeitos: uma perda de potência do sinal e uma deformação da resposta em frequência, a qual normalmente é corrigida através de ajustes manuais, filtro a filtro, na bancada.

Com o advento dos primeiros circuitos integrados, e em particular dos amplificadores operacionais, tornou-se possível a eliminação dos indutores com a construção de filtros ativos. Esses filtros solucionam equações diferenciais da mesma forma que os antigos computadores analógicos o faziam, utilizando apenas resistores e capacitores, além é claro dos próprios ampli-

ficadores operacionais. A eliminação dos indutores resolve, em boa parte, o problema dos elementos parasitas, e oferece como vantagem adicional a possibilidade de que o filtro possa vir a ser fabricado completamente dentro de um circuito integrado. Infelizmente, dentro de um circuito integrado não se consegue fazer capacitores muito grandes (no máximo, alguns pF), e os resistores gigantescos necessários para manter as constantes RC simplesmente não podem ser fabricados com precisão razoável. Portanto, quando se busca integração, outra solução deve ser procurada.

Uma evolução no sentido de se permitir a integração dos filtros foi obtida com a técnica de capacitores chaveados [1]. A idéia desta técnica é, a grosso modo, a substituição dos resistores por chaves que se abrem e fecham periodicamente, bombeando carga elétrica entre os capacitores. Ao invés do produto RC, esta técnica requer que a razão entre capacitâncias seja bem controlada, o que em circuitos integrados pode ser feito facilmente com precisão da ordem de 0,1%. Este tipo de filtro já mostra uma certa tendência à discretização dos sinais, pois as tensões e correntes no interior do filtro só se alteram a cada abertura e fechamento das chaves, fazendo com que os sinais adquiram um formato de escada semelhante àquele que aparece na saída de um circuito "sample and hold". No entanto, o filtro continua sendo analógico. Continua existindo o problema das capacitâncias parasitas, do ruído nas linhas de alimentação, do "lay-out" que somente projetistas experientes podem fazer, além de problemas totalmente novos, como a indução de carga nos capacitores pela ação do relógio nas chaves ("clock feedthrough" [1]). Além disso, existe um limite de razão máxima entre capacitores que se pode implementar, restringindo a gama de aplicações possíveis.

E chegamos, finalmente, aos filtros totalmente digitais. Por trás da utilização desses filtros está uma mudança sutil no modo como visualizamos e modelamos os sistemas. Ao invés de representá-los através de equações diferenciais, passamos a representá-los por equações a diferenças. Os sinais passam a ser discretos no tempo, e a matemática utilizada para representá-los é auto-consistente e completamente independente da matemática dos sinais contínuos. Os filtros digitais simplesmente implementam algoritmos de cálculo em tempo real e, portanto, são insensíveis a variações ambientais, são perfeitamente reprodutíveis e são facilmente integráveis. Eles não são, entretanto, perfeitos. Como veremos nos capítulos seguintes, a limitação do número de bits com os quais as variáveis do algoritmo são representadas leva a efeitos não ideais que, no entanto, não chegam nem de longe a comprometer os filtros digitais frente a seus concorrentes analógicos.

1.2 Filtros recursivos versus não-recursivos

O advento das técnicas digitais e o amadurecimento da teoria do processamento digital de sinais tornaram possíveis implementações de algoritmos que antes eram impensáveis de se realizar com o processamento analógico. Da manipulação de grandezas físicas (tensão e corrente) ao puro e simples cálculo numérico dos algoritmos ocorreu uma verdadeira revolução na maneira de se implementar sistemas eletrônicos. Não só as aplicações se multiplicam, como também as formas de implementação são variadas, reflexo da flexibilidade de escolha dos algoritmos.

No caso dos filtros digitais, essa flexibilidade se traduz na possibilidade de escolha entre os filtros recursivos, ou IIR[1], e os não-recursivos, ou FIR[2]. Muito se discute na literatura sobre as vantagens e desvantagens de um filtro em relação ao outro [2,13,22]. Os filtros FIR são sempre estáveis e podem ser feitos com fase perfeitamente linear. São, portanto, ideais para sistemas adaptativos e sistemas que não tolerem distorção de fase. Já os filtros IIR são extremamente eficientes em termos de complexidade de implementação (número de atrasadores e de multiplicadores), principalmente para filtros muito seletivos. Assim, a única coisa que se pode afirmar com certeza é que a escolha de um ou outro filtro depende da aplicação em particular.

Do ponto de vista do nosso trabalho, optamos por estudar e implementar um programa para o projeto de filtros IIR. Este tipo de filtro é o que apresenta os maiores problemas com relação à precisão finita dos registros, mas justamente devido a este fato, nos oferece um maior desafio e potencial de aprendizado. Acreditamos que os conceitos e os resultados obtidos neste estudo possam ser estendidos a outros tipos de filtros e a outras estruturas de processamento digital.

1.3 Resumo dos capítulos seguintes

No Capítulo 2 apresentaremos o método de projeto de filtros digitais recursivos através da transformação bilinear de um filtro protótipo analógico. Apresentaremos, em detalhe, a teoria de cálculo das aproximações de Butterworth, Chebychev e elíptica.

No Capítulo 3 faremos um apanhado dos efeitos da precisão finita dos registros nos filtros digitais. Analisaremos os problemas da quantização dos coeficientes, do escalonamento dos registros para se evitar "overflow" e do ruído de quantização do sinal, tanto em aritmética de ponto fixo como de ponto flutuante.

No Capítulo 4 será apresentada a teoria dos filtros ótimos, baseada numa

[1] do inglês Infinite Impulse Response
[2] do inglês Finite Impulse Response

descrição matricial muito elegante e poderosa dos filtros digitais: a descrição por variáveis de estado. Os filtros ótimos desenvolvidos neste capítulo são notáveis por apresentarem bom desempenho em relação aos efeitos de quantização, quando comparados às formas tradicionais de implementação.

No Capítulo 5 descreveremos o programa FOREST, que sintetiza filtros digitais IIR na forma de cascata de seções de segunda ordem, geradas na forma direta e na forma ótima. Serão apresentados, também, exemplos de aplicação do programa, os quais ilustrarão a teoria apresentada nos capítulos anteriores.

Finalmente, no Capítulo 6 apresentaremos as observações finais, conclusões e sugestões para continuação do trabalho.

Capítulo 2

O projeto de filtros digitais recursivos

2.1 Introdução

Neste capítulo apresentaremos o método de projeto de filtros digitais recursivos baseado no projeto de um filtro analógico correspondente. Este método de cálculo tem a grande vantagem de utilizar todo o arcabouço de conhecimento existente para o projeto de filtros analógicos, acumulado durante décadas e consagrado pela prática.

A idéia central do método é o estabelecimento de algum tipo de mapeamento entre o plano s, no qual se projeta os filtros analógicos, e o plano z dos filtros digitais. Para que seja de alguma utilidade, este mapeamento deve preservar as características essenciais da resposta em frequência e a estabilidade do filtro. Para tanto, deve mapear o eixo imaginário do plano s na Circunferência de Raio Unitário (C.R.U.), e o semi-plano esquerdo do plano s, no interior da C.R.U. O tipo de mapeamento mais utilizado, e o único que adotaremos aqui, é o da transformação bilinear [2], que analisaremos a seguir.

2.2 A transformação bilinear e o projeto do filtro

A idéia que está por trás da transformação bilinear é a transformação de uma equação diferencial em uma equação a diferenças utilizando-se a regra do trapézio para a aproximação das integrais [2]. Sendo T o período de amostragem (ou o espaçamento de cálculo da regra do trapézio), a transformação bilinear é dada pelo seguinte mapeamento entre s e z:

$$s = \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}} \tag{2.1}$$

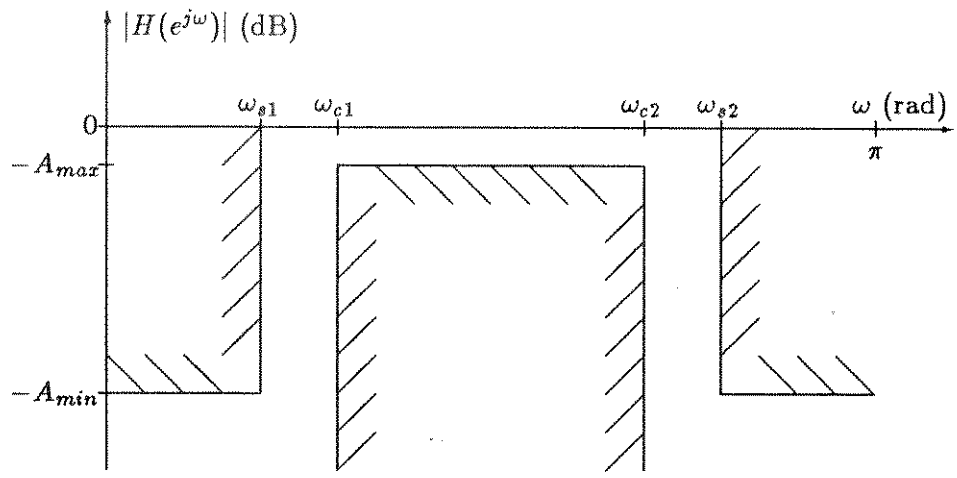

Figura 2.1: Máscara de especificações de um filtro passa-faixa.

Vamos mostrar agora a sequência de cálculos envolvida no projeto de um filtro digital recursivo, e como a transformação bilinear se encaixa neste processo. Com relação às especificações do filtro, poderíamos partir do pressuposto de que elas são fornecidas diretamente em termos de ângulos relacionados à C.R.U. do plano z. Este tipo de enfoque teria a vantagem de realçar o fato de que o filtro digital não é uma aproximação do filtro analógico, mas ao invés disso, utiliza um filtro analógico como etapa intermediária de projeto do filtro digital. Não obstante, quando se processa sinais analógicos, estes são convertidos para sinais discretos através de um processo de amostragem que, no nosso entender, seria interessante deixar explícito. Uma vantagem disto, por exemplo, é a possibilidade de se avaliar o efeito da variação da frequência de amostragem no desempenho do filtro com relação aos efeitos de comprimento finito dos registros. Seguindo-se esta linha, a especificação do filtro seria composta pela frequência de amostragem e pelas frequências limites das faixas de passagem e rejeição em Hz, com as respectivas atenuações máxima e mínima. Essas frequências são, então, convertidas para ângulos da C.R.U. através da relação:

$$\omega = \frac{2\pi f}{f_a} \tag{2.2}$$

onde ω é o ângulo em radianos da C.R.U., f é a frequência em Hz a ser convertida e $f_a = 1/T$ é a frequência de amostragem. Essas especificações definem uma máscara na qual a resposta em frequência do filtro deve se encaixar, como aquela mostrada na Fig. 2.1.

Depois de obtidas as especificações no plano z, deve-se convertê-las em

especificações de um filtro analógico equivalente no plano s. Para tanto, observemos que a relação entre um ângulo da C.R.U. e uma frequência no eixo $j\Omega$ do plano s é dada por:

$$\Omega = \frac{2}{T} \tan \frac{\omega}{2} \tag{2.3}$$

Para simplicidade de cálculo, utilizaremos a expressão acima com $T = 2$. Observe que T não precisa necessariamente ser o inverso da frequência de amostragem definida anteriormente, desde que utilizemos o mesmo valor de T quando fizermos a transformação bilinear do filtro analógico. Na verdade, o único papel desempenhado pela frequência de amostragem é na obtenção das especificações do filtro no plano z. Nas expressões da transformação bilinear, o seu valor é irrelevante.

Uma vez obtida a especificação do filtro analógico correspondente, pode-se adotar qualquer das técnicas disponíveis para o projeto de filtros analógicos. Este é o assunto da Seção 2.3. De posse dos pólos e zeros do filtro analógico, podemos obter os pólos e zeros do filtro digital através da transformação bilinear. Fazendo $T = 2$ na equação (2.1) e resolvendo para z, obtemos:

$$z = \frac{1+s}{1-s} \tag{2.4}$$

Ao se fazer a transformação bilinear dos pólos e zeros pela expressão acima, é preciso calcular o efeito da transformação sobre o fator de ganho constante do filtro. Seja, por exemplo, um termo de primeira ordem em s da forma $(s - s_z)/(s - s_p)$. Substituindo-se s pelo valor dado por (2.1) com $T = 2$, temos:

$$\frac{(1-z^{-1})/(1+z^{-1}) - s_z}{(1-z^{-1})/(1+z^{-1}) - s_p} = \frac{1-s_z}{1-s_p} \frac{z - (1+s_z)/(1-s_z)}{z - (1+s_p)/(1-s_p)} \tag{2.5}$$

Na expressão acima vemos que o pólo e o zero foram transformados segundo (2.4), sendo que o fator de ganho constante foi alterado pelo fator $(1 - s_z)/(1 - s_p)$. Deste resultado se conclui que, ao se transformar um pólo real s_p, deve-se dividir o ganho por $(1 - s_p)$, e ao se transformar um zero real s_z, deve-se multiplicar o ganho por $(1 - s_z)$. No caso de pólos e zeros complexos, lembremos que eles sempre aparecem em pares conjugados. Assim sendo, se s_p for um pólo de $H(s)$, s_p^* também o será. Da mesma forma, se s_z for um zero de $H(s)$, s_z^* também será. Portanto, ao se aplicar a transformação bilinear a um pólo complexo s_p, deve-se dividir o ganho por:

$$(1 - s_p)(1 - s_p^*) = 1 - 2\Re(s_p) + |s_p|^2$$

Analogamente, ao se aplicar a transformação bilinear a um zero s_z, deve-se multiplicar o ganho por:

$$(1 - s_z)(1 - s_z^*) = 1 - 2\Re(s_z) + |s_z|^2$$

Uma vez obtidos os pólos e zeros no plano z, vemo-nos frente ao problema de agrupar esses pólos e zeros para a formação da função de transferência do filtro. Podemos, por exemplo, agrupar todos os N pólos e os M zeros em polinômios de graus N e M, respectivamente, e expressar a função de transferência na forma:

$$H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_M z^{-M}}{1 + c_1 z^{-1} + \cdots + c_N z^{-N}} \tag{2.6}$$

Entretanto, como veremos mais tarde, um filtro digital que implemente diretamente a expressão acima (daí receber o nome de Forma Direta) é muito sensível aos efeitos de comprimento finito dos registros e, portanto, raramente é utilizado. Muito mais comum é o agrupamento de pólos e zeros em seções de primeira e segunda ordem, as quais são depois associadas em paralelo ou em cascata para a realização de $H(z)$. Mostraremos depois que tais estruturas são menos sensíveis aos efeitos de quantização. Escolhendo a associação em cascata, podemos expressar $H(z)$ na forma:

$$H(z) = A \prod_{k=1}^{N_1} \frac{1 + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 - c_{1k} z^{-1} - c_{2k} z^{-2}} \tag{2.7}$$

onde N_1 é a parte inteira de $(N+1)/2$.

Note que esta expressão só possui termos de segunda ordem. Isto não chega a ser problema, pois se N for ímpar, alguns dos coeficientes serão nulos. Mais explicitamente, se houver um pólo ou zero real que não possa ser agrupado com outro para formar um termo de segunda ordem, os coeficientes do termo correspondente serão:

$$b_{2k} = 0 \tag{2.8}$$
$$c_{2k} = 0 \tag{2.9}$$
$$b_{1k} = -\alpha \tag{2.10}$$
$$c_{1k} = \beta \tag{2.11}$$

onde α e β são os zeros e pólos reais respectivamente.

Para os pólos e zeros complexos, os coeficientes dos termos de segunda ordem são dados por:

$$b_{1k} = -2\Re(p_k) \tag{2.12}$$
$$b_{2k} = |p_k|^2 \tag{2.13}$$

$$c_{1k} = 2\Re(q_k) \quad (2.14)$$
$$c_{2k} = -|q_k|^2 \quad (2.15)$$

onde p_k e q_k são os zeros e pólos do termo k de $H(z)$.

É importante fazer um último comentário sobre a formação da função de transferência. Existe alguma arbitrariedade no agrupamento de pólos e zeros para a formação das seções de segunda ordem, assim como na ordenação das seções. Na verdade, como mostraremos no Capítulo 4, existem infinitas maneiras de se implementar a função de transferência $H(z)$, todas elas idênticas para precisão de cálculo infinita. Quando se considera registros de comprimento finito, entretanto, todas elas possuem desempenhos diferentes, fato que precisa ser levado em conta no projeto do filtro. Este será o assunto dos capítulos seguintes. Antes, porém, apresentaremos a teoria de projeto dos filtros analógicos, etapa fundamental e nada trivial no projeto do filtro digital recursivo.

2.3 O projeto de Filtros Analógicos

2.3.1 Métodos de Aproximação

O problema que se nos coloca aqui é o da determinação de uma função matemática para a representação da resposta em frequência, o chamado problema da aproximação. Essa função matemática deverá satisfazer duas condições básicas: ter um formato tal que se encaixe dentro da máscara de especificações e ser passível de implementação física. Esta última condição poderá ser satisfeita se a função de aproximação for expressa através da razão entre dois polinômios, já que no domínio do tempo tal função é convertida para uma equação diferencial, no caso do plano s, ou para uma equação a diferenças, no caso do plano z[1].

Existem duas metodologias diferentes que podem ser utilizadas para a obtenção de uma função racional de polinômios que atenda à máscara de especificações. Uma delas é a utilização de algoritmos numéricos iterativos que façam uma varredura sistemática dos coeficientes dos polinômios até que a função caia dentro da máscara. A outra metodologia, a que utilizaremos, procura por funções matemáticas polinomiais cujos gráficos apresentem características de seletividade parecidas com as de um filtro passa-baixa. Essas funções são, então, utilizadas para a aproximação de um filtro protótipo passa-baixa, que depois é transformado para o tipo desejado – passa-alta, passa-faixa ou corta-faixa – através de uma simples

[1]Uma função racional de polinômios pode representar qualquer circuito linear e invariante no tempo que possua somente parâmetros concentrados. Isto não se aplica a circuitos com parâmetros distribuídos, que contenham elementos como linhas de transmissão ou guias de onda.

transformação de variáveis.

Nas seções seguintes apresentaremos os três tipos de aproximação que utilizaremos – Butterworth, Chebychev e elíptica – e depois mostraremos como é feita a transformação do protótipo passa-baixa para os demais tipos.

2.3.2 Aproximação de Butterworth

Este e os outros métodos de aproximação que apresentaremos tem como objetivo expressar a função de transferência $H(s)$ na forma da razão de dois polinômios:

$$H(s) = \frac{q(s)}{e(s)} \tag{2.16}$$

No método de aproximação de Butterworth, as primeiras $2n-1$ derivadas da função de atenuação de potência são feitas iguais a zero no ponto $\Omega = 0$, ou seja:

$$\frac{d^i A(\Omega)}{d\Omega^i} = 0 \qquad i = 1, 2, \cdots, 2n-1 \tag{2.17}$$

onde

$$A(\Omega) = \left| \frac{1}{H(j\Omega)} \right|^2 \tag{2.18}$$

é a atenuação de potência imposta pelo filtro.

A função de atenuação que atende à condição acima é dada por:

$$A(j\Omega) = 1 + \epsilon^2 \left(\frac{\Omega}{\Omega_c} \right)^{2n} \tag{2.19}$$

onde Ω_c é a frequência limite da faixa de passagem.

Usando a expressão acima em (2.18), temos:

$$|H(j\Omega)|^2 = \frac{1}{1 + \epsilon^2 (\Omega/\Omega_c)^2} \tag{2.20}$$

O parâmetro ϵ está ligado à atenuação máxima na faixa de passagem A_{max} (no ponto $\Omega = \Omega_c$), que pode ser expressa em dB como:

$$A_{max} = 10 \log(1 + \epsilon^2) \tag{2.21}$$

donde:

$$\epsilon = \sqrt{10^{0,1 A_{max}} - 1} \tag{2.22}$$

Sendo A_{min} e Ω_s, respectivamente, a atenuação mínima e a frequência de início da faixa de rejeição, podemos escrever:

$$A_{min} = 10\log[1 + \epsilon^2(\Omega_s/\Omega_c)^{2n}] \tag{2.23}$$

Rearranjando-se esta expressão, obtemos:

$$\epsilon^2 \left(\frac{\Omega_s}{\Omega_c}\right)^{2n} = 10^{0,1A_{min}} - 1 \tag{2.24}$$

Substituindo-se (2.22) em (2.24) e isolando n, chegamos a:

$$n = \frac{\log \sqrt{(10^{0,1A_{min}} - 1)/(10^{0,1A_{max}} - 1)}}{\log(\Omega_s/\Omega_c)} \tag{2.25}$$

Esta expressão fornece o valor da ordem mínima do filtro que satisfaz as especificações de máscara. Obviamente, n deve ser inteiro, de forma que se deve tomar o inteiro imediatamente acima do valor fornecido pela equação.

Generalizando a expressão (2.20), podemos expressar o termo $H(s)H(-s)$ na forma:

$$H(s)H(-s) = \frac{1}{1 + \epsilon^2(s/j\Omega_c)^{2n}} \tag{2.26}$$

Como o valor de n é dado pela expressão (2.25), resta-nos encontrar as raízes do denominador da equação acima, que são os pólos de $H(s)H(-s)$, e escolher as raízes do semiplano esquerdo, que correspondem aos pólos de $H(s)$. Igualando o denominador a zero para se encontrar as raízes, temos:

$$1 + \epsilon^2 \left(\frac{s}{j\Omega_c}\right)^{2n} = 0 \tag{2.27}$$

donde se extrai os pólos de $H(s)H(-s)$:

$$s_p = \frac{\Omega_c}{\epsilon^{1/n}} \exp\left[j\frac{2i + n + 1}{2n}\pi\right] \qquad i = 1, 2, \cdots, 2n \tag{2.28}$$

Os pólos calculados segundo a expressão acima se distribuem uniformemente em cima de uma circunferência de raio $\Omega_c/\epsilon^{1/n}$. Os pólos do semiplano esquerdo pertencem a $H(s)$, e os pólos do semiplano direito pertencem a $H(-s)$. Para facilitar o cálculo por computador, os pólos podem ser renumerados de forma que os n primeiros pólos estejam no semiplano esquerdo. Fazendo-se isto, os pólos de $H(s)$ serão dados por:

$$s_{p_i} = \frac{\Omega_c}{\epsilon^{1/n}} \exp\left[j\frac{2i + n - 1}{2n}\pi\right] \qquad i = 1, 2, \cdots, n \tag{2.29}$$

Conhecidos os pólos, $H(s)$ pode ser expressa na forma:

$$H(s) = \frac{\Omega_c^n}{\epsilon \prod_{i=1}^{n}(s - s_{p_i})} \tag{2.30}$$

onde o fator Ω_c^n/ϵ apareceu devido à conveniência de se ter $|H(j\Omega)| = 1$ para $\Omega = 0$.

2.3.3 Aproximação de Chebychev

No caso do filtro de Butterworth, o desvio da resposta em frequência em relação à resposta ideal é zero em $\Omega = 0$ e vai aumentando até atingir um máximo em $\Omega = \Omega_c$. A idéia da aproximação de Chebychev é a de distribuir o erro da resposta em frequência ao longo da faixa de passagem do filtro, mantendo a resposta monotônica na faixa de rejeição. Com isto se consegue, para uma mesma ordem do filtro, menor largura da faixa de transição. Assim como no caso da aproximação de Butterworth, a aproximação de Chebychev se utilizará de um polinômio para exprimir a função de atenuação do filtro. Esse polinômio será o *polinômio de Chebychev* de grau n, denotado por $T_n(x)$.

Para satisfazer os requisitos de resposta ondulante na faixa de passagem e monotônica na faixa de rejeição, $T_n(x)$ deve apresentar as seguintes propriedades:

1. T_n é par (ímpar) se n for par (ímpar).
2. T_n tem todos os seus zeros no intervalo $-1 < x < 1$.
3. T_n oscila entre ± 1 no intervalo $-1 < x < 1$.
4. $T_n(1) = +1$.

A partir das propriedades acima, pode-se mostrar [3] que $T_n(x)$ pode ser escrito como:

$$T_n(x) = \cos(n \cos^{-1} x) \tag{2.31}$$

O caráter polinomial da expressão acima pode ser percebido se descrevermos os polinômios de Chebychev através da seguinte relação de recorrência:

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x) \tag{2.32}$$

onde:

$$T_0(x) = 1 \tag{2.33}$$
$$T_1(x) = x \tag{2.34}$$

A partir dos polinômios de Chebychev, a função de atenuação de potência em dB pode ser expressa na forma:

$$A(\Omega) = 10 \log \left[1 + \epsilon^2 T_n^2(\Omega/\Omega_c)\right] \tag{2.35}$$

O valor de ϵ é obtido observando-se que $T_n(1) = 1$ e $A(\Omega) = A_{max}$ em $\Omega = \Omega_c$. Desta forma temos:

$$\epsilon = \sqrt{10^{0,1A_{max}} - 1} \tag{2.36}$$

que é a mesma expressão obtida para o filtro de Butterworth.

Para $\Omega > \Omega_c$, $\cos^{-1}(\Omega/\Omega_c)$ se torna imaginário. Portanto, a expressão (2.31) pode ser escrita como:

$$T_n(\frac{\Omega}{\Omega_c}) = \cosh(n \cosh^{-1} \frac{\Omega}{\Omega_c}) \quad , \quad \Omega > \Omega_c \tag{2.37}$$

que leva a:

$$A(\Omega) = 10 \log \left[1 + \epsilon^2 \cosh^2(n \cosh^{-1} \frac{\Omega}{\Omega_c})\right] \quad , \quad \Omega > \Omega_c \tag{2.38}$$

Na frequência $\Omega = \Omega_s$ temos a atenuação mínima da faixa de rejeição dada por:

$$A_{min} = 10 \log \left[1 + \epsilon^2 \cosh^2(n \cosh^{-1} \frac{\Omega_s}{\Omega_c})\right] \tag{2.39}$$

Substituindo (2.36) na expressão acima e isolando n, obtemos:

$$n = \frac{\cosh^{-1} \sqrt{(10^{0,1A_{min}} - 1)/(10^{0,1A_{max}} - 1)}}{\cosh^{-1}(\Omega_s/\Omega_c)} \tag{2.40}$$

A equação acima fornece o valor mínimo da ordem do filtro de Chebychev em função das especificações. Note que existe uma certa semelhança com a expressão (2.25) que fornece a ordem mínima do filtro de Butterworth.

Para se obter $H(s)$, lembremos que a atenuação é dada por:

$$A(\Omega) = [H(j\Omega)H(-j\Omega)]^{-1} = 1 + \epsilon^2 T_n^2(\frac{\Omega}{\Omega_c}) \tag{2.41}$$

Substituindo-se $j\Omega$ por s temos:

$$[H(s)H(-s)]^{-1} = 1 + \epsilon^2 T_n^2(\frac{s}{j\Omega_c}) \tag{2.42}$$

Os pólos de $H(s)H(-s)$ serão os zeros da equação:

$$1 + \epsilon^2 T_n^2(\frac{s}{j\Omega_c}) = 0 \tag{2.43}$$

ou

$$\cos(n \cos^{-1} \frac{s}{j\Omega_c}) = \pm \frac{j}{\epsilon} \tag{2.44}$$

Fazendo a substituição

$$n\cos^{-1}(s/j\Omega_c) = u + jv \tag{2.45}$$

na equação acima, obtemos após alguma munipulação:

$$\cos u \cosh v - j \sin u \sinh v = \pm j/\epsilon \tag{2.46}$$

e, igualando as partes real e imaginária de ambos os membros:

$$\cos u \cosh v = 0 \tag{2.47}$$

$$\sin u \sinh v = \pm 1/\epsilon \tag{2.48}$$

Na equação (2.47), $\cosh v$ é um valor maior ou igual a 1, concluindo-se então que:

$$\cos u = 0 \tag{2.49}$$

e portanto,

$$\sin u = \pm 1 \tag{2.50}$$

o que nos permite escrever (2.48) na seguinte forma:

$$\sinh v = \pm 1/\epsilon \tag{2.51}$$

ou

$$v = \pm \sinh^{-1} \frac{1}{\epsilon} \tag{2.52}$$

Retomemos agora a igualdade (2.45). Dividindo-se ambos os membros por n e isolando s no primeiro membro, obtemos:

$$s = j\Omega_c \cos\left(\frac{u}{n} + j\frac{v}{n}\right) \tag{2.53}$$

que ainda pode ser expressa na forma:

$$s = \Omega_c \left(\sin\frac{u}{n}\sinh\frac{v}{n} + j\cos\frac{u}{n}\cosh\frac{v}{n}\right) \tag{2.54}$$

Agora, usando (2.49) e (2.52) na equação acima, podemos expressar os pólos do filtro de Chebychev da seguinte forma:

$$s_{p_i} = \alpha_i + j\beta_i \qquad i = 1, 2, \cdots, n \tag{2.55}$$

onde:

$$\alpha_i = \pm\Omega_c \sin\left(\frac{2i-1}{n}\frac{\pi}{2}\right) \sinh\left(\frac{\sinh^{-1}(1/\epsilon)}{n}\right) \tag{2.56}$$

$$\beta_i = \Omega_c \cos\left(\frac{2i-1}{n}\frac{\pi}{2}\right) \cosh\left(\frac{\sinh^{-1}(1/\epsilon)}{n}\right) \tag{2.57}$$

Os pólos dados por esta expressão se distribuem ao longo de uma elipse no plano s. Já a função de transferência $H(s)$ será dada por:

$$H(s) = \frac{C_H}{\prod_{i=1}^{n}(s - s_{p_i})} \tag{2.58}$$

onde:

$$C_H = \frac{|s_{p_1}|^2 |s_{p_2}|^2 \cdots |s_{p_{n/2}}|^2}{\sqrt{1+\epsilon^2}} \quad , \quad \text{n par} \tag{2.59}$$

ou

$$C_H = \sigma |s_{p_1}|^2 |s_{p_2}|^2 \cdots |s_{p_{(n-1)/2}}|^2 \quad , \quad \text{n ímpar} \tag{2.60}$$

sendo σ o pólo real de $H(s)$.

2.3.4 Aproximação Elíptica

Introdução

O filtro elíptico, também chamado filtro de Cauer, se distingue dos anteriores por possuir resposta ondulante tanto na faixa de passagem como nas faixas de rejeição. Pode-se mostrar que, para faixas de passagem e rejeição planas e com simetria geométrica, este filtro é ótimo no sentido de que ele implementa uma determinada máscara com a menor complexidade (menor ordem) possível. O cálculo dos pólos e zeros é bem mais complexo que o dos filtros anteriores, envolvendo o cálculo de algumas funções elípticas e processos numéricos iterativos para manipulação de polinômios.

A metodologia de projeto não será diferente da usada nos filtros anteriores. Procura-se uma função que aproxime a resposta de atenuação do filtro. No caso do filtro de Chebychev, esta função era o polinômio de Chebychev. No caso do filtro elíptico, essa função será a Função Racional de Chebychev, que pode ser expressa como a razão entre dois polinômios cujas raízes são funções elípticas das especificações do filtro.

A função Racional de Chebychev

A função racional de Chebychev, que denotaremos por $R_n(x, L)$ e que usaremos para a aproximação do filtro elíptico, é uma função racional de x que possui as seguintes propriedades:

1. R_n é par (ímpar) se n for par (ímpar).
2. R_n tem todos os seus n zeros no intervalo $-1 < x < 1$ e todos os seus n pólos fora deste intervalo.

3. R_n oscila entre ± 1 no intervalo $-1 \leq x \leq 1$.

4. $R_n(1, L) = +1$.

5. $1/R_n$ oscila entre $\pm 1/L$ no intervalo $|x| < x_L$, onde x_L é o primeiro valor de x para o qual $R_n(x, L) = L$.

A função racional de Chebychev será utilizada para se aproximar a resposta em frequência do filtro elíptico, da mesma forma como anteriormente utilizamos os polinômios de Butterworth e Chebychev. Na verdade, pode-se mostrar [3] que a função de atenuação de um filtro elíptico é dada por :

$$A(\Omega) = 10 \log \left\{ 1 + \left[\epsilon R_n \left(\frac{\Omega}{\Omega_c}, L \right) \right]^2 \right\} \tag{2.61}$$

Os parâmetros n, ϵ e L podem ser obtidos a partir das especificações do filtro da seguinte forma:

$$\epsilon = \sqrt{10^{0,1Amax} - 1} \tag{2.62}$$

$$L = \sqrt{\frac{10^{0,1Amin} - 1}{10^{0,1Amax} - 1}} \tag{2.63}$$

$$n = \frac{\mathrm{K}(x_L^{-1})\mathrm{K}'(L^{-1})}{\mathrm{K}'(x_L^{-1})\mathrm{K}(L^{-1})} \tag{2.64}$$

Na expressão acima para o cálculo da ordem do filtro aparece a *integral elíptica completa de primeira espécie* $K(k)$ e a *integral elíptica completa complementar de primeira espécie* $K'(k)$. Essas funções são tratadas no Apêndice A.

Não se descreverá aqui o procedimento detalhado para se chegar a uma expressão polinomial de $R_n(x, L)$. Este procedimento pode ser encontrado em [3]. Basicamente, o que se faz é obter uma equação diferencial a partir das propriedades de $R_n(x, L)$ enumeradas anteriormente e procurar uma solução para esta equação, que é expressa em termos de funções elípticas. Seguindo-se este procedimento, pode-se mostrar que $R_n(x, L)$ pode ser expressa por:

- n ímpar:

$$R_n(x, L) = C_1 x \prod_{i=1}^{(n-1)/2} \frac{x^2 - f_i^2}{x^2 - [x_L/f_i]^2} \tag{2.65}$$

onde:

$$f_i = \mathrm{sn}\left[2i\mathrm{K}(x_L^{-1})/n, L^{-1} \right] \tag{2.66}$$

$$C_1 = \prod_{i=1}^{(n-1)/2} \frac{1 - (x_L/f_i)^2}{1 - f_i^2} \tag{2.67}$$

- n par:

$$R_n(x, L) = C_2 \prod_{i=1}^{n/2} \frac{x^2 - f_i^2}{x^2 - [x_L/f_i]^2} \tag{2.68}$$

onde:

$$f_i = \mathrm{sn}\left[(2i-1)\mathrm{K}(x_L^{-1})/n, L^{-1}\right] \tag{2.69}$$

$$C_2 = \prod_{i=1}^{n/2} \frac{1 - (x_L/f_i)^2}{1 - f_i^2} \tag{2.70}$$

A função *seno elíptico* $\mathrm{sn}(u,k)$ e a *integral elíptica completa de primeira espécie* $\mathrm{K}(k)$ podem ser calculadas através de séries numéricas altamente convergentes [5] conforme mostrado no Apêndice A, e a constante x_L é dada por:

$$x_L = \frac{\Omega_s}{\Omega_c} \tag{2.71}$$

A função de transferência $H(s)$

A nossa preocupação agora é como obter a *função de transferência* $H(s)$ a partir da função racional de Chebychev. A partir de (2.61) podemos relacionar $R_n(\Omega/\Omega_c, L)$ e $H(j\Omega)$ da seguinte forma:

$$|H(j\Omega)|^2 = \frac{1}{1 + [\epsilon R_n(\frac{\Omega}{\Omega_c}, L)]^2} \tag{2.72}$$

Generalizando para a variável s, temos:

$$H(s)H(-s) = \frac{1}{1 + \epsilon^2 R_n(\frac{s}{j\Omega_c}) R_n(\frac{-s}{j\Omega_c})} \tag{2.73}$$

Esta expressão pode ainda ser colocada na seguinte forma:

$$[H(s)H(-s)]^{-1} = \frac{e(s)e(-s)}{q(s)q(-s)} = 1 + \frac{f(s)f(-s)}{q(s)q(-s)} \tag{2.74}$$

Examinando a expressão acima vemos que os zeros de $H(s)$ são iguais aos pólos de $R_n(s)$, de forma que o numerador de $H(s)$ é igual ao denominador de $R_n(s)$. Desta forma, os zeros de $H(s)$ são dados por:

$$s_{z_i} = \pm j x_L / f_i \tag{2.75}$$

com f_i dado por (2.66) e (2.69).

Resta-nos então calcular os pólos de $H(s)$, o que pode ser feito seguindo-se o seguinte roteiro:

- obter $f(s)$ e $q(s)$ a partir das especificações
- obter $e(s)e(-s) = f(s)f(-s) + q(s)q(-s)$
- fatorar $e(s)e(-s)$ para obter os pólos
- escolher os pólos do semiplano esquerdo e obter $H(s)$

As seções seguintes tratarão do cálculo dos pólos de $H(s)$ segundo o roteiro acima. Porém, isto não poderá ser feito no domínio da variável s.

A variável transformada

No procedimento descrito acima para o cálculo de $H(s)$ a partir de $R_n(s)$, nota-se que é necessário realizar operações de soma, multiplicação e fatoração de polinômios. Como os pólos de $H(s)$ tendem a se aglutinar muito em torno da frequência de corte Ω_c, essas operações com polinômios normalmente resultam em imprecisão numérica inaceitável. Para fugir desta imprecisão numérica, normalmente se utiliza uma transformação numérica que mapeia a variável s para uma outra variável complexa, que na literatura infelizmente recebe o nome de *variável transformada Z*. Para não confundirmos com a variável z do domínio digital que vínhamos utilizando, mudaremos o nome da variável transformada, que passaremos a chamar de r. Esta transformação mapeia o intervalo $[0, j\Omega_c]$ no eixo imaginário da variável s para todo o eixo imaginário da variável r, de forma que ocorre uma separação dos pólos situados perto da frequência de corte, permitindo maior precisão de cálculo. A variável transformada r se relaciona à variável s pelas relações:

$$r^2 = 1 + \frac{\Omega_c^2}{s^2} \tag{2.76}$$

$$s^2 = \frac{\Omega_c^2}{r^2 - 1} \tag{2.77}$$

A idéia é aplicar esta transformação a $R_n(s)$, realizar as operações polinomiais como indicado anteriormente para encontrar $H(r)$, e aplicar a transformação inversa para finalmente se obter H(s). Como vimos anteriormente, $R_n(s)$ pode ser escrito como a razão entre dois polinômios:

$$R_n(s) = \frac{f(s)}{\epsilon q(s)} \tag{2.78}$$

onde:

$$f(s) = C_f s^{\Delta} \prod_{i=1}^{N_1} (s^2 + f_i^2) \tag{2.79}$$

$$q(s) = \prod_{i=1}^{N_1} (s^2 + q_i^2) \tag{2.80}$$

e

$$q_i = \frac{x_L}{f_i}$$

$$N_1 = \begin{cases} n/2 & \text{se } n \text{ for par} \\ (n-1)/2 & \text{se } n \text{ for impar} \end{cases}$$

$$\Delta = \begin{cases} 0 & \text{se } n \text{ for par} \\ 1 & \text{se } n \text{ for impar} \end{cases}$$

Assumindo temporariamente $\Omega_c = 1$ e aplicando (2.77), obtemos:

$$f(r) = C_F(r^2-1)^{-\Delta/2} \prod_{i=1}^{N_1} \left(\frac{1 - f_i^2 + f_i^2 r^2}{r^2 - 1} \right) \tag{2.81}$$

A expressão de $f(r)$ obtida acima é uma razão entre dois polinômios em r. Entretanto, para simplificar os cálculos a serem feitos, gostaríamos que a função transformada fosse um polinômio semelhante a $f(s)$. Pode-se conseguir isto se dividirmos $f(s)$ e $q(s)$ pelo fator s^n. Isto não altera $R_n(s)$, porém aplicando-se (2.77) a $F(s) = f(s)/s^n$ e $Q(s) = q(s)/s^n$ obtém-se:

$$F(r) = C_F \prod_{i=1}^{N_1} (r^2 - F_i^2) \tag{2.82}$$

onde

$$C_F = C_f \prod_{i=1}^{N_1} (\frac{f_i}{\Omega_c})^2 \tag{2.83}$$

$$F_i^2 = 1 - (\frac{\Omega_c}{f_i})^2 \tag{2.84}$$

$$Q(r) = C_Q(r^2-1)^{\Delta/2} \prod_{i=1}^{N_1} (r^2 - Q_i^2) \tag{2.85}$$

onde

$$C_Q = \frac{1}{\Omega_c^{\Delta}} \prod_{i=1}^{N_1} (\frac{\Omega_i}{\Omega_c})^2 \tag{2.86}$$

$$Q_i^2 = 1 - (\frac{\Omega_c}{\Omega_i})^2 \tag{2.87}$$

Assim, através de um pequeno artifício, obtém-se expressões polinomiais para $F(r)$ e $Q(r)$, que serão usadas no lugar de $f(s)$ e $q(s)$ nos cálculos a seguir.

Obtenção de $E(r)E^*(r)$

Seguindo-se o procedimento descrito na Seção 2.3.4, devemos agora calcular $e(s)e(-s)$, só que agora no domínio da variável transformada r. A partir de (2.74) podemos escrever:

$$e(s)e(-s) = f(s)f(-s) + q(s)q(-s) \tag{2.88}$$

A função $q(s)$ é sempre par e a função $f(s)$ é par se n for par, e ímpar se n for ímpar. Desta forma, pode-se reescrever a equação acima na seguinte forma:

$$e(s)e(-s) = (-1)^{\Delta} f^2(s) + q^2(s) \tag{2.89}$$

Dividindo ambos os membros dessa equação por s^{2n} temos:

$$\frac{e(s)e(-s)}{s^{2n}} = (-1)^{\Delta}\frac{f^2(s)}{s^{2n}} + \frac{q^2(s)}{s^{2n}} \tag{2.90}$$

Chamando de $E(r)$ a transformada de $e(s)/s^n$, e de $E^*(r)$ a transformada de $e(-s)/s^n$, e aplicando a transformação dada por (2.77), temos:

$$E(r)E^*(r) = (-1)^{\Delta} F^2(r) + Q^2(r) \tag{2.91}$$

Vejamos agora como se calcula a expressão acima numericamente. Para simplificar um pouco os cálculos daqui para frente, vamos fazer a substituição $y = r^2$ nas expressões de $F(r)$ e $Q(r)$, e depois colocá-las na forma de produto de monômios em y. Fazendo isto, obtemos:

$$F^2(y) = C_F^2 \prod_{i=0}^{m-1} (y + t_i) \tag{2.92}$$

$$Q^2(y) = C_Q^2 (y-1)^{\Delta} \prod_{i=0}^{m-1} (y + t_i) \tag{2.93}$$

onde:

$$m = \begin{cases} n & \text{se } n \text{ for par} \\ n-1 & \text{se } n \text{ for ímpar} \end{cases}$$

$$t_i = t_{m-i-1} = F_i^2 \text{ ou } Q_i^2 \quad , \quad i = 0, 1, \cdots, N_1 - 1$$

Observe que fizemos uma renumeração dos termos F_i e Q_i, fazendo i variar de 0 a $N_1 - 1$ em vez de 1 a N_1. Para fazer a soma indicada por (2.91) devemos desenvolver o produto acima. Para tanto, usaremos um algoritmo numérico que faz o produto de m monômios como indicado a seguir. Sejam $e_0, e_1, \cdots, e_{m-1}$ os coeficientes do polinômio produto. Inicialmente faz-se $e_0 = t_0$ e $e_1 = 1$. Depois aplica-se as seguintes relações de recorrência:

$$\left.\begin{array}{ll} A_0 = t_j e_0 & \\ A_i = e_{i-1} + t_j e_i & i = 1, 2, \cdots, j \\ \text{fazendo-se depois:} & \\ e_i = A_i & i = 0, 1, \cdots, j \\ e_{j+1} = 1 & \end{array}\right\} \quad j = 1, 2, \cdots, m-1$$

Como resultado do algoritmo acima, obtem-se os polinômios $F^2(y)$ e $Q^2(y)$ no seguinte formato:

$$F^2(y) = C_F^2(y^m + e_{f_{m-1}} y^{m-1} + \cdots + e_{f_2} y^2 + e_{f_1} y + e_{f_0}) \tag{2.94}$$

$$Q^2(y) = C_Q^2(y^n + e_{q_{n-1}} y^{n-1} + \cdots + e_{q_2} y^2 + e_{q_1} y + e_{q_0}) \tag{2.95}$$

Finalmente, efetuando-se a operação indicada por (2.91), obtém-se:

$$E(y)E^*(y) = C_E^2(y^n + e_{n-1} y^{n-1} + \cdots + e_2 y^2 + e_1 y + e_0) \tag{2.96}$$

onde

$$C_E^2 = \begin{cases} C_Q^2 + C_F^2 & \text{se } n \text{ for par} \\ C_Q^2 & \text{se } n \text{ for ímpar} \end{cases} \tag{2.97}$$

Fatoração de $E(r)E^*(r)$

Uma vez obtidos os coeficientes do polinômio $E(r)E^*(r)$, deve-se fatorá-lo em termos de segunda ordem antes de aplicar a transformação $r \rightarrow s$. A fatoração pode ser feita pelo método de Bairstow, que apresentaremos de forma resumida a seguir.

Seja $F(y)$ um polinômio em y de grau n:

$$F(y) = y^n + e_{n-1} y^{n-1} + \cdots + e_2 y^2 + e_1 y + e_0 \tag{2.98}$$

Este polinômio pode ser escrito como o produto de um termo de segunda ordem e um polinômio de grau $n-2$. Desenvolvendo este produto e igualando a zero para se obter as raízes, temos:

$$\begin{aligned} &(y^{n-2} + h_{n-1} y^{n-3} + \cdots + h_4 y^2 + h_3 y + h_2)(y^2 + p_m y + q_m) = \\ &\quad y^n + (p_m + h_{n-1}) y^{n-1} + (q_m + h_{n-1} p_m + h_{n-2}) y^{n-2} + \cdots \\ &\quad + (h_4 q_m + h_3 p_m + h_2) y^2 + (h_3 q_m + h_2 p_m) y + h_2 q_m = 0 \end{aligned} \tag{2.99}$$

Comparando os coeficientes de (2.98) e (2.99), podemos montar o seguinte sistema de equações:

$$\begin{aligned}
e_0 &= h_2 q_m + h_1 p_m + h_0 \\
e_1 &= h_3 q_m + h_2 p_m + h_1 \\
&\vdots \\
e_r &= h_{r+2} q_m + h_{r+1} p_m + h_r \\
&\vdots \\
e_{n-2} &= h_n q_m + h_{n-1} p_m + h_{n-2} \\
e_{n-1} &= h_{n+1} q_m + h_n p_m + h_{n-1}
\end{aligned}$$

onde:

$$\begin{aligned}
h_n &= 1 \\
h_{n+1} &= 0
\end{aligned}$$

Os termos h_0 e h_1 que foram incluídos nas equações acima são essenciais ao método de Bairstow. Para ver como isto acontece, reescrevamos as equações para fornecer os valores de h:

$$\begin{aligned}
h_0 &= e_0 - h_1 p_m - h_2 q_m \\
h_1 &= e_1 - h_2 p_m - h_3 q_m \\
&\vdots \\
h_r &= e_r - h_{r+1} p_m - h_{r+2} q_m \\
&\vdots \\
h_{n-2} &= e_{n-2} - h_{n-1} p_m - h_n q_m \\
h_{n-1} &= e_{n-1} - h_n p_m - h_{n+1} q_m
\end{aligned}$$

Os valores de h acima são calculados de baixo para cima, começando por h_{n-1}, passando para h_{n-2} e assim por diante até h_0. Observando a equação (2.99), vemos que se p_m e q_m são raízes de $F(y)$, então h_1 e h_0 serão necessariamente iguais a zero. Pelo método de Bairstow, escolhemos inicialmente valores arbitrários de p_m e q_m, para os quais h_0 e h_1 não serão nulos. Então desenvolvemos h_0 e h_1 em série de Taylor, considerando somente os termos de primeira ordem, e impomos que eles sejam nulos. Desta forma, obtemos o seguinte sistema:

$$h_0 + \frac{\partial h_0}{\partial p_m}\Delta p + \frac{\partial h_0}{\partial q_m}\Delta q = 0 \qquad (2.100)$$

$$h_1 + \frac{\partial h_1}{\partial p_m}\Delta p + \frac{\partial h_1}{\partial q_m}\Delta q = 0 \tag{2.101}$$

Para resolver este sistema, devemos encontrar os valores de h_0 e h_1 correspondentes aos valores iniciais de p_m e q_m, bem como as derivadas parciais de h_0 e h_1 em relação a p_m e q_m. Os valores de h_0 e h_1 são obtidos do sistema de equações mostrado acima. Já as derivadas parciais de h em relação a p_m são dadas por:

$$\begin{aligned}
\frac{\partial h_0}{\partial p_m} &= -p_m\frac{\partial h_1}{\partial p_m} - h_1 - q_m\frac{\partial h_2}{\partial p_m} = c_0 \\
\frac{\partial h_1}{\partial p_m} &= -p_m\frac{\partial h_2}{\partial p_m} - h_2 - q_m\frac{\partial h_3}{\partial p_m} = c_1 \\
&\vdots \\
\frac{\partial h_{n-3}}{\partial p_m} &= -p_m\frac{\partial h_{n-2}}{\partial p_m} - h_{n-2} - q_m\frac{\partial h_{n-1}}{\partial p_m} = c_{n-3} \\
\frac{\partial h_{n-2}}{\partial p_m} &= -p_m\frac{\partial h_{n-1}}{\partial p_m} - h_{n-1} = c_{n-2} \\
\frac{\partial h_{n-1}}{\partial p_m} &= -1 = c_{n-1}
\end{aligned}$$

Utilizando-se c_{n-1} calcula-se c_{n-2}. A partir de c_{n-2} e c_{n-1} calcula-se c_{n-3}, a assim por diante até chegar a c_1 e c_0.

Considerando agora as derivadas de h com relação a q_m, temos:

$$\begin{aligned}
\frac{\partial h_0}{\partial q_m} &= -p_m\frac{\partial h_1}{\partial q_m} - h_2 - q_m\frac{\partial h_2}{\partial q_m} \\
\frac{\partial h_1}{\partial q_m} &= -p_m\frac{\partial h_2}{\partial q_m} - h_3 - q_m\frac{\partial h_3}{\partial q_m} \\
&\vdots \\
\frac{\partial h_{n-3}}{\partial q_m} &= -p_m\frac{\partial h_{n-2}}{\partial q_m} - h_{n-1} \\
\frac{\partial h_{n-2}}{\partial q_m} &= -p_m\frac{\partial h_{n-1}}{\partial q_m} - 1 \\
\frac{\partial h_{n-1}}{\partial q_m} &= 0
\end{aligned}$$

Substituindo-se $\partial h_{n-1}/\partial q_m = 0$ da última equação na equação de cima, concluimos que $\partial h_{n-2}/\partial q_m = -1 = c_{n-1}$. Substituindo-se esse valor na equação de cima, e assim sucessivamente, pode-se mostrar que:

$$\frac{\partial h_0}{\partial q_m} = c_1 \tag{2.102}$$

$$\frac{\partial h_1}{\partial q_m} = c_2 \tag{2.103}$$

Desta forma, (2.100) e (2.101) podem ser reescritas na seguinte forma:

$$c_0\Delta p + c_1\Delta q = -h_0 \tag{2.104}$$
$$c_1\Delta p + c_2\Delta q = -h_1 \tag{2.105}$$

cuja solução é:

$$\Delta p = \frac{h_1c_1 - h_0c_2}{c_0c_2 - c_1^2} \tag{2.106}$$
$$\Delta q = \frac{h_0c_1 - h_1c_0}{c_0c_2 - c_1^2} \tag{2.107}$$

Somando-se Δp e Δq a p e q, obtem-se novos valores que devem ser usados na próxima iteração. O procedimento descrito acima deve ser repetido até que h_1 e h_0 sejam iguais a um certo valor de tolerância proximo de zero.

Uma vez que um termo de segunda ordem seja assim obtido, deve-se obter o polinômio reduzido através do cálculo de $h_0, h_1, \cdots, h_{n-2}$, e aplicar novamente o método até que se esgotem todos os termos de segunda ordem. Se n for ímpar, no final do processo teremos um termo de primeira ordem sobrando, que chamaremos de s_0. Desta forma, como resultado da fatoração, o polinômio $E(y)E^*(y)$ será escrito na forma:

$$E(y)E^*(y) = C_E^2(y + s_0)^\Delta \prod_{i=1}^{N_1}(y^2 + p_iy + q_i) \tag{2.108}$$

onde:

$$N_1 = \begin{cases} n/2 & \text{se } n \text{ for par} \\ (n-1)/2 & \text{se } n \text{ for ímpar} \end{cases}$$

A transformação $r \rightarrow s$

Vejamos agora como se calcula $e(s)$ a partir do polinômio $E(r)E^*(r)$ expresso segundo (2.108). Assumindo $\Omega_c = 1$, aplicando (2.76) em (2.108) e lembrando que $E(r)E^*(r)$ é a transformada de $e(s)e(-s)/s^{2n}$, temos:

$$\frac{e(s)e(-s)}{s^{2n}} = C_E^2(1 + \frac{1}{s^2} + s_0)^\Delta \prod_{i=1}^{N_1}\left[(1 + \frac{1}{s^2})^2 + (1 + \frac{1}{s^2})p_i + q_i\right] \tag{2.109}$$

Desenvolvendo esta expressão e escolhendo os termos que se encontram no semiplano esquerdo do plano s, chegamos a:

$$e(s) = C_e(s+\sigma)^{\Delta} \prod_{i=1}^{N_1} (s^2 + a_i s + b_i) \tag{2.110}$$

onde:

$$b_i = \frac{1}{\sqrt{1+p_i+q_i}} \tag{2.111}$$

$$a_i = \sqrt{2b_i - \frac{2+p_i}{1+p_i+q_i}} \tag{2.112}$$

$$\sigma = \sqrt{-\frac{1}{1+s_0}} \tag{2.113}$$

$$C_e = \frac{C_E}{\sigma^{\Delta}} \prod_{i=1}^{m} \sqrt{1+p_i+q_i} \tag{2.114}$$

Isto completa o cálculo da função de transferência $H(s)$, que será dada por:

$$H(s) = C_e^{-1} \frac{\prod_{i=1}^{N_1}(s^2 + x_L^2/f_i^2)}{(s+\sigma)^{\Delta} \prod_{i=1}^{N_1}(s^2 + a_i s + b_i)} \tag{2.115}$$

2.3.5 Transformação em Frequência

Passa-Baixa para Passa-Alta

Sejam S e s os argumentos das funções de transferência dos filtros passa-baixa e passa-alta, respectivamente. A transformação passa-baixa para passa-alta se dá através da seguinte relação:

$$S = \frac{\Omega_{ch}}{s} \tag{2.116}$$

onde Ω_{ch} é a frequência limite da faixa de passagem do filtro passa-alta.

Dadas as especificações de um filtro passa-alta (A_{max}, A_{min}, Ω_{ch} e Ω_{sh}), primeiramente devemos obter as especificações do filtro protótipo passa-baixa equivalente. Por simplicidade, faremos a frequência limite da faixa de passagem do protótipo passa-baixa sempre igual a 1, restando-nos apenas encontrar a frequência de início da faixa de rejeição Ω_{sl}. Isto pode ser feito facilmente através de (2.116), que nos fornece:

$$\Omega_{sl} = \frac{\Omega_{ch}}{\Omega_{sh}} \tag{2.117}$$

Uma vez obtidos os pólos e zeros do protótipo passa-baixa, os pólos do filtro passa-alta serão dados por:

$$s_i = \frac{\Omega_{ch}}{S_i} \tag{2.118}$$

onde s_i e S_i são pólos ou zeros dos filtros passa-alta e passa-baixa, respectivamente. Observe que pólos ou zeros no infinito se transformarão em pólos ou zeros na origem. No caso dos filtros de Butterworth e Chebychev, por exemplo, que possuem n zeros no infinito, aparecerão n zeros na origem.

Ao se transformar um pólo ou um zero, deve-se levar em conta o efeito desta transformação sobre o fator de ganho constante da função de transferência. Consideremos um termo que contenha um pólo e um zero reais da forma:

$$\frac{S+a}{S+b}$$

Aplicando (2.116) ao termo acima obtemos:

$$\frac{a}{b}\left(\frac{s+\Omega_{ch}/a}{s+\Omega_{ch}/b}\right)$$

Como se conclui do resultado acima, ao se transformar um pólo real deve-se dividir o ganho pelo módulo do pólo, e ao se transformar um zero real deve-se multiplicar o ganho pelo módulo do zero. Generalizando para pólos e zeros complexos, ao se transformar um par conjugado de pólos complexos deve-se dividir o ganho pelo módulo ao quadrado do pólo, e ao se transformar um par conjugado de zeros complexos deve-se multiplicar o ganho pelo módulo ao quadrado do zero.

Passa-Baixa para Passa-Faixa

Como no caso anterior, continuaremos chamando de S o argumento da função de transferência referente ao filtro passa-baixa, e de s, o argumento da função de transferência do filtro passa-faixa. A transformação passa-baixa para passa-faixa é dada pela relação:

$$S = \frac{1}{\gamma}\left(\frac{s}{\Omega_0} + \frac{\Omega_0}{s}\right) \tag{2.119}$$

onde:

$$\gamma = \frac{\Omega_{c2} - \Omega_{c1}}{\Omega_0} \tag{2.120}$$

e

$$\Omega_0 = \sqrt{\Omega_{c1}\Omega_{c2}} \tag{2.121}$$

é a frequência central da faixa de passagem do filtro. A resposta em frequência obtida através desta transformação possui simetria geométrica em torno da frequência central Ω_0. Pode-se mostrar [3] que as frequências limites

da faixa de rejeição Ω_{s1} e Ω_{s2} também são geometricamente simétricas em relação a Ω_0, ou seja, $\Omega_{s1}\Omega_{s2} = \Omega_0^2$.

Vejamos agora como encontrar as especificações do protótipo passa-baixa a partir das especificações do filtro passa-faixa. Para tanto, vamos mostrar que a razão entre duas frequências no protótipo passa-baixa corresponde à razão entre larguras de faixa no filtro passa-faixa. Sejam, por exemplo, θ_b e θ_c duas frequências do protótipo passa-baixa. Aplicando-se a transformação passa-baixa para passa-faixa à razão θ_b/θ_c obtém-se:

$$\frac{\theta_b}{\theta_c} = \frac{1/\gamma\left(\Omega_b/\Omega_0 - \Omega_0/\Omega_b\right)}{1/\gamma\left(\Omega_c/\Omega_0 - \Omega_0/\Omega_c\right)} = \frac{\Omega_b - \frac{\Omega_0^2}{\Omega_b}}{\Omega_c - \frac{\Omega_0^2}{\Omega_c}}$$

Ora, Ω_b e Ω_0^2/Ω_b são duas frequências dispostas em simetria geométrica em torno de Ω_0. O mesmo acontece com Ω_c e Ω_0^2/Ω_c. Desta forma, depois da transformação, a razão entre duas frequências foi convertida na razão entre duas larguras de faixa onde as frequências extremas apresentam simetria geométrica em torno de Ω_0. Se fizermos agora $\theta_c = \Omega_{cl} = 1$ (que se mapeará para o par Ω_{c1}, Ω_{c2}) e $\theta_b = \Omega_{sl}$ (que se mapeará para o par Ω_{s1}, Ω_{s2}), obteremos a frequência limite da faixa de rejeição do protótipo passa-baixa a partir das especificações do filtro passa-faixa:

$$\Omega_{sl} = \frac{\Omega_{s2} - \Omega_{s1}}{\Omega_{c2} - \Omega_{c1}} \tag{2.122}$$

A expressão acima pressupõe que as frequências limites das faixas de rejeição Ω_{s1} e Ω_{s2} possuam simetria geométrica em torno da frequência central Ω_0. Isto geralmente não é verdade, pois ao se especificar o filtro, não é desejável que se defronte com tais restrições de simetria. O que podemos fazer é calcular a frequência central como a média geométrica das frequências limites da faixa de passagem, e obter Ω_{sl} através de:

$$\Omega_{sl} = \min\left(\frac{\Omega_{s2} - \Omega_0^2/\Omega_{s2}}{\Omega_{c2} - \Omega_{c1}}\,,\ \frac{\Omega_0^2/\Omega_{s1} - \Omega_{s1}}{\Omega_{c2} - \Omega_{c1}}\right) \tag{2.123}$$

onde se calcula o pior caso entre as faixas de rejeição geometricamente simétricas obtidas a partir de Ω_{s1} e Ω_{s2}.

Os pólos e zeros do filtro passa-faixa podem ser obtidos a partir de (2.119), que pode ser colocada na forma:

$$s_i = \Omega_0\left[\frac{\gamma}{2}S_i \pm j\sqrt{1 - \left(\frac{\gamma S_i}{2}\right)^2}\right] \tag{2.124}$$

O par de pólos $S_i = -R \pm j\theta$ $(R > 0)$ do protótipo passa-baixa gera dois pares de pólos do filtro passa-faixa:

$$s_i = \frac{\Omega_{c2} - \Omega_{c1}}{2} \left\{ -R + \left[\frac{\sqrt{A^2 + (2R\theta)^2} - A}{2} \right]^{1/2} \right.$$
$$\left. \pm j \left[\theta - \left(\frac{\sqrt{A^2 + (2R\theta)^2} + A}{2} \right)^{1/2} \right] \right\} \qquad (2.125)$$

e

$$s_i = \frac{\Omega_{c2} - \Omega_{c1}}{2} \left\{ -R - \left[\frac{\sqrt{A^2 + (2R\theta)^2} - A}{2} \right]^{1/2} \right.$$
$$\left. \pm j \left[\theta + \left(\frac{\sqrt{A^2 + (2R\theta)^2} + A}{2} \right)^{1/2} \right] \right\} \qquad (2.126)$$

onde

$$A = \theta^2 + \frac{4\Omega_{c1}\Omega_{c2}}{(\Omega_{c2} - \Omega_{c1})^2} - R^2 \qquad (2.127)$$

Ao transformar um pólo segundo as expressões acima, deve-se considerar o efeito sobre o fator de ganho constante. Seja, por exemplo, um zero real do protótipo passa-baixa situado no ponto a. O termo da função de transferência associado a esse zero é $S + a$. Aplicando-se a transformação dada por (2.119) a esse termo, temos:

$$S + a \rightarrow \frac{1}{\gamma} \left(\frac{s}{\Omega_0} + \frac{\Omega_0}{s} \right) + a = \frac{1}{\Omega_{c2} - \Omega_{c1}} \frac{s^2 + a(\Omega_{s2} - \Omega_{s1}) + \Omega_0^2}{s}$$

Como se nota, aparece o termo $1/(\Omega_{c2} - \Omega_{c1})$ como fator de ganho constante. Portanto, quando se aplica a transformação num zero real, deve-se dividir o ganho pela largura de faixa do filtro. Analogamente, quando se aplica a transformação a um pólo, deve-se multiplicar o ganho pela largura de faixa. Para pares de pólos e zeros complexos, deve-se multiplicar ou dividir o ganho pela largura de faixa ao quadrado.

Passa-Baixa para Corta-Faixa

Essa transformação é o inverso da transformação passa-baixa para passa-faixa, ou seja:

$$S = \frac{\gamma}{s/\Omega_0 + \Omega_0/s} \qquad (2.128)$$

Analogamente ao caso anterior, também aqui a razão entre duas frequências do filtro passa-baixa corresponde à razão entre larguras de faixa do filtro

corta-faixa. A diferença é que a frequência central Ω_0 é calculada em função da faixa de rejeição ao invés da faixa de passagem, ou seja:

$$\Omega_0 = \sqrt{\Omega_{s1}\Omega_{s2}} \tag{2.129}$$

Desta forma, a frequência limite da faixa de rejeição do filtro protótipo passa-baixa equivalente é dada por:

$$\Omega_{sl} = \frac{B}{\Omega_{s2} - \Omega_{s1}} \tag{2.130}$$

onde:

$$B = \min\left(\Omega_{c2} - \frac{\Omega_0^2}{\Omega_{c2}}\,,\ \frac{\Omega_0^2}{\Omega_{c1}} - \Omega_{c1}\right) \tag{2.131}$$

Os pólos e zeros do filtro corta-faixa são dados por:

$$s_i = \Omega_0\left[\frac{\gamma}{2S_i} \pm j\sqrt{1 - \left(\frac{\gamma}{2S_i}\right)^2}\right] \tag{2.132}$$

Comparando (2.124) e (2.132), vemos que podemos utilizar as expressões (2.125) e (2.126) para o filtro corta-faixa simplesmente invertendo-se os pólos e zeros do protótipo passa-baixa antes de entrar nas expressões. Note que zeros no infinito são transformados em zeros em $\pm j\Omega_0$.

Para se levar em conta o efeito da transformação dos pólos e zeros sobre o fator de ganho constante, observe que essa transformação corresponde a uma transformação para passa-alta seguida de uma transformação para passa-faixa. Logo, o fator de ganho constante deve ser multiplicado por $a/(\Omega_{c2} - \Omega_{c1})$ quando se transforma um zero real a, ou por $(\Omega_{c2} - \Omega_{c1})/b$ quando se transforma um pólo real b. Generalizando para pólos e zeros complexos, quando se transforma um par conjugado de pólos de módulo $|s_p|$, deve-se multiplicar o ganho por

$$\frac{(\Omega_{c2} - \Omega_{c1})^2}{|s_p|^2}$$

e quando se transforma um par conjugado de zeros de módulo $|s_z|$, deve-se multiplicar o ganho por

$$\frac{|s_z|^2}{(\Omega_{c2} - \Omega_{c1})^2}$$

Capítulo 3

Efeitos de precisão finita dos registros

3.1 Introdução

Depois do cálculo de sua função de transferência no plano z, um filtro digital pode ser implementado em uma das seguintes maneiras: como um programa num computador de uso geral, como um programa num processador especializado em processamento digital de sinais, ou num "hardware" dedicado composto de somadores, multiplicadores e atrasadores. Em qualquer uma destas implementações os dados de entrada, os coeficientes do filtro e os resultados das operações aritméticas estão restritos ao número finito de bits dos registros da máquina, fazendo com que o desempenho do filtro seja degradado em relação ao projeto original. Esta degradação pode se manifestar basicamente de três maneiras diferentes: como um ruído na saída do filtro, como um desvio da resposta em frequência teórica, ou como instabilidades e oscilações no caso de filtros de resposta impulsiva infinita.

Vale salientar que hoje em dia este problema já não é tão sério quanto foi há uns quinze anos atrás, quando se utilizavam microcomputadores de 8 bits para a implementação de filtros digitais para aplicações em tempo real. Hoje, processadores de 16 bits já se tornaram padrão, e processadores de 32 bits com aritmética de ponto flutuante já se oferecem como uma opção atraente [6]. Entretanto, quando se implementam sistemas que utilizam algum processamento digital de sinais em circuitos integrados dedicados, não é desejável despender mais "hardware" do que o estritamente necessário. O que se deseja nesses casos é colocar a maior parte possível do sistema num único circuito integrado, ocupando a menor área de silício possível. Desta forma, os filtros digitais devem ser implementados com o menor comprimento possível dos registros, dando-se preferência à aritmética de ponto fixo pela sua simplicidade.

Os efeitos da precisão finita dos registros nos filtros digitais começou a ser estudada na década de 60. Um artigo tutorial de Oppenheim e Weinstein [8] publicado em 1972 já apontava nada menos que 71 referências. Até esta época, enfatizava-se a análise de filtros implementados nas formas direta, em cascata e em paralelo, obtendo-se como resultado a variância (ou a norma) do ruído na saída expressa através de integrais de contorno de funções de transferência. A partir de meados da década de 70, os trabalhos passaram a enfatizar a formulação de variáveis de estado para a análise do ruído de quantização e para a procura de estruturas que minimizam a geração do ruído. Neste e no capítulo seguinte procuraremos fazer um apanhado geral destas duas formas de se atacar o problema, para se formar um quadro de como as pesquisas evoluíram e que resultados obtiveram.

Para efeito de estudo, os efeitos do número finito de bits dos registros podem ser divididos em 3 categorias:

- quantização do sinal de entrada no conversor analógico-digital;
- quantização dos coeficientes do filtro;
- quantização dos resultados das multiplicações e, no caso de aritmética de ponto flutuante, das adições.

A quantização do sinal de entrada é um problema inerente a todos os sistemas digitais que processam sinais de natureza analógica. Este é um problema importante, analisado extensamente na literatura [2,9,10,11,22], porém essencialmente externo ao filtro digital. Por este motivo, estudaremos apenas os dois últimos ítens da relação acima, começando pela quantização dos coeficientes.

3.2 Quantização dos coeficientes

3.2.1 Análise do erro

A análise dos efeitos da quantização dos coeficientes sobre o desempenho do filtro digital é determinística e exata. De posse dos coeficientes quantizados, podemos simplesmente traçar a resposta em frequência do filtro e verificar se o desvio da resposta ideal é aceitável, ou seja, se ele ainda obedece à máscara para o qual foi especificado.

Uma forma alternativa de análise foi proposta por Knowles e Olcayto [12], onde se modela a função de transferência quantizada como a associação em paralelo da função de transferência ideal $H(z)$ e de uma função de transferência "fantasma" $G(z)$ conforme mostrado na Fig. 3.1.

Seja $H(z)$ da forma:

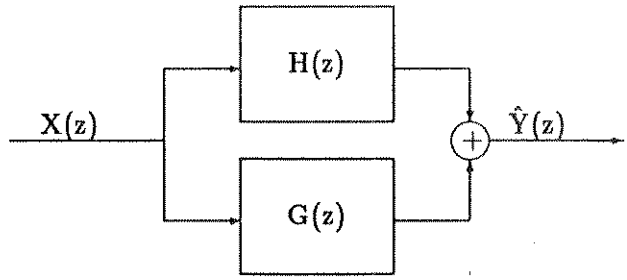

Figura 3.1: Modelamento da função de transferência do filtro quantizado.

$$H(z) = \frac{\sum_{k=0}^{N} a_k z^{-k}}{1 + \sum_{k=1}^{N} b_k z^{-k}} \tag{3.1}$$

A função de transferência quantizada será:

$$\hat{H}(z) = \frac{\sum_{k=0}^{N} \hat{a}_k z^{-k}}{1 + \sum_{k=1}^{N} \hat{b}_k z^{-k}} \tag{3.2}$$

onde

$$\hat{a}_k = a_k + \alpha_k \tag{3.3}$$

$$\hat{b}_k = b_k + \beta_k \tag{3.4}$$

sendo α_k e β_k os erros de quantização dos coeficientes.

As equações a diferença correspondentes a (3.1) e (3.2) são:

$$y(n) = \sum_{k=0}^{N} a_k x(n-k) - \sum_{k=1}^{N} b_k y(n-k) \tag{3.5}$$

$$\hat{y}(n) = \sum_{k=0}^{N} \hat{a}_k x(n-k) - \sum_{k=1}^{N} \hat{b}_k \hat{y}(n-k) \tag{3.6}$$

O erro na saída do filtro será definido por:

$$e(n) = \hat{y}(n) - y(n) \tag{3.7}$$

Substituindo (3.5) e (3.6) em (3.7) e desprezando-se os termos de segunda ordem temos:

$$e(n) = \sum_{k=0}^{N} \alpha_k x(n-k) - \sum_{k=1}^{N} b_k e(n-k) - \sum_{k=1}^{N} \beta_k y(n-k) \tag{3.8}$$

Achando a transformada z de ambos os membros e resolvendo para $E(z)$ chegamos a:

$$E(z) = \frac{\alpha(z) - \beta(z)H(z)}{B(z)} X(z) \tag{3.9}$$

onde $\alpha(z)$, $\beta(z)$ e $B(z)$ são definidos por

$$\alpha(z) = \sum_{k=0}^{N} \alpha_k z^{-k}$$

$$\beta(z) = \sum_{k=1}^{N} \beta_k z^{-k}$$

$$B(z) = 1 + \sum_{k=1}^{N} b_k z^{-k}$$

De (3.7) e (3.9), concluímos que:

$$\hat{Y}(z) = \left[H(z) + \frac{\alpha(z) - \beta(z)H(z)}{B(z)} \right] X(z) \tag{3.10}$$

donde se tira que $G(z)$ na Fig. 3.1 vale:

$$G(z) = \frac{\alpha(z) - \beta(z)H(z)}{B(z)} \tag{3.11}$$

Fazendo um modelo estatístico para o erro e para os coeficientes, Knowles e Olcayto utilizam a expressão de $G(z)$ acima para se obter a variância do erro $e(n)$ na saída do filtro:

$$\sigma^2 = \frac{1}{2\pi j} \frac{\Delta^2}{12} \left\{ \mu \oint_C \frac{1}{B(z)B(z^{-1})} z^{-1} dz + \nu \oint_C \frac{A(z)A(z^{-1})}{B(z)B(z^{-1})} z^{-1} dz \right\} \tag{3.12}$$

onde Δ é o intervalo de quantização, $A(z)$ e $B(z)$ são as transformadas z de a_k e b_k, μ e ν são os números de coeficientes não nulos e não unitários do numerador e do denominador respectivamente. A expressão acima pode ser útil quando se deseja um parâmetro de comparação entre diferentes formas de implementação para se verificar qual delas se desvia menos da implementação ideal.

3.2.2 Estabilidade

A instabilidade é um problema que pode ocorrer em filtros IIR. Se o número de bits empregado para representar os coeficientes for pequeno, um ou mais pólos podem eventualmente cair fora da C.R.U., fazendo com que o filtro se torne instável. Esta situação pode ocorrer mesmo que o número de bits não seja pequeno desde que os pólos estejam muito aglutinados. Para mostrar que isto é verdade, reescrevamos o denominador de (3.1) na forma fatorada:

$$B(z) = 1 + \sum_{k=1}^{N} b_k z^{-k} = \prod_{i=1}^{N} \left(1 - p_i z^{-1}\right) \tag{3.13}$$

Partindo-se desta expressão, a mudança na localização dos pólos em função de uma pequena mudança nos coeficientes pode ser expressa na forma:

$$dp_i = \sum_{k=1}^{N} \left.\frac{\partial p_i}{\partial b_k}\right|_{z=p_i} db_k \tag{3.14}$$

Calculando-se as derivadas parciais através de:

$$\frac{\partial p_i}{\partial b_k} = \frac{\partial B/\partial b_k}{\partial B/\partial p_i} \tag{3.15}$$

chega-se a:

$$dp_i = -\sum_{k=1}^{N} \frac{p_i^{-k+1} db_k}{\prod_{\substack{j=1 \\ j\neq i}}^{N} \left(1 - p_j p_i^{-1}\right)} \tag{3.16}$$

Se os pólos estiverem muito próximos entre si, os termos $(1-p_j p_i^{-1})$ serão muito pequenos. Portanto, quanto mais aglutinados estiverem os pólos, menor será a mudança em b_k necessária para gerar problemas de instabilidade. Esta aglutinação acontece quando a seletividade do filtro for alta, o que ocorre nos casos onde a ordem do filtro ou a frequência de amostragem[1], ou ambos, forem elevados. Kaiser [14] mostrou que se dobrarmos a ordem do filtro, necessitaremos do dobro de dígitos decimais para representação dos coeficientes. De forma similar, dobrando-se a frequência de amostragem, torna-se necessário aproximadamente $0{,}3N$ dígitos adicionais para a representação dos coeficientes. Ainda segundo Kaiser, um filtro passa-baixa de ordem N – obtido a partir de um filtro no plano s com pólos p_k $(k = 1, \cdots, N)$ através de um mapeamento $s \rightarrow z$ com período de amostragem T – será estável se o número de dígitos decimais m_d utilizado para representar o coeficiente b_k satisfizer:

[1]Frequência de amostragem elevada implica em aproximação dos pólos em direção ao eixo real do plano z, com consequente aproximação dos pares de pólos complexos conjugados

$$m_d > 1 + \left\lceil -\log_{10}\left(\frac{5\sqrt{N}}{2^{N+2}}\prod_{k=1}^{N} p_k T\right)\right\rceil \tag{3.17}$$

onde $\lceil\ \rceil$ indica o menor inteiro que é maior ou igual ao argumento.

Para uma seção de segunda ordem na forma direta com função de transferência

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \tag{3.18}$$

pode-se mostrar [22] que os pólos estarão no interior da C.R.U. se e somente se:

$$b_2 < 1 \tag{3.19}$$
$$1 + b_1 + b_2 > 0 \tag{3.20}$$
$$1 - b_1 + b_2 > 0 \tag{3.21}$$

ou, escrito de outra forma:

$$b_2 < 1 \tag{3.22}$$
$$|b_1| - b_2 < 1 \tag{3.23}$$

É importante observar que o tipo de instabilidade a que nos referimos aqui se deve exclusivamente ao movimento dos pólos em direção ao exterior da C.R.U., o que é bem diferente das oscilações provocadas por "overflow" das operações aritméticas e das oscilações tipo ciclo-limite [2], as quais não serão analisadas aqui. Tendências de instabilidade devido à quantização dos coeficientes podem ser facilmente detectadas verificando-se a localização dos pólos quantizados e a presença de picos de ganhos elevados na curva de resposta em frequência.

3.3 Quantização das operações aritméticas

3.3.1 Introdução

Os resultados das operações de multiplicação (e no caso de aritmética de ponto flutuante, de adição) dentro de um filtro digital são normalmente truncados de forma a caber num registro com um número menor de bits. Ao erro gerado neste procedimento dá-se o nome de *ruído de quantização do sinal.* Este fenômeno é não linear quanto à amplitude do sinal e depende da estrutura de implementação. A dificuldade de tratamento matemático no caso geral leva à adoção de um modelo estatístico, onde normalmente

o erro é modelado através de uma variável aleatória com função densidade de probabilidade constante [2], adicionada depois da operação causadora do erro. As características desta variável aleatória dependem do tipo de representação numérica utilizada: sinal e magnitude, complemento de um ou complemento de dois. Dependem também do fato de se usar truncamento ou arredondamento dos resultados das operações aritméticas. Normalmente, porém, as seguintes hipóteses são feitas:

- o ruído de quantização do sinal é uma variável aleatória não correlacionada de amostra para amostra;
- quaisquer duas fontes de erro são não correlacionadas entre si;
- cada fonte de erro é não correlacionada com a entrada.

Estas hipótese não são válidas para qualquer tipo de representação numérica ou qualquer tipo de sinal de entrada, porém normalmente elas são utilizadas por virtualmente todos os autores de trabalhos na área, e os resultados teóricos obtidos pelo uso destas aproximações tem concordado com estudos experimentais [15]. Na verdade, não há muito sentido em se procurar um modelamento exato para cada tipo de representação numérica ou para cada tipo de sinal de entrada, pois como veremos mais adiante, nas expressões da variância de ruído geralmente aparece um termo 2^{-2b} multiplicando o resto da expressão, onde b é o número de bits do registro. Ora, a adição de um bit ao registro reduz a variância de ruído a 1/4 de seu valor original. Desta forma, um modelamento que apresente erros de 30-40% é perfeitamente satisfatório.

Faremos agora um estudo em separado para os casos de implementação com aritmética de ponto fixo e de ponto flutuante, devido às diferentes características destes dois tipos de implementação.

3.3.2 Aritmética de ponto fixo

Cálculo do ruído de quantização

No caso de aritmética de ponto fixo com representação numérica em complemento de dois e utilizando arredondamento, o erro $e(n)$ provocado pelas operações de multiplicação tem a função densidade de probabilidade mostrada na Fig. 3.2, onde Δ representa o menor número que pode ser escrito num registro de $b+1$ bits. Neste caso, a média é zero e a variância do erro é dada por:

$$\sigma_e^2 = \frac{\Delta^2}{12} = \frac{2^{-2b}}{12} \tag{3.24}$$

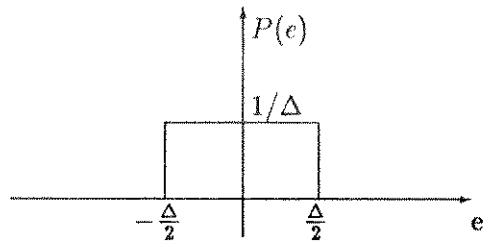

Figura 3.2: Função densidade de probabilidade do erro no caso de ponto fixo, arredondamento e complemento de dois.

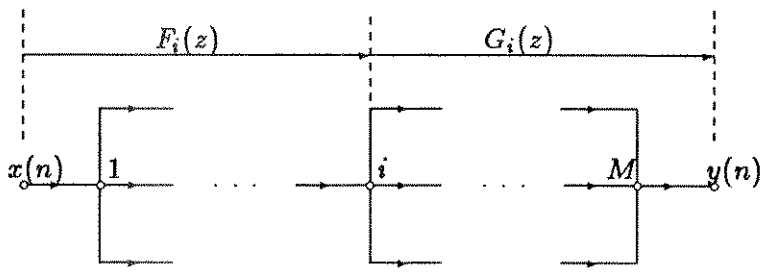

Figura 3.3: Representação de um filtro digital através de um grafo.

onde b é o número de bits do registro, excluindo-se o bit de sinal. Nas análises que se seguem, utilizaremos exclusivamente o valor acima para o modelamento do erro. Embora válido apenas para arredondamento, os resultados assim obtidos podem ser utilizados como aproximações dos outros casos.

Como sabemos, um filtro digital pode ser representado por um grafo onde os nós correspondem a pontos de soma (ou bifurcação) e os ramos correspondem a multiplicadores ou atrasadores, conforme mostrado na Fig. 3.3. Seja $f_i(n)$ o valor do nó i (a soma de suas entradas) quando a entrada $x(n)$ é uma amostra unitária, e seja $g_i(n)$ o valor da saída $y(n)$ quando o valor do nó i é uma amostra unitária e todos os demais nós estão com valores nulos. Desta forma, definimos as funções de transferência $F_i(z)$ e $G_i(z)$ da entrada para o nó i e do nó i para a saída respectivamente. Se o nó i possui k_i ramos com multiplicadores entrando no nó, haverá k_i fontes de erro entrando nesse nó, cada uma contribuindo com uma variância de $\Delta^2/12$. Com a hipótese de erros não correlacionados, as variâncias dos erros dos M nós se propagam independentemente até a saída, de forma que a variância do ruído adicionado ao sinal $y(n)$ é dada por:

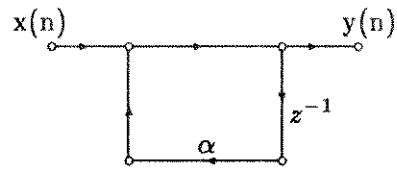

Figura 3.4: Filtro digital de primeira ordem.

$$\sigma_{e_y}^2 = \sum_{i=1}^{M} k_i \frac{\Delta^2}{12} \sum_{n=-\infty}^{\infty} g_i^2(n) \tag{3.25}$$

Aplicando-se o teorema de Parseval, a equação acima pode ser reescrita como:

$$\sigma_{e_y}^2 = \sum_{i=1}^{M} k_i \frac{\Delta^2}{12} \left[\frac{1}{2\pi j} \oint_C G_i(z) G_i(z^{-1}) z^{-1} dz \right] \tag{3.26}$$

Exemplos de aplicação de (3.25) e (3.26) para sistemas de primeira e segunda ordem podem ser encontrados em [2]. Para o filtro de primeira ordem mostrado na Fig. 3.4 com função de transferência dada por

$$H(z) = \frac{1}{1 - \alpha z^{-1}} \tag{3.27}$$

a variância do ruído de quantização adicionado ao sinal de saída é:

$$\sigma_{e_y}^2 = \frac{\Delta^2}{12} \frac{1}{1 - \alpha^2} \tag{3.28}$$

Para o filtro de segunda ordem mostrado na Fig. 3.5, com função de transferência dada por

$$H(z) = \frac{1}{1 - 2r\cos\theta z^{-1} + r^2 z^{-2}} \tag{3.29}$$

a variância do erro de quantização na saída é dada por:

$$\sigma_{e_y}^2 = 2 \frac{\Delta^2}{12} \frac{1 + r^2}{1 - r^2} \frac{1}{r^4 + 1 - 2r^2 \cos 2\theta} \tag{3.30}$$

Já para o filtro FIR mostrado na Fig. 3.6 com função de transferência dada por:

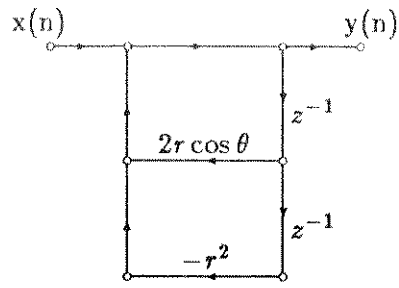

Figura 3.5: Filtro digital de segunda ordem.

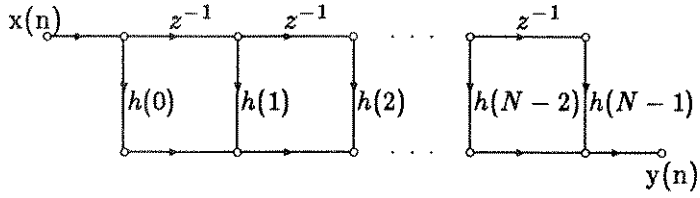

Figura 3.6: Filtro FIR implementado na forma direta.

$$H(z) = \sum_{k=0}^{N-1} h(k) z^{-k} \tag{3.31}$$

a variância do erro na saída será simplesmente:

$$\sigma_{e_y}^2 = N \frac{\Delta^2}{12} \tag{3.32}$$

Escalonamento

No caso de ponto fixo, a implementação do filtro digital deve levar em conta a possibilidade de "overflow". O tamanho dos registros empregados determina a faixa dinâmica do sinal a ser filtrado, mas mesmo que o sinal de entrada esteja dentro desta faixa, deve-se garantir que não ocorra "overflow" nos nós internos do filtro para que o sinal de saída não seja distorcido. Na verdade, nem todos os nós do grafo estão proibidos de apresentarem "overflow". Quando se utiliza representação em complemento de um ou complemento de dois, a soma parcial de dois números pode sofrer "overflow" desde que o resultado final da soma total caiba no número de bits disponível. Desta forma, somente os nós de soma cuja saída for entrada de ramos multiplicativos não

unitários não podem sofrer "overflow". Para estes nós, podemos escrever:

$$v_i(n) = \sum_{k=0}^{\infty} f_i(k)x(n-k) \tag{3.33}$$

onde $v_i(n)$ é o resultado da soma do nó i. Se $x(n)$ é limitado em magnitude por algum valor M para todo n, um limite superior para a magnitude de $v_i(n)$ é dada por:

$$|v_i(n)| \leq M \sum_{k=0}^{\infty} |f_i(k)| \tag{3.34}$$

Portanto, se $v_i(n)$ também deve ser limitado em magnitude por M, devemos ter:

$$\sum_{k=0}^{\infty} |f_i'(k)| \leq 1 \tag{3.35}$$

onde $f_i'(k)$ é a resposta impulsiva da entrada para o nó i depois de um escalonamento apropriado. Se não impusermos nenhuma restrição sobre a natureza de $x(n)$, a condição acima é necessária e suficiente para que não haja "overflow" no nó i. A igualdade na expressão acima é satisfeita se fizermos $x(n) = \pm M$ para todo n, com $\text{sgn}[x(n_0 - k)] = \text{sgn}[f_i(k)]$ para algum n_0 e todo $k \geq 0$.

Para ilustrarmos como é feito o escalonamento, tomemos o exemplo de um filtro IIR implementado na forma direta II [2], cujo grafo já escalonado é mostrado na Fig. 3.7. Note que somente os nós 1 e 2 não podem sofrer "overflow", já que os demais realizam somas parciais. A função de transferência escalonada da entrada para o nó 1 é:

$$F_1'(z) = \frac{A}{1 - \sum_{k=1}^{N} b_k z^{-k}} = AF_1(z) \tag{3.36}$$

Neste caso, a condição (3.35) será satisfeita se:

$$A \leq \frac{1}{\sum_{k=0}^{\infty} |f_1(k)|} \tag{3.37}$$

onde a igualdade maximiza a faixa dinâmica do sinal. Note que se o escalonamento de $F_1(z)$ não deve alterar o ganho da função de transferência, os coeficientes $a_k s$ devem ser divididos por A.

Para o nó 2 a função de transferência escalonada será:

$$F_2'(z) = H'(z) = BH(z) \tag{3.38}$$

onde a constante B, necessária para satisfazer (3.35), vale:

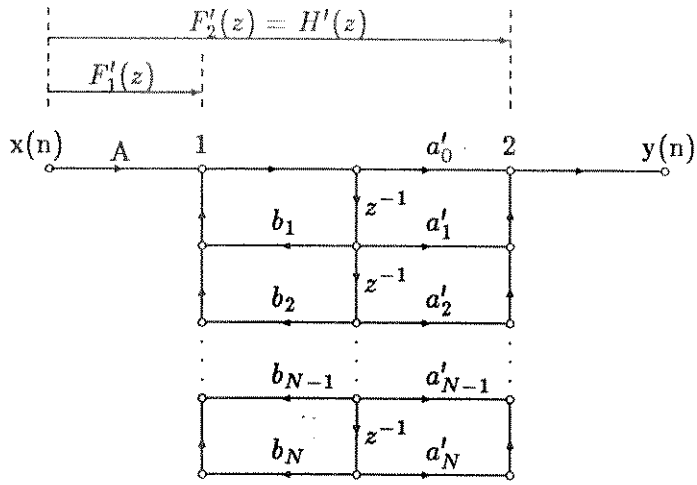

Figura 3.7: Filtro na forma direta II escalonado.

$$B \leq \frac{1}{\sum_{k=0}^{\infty} |h(k)|} \tag{3.39}$$

Esta constante pode ser incorporada aos coeficientes $a_k s$, de forma que a função escalonada resultante se torna:

$$H'(z) = A \frac{\sum_{k=0}^{N} a_k' z^{-k}}{1 - \sum_{k=1}^{N} b_k z^{-k}} \tag{3.40}$$

onde

$$a_k' = \frac{B}{A} a_k$$

A condição (3.35) é de cálculo simples no caso de filtros FIR, pois neste caso a estrutura do filtro é dada diretamente em termos da resposta impulsiva. Além disto, a probabilidade de ocorrer igualdade na expressão (3.34) não é muito baixa no caso de filtros FIR, pois a sequência de pior caso com $x(n) = \pm M$ e $\text{sgn}[x(n_0 - k)] = \text{sgn}[f_i(k)]$ tem um número finito de termos. No caso de filtros IIR entretanto, o cálculo de (3.35) é bastante difícil, já que o filtro é normalmente projetado no domínio da frequência. Torna-se interessante portanto, obter-se critérios de escalonamento em termos das transformadas z das funções de transferência. Além disto, a condição (3.35) é por demais pessimista no caso de filtros IIR, pois $h(n)$ tem comprimento infinito. Se levarmos em conta as características espectrais do sinal de entrada, obteremos critérios mais específicos e menos restritivos para cada tipo de sinal.

Tomando $v_i(n)$ dado por (3.33) , e sendo $V_i(z)$ sua transformada z, podemos escrever:

$$v_i(n) = \frac{1}{2\pi j} \oint_C V_i(z) z^{n-1} dz \tag{3.41}$$

onde C é uma curva fechada dentro da região de convergência de $V_i(z)$. Como $V_i(z)$ é estável, podemos calcular a integral acima na C.R.U., substituindo $V_i(z)$ por $F_i(z)X(z)$:

$$v_i(n) = \frac{1}{2\pi} \int_0^{2\pi} F_i(e^{j\omega}) X(e^{j\omega}) e^{j\omega n} d\omega \tag{3.42}$$

Se supusermos que $|X(e^{j\omega})|$ é limitado por algum valor M, $0 \leq \omega < 2\pi$, podemos escrever:

$$|v_i(n)| \leq M \frac{1}{2\pi} \int_0^{2\pi} |F_i(e^{j\omega})| d\omega \tag{3.43}$$

Alternativamente, aplicando a desigualdade de Schwarz [32] à expressão (3.42) temos:

$$|v_i(n)|^2 \leq \frac{1}{4\pi^2} \int_0^{2\pi} |F_i(e^{j\omega})|^2 d\omega \int_0^{2\pi} |X(e^{j\omega})|^2 d\omega \tag{3.44}$$

As expressões acima podem ser reescritas de uma forma mais elegante em termos da norma L_p, definida para uma função periódica $A(\cdot)$ arbitrária de período ω_s por:

$$||A||_p = \left[\frac{1}{\omega_s} \int_0^{\omega_s} |A(\omega)|^p d\omega \right]^{1/p} \tag{3.45}$$

para todo $p \geq 1$ tal que

$$\int_0^{\omega_s} |A(\omega)|^p d\omega < \infty$$

Se a função $A(\cdot)$ for contínua, o limite de $||A||_p$ quando $p \to \infty$ existe e é dado por [16]:

$$||A||_\infty = \max |A(\omega)| \quad , \quad 0 \leq \omega \leq \omega_s \tag{3.46}$$

Usando-se (3.45) e (3.46), (3.43) e (3.44) podem ser reescritas como:

$$|v_i(n)| \leq ||F_i||_1 \cdot ||X||_\infty \tag{3.47}$$

$$|v_i(n)| \leq ||F_i||_2 \cdot ||X||_2 \tag{3.48}$$

Pode-se mostrar [16] que as expressões acima seguem uma condição mais geral dada por:

$$|v_i(n)| \leq ||F_i||_p \cdot ||X||_q \quad , \quad \left(\frac{1}{p} + \frac{1}{q} = 1\right) \tag{3.49}$$

válida para $p \geq 1$ e $q \geq 1$.

Um caso particular, mas importante, de (3.49) acontece quando fazemos $F_i(e^{j\omega}) = 1$. Como $||1||_p = 1$ para qualquer $p \geq 1$, temos:

$$|x(n)| \leq ||X||_q \quad , \quad q \geq 1 \tag{3.50}$$

Se assumirmos, portanto, que $||X||_q$ é limitado por um valor M, então $x(n)$ também será limitado por M para todo n. Se $|v_i(n)|$ também deve ser limitado por M, a função escalonada $F_i'(e^{j\omega})$ deve obedecer:

$$||F_i'||_p \leq 1 \qquad (||X||_q \leq M) \tag{3.51}$$

para $p = q/(q-1)$. Esta é a condição de escalonamento procurada para substituir a condição mais restritiva dada por (3.35).

Se o sinal de entrada $x(n)$ for aleatório, a expressão acima não é diretamente aplicável, pois a transformada z de um sinal aleatório (de energia infinita) não é definida. Entretanto, uma condição similar pode ser obtida considerando-se a função de autocorrelação do sinal de entrada:

$$\phi_{xx}(m) = \mathrm{E}[x(n)x(n+m)] \tag{3.52}$$

Vamos assumir que o sinal de entrada é ergódico e tem média zero. Neste caso, a variância é dada por:

$$\sigma_x^2 = \mathrm{E}[x^2(n)] = \phi_{xx}(0) \tag{3.53}$$

O sinal $v_i(n)$ também será aleatório, com função de autocorrelação $\phi_{v_iv_i}(m)$. Chamando de $\Phi_{xx}(z)$ e $\Phi_{v_iv_i}(z)$ as transformadas z de $\phi_{xx}(m)$ e $\phi_{v_iv_i}(m)$, podemos escrever [2]:

$$\Phi_{v_iv_i}(z) = F_i(z)F_i(z^{-1})\Phi_{xx}(z) \tag{3.54}$$

A variância de $v(n)$ será dada por:

$$\sigma_{v_i}^2 = \phi_{v_iv_i}(0) = \frac{1}{2\pi j}\oint_C F_i(z)F_i(z^{-1})\Phi_{xx}(z)z^{-1}dz \tag{3.55}$$

Fazendo C ser a C.R.U. ($z = e^{j\omega}$), a equação acima se torna:

$$\sigma_{v_i}^2 = \frac{1}{2\pi}\int_0^{2\pi} |F_i(e^{j\omega})|^2\Phi_{xx}(e^{j\omega})d\omega \tag{3.56}$$

Como esta equação é da mesma forma que (3.42), um desenvolvimento similar ao anterior pode ser feito, levando ao seguinte resultado:

$$\sigma_{v_i}^2 \leq ||F_i||_{2p}^2 \cdot ||\Phi_{xx}||_q \qquad , \qquad \left(\frac{1}{p} + \frac{1}{q} = 1\right) \tag{3.57}$$

A partir da expressão acima podemos tirar uma condição de escalonamento para o caso em que a entrada é um ruído branco gaussiano, ou seja:

$$\phi_{xx}(m) = \sigma_x^2 \delta(m) \tag{3.58}$$

Tomando a equação (3.57) com $p = 1$ e $q = \infty$ temos:

$$\sigma_{v_i}^2 \leq ||F_i||_2^2 \cdot ||\Phi_{xx}||_\infty \tag{3.59}$$

Como $\Phi_{xx}(e^{j\omega}) = \sigma_x^2$, então $\sigma_{v_i}^2 \leq ||F_i||_2^2 \sigma_x^2$. Isto significa que a probabilidade de "overflow" da sequência $v_i(n)$ será menor ou igual à probabilidade de "overflow" da entrada se a função escalonada $F_i'(e^{j\omega})$ obedecer à condição:

$$||F_i'||_2 \leq 1 \tag{3.60}$$

que é similar à desenvolvida para o caso de sinal de entrada determinístico.

Em resumo, a condição para se evitar distorção devido a "overflow" é:

$$||F_i'||_p \leq 1 \qquad , \qquad p \geq 1 \tag{3.61}$$

onde $F_i'(e^{j\omega})$ é a função de transferência escalonada da entrada para o nó i. O valor de p deve ser escolhido em função da forma do espectro de frequências do sinal de entrada. Se o sinal de entrada for uma senóide, deve-se usar $p = \infty$, já que neste caso importa o valor de pico do sinal; se o sinal de entrada for um ruído branco, deve-se usar $p = 2$, pois neste caso é importante a variância do sinal.

Voltemos agora ao exemplo de escalonamento da forma direta II. No grafo da Fig. 3.7, calculando-se a constante A para obedecermos a condição dada por (3.61), obtém-se:

$$A = \frac{1}{\left|\left|\frac{1}{D}\right|\right|_p} \tag{3.62}$$

$$B = \frac{1}{||H||_p} \tag{3.63}$$

onde D é o denominador de $H(e^{j\omega})$. Para o caso geral, a condição (3.61) é satisfeita se os fatores de escalonamento s_i definidos por:

$$F_i'(e^{j\omega}) = s_i F_i(e^{j\omega}) \tag{3.64}$$

forem dados por:

$$s_i = \frac{1}{||F_i||_p} \tag{3.65}$$

Se escolhermos $p = 2$, podemos utilizar o teorema de Parseval para expressar $||F_i||_2$ em termos da variável z. Neste caso, o fator de escalonamento s_i será dado por:

$$s_i = \sqrt{\frac{1}{\frac{1}{2\pi j}\oint_C F_i(z)F_i(z^{-1})z^{-1}dz}} \tag{3.66}$$

que utiliza a mesma integral que aparece na expressão (3.26) do cálculo da variância de ruído na saída do filtro.

As associações em cascata e em paralelo

Vamos agora exemplificar a aplicação dos resultados anteriores analisando as duas formas mais comuns de implementação de filtros IIR: a forma paralela e a forma em cascata. A implementação em paralelo, já com os fatores de escalonamento incluídos, é mostrada na Fig. 3.8. Para simplicidade de análise, vamos assumir que o ganho do filtro já tenha sido ajustado para que a saída não apresente "overflow", ou seja,

$$||H(e^{j\omega})||_p \leq 1 \tag{3.67}$$

Observe que se $p = \infty$, a condição acima equivale a afirmar que o pico da resposta em frequência do filtro deve ser menor ou igual a um. Na verdade, se $|H(e^{j\omega})| \leq 1$, então a condição acima é satisfeita para qualquer valor de p, pois pode-se mostrar que [16]:

$$||H||_p \leq ||H||_\infty \tag{3.68}$$

A função de transferência de um filtro implementado na forma paralela é:

$$H(z) = \gamma_0 + \sum_{i=1}^{M} \frac{\gamma_{0i} + \gamma_{1i}z^{-1}}{1 + \beta_{1i}z^{-1} + \beta_{2i}z^{-2}} = \gamma_0 + \sum_{i=1}^{M} H_i(z) \tag{3.69}$$

Com a introdução dos fatores de escalonamento A_i, a função acima se torna:

$$H(z) = \gamma_0 + \sum_{i=1}^{M} A_i \frac{\gamma'_{0i} + \gamma'_{1i}z^{-1}}{1 + \beta_{1i}z^{-1} + \beta_{2i}z^{-2}} = \gamma_0 + \sum_{i=1}^{M} A_i H'_i(z) \tag{3.70}$$

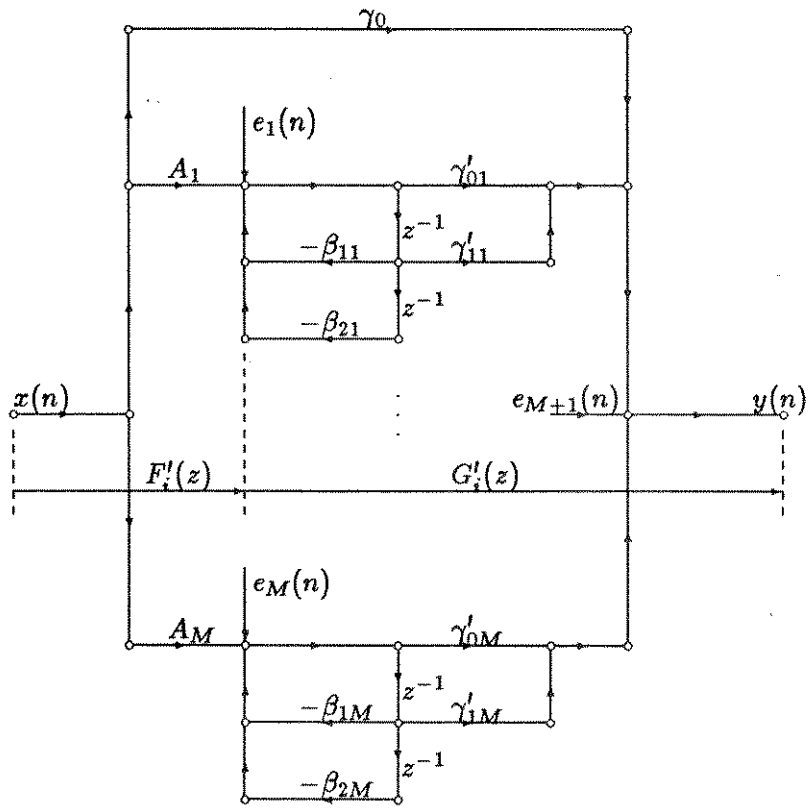

Figura 3.8: Implementação de um filtro IIR em paralelo.

onde

$$A_i = \frac{1}{\|F_i\|_p}$$

$$\gamma'_{ki} = \frac{1}{A_i}\gamma_{ki}$$

A função $F_i(z)$ não escalonada é dada por:

$$F_i(z) = \frac{1}{1 + \beta_{1i}z^{-1} + \beta_{2i}z^{-2}} \qquad , \; i = 1, 2, \cdots, M \tag{3.71}$$

Assumindo $p = 2$ (caso que se aplica quando a entrada é um ruído branco) temos:

$$A_i = \sqrt{\frac{1}{\frac{1}{2\pi j}\oint_C F_i(z)F_i(z^{-1})z^{-1}dz}} \tag{3.72}$$

Com a hipótese de erros não correlacionados, a variância do ruído da saída é simplesmente a soma das contribuições individuais dos nós internos. Para simplicidade de análise, vamos assumir que N é par, de forma que todas as seções são de segunda ordem. Olhando o grafo da Fig. 3.8, vemos que as fontes $e_1(n), \cdots, e_M(n)$ correspondem ao erro de 3 multiplicadores cada uma, e a fonte $e_{M+1}(n)$ corresponde ao erro de $N+1$ multiplicadores. Aplicando-se a expressão (3.26) para o cálculo da variância do ruído na saída obtemos:

$$\sigma^2_{e_y} = (N+1)\frac{\Delta^2}{12} + \frac{3\Delta^2}{12}\sum_{i=1}^{M}\frac{1}{2\pi j}\oint_C G'_i(z)G'_i(z^{-1})z^{-1}dz \tag{3.73}$$

onde a função $G'_i(z)$ é dada por:

$$G'_i(z) = \frac{1}{A_i}H_i(z) \tag{3.74}$$

Podemos ainda expressar $\sigma^2_{e_y}$ em termos das funções não escalonadas. Substituindo-se (3.74) em (3.73) e utilizando (3.72) chegamos a:

$$\sigma^2_{e_y} = (N+1)\frac{\Delta^2}{12} + \frac{3\Delta^2}{12}\sum_{i=1}^{M} I_{F_i}I_{H_i} \tag{3.75}$$

onde

$$I_{F_i} = \frac{1}{2\pi j}\oint_C F_i(z)F_i(z^{-1})z^{-1}dz$$

$$I_{H_i} = \frac{1}{2\pi j}\oint_C H_i(z)H_i(z^{-1})z^{-1}dz$$

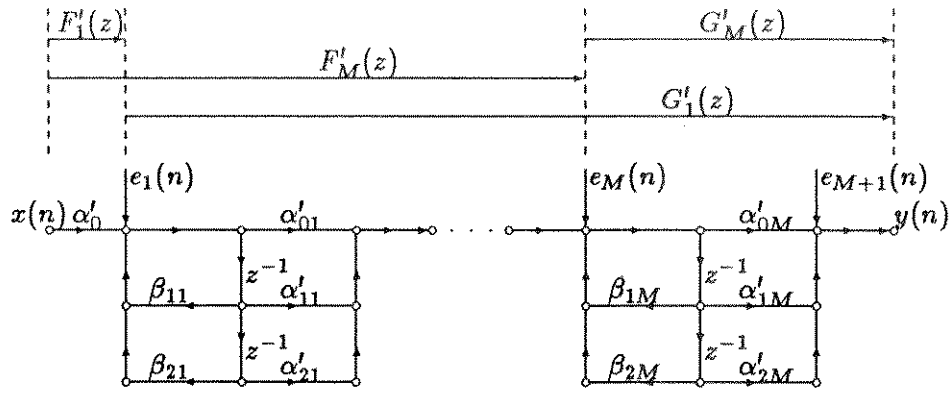

Figura 3.9: Implementação de um filtro IIR em cascata.

A implementação em cascata, já com os coeficientes escalonados, é mostrada na Fig. 3.9. Observe nesta figura que não existem multiplicadores entre as seções de segunda ordem para realizar o escalonamento. Esses multiplicadores, na verdade, foram absorvidos pelos coeficientes α'_{ki} das seções imediatamente anteriores. Para simplificação do problema vamos assumir, assim como fizemos no caso anterior, que o nó de saída não sofre "overflow". A função de transferência de um filtro implementado em cascata é da forma:

$$H(z) = a_0 \prod_{i=1}^{M} \frac{1 + \alpha_{1i} z^{-1} + \alpha_{2i} z^{-2}}{1 + \beta_{1i} z^{-1} + \beta_{2i} z^{-2}} = a_0 \prod_{i=1}^{M} \frac{\alpha_i(z)}{\beta_i(z)} \tag{3.76}$$

Com o escalonamento, esta função se torna:

$$H(z) = \alpha'_0 \prod_{i=1}^{M} \frac{\alpha'_{0i} + \alpha'_{1i} z^{-1} + \alpha'_{2i} z^{-2}}{1 + \beta_{1i} z^{-1} + \beta_{2i} z^{-2}} = \alpha'_0 \prod_{i=1}^{M} \frac{\alpha'_i(z)}{\beta_i(z)} \tag{3.77}$$

Sendo s_i o fator de escala requerido em cada seção, os coeficientes da função acima são:

$$\alpha'_0 = s_1$$

$$\alpha'_{0i} = \frac{s_{i+1}}{s_i}$$

$$\alpha'_{1i} = \frac{s_{i+1}}{s_i} \alpha_{1i}$$

$$\alpha'_{2i} = \frac{s_{i+1}}{s_i} \alpha_{2i}$$

onde

$$s_{M+1} = a_0$$

As funções de transferência não escalonadas da entrada até os nós cujos valores não podem sofrer "overflow" (no grafo, aqueles onde entram as fontes de ruído) são dadas por:

$$F_i(z) = \frac{1}{\beta_i(z)} \prod_{j=1}^{i-1} \frac{\alpha_j(z)}{\beta_j(z)} \quad , \quad i = 1, 2, \cdots, M \tag{3.78}$$

onde definimos $\prod_{j=1}^{0}(\cdot) = 1$.

Em vista de (3.64) e (3.65) temos:

$$s_i = \frac{1}{||F_i||_p} \tag{3.79}$$

Observe na Fig. 3.9 que em cada seção (a menos da última) somente um nó concentra o ruído dos vários multiplicadores (da própria seção e da anterior). A função de transferência desses nós para a saída é:

$$G'_j(z) = \prod_{i=j}^{M} \frac{\alpha'_i(z)}{\beta_i(z)} \quad , \quad j = 1, 2, \cdots, M \tag{3.80}$$

com

$$G'_{M+1}(z) = 1$$

Para simplicidade de cálculo, vamos supor novamente que a ordem do filtro é par, de forma que só há seções de segunda ordem. As fontes $e_1(n)$ e $e_{M+1}(n)$ concentram o erro de 3 multiplicadores. As demais fontes concentram o erro de 5 multiplicadores. Desta forma, a variância do erro na saída será:

$$\sigma^2_{e_y} = \frac{3\Delta^2}{12} + \frac{\Delta^2}{12} \sum_{i=1}^{M} k_i I'_{G_i} \tag{3.81}$$

onde

$$k_1 = 3$$

$$k_i = 5 \quad , \quad i = 2, 3, \cdots, M$$

$$I'_{G_i} = \frac{1}{2\pi j} \oint_C G'_i(z) G'_i(z^{-1}) z^{-1} dz$$

Da mesma forma que fizemos no caso da implementação em paralelo, podemos expressar $\sigma^2_{e_y}$ em termos das funções de transferência não escalonadas. A função $G'_j(z)$ pode ser expressa como:

$$G'_j(z) = \frac{a_0}{s_j} \prod_{i=j}^{M} \frac{\alpha_i(z)}{\beta_i(z)} = \frac{a_0}{s_j} G_j(z) \tag{3.82}$$

Substituindo (3.78), (3.79) e (3.82) em (3.81) e considerando $p = 2$ chegamos a:

$$\sigma^2_{e_y} = \frac{3\Delta^2}{12} + \frac{\Delta^2}{12} a_0^2 \sum_{i=1}^{M} k_i I_{F_i} \cdot I_{G_i} \tag{3.83}$$

onde

$$I_{F_i} = \oint_C F_i(z) F_i(z^{-1}) z^{-1} dz$$

$$I_{G_i} = \oint_C G_i(z) G_i(z^{-1}) z^{-1} dz$$

A função $F_i(z)$ é dada por (3.78) e $G_i(z)$ é dada por:

$$G_i(z) = \prod_{j=i}^{M} \frac{\alpha_j(z)}{\beta_j(z)} \quad , \quad i = 1, 2, \cdots, M \tag{3.84}$$

Examinando-se as expressões acima, vemos que a variância do ruído de um filtro implementado em cascata depende da ordenação das seções de segunda ordem, pois o produto $I_{F_i} I_{G_i}$ depende da ordenação. Não somente da ordenação das seções, mas também do emparelhamento dos pólos e zeros para a formação dos termos de segunda ordem. Na verdade o problema é mais geral. Dada uma certa função de transferência, existem inúmeras topologias que podem realizá-la, todas exatamente equivalentes se a precisão de cálculo for infinita. No caso real porém, as topologias apresentam desempenhos diferentes quanto ao ruído de quantização do sinal, e o problema de se encontrar a topologia ótima tem sido objeto de inúmeros trabalhos [20]–[30]. Voltaremos a esse assunto no capítulo seguinte.

Jackson [17] propôs algumas regras heurísticas para o emparelhamento de pólos e zeros e para a ordenação das seções. Segundo estas regras, pólos e zeros devem ser emparelhados de tal forma que as seções de segunda ordem resultantes apresentem a menor norma L_∞ possível ou seja, que as respostas em frequência não apresentem picos acentuados. Em termos práticos, isto significa que as seções de segunda ordem devem ser formadas emparelhando-se os pólos e zeros mais próximos entre si. Ainda segundo essas regras, a melhor ordenação das seções de segunda ordem depende de um parâmetro ρ_i, definido para cada seção por:

$$\rho_i = \frac{||\alpha_i/\beta_i||_\infty}{||\alpha_i/\beta_i||_2} \tag{3.85}$$

A melhor ordenação seria conseguida fazendo-se ρ_i crescer ou decrescer de seção para seção, conforme o tipo de estrutura da seção de segunda ordem e conforme se considere a potência de pico ou a variância do ruído como parâmetro de otimização. Para a estrutura mostrada na Fig. 3.9, a ordenação com seções de ρ_i crescentes forneceria a menor variância de ruído.

Não se garante que as regras heurísticas de Jackson forneçam a melhor ordenação. Na verdade, Hwang [19] apresentou um exemplo onde a ordenação e o emparelhamento de pólos e zeros por ele obtida é melhor que aquela obtida pelas regras heurísticas de Jackson (1,4 db de diferença para um filtro de oitava ordem). O algoritmo de Hwang se baseia no cálculo de (3.83) para todos os possíveis $F_i(z)$ e $G_i(z)$. Como porém, a ordem dos termos de segunda ordem dentro de cada $F_i(z)$ e $G_i(z)$ não altera o cálculo das integrais I_{F_i} e I_{G_i}, não é necessário calcular a variância de ruído para todas as permutações possíveis de termos, mas apenas as combinações. Desta forma, ao invés de se analisar todas as possíveis $(M!)^2$ implementações, o que involveria o cálculo de $2M(M!)^2$ integrais, o algoritmo de Hwang requer o cálculo de $\sum_{i=1}^{M-1} C_{i+1}^{M} C_{i}^{M} + C_{M-i}^{M} C_{M-i}^{M}$ integrais. Para um filtro de ordem elevada, esse ainda é um número muito alto, o que praticamente inviabiliza o uso do algoritmo. Vale lembrar que a aplicação das regras heurísticas de Jackson também requer uma grande quantidade de cálculo.

3.3.3 Aritmética de ponto flutuante

Quando se utiliza aritmética de ponto flutuante, um número F é representado na forma:

$$F = s \cdot M \cdot 2^c \tag{3.86}$$

onde s é o bit de sinal, M é um número entre 0,5 e 1 (ou igual a zero) chamado de mantissa, e c é um número inteiro positivo ou negativo chamado característica. Normalmente se assume que a característica possui um número suficientemente grande de bits de tal forma que nunca ocorre "overflow".

No caso de ponto aritmética de fixo, somente a operação de multiplicação introduz erro de quantização no sinal, que aparece na forma de uma variável aleatória independente da amplitude do sinal arredondado. No caso de aritmética de ponto flutuante, tanto a operação de multiplicação quanto a operação de adição introduzem erros de quantização. Neste caso, o erro devido ao arredondamento (ou truncamento) da mantissa não é independente da amplitude do número quantizado, pois o erro de quantização da mantissa também é multiplicado por 2^c. Sendo x o número de precisão infinita e $Q[x]$ o número quantizado, teremos:

$$Q[x] = x(1+\epsilon) \tag{3.87}$$

onde ϵ é o erro de quantização relativo.

Sendo b o número de bits da mantissa M, para o caso de arredondamento o erro será limitado por:

$$-2^c\frac{2^{-b}}{2} < Q[x] - x \leq 2^c\frac{2^{-b}}{2} \tag{3.88}$$

ou, como $Q[x] - x = \epsilon x$,

$$-2^c\frac{2^{-b}}{2} < \epsilon x \leq 2^c\frac{2^{-b}}{2} \tag{3.89}$$

Como a mantissa é um número entre 0,5 e 1, o número x estará entre 2^{c-1} e 2^c. Desta forma, podemos escrever:

$$-2^{-b} < \epsilon \leq 2^{-b} \tag{3.90}$$

Pode-se mostrar também [2] que para truncamento em complemento de dois temos:

$$-2\cdot 2^{-b} < \epsilon \leq 0 \tag{3.91}$$

Como fizemos no caso de aritmética de ponto fixo, consideraremos somente arredondamento e complemento de dois, sendo que os resultados assim obtidos podem ser usados como boas aproximações dos outros casos. Também suporemos que a variável ϵ tem uma função densidade de probabilidade constante uniformemente distribuida entre -2^{-b} e 2^{-b}. Sendo assim, a variância de ϵ será:

$$\sigma_\epsilon^2 = \frac{2^{-2b}}{3} \tag{3.92}$$

O procedimento para se obter a variância do ruído na saída de um filtro implementado com aritmética de ponto flutuante é bem mais complexo que no caso de aritmética de ponto fixo, pois além dos pontos de soma também introduzirem ruído, o ruído introduzido em cada nó depende da amplitude do sinal nesse nó. Para o caso de filtros IIR de primeira e segunda ordem, a análise ainda é relativamente simples. Para o filtro de primeira ordem mostrado na Fig. 3.4, pode-se mostrar [2] que a variância do erro na saída do filtro é dada por:

$$\sigma_e^2 = \frac{2^{-2b}}{3}\sigma_y^2\frac{1+\alpha^2}{1-\alpha^2} \tag{3.93}$$

onde σ_y é a variância do sinal na saída.

Para o filtro de segunda ordem mostrado na Fig. 3.5, a variância do erro na saída, no caso em que a entrada é um ruído branco, será dada por [2]:

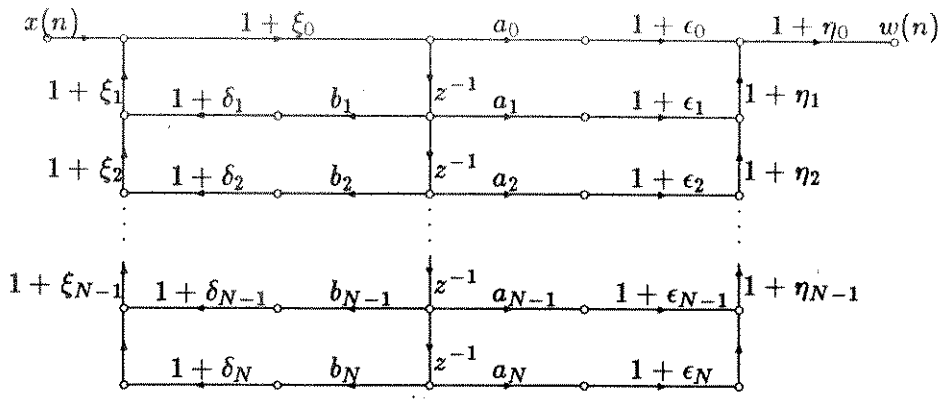

Figura 3.10: Filtro IIR implementado na forma direta com aritmética de ponto flutuante.

$$\sigma_e^2 = \frac{2^{-2b}}{3}\sigma_y^2\{(2 + r^4 + 4r^2\cos^2\theta)G - 1\} \tag{3.94}$$

onde

$$G = \frac{1 + r^2}{1 - r^2}\,\frac{1}{r^4 + 1 - 2r^2\cos 2\theta}$$

A análise de filtros de ordens maiores torna-se bem mais complicada. Tomemos o exemplo de um filtro implementado na forma direta que realize a equação:

$$y(n) = \sum_{k=0}^{N} a_k x(n-k) + \sum_{k=1}^{N} b_k y(n-k) \tag{3.95}$$

cujo grafo correspondente à implementação em aritmética de ponto flutuante é mostrado na Fig. 3.10. Com base neste grafo, podemos escrever a saída quantizada $w(n)$ como:

$$w(n) = \sum_{k=0}^{N} a_k A_k(n) x(n-k) + \sum_{k=1}^{N} b_k B_k(n) w(n-k) \tag{3.96}$$

onde

$$A_k(n) = [1 + \epsilon_k(n)] \prod_{j=0}^{k} [1 + \eta_j(n)] \quad , \quad k = 0, 1, 2, \cdots, N-1$$

$$A_N(n) = [1 + \epsilon_N(n)] \prod_{j=0}^{N-1} [1 + \eta_j(n)]$$

$$B_k(n) = [1 + \delta_k(n)] \prod_{j=0}^{k} [1 + \xi_j(n)] \quad , \quad k = 1, 2, \cdots, N-1$$

$$B_N(n) = [1 + \delta_N(n)] \prod_{j=0}^{N-1} [1 + \xi_j(n)]$$

O erro de quantização na saída do filtro pode ser calculado por:

$$e(n) = w(n) - y(n) \tag{3.97}$$

Infelizmente, a expressão (3.96) não pode ser utilizada diretamente para a obtenção das estatísticas de $e(n)$ devido à presença dos coeficientes variantes com o tempo A_k e B_k. Liu e Kaneko [7] apresentaram uma abordagem para o problema, onde se substitui $A_k(n)$ e $B_k(n)$ pelos seus valores médios de forma a se obter uma boa aproximação para o erro $e(n)$. Eles obtiveram expressões (razoavelmente longas) para o cálculo da variância do erro de filtros IIR implementados nas formas direta, em cascata e em paralelo. A utilidade prática destas expressões, entretanto, é bastante discutível, já que sistemas que utilizam aritmética de ponto flutuante normalmente são implementados com registros de mais de 20 bits, e o trabalho de se calcular a variância do erro através de expressões complicadas não se justifica, dado que neste caso o erro é muito pequeno.

Há um caso particular porém, onde a análise torna-se bem mais simples. É o caso dos filtros FIR implementados na forma direta. Como veremos mais adiante, a análise de filtros FIR para o caso em que a entrada é um ruído branco gaussiano levará a uma expressão da variância do ruído extremamente simples. Partimos das expressões (3.95) e (3.96), que para o caso de filtros FIR com N coeficientes podem ser reescritas como:

$$y(n) = \sum_{k=0}^{N-1} a_k x(n-k) \tag{3.98}$$

$$w(n) = \sum_{k=0}^{N-1} a_k A_k(n) x(n-k) \tag{3.99}$$

O erro $e(n)$ será dado por:

$$e(n) = w(n) - y(n) = \sum_{k=0}^{N-1} a_k [A_k(n) - 1] x(n-k) \tag{3.100}$$

Pode-se mostrar que a variável $[A_k(n) - 1]$ tem média zero, de forma que $e(n)$ também terá média zero. A variância do erro então será calculada por:

$$\sigma_e^2 = E[e^2(n)] = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} E\{[A_k(n)-1][A_l(n)-1]\}a_k a_l \phi_{xx}(l-k) \quad (3.101)$$

Para o caso em que a entrada é um ruído branco, ou seja $\phi_{xx}(m) = \sigma_x^2\delta(m)$, a expressão acima torna-se:

$$\sigma_e^2 = \sigma_x^2 \sum_{k=0}^{N-1} a_k^2\{E[A_k^2(n)]-1\} \quad (3.102)$$

Pode-se verificar facilmente que

$$E[A_{N-1}^2] = \left(1+\frac{2^{-2b}}{3}\right)^N$$

$$E[A_k^2] = \left(1+\frac{2^{-2b}}{3}\right)^{k+2} \quad , \quad k \neq N-1$$

Como $2^{-2b}/3 \ll 1$, o erro introduzido na expressão de baixo quando se faz $k = N-1$ é muito pequeno, de forma que por simplicidade utilizaremos somente esta expressão. Utilizando uma aproximação binomial, podemos expressar a equação acima na seguinte forma:

$$E[A_k^2(n)] = 1 + (k+2)\frac{2^{-2b}}{3} \quad (3.103)$$

Assim, a variância do erro na saída do filtro será simplesmente:

$$\sigma_e^2 = \frac{2^{-2b}}{3}\sigma_x^2 \sum_{k=0}^{N-1}(k+2)a_k^2 \quad (3.104)$$

3.3.4 Comparação entre ponto fixo e ponto flutuante

O objetivo desta seção é apresentar uma comparação entre as aritméticas de ponto fixo e de ponto flutuante sob o ponto de vista do desempenho em relação ao ruído de quantização do sinal. Como a análise de um filtro implementado com aritmética de ponto flutuante é muito complexa, basearemos nossa comparação em três casos particulares: os filtros IIR de primeira e segunda ordem mostrados nas figuras 3.4 e 3.5, e o filtro FIR da Fig. 3.6. Para simplificar a análise faremos uma comparação de certa forma "injusta" para a aritmética de ponto fixo, pois consideraremos que a mantissa da representação em ponto flutuante tem o mesmo número de bits utilizado na representação em ponto fixo inteira, não se considerando os bits necessários

para representar a característica. Tomemos os filtros IIR de primeira e segunda ordem, que apresentam variâncias de ruído de quantização de sinal dadas por (3.28) e (3.30) no caso de ponto fixo, e por (3.93) e (3.94) no caso de ponto flutuante. No caso de aritmética de ponto flutuante, a relação ruído-sinal é independente da amplitude do sinal, sendo dada por:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{3}\frac{1+\alpha^2}{1-\alpha^2} \tag{3.105}$$

para o filtro de primeira ordem, e por:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{3}\{(2+r^4+4r^2\cos^2\theta)G-1\} \tag{3.106}$$

para o filtro de segunda ordem.

Para efeito de comparação, vamos particularizar estas expressões para o caso em que os pólos do filtro estão muito próximos da C.R.U. Com este procedimento obteremos expressões mais simples e estaremos considerando a situação onde o efeito do ruído de quantização é mais severo. Seja o parâmetro δ definido por

$$\delta = 1 - |\alpha| \tag{3.107}$$

no caso do filtro de primeira ordem, e por

$$\delta = 1 - r \tag{3.108}$$

no caso do filtro de segunda ordem. Se $\delta \ll 1$, as expressões (3.105) e (3.106) podem ser aproximadas para [2]:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{3}\frac{1}{\delta} \tag{3.109}$$

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{12}\frac{3+4\cos^2\theta}{\delta\sin^2\theta} \tag{3.110}$$

No caso de aritmética de ponto fixo a situação é um pouco mais complicada, pois a relação ruído-sinal depende da amplitude do sinal, e por consequência, do tipo de escalonamento utilizado. Se o escalonamento for baseado no critério da norma L_∞, ambos os filtros devem apresentar os picos da resposta em frequência menores que 1 (ou seja, $||H||_\infty \leq 1$). Neste caso, aos filtros de primeira e segunda ordem deve-se acrescentar um multiplicador na entrada igual ao inverso do ganho de $H(e^{j\omega})$ na frequência de ressonância. Assim sendo, as funções de transferência escalonadas se tornam respectivamente:

$$H_1'(z) = \frac{1 - |\alpha|}{1 - \alpha z^{-1}} \tag{3.111}$$

$$H_2'(z) = \frac{(1-r)\sqrt{1 - 2r\cos 2\theta + r^2}}{1 - 2r\cos\theta z^{-1} + r^2 z^{-2}} \tag{3.112}$$

Com as funções de transferência escalonadas acima, o sinal de entrada pode excursionar no intervalo $[-1, 1]$. Se a entrada for um ruído branco com função densidade de probabilidade constante entre -1 e 1, a relação ruído-sinal na saída será:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{4} \frac{1}{(1 - |\alpha|)^2} \tag{3.113}$$

para o filtro de primeira ordem, e:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{2} \frac{1}{(1-r)^2(1 + r^2 - 2r\cos 2\theta)} \tag{3.114}$$

para o filtro de segunda ordem.

Aqui novamente vamos aproximar estas expressões para o caso de pólos muito próximos da C.R.U. Com $\delta \ll 1$, onde δ é definido por (3.107) e (3.108), as expressões (3.113) e (3.114) se tornam respectivamente [2]:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{4} \frac{1}{\delta^2} \tag{3.115}$$

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{2} \frac{1}{4\delta^2 \sin^2\theta} \tag{3.116}$$

Comparando-se as expressões (3.109) e (3.110) com (3.115) e (3.116) vemos que a relação ruído-sinal para a implementação com aritmética de ponto flutuante é proporcional a $1/\delta$, enquanto que para ponto fixo, ela é proporcional a $1/\delta^2$. Portanto, para pólos se aproximando da C.R.U., ou seja, para filtros de alta seletividade, a relação ruído-sinal é bem maior no caso de aritmética de ponto fixo.

Pode-se chegar à mesma conclusão acima através da análise do filtro FIR da Fig. 3.6. A relação ruído-sinal para a implementação em aritmética de ponto flutuante pode ser escrita na forma:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{\frac{2^{-2b}}{3}(N+1)\sum_{k=0}^{N-1} \frac{k+2}{N+1} h^2(k)}{\sum_{k=0}^{N-1} h^2(k)} \tag{3.117}$$

que como se observa independe da amplitude do sinal. Notando-se que:

$$\sum_{k=0}^{N-1} \frac{k+2}{N+1} h^2(k) \leq \sum_{k=0}^{N-1} h^2(k)$$

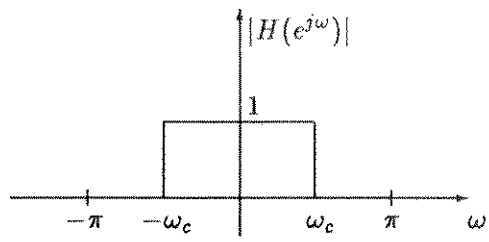

Figura 3.11: Função de transferência de um filtro passa-baixa ideal.

podemos encontrar um limite superior para a relação ruído-sinal acima:

$$\frac{\sigma_e^2}{\sigma_y^2} \leq \frac{2^{-2b}}{3}(N+1) \tag{3.118}$$

Para ponto fixo, a relação ruído-sinal depende da amplitude do sinal. Considerando-se que o filtro foi escalonado corretamente, pode-se aplicar um sinal na entrada com variância de até 1/3. Neste caso a relação ruído-sinal será:

$$\frac{\sigma_e^2}{\sigma_y^2} = \frac{2^{-2b}}{4}\frac{N}{\sum_{k=0}^{N-1} h^2(k)} \tag{3.119}$$

Comparando-se as expressões (3.118) e (3.119) acima vemos que a principal diferença entre as duas está na presença do denominador $\sum_{k=0}^{N-1} h^2(k)$ na expressão relativa à implementação em aritmética de ponto fixo. Ora, pelo teorema de Parseval podemos afirmar que

$$\sum_{k=0}^{N-1} h^2(k) = \frac{1}{2\pi}\int_{-\pi}^{\pi} |H(e^{j\omega})|^2 d\omega$$

Se a função de transferência do filtro puder ser aproximada por aquela mostrada na Fig. 3.11, a integral acima será dada aproximadamente por:

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} |H(e^{j\omega})|^2 d\omega \approx \frac{\omega_c}{\pi} \tag{3.120}$$

Para um filtro de faixa estreita temos $\omega_c/\pi \ll 1$, de forma que a relação ruído-sinal da implementação em aritmética de ponto fixo dada por (3.119) torna-se bem maior que aquela para ponto flutuante dada por (3.118).

Como conclusão, podemos afirmar que a implementação em aritmética de ponto flutuante produz uma relação ruído-sinal menor que a implementação em aritmética de ponto fixo. É importante se lembrar que esta comparação foi feita supondo-se que o número de bits utilizado na mantissa é igual ao

número total de bits da representação em ponto fixo. Por outro lado, no caso de ponto fixo consideramos que a amplitude do sinal fosse tão grande quanto possível. Se o sinal diminuir de amplitude, a relação ruído-sinal aumenta proporcionalmente, o que não acontece no caso de ponto flutuante.

Não obstante as desvantagens da aritmética de ponto fixo, tanto em termos de faixa dinâmica quanto em termos de ruído de quantização do sinal, ela é muito utilizada no processamento digital de sinais. Sua grande vantagem é a simplicidade de implementação e rapidez de cálculo, o que a torna ideal para o processamento em tempo real de sinais de frequências mais elevadas. Entretanto, processadores de sinais empregando aritmética de ponto flutuante (geralmente com mais de 20 bits) já conquistam um espaço importante entre as aplicações de processamento digital de sinais, e a evolução da tecnologia de circuitos integrados VLSI pode, num futuro próximo, mudar completamente o cenário tecnológico em favor da aritmética de ponto flutuante.

Capítulo 4

Variáveis de estado e filtros ótimos

4.1 Introdução

Vimos no capítulo anterior que as várias formas possíveis de implementação de um filtro digital possuem desempenhos diferentes quanto ao ruído de quantização do sinal. Vimos por exemplo, que na forma de implementação em cascata, o emparelhamento dos pólos e zeros e a ordenação das seções de segunda ordem produzem resultados bem diferentes quanto à relação sinal-ruído do filtro. Na verdade, como iremos mostrar mais adiante, existem infinitas maneiras de se implementar um filtro digital, todas elas com desempenhos diferentes quanto à complexidade de implementação (número de multiplicadores, somadores e atrasadores) e quanto ao ruído de quantização.

O problema que se coloca aqui é o seguinte: existirá algum método analítico de se determinar uma estrutura ou uma classe de estruturas que apresentem a menor relação ruído-sinal possível? Mais ainda, como serão estas estruturas do ponto de vista da complexidade de implementação? Será possível minimizar simultaneamente a relação ruído-sinal e a complexidade? Estas questões ainda não foram totalmente resolvidas e permanecem como um vasto campo de pesquisas, porém já existe um considerável material teórico sobre o assunto, oferecendo estruturas com nível de ruído bem inferior que as formas canônicas mais tradicionais, às custas de uma complexidade de implementação um pouco maior.

Neste capítulo, abordaremos os aspectos principais da teoria dos assim chamados *filtros ótimos*. Esta teoria se aplica a filtros implementados com aritmética de ponto fixo e baseia-se no grande poder analítico e algébrico de uma descrição matricial dos filtros digitais: *a descrição por variáveis de estado*, que introduziremos a seguir.

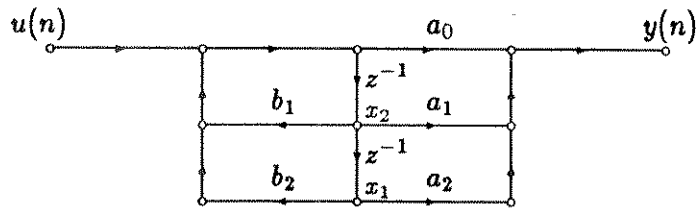

Figura 4.1: Filtro de segunda ordem na forma direta.

4.2 A representação por variáveis de estado

Na representação por variáveis de estado, o filtro é descrito por um conjunto de equações relacionando o conteúdo dos elementos de memória (no grafo, as saídas dos atrasadores), as entradas e as saídas. Define-se como *estado* do filtro o conteúdo dos elementos de memória. Sendo $x_i(n)$ a saída do atrasador i, o estado do filtro no instante n será dado pelo vetor

$$\mathbf{x}^T(n) = [x_1(n), x_2(n), \cdots, x_N(n)] \tag{4.1}$$

De uma forma geral, as equações da representação por variáveis de estado são da seguinte forma

$$\mathbf{x}(n+1) = \mathbf{A}\mathbf{x}(n) + \mathbf{B}\mathbf{u}(n) \tag{4.2}$$

$$\mathbf{y}(n) = \mathbf{C}\mathbf{x}(n) + \mathbf{D}\mathbf{u}(n) \tag{4.3}$$

onde $\mathbf{y}(n)$ é o vetor das L saídas, $\mathbf{u}(n)$ é o vetor das M entradas e $\mathbf{x}(n)$ é o vetor dos N estados. As matrizes de coeficientes **A**,**B**,**C**,**D** tem ordens $N \times N$, $N \times M$, $L \times N$ e $L \times M$ respectivamente. Para o caso mais comum de apenas uma entrada e uma saída, a matriz **B** se reduz a um vetor coluna de N elementos, a matriz **C** se reduz a um vetor linha de N elementos e a matriz **D** torna-se um escalar. Por exemplo, o filtro de segunda ordem implementado na forma direta mostrado na Fig. 4.1, com o vetor de estados $\mathbf{x}^T = [x_1, x_2]$, tem a seguinte representação por variáveis de estado:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ b_2 & b_1 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} a_2 + b_2 a_0 & a_1 + b_1 a_0 \end{bmatrix} \quad D = a_0$$

Já o filtro mostrado na Fig. 4.2 tem a seguinte representação:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \qquad D = d$$

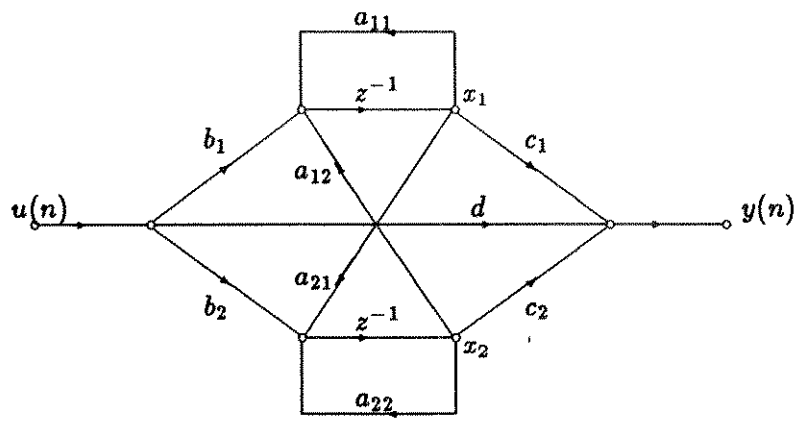

Figura 4.2: Filtro de segunda ordem implementado na forma de variáveis de estado.

É importante notar que embora o filtro da Fig. 4.1 tenha uma representação por variáveis de estado, ele próprio não é uma implementação por variáveis de estado, pois os coeficientes da matriz C são dados por combinações lineares dos coeficientes do grafo. Para que um grafo represente uma implementação por variáveis de estado, é necessário que ele satisfaça as seguintes restrições:

- todos os ramos multiplicativos devem se originar da saída de um atrasador ou de uma entrada, e terminar na entrada de um atrasador ou numa saída;
- não deve haver nenhum registro além daqueles que armazenam os estados.

O grafo da Fig. 4.2 é a implementação por variáveis de estado de uma seção de segunda ordem, mas uma associação em cascata de tais seções não é, pois neste caso existe a necessidade de registros entre as seções para armazenar resultados intermediários. Daqui para frente, sempre que nos referirmos a um filtro de variáveis de estado, estaremos nos referindo a um filtro cujo grafo obedece às restrições acima.

A função de transferência e a resposta impulsiva do filtro podem ser obtidas diretamente a partir das matrizes de estado. Tomando-se a transformada z de (4.2) e (4.3) e restringindo-se ao caso de apenas uma entrada e uma saída verifica-se facilmente que:

$$H(z) = D + \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} \tag{4.4}$$

e, através da recorrência destas equações, demonstra-se que:

$$h(n) = \begin{cases} 0, & n < 0 \\ D, & n = 0 \\ \mathbf{CA}^{n-1}\mathbf{B}, & n > 0 \end{cases} \tag{4.5}$$

Observe que na expressão de $h(n)$ acima, a matriz $\mathbf{A}$ é elevada à potência $n-1$. Na verdade, pode-se mostrar [22] que as potências de uma matriz $\mathbf{A}$ $N \times N$ são soluções de uma equação a diferenças de ordem N, o que está de acordo com a expressão de $h(n)$ acima.

A expressão (4.4) nos permite estabelecer o critério de estabilidade em termos da descrição por variáveis de estado. Observe que o denominador de $H(z)$ é calculado por:

$$a(z) = \det(z\mathbf{I} - \mathbf{A}) \tag{4.6}$$

Ora, as raízes do polinômio acima são exatamente os autovalores da matriz $\mathbf{A}$, que portanto são iguais aos pólos de $H(z)$. Concluímos então, que a condição necessária e suficiente para que o filtro seja estável é que os autovalores λ_k da matriz $\mathbf{A}$ satisfaçam:

$$|\lambda_k| < 1 \quad , \quad k = 1, 2, \cdots, N \tag{4.7}$$

Assumindo que a condição de estabilidade acima seja satisfeita, é fácil verificar por recorrência da expressão (4.2), que o vetor de estados $\mathbf{x}(n)$ num instante k qualquer pode ser expresso em função da entrada em todos os instantes anteriores a k na forma:

$$\mathbf{x}(k) = \sum_{l=1}^{\infty} \mathbf{A}^{l-1}\mathbf{B}\mathbf{u}(k-l) \tag{4.8}$$

4.3 Transformações matriciais

Vamos agora provar uma afirmação feita na introdução do capítulo, que dizia existirem infinitas maneiras de se implementar um filtro digital a partir da mesma função de transferência. Introduziremos uma transformação de coordenadas no vetor de estados $\mathbf{x}(n)$ que, embora sem alterar a função de transferência ou a resposta impulsiva, levará à formação de um conjunto diferente de equações de estado. Seja $\mathbf{T}$ uma matriz $N \times N$ não singular, e seja

$$\mathbf{q}(n) = \mathbf{T}^{-1}\mathbf{x}(n) \tag{4.9}$$

Substituindo-se $\mathbf{x}(n)$ por $\mathbf{q}(n)$ em (4.2) e (4.3), e supondo a existência de apenas uma entrada e uma saída, temos:

$$\mathbf{q}(n+1) = \mathbf{T}^{-1}\left[\mathbf{A}\mathbf{x}(n) + \mathbf{B}u(n)\right] = \mathbf{T}^{-1}\mathbf{A}\mathbf{T}\mathbf{q}(n) + \mathbf{T}^{-1}\mathbf{B}u(n) \quad (4.10)$$

$$y(n) = \mathbf{C}\mathbf{T}\mathbf{q}(n) + Du(n) \quad (4.11)$$

Com esta transformação de coordenadas obtivemos um novo conjunto de equações de estado em função do vetor $\mathbf{q}$, onde os coeficientes são obtidos das matrizes de estado originais através das seguintes transformações matriciais:

$$(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}) \Longrightarrow (\mathbf{T}^{-1}\mathbf{A}\mathbf{T}, \mathbf{T}^{-1}\mathbf{B}, \mathbf{C}\mathbf{T}, \mathbf{D}) \quad (4.12)$$

Pode-se facilmente verificar que a nova descrição por variáveis de estado possui exatamente a mesma função de transferência da descrição inicial, porém muito provavelmente com desempenho diferente em termos do ruído de quantização do sinal[1]. Desta forma, qualquer matriz não singular $\mathbf{T}$ aplicada às matrizes de estado segundo (4.12) fornece uma implementação diferente, comprovando nossa afirmação inicial de que há infinitas formas de se implementar um filtro. Como veremos mais adiante, a chave para se otimizar um filtro em termos de ruído de quantização consiste na procura de uma transformação de similaridade $\mathbf{T}$ apropriada, que forneça a estrutura com a menor potência de ruído de quantização possível.

4.4 Variáveis de estado e o escalonamento

A descrição por variáveis de estado permite um tratamento matricial muito elegante do problema do escalonamento para se evitar "overflow", e também do cálculo da potência do ruído de quantização de sinal gerado pelo filtro. O problema de se evitar "overflow" nas variáveis internas do grafo é bastante simplificado, pois como o vetor de estados $\mathbf{x}(n)$ representa, na verdade, o conjunto de registros da implementação física, precisamos tão somente garantir que este vetor não sofra "overflow".

De forma análoga à que fizemos no capítulo anterior, vamos definir o vetor coluna $\mathbf{f}(n)$ de dimensão N como sendo o vetor das respostas impulsivas da entrada $u(n)$ ao vetor de estados $\mathbf{x}(n)$. Pode-se verificar facilmente através da substituição de $u(n)$ por $\delta(n)$ em (4.2), que o vetor $\mathbf{f}(n)$ é dado por

$$\mathbf{f}(n) = \begin{cases} \mathbf{0}, & n \leq 0 \\ \mathbf{A}^{n-1}\mathbf{B}, & n > 0 \end{cases} \quad (4.13)$$

[1]Note que a matriz $\mathbf{A}$ sofre uma transformação de similaridade, de forma que os pólos do filtro (autovalores de $\mathbf{A}$) permanecem inalterados. Por este motivo, diz-se que $\mathbf{T}$ é uma transformação de similaridade aplicada sobre o filtro.

Utilizando-se o fato de que $\sum_{l=0}^{\infty} \mathbf{A}^l = (\mathbf{I} - \mathbf{A})^{-1}$ se os autovalores λ_i de $\mathbf{A}$ satisfizerem $|\lambda_i| < 1$ para $i = 1, \cdots, N$, podemos expressar a transformada z de $\mathbf{f}(n)$ como:

$$\mathbf{F}(z) = (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} \tag{4.14}$$

A seguir vamos tomar a matriz $\mathbf{K}$ definida por

$$\mathbf{K} = \sum_{k=0}^{\infty} \mathbf{f}(k)\mathbf{f}^T(k) \tag{4.15}$$

Esta matriz $N \times N$ é simétrica e, segundo [22], positiva definida. Ela é conhecida como a matriz de covariância do vetor de estados, pois no contexto dos sinais de entrada aleatórios a matriz calculada pela relação acima é idêntica àquela dada por

$$\mathbf{K} = \mathrm{E}[\mathbf{x}(k)\mathbf{x}^T(k)] \tag{4.16}$$

Do ponto de vista do problema de escalonamento, o que nos interessa desta matriz são os elementos da diagonal principal dados por:

$$K_{ii} = \sum_{k=0}^{\infty} f_i^2(k) = ||f_i||_2^2 \tag{4.17}$$

Em termos desses elementos, o critério de escalonamento segundo a norma L_2 impõe:

$$\sqrt{K'_{ii}} \leq 1 \tag{4.18}$$

O nosso problema agora é saber como modificar as matrizes $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ para se satisfazer a condição acima sem alterar a função de transferência. Para tanto, precisamos estabelecer uma relação entre essas matrizes e a matriz $\mathbf{K}$. Lembrando que $\mathbf{x}(0) = \mathbf{f}(0) = \mathbf{0}$ para $u(n) = \delta(n)$ (porque $\mathbf{x}(n)$ é a saída dos atrasadores e $\delta(-1) = 0$), podemos afirmar que:

$$\sum_{k=0}^{\infty} \mathbf{f}(k)\mathbf{f}^T(k) = \sum_{k=0}^{\infty} \mathbf{f}(k+1)\mathbf{f}^T(k+1)$$

Como $\mathbf{f}(k+1) = \mathbf{A}\mathbf{f}(k) + \mathbf{B}\delta(k)$, a expressão (4.15) torna-se:

$$\mathbf{K} = \sum_{k=0}^{\infty} [\mathbf{A}\mathbf{f}(k) + \mathbf{B}\delta(k)][\mathbf{f}^T(k)\mathbf{A}^T + \delta(k)\mathbf{B}^T] \tag{4.19}$$

Desenvolvendo-se os produtos e lembrando que $\mathbf{f}(0) = \mathbf{0}$ para $u(n) = \delta(n)$, a expressão acima se torna:

$$\mathbf{K} = \sum_{k=0}^{\infty} \mathbf{A}\mathbf{f}(k)\mathbf{f}^T(k)\mathbf{A}^T + \mathbf{B}\mathbf{B}^T \tag{4.20}$$

ou, em vista de (4.15):

$$\mathbf{K} = \mathbf{A}\mathbf{K}\mathbf{A}^T + \mathbf{B}\mathbf{B}^T \tag{4.21}$$

Embora esta expressão forneça a matriz **K** apenas implicitamente, ela pode ser resolvida através da solução de um sistema de equações lineares cujas incógnitas são as componentes de **K**. No Apêndice C apresentaremos um algoritmo eficiente para o cálculo dessa matriz.

Aplicando-se uma transformação **T** ao conjunto de matrizes (**A**,**B**,**C**,**D**) segundo (4.12), pode-se verificar através da expressão (4.21) acima que a matriz **K** será transformada segundo

$$\mathbf{K} \longrightarrow \mathbf{T}^{-1}\mathbf{K}\mathbf{T}^{-T} \tag{4.22}$$

Em particular, se

$$\mathbf{T} = \mathrm{Diag}\{t_1, t_2, \cdots, t_N\}$$

então

$$\mathbf{T}^{-1} = \mathrm{Diag}\left\{\frac{1}{t_1}, \frac{1}{t_2}, \cdots, \frac{1}{t_N}\right\}$$

e, portanto:

$$[\mathbf{T}^{-1}\mathbf{K}\mathbf{T}^{-T}]_{ij} = \frac{K_{ij}}{t_i t_j}$$

Desta forma, a condição de escalonamento dada por (4.18) será satisfeita se a descrição por variáveis de estado for transformada através de uma matriz diagonal **T** onde os elementos da diagonal são dados por:

$$t_i \geq \sqrt{K_{ii}} \quad , \quad i = 1, 2, \cdots, N \tag{4.23}$$

onde K_{ii} são as componentes da diagonal principal da matriz **K** inicial calculada através de (4.21). Observe que no procedimento de escalonamento descrito acima, não é necessário o cálculo de integrais de linha das funções de transferência da entrada aos nós a serem escalonados. É necessário apenas, resolver um sistema de equações lineares para se calcular a matriz **K** e depois calcular uma transformação diagonal **T** segundo (4.23), que aplicada às matrizes (**A**,**B**,**C**,**D**) fornecerão o filtro escalonado.

O sinal de desigualdade da expressão (4.23) acima pode ser removido se introduzirmos um fator de segurança $\delta \geq 1$ multiplicando o segundo membro da equação. Procedendo-se desta forma, obtém-se:

$$t_i = \delta\sqrt{K_{ii}} \quad , \quad i = 1, 2, \cdots, N \tag{4.24}$$

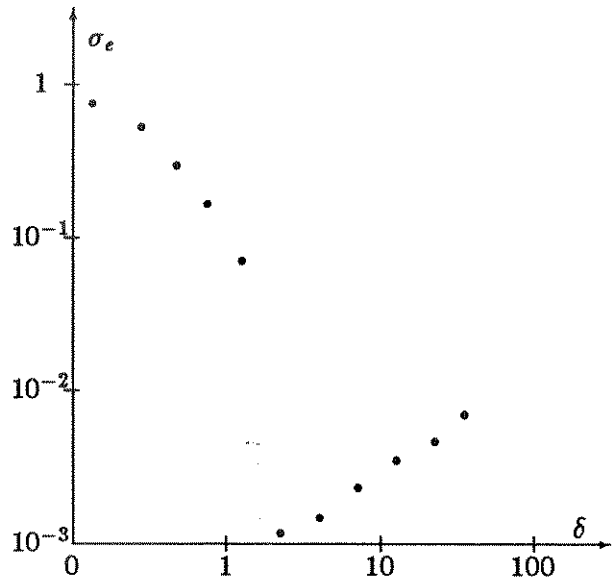

Figura 4.3: Variação da potência de ruído em função do parâmetro de escalonamento δ

e a condição de escalonamento (4.18), após a transformação $\mathbf{T}$, pode ser reescrita como:

$$\delta^2 K'_{ii} = 1 \tag{4.25}$$

onde K'_{ii} são os elementos da diagonal principal da matriz $\mathbf{K}$ após a atuação da transformação $\mathbf{T}$.

O fator δ deve ser escolhido grande o suficiente para se evitar erros devido à ocorrência de "overflow", porém não muito grande de forma a não se aumentar o ruído de quantização do sinal. Experimentos numéricos realizados por Mullis e Roberts [20] indicam que existe uma variação abrupta na potência de ruído total gerado pelo filtro para valores de δ próximo de 3. Abaixo deste valor a potência de ruído é grande devido à ocorrência de "overflow", e acima deste valor a potência de ruído se torna ordens de grandeza mais baixa, crescendo linearmente com δ (veja a Fig. 4.3, onde se mostra resultados obtidos por Mullis e Roberts [20]). Desta forma, um valor de δ de 4 ou 5 é recomendado.

4.5 Variáveis de estado e o ruído

Veremos agora como podemos expressar a variância do ruído de quantização do sinal (para aritmética de ponto fixo) na saída do filtro em termos da

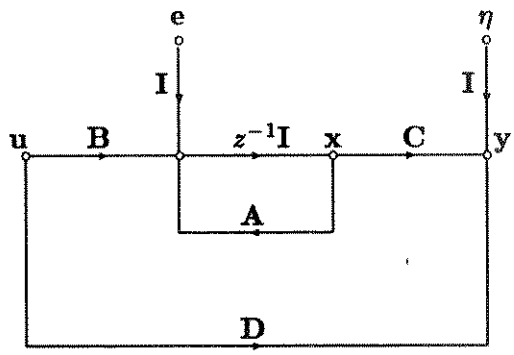

Figura 4.4: Grafo de um filtro de variáveis de estado. Os nós são vetores e os ramos são matrizes.

descrição por variáveis de estado. A Fig. 4.4 mostra o grafo de um filtro de variáveis de estado com as fontes de erro de quantização de sinal incluídas. Observe que este é um grafo matricial, onde os nós correspondem a vetores e os ramos correspondem a matrizes. Restringindo-nos novamente ao caso particular de apenas uma entrada e uma saída, as equações de estado correspondentes a esse grafo se tornam:

$$\mathbf{x}(n+1) = \mathbf{A}\mathbf{x}(n) + \mathbf{B}u(n) + \mathbf{e}(n) \tag{4.26}$$

$$y(n) = \mathbf{C}\mathbf{x}(n) + Du(n) + \eta(n) \tag{4.27}$$

onde $\mathbf{e}(n)$ é o vetor de erros de quantização do registro de estados, e $\eta(n)$ é o erro de quantização provocado no nó de saída.

De forma análoga à que definimos o vetor coluna $\mathbf{f}(n)$, definiremos o vetor linha $\mathbf{g}(n)$ de dimensão N como sendo o vetor das respostas impulsivas das componentes de $\mathbf{e}(n)$ até a saída $y(n)$. Fazendo $u(n) = 0$ e $\eta(n) = \mathbf{0}$ nas expressões acima, encontrando a transformada z de ambos os membros das duas equações, e resolvendo para $Y(z)$ em função de $\mathbf{E}(z)$, podemos expressar $\mathbf{G}(z)$ como:

$$\mathbf{G}(z) = \mathbf{C}\,(z\mathbf{I} - \mathbf{A})^{-1} \tag{4.28}$$

e, calculando a anti-transformada z, podemos expressar $\mathbf{g}(n)$ por:

$$\mathbf{g}(n) = \begin{cases} \mathbf{0}, & n \leq 0 \\ \mathbf{C}\mathbf{A}^{n-1}, & n > 0 \end{cases} \tag{4.29}$$

Assumindo-se que a variância de ruído introduzida por cada multiplicador é $\Delta^2/12$, a variância total de ruído produzida no nó de saída será:

$$\sigma_{e_y}^2 = \frac{\Delta^2}{12}\left\{\sum_{i=1}^{N} v_i \left[\sum_{k=0}^{\infty} g_i^2(k)\right] + v_{N+1}\right\} \tag{4.30}$$

onde g_i é a i-ésima componente do vetor $\mathbf{g}$, v_i é o número de multiplicadores que entra no nó i, e $N+1$ é o número associado ao nó de saída. Esta expressão vem diretamente da expressão (3.25) apresentada no capítulo anterior, e supõe que o resultado de cada multiplicação é quantizado antes de ser somado. Na prática, porém, é muito comum a utilização de acumuladores de dupla precisão para permitir que somente o resultado final da soma de cada nó seja quantizado. Com esta simplificação, e considerando-se que a contribuição do nó de saída ao ruído total é pequena (se N for grande), a expressão acima se torna:

$$\sigma_{e_y}^2 = \frac{\Delta^2}{12}\sum_{i=1}^{N}\sum_{k=0}^{\infty} g_i^2(k) \tag{4.31}$$

Da mesma forma como definimos a matriz $\mathbf{K}$ a partir do vetor $\mathbf{f}$, definiremos uma matriz $\mathbf{W}$, simétrica e positiva definida, em função do vetor $\mathbf{g}$:

$$\mathbf{W} = \sum_{k=0}^{\infty} \mathbf{g}^T(k)\mathbf{g}(k) \tag{4.32}$$

e de uma forma análoga à que fizemos para a matriz $\mathbf{K}$, pode-se mostrar que a matriz $\mathbf{W}$ relaciona-se às matrizes de estado através da equação:

$$\mathbf{W} = \mathbf{A}^T\mathbf{W}\mathbf{A} + \mathbf{C}^T\mathbf{C} \tag{4.33}$$

Aqui também, a matriz $\mathbf{W}$ é calculada de forma implícita através da solução de um sistema de equações lineares. Através da expressão acima pode-se mostrar também que uma transformação $\mathbf{T}$ aplicada ao vetor de estados $\mathbf{x}$ se reflete na matriz $\mathbf{W}$ da seguinte forma:

$$\mathbf{W} \longrightarrow \mathbf{T}^T\mathbf{W}\mathbf{T} \tag{4.34}$$

Com o uso da matriz $\mathbf{W}$, a expressão (4.31) pode ser reescrita na forma:

$$\sigma_{e_y}^2 = \frac{\Delta^2}{12}\sum_{i=1}^{N} W_{ii} \tag{4.35}$$

onde W_{ii} é o i-ésimo elemento da diagonal principal da matriz $\mathbf{W}$. Observe que o cálculo desta expressão envolve somente a solução de um sistema de equações lineares, em vez do cálculo de integrais de linha.

Vejamos agora como a expressão acima é afetada pelo escalonamento. Escalonar um filtro implementado por variáveis de estado significa aplicar uma transformação diagonal $\mathbf{T}$ ao vetor de estados $\mathbf{x}$ dada por:

$$\mathbf{T} = \text{Diag}\{\delta\sqrt{K_{11}}, \delta\sqrt{K_{22}}, \cdots, \delta\sqrt{K_{NN}}\} \tag{4.36}$$

Esta transformação altera a matriz $\mathbf{W}$ segundo (4.34), resultando:

$$W'_{ij} = W_{ij}\delta^2\sqrt{K_{ii}K_{jj}} \tag{4.37}$$

onde W'_{ij} é o elemento ij da matriz $\mathbf{W}'$ do filtro escalonado. Desta forma teremos:

$$W'_{ii} = \delta^2 W_{ii}K_{ii} \tag{4.38}$$

e a variância do ruído na saída se torna:

$$\sigma^2_{e_y} = \frac{\Delta^2}{12}\delta^2 \sum_{i=1}^{N} W_{ii}K_{ii} \tag{4.39}$$

Note a semelhança desta expressão com as equações (3.75) e (3.83), que também apresentam um termo referente ao escalonamento e um termo referente à propagação do ruído até a saída. A equação acima será a chave para a obtenção de filtros de baixo ruído.

4.6 Filtros de baixo ruído

O nosso objetivo agora é procurar uma implementação em variáveis de estado que apresente a característica de produzir a menor potência de ruído de quantização dentre todas as implementações possíveis. Dito de outra forma, vamos procurar uma transformação $\mathbf{T}$, que aplicada ao filtro projetado inicialmente, minimize a variância de ruído dada pela expressão (4.39). Esta transformação mudará as matrizes $(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{K},\mathbf{W})$ da seguinte forma:

$$(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{K},\mathbf{W}) \longrightarrow (\mathbf{T}^{-1}\mathbf{A}\mathbf{T}, \mathbf{T}^{-1}\mathbf{B}, \mathbf{C}\mathbf{T}, \mathbf{D}, \mathbf{T}^{-1}\mathbf{K}\mathbf{T}^{-T}, \mathbf{T}^{T}\mathbf{W}\mathbf{T}) \tag{4.40}$$

Pode-se verificar facilmente que a matriz produto $\mathbf{KW}$ sofre uma transformação de similaridade:

$$\mathbf{KW} \longrightarrow \mathbf{T}^{-1}(\mathbf{KW})\mathbf{T} \tag{4.41}$$

de forma que os autovalores de $\mathbf{KW}$ não são afetados pela transformação $\mathbf{T}$ (assim como os autovalores de $\mathbf{A}$, que são os pólos do filtro).

O grau de desempenho a ser obtido do filtro de mínimo ruído depende essencialmente dos graus de liberdade admitidos durante o processo de otimização. Por exemplo, o melhor desempenho possível será obtido se não impusermos nenhuma restrição em relação ao número de multiplicadores e se escolhermos o número de bits de cada registro separadamente (impondo-se apenas, que o número médio de bits por registro seja b). Na prática, entretanto, normalmente há a necessidade de se impor restrições ao processo de otimização, levando a *filtros sub-ótimos* em termos de ruído de quantização, porém com características mais desejáveis do ponto de vista de implementação. Vamos analisar primeiro o caso em que não se impõe nenhuma restrição, que nos fornecerá resultados teóricos importantes, para depois analisar casos mais práticos.

4.6.1 Filtros ótimos

Se permitirmos que o número total de bits disponível seja distribuído livremente entre os N registros, cada registro receberá b_i bits de forma que se satisfaça:

$$\sum_{i=1}^{N} b_i = Nb \tag{4.42}$$

onde b é o número médio de bits por registro. Desta forma, o intervalo de quantização de cada registro será:

$$\Delta_i = 2^{-b_i+1} \tag{4.43}$$

que substituído na expressão (4.39) fornecerá:

$$\sigma_{e_y}^2 = \frac{\delta^2}{3} \sum_{i=1}^{N} \frac{W_{ii} K_{ii}}{2^{2b_i}} \tag{4.44}$$

Para se determinar o número de bits de cada registro, usaremos a seguinte propriedade das médias aritmética e geométrica [33]:

$$\frac{1}{N} \sum_{i=1}^{N} r_i \geq \left[\prod_{i=1}^{N} r_i \right]^{1/N} \tag{4.45}$$

onde a igualdade é obtida se e somente se $r_1 = r_2 = \cdots = r_N$. Usando-se esta propriedade pode-se afirmar que:

$$\frac{1}{N} \sum_{i=1}^{N} \frac{W_{ii} K_{ii}}{2^{2b_i}} \geq \left[\prod_{i=1}^{N} \frac{W_{ii} K_{ii}}{2^{2b_i}} \right]^{1/N} \tag{4.46}$$

Impondo-se que todos os termos da soma sejam iguais (para se conseguir a igualdade) e levando-se em consideração a restrição imposta por (4.42) chega-se a:

$$b_i = b + \frac{1}{2}\log_2(K_{ii}W_{ii}) - \frac{1}{2N}\sum_{j=1}^{N}\log_2(K_{jj}W_{jj}) \tag{4.47}$$

A condição de igualdade em (4.46) aplicada a (4.44) leva a:

$$\sigma_{e_y}^2 = \frac{N}{3}\frac{\delta^2}{2^{2b}}\left[\prod_{i=1}^{N} K_{ii}W_{ii}\right]^{1/N} \tag{4.48}$$

A minimização desta expressão baseia-se na desigualdade de Hadamard [33], que diz o seguinte: se $\mathbf{P}$ for uma matriz simétrica positiva definida, então o escalar $\mathrm{e}(\mathbf{P})$ definido por

$$e(\mathbf{P}) = \left[\frac{\det(\mathbf{P})}{\prod_{i=1}^{N} P_{ii}}\right]^{1/2} \tag{4.49}$$

satisfaz a desigualdade $0 < e(\mathbf{P}) \leq 1$, e vale 1 se e somente se $\mathbf{P}$ for uma matriz diagonal. Utilizando-se a expressão acima podemos reescrever o produto dos elementos das diagonais de $\mathbf{K}$ e $\mathbf{W}$ da seguinte forma:

$$\prod_{i=1}^{N} K_{ii}W_{ii} = \prod_{i=1}^{N} K_{ii}W_{ii}\frac{\det\mathbf{KW}}{\det\mathbf{KW}} =$$

$$\prod_{i=1}^{N}\frac{K_{ii}}{\det\mathbf{K}}\prod_{i=1}^{N}\frac{W_{ii}}{\det\mathbf{W}}\det\mathbf{KW} = \left[\frac{1}{e(\mathbf{K})}\right]^2\left[\frac{1}{e(\mathbf{W})}\right]^2\det\mathbf{KW}$$

Como afirmamos anteriormente, os autovalores do produto $\mathbf{KW}$ não são afetados por uma transformação não singular $\mathbf{T}$. Desta forma, o determinante de $\mathbf{KW}$ dado por

$$\det\mathbf{KW} = \prod_{i=1}^{N}\mu_i^2 \tag{4.50}$$

onde μ_i^2 são os autovalores de $\mathbf{KW}$, também será invariante à transformação $\mathbf{T}$. Definindo-se o número M_g dado por

$$M_g = \left[\prod_{i=1}^{N}\mu_i\right]^{1/N} = [\det\mathbf{KW}]^{1/2N} \tag{4.51}$$

a expressão (4.48) pode ser reescrita na forma

$$\sigma_{e_y}^2 = \frac{N}{3}\frac{\delta^2}{2^{2b}}\frac{M_g^2}{[e(\mathbf{K})e(\mathbf{W})]^{2/N}} \tag{4.52}$$

Pela desigualdade de Hadamard, a variância do ruído será minimizada quando $e(\mathbf{K})$ e $e(\mathbf{W})$ forem iguais a 1, o que acontece *somente quando as matrizes* $\mathbf{K}$ *e* $\mathbf{W}$ *forem simultaneamente diagonais.* Se isto acontecer, a variância do ruído se torna:

$$\sigma_{e_y}^2 = \frac{N}{3}\frac{\delta^2}{2^{2b}}M_g^2 \tag{4.53}$$

Esta é a potência de ruído de quantização de sinal gerada pelo filtro ótimo. Observe que ela é proporcional ao quadrado da média geométrica dos números μ_i (as raízes quadradas dos autovalores de $\mathbf{KW}$), que recebem o nome de modos de segunda ordem do filtro. A estrutura ótima é obtida através de uma transformação $\mathbf{T}$ aplicada ao vetor de estados, que modifica as matrizes $(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})$ originais segundo (4.40), com a restrição de que esta transformação deve resultar em matrizes $\mathbf{K}$ e $\mathbf{W}$ simultaneamente diagonais. Além disso, o número de bits de cada registro deve ser calculado segundo (4.47).

O problema da diagonalização simultânea de duas matrizes é um assunto bem conhecido da álgebra linear [33] e não será discutido aqui devido ao pouco interesse prático que a estrutura obtida por este método apresenta. De uma forma geral, as matrizes $(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})$ resultantes após o procedimento acima apresentam números diferentes de 0 e 1 em todas as suas componentes, o que resulta num número de $(N+1)^2$ multiplicações por amostra, número muito maior do que o apresentado pelas estruturas mais tradicionais. Desta forma, na grande maioria das aplicações, a redução da potência de ruído simplesmente não justifica o grande número de multiplicadores necessário. Para se obter estruturas mais práticas, necessitaremos impor algumas restrições ao processo de otimização de forma a obter filtros de desempenho muito próximo ao do filtro ótimo, porém com um número de multiplicadores menor. Ainda que os filtros "sub-ótimos" assim obtidos apresentem um número de multiplicadores um pouco maior que as formas diretas, o seu uso pode ser plenamente justificado em casos nos quais o ruído de quantização for mais crítico, como por exemplo, em filtros de faixa muito estreita.

Uma observação importante a se fazer aqui diz respeito à aproximação utilizada para se chegar à equação (4.31) a partir da equação (4.30). Se em vez de supor que cada nó apresentasse apenas uma fonte de ruído, supuséssemos que cada nó apresentasse N fontes de ruído, como é o caso do filtro ótimo obtido acima, a variância de ruído do nó de saída apareceria multiplicada por N ao longo de toda a análise apresentada. O resultado final e as conclusões, entretanto, não mudariam.

4.6.2 Filtros sub-ótimos

A primeira restrição que imporemos no processo de otimização é a de que o número de bits seja igual para todos os registros. Neste caso, a expressão da potência de ruído é:

$$\sigma_{e_y}^2 = \frac{1}{3}\frac{\delta^2}{2^{2b}}\sum_{i=1}^{N} K_{ii}W_{ii} \tag{4.54}$$

cuja minimização é baseada no seguinte resultado [20]: se $\mathbf{K}$ e $\mathbf{W}$ forem matrizes reais positivas definidas, então:

$$\frac{1}{N}\sum_{i=1}^{N} K_{ii}W_{ii} \geq \left[\frac{1}{N}\sum_{i=1}^{N}\mu_i\right]^2 = M_a^2 \tag{4.55}$$

onde μ_i são os modos de segunda ordem (raízes quadradas dos autovalores de $\mathbf{KW}$). A igualdade acontece se e somente se

$$\mathbf{K} = \mathbf{DWD} \quad \text{para alguma matriz diagonal } \mathbf{D} \tag{4.56}$$

$$K_{ii}W_{ii} = K_{jj}W_{jj} \quad , \quad i,j = 1,2,\cdots,N \tag{4.57}$$

caso em que a potência de ruído é minimizada, valendo:

$$\sigma_{e_y}^2 = \frac{N}{3}\frac{\delta^2}{2^{2b}}M_a^2 \tag{4.58}$$

Comparando-se (4.58) com (4.53) podemos ver que a potência de ruído neste caso foi aumentada pelo fator $(M_a/M_g)^2$, ou seja, pelo quadrado da razão entre as médias aritmética e geométrica dos modos de segunda ordem. Esta razão só será grande se houver uma variação muito grande entre os autovalores de $\mathbf{KW}$.

A segunda restrição que imporemos é a de que o número de multiplicadores não seja muito maior que o da forma direta de implementação. A maneira mais fácil de se atingir este objetivo é através da divisão do filtro original em blocos de menor ordem, e da aplicação do método de otimização a esses blocos. Através desse procedimento, obtém-se filtros sub-ótimos que são associações em paralelo ou em cascata de filtros sub-ótimos menores.

Suponha que tenhamos inicialmente um filtro composto por uma associação em paralelo ou em cascata de seções de segunda ordem, cuja estrutura desejamos alterar através de uma transformação $\mathbf{T}$ de forma a minimizar o ruído de quantização do sinal. Para que esta transformação preserve a construção em blocos de segunda ordem, é necessário que ela seja diagonal em blocos 2×2. Desta forma, se definirmos $\mathbf{P}_i$ como sendo o i-ésimo bloco diagonal 2×2 de uma matriz $\mathbf{P}$ $N \times N$, então $\mathbf{T}$ deve satisfazer às seguintes condições:

$$
\begin{aligned}
(\mathbf{T}^{-1}\mathbf{A}\mathbf{T})_i &= \mathbf{T}_i^{-1}\mathbf{A}_i\mathbf{T}_i \\
(\mathbf{T}^{-1}\mathbf{K}\mathbf{T}^{-T})_i &= \mathbf{T}_i^{-1}\mathbf{K}_i\mathbf{T}_i^{-T} \\
(\mathbf{T}^{T}\mathbf{W}\mathbf{T})_i &= \mathbf{T}_i^{T}\mathbf{W}_i\mathbf{T}_i
\end{aligned}
$$

Como exemplo, tomemos uma cascata de 2 blocos de segunda ordem. Sendo $(\mathbf{A}_1,\mathbf{B}_1,\mathbf{C}_1,D_1)$ e $(\mathbf{A}_2,\mathbf{B}_2,\mathbf{C}_2,D_2)$ as matrizes de estado dos dois blocos, e $(\mathbf{A},\mathbf{B},\mathbf{C},D)$ as matrizes de estado da associação em cascata, pode-se verificar facilmente que:

$$
\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} \\ \mathbf{B}_2\mathbf{C}_1 & \mathbf{A}_2 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 D_1 \end{bmatrix}
$$

$$
\mathbf{C} = \begin{bmatrix} \mathbf{D}_2\mathbf{C}_1 & \mathbf{C}_2 \end{bmatrix} \quad D = D_2 D_1
$$

A transformação a ser aplicada às matrizes acima, e que preserva a estrutura cascata, será:

$$
\mathbf{T} = \begin{bmatrix} \mathbf{T}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_2 \end{bmatrix} \tag{4.59}
$$

onde $\mathbf{T}_1$ e $\mathbf{T}_2$ são matrizes 2×2 não singulares. Aplicando-se esta transformação ao conjunto $(\mathbf{A},\mathbf{B},\mathbf{C},D)$ obtém-se as matrizes

$$
\mathbf{T}^{-1}\mathbf{A}\mathbf{T} = \begin{bmatrix} \mathbf{T}_1^{-1}\mathbf{A}_1\mathbf{T}_1 & \mathbf{0} \\ \mathbf{T}_2^{-1}\mathbf{B}_2\mathbf{C}_1\mathbf{T}_1 & \mathbf{T}_2^{-1}\mathbf{A}_2\mathbf{T}_2 \end{bmatrix} \tag{4.60}
$$

$$
\mathbf{T}^{-1}\mathbf{B} = \begin{bmatrix} \mathbf{T}_1^{-1}\mathbf{B}_1 \\ \mathbf{T}_2^{-1}\mathbf{B}_2 D_1 \end{bmatrix} \tag{4.61}
$$

$$
\mathbf{C}\mathbf{T} = \begin{bmatrix} D_2\mathbf{C}_1\mathbf{T}_1 & \mathbf{C}_2\mathbf{T}_2 \end{bmatrix} \tag{4.62}
$$

$$
D = D_2 D_1 \tag{4.63}
$$

que correspondem à associação em cascata de dois blocos com matrizes de estado $(\mathbf{T}_1^{-1}\mathbf{A}_1\mathbf{T}_1, \mathbf{T}_1^{-1}\mathbf{B}_1, \mathbf{C}_1\mathbf{T}_1, D_1)$ e $(\mathbf{T}_2^{-1}\mathbf{A}_2\mathbf{T}_2, \mathbf{T}_2^{-1}\mathbf{B}_2, \mathbf{C}_2\mathbf{T}_2, D_2)$. Desta forma, a aplicação da transformação $\mathbf{T}$ à associação em cascata corresponde à aplicação das transformações $\mathbf{T}_1$ e $\mathbf{T}_2$ individualmente a cada uma das seções.

Existem 2 maneiras de se calcular a transformação $\mathbf{T}$ dada a restrição de que ela seja bloco-diagonal. A maneira mais fácil e direta é a aplicação pura e simples do processo de otimização às seções de segunda ordem isoladamente, resultando no que chamaremos de um *filtro ótimo por seção*. A grande

limitação deste método é não garantir que a associação em cascata total resultará escalonada, mesmo que cada seção esteja escalonada isoladamente[2].

A segunda maneira, um pouco mais elaborada, consiste em se tomar as matrizes **K** e **W** da associação em cascata, extrair os blocos 2×2 das diagonais e aplicar o método de otimização a esses blocos, resultando no que chamaremos de um *filtro ótimo por bloco*. Como os i-ésimos blocos 2×2 das matrizes **K** e **W** incorporam as influências acumuladas das funções $f_1, f_2, \cdots, f_i$ e $g_1, g_2, \cdots, g_i$, a associação em cascata das seções otimizadas resultará escalonada, e em consequência, com desempenho melhor em relação ao filtro ótimo por seção. É claro que no caso da associação em paralelo as duas maneiras se equivalem, já que o escalonamento de cada seção implica no escalonamento do filtro total.

Filtro ótimo por seção

Como o filtro ótimo, propriamente dito, não é de utilidade prática, a partir deste ponto adotaremos uma simplificação de notação, chamando os filtros sub-ótimos, em suas várias formas, simplesmente de filtros ótimos.

Para uma seção de segunda ordem isolada com matrizes de estado dadas por

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \qquad \mathbf{C} = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \qquad D = d$$

Jackson [18] mostrou que, no caso de pólos complexos, as condições de otimização dadas por (4.56) e (4.57) são equivalentes a:

$$a_{11} = a_{22} \tag{4.64}$$

$$b_1 c_1 = b_2 c_2 \tag{4.65}$$

com a restrição adicional de que a seção deva estar escalonada.

Na verdade, embora (4.56) e (4.57) sejam condições necessárias e suficientes, as condições acima são apenas suficientes para a obtenção de um filtro de segunda ordem ótimo. Em outras palavras, não se garante que não exista uma outra possível estrutura ótima que não satisfaça (4.64) e (4.65). Condições necessárias e suficientes foram desenvolvidas por Tavsanoglu [27], que mostrou serem as condições de Jackson um caso particular das desenvolvidas por ele. Além disso, ele mostrou também que as condições acima,

[2] A primeira pessoa a propor a idéia da otimização de seções de segunda ordem isoladamente foi Jackson [18], que na verdade incluiu o escalonamento *a posteriori* da associação em cascata como parte integrante do método. No nosso trabalho, entretanto, consideraremos que o filtro ótimo por seção deva ser escalonado apenas no âmbito das seções de segunda ordem para se manter a simplicidade computacional. De outra forma, seria difícil justificar a existência do método, já que o filtro ótimo por bloco é sempre melhor.

apesar de não valerem para qualquer tipo de função de transferência de segunda ordem, se aplicam à otimização dos tipos de filtros mais importantes — passa-baixa, passa-alta, passa-faixa, corta-faixa e passa-tudo — de pólos reais ou complexos. Desta forma, o problema da síntese ótima para esses tipos de filtros pode ser resolvido inteiramente pelo uso das condições (4.64) e (4.65) e por um procedimento puramente algébrico [30].

Seja a função de transferência de segunda ordem dada por

$$H(z) = D + \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} = d + \frac{q_1 z^{-1} + q_2 z^{-2}}{1 + p_1 z^{-1} + p_2 z^{-2}} \qquad (4.66)$$

Igualando os coeficientes de ambos os membros obtém-se quatro equações:

$$\begin{aligned} q_1 &= c_1 b_1 + c_2 b_2 \\ q_2 &= c_1 b_2 a_{12} + c_2 b_1 a_{21} - c_1 b_1 a_{22} - c_2 b_2 a_{11} \\ p_1 &= -(a_{11} + a_{22}) \\ p_2 &= a_{11} a_{22} - a_{12} a_{21} \end{aligned}$$

Estas equações, junto com (4.64) e (4.65), formam um sistema de 6 equações com 8 incógnitas (os parâmetros das matrizes $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{C}$). As duas equações a mais necessárias para a solução do sistema podem ser obtidas das condições de escalonamento

$$K_{11} = K_{22} = \frac{1}{\delta^2} \qquad (4.67)$$

através do uso da relação (4.21), que para um filtro de segunda ordem se torna:

$$\begin{bmatrix} a_{11}^2 - 1 & 2a_{11}a_{12} & a_{12}^2 \\ a_{11}a_{21} & a_{12}a_{21} + a_{11}a_{22} - 1 & a_{12}a_{22} \\ a_{21}^2 & 2a_{21}a_{22} & a_{22}^2 - 1 \end{bmatrix} \begin{bmatrix} K_{11} \\ K_{12} \\ K_{22} \end{bmatrix} = \begin{bmatrix} -b_1^2 \\ -b_1 b_2 \\ -b_2^2 \end{bmatrix}$$

Após alguma manipulação algébrica, através das oito equações obtém-se os valores dos coeficientes das matrizes $\mathbf{A}$, $\mathbf{B}$, e $\mathbf{C}$ em função dos coeficientes da função de transferência (d já aparece explicitamente em $H(z)$):

$$a_{11} = a_{22} = \frac{-p_1}{2} \qquad (4.68)$$

$$a_{21} = \sqrt{\frac{(b_{2x}^2 + v_5)v_8}{b_{1x}^2 + v_5}} \qquad (4.69)$$

$$a_{12} = \frac{v_8}{a_{21}} \qquad (4.70)$$

$$b_1 = b_{1x}/\delta \qquad (4.71)$$

$$b_2 = b_{2x}/\delta \tag{4.72}$$

$$c_1 = \frac{q_1}{2b_1} \tag{4.73}$$

$$c_2 = \frac{q_1}{2b_2} \tag{4.74}$$

onde

$$\begin{array}{ll} v_1 = q_2/q_1 & v_2 = \sqrt{v_1^2 - p_1 v_1 + p_2} \\ v_3 = v_1 - v_2 & v_4 = v_1 + v_2 \\ v_5 = p_2 - 1 & v_6 = p_2 + 1 \\ v_7 = v_5(v_6^2 - p_1^2) & v_8 = (p_1/2)^2 - p_2 \\ b_{1x} = \sqrt{\frac{v_7}{2p_1 v_3 - v_6(1+v_3^2)}} & b_{2x} = \sqrt{\frac{v_7}{2p_1 v_4 - v_6(1+v_4^2)}} \end{array}$$

Se o filtro for de ordem ímpar, a seção de primeira ordem ótima é idêntica à seção na forma direta (devido à falta de graus de liberdade para otimização). Não obstante, ela necessita ser escalonada. A matriz $\mathbf{K}$ desta seção pode ser obtida de (4.21), lembrando-se que neste caso $\mathbf{A}$ e $\mathbf{B}$ são escalares:

$$k = \frac{b^2}{1 - a^2} \tag{4.75}$$

onde a, b e k são os escalares a que se reduziram as matrizes $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{K}$. Para que a condição de escalonamento $k = 1/\delta^2$ seja obedecida, torna-se necessário aplicar-se uma transformação de escalonamento às matrizes de estado da seção de primeira ordem, o que se traduz simplesmente em se dividir b e multiplicar c por um fator de escalonamento s dado por:

$$s = \frac{\delta b}{\sqrt{1 - a^2}} \tag{4.76}$$

É bom lembrar novamente que os filtros de primeira e segunda ordem calculados pelas equações acima são ótimos e escalonados, porém o filtro ótimo por seção formado pela associação em cascata desses filtros não é ótimo nem necessariamente escalonado. Pode-se amenizar o problema da incerteza quanto ao correto escalonamento da associação em castata escolhendo-se um valor de δ um pouco maior para cada seção, às custas de algum aumento na potência de ruído de quantização.

Filtro ótimo por bloco

Analisemos agora o projeto de um filtro ótimo por bloco. Restringir-nos-emos à associação em cascata, já que a associação em paralelo pode ser resolvida mais facilmente pelo método anterior. O ponto de partida é alguma descrição por variáveis de estado de cada seção, por exemplo, de uma implementação na forma direta. A partir daí, calculam-se as matrizes ($\mathbf{A}$, $\mathbf{B}$,

$\mathbf{C}, \mathbf{D}$) da associação em cascata através do método descrito na Apêndice B. Tomam-se então os blocos 2×2 $\mathbf{K}_i$ e $\mathbf{W}_i$ das diagonais principais das matrizes $\mathbf{K}$ e $\mathbf{W}$, e determinam-se as transformações $\mathbf{T}_i$ necessárias para que cada par $(\mathbf{K}_i, \mathbf{W}_i)$ obedeça às condições de otimização (4.56) e (4.57). Isto pode ser feito através do seguinte procedimento [22]: sejam as matrizes $\mathbf{K}$ e $\mathbf{W}$, de ordem 2×2 dadas por

$$\mathbf{K} = \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \qquad \mathbf{W} = \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}$$

O primeiro passo é transformar $\mathbf{K}$ na matriz identidade através da seguinte transformação:

$$\mathbf{T}_c = \begin{bmatrix} \sqrt{\frac{k_{11}k_{22}-k_{12}^2}{k_{22}}} & \frac{k_{12}}{\sqrt{k_{22}}} \\ 0 & \sqrt{k_{22}} \end{bmatrix} \tag{4.77}$$

obtendo-se

$$\mathbf{K}' = \mathbf{T}_c^{-1}\mathbf{K}\mathbf{T}_c^{-T} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{W}' = \mathbf{T}_c^{T}\mathbf{W}\mathbf{T}_c = \begin{bmatrix} w'_{11} & w'_{12} \\ w'_{21} & w'_{22} \end{bmatrix}$$

Em seguida, aplica-se uma transformação de rotação $\mathbf{R}(\theta)$ dada por

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{4.78}$$

onde

$$\theta = \begin{cases} \frac{\pi}{4} & \text{para } w'_{11} = w'_{22} \\ \frac{1}{2}\tan^{-1}\left(\frac{2w'_{12}}{w'_{11}-w'_{22}}\right) & \text{para } w'_{11} \neq w'_{22} \end{cases}$$

obtendo-se:

$$\mathbf{K}'' = \mathbf{I} \qquad \mathbf{W}'' = \mathbf{R}(-\theta)\mathbf{W}'\mathbf{R}(\theta) = \begin{bmatrix} \mu_1^2 & 0 \\ 0 & \mu_2^2 \end{bmatrix}$$

Note que $\mathbf{K}''$ e $\mathbf{W}''$ satisfazem a condição (4.56) com $\mathbf{D} = \text{diag}\{1/\mu_1, 1/\mu_2\}$. Para satisfazer também (4.57), aplica-se a transformação $\mathbf{T}$ dada por:

$$\mathbf{T} = \frac{\delta}{2}\begin{bmatrix} \sqrt{1+\mu} & \sqrt{1+\mu} \\ -\sqrt{1+\frac{1}{\mu}} & \sqrt{1+\frac{1}{\mu}} \end{bmatrix} \tag{4.79}$$

onde $\mu = \mu_2/\mu_1$ e δ é o fator de escalonamento. Como resultado final obtém-se:

$$\mathbf{K}''' = \frac{1}{\delta^2}\begin{bmatrix} 1 & \frac{\mu_1-\mu_2}{\mu_1+\mu_2} \\ \frac{\mu_1-\mu_2}{\mu_1+\mu_2} & 1 \end{bmatrix} \tag{4.80}$$

$$\mathbf{W}''' = \delta^2\begin{bmatrix} \left(\frac{\mu_1+\mu_2}{2}\right)^2 & \frac{\mu_1^2-\mu_2^2}{4} \\ \frac{\mu_1^2-\mu_2^2}{4} & \left(\frac{\mu_1+\mu_2}{2}\right)^2 \end{bmatrix} \tag{4.81}$$

que satisfazem (4.56) e (4.57). Desta forma, a transformação a ser aplicada a cada seção de segunda ordem é $\mathbf{T}_c\mathbf{R}(\theta)\mathbf{T}$.

Se o filtro for de ordem ímpar, a seção de primeira ordem precisa ser escalonada. Para facilitar o procedimento de cálculo, vamos assumir que a seção de primeira ordem é sempre a última da cadeia, de forma que o elemento K_{NN} da matriz $\mathbf{K}$ pode ser usado para o escalonamento desta seção. Assim, sendo (a,b,c,d), as matrizes de estado da seção de primeira ordem (na verdade escalares), o escalonamento é feito dividindo-se b e multiplicando-se c pelo fator

$$s = \delta\sqrt{K_{NN}} \tag{4.82}$$

Observe que o projeto de um filtro ótimo por bloco não é muito mais trabalhoso que o de um filtro ótimo por seção. O trabalho a mais é o de se calcular as matrizes de estado e as matrizes $\mathbf{K}$ e $\mathbf{W}$ da associação em cascata. Se no projeto do filtro ótimo por seção adotássemos o procedimento de escalonar a associação em cascata globalmente, então os dois métodos seriam praticamente equivalentes em termos de complexidade computacional.

Escalonamento dos registros entre seções

Os procedimentos de escalonamento descritos até aqui se aplicam aos registros que armazenam estados. Acontece que a associação em cascata de seções de segunda ordem *não é uma implementação por variáveis de estado*, já que entre as seções existe a necessidade de registros para resultados intermediários, os quais não armazenam estados. Assim, nos procedimentos descritos acima, os registros entre as seções simplesmente não foram considerados para efeito de escalonamento ou para o cálculo do ruído de quantização. Entretanto, isto não chega a invalidar os métodos de otimização apresentados. Na verdade, os registros entre seções podem ser facilmente escalonados a partir da matriz $\mathbf{K}$ da associação em cascata, como mostraremos a seguir.

Seja $\hat{H}_i(z)$ a função de transferência da entrada até a saída da i-ésima seção. O escalonamento do registro entre as seções i e $i+1$ impõe que:

$$||\hat{H}_i||_2 = 1/\delta \tag{4.83}$$

ou, em termos da resposta impulsiva:

$$\sum_{k=0}^{\infty} \hat{h}_i^2(k) = 1/\delta^2 \tag{4.84}$$

Usando-se (4.5), a expressão acima pode ser reescrita em termos das matrizes de estado da associação em cascata das i primeiras seções:

$$\hat{D}_i^2 + \sum_{k=1}^{\infty} (\hat{\mathbf{C}}_i \hat{\mathbf{A}}_i^{k-1} \hat{\mathbf{B}}_i)(\hat{\mathbf{C}}_i \hat{\mathbf{A}}_i^{k-1} \hat{\mathbf{B}}_i) = 1/\delta^2 \tag{4.85}$$

Como $(\hat{\mathbf{C}}_i \hat{\mathbf{A}}_i^{k-1} \hat{\mathbf{B}}_i)$ é um escalar, podemos substituí-lo por seu transposto, de forma que a expressão acima pode ser escrita na forma:

$$\hat{D}_i^2 + \hat{\mathbf{C}}_i \left[\sum_{k=1}^{\infty} (\hat{\mathbf{A}}_i^{k-1} \hat{\mathbf{B}}_i)(\hat{\mathbf{A}}_i^{k-1} \hat{\mathbf{B}}_i)^T \right] \hat{\mathbf{C}}_i^T = 1/\delta^2 \tag{4.86}$$

Ora, em vista de (4.13), a somatória acima é exatamente a matriz de covariância parcial $\hat{\mathbf{K}}_i$ da associação em cascata das i primeiras seções, de forma que a condição de escalonamento se torna:

$$\hat{D}_i^2 + \hat{\mathbf{C}}_i \hat{\mathbf{K}}_i \hat{\mathbf{C}}_i^T = 1/\delta^2 \tag{4.87}$$

Pode-se verificar facilmente examinando-se (4.15), que a matriz de covariância parcial $\hat{\mathbf{K}}_i$ é formada pelas i primeiras linhas e colunas da matriz de covariância completa $\mathbf{K}$. Note também que, como a função de transferência $\hat{H}_i(z)$ não é afetada por uma transformação bloco-diagonal qualquer, a condição de escalonamento (4.87) independe da forma como cada seção foi implementada. Se as seções foram implementadas na forma de variáveis de estado, como aquela mostrada na Fig. 4.2, então o escalonamento do registro entre as seções i e $i+1$ pode ser realizado multiplicando-se os coeficientes c_1, c_2 e d da i-ésima seção por um escalar s_i dado por:

$$s_i = \frac{1}{\delta} \sqrt{\frac{1}{\hat{D}_i^2 + \hat{\mathbf{C}}_i \hat{\mathbf{K}}_i \hat{\mathbf{C}}_i^T}} \tag{4.88}$$

Para que a função de transferência do filtro não seja afetada, os coeficientes b_1, b_2 e d da seção $i+1$ devem ser divididos por s_i. Observe que após a operação de escalonamento dos registros entre seções, as matrizes de estado e as matrizes $\mathbf{K}$ e $\mathbf{W}$ da associação em cascata permanecem inalterados.

O ruído de quantização provocado pelos registros entre as seções pode ser calculado de forma aditiva à dos demais registros de estado. Assim, sendo $F_i(z)$ e $G_i(z)$ as funções de transferência associadas ao registro entre seções i, a expressão da potência de ruído (4.39) se torna:

$$\sigma_{e_y}^2 = \frac{\Delta^2}{12}\delta^2 \left\{ \sum_{i=1}^{N} K_{ii} W_{ii} + \sum_{i=1}^{N/2-1} I_{F_i} I_{G_i} \right\} \tag{4.89}$$

onde

$$I_{F_i} = \frac{1}{2\pi j} \oint_C F_i(z) F_i(z^{-1}) z^{-1} dz$$

$$I_{G_i} = \frac{1}{2\pi j} \oint_C G_i(z) G_i(z^{-1}) z^{-1} dz$$

De forma grosseira, pode-se afirmar que as integrais acima, calculadas sobre a C.R.U., são aproximadamente proporcionais à largura de faixa. Desta forma, para filtros de faixa estreita, o termo de ruído associado aos registros entre seções torna-se geralmente desprezível. Para filtros de faixa larga, este termo torna-se significativo, mas neste caso talvez nem se justifique o emprego de um filtro ótimo devido ao maior número de multiplicadores (nove no filtro ótimo contra cinco na forma direta para uma seção de segunda ordem).

4.6.3 Propriedades dos filtros de baixo ruído

Os filtros ótimos e os sub-ótimos apresentam algumas propriedades interessantes, além do baixo ruído, que ajudam a torná-los atrativos apesar do maior número de multiplicadores. A primeira propriedade se refere à invariância da potência de ruído com relação à largura de faixa. Observando-se as expressões (4.53) e (4.58) pode-se notar que a potência de ruído causada por um filtro ótimo é uma função direta dos modos de segunda ordem do filtro. Acontece que estes modos de segunda ordem são invariantes em relação a uma transformação de frequência PB-PB [21]. Como consequência, a potência de ruído produzida por um filtro ótimo não depende da largura de faixa, contrastando com os filtros implementados na forma direta, nos quais a potência de ruído cresce com a diminuição da largura de faixa [21]. Desta forma, para faixas muito estreitas ($f_c < 0{,}1 f_a$) o desempenho de um filtro ótimo se torna ordens de magnitude melhor que o da forma direta. Para larguras de faixa maiores, o desempenho de ambos os filtros é semelhante, e pode acontecer mesmo que a cascata de seções na forma direta forneça uma potência de ruído menor que as formas sub-ótimas, devido ao número menor de multiplicadores [18], embora isto raramente aconteça, e mesmo assim nestes casos a diferença é muito pequena.

A segunda propriedade que iremos mencionar diz respeito à sensibilidade da resposta em frequência com relação à variação dos coeficientes das matrizes de estado. Mostraremos que existe uma estreita correlação entre a sensibilidade à variação dos coeficientes e o desempenho em relação ao ruído

de quantização do sinal. Para mostrar isto, tomemos as derivadas de $H(z)$ com relação aos coeficientes:

$$\frac{\partial H(z)}{\partial b_i} = \left[\mathbf{C}(z\mathbf{I}-\mathbf{A})^{-1}\right]_i \tag{4.90}$$

$$\frac{\partial H(z)}{\partial c_i} = \left[(z\mathbf{I}-\mathbf{A})^{-1}\mathbf{B}\right]_i \tag{4.91}$$

$$\frac{\partial H(z)}{\partial a_{ij}} = \left[\mathbf{C}(z\mathbf{I}-\mathbf{A})^{-2}\mathbf{B}\right]_{ij} \tag{4.92}$$

onde $[]_i$ e $[]_{ij}$ indicam as componentes i e ij, respectivamente, das matrizes. Em vista de (4.14) e (4.28), as derivadas acima podem ser reescritas como

$$\frac{\partial H(z)}{\partial b_i} = G_i(z) \tag{4.93}$$

$$\frac{\partial H(z)}{\partial c_i} = F_i(z) \tag{4.94}$$

$$\frac{\partial H(z)}{\partial a_{ij}} = G_i(z)F_j(z) \tag{4.95}$$

Definindo-se $s_{ij} = \partial H(z)/\partial a_{ij}$ e utilizando-se a desigualdade de Schwarz

$$\left[\frac{1}{2\pi}\int_{-\pi}^{\pi}|F(e^{j\theta})G(e^{j\theta})|d\theta\right]^2 \leq \left[\frac{1}{2\pi}\int_{-\pi}^{\pi}|F(e^{j\theta})|^2 d\theta\right]\left[\frac{1}{2\pi}\int_{-\pi}^{\pi}|G(e^{j\theta})|^2 d\theta\right]$$

pode-se escrever:

$$||s_{ij}||_1 \leq ||G_i||_2||F_j||_2 \tag{4.96}$$

Se o filtro estiver escalonado de forma que $||F_j||_2 = 1$, temos:

$$||s_{ij}||_1 \leq ||G_i||_2 \tag{4.97}$$

Esta desigualdade nos leva a duas conclusões importantes: primeiro, que a norma L_2 de $G_i(z)$ fornece um limite superior para $||s_{ij}||_1$, e segundo, que a norma L_1 de s_{ij} fornece um limite inferior para $||G_i||_2$. Ora, examinando-se (4.31) vemos que $||G_i||_2$ nada mais é que o ganho de ruído do nó i para a saída, de forma que a sensibilidade é um limite inferior para ganho de ruído, e ganho de ruído é um limite superior para sensibilidade. Se esses limites forem razoavelmente apertados — e a experiência mostrou que realmente este é o caso [18] — então *filtros com baixa sensibilidade à variação de coeficientes também possuem baixo ganho de ruído de quantização de sinal e vice-versa.* Isto significa que o filtro ótimo terá baixa sensibilidade à variação dos parâmetros das matrizes de estado. Da mesma forma, um filtro

de baixa sensibilidade como aquele proposto por Rader e Gold [34], onde os pólos se situam ao longo de uma grade uniformemente distribuída no interior da C.R.U., também apresentará baixo ganho de ruído de quantização do sinal. De fato, Jackson [18] mostrou que o filtro de Rader e Gold e o filtro ótimo se relacionam através de uma simples transformação diagonal de escalonamento.

A terceira e última propriedade que mencionaremos diz respeito à ausência de ciclo-limites de grande amplitude (à entrada nula) causados pela ocorrência de "overflow". Mostraremos que nos filtros ótimos não há a possibilidade da existência deste tipo de instabilidade. Para começar, lembremos que num filtro implementado por variáveis de estado, somente os registros de estado podem sofrer overflow, e quando isto ocorre, o comprimento do resultado final é sempre menor ou igual ao comprimento dos registros. Desta forma, a única possibilidade de ocorrência de um novo "overflow" na ausência de um sinal de entrada, é se o vetor de estados $\mathbf{x}$ aumentar de amplitude depois de multiplicado pela matriz $\mathbf{A}$. Podemos então, afirmar que o filtro estará livre de oscilações de grande amplitude se alguma norma da matriz $\mathbf{A}$, definida de forma coerente com alguma norma de $\mathbf{x}$, for sempre menor que 1 para qualquer $\mathbf{x}$.

Tomemos a norma euclidiana do vetor $\mathbf{x}$, que é definida por:

$$||\mathbf{x}||_E = (\mathbf{x}^T\mathbf{x})^{1/2} = \left(\sum_{i=1}^{N} x_i^2\right)^{1/2} \tag{4.98}$$

Subordinada à norma euclidiana acima, define-se a norma espectral da matriz $\mathbf{A}$ através de [35]:

$$||\mathbf{A}||_S = \max_{1\leq i\leq N} |\sqrt{\mu_i}| \tag{4.99}$$

onde μ_i , $i = 1, 2, \cdots, N$, são os autovalores de $\mathbf{A}^T\mathbf{A}$. Pode-se mostrar [35] que a norma espectral de $\mathbf{A}$ se relaciona à norma euclidiana de $\mathbf{x}$ da seguinte forma:

$$||\mathbf{A}||_S = \max_{\mathbf{x}\neq 0} \frac{||\mathbf{A}\mathbf{x}||_E}{||\mathbf{x}||_E} = \max_{\mathbf{x}\neq 0} \left(\frac{\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}}{\mathbf{x}^T\mathbf{x}}\right)^{1/2} \tag{4.100}$$

Assim sendo, a condição para ausência de oscilação fica:

$$||\mathbf{A}||_S \leq 1 \tag{4.101}$$

A prova de que a condição acima é válida para os filtros ótimos baseia-se no seguinte teorema [31]: "a matriz de transição de estados $\mathbf{A}$ de um filtro de variáveis de estado com

$$\mathbf{W} = \rho^2 \mathbf{DKD} \tag{4.102}$$

satisfaz:

$$\mathbf{D} - \mathbf{A}^T \mathbf{DA} \geq \mathbf{0} \tag{4.103}$$

onde $\mathbf{D}$ é uma matriz diagonal e ρ é um escalar".

Os filtros ótimos se encaixam no teorema acima fazendo-se simplesmente $\mathbf{D} = \mathbf{I}$, podendo-se afirmar então que:

$$\mathbf{I} - \mathbf{A}^T \mathbf{A} \geq \mathbf{0} \tag{4.104}$$

Multiplicando-se ambos os membros da equação acima por $\mathbf{x}^T$ à esquerda e $\mathbf{x}$ à direita, obtém-se

$$\mathbf{x}^T \mathbf{x} - \mathbf{x}^T \mathbf{A}^T \mathbf{A} \mathbf{x} \geq 0 \tag{4.105}$$

que é exatamente equivalente a $||\mathbf{A}||_S^2 \leq 1$, provando que o filtro ótimo é livre de ciclo-limites de grande amplitude à entrada nula.

É muito fácil de se verificar que esta propriedade se aplica de forma imediata aos filtros formados pela associação em paralelo de seções de segunda ordem ótimas. Será que o mesmo é verdade para a associação em cascata? Mostraremos que sim através da argumentação seguinte. Se um ou mais registros do vetor de estados sofrer "overflow", não se pode afirmar que o vetor diminuirá de amplitude na iteração seguinte, mesmo que a matriz $\mathbf{A}_i$ de cada seção obedeça $||\mathbf{A}_i|| < 1$, pois o sinal de entrada de cada seção é fornecido pela seção anterior. No entanto, a primeira seção terá entrada nula, e portanto, os registros desta seção tenderão rapidamente a zero se $||\mathbf{A}_1|| < 1$. Assim, depois de algum tempo, a segunda seção também terá entrada nula, e assim sucessivamente até a última seção, de forma que se cada seção for livre de ciclo-limites de grande amplitude, então a associação em cascata também será.

Capítulo 5

O programa FOREST

5.1 Introdução

Neste capítulo descreveremos o programa FOREST (Filtro digital Ótimo Recursivo por variáveis de ESTado), que permite o projeto de filtros digitais recursivos na forma de cascata de seções de segunda ordem, com aritmética de ponto fixo e representação em complemento de dois. A teoria apresentada nos capítulos anteriores é utilizada pelo programa para o projeto de filtros escalonados de 3 tipos:

- cascata de seções de segunda ordem na forma direta;
- ótimo por seção;
- ótimo por bloco.

Além da descrição do programa, apresentaremos alguns exemplos de aplicação do programa no projeto de filtros, nos quais ilustraremos as diferenças de desempenho entre os filtros com seções de segunda ordem ótimas e o filtro com seções de segunda ordem na forma direta.

5.2 Descrição do programa

O programa FOREST foi desenvolvido em linguagem C para o microcomputador IBM-PC, utilizando-se o compilador Microsoft-C versão 5.1. Para efeitos didáticos, e com o objetivo de se definir uma taxionomia funcional, as rotinas que constituem o programa podem ser divididas em classes funcionais mais ou menos distintas:

- comunicação com o usuário;
- cálculo do protótipo analógico;

- cálculo das seções de segunda ordem do filtro digital;
- quantização dos coeficientes;
- análise da resposta em frequência;
- análise transiente no domínio do "tempo".

Esta classificação será utilizada como base da descrição do programa dada a seguir.

5.2.1 Comunicação com o usuário

As funções de comunicação com o usuário são responsáveis pela entrada e saída de dados, e pela apresentação dos resultados das análises no "tempo" e em frequência na forma gráfica.

A entrada de dados pode ser feita de duas maneiras: de forma conversacional ou através de arquivo de dados. Na forma conversacional, o usuário inicia a execução do programa através do comando FOREST e entra os dados que forem sendo requisitados. Se o usuário preferir entrar os dados através de arquivo (se, por exemplo, ele for executar o programa várias vezes com os mesmos dados de entrada), ele cria um arquivo com os dados de entrada no formato que daremos a seguir, e dá o comando FOREST seguido do nome do arquivo de dados. Se este arquivo não contiver todos os dados necessários, o programa informa qual dado está faltando, e interrompe a execução. O formato do arquivo de dados é o seguinte:

1. Qualquer linha cujo primeiro caracter (pulando-se os espaços) não seja o ponto é ignorada.
2. Cada dado é fornecido através de uma sigla (iniciada por um ponto) e, quando for o caso, um ou mais valores numéricos.
3. Os dados que devem obrigatoriamente ser fornecidos são:
   - frequência de amostragem (em Khz): .fa $\langle f_a \rangle$
   - tipo de filtro: .but, .che ou .eli
   - tipo de resposta em frequência: .pb, .pa, .pf ou .cf
   - atenuação máxima na faixa de passagem (em dB): .amax $\langle a_{max} \rangle$
   - atenuação mínima na faixa de rejeição (em dB): .amin $\langle a_{min} \rangle$
   - frequências da máscara (em Khz): .f $\langle f_1 \rangle$ $\langle f_2 \rangle$ ou .f $\langle f_1 \rangle$ $\langle f_2 \rangle$ $\langle f_3 \rangle$ $\langle f_4 \rangle$

4. Exemplo:

```
.fa 100
.eli
.pf
.amax 0.5
.amin 60
.f 10 12 20 22
```

O arquivo de saída, cujo nome o programa pede ao final da execução, contém todas as informações geradas durante a execução, tais como a ordem do filtro, localização dos pólos e zeros, coeficientes das seções de segunda ordem e resultados das análises transientes. O arquivo de saída também contém as especificações do filtro no mesmo formato do arquivo de entrada, de forma que o arquivo de saída pode ser usado diretamente como arquivo de entrada de uma nova execução.

Além do arquivo de saída textual, o usuário dispõe de saídas na forma gráfica. A resposta em frequência e os resultados da análise transiente podem ser visualizados diretamente na tela do computador através de gráficos com resolução de 600×200 pixels. Existe, porém, um preço a ser pago por essa facilidade fornecida ao usuário. A função que executa a saída gráfica é a única parte do programa que não é portátil, pois as funções gráficas em C normalmente variam de um compilador para outro. Assim, se o programa tiver que ser transportado para uma outra máquina, ou mesmo para outro compilador, as funções gráficas precisarão ser totalmente reescritas.

5.2.2 Cálculo do protótipo analógico

As funções pertencentes a esta classe realizam a tarefa nada trivial de calcular os pólos e zeros do filtro protótipo analógico, utilizando as aproximações de Butterworth, Chebychev ou elíptica. A sequência de cálculos realizada é a seguinte. Partindo-se das especificações do filtro digital, obtém-se as especificações do protótipo analógico através da transformação bilinear. A seguir, calculam-se pólos e zeros do filtro protótipo passa-baixa e realiza-se a transformação em frequência utilizando-se os procedimentos descritos no Capítulo 2. Essa sequência de cálculos não leva mais que 1 segundo num IBM-PC com co-processador de ponto flutuante e relógio de 8 MHz.

5.2.3 Cálculo das seções de segunda ordem do filtro digital

Essa classe de funções tem a tarefa de calcular as seções de segunda ordem da associação em cascata de três formas diferentes: forma direta, ótimo por seção e ótimo por bloco. Vejamos com mais detalhes a sequência de cálculo.

Montagem dos termos de segunda ordem

De posse dos pólos e zeros do filtro analógico, aplica-se a transformação bilinear, obtendo-se os pólos e zeros no plano z. Parte-se, então, para a montagem dos termos de segunda ordem na forma:

$$H_i(z) = \frac{b_{0_i} + b_{1_i} z^{-1} + b_{2_i} z^{-2}}{1 - c_{1_i} z^{-1} - c_{2_i} z^{-2}} \tag{5.1}$$

Aqui surge o problema do emparelhamento dos pólos e zeros e da ordenação das seções para minimização do ruído de quantização. Como se explicou nos capítulos anteriores, este é ainda um problema sem solução analítica. As regras heurísticas de Jackson apresentadas no Capítulo 3 são de cálculo difícil e de resultados duvidosos. Não obstante, algum critério precisa ser adotado, ainda que seja simplesmente o da ordenação aleatória. No nosso caso, optamos por uma adaptação das regras de Jackson com o objetivo de simplificar os cálculos. O emparelhamento de pólos e zeros segue o mesmo critério: *começando pelos pólos mais próximos da C.R.U., emparelhamos os pólos e zeros mais próximos entre si.* Já com relação à ordenação das seções, em vez de utilizarmos o parâmetro ρ_i definido por (3.85), utilizamos o ganho de pico de cada seção como parâmetro de ordenação. Restava-nos decidir qual critério de ordenação seria adotado. As regras heurísticas de Jackson sugerem a ordenação pela ordem crescente ou decrescente dos ganhos de pico, porém essas não são as únicas possibilidades. Existe, por exemplo, a alternativa de se intercalar seções de ganho alto com seções de ganho baixo, de forma a manter o ganho mais ou menos constante ao longo da cascata.

Para que pudéssemos escolher entre essas opções, fizemos o cálculo de um filtro de cada tipo (passa-baixa, passa-alta, passa-faixa e corta-faixa), determinando para cada um o ganho de ruído de três ordenações de ganho de pico diferentes: crescente, decrescente e constante. Os resultados são mostrado na Tabela 5.1, onde em cada posição aparecem, de cima para baixo, os ganhos de ruído dos filtros ótimo por seção, de seções na forma direta e ótimo por bloco, respectivamente. Como se observa nesta tabela, a ordenação com ganho constante forneceu os melhores resultados para todos os filtros, sendo que no corta-faixa a melhora chega a ser de uma ordem de grandeza. Portanto, esta ordenação foi escolhida e foi implementada da seguinte forma. Primeiro, ordena-se as seções pelos ganhos de pico de forma crescente, e depois faz-se uma troca de posições entre a segunda e a penúltima seção, e assim por diante, em seções alternadas.

Escolha do fator de escalonamento

Antes de prosseguir no cálculo das formas ótimas, o programa pede que o usuário forneça o fator de escalonamento δ. Se mais tarde, durante a análise

ordenação	*PB*	*PA*	*PF*	*CF*
crescente	12,77	15,62	5,719	798,3
	1389	15,19	24,04	538,8
	12,00	14,50	5,665	495,5
decrescente	12,77	15,62	5,719	798,3
	1361	15,89	24,73	622,2
	12,00	14,50	5,665	495,5
constante	4,886	5,323	4,421	40,08
	568,2	5,632	18,92	61,68
	4,814	5,222	4,410	31,92

Tabela 5.1: Ganho de ruído das diversas ordenações de ganho de pico.

transiente, o usuário verificar que o escalonamento não foi eficaz e que ainda ocorre "overflow" nos registros do filtro, ele tem a opção de voltar a este ponto do programa, mudar o fator de escalonamento e refazer o projeto.

Filtro ótimo por seção

De posse dos coeficientes das seções na forma direta (ainda não escalonados) e do fator de escalonamento, o programa calcula os coeficientes das seções do filtro ótimo por seção através das equações (4.68) a (4.74). O projeto do filtro ótimo por seção estaria encerrado aqui, porém o programa faz ainda alguns cálculos adicionais para a determinação de um parâmetro de mérito que será utilizado como base de comparação entre os três tipos de filtros. Esse parâmetro de mérito é o ganho de ruído, definido pela relação:

$$g_r = \sum_{i=1}^{N} K_{ii} W_{ii} \tag{5.2}$$

onde K_{ii} e W_{ii} são os elementos da diagonal principal das matrizes $\mathbf{K}$ e $\mathbf{W}$ da associação em cascata. O nome ganho de ruído é bem apropriado, pois de (4.39) vemos que a potência de ruído gerada por um *filtro escalonado*[1] é dada pelo produto de uma constante (que não depende da estrutura do filtro) pelo ganho de ruído definido acima. Note que as matrizes $\mathbf{K}$ e $\mathbf{W}$ usadas no cálculo do ganho de ruído são as matrizes da associação em cascata, e não as matrizes das seções individuais. Note ainda que a comparação em termos de ganho de ruído é válida desde que o arredondamento nas operações

[1]Na verdade, o filtro ótimo por seção foi escalonado somente a nível das seções individuais, de forma que a associação em cascata não foi (e nem será) escalonada. Porém, se o ganho de ruído for utilizado como parâmetro de comparação do nível de ruído entre os filtros, ele deve se abstrair do problema do escalonamento, devendo ser calculado para filtros escalonados da mesma forma.

de soma de produtos seja realizado apenas uma única vez (através de um acumulador de dupla precisão), hipótese que foi feita para se chegar a (4.39) a partir da expressão mais geral (4.30). Não obstante, o ganho de ruído é uma figura de mérito que pode ser perfeitamente utilizada para uma comparação aproximada, mesmo que a hipótese acima não seja válida.

Filtro com seções na forma direta

Depois do cálculo do filtro ótimo por seção, o próximo passo é o escalonamento do filtro com seções na forma direta. Primeiro calculam-se as matrizes **K** e **W** da associação em cascata não escalonada (que depois serão também utilizadas para o cálculo do filtro ótimo por bloco). De posse dessas matrizes, e antes de realizar o escalonamento, o programa calcula o ganho de ruído do filtro escalonado através de (5.2). O escalonamento, propriamente dito, é realizado através do mesmo procedimento descrito na Seção 3.3.2 do Capítulo 3, só que os coeficientes de escalonamento s_i de cada seção são obtidos a partir da matriz **K** ao invés de (3.79). Este procedimento equivale a se aplicar às matrizes de estado do filtro uma transformação de escalonamento **T** da forma:

$$\mathbf{T} = \mathrm{diag}\{s_1, s_1, s_2, s_2, \cdots, s_M, s_M\} \tag{5.3}$$

onde M é o número de seções, e o coeficiente de escalonamento da i-ésima seção é dado por:

$$s_i = \delta\sqrt{K_{jj}} \quad , \quad j = 2i \tag{5.4}$$

Note que o escalonamento através de uma matriz **T** da forma acima só é possível porque a matriz **K** de um filtro de segunda ordem na forma direta possui uma diagonal principal com elementos iguais, de forma que a matriz **K** da associação em cascata terá uma diagonal principal formada por pares de elementos iguais, cada par correspondendo a uma seção.

Filtro ótimo por bloco

O filtro ótimo por bloco é calculado através do procedimento descrito na Seção 4.6.2, a partir das matrizes **K** e **W** calculadas antes do escalonamento do filtro com seções na forma direta. Observe que para se calcular o ganho de ruído do filtro ótimo por bloco, não é necessário um novo cálculo das matrizes **K** e **W**, pois estas aparecem como subproduto do processo de cálculo do filtro.

Escalonamento dos registros entre seções

A última etapa no processo de cálculo das seções de segunda ordem é o escalonamento dos registros entre as seções seguindo o procedimento descrito na seção 4.6.2. Ele só não é feito no filtro com seções na forma direta porque, neste caso, os registros entre seções somente armazenam resultados de somas parciais, e como já foi mencionado anteriormente, as somas parciais em complemento de dois podem sofrer "overflow" desde que o resultado final não sofra.

5.2.4 Quantização dos coeficientes

As funções pertencentes a essa classe permitem ao usuário quantizar (através de arredondamento) os coeficientes do filtro. A precisão de quantização é fornecida pelo usuário em termos do número de bits, incluindo o bit de sinal.

Antes de quantizar os coeficientes, porém, o programa precisa determinar a posição do ponto na representação binária dos coeficientes. Como o filtro funcionará com aritmética de ponto fixo, a posição do ponto binário deve ser a mesma para todos os coeficientes. Assim, o programa determina o coeficiente de maior valor absoluto, e para esse coeficiente, determina o número de bits necessário para se representar a parte inteira (incluindo o sinal), o qual será usado também para os demais coeficientes.

Sendo c o coeficiente de precisão infinita, e c_q o coeficiente quantizado, a quantização é realizada através da expressão:

$$c_q = \mathcal{R}(c/\Delta) \cdot \Delta \tag{5.5}$$

onde $\mathcal{R}(\cdot)$ denota o inteiro mais próximo do argumento, e Δ é dado por:

$$\Delta = 2^{-b} \tag{5.6}$$

onde b é o número de bits da parte fracionária.

A última etapa no processo de quantização consiste em se verificar se os pólos ainda permanecem no interior da C.R.U. depois da quantização. Para as seções na forma direta utiliza-se diretamente as condições de estabilidade (3.22) e (3.23) vistas no Capítulo 3. Para as seções descritas por variáveis de estado, primeiro calcula-se o denominador da função de transferência:

$$(z\mathbf{I} - \mathbf{A})^{-1} = z^2 + az + b \tag{5.7}$$

onde

$$a = -(a_{11} + a_{22}) \tag{5.8}$$

$$b = a_{11}a_{22} - a_{12}a_{21} \tag{5.9}$$

e depois aplicam-se as condições (3.22) e (3.23).

Caso os pólos tenham se deslocado para fora da C.R.U., o programa avisa que o filtro se tornou instável e pede para o usuário requantizar os coeficientes com um número maior de bits.

5.2.5 Análise da resposta em frequência

As funções que fazem a análise da resposta em frequência permitem que se visualize na tela do computador, em forma gráfica, as curvas de resposta em frequência (módulo, módulo em dB e fase) de cada um dos três tipos de filtros. Além de permitir ao usuário verificar se o projeto do filtro foi feito corretamente, as curvas de resposta em frequência são úteis para se determinar o número mínimo de bits necessário para quantizar os coeficientes. O usuário quantiza os coeficientes e depois verifica se o filtro ainda obedece às máscaras de especificação, que também são desenhadas no mesmo gráfico da resposta em frequência. Caso contrário, aumenta o número de bits e repete o processo até que o filtro caia dentro da máscara.

5.2.6 Análise transiente no domínio do "tempo"

As funções que realizam a análise transiente permitem que se obtenha na tela do computador, em forma gráfica, a resposta do filtro no domínio do "tempo" para alguns tipos de sinais de entrada: ruído branco, senóide, amostra unitária e degrau unitário. Para cada tipo de filtro, são produzidas duas simulações: uma do filtro funcionando com aritmética de ponto flutuante de dupla precisão da linguagem C (64 bits), e outra do filtro funcionando com aritmética de ponto fixo, complemento de dois e número de bits fornecido pelo usuário.

Na simulação em ponto fixo, os coeficientes utilizam o mesmo número de bits com os quais foram quantizados, de forma que o número de bits dos coeficientes não necessariamente é igual ao número de bits utilizado para a representação do sinal. Para cada tipo de filtro os resultados das duas simulações, a de ponto flutuante de dupla precisão e a de ponto fixo, são desenhadas simultaneamente na tela do computador, juntamente com outras duas informações: a relação sinal-ruído (obtida através da comparação dos dois resultados), e o número de "overflows" ocorrido. Esses dois resultados também estarão disponíveis no arquivo de resultados.

Um problema que surge ao se realizar a análise transiente é a ocorrência de "overflow" no registro de saída do filtro. Embora esse registro esteja escalonado pela norma L_∞ (porque $|H(z)| \leq 1$) e, consequentemente, pela norma L_2, ainda existe a possibilidade de ocorrer "overflow" (o que foi verificado na prática). Se permitíssemos que o fator de escalonamento δ também afetasse esse registro, o ganho do filtro teria que ser alterado, o que não é desejável que se faça sem se saber a priori a função deste filtro no sistema onde ele será

inserido. Do ponto de vista da análise transiente, em vez de se diminuir o ganho do filtro, a solução adotada foi a de se diminuir a excursão máxima do sinal de entrada. O fator pelo qual o sinal de entrada será multiplicado para a limitação da excursão é o mesmo que seria usado para o escalonamento do registro de saída, que por sua vez, é calculado pelo mesmo método utilizado para o escalonamento dos registros entre seções. Obviamente, se este fator for maior que 1, o sinal de entrada não deve ser alterado.

5.3 Exemplos de aplicação do programa

Para ilustrar a teoria apresentada nos capítulos anteriores e mostrar o funcionamento do programa, apresentaremos dois exemplos de projeto de filtros digitais: um filtro passa-faixa e um filtro passa-baixa de faixa estreita.

5.3.1 Filtro passa-faixa

Vamos projetar um filtro passa-faixa elíptico com as seguintes especificações:

- frequência de amostragem: 40 KHz;
- atenuação máxima na faixa de passagem: 1,0 dB;
- atenuação mínima na faixa de rejeição: 40 dB;
- frequência limite da faixa de rejeição inferior: 1,5 KHz;
- frequência limite da faixa de passagem inferior: 2,0 KHz;
- frequência limite da faixa de passagem superior: 8,0 KHz;
- frequência limite da faixa de rejeição superior: 8,5 KHz.

Ao invocarmos o programa com o comando FOREST sem nenhum parâmetro adicional, os dados acima serão requisitados, um a um, pelo programa. Após a introdução dos dados, o programa fornece a ordem mínima do protótipo passa-baixa e pede que o usuário entre a ordem desejada:

```
Ordem minima do prototipo passa-baixa = 6
Entre ordem do prototipo passa-baixa:
```

Façamos a ordem igual a 6, o que significa que a ordem do filtro passa-faixa será 12. O programa então calcula os pólos e zeros no plano s, faz a transformação bilinear e monta os termos de segunda ordem no plano z. Logo em seguida, pede que o usuário escolha o fator de escalonamento. Para este exemplo vamos, em princípio, fazer o fator de escalonamento igual a 1.

Se depois for constatado que o escalonamento não foi eficaz, voltaremos e alteraremos esse valor.

Após alguns segundos de cálculo, o programa escreve as seguintes mensagens:

```
Filtro otimo por secao calculado
Ganho de ruido = 4.42134

Forma canonica calculada e escalonada
Ganho de ruido = 18.9216

Filtro otimo por bloco calculado
Ganho de ruido = 4.41035

Registros entre secoes dos filtros otimos escalonados

                              Tecle a barra de espaco
```

Neste caso, os filtros ótimo por seção e ótimo por bloco apresentam desempenhos muito semelhantes, e o filtro com seções na forma direta é pouco mais de 4 vezes pior.

Ao teclarmos a barra de espaço, o programa apresenta o seguinte menu de alternativas:

```
1. Mostrar graficos da forma canonica
2. Mostrar graficos do filtro otimo por secao
3. Mostrar graficos do filtro otimo por bloco
4. Quantizar coeficientes
5. Fazer analise transiente
6. Mudar fator de escalonamento
7. Gravar resultados e sair
Sua escolha:
```

Antes de mais nada, vamos quantizar os coeficientes. Ao escolhermos a opção 4, o programa pede o número de bits com os quais os coeficientes serão representados. Tentemos, primeiramente, 8 bits. Neste caso, o programa escreverá a seguinte mensagem:

```
Forma canonica instavel!
Requantize com mais bits !
                              Tecle a barra de espaco
```

Tentemos novamente, desta vez com 12 bits. Agora o programa escreve:

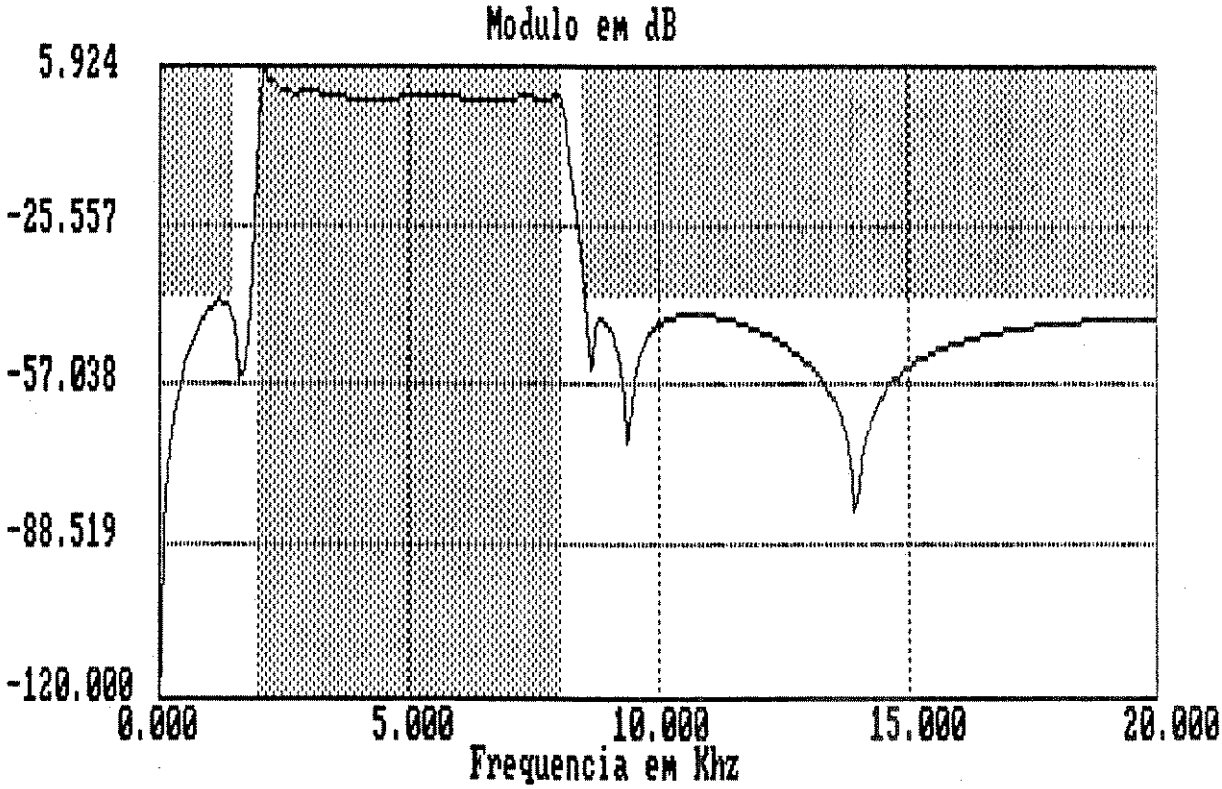

Figura 5.1: Resposta em frequência do filtro passa-faixa com seções na forma direta e quantizado com 12 bits.

```
Quantizacao realizada ok com 12 bits
                              Tecle a barra de espaco
```

O próximo passo é verificar se, para esse número de bits, o filtro ainda obedece às especificações. Escolhendo a opção 1 e pedindo o gráfico do módulo em dB, obtemos a curva de resposta em frequência mostrada na Fig. 5.1. Como se nota nesta figura, o filtro com seções na forma direta, quantizado com 12 bits, não obedece à máscara. Para se verificar o filtro ótimo por bloco, escolhemos a opção 3, obtendo o gráfico mostrado na Fig. 5.2. Agora o filtro obedece à máscara, confirmando uma conclusão tirada anteriormente de que o bom desempenho em relação ao ruído de quantização do sinal implica também em baixa sensibilidade à variação dos coeficientes.

Suponha que pretendamos utilizar o filtro ótimo por bloco e que, portanto, estejamos satisfeitos com a quantização realizada. Partamos, então,

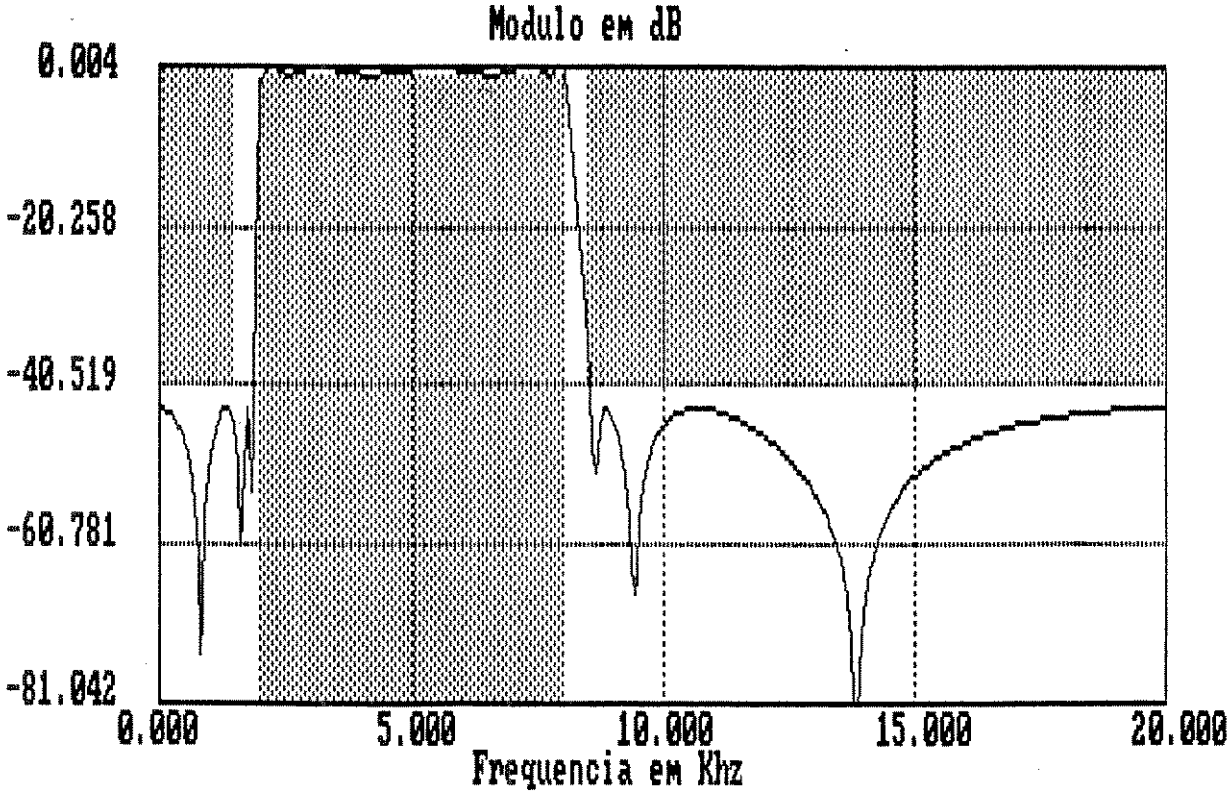

Figura 5.2: Resposta em frequência do filtro passa-faixa ótimo por bloco quantizado com 12 bits.

para a opção 5, e realizemos uma análise transiente no domínio do "tempo". O programa pede inicialmente o número de bits para representação do sinal, que faremos igual a 12, e depois oferece as seguintes opções de sinais de entrada:

```
Sinal de entrada:
1. Ruido branco
2. Senoide
3. Amostra unitaria
4. Degrau unitario
5. Volta ao menu anterior
Sua escolha:
```

Uma vez escolhido o sinal de entrada, o programa faz a análise transiente para os três tipos de filtros e apresenta os resultados em forma gráfica, um por vez, em sequência. O resultado da análise com ruído branco para o filtro ótimo por bloco é mostrado na Fig. 5.3. A relação sinal-ruído é de apenas 0,9 dB porque ocorreram "overflows" nos nós internos, indicando que o escalonamento efetuado não foi eficaz. Embora não estejamos mostrando, o mesmo ocorre com os outros tipos de filtros. Torna-se, então, necessário refazer o projeto com um fator de escalonamento maior. Escolhendo-se a opção 6 do menu de alternativas, o programa pede um novo fator de escalonamento, que agora faremos igual a 2, e o projeto é então refeito.

Ao se realizar a análise transiente do filtro reescalonado, o programa escreve a seguinte mensagem antes de apresentar os gráficos:

```
Para evitar overflow no registro de saida,
o sinal de entrada tera o valor maximo de 0.960451
                          Tecle a barra de espaco
```

O resultado da análise do filtro ótimo por bloco reescalonado, mostrada na Fig. 5.4, indica que não ocorreram "overflows" e que a relação sinal-ruído ficou em 49 dB. Somente para efeito de comparação, a análise transiente do filtro com seções na forma direta resultou numa relação sinal-ruído de 35,9 dB, e a do filtro ótimo por seção, de 44,5 dB. Se a relação sinal-ruído obtida for suficiente para a aplicação à qual o filtro se destina, o projeto estará terminado, podendo-se então gravar os resultados e sair do programa. O arquivo de resultados deste exemplo é apresentado no Apêndice D.

5.3.2 Filtro passa-baixa

No nosso segundo exemplo, mostraremos o projeto de um filtro elíptico passa-baixa de faixa estreita. Este exemplo foi escolhido para mostrar o caso onde a diferença entre o filtro com seções ótimas e o filtro com seções na forma

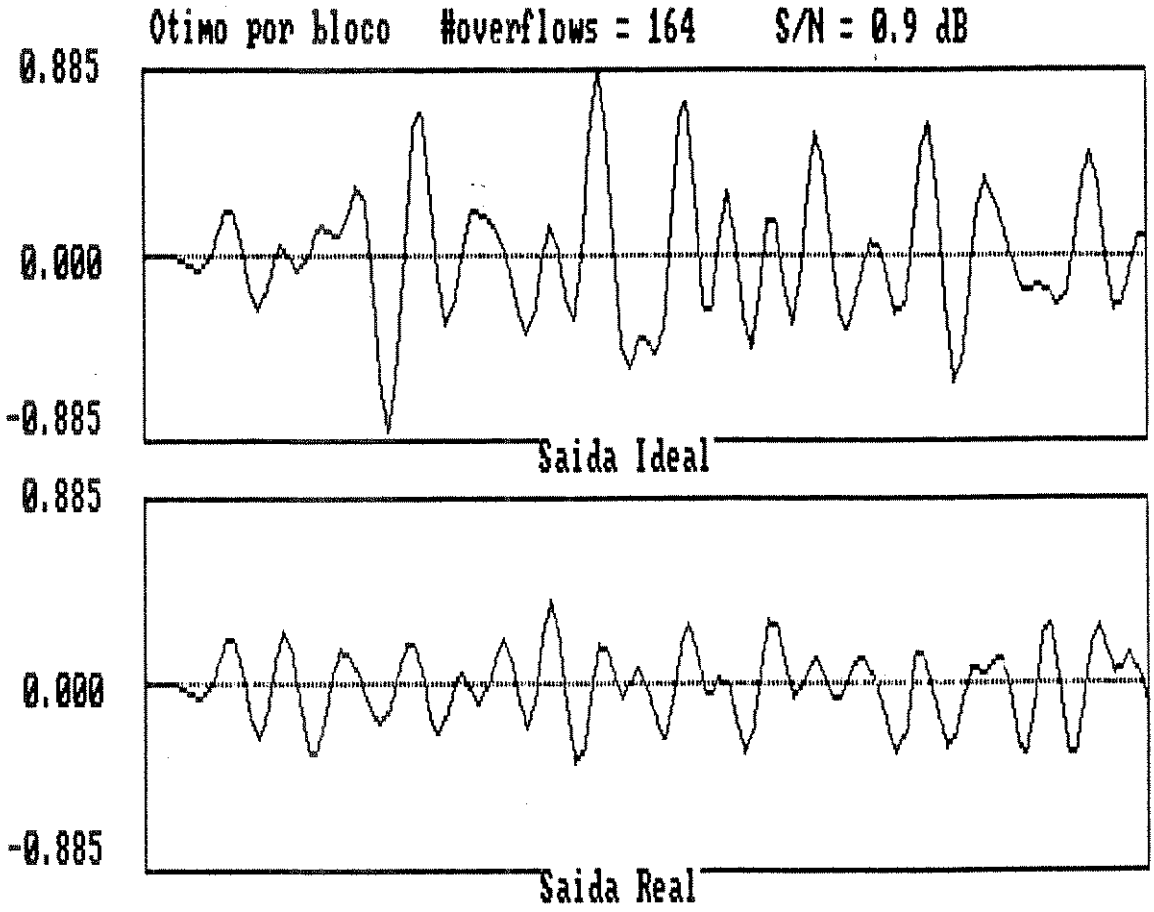

Figura 5.3: Análise transiente do filtro passa-faixa ótimo por bloco com $\delta = 1$ e ruído branco na entrada.

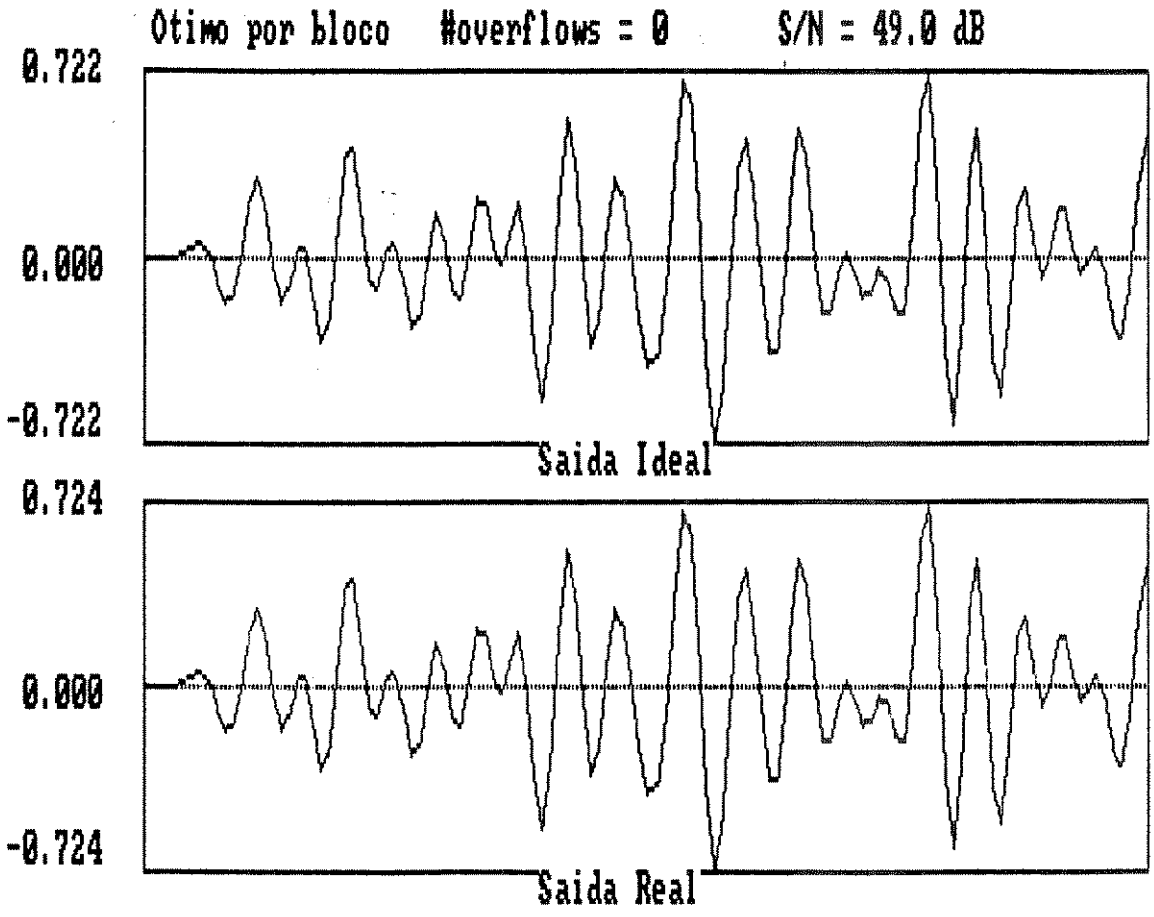

Figura 5.4: Análise transiente do filtro passa-faixa ótimo por bloco com $\delta = 2$ e ruído branco na entrada.

Tipo de filtro	*Ganho de ruído*
cascata de formas diretas	415,7
ótimo por bloco	1,484
ótimo por seção	1,487

Tabela 5.2: Ganhos de ruído do filtro passa-baixa.

direta é mais marcante, ou seja, quando todos os pólos estão aglutinados. As especificações do filtro são:

- frequência de amostragem: 100 KHz;
- atenuação máxima na faixa de passagem: 0,5 dB;
- atenuação mínima na faixa de rejeição: 40 dB;
- frequência limite da faixa de passagem: 1 KHz;
- frequência limite da faixa de rejeição: 1,5 KHz;

Desta vez pouparemos os detalhes de uso do programa, e apresentaremos apenas os resultados finais. O filtro foi projetado com fator de escalonamento igual a 2 e os coeficientes foram quantizados com 16 bits. Os ganhos de ruído dos três tipos de filtros são mostrados na Tabela 5.2, onde se observa que a diferença de desempenho entre os filtros é maior que duas ordens de grandeza.

As curvas de resposta em frequência do filtro com seções na forma direta e do filtro ótimo por bloco são mostradas nas Figs. 5.5 e 5.6, respectivamente. Aqui, como no caso anterior, o filtro com seções na forma direta quantizado não obedece à máscara, mesmo com 16 bits, enquanto que o filtro ótimo por bloco satisfaz à máscara.

Os resultados da análise transiente, com ruído branco na entrada, são mostrados na Tabela 5.3, confirmando a grande diferença à favor dos filtros ótimos. Note que a largura de faixa do filtro é de aproximadamente 1% da frequência de amostragem. Para larguras de faixa menores, a potência de ruído do filtro com seções na forma direta cresce explosivamente.

O Apêndice D traz o arquivo de resultados do projeto deste filtro.

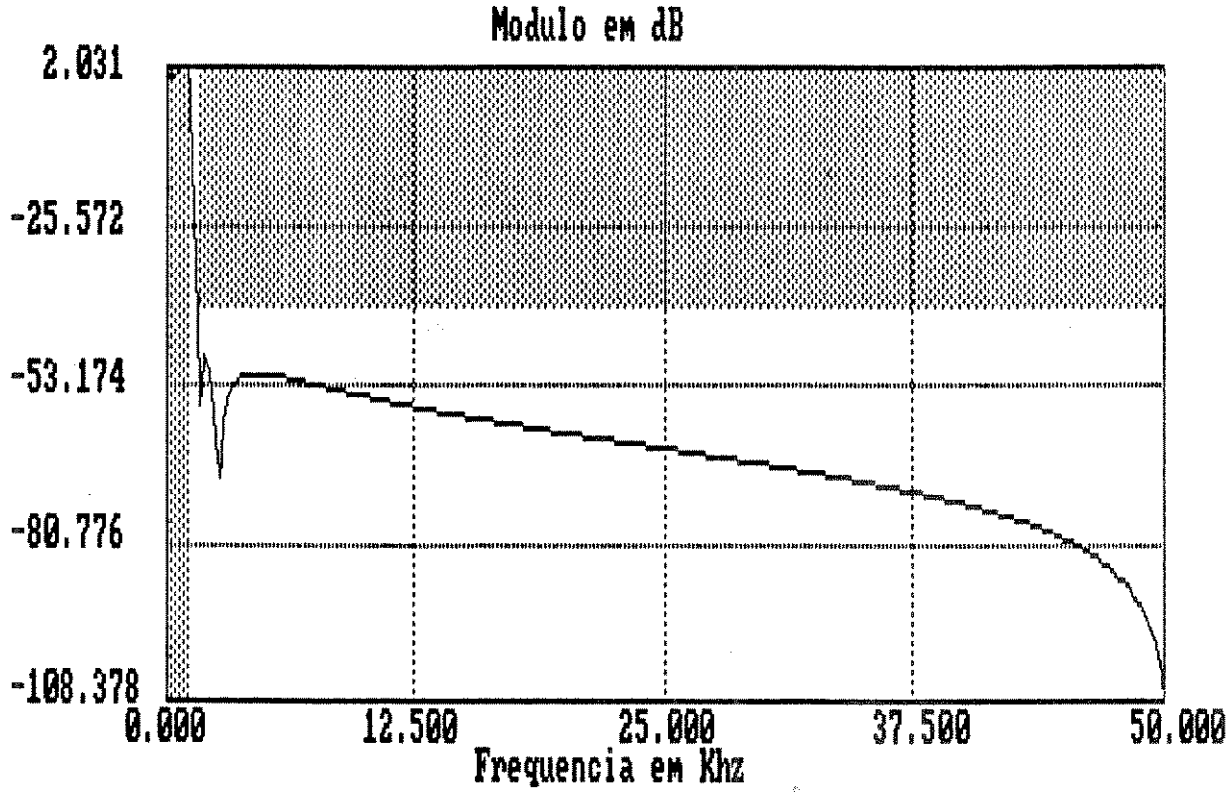

Figura 5.5: Resposta em frequência do filtro passa-baixa com seções na forma direta quantizado com 16 bits.

Tipo de filtro	*S/N (dB)*
cascata de formas diretas	32,6
ótimo por bloco	59,3
ótimo por seção	57,0

Tabela 5.3: Relações sinal-ruído do filtro passa-baixa obtidas na análise transiente.

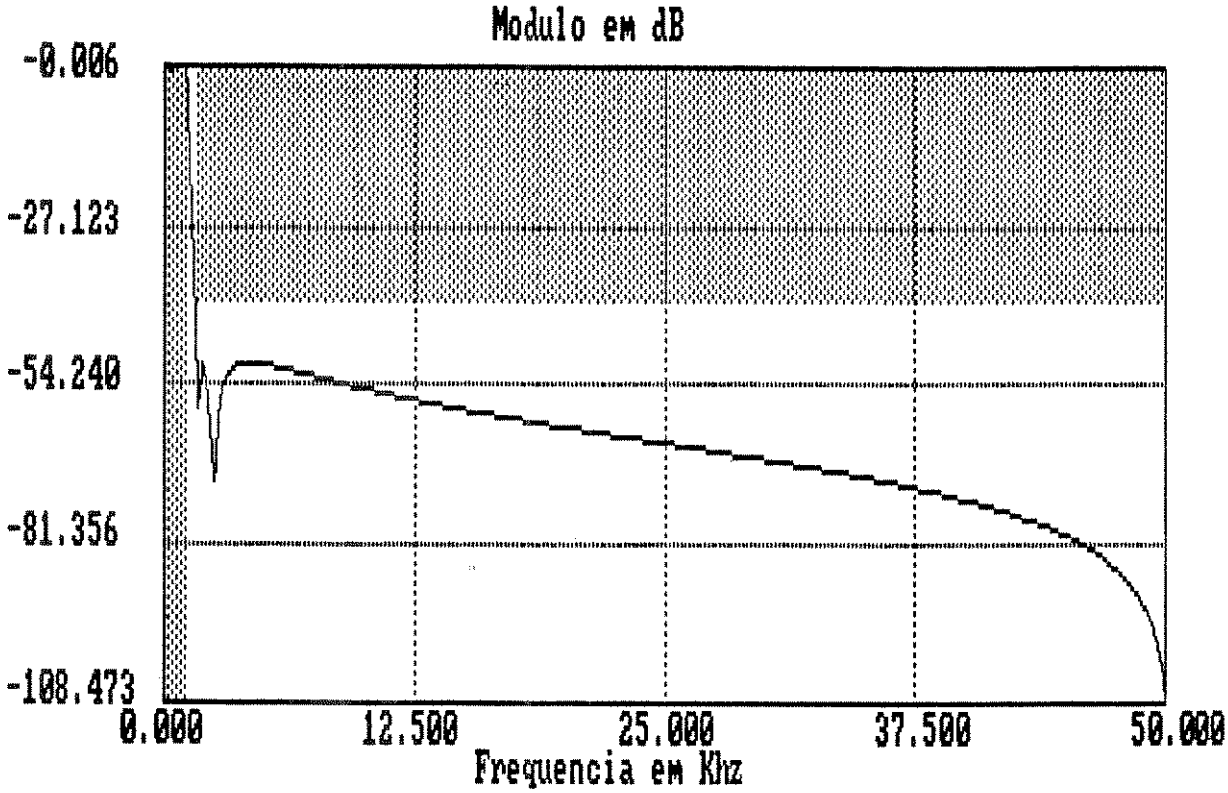

Figura 5.6: Resposta em frequência do filtro passa-baixa ótimo por bloco quantizado com 16 bits.

Capítulo 6

Conclusão

6.1 Retrospectiva

Este trabalho apresentou, em detalhe, toda a teoria necessária ao projeto de um filtro digital recursivo, levando-se em conta os efeitos de comprimento finito dos registros. No Capítulo 2 apresentamos a metodologia de projeto baseada na transformação bilinear, na qual o projeto de um filtro analógico é executado como etapa intermediária do projeto do filtro digital. Foram estudadas as aproximações de Butterworth e Chebychev, cujos cálculos são de natureza essencialmente analítica, e a aproximação elíptica, cujo cálculo envolve algoritmos iterativos e transformações de variáveis para incremento da precisão de cálculo. Assim sendo, o Capítulo 2 condensa de forma lógica e ordenada uma vasta gama de conhecimentos necessários ao projeto de filtros digitais recursivos que, de forma geral, encontra-se muito espalhada na literatura.

Nos Capítulos 3 e 4 estudamos os efeitos da utilização de registros de comprimento finito para a representação de coeficientes e variáveis. Os dois capítulos abordaram o mesmo tema sob enfoques e métodos diferentes. No Capítulo 3 o erro de quantização do sinal foi modelado através de uma fonte de ruído branco aditiva, cuja propagação até a saída é calculada através de integrais de linha de funções de transferência parciais. Foi apresentada, também, uma metodologia para a solução do problema da ocorrência de "overflow" nos registros quando da implementação em ponto fixo, baseada também no cálculo de integrais de linha de funções de transferência. O Capítulo 4 pressupõe o mesmo tipo de modelamento estatístico do erro de quantização do sinal, porém a análise dos seus efeitos é feita através da descrição por variáveis de estado. Esta descrição é de natureza essencialmente matricial, de forma que o cálculo de integrais de linha do método anterior é substituído pela solução de sistemas de equações lineares através de algoritmos matriciais. Mais que isso, a descrição por variáveis de estado

mostrou ter um poder analítico bem superior aos métodos usuais de representação, tornando possível a solução do gigantesco problema da procura de filtros digitais que gerem a menor potência de ruído de quantização possível. Mostramos que existe uma solução teórica desse problema (muito elegante!), que no entanto não é de aplicação prática devido ao grande número de multiplicadores do filtro ótimo obtido no processo. Mostramos também que, impondo-se algumas restrições ao processo de otimização, chega-se a filtros sub-ótimos na forma de cascata de seções de segunda ordem, com um número de multiplicadores um pouco maior (nove contra cinco), mas com desempenho bem superior ao filtro com seções na forma direta. Por desempenho bem superior queremos dizer:

- menor potência de ruído de quantização do sinal;
- potência de ruído de quantização que não depende da largura de faixa;
- ausência de ciclo-limites de grande amplitude à entrada nula provocados por "overflow".

No Capítulo 5 descrevemos o programa de computador FOREST, que implementa a teoria apresentada ao longo dos capítulos anteriores. O programa permite o projeto de filtros digitais recursivos na forma de cascata de seções de segunda ordem operando com aritmética de ponto fixo, com seções na forma direta e na forma ótima por variáveis de estado. Além do projeto em si, o programa também executa análises no domínio do "tempo" e da frequência, permitindo uma avaliação precisa do desempenho do filtro quando de sua implementação física com aritmética de ponto fixo e complemento de dois. Consideramos, portanto, que o programa FOREST representa uma importante contribuição ao Departamento de Comunicações da Faculdade de Engenharia Elétrica da UNICAMP, que agora dispõe do ferramental necessário ao projeto bem sucedido de filtros digitais recursivos.

6.2 Aplicabilidade dos filtros ótimos

Uma das propriedades mais importantes dos filtros ótimos é a invariância da potência de ruído de quantização do sinal com relação à largura de faixa. É esta propriedade que, fundamentalmente, diferencia as formas ótimas das formas diretas, pois para faixa larga o desempenho de ambas as classes de filtros é semelhante. Dito desta forma, poderíamos concluir que os filtros mais problemáticos são os filtros de faixa muito estreita. Isto certamente é verdade para os filtros passa-baixa e passa-alta, mas não necessariamente para os outros casos. Para vermos o porquê disto, devemos relembrar uma conclusão tirada na seção 3.2.2 de que os problemas com relação ao ruído

de quantização[1] são mais severos sempre que os polos do filtro estiverem muito aglutinados. Acontece que, na implementação em cascata, cada seção de segunda ordem incorpora um par de pólos complexos conjugados, e estes pólos estarão sempre bem distanciados entre si, a não ser que eles estejam próximos do eixo real do plano z. Assim, um filtro passa-faixa (ou corta-faixa) será problemático somente se a faixa de passagem (ou rejeição), ou pelo menos um dos lados dela, estiver próxima de zero ou de metade da frequência de amostragem (correspondendo, no plano z, aos pontos $z = 1$ e $z = -1$). Quando isto acontecer, mesmo dentro de uma seção os polos se aproximam muito, e o desempenho dos filtros ótimos passa a ser bem melhor, justificando o seu emprego apesar do número maior de multiplicadores.

6.3 Escolha do fator de escalonamento

Na seção 4.4 apresentamos um gráfico com resultados obtidos por Mullis e Roberts [20] mostrando a potência de ruído de quantização do sinal versus o fator de escalonamento δ. Este gráfico apresenta uma variação abrupta em torno de $\delta = 3$, ponto onde cessaria a ocorrência de "overflow", sugerindo que se use um fator de escalonamento de no mínimo 4. A prática de uso do programa FOREST nos mostrou, entretanto, que a regra acima não precisa ser seguida sempre ao pé da letra. Na verdade, verificamos que o fator de escalonamento mínimo necessário para se evitar "overflow" varia dependendo de fatores como o tipo de filtro, largura de faixa e tipo de sinal de entrada. Ao invés de tentarmos levantar exatamente essa dependência ou estabelecer regras para a fixação do valor de δ, deixamos que o próprio usuário do programa faça essa escolha, já que o programa tem suficiente poder de análise e facilidade de interação para permitir a tentativa de vários valores de δ.

6.4 Sugestões para continuação do trabalho

Daremos aqui algumas sugestões para a continuação e o aperfeiçoamento do trabalho. Estas sugestões não pretendem ser exaustivas, nem estamos afirmando que elas são imperativas. São apenas idéias que surgiram ao longo do trabalho, que não implementamos ainda somente por falta de tempo.

6.4.1 Cálculo das matrizes K e W

Em termos do tempo de processamento, o cálculo das matrizes **K** e **W** através do algoritmo do Apêndice C é o mais demorado. Por exemplo, para

[1] Na verdade a argumentação da seção 3.2.2 foi feita em termos da quantização dos coeficientes. No entanto, como concluímos mais tarde na seção 4.6.3, o desempenho em relação à quantização dos coeficientes e em relação à quantização do sinal são fortemente correlacionados.

um filtro de ordem 12, o cálculo da matriz **K** leva aproximadamente 10 segundos num IBM-PC de 8 MHz com co-processador de ponto flutuante. Para um filtro de ordem 26, esse tempo passa para pouco mais de um minuto e meio. Esses tempos poderiam ser diminuídos se utilizássemos um algoritmo mais eficiente para o cálculo de **K** e **W**. Poder-se-ia, por exemplo, utilizar o algoritmo criado por Mullis e Roberts [20,22] que, como etapa intermediária, utiliza o algoritmo de Levinson-Durbin [36,37] para a solução de um sistema de equações Toeplitz.

6.4.2 Equalização de fase

Uma das grandes desvantagens dos filtros recursivos é a distorção de fase inerente a esses filtros. Por isso, uma possibilidade que merece ser investigada é a da incorporação ao programa de funções que projetem equalizadores de fase a serem acoplados aos filtros recursivos projetados pelo programa. Não é muito claro se tal estrutura – filtro recursivo mais equalizador de fase – teria alguma vantagem sobre os filtros FIR de fase linear, porém esta seria mais uma alternativa de projeto oferecida ao usuário.

6.4.3 Máscaras arbitrárias

Uma possibilidade interessante é a inclusão de facilidades para geração de máscaras arbitrárias. O usuário poderia especificar uma resposta em frequência arbitrária, e algum algoritmo iterativo tentaria sintetizar uma função de transferência $H(z)$ que aproximasse a resposta desejada. Outra possibilidade, mais fácil de incluir no programa, é permitir que o usuário forneça a função de transferência $H(z)$ diretamente, deixando que o programa realize as operações de escalonamento, geração das estruturas ótimas e as análises no "tempo" e em frequência.

6.4.4 Implementação física

Para concluir, deixaremos a sugestão de um programa (que pode se tornar parte do programa FOREST) que auxilie o usuário na implementação física do filtro. Tal programa poderia ser específico para um dado tipo de implementação, ou então oferecer alternativas de implementação, tais como:

- programa de um processador digital de sinais;
- circuito lógico para implementação em placas de circuito impresso;
- circuito lógico para implementação em circuitos integrados dedicados.

Acreditamos que a primeira e a terceira alternativas sejam as mais promissoras em termos de tendência tecnológica e viabilidade econômica. A

primeira alternativa basicamente consiste em se implementar um programa montador de código baseado no conjunto de instruções de um processador de sinais específico, como por exemplo, o TMS320 da Texas Instruments. A terceira alternativa oferece um leque de possibilidades maior, já que no âmbito dos circuitos integrados dedicados existe uma liberdade muito grande em termos de escolha de arquiteturas. Essa escolha normalmente envolve uma solução de compromisso entre paralelismo de processamento e área de silício utilizada, baseando-se em critérios de velocidade de processamento e custo do componente. O que nós propomos para a implementação do programa é a escolha de uma arquitetura específica, a qual satisfaria a maior parte das aplicações, e em cima desta arquitetura, a geração da descrição lógica de um circuito que implementa o filtro. Sugerimos que a linguagem de descrição lógica seja o EDIF [38] ou o VHDL [39], que tendem a ser padronizados pelos compiladores de silício[2] à venda no mercado. Desta forma, a saída do programa poderia ser alimentada na entrada de um compilador de silício, tornando todo o processo de projeto do filtro – das especificações ao layout do circuito integrado – totalmente automático.

[2]Compiladores de silício são programas que, a grosso modo, aceitam como entrada uma descrição lógica fornecida pelo usuário, através de uma entrada esquemática ou textual, e fornecem como resultado o "layout" do circuito integrado dedicado.

Bibliografia

[1] P. E. Allen and E. Sánchez-Sinencio, *Switched Capacitor Circuits*, Van Nostrand Reinhold Company Inc., New York, 1984.

[2] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, N.J., 1975.

[3] R. W. Daniels, *Approximation Methods for Electronic Filter Design*, McGraw-Hill Book Company, New York, 1974.

[4] C. P. Serra, *Teoria e Projeto de Filtros*, Cartgraf Editora Ltda., Campinas, S.P., 1983.

[5] M. Abramowitz e I. A. Stegun, *Handbook of Mathematical Functions*, Dover Publications, INC., New York, 1972.

[6] J. Wiegand, "Digital Signal Processing Enters the Mainstream", *EDN*, pp. 111-118, aug. 6, 1987.

[7] B. Liu, "Effect of Finite Word Length on the Accuracy of Digital Filters - A Review", *IEEE Trans. Circuit Theory*, vol. CT-18, pp. 670-677, nov. 1971.

[8] A. V. Oppenheim and C. J. Weinstein, "Effects of Finite Register Length in Digital Filtering and the Fast Fourier Transform",*Proceedings of the IEEE*, vol. 60, no. 8, pp. 957-976, ago. 1972.

[9] W. R. Bennet, "Spectra of Quantized Signals", *Bell System Tech. Journal*, vol. 27, pp. 446-472, 1948.

[10] B. Widrow, "A Study of Rough Amplitude Quantization by Means of Nyquist Sampling Theory", *IRE Trans. Circuit Theory*, vol. CT-3, pp. 266-276, dec. 1956.

[11] B. Widrow, "Statistical Analysis of Amplitude-Quantized Sampled-Data Systems", *AIEE Trans. (Appl. Indust.)*, vol. 81, pp. 555-568, jan. 1961.

[12] J. B. Knowles and E. M. Olcayto, "Coefficient Accuracy and Digital Filter Response", *IEEE Trans. Circuit Theory*, vol. CT-15, pp. 31-41, mar. 1968.

[13] N. K. Bose, *Digital Filters - Theory and Applications*, Elsevier Publishing Co., New York, 1985.

[14] J. F. Kaiser, "Some Practical Considerations in the Realization of Linear Digital Filters", *Proc. 3rd Annual Allerton Conf. on Circuit and System Theory*, pp. 621-633, 1965.

[15] A. Sekey, "Finite Word Length Effects in Cascade Realization of Comb Filters", *AEU*, Band 29, pp. 257-265, 1975.

[16] L. B. Jackson, "On the Interaction of Roundoff-Noise and Dynamic Range in Digital Filters", *Bell System Tech. Journal*, vol. 49, no. 2, pp. 159-184, feb. 1970.

[17] L. B. Jackson, "Roundoff-Noise Analysis for Fixed-Point Digital Filters Realized in Cascade or Parallel Form", *IEEE Trans. on Audio and Electroacoustics*, vol. AU-18, no. 2, pp. 107-122, jun. 1970.

[18] L. B. Jackson, A. G. Lindgren and Y. Kim, "Optimal Synthesis of Second-Order State-Space Structures for Digital Filters", *IEEE Trans. on Circuits and Systems*, vol. CAS-26, no. 3, pp. 149-153, mar. 1979.

[19] S. Y. Hwang, "On Optimization of Cascade Fixed-Point Digital Filters", *IEEE Trans. on Circuits and Systems*, pp. 163-166, jan. 1974.

[20] C. T. Mullis and R. A. Roberts, "Synthesis of Minimum Roundoff Noise Fixed Point Digital Filters", *IEEE Trans. on Circuits and Systems*, vol. CAS-23, no. 9, pp. 551-562, sep. 1976.

[21] C. T. Mullis and R. A. Roberts, "Roundoff Noise in Digital Filters: Frequency Transformations and Invariants", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-24, no. 6, pp. 538-550, dec. 1976.

[22] R. A. Roberts and C. T. Mullis, *Digital Signal Processing*, Addison-Wesley Publishing Co., 1987.

[23] S. Y. Hwang, "Minimum Uncorrelated Unit Noise in State-Space Digital Filtering", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-25, no. 4, pp. 273-281, aug. 1977.

[24] P. P. Vaidyanathan and S. K. Mitra, "Low Passband Sensitivity Digital Filters: A Generalized Viewpoint and Synthesis Procedures", *Proceedings of the IEEE*, vol. 72, no. 4, pp. 404-423, apr. 1984.

[25] W. L. Mills, C. T. Mullis and R. A. Roberts, "Low Roundoff Noise and Normal Realizations of Fixed Point IIR Digital Filters", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-29, no. 4, pp. 893-903, aug. 1981.

[26] G. Amit and U. Shaked, "Small Roundoff Noise Realization of Fixed-Point Digital Filters and Controllers", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 36, no. 6, pp. 880-891, jun. 1988.

[27] V. Tavsanoglu, "The Necessary and Sufficient Conditions for Minimum Roundoff-Noise in Second-Order State-Space Digital Filters and their Optimal Synthesis", *IEEE Trans. on Circuits and Systems*, vol. CAS-34, no. 6, pp. 669-677, jun. 1987.

[28] T. Saramaki, T. Yu and S. K. Mitra, "Very Low Sensitivity Realization of IIR Digital Filters Using a Cascade of Complex All-Pass Structures", *IEEE Trans. on Circuits and Systems*, vol. CAS-34, no. 8, pp. 876-886, aug. 1987.

[29] P. Burrascano, G. Martinelli and G. Orlandi, "Low-Sensitivity Digital Filters Based on Zero Extraction", *IEEE Trans. on Circuits and Systems*, vol. CAS-34, no. 12, pp. 1581-1587, dec. 1987.

[30] B. W. Bomar, "New Second-Order State-Space Structures for Realizing Low Roundoff Noise Digital Filters", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-33, no. 1, pp. 106-110, feb. 1985.

[31] M. Kawamata and T. Higuchi, "On the Absence of Limit Cycles in a Class of State-Space Digital Filters Which Contains Minimum Noise Realizations", *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-32, no. 4, pp. 928-930, aug. 1984.

[32] M. R. Spiegel, *Manual de Fórmulas e Tabelas Matemáticas*, McGraw-Hill do Brasil Ltda., 1973.

[33] B. Noble, *Applied Linear Algebra*, Prentice-Hall, Englewood Cliffs, NJ, 1969.

[34] C. M. Rader and B. Gold, "Effects of Parameter Quantization on the Poles of a Digital Filter", *Proc. IEEE*, vol. 55, pp. 688-689, may 1967.

[35] P. Albrecht, *Análise Numérica – um curso moderno*, Livros Técnicos e Científicos Editora S.A., Rio de Janeiro, 1973.

[36] N. Levinson, "The Wiener RMS Error Criterion in Filter Design and Prediction",*J. Math. Phys.*, vol. 25, no. 4, pp. 261-278, 1947.

[37] J. Durbin, "The Fitting of Time Series Models", *Rev. Instrum. Int. Statist.*, vol. 28, no. 3, pp. 233-243, 1960.

[38] Electronic Design Interchange Format Steering Committee, *EDIF Specification Version 1 1 0*, 1985.

[39] M. Shahdad et al. ,"VHSIC Hardware Description Language", *Computer*, vol. 18, no. 2, pp. 94-102, 1985.

Apêndice A

Cálculo das funções elípticas

A.1 Integral elíptica completa de primeira espécie

A integral elíptica de primeira espécie é definida por:

$$u(\phi, k) = \int_0^{\phi} \frac{dx}{\sqrt{1 - k^2 \text{sen}^2 x}} \tag{A.1}$$

Fazendo-se $\phi = \pi/2$ na expressão acima, define-se a *integral elíptica completa de primeira espécie* como:

$$K(k) = \int_0^{\pi/2} \frac{dx}{\sqrt{1 - k^2 \text{sen}^2 x}} \tag{A.2}$$

O cálculo desta integral pode ser feito através da seguinte série numérica [5]:

$$K(k) = \frac{\pi}{2} \prod_{i=1}^{\infty} (1 + s_i) \tag{A.3}$$

onde:

$$s_0 = k \tag{A.4}$$

$$s_{i+1} = \frac{1 - \sqrt{1 - s_i^2}}{1 + \sqrt{1 - s_i^2}} \tag{A.5}$$

Nesta série, o valor de s_i tende rapidamente a zero. Na prática, trunca-se o cálculo quando s_i for menor que um certo valor de tolerância próximo de zero.

Relacionada à função acima, define-se também a *integral elíptica completa complementar de primeira espécie* como:

$$K'(k) = \int_0^{\pi/2} \frac{dx}{\sqrt{1 - k'^2 \text{sen}^2 x}} \tag{A.6}$$

onde

$$k' = \sqrt{1 - k^2} \tag{A.7}$$

é o *módulo complementar* de k.

A.2 As funções seno e cosseno elíptico

A partir da *integral elíptica de primeira espécie* definem-se:

- seno elíptico: $\mathrm{sn}(u, k) = \mathrm{sen}\phi$
- cosseno elíptico: $\mathrm{cn}(u, k) = \cos\phi$

O cálculo numérico do seno elíptico pode ser realizado através da seguinte fórmula:

$$\mathrm{sn}(u, k) = \frac{2\pi G}{kK(k)} \tag{A.8}$$

onde:

$$G = \sum_{i=1}^{\infty} \frac{y_i}{1 - y_i^2} \mathrm{sen} \frac{(2i-1)u}{2K(k)} \pi \tag{A.9}$$

$$y_i = \exp\left[-\frac{K'(k)}{K(k)} \left(i - \frac{1}{2} \right) \pi \right] \tag{A.10}$$

Note que no cálculo do seno elíptico entram as integrais elípticas descritas no ítem anterior.

Apêndice B

Matrizes ABCD da associação em cascata

As matrizes de estado de uma associação em cascata de filtros de segunda ordem podem ser facilmente obtidas através de uma simples inspeção do grafo. Seja por exemplo, o filtro de sexta ordem implementado através do grafo matricial mostrado na Fig. B.1. O vetor de estados deste filtro é dado por:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix} \tag{B.1}$$

Sendo $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i, D_i$ as matrizes de estado das seções de segunda ordem, e sendo $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, D as matrizes da associação em cascata, é fácil ver que:

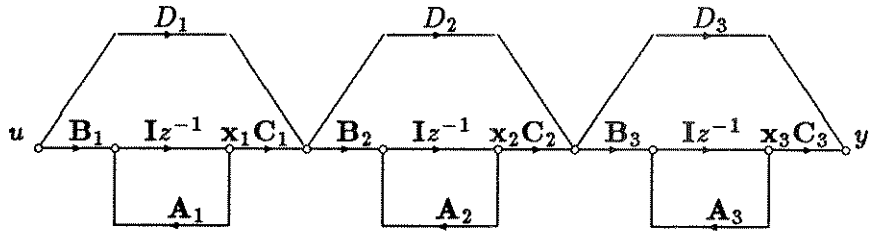

Figura B.1: Grafo matricial de uma associação em cascata de filtros de segunda ordem implementados por variáveis de estado.

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{B}_2\mathbf{C}_1 & \mathbf{A}_2 & \mathbf{0} \\ \mathbf{B}_3 D_2 \mathbf{C}_1 & \mathbf{B}_3\mathbf{C}_2 & \mathbf{A}_3 \end{bmatrix} \tag{B.2}$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 D_1 \\ \mathbf{B}_3 D_2 D_1 \end{bmatrix} \tag{B.3}$$

$$\mathbf{C} = \begin{bmatrix} D_3 D_2 \mathbf{C}_1 & D_3\mathbf{C}_2 & \mathbf{C}_3 \end{bmatrix} \tag{B.4}$$

$$D = D_3 D_2 D_1 \tag{B.5}$$

Generalizando os resultados acima, podemos afirmar que as matrizes de estado da associação em cascata de M seções de segunda ordem serão:

$$\mathbf{A} = \begin{bmatrix} \hat{\mathbf{A}}_{11} & \hat{\mathbf{A}}_{12} & \cdots & \hat{\mathbf{A}}_{1M} \\ \hat{\mathbf{A}}_{21} & \hat{\mathbf{A}}_{22} & \cdots & \hat{\mathbf{A}}_{2M} \\ \vdots & \vdots & & \vdots \\ \hat{\mathbf{A}}_{M1} & \hat{\mathbf{A}}_{M2} & \cdots & \hat{\mathbf{A}}_{MM} \end{bmatrix} \tag{B.6}$$

$$\mathbf{B} = \begin{bmatrix} \hat{\mathbf{B}}_1 \\ \hat{\mathbf{B}}_2 \\ \vdots \\ \hat{\mathbf{B}}_M \end{bmatrix} \tag{B.7}$$

$$\mathbf{C} = \begin{bmatrix} \hat{\mathbf{C}}_1 & \hat{\mathbf{C}}_2 & \cdots & \hat{\mathbf{C}}_M \end{bmatrix} \tag{B.8}$$

$$D = D_M D_{M-1} \cdots D_2 D_1 \tag{B.9}$$

onde

$$\hat{\mathbf{A}}_{ij} = \begin{cases} \mathbf{0}\,, & i < j \\ \mathbf{A}_i\,, & i = j \\ \mathbf{B}_i\mathbf{C}_j\,, & i = j+1 \\ \mathbf{B}_i(D_{i-1}\cdots D_{j+1})\mathbf{C}_j\,, & i > j+1 \end{cases} \tag{B.10}$$

$$\hat{\mathbf{B}}_i = \begin{cases} \mathbf{B}_1\,, & i = 1 \\ \mathbf{B}_i(D_{i-1}D_{i-2}\cdots D_1)\,, & i > 1 \end{cases} \tag{B.11}$$

$$\hat{\mathbf{C}}_i = \begin{cases} D_M D_{M-1}\cdots D_{i+1}\mathbf{C}_i\,, & i < M \\ \mathbf{C}_M\,, & i = M \end{cases} \tag{B.12}$$

e $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i, D_i$ são as matrizes de estado da i-ésima seção.

Apêndice C

Cálculo das matrizes K e W

Existem vários algoritmos possíveis para o cálculo das matrizes $\mathbf{K}$ e $\mathbf{W}$ [22]. O que apresentaremos aqui não chega a ser o mais eficiente, porém nós o escolhemos devido à sua extrema simplicidade.

Partindo-se das expressões (4.15) e (4.32), e utilizando-se (4.13) e (4.29), podemos escrever:

$$\mathbf{K} = \sum_{n=1}^{\infty} \mathbf{f}(n)\mathbf{f}^T(n) = \sum_{n=0}^{\infty} (\mathbf{A}^n\mathbf{B})(\mathbf{A}^n\mathbf{B})^T \tag{C.1}$$

$$\mathbf{W} = \sum_{n=1}^{\infty} \mathbf{g}^T(n)\mathbf{g}(n) = \sum_{n=0}^{\infty} (\mathbf{C}\mathbf{A}^n)^T(\mathbf{C}\mathbf{A}^n) \tag{C.2}$$

Se o filtro é estável, os módulos dos autovalores da matriz $\mathbf{A}$ são todos menores que 1, valendo então a seguinte propriedade:

$$\lim_{n\to\infty} \mathbf{A}^n = \mathbf{0} \tag{C.3}$$

Baseado neste fato, pode-se implementar um algoritmo muito simples para o cálculo de $\mathbf{K}$ e $\mathbf{W}$ utilizando-se as somatórias acima. O mesmo algoritmo utilizado para $\mathbf{K}$ pode ser utilizado para $\mathbf{W}$, bastando para isso substituir $\mathbf{A}$ por $\mathbf{A}^T$ e $\mathbf{B}$ por $\mathbf{C}^T$. O algoritmo para o cálculo de $\mathbf{K}$ é:

- inicialize:

$$\mathbf{F} \leftarrow \mathbf{A} \;\; ; \;\; \mathbf{K} = \mathbf{B}\mathbf{B}^T$$

- repita o laço:

$$\begin{cases} \mathbf{K} \leftarrow \mathbf{F}\mathbf{K}\mathbf{F}^T + \mathbf{K} \\ \mathbf{F} \leftarrow \mathbf{F}^2 \end{cases}$$

- até que:

$$\mathbf{F} \approx \mathbf{0}$$

Apêndice D

Arquivos de resultados

D.1 Introdução

Este apêndice traz os arquivos de resultados dos exemplos de projeto apresentados no Capítulo 5: do filtro passa-faixa e do filtro passa-baixa.

D.2 Filtro passa-faixa

```
************************************************************
FILTRO DIGITAL IIR TIPO PASSA-FAIXA COM APROXIMACAO ELIPTICA
************************************************************

Data: Mon Aug 06 13:26:36 1990


*** DADOS DE ENTRADA ***

.eli
.pf
.fa 40
.amax 1
.amin 40
.f 1.5 2 8 8.5


*** RESULTADOS ***

Ordem do filtro: 12

Polos no plano z:
```

```
0.9458673903966585 +- j0.3074510327700565
0.3043986228410504 +- j0.9351289242862184
0.9152410010801084 +- j0.3261426012884506
0.3603538161813554 +- j0.8509549228403885
0.798642374812454 +- j0.4019118479921469
0.5357549274245197 +- j0.630889850521692

Zeros no plano z:
Numero de zeros em z=1 : 0
Numero de zeros em z=-1 : 0
Zeros complexos:
0.9591402961872709 +- j0.2829308965627372
0.223303168069043 +- j0.9747490421284489
0.9681835699536033 +- j0.2502410335494479
0.09941703726253381 +- j0.9950458545725116
0.992444704851314 +- j0.1226927374076738
-0.5547701756573964 +- j0.8320036371321111


Fator de escalonamento: 2
Numero de bits dos coeficientes (incluindo o bit de sinal): 12
Numero de bits da parte inteira (bit de sinal incluido):
  - forma direta: 5
  - otimo por secao: 2
  - otimo por bloco: 1

Coeficientes:

Forma direta:
Ganho de ruido: 18.9216
Coeficiente na entrada da primeira secao : 0.140625
Secao #1
   b0 = 0.2421875
   b1 = -0.484375
   b2 = 0.2421875
   c1 = 1.59375
   c2 = -0.796875
Secao #2
   b0 = 0.9921875
   b1 = -1.90625
   b2 = 0.9921875
   c1 = 1.890625
   c2 = -0.9921875
```

```
Secao #3
   b0 = 4.546875
   b1 = -8.796875
   b2 = 4.546875
   c1 = 1.828125
   c2 = -0.9453125
Secao #4
   b0 = 0.421875
   b1 = -0.0859375
   b2 = 0.421875
   c1 = 0.71875
   c2 = -0.8515625
Secao #5
   b0 = 0.15625
   b1 = 0.1796875
   b2 = 0.15625
   c1 = 1.0703125
   c2 = -0.6875
Secao #6
   b0 = 1.96875
   b1 = -0.875
   b2 = 1.96875
   c1 = 0.609375
   c2 = -0.96875

Otimo por bloco:
Ganho de ruido: 4.41035
Secao #1
   A =        0.79931640625          0.4375
               -0.369140625       0.7978515625
   B =         -0.287109375
              0.15185546875
   C =         0.2333984375      -0.5419921875
   D =        0.38525390625
Secao #2
   A =        0.93408203125        0.291015625
              -0.3251953125       0.95751953125
   B =       -0.22607421875
              0.1376953125
   C =         0.0009765625      -0.18017578125
   D =          0.943359375
Secao #3
   A =           0.91796875       0.30712890625
```

```
                -0.3466796875        0.91259765625
   B =          -0.24853515625
                0.08447265625
   C =             0.19140625        -0.435546875
   D =          0.79638671875
Secao #4
   A =             0.353515625        0.904296875
                   -0.80078125          0.3671875
   B =          -0.5712890625
                 0.2373046875
   C =          -0.29638671875        0.5634765625
   D =          0.58056640625
Secao #5
   A =             0.533203125        0.59423828125
                   -0.669921875       0.53857421875
   B =          -0.5361328125
                   0.115234375
   C =          -0.46826171875        0.72412109375
   D =           0.1533203125
Secao #6
   A =          0.24072265625          0.951171875
                   -0.923828125       0.3681640625
   B =          0.43896484375
                0.03369140625
   C =             0.302734375        -0.1572265625
   D =          0.78662109375

Otimo por secao:
Ganho de ruido: 4.42134
Secao #1
   A =             0.798828125          -0.4296875
                  0.3759765625         0.798828125
   B =             0.291015625
                  0.1416015625
   C =          -0.2568359375         -0.5263671875
   D =             0.384765625
Secao #2
   A =          0.9462890625           -0.30859375
                   0.306640625       0.9462890625
   B =          0.0947265625
                   0.025390625
   C =          -0.1318359375          -0.498046875
   D =             0.943359375
```

```
Secao #3
   A =        0.9150390625        -0.333984375
              0.318359375          0.9150390625
   B =        0.208984375
              0.07421875
   C =       -0.2021484375        -0.5654296875
   D =        0.7958984375
Secao #4
   A =        0.3603515625        -0.8310546875
              0.87109375           0.3603515625
   B =        0.3212890625
              0.1640625
   C =        0.4716796875         0.923828125
   D =        0.5810546875
Secao #5
   A =        0.5361328125        -0.5908203125
              0.673828125          0.5361328125
   B =        0.4423828125
              0.185546875
   C =        0.3779296875           0.90234375
   D =        0.1533203125
Secao #6
   A =          0.3046875         -0.931640625
              0.9384765625           0.3046875
   B =          0.10546875
              0.0634765625
   C =        0.6044921875           1.01171875
   D =          0.787109375

Resultados da analise transiente com 12 bits
e amplitude maxima do sinal de entrada limitada a 0.960451

Ruido branco:

            Forma direta     Ot. secao     Ot. bloco
    S/N         35.9            44.5          49.0
#overflows       0               0             0
```

D.3 Filtro passa-baixa

```
************************************************************
FILTRO DIGITAL IIR TIPO PASSA-BAIXAS COM APROXIMACAO ELIPTICA
************************************************************

Data: Mon Aug 06 13:28:46 1990


*** DADOS DE ENTRADA ***

.eli
.pb
.fa 100
.amax 0.5
.amin 40
.f 1 1.5


*** RESULTADOS ***

Ordem do filtro: 5

Polos no plano z:
0.9928668150876638 +- j0.06325048533121809
0.981287224584105 +- j0.04340032689553416
0.9735849307768963

Zeros no plano z:
Numero de zeros em z=1 : 0
Numero de zeros em z=-1 : 1
Zeros complexos:
0.9952164765679931 +- j0.09769424122019235
0.9893060702866517 +- j0.1458543770134525


Fator de escalonamento: 2
Numero de bits dos coeficientes (incluindo o bit de sinal): 16
Numero de bits da parte inteira (bit de sinal incluido):
  - forma direta: 5
  - otimo por secao: 2
  - otimo por bloco: 1
```

```
Coeficientes:

Forma direta:
Ganho de ruido: 415.729
Coeficiente na entrada da primeira secao : 0.00634765625
Secao #1
   b0 = 0.09814453125
   b1 = -0.19384765625
   b2 = 0.09814453125
   c1 = 1.96240234375
   c2 = -0.96484375
Secao #2
   b0 = 6.72900390625
   b1 = -13.3935546875
   b2 = 6.72900390625
   c1 = 1.98583984375
   c2 = -0.98974609375
Secao #3
   b0 = 0.14111328125
   b1 = 0.14111328125
   b2 = 0
   c1 = 0.9736328125
   c2 = 0

Otimo por bloco:
Ganho de ruido: 1.48434
Secao #1
   A =   0.981964111328125   0.038909912109375
        -0.048431396484375     0.9805908203125
   B =  -0.125396728515625
         0.01043701171875
   C =    0.1075439453125   0.871612548828125
   D =   0.273468017578125
Secao #2
   A =   0.992034912109375    0.07354736328125
        -0.054412841796875    0.99371337890625
   B =  -0.041534423828125
        -0.0025634765625
   C =  -0.012542724609375         0.7919921875
   D =     0.3226318359375
Secao #3
   A =    0.97357177734375                    0
                         0                    0
```

```
   B =              0.046875
                           0
   C =    0.27874755859375                    0
   D =   0.006622314453125

Otimo por secao:
Ganho de ruido: 1.48724
Secao #1
   A =    0.98126220703125   -0.05145263671875
             0.03662109375    0.98126220703125
   B =    0.00262451171875
            0.1226806640625
   C =   -0.84307861328125   -0.01788330078125
   D =            0.2734375
Secao #2
   A =    0.99285888671875   -0.06427001953125
             0.062255859375    0.99285888671875
   B =      0.000732421875
            0.027099609375
   C =   -1.02996826171875   -0.02801513671875
   D =     0.3226318359375
Secao #3
   A =    0.97357177734375                    0
                           0                   0
   B =     0.0572509765625
                           0
   C =    0.22833251953125                    0
   D =      0.006591796875


Resultados da analise transiente com 16 bits
e amplitude maxima do sinal de entrada limitada a 0.999969

Ruido branco:

            Forma direta     Ot. secao     Ot. bloco
    S/N             32.6          57.0          59.3
#overflows             0             0             0
```