UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

DEPARTAMENTO DE ENGENHARIA DE SISTEMAS

# Programação Linear para Aproximação de Funções aplicada ao Projeto de Filtros Digitais

Autora: **Jeanne Dobgenski**

Orientador: **Christiano Lyra Filho**

Dissertação submetida à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, para preenchimento dos pré-requisitos parciais para obtenção do Título de Mestre em Engenharia Elétrica .

dezembro 1997

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Dobgenski, Jeanne
D653p Programação linear para aproximação de funções aplicada ao projeto de filtros digitais. / Jeanne Dobgenski.--Campinas, SP: [s.n.], 1997.

Orientador: Christiano Lyra Filho
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Programação linear. 2. Filtros digitais (Matemática). 3. Telecomunicações. 4. Teoria da aproximação. 5. Funções (Matemática). I. Lyra Filho, Christiano. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

*À minha família*

Este trabalho teve suporte financeiro do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

# Resumo

Este trabalho estuda o problema clássico de aproximação de funções e propõe técnicas de programação linear para resolvê-lo.

O problema de aproximar funções existe em diversas circunstâncias, entre as quais é destacado o problema de projetar filtros digitais, principal aplicação deste trabalho. O projeto de filtros digitais é amplamente empregado em sistemas de telecomunicações (transmissão de sinais elétricos). A resposta em freqüência de um filtro deve ser aproximada, porque a ideal é impossível de ser realizada sob o ponto de vista físico.

O uso de programação linear proporciona flexibilidade na aproximação de funções por permitir a inclusão de restrições especiais sem prejudicar a aplicação do método. O algoritmo desenvolvido usa a idéia central do método Simplex Revisado; aproveita as informações sobre a estrutura do problema, evitando cálculos desnecessários e economizando espaço no armazenamento dos dados. Os resultados alcançados são comparados aos obtidos pelo trabalho realizado por Steiglitz, Parks e Kaiser, em 1992, mostrando as boas qualidades do método desenvolvido.

# Abstract

This work studies the classical problem of function approximation and proposes linear programming solution methods.

Function approsimation problem occurs in many situations, one of which is the digital filter design problem, the main application of this work. Digital filter design is widely used in telecomunication systens (eletrical signal transmission). The frequency response of a filter has to be approximated, because it is impossible to reproduce the responce exactly from the physical point of view.

The use of linear programming provides flexibility in function approximation, since it allows inclusion of special constrains without harming the application of method. The algorithm developed uses the essencial ideia of revised simplex method; it uses information concerning problem structure, avoiding unnecessary calculation and saving data storage space. The results obtained are compared with those of Steiglitz, Parks and Kaiser (1992), demonstrating the good qualities of the method developed.

# Agradecimentos

Foram muitas as pessoas que interferiram, direta ou indiretamente, no desenvolvimento desta pesquisa. Em especial, quero agradecer:

- à Deus pelo presente desta vida;
- ao querido professor Christiano Lyra Filho por aceitar navegar em outras águas, ajudando-me a concluir este trabalho. Também pelas agradáveis conversas e orientações;
- ao professor Amir Said pela orientação inicial desta pesquisa;
- ao professor João M. Romano pelos esclarecimentos sobre filtros digitais e sugestões dadas ao trabalho;
- aos meus pais, Edgard e Ivone, por serem exatamente como são: maravilhosos!! Obrigada pelo amor, carinho, conselhos, amizade, ...
- aos meus irmãos, Jean, Jivago e Juliano, por "cuidar" dos nossos pais durante todo o tempo que estou longe;
- ao Aredis, por estar sempre presente em todos os momentos, pelo amor e carinho;
- à Renata pelas valiosas conversas, pelas orientações e principalmente pela amizade;
- aos amigos do DENSIS, especialmente, ao Walcir e à Luciana pela disposição em ajudar; à Débora, Fran, Regina, Cíntia, Vitória, ao Hamilton e Leonardo pelo apoio;
- ao CNPq pela ajuda finaceiro.

# Conteúdo

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

O processo de aproximar uma função, $f(x)$, tem importância quando $f(x)$ possui uma expressão que operações como diferenciação e integração são difíceis (ou mesmo impossíveis) de serem realizadas [7]. Uma aplicação importante de aproximação de funções é o projeto de filtros digitais. Este problema consiste em encontrar uma boa aproximação para a função da resposta em freqüência, que deve atender às especificações do filtro desejado — somente após resolver esta etapa é que o filtro pode ser construído fisicamente.

Este trabalho mostra alguns métodos clássicos para aproximar funções, e destaca a possibilidade de encontrar aproximações através de técnicas de programação linear. A eficácia do método Simplex foi bastante discutida e comprovada em vários problemas clássicos da otimização linear. Esta pesquisa propõe o emprego do Simplex Revisado para encontrar a melhor aproximação da resposta em freqüência, que respeite as especificações estabelecidas para o filtro a ser construído.

Muitas são as técnicas usadas para aproximar funções. Especialmente no caso dos filtros digitais, diversos autores usam os polinômios de Chebyshev, acompanhados do algoritmo de trocas de Remez. O emprego de programação linear nesta área tem sido divulgado desde o início da década de 70 através de breves citações; estudos que utilizam PL para formular e resolver esse problema apareceram em seguida. Outros trabalhos na mesma linha incluem restrições adicionais ao problema.

O algoritmo construído neste trabalho visa projetar filtros digitais não recursivos do tipo passa-baixa com fase linear. Os resultados encontrados por este algoritmo são comparados com os obtidos pelo programa Meteor, idealizado por Steiglitz, Parks e Kaiser [43] em 1992.

O próximo capítulo discute o problema de aproximar funções, citando alguns métodos clássicos utilizados para sua solução. Analisa as possíveis formas de medir o erro, entre a função original e a de aproximação, e os diferentes problemas gerados através de cada medida de erro.

O terceiro capítulo discute, brevemente, os principais conceitos associados ao projeto de filtros digitais, interpreta o problema sob o âmbito de aproximação de funções e faz rápida revisão bibliográfica.

A utilização do Simplex Revisado em aproximação de funções é o tema do quarto capítulo, que descreve sumariamente a programação linear e apresenta o método Simplex Revisado. Mostra também a formulação matemática da aproximação da resposta em freqüência de um filtro digital, e considera as propriedades da matriz de restrições do sistema.

O quinto capítulo descreve o algoritmo desenvolvido (SemIntervalo) e suas variantes, mostra os resultados obtidos em cada interpretação do problema (com relação ao intervalo de trabalho). Soluções de SemIntervalo e de Meteor são apresentadas e comparadas.

O sexto capítulo é dedicado às conclusões.

# Capítulo 2

# Aproximação de Funções

O problema de aproximar uma função surge em diversas situações. As funções matemáticas estão presentes em diversas áreas [3]. Na biologia, por exemplo, uma função pode representar o crescimento populacional de uma colônia de bactérias; na medicina, mostrando o comportamento de um tumor submetido à algum tipo de tratamento; na economia, como a taxa de variação de moedas; em censos demográficos, feitos a cada decênio.

Muitas dessas funções são definidas por expressões intratáveis. Os métodos de aproximação são destinados aos problemas dessa natureza, pois visam construir uma função simples muito próxima da função original, difícil de ser calculada. Espera-se que o resultado da função aproximada tenha um desvio mínimo, de modo que não haja influência negativa sobre o problema tratado.

Neste capítulo, será apresentada uma descrição geral do problema de aproximação de funções. Na notação usada, os vetores são representados por letras minúsculas e as matrizes por maiúsculas, ambos em negrito.

Na próxima seção é dada uma visão geral do problema. A seguir são descritos os métodos clássicos de aproximações de funções. A parte final do capítulo trata das análises das medidas de erros no caso contínuo e no discreto, exemplificando os problemas que surgem em cada situação.

## 2.1 Problema de Aproximação de Funções

O problema geral consiste em encontrar uma função simples $\bar{g}_{\mathbf{a}}(\bar{x})$ que melhor aproxime uma dada função $\bar{f}(\bar{x})$, em geral complicada, com $\bar{x} \in [\alpha, \beta]$ — $\mathbf{a}$ é o vetor que contém os parâmetros da aproximação. Considera-se que as funções sejam definidas sobre o espaço $C$ das funções contínuas $\bar{f}(\bar{x})$ de valores reais ou complexos sobre o intervalo $[\alpha, \beta]$ .

Sem perda de generalidade, pode-se transformar as variáveis para um novo intervalo. Fazendo

$$\bar{x} = \frac{\alpha + \beta}{2} - \frac{\alpha - \beta}{2} x \tag{2.1}$$

a restrição $x \in [-1, 1]$. Então é estabelecida a nova representação das funções por:

$$f(x) = \bar{f}(\bar{x}) \text{ e } g_{\mathbf{a}}(x) = \bar{g}_{\mathbf{a}}(\bar{x}), \tag{2.2}$$

sendo válidas as aplicações:

$$g_{\mathbf{a}} : C[-1,1] \longrightarrow C[-1,1], \quad f : C[-1,1] \longrightarrow C[-1,1]. \tag{2.3}$$

Na teoria que estuda o problema de aproximação de funções, encontram-se várias possibilidades para a definição da função especial $g_{\mathbf{a}}(x)$. Entre essas, encontramos aproximações através de polinômios, funções trigonométricas, funções exponenciais, polinômios de Chebyshev [11, 27] e outros. A seguir, apresentam-se alguns dos métodos clássicos empregados nesse problema.

## 2.2 Métodos clássicos

DeVore [11] divide o problema de aproximação de funções em três classes: aproximação polinomial, aproximação trigonométrica e aproximação através de splines. Segundo ele, qualquer aproximação pode ser enquadrada em uma dessas classes.

Cunha [7] os classifica em dois grupos. O primeiro é chamado de interpolação, onde os dados do problema são precisos e a curva de ajuste coincide com os pontos dados, isto é, não há erro. O segundo leva em conta o erro introduzido na obtenção dos dados, onde o Método dos Mínimos Quadrados desempenha papel fundamental.

Independentemente das diferentes classificações, destaca-se que os polinômios de Chebyshev são ferramentas importantes na aproximação de funções contínuas, e que o algoritmo de Remez é um método numérico para a aproximação linear de Chebyshev. Este método foi bastante empregado, por vários autores, no problema de projeto de filtros digitais, a aplicação principal deste trabalho.

### 2.2.1 Caso Geral

A função de aproximação usada nos problemas que se encontram na forma geral linear, pode ser definida através da equação abaixo:

$$g_{\mathbf{a}}(x) = \sum_{i=0}^{M} a_i r_i(x), \tag{2.4}$$

sendo que $r_i(x)$ denota uma função de interpolação usada em cada tipo de aproximação — em geral, definida a partir de certos requisitos práticos, como a simplicidade de cálculo ou o tipo de modelagem. Assume-se que $M$ será sempre o grau do polinômio ou o número de elementos da série de interpolação. Observe que essa série iniciará sempre em 0 e, portanto, o número de termos neste caso será $M + 1$.

### 2.2.2 Aproximação Polinomial

Deseja-se encontrar um polinômio $g_{\mathbf{a}}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_M x^M$ que forneça o ajuste de uma função $f(x)$ em $N$ pontos do intervalo onde a função está definida. Se $M = N$, pode-se determinar de modo único o ajuste exato; se $M < N$, define-se de algum modo o melhor ajuste [13]. A aproximação é feita por um polinômio de grau $M$ definido como:

$$g_{\mathbf{a}}(x) = \sum_{i=0}^{M} a_i x^i, \qquad x \in [-1, 1]. \tag{2.5}$$

### 2.2.3 Aproximação Trigonométrica

Neste caso, o ajuste dos dados é feito pela seguinte função trigonométrica:

$$g_{\mathbf{a},\mathbf{b}}(x) = \frac{a_0}{2} + \sum_{i=1}^{L} [a_i \cos(i\pi x) + b_i \sin(i\pi x)] \qquad x \in [-1, 1], \tag{2.6}$$

onde $2L + 1$ é o número máximo de elementos da série. Em particular, para qualquer valor de $L$ a equação (2.6) é uma função de período $2/i$. Quando $L = \infty$, essa série trigonométrica se denomina série de Fourier — $a_k$ e $b_k$ são chamados de coeficientes de Fourier; a série converge uniformemente para uma função $f$ [5].

As séries de Fourier desempenham um importante papel no estudo de aproximação de funções. Elas são baseadas na propriedade matemática que diz que toda função periódica pode ser decomposta em uma série de natureza senoidal, guardando entre si relações de amplitude e fase adequadas. Por esse motivo, as séries de Fourier são amplamente empregadas em sistemas de sinais analógicos e digitais (da área de engenharia de telecomunicações).

### 2.2.4 Aproximação através de polinômios de Legendre

A aproximação com o uso de polinômios de Legendre visa mostrar de maneira simples que há muitos modos de aproximar uma função. O polinômio de Legendre é definido [25] pelos parâmetros $\gamma_n(x)$ e $l_n(x)$, a seguir:

$$\gamma_n(x) = \frac{(x^2-1)^n}{2^n n!} \tag{2.7}$$

e

$$l_n(x) = \frac{d^n}{dx^n}\gamma_n(x). \tag{2.8}$$

$l_n$ é um polinômio de Legendre de grau $n$. Da equação (2.8), pode-se mostrar que o coeficiente do termo de grau $n$, $l_n(x)$, é dado por:

$$c_n = \frac{(2n)!}{2^n(n!)^2}. \tag{2.9}$$

A função de aproximação é dada por:

$$g_{\mathbf{a}}(x) = \sum_{i=0}^{M} a_i l_i(x). \tag{2.10}$$

### 2.2.5 Polinômios de Chebyshev

O polinômio de Chebyshev do primeiro tipo, denotado por $T_n(x)$, é dado pela fórmula [10]

$$T_n(x) = \cos(n \arccos x) \quad \text{com} \quad x^2 \leq 1, \tag{2.11}$$

Pode-se verificar que $T_0(x) = 1$, $T_1(x) = x$ e, de uma maneira geral, os termos satisfazem a relação recorrente

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x) \qquad n \geq 1. \tag{2.12}$$

A propriedade (2.12) facilita os cálculos dos polinômios de Chebyshev com grau $n \geq 1$.

Esses polinômios são ortogonais no intervalo $[-1, 1]$ quando associados à função peso não negativa $w(x) = \frac{1}{\sqrt{1-x^2}}$ com $x \in (-1, 1)$, e atingem seu valor máximo quando $T_n(x) = 1$ e mínimo para $T_n(x) = -1$.

É possível mostrar [18] que eles podem ser usados para aproximar uma função contínua em $[-1, 1]$, do modo

$$g_{\mathbf{a}}(x) = \left[\sum_{i=0}^{Q} a_i T_i(x)\right] - \frac{1}{2}\, a_0. \tag{2.13}$$

Vamos supor que $Q$ seja suficientemente grande para que (2.13) seja uma aproximação perfeita para uma função $f(x)$. Considerando-se a aproximação truncada

$$g_{\mathbf{a}}(x) = \left[\sum_{i=0}^{M} a_i T_i(x)\right] - \frac{1}{2}\, a_0, \tag{2.14}$$

que gera um polinômio de grau $M \ll Q$; o erro máximo ($E_{max}$) não excederá a melhor aproximação de grau $M$ [18]. Então, o erro é dominado por $a_M T_M(x)$, ou seja, se $a_M = 0.0001$ com certeza $E_{max} < 0.0001$.

Os polinômios de Chebyshev produzem boas aproximações para funções contínuas e bem comportadas, ou seja, aquelas que podem ser aproximadas, sob um nível aceitável, por um polinômio de baixo grau no intervalo de interesse.

Há métodos que exploram as propriedades de uma série de Chebyshev da forma (2.13). Entre esses o Algoritmo de Remez é um ponto de partida para um processo iterativo.

### 2.2.6 Algoritmo de Remez

O teorema de alternação de Chebyshev, sugere caminhos para computar numericamente a melhor aproximação uniforme $P^*$ para $f \in C[a, b]$ , através de polinômios um sistema de Haar $\Phi = \{\phi_1, \ldots, \phi_n\}$ [11].

O algoritmo consiste em selecionar uma seqüência $T$, constituída de pontos $a \leq t_0 < t_1 < \ldots < t_n \leq b$ que aproximem $P^*$ através do polinômio $P_T = \sum_{j=1}^{n} a_j \phi_j(t_i)$, o qual melhor aproxima $f$ sobre o conjunto $T$. De acordo com o teorema de alternação de Chebyshev [11], $P_T$ pode ser

encontrado através do seguinte sistema de equações:

$$\sum_{j=1}^{n} a_j \phi_j(t_i) + (-1)^i d = f(t_i), \qquad i = 0, 1, \ldots, n \tag{2.15}$$

sendo $d$ e $a_j$, $j = 1, \ldots, n$ as variáveis. Segue que $|d|$ é o erro da melhor aproximação para $f$ sobre o conjunto de pontos $T$.

O teorema de *la Vallée Poussin* [11]:

**Theorem 1 (Teorema de la Vallée Poussin)** *Seja $\Phi$ um sistema de Haar sobre $A = [a, b]$ ou $\mathbf{T}$ (círculo), e seja $B$ um subconjunto fechado de $A$ contendo, no mínimo, $n+1$ pontos. Se para um $\phi$-polinômio $P$, existe um conjunto ordenado de $n+1$ pontos distintos $x_i$ pertencentes a $B$ para o qual $f(x_i) - P(x_i) = \epsilon(-1)^i \mu_i$, com $\epsilon = +-1$ e $\mu > 0$, $i = 0, 1, \ldots, n$ então $E_n(f) \geq \min_i \mu_i$.*

Este teorema assegura que o erro, $E(f)$, para a aproximação da função $f$ sobre o espaço contínuo $A = [a, b]$ satisfaz:

$$|d| \leq E(f) \leq D, \tag{2.16}$$

onde $D = ||f - P_T||(A)$, isto é, $D$ é definido como a norma $||f - P_T||$ sobre o espaço contínuo $A = [a, b]$. A meta é escolher uma seqüência $T$ tal que $\Delta = D - |d|$ seja pequeno. Isto garante que $P_T$ é próximo de $P^*$ .

De fato, pelo seguinte teorema [11],

**Theorem 2 (Newman - Shapiro)** *O operador da melhor aproximação $P$ é fortemente único: para cada $f_0 \in C(A)$ existe uma constante $\gamma > 0$ com a propriedade*

$$||f_0 - Q|| \geq ||f_0 - P|| + \gamma ||Q - P|| \tag{2.17}$$

*para algum polinômio $Q$.*

Substituindo $f_0$ por $P_T$ e $Q$ por $P^*$, tem-se:

$$||P_T - P^*|| \leq \gamma^{-1} [||f - P_T|| - ||f - P^*||] \leq \gamma^{-1} \Delta, \tag{2.18}$$

com a contante $\gamma$ denpendendo de $f$. Observe que $||f - P_T|| = D$ e $||f - P^*|| = E(f)$.

Para o algoritmo de Remez, seqüências de $n+1$ pontos $(T_0, T_1, \ldots)$ e melhores aproximações $(P_0, P_1, \ldots)$ são definidas recursivamente. A seqüência $T_{k+1}$ é determinada pela alteração de algum ponto em $T_k$, cuja mudança aumenta o tamanho de $|d_k|$. Considerando o algoritmo de troca simples, somente um ponto é mudado em cada passo.

Suponha $T_{k+1}$ a seqüência atualizada a partir de $T_k$. Seja $\gamma$ um ponto, a ser determinado numericamente, onde $|f(\gamma) - P_T(\gamma)| = D$. A seqüência $T_{k+1}$ é obtida de $T_k$ substituindo um dos pontos $t_s$ por $\gamma$ e sem alterar os outros pontos. $s$ é escolhido de tal forma que $f - P_T$ alterna em sinal nos pontos da nova seqüência $T_{k+1}$.

A prova da convergência e maiores detalhes do algoritmo se encontram em [11, 27].

### 2.2.7 Spline

Os splines são funções polinomiais por partes que possuem boas propriedades de aproximação, convergência e estabilidade com relação aos erros de arredondamento [7]. Isso ocorre porque elas têm um certo grau de descontinuidade, das suas derivadas de ordem mais elevadas, em alguns pontos do intervalo.

Essas funções estão associadas a uma partição predeterminada no intervalo de trabalho $[\alpha, \beta]$. Uma partição $\Omega$ será definida pelos pontos $x_0, x_1, \ldots, x_N$ tais que $\alpha = x_0 < x_1 < x_2 < \ldots < x_{N-1} < x_N = \beta$.

Em cada subintervalo $(x_i, x_{i+1})$, $i = 0, 2, \ldots, N-1$, os splines são polinômios. Na definição geral dos splines são impostas algumas restrições. A função $s_M(x)$ é chamada de spline de grau $M$, com nós nos pontos $x_i$ $(i = 0, 1, \ldots, N)$, se:

1. $s_M(x)$ é um polinômio de grau $M$ em cada subintervalo $(x_i, x_{i+1})$;
2. $s_M(x)$ é contínua e tem derivada contínua até ordem $(M-1)$ em $(\alpha, \beta)$.

Se, além disso, $s_M(x)$ também satisfaz a condição $s_M(x) = f(x_i)(\ i = 0, 1, \ldots, N)$, então será denominada *spline interpolante.*

O conjunto das funções splines de grau $M$, associado à partição $\Omega$, forma um espaço vetorial cuja dimensão depende de $M$ e do número de nós da partição $N+1$.

Pode-se estabelecer bases para o espaço vetorial das funções splines [7]. A seguir, é apresentada a base para os splines lineares ($M = 1$) [7].

Em cada nó $x_i$ da partição é definida a função:

$$l_i(x) = \begin{cases} \frac{x - x_{i-1}}{x_i - x_{i-1}}, & \text{se } x_{i-1} \leq x \leq x_i \\ \frac{x_{i+1} - x}{x_{i+1} - x_i}, & \text{se } x_i \leq x \leq x_{i+1} \\ 0 & \text{se } x < x_{i-1} \text{ ou } x > x_{i+1} \end{cases} \tag{2.19}$$

Os splines de grau 1, associados à partição definida por $x_i$, $i = 0, 1, \ldots, N$, podem ser escritos na forma:

$$s(x) = s_0 l_0(x) + s_1 l_1(x) + \ldots + s_m l_m(x), \tag{2.20}$$

sendo $s_i = s(x_i)$ os valores que $s(x)$ assume em $x_i$.

Se $\{\gamma_i(x)\}$ é uma base para o espaço vetorial dos splines de grau $M$ e $s_M(x)$ é um spline de grau $M$, então:

$$s_M(x) = c_0\gamma_0(x) + c_1\gamma_1(x) + c_2\gamma_2(x) +, \ldots, c_s\gamma_s(x), \tag{2.21}$$

sendo $c_0, c_1, c_2, \ldots, c_s$ os coeficientes da combinação linear e $s$ a dimensão do espaço vetorial.

Para usar as funções splines em interpolação, toma-se os pontos de interpolação como nós da partição à qual as funções splines são associadas. Para interpolação linear por partes os coeficientes $c_i$ são procurados de forma que:

$$s_1(x) = \sum_{i=0}^{M} c_i l_i(x) \qquad \text{e} \qquad s_1(x_j) = f(x_j) = f_j, \quad j = 0, \ldots, N \tag{2.22}$$

onde $l_i(x)$ é a função (2.19) para a base linear [7, 40].

Como foi visto, $l_i(x_j) = 0$ se $i \neq j$ e $l_i(x_i) = 1$, portanto

$$s_1(x_i) = c_i = f(x_i) = f_i. \tag{2.23}$$

Para o caso da base linear, o spline que interpola $f_0, f_1, \ldots, f_M$, é obtido por:

$$s_1(x) = \sum_{i=0}^{M} f_i l_i(x). \tag{2.24}$$

Splines são empregados em diversas áreas de aproximação de funções, inclusive os mínimos quadrados [7].

A seguir, são definidas algumas medidas de erro empregadas no problema de aproximação.

## 2.3 Análise do Erro no Caso Contínuo

Geralmente é necessário um parâmetro de comparação entre o valor exato e o aproximado, para saber quanto a aproximação é eficiente. Intuitivamente, uma medida satisfatória do erro seria zero se a aproximação fosse exata, pequena se os dois valores fossem próximos e grande se a aproximação fosse pobre. Mas os termos que foram usados, como pequenos, próximos, grande ou pobre, mudam de caso para caso, pois um valor que representaria uma boa aproximação numa determinada situação, poderia se tornar pobre em outra. Isso dificulta a definição de uma medida única de erro, pois cada uma delas podem ser adequadas para certas situações e não serem para outras.

O erro absoluto é calculado, por definição, como:

$$E_{\mathrm{a}}(x) = |f(x) - g_{\mathrm{a}}(x)|\,. \tag{2.25}$$

O erro absoluto não é satisfatório em todas as situações. Por esta razão consideram-se diferentes tipos de medidas como as descritas abaixo. Escolheu-se $\varepsilon$ para representar o erro no caso contínuo e $w(x)$ para uma função peso não negativa. A função peso, permite que diferentes graus de importância sejam atribuídos à aproximação de certas partes do intervalo. Um exemplo clássico dessa função é:

$$w(x) = \frac{1}{\sqrt{1-x^2}} \quad \text{com } x \in (-1,1), \tag{2.26}$$

que, como foi visto, ortogonaliza os polinômios de Chebyshev no intervalo $[-1,1]$. Esta função atribui menos peso à região central do intervalo $(-1,1)$ e dá maior ênfase aos pontos próximos dos extremos desse intervalo. A função peso é usada quando se deseja um tratamento diferenciado para determinadas partes do intervalo de trabalho. Por exemplo, pode-se considerar que uma determinada região do intervalo não produz efeito sobre o problema e, portanto, pode ser desconsiderado. Então, nessa região a função peso assume o valor zero.

**Medidas de Erro freqüentemente usadas:**

Há três medidas diferentes normalmente usadas nos problemas de ajuste de dados ou de aproximação de funções, sendo elas

1. Erro Quadrático Médio Ponderado

$$\varepsilon_2 = \frac{1}{2}\int_{-1}^{1} w(x)E_{\mathrm{a}}^2(x)dx, \tag{2.27}$$

2. Erro Médio Ponderado

$$\varepsilon_1 = \frac{1}{2}\int_{-1}^{1} w(x)E_a(x)dx, \tag{2.28}$$

3. Erro Máximo Absoluto Ponderado

$$\varepsilon_{max} = \max_{x\in[-1,1]} w(x)E_a(x). \tag{2.29}$$

Os exemplos 1,2 e 3 são classificados dentro das normas $L_2(a), L_1(a)$ e $L_\infty(a)$ [28], respectivamente, onde o exemplo 1 corresponde ao quadrado da norma $L_2(a)$.

Deseja-se encontrar a melhor aproximação possível, então é óbvio que o erro deverá ser o mais próximo de zero. Logo, o erro encontrado no problema de aproximação deve ser minimizado. Cada tipo de erro descrito gera uma classe de problemas que requer técnicas de solução específicas. O próximo ítem discute as características do problema surgido na minimização do erro quadrático médio ponderado.

### 2.3.1 Minimização do Erro Quadrático Médio Ponderado

Outra técnica que usa como critério de aproximação a minimização dos resíduos é o método dos mínimos quadrados, cujo objetivo é procurar parâmetros que minimizem a medida do erro, que no caso contínuo é dado pela integral da função resíduo ao quadrado [7, 8], portanto

$$\int_a^b \mid f(x) - g(x) \mid^2 dx, \tag{2.30}$$

onde $g(x)$ é a função de aproximação e $f(x)$ é a função a ser aproximada.

Dentre as medidas de erro mais usadas, considera-se como exemplo a minimização do erro quadrático médio ponderado para o caso geral, que por sua definição é semelhante ao critério usado pelo método dos mínimos quadrados.

A seguir, será realizada a minimização de $\varepsilon_2$, resultando no problema clássico definido acima.

$$\varepsilon_2 = \frac{1}{2}\int_{-1}^{1} w(x)E_a^2(x)dx$$

$$\varepsilon_2 = \frac{1}{2}\int_{-1}^{1} w(x) \mid f(x) - g_a(x) \mid^2 dx$$

$$\varepsilon_2 = \frac{1}{2}\int_{-1}^{1} w(x)f^2(x)dx - \int_{-1}^{1} w(x)f(x)\sum_{i=0}^{M} a_i r_i(x)dx$$

$$+\frac{1}{2}\int_{-1}^{1} w(x)\left[\sum_{i=0}^{M} a_i r_i(x)\right]^2 dx. \tag{2.31}$$

Desmembrando o termo com somatório ao quadrado, tem-se:

$$\left[\sum_{i=0}^{M} a_i r_i(x)\right]^2 = \sum_{i=0}^{M} a_i r_i(x)\sum_{j=0}^{M} a_j r_j(x) = \sum_{i=0}^{M}\sum_{j=0}^{M} a_i a_j r_i(x) r_j(x),$$

substituindo o resultado encontrado na equação (2.31) obtem-se:

$$\varepsilon_2 = \frac{1}{2}\int_{-1}^{1} w(x)f^2(x)dx - \sum_{i=0}^{M} a_i \int_{-1}^{1} w(x)r_i(x)f(x)dx +$$

$$\frac{1}{2}\sum_{i=0}^{M}\sum_{j=0}^{M} a_i a_j \int_{-1}^{1} w(x)r_i(x)r_j(x)dx. \tag{2.32}$$

Após resolver as integrais, o primeiro termo tem um resultado independente do valor de $a_i$. Portanto, este resultado não interfere no problema de minimização e pode ser desconsiderado quando o erro for minimizado. No segundo termo, aparecerá um elemento em função de $i$ , sendo este denominado $v_i$. Finalmente, o resultado da integral do terceiro termo estará em função de $i$ e $j$, sendo denotado por $\psi_{ij}$. Define-se, então:

$$v_i = \int_{-1}^{1} w(x)r_i(x)f(x)dx \qquad \text{e} \qquad \psi_{ij} = \int_{-1}^{1} w(x)r_i(x)r_j(x)dx,$$

obtendo-se

$$\varepsilon_2 = \frac{1}{2}\int_{-1}^{1} w(x)f^2(x)dx + \frac{1}{2}\sum_{i=0}^{M}\sum_{j=0}^{M} a_i\psi_{ij}a_j - \sum_{i=0}^{M} a_i v_i \tag{2.33}$$

Conseqüentemente, a minimização deste erro em termos de $\mathbf{a}$, é definido na forma matricial:

$$\min_{\mathbf{a}} \quad \varepsilon_2(\mathbf{a}) = \frac{1}{2}\,\mathbf{a}^t\boldsymbol{\Psi}\mathbf{a} - \mathbf{v}^t\mathbf{a},$$

(2.34)

se a matriz $\Psi$ puder ser invertida, a solução de (2.34) é dada por:

$$\mathbf{a} = \Psi^{-1}\mathbf{v}, \tag{2.35}$$

É possível mostrar que, para o problema de minimização de $\varepsilon_2$, existe uma solução fechada relativamente fácil de ser encontrada, dependendo da matriz $\Psi$ [7, 15].Por outro lado, na minimização de $\varepsilon_1$ e $\varepsilon_{max}$ não é possível obter soluções exatas. Entretanto, pode-se lançar mão de métodos de otimização, como a Programação Linear, proposto neste trabalho.

Dividindo o intervalo em $N$ pontos, obtem-se uma aproximação discreta que resulta num problema de programação linear com $N$ restrições e número finito de variáveis. Se o número de pontos ($N$) crescesse indefinidamente, o problema seria de programação linear semi-infinito, com o número de restrições crescendo indefinidamente e número finito de variáveis.

Na próxima seção, será dada a descrição do que significa discretizar um intervalo e como fica a estrutura de cada problema, quando se deseja minimizar as medidas de erro já comentadas. Mas antes de prosseguir esta discussão, serão mostrados alguns exemplos de caracterização da matriz $\Psi$ (2.35).

### Natureza da matriz $\Psi$

A natureza da matriz $\Psi$ dependerá do tipo da função de interpolação $r_i(x)$ escolhido. A seguir, analisam-se alguns exemplos, considerando a função peso $w(x) = 1,\ \forall x \in [-1, 1]$.

1. **Quando a Função de Interpolação é um Polinômio**

   Se a função de interpolação é um polinômio (2.5), a matriz $\Psi$ possui a seguinte lei de formação:

   $$\psi_{ij} = \begin{cases} 0, & \text{se } i+j \text{ não for par} \\ \frac{2}{i+j+1}, & \text{caso contrário,} \end{cases} \tag{2.36}$$

   onde $i, j = 0, 1, \ldots, M$. O resultado é uma matriz de Hankel [19], como mostra o exemplo a seguir.

Para $i, j = 0, 1, 2, 3, 4$,

$$\Psi = \begin{pmatrix} 2 & 0 & 2/3 & 0 & 2/5 \\ 0 & 2/3 & 0 & 2/5 & 0 \\ 2/3 & 0 & 2/5 & 0 & 2/7 \\ 0 & 2/5 & 0 & 2/7 & 0 \\ 2/5 & 0 & 2/7 & 0 & 2/9 \end{pmatrix} \tag{2.37}$$

Em [15] são descritas algumas maneiras de resolver sistemas com esta estrutura de matriz.

2. **Quando a Função de Interpolação é trigonométrica**

Sendo $r(x)$ uma função trigonométrica, como a definida em (2.6), pode-se desconsiderar os termos dependentes de $b_i$, se a função a ser aproximada for par ($f(-x) = f(x)$), e a matriz $\Psi$ é dada por:

$$\psi_{ij} = \begin{cases} \frac{1}{2}, & \text{se } i = j = 0 \\ 1, & \text{se } i = j \neq 0 \\ 0, & \text{se } i \neq j. \end{cases} \tag{2.38}$$

Substituindo o resultado acima em (2.35), conclui-se que o $a_i$ ótimo possui a forma:

$$a_i = \int_{-1}^{1} f(x)\cos(i\pi x)dx \quad \text{para} \quad i = 0, 1, \ldots, M. \tag{2.39}$$

Usando a mesma técnica para encontrar o valor ótimo dos $b_i$, tem-se

$$b_i = \int_{-1}^{1} f(x)\sin(i\pi x)dx \quad \text{para} \quad i = 0, 1, \ldots, M, \tag{2.40}$$

que correspondem aos coeficientes de Fourier.

3. **Quando a Função de Interpolação é um polinômio de Legendre**

Seja a função de aproximação definida em (2.10), então a estrutura da matriz $\Psi$ é

$$\psi_{ij} = \begin{cases} 0, & \text{se } i \neq j, \\ \frac{2}{2i+1}, & \text{se } i = j, \end{cases} \tag{2.41}$$

tornando a solução do sistema (2.35) trivial, pois a matriz $\Psi$ é diagonal.

## 2.4 Problema Discretizado

O problema de aproximação foi até o momento observado no caso contínuo, cujo objetivo é fazer uma aproximação em todo o intervalo de interesse. Em alguns casos, devido à dificuldade

de cálculo ou mesmo por características particulares do problema, utiliza-se um número finito de divisões do intervalo. A divisão do intervalo pode ser feita de várias maneiras, considerando a natureza do problema ou se adequando ao objetivo desejado.

Para resolver este problema, considera-se $N+1$ como o número de divisões do intervalo discretizado, onde os pontos relacionados $(x_0, f(x_0)), (x_1, f(x_1)), \ldots, (x_N, f(x_N))$ são tais que $x_k \neq x_s$ quando $k \neq s$. O vetor $\mathbf{x}$, que possui os valores dos pontos selecionados no intervalo discretizado, pode, por exemplo, ser definido em $[\alpha, \beta]$ do seguinte modo:

$$x_k = \alpha + \frac{k(\beta - \alpha)}{N} \quad \text{para} \quad k = 0, 1, \ldots, N, \tag{2.42}$$

e, no intervalo $[-1, 1]$

$$x_k = \frac{2k}{N} - 1, \quad \text{para} \quad k = 0, 1, \ldots, N. \tag{2.43}$$

Torna-se necessário redefinir o problema para o caso discreto. Para facilitar os cálculos futuros, assume-se que as funções estão pré-multiplicadas pela função peso $w(x_k)$, $\forall\ k \in \{0, 1, \ldots, N\}$, e define-se $f_k = w(x_k) f(x_k)$.

Seja $\mathbf{f}$ o vetor dos valores ponderados da função original nos pontos discretizados e $g_k$ definido pela equação a seguir:

$$\begin{aligned} g_k = \quad w(x_k) g_a(x_k) \quad &= \sum_{i=0}^{M} a_i w(x_k) r_i(x_k) \\ &= \sum_{i=0}^{M} a_i w_k r_{ki}. \end{aligned} \tag{2.44}$$

Nota-se que para todos os pontos de $\mathbf{x}$, a função de aproximação ponderada pode ser escrita na forma matricial:

$$\mathbf{g} = \mathbf{Ra}. \tag{2.45}$$

$\mathbf{R}$ é a matriz dos resultados da função de interpolação ponderada nos pontos discretos, $\mathbf{a}$ é o vetor dos parâmetros da aproximação e $\mathbf{g}$ é o vetor dos valores da função de aproximação associada à função peso.

No caso discreto o erro é denotado por $\epsilon$. Quando o Erro Médio Ponderado, o Erro Quadrático Médio Ponderado e o Erro Máximo Absoluto Ponderado são minimizados, sobre um intervalo discreto, geram problemas que necessitam de técnicas específicas para a sua solução.

## 2.5 Análise do Erro no Caso Discreto

Note que, no caso discreto, a definição do erro é a mesma de (2.25) e que na análise de cada caso surgido com a minimização dos tipos de erros descritos, os parâmetros $\mathbf{f}, \mathbf{R}$, $f_k$ e $g_k$ estão associados à função peso $w(x_k)$.

### 2.5.1 Erro Médio Ponderado

Seja o erro de aproximação dado por:

$$\epsilon_1 = \frac{1}{N+1} \sum_{k=0}^{N} w(x_k) E_{\mathbf{a}}(x_k). \tag{2.46}$$

Para minimizar este erro, precisaremos resolver o seguinte problema:

$$w(x_k) E_{\mathbf{a}}(x_k) = \mid w(x_k) f(x_k) - w(x_k) g_{\mathbf{a}}(x_k) \mid = \mid f_k - g_k \mid . \tag{2.47}$$

Para minimizar $\epsilon_1$ em $\mathbf{a}$, pode-se escrever:

$$\min_{\mathbf{a}} \epsilon_1(\mathbf{a}) = \min_{\mathbf{a}} \sum_{k=0}^{N} \mid f_k - g_k \mid . \tag{2.48}$$

$f_k$ e $g_k$ são funções da variável de otimização $x_k$, que é irrestrita. A diferença de funções $(f_k - g_k)$ pode ser escrita como a subtração de duas variáveis não negativas [28]. Ou seja,

$$f_k - g_k = u_k - v_k, \tag{2.49}$$

onde $u_k \geq 0$, $v_k \geq 0$. Por definição:

$$u_k = \begin{cases} 0 & \text{se } (f_k - g_k) \leq 0 \\ (f_k - g_k) & \text{se } (f_k - g_k) > 0 \end{cases} \tag{2.50}$$

$$v_k = \begin{cases} 0 & \text{se } (f_k - g_k) \geq 0 \\ -(f_k - g_k) & \text{se } (f_k - g_k) < 0 \end{cases} \tag{2.51}$$

Pode-se garantir que $u_k v_k = 0$ [28]. Aplicando-se a desigualdade triangular à equação (2.49), tem-se:

$$\mid f_k - g_k \mid \leq \mid u_k \mid + \mid v_k \mid . \tag{2.52}$$

Como $u_k$ e $v_k$ são sempre positivos:

$$\mid f_k - g_k \mid \leq u_k + v_k, \tag{2.53}$$

e satisfazendo $u_k \geq 0,\ v_k \geq 0$, (2.50) e (2.51)

$$| f_k - g_k | = u_k + v_k. \tag{2.54}$$

Substituindo (2.54) no problema (2.48), tem-se:

$$\begin{array}{l} \min_{u,v} \quad \sum_{k=0}^{N}(u_k + v_k) \\ \text{s. a} \\ \left.\begin{array}{rcl} g_k + u_k - v_k & = & f_k \\ u_k & \geq & 0 \\ v_k & \geq & 0 \end{array}\right\} \quad k = 0, 1, \ldots, N \end{array} \tag{2.55}$$

Ou escrevendo o problema (2.55) na forma matricial, tem-se:

$$\begin{array}{l} \min_{u,v} \quad \mathbf{u} + \mathbf{v} \\ \text{s. a} \\ \left.\begin{array}{rcl} \mathbf{R^t a} + \mathbf{u} - \mathbf{v} & = & \mathbf{f} \\ \mathbf{u} & \geq & 0 \\ \mathbf{v} & \geq & 0 \end{array}\right\} \quad k = 0, 1, \ldots, N \end{array} \tag{2.56}$$

Observa-se que (2.55) e (2.56) são problemas de programação linear (PL); e podem ser resolvidos por alguma das técnicas tradicionais da programação linear [4, 26].

### 2.5.2 Erro Quadrático Médio Ponderado

No caso discretizado, a minimização do erro quadrático médio ponderado também produz uma solução exata como no caso contínuo. A sua definição possui o mesmo critério que o método dos mínimos quadrados usa para a minimização dos resíduos no caso discreto. O erro quadrático médio ponderado é caracterizado pela fórmula a seguir.

$$\epsilon_2 = \frac{1}{N+1} \sum_{k=0}^{N} w(x) E_{\mathrm{a}}^2(x_k). \tag{2.57}$$

Usando a notação matricial, obtém-se:

$$\begin{aligned} \mathbf{E}_2 &= \| \mathbf{f} - \mathbf{g} \|^2 \\ &= \mathbf{f}^t\mathbf{f} - 2\mathbf{f}^t\mathbf{g} + \mathbf{g}^t\mathbf{g} \\ &= \mathbf{f}^t\mathbf{f} - 2\mathbf{f}^t\mathbf{R}\mathbf{a} + \mathbf{a}^t\mathbf{R}^t\mathbf{R}\mathbf{a}, \end{aligned} \tag{2.58}$$

como o termo $\mathbf{f}^t\mathbf{f}$ é uma constante positiva e não possui influência sobre a minimização do erro dos parâmetros da aproximação do vetor $\mathbf{a}$, pode-se abandoná-lo e procurar o erro mínimo em $\mathbf{a}$, um vetor de variáveis irrestritas.

$$\min_{\mathbf{a}} \quad \mathbf{E}_2(\mathbf{a}) \tag{2.59}$$

cuja solução é dada através de:

$$\mathbf{a} = (\mathbf{R}^t\mathbf{R})^{-1}\mathbf{R}^t\mathbf{f}. \tag{2.60}$$

Para resolver o sistema de equações (2.60) existem métodos especiais [15].

### 2.5.3 Erro Máximo Absoluto Ponderado

Define-se

$$\epsilon_{max} = \max_k \; w(x_k)E_{\mathbf{a}}(x_k), \quad \text{para todo} \quad k \in \{0, 1, \ldots, N\}. \tag{2.61}$$

Será dada maior atenção a este problema, pois para a aplicação em projeto de filtros digitais, a definição de otimalidade que será usada é o erro absoluto máximo.

Na minimização dessa medida de erro surgirá um problema $\min_{\mathbf{a}} \max_k$, que pode ser decomposto em um problema do tipo:

$$\epsilon_{max} \quad = \quad \| \mathbf{f} - \mathbf{g} \|_\infty \tag{2.62}$$

onde

$$\| \mathbf{f} - \mathbf{g} \|_\infty = \max_k |f_k - g_k| = Z \tag{2.63}$$

Retomando a definição de $g_k$ (2.44), o problema pode ser reescrito na forma a seguir:

$$\begin{array}{lll} \min & Z & \\ \text{s. a} & & \\ -g_k - Z & \leq & -f_k \\ g_k - Z & \leq & f_k \\ g_k & = & \sum_{i=0}^{M} a_i w(x_k) r_i(x_k). \end{array} \quad \Bigg\} \quad k = 0, 1, \ldots, N \tag{2.64}$$

ou ainda na forma matricial:

$$\begin{aligned} \min \quad & Z \\ \text{s.a} \quad & \\ & -\mathbf{R}^t\mathbf{a} - Z.\mathbf{1} \leq -\mathbf{f} \\ & \mathbf{R}^t\mathbf{a} - Z.\mathbf{1} \leq \mathbf{f}, \end{aligned} \tag{2.65}$$

sendo que $\mathbf{1}$ representa um vetor de dimensão $N+1$ com todos os elementos iguais a 1. Este é um problema de Minimax bem conhecido dentro da programação linear.

### 2.5.4 Considerações sobre o problema de aproximação

De modo análogo ao que foi descrito no caso contínuo sobre as norma $L_1(a)$, a norma $L_2(a)$ (mais especificamente o quadrado desta norma) e $L_\infty(a)$, no caso discreto elas também representam as medidas $\epsilon_1, \epsilon_2$ e $\epsilon_{max}$, respectivamente.

As normas $L_1(a)$ e $L_\infty(a)$ são empregadas, com maior freqüência, nos problemas de ajuste de dados, sendo normalmente os valores das funções objetivos de (2.55) e (2.65) muito pequenos. Quando o resultado obtido é diferente de zero, alguns autores sugerem [14] que o problema seja resolvido através do quadrado da norma $L_2$. No entanto, essa forma de minimização do erro só é eficaz se a matriz produzida pela função de interpolação for inversível e/ou simples de ser calculada. Mesmo que essas restrições sejam satisfeitas, pode-se afirmar que a solução de um problema de aproximação através de Programação Linear é muito mais vantajosa e abrangente, pois não há problemas no caso de inserir algum tipo de restrição no modelo. Um exemplo dessas restrições adicionais é forçar algum dos coeficientes a ter um valor específico, o que não é possível fazer diretamente pelo método dos mínimos quadrados.

Programação linear é uma ferramenta poderosa no problema de aproximação, embora ainda seja pouco divulgada nesse campo de problemas. Recomenda-se o uso do "simplex revisado para resolver a forma dual do problema, explorando melhor sua estrutura e economizando esforço computacional. As soluções encontradas são de boa qualidade e o método muito simples de ser implementado. Além do mais, a maioria dos métodos existentes para aproximar uma função destinam-se apenas aos casos em que a função a ser aproximada é contínua. Programação linear, pode ser empregada para aproximar qualquer tipo de função, contínua ou descontínua (em algum ponto do intervalo de trabalho).

Maiores detalhes desse método serão apresentados nos próximos capítulos, onde se desenvolverá uma aplicação no problema de projeto de filtros digitais.

# Capítulo 3

# Filtros Digitais

O projeto de filtros digitais pode ser visto como um caso especial do problema de aproximação de funções. Este capítulo discute brevemente os principais conceitos associados ao projeto desses filtros, apresenta interpretações do problema sobre a ótica de aproximações de funções e faz uma breve revisão bibliográfica.

A aproximação da resposta em freqüência, que corresponde à primeira fase do projeto de filtros digitais (seção 3.1.5), obteve grande atenção dos pesquisadores a partir do final da década de 60 e início da década de 70. A atividade de pesquisa na área continua intensa devido a suas aplicações em sistemas de telecomunicações.

## 3.1 Introdução

Os filtros são usados para transformar sinais elétricos, eliminando freqüências indesejáveis. Eles são essenciais no projeto de sistemas de comunicações, cuja rápida expansão tornou o mercado extremamente competitivo e cada vez mais exigente quanto à qualidade e ao volume do tráfego da informação. Devido a essas exigências os filtros são constantemente aperfeiçoados.

Nos itens a seguir, faz-se uma revisão sobre alguns elementos fundamentais para o projeto de filtros digitais.

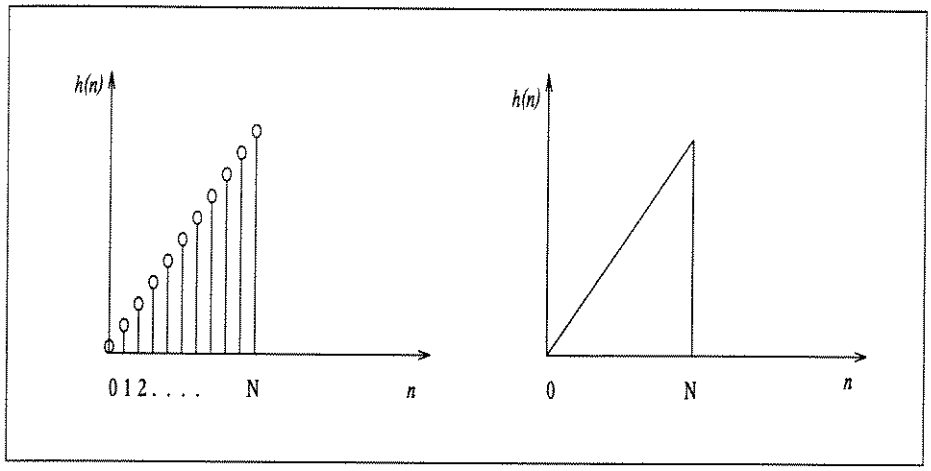


Figura 3.1: Representações de uma seqüência

### 3.1.1 Sinal Analógico e Sinal Digital

Normalmente, os sinais elétricos são produzidos a partir de fenômenos que apresentam variações contínuas no tempo. Em algumas aplicações, assumem valores contínuos, constituindo-se réplicas dos fenômenos dos quais se originam. Por serem réplicas, costuma-se chamar os sinais elétricos assim produzidos de *sinais analógicos* ou *sinais contínuos no tempo.*

Existem aplicações, entretanto, em que os sinais elétricos assumem alguns valores discretos, aos quais se pode associar dígitos; por isso, são chamados de *sinais digitais* ou *sinais discretos no tempo.*

Os sinais digitais possuem aplicações variadas. Por exemplo, companhias telefônicas, rádios e televisões usam sinais digitais para representar a voz humana. Outros sistemas de som e imagens, como o toca-disco laser e o CD-ROM dos computadores, também são movidos pela tecnologia digital. Ela proporciona fidelidade superior, ausência de ruídos e maior flexibilidade de processamento de sinais.

Os sinais discretos no tempo surgem de duas formas distintas: eles podem ser inerentemente discretos no tempo, como, por exemplo, médias diárias de temperatura, ou podem ser versões amostradas de sinais contínuos no tempo. Um sinal é uma seqüência de números reais ou complexos. Uma seqüência completa pode ser denotada por "$\mathbf{y}$" ou "$y(n)$", ou seja,

$$\mathbf{y} = \{y(n) : n = \ldots, -2, -1, 0, 1, 2, \ldots\} \tag{3.1}$$

Geralmente o tempo é amostrado em intervalos uniformes, ou seja, $t = nT$, onde $T$ é o intervalo

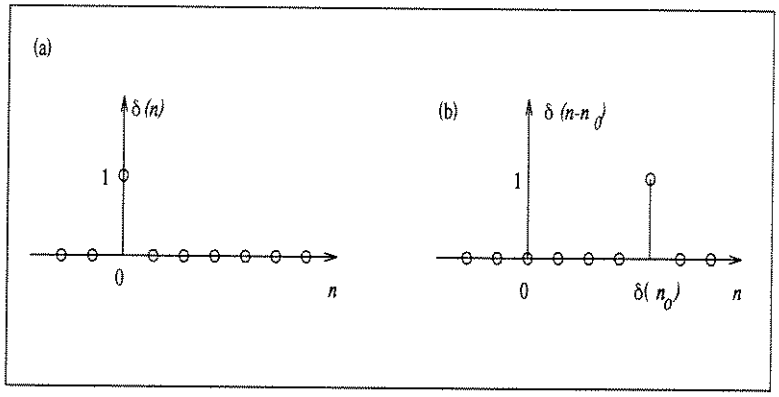


Figura 3.2: Exemplos de seqüências comuns em processamento de sinal digital

entre os instantes de amostragem. A seqüência $\mathbf{y}$ é obtida de um sinal contínuo $y(t)$ através da amostragem uniforme no tempo, isto é, $\mathbf{y} = y(nT)$. Por exemplo, na Fig. 3.1, os números $0, 1, 2, \ldots, N$ formam uma seqüência crescente $y(n) = n,\ 0 \leq n \leq N$.

Duas seqüências de sinais discretos no tempo freqüentemente usadas no processamento de sinais digitais, são mostradas na Figura 3.2. Na Figura 3.2 (a), $\delta(n)$ mostra um impulso digital ou amostra unitária definida pela relação:

$$\delta(n) = \begin{cases} 1, & \text{se } n = 0, \\ 0, & \text{caso contrário.} \end{cases} \tag{3.2}$$

Na Fig. 3.2 (b) $\delta(n - n_0)$ mostra um impulso atrasado por $n_0$ amostras, definido abaixo:

$$\delta(n - n_0) = \begin{cases} 1, & \text{se } n = n_0, \\ 0, & \text{caso contrário.} \end{cases} \tag{3.3}$$

Os sinais (seqüências) podem ser manipulados de várias maneiras, por exemplo um sinal de saída pode satisfazer a seguinte recursão:

$$y(n) = x(n) + \frac{1}{2}y(n-1), \qquad n \geq 0. \tag{3.4}$$

Por definição, $\mathbf{h}$ é a resposta para o impulso digital [36], ou seja, $\mathbf{h}$ é o sinal de saída quando o sinal de entrada é $\mathbf{x} = \delta$. Neste caso, se $\mathbf{x} = \delta$ então $\mathbf{y} = \mathbf{h}$,

$$h(n) = \delta(n) + \frac{1}{2}h(n-1), \qquad n \geq 0. \tag{3.5}$$

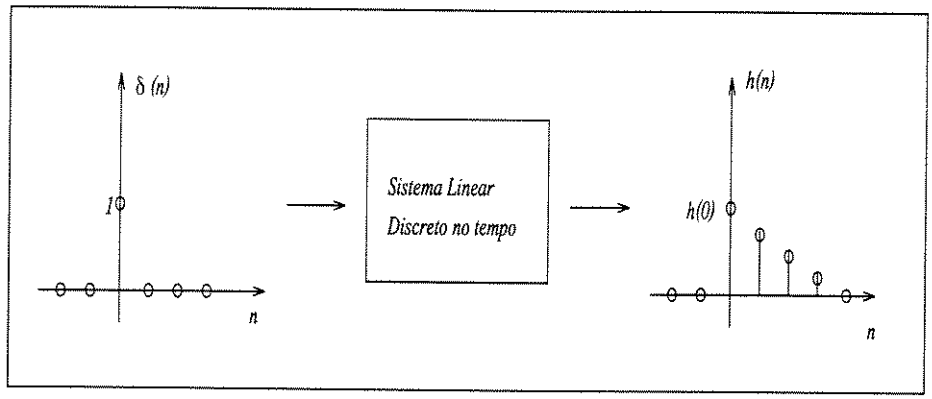


Figura 3.3: Representações de um sistema estático linear discreto no tempo

Para $n = 0$, tem-se:

$$h(0) = \delta(0) + \frac{1}{2}h(-1) = 1 \tag{3.6}$$

continuando

$$h(1) = \delta(1) + \frac{1}{2}h(0) = \frac{1}{2}, \tag{3.7}$$

$$h(2) = \delta(2) + \frac{1}{2}h(1) = (\frac{1}{2})^2, \tag{3.8}$$

$$\vdots \tag{3.9}$$

$$h(n) = (\frac{1}{2})^k. \tag{3.10}$$

Um sistema de processamento de sinais discreto no tempo é essencialmente um algoritmo para converter uma seqüência de entrada, $x(n)$ ou $\delta(n)$, em uma seqüência de saída, $y(n)$ ou $h(n)$. Uma representação simples de um sistema linear e discreto no tempo, é dado pela figura 3.3 que representa as entradas e saídas da equação (3.5).

Usando a seqüência de impulso digital e devido à facilidade de manipular as seqüências de sinais, pode-se representar seqüências arbitrárias em termos de impulsos atrasados e escalonados [36]. Assim, $x(n)$ é representado por:

$$x(n) = \sum_{m=-\infty}^{\infty} x(m)\delta(n-m). \tag{3.11}$$

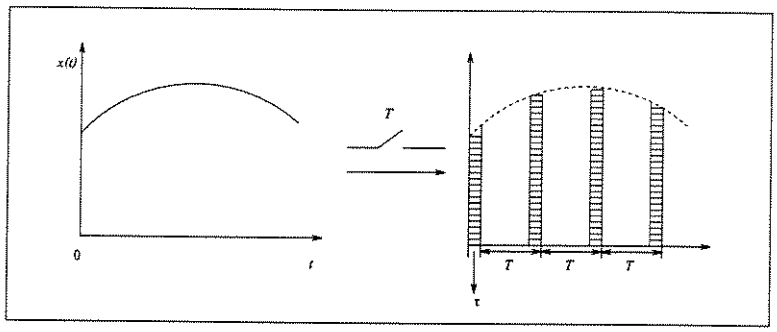


Figura 3.4: Modelo do processo de amostragem

Como $h(n)$ é a resposta para a seqüência $\delta(n)$, e uma das características do sistema é a invariância no tempo, pode-se dizer que $h(n-m)$ é a resposta para a seqüência $\delta(n-m)$. Da mesma forma, pela linearidade do sistema, a resposta para a seqüência $x(m)\delta(n-m)$ deve ser $x(m)h(n-m)$. Logo, a resposta para o sinal de entrada $x(n)$ é:

$$y(n) = \sum_{m=-\infty}^{\infty} x(m)h(n-m). \tag{3.12}$$

A equação (3.12) também pode ser escrita como:

$$y(n) = \sum_{m=-\infty}^{\infty} h(m)x(n-m). \tag{3.13}$$

O processo de amostragem [2, 45], ou de conversão de um sinal analógico para a forma discreta, possui grande importância no processamento computacional; para ser processado num computador o sinal deve ser discreto no tempo e limitado em amplitude. A seguir é dado um modelo simples, mas ilustrativo, sobre esse processo. Considere um sinal de entrada contínuo no tempo, $x(t)$, a ser amostrado através de um interruptor que se fecha periodicamente, por um pequeno intervalo de tempo de $\tau$ segundos, e com um intervalo de amostragem de $T$ segundos. O sinal resultante é discreto no tempo, como ilustrado na Figura 3.4. Esse sinal de saída , por sua vez, pode ser convertido novamente num sinal contínuo no tempo, através de um conversor digital/analógico [45]. O próximo item faz uma breve discussão sobre esse procedimento para conversão e filtragem de sinais.

### 3.1.2 Processo Digital de Filtragem

A manipulação de sinais analógicos por dispositivos digitais, como ocorre no conversor analógico-digital, possibilita o processo típico de filtragem digital, mostrado esquematicamente na Figura 3.5.

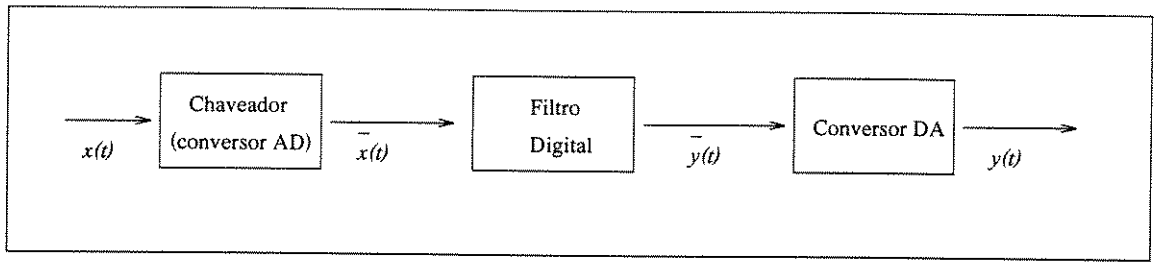


Figura 3.5: Diagrama em blocos de um sistema de filtragem digital

O chaveador converte o sinal puramente analógico $x(t)$ em um sinal amostrado $\bar{x}(t)$. Esse sinal é processado através do filtro digital, que apresenta em sua saída o sinal filtrado $\bar{y}(t)$. Este sinal possui uma forma discreta, tal como o sinal amostrado $\bar{x}(t)$. Segue-se um conversor digital-analógico (DA) que recompõe o sinal analógico filtrado $y(t)$ a partir do sinal $\bar{y}(t)$.

Essa técnica, apesar das conversões necessárias (AD e DA) oferece as seguintes vantagens em relação à filtragem puramente analógica [30, 45]:

- é programável, ou seja, as características do filtro podem ser facilmente mudadas;
- possui desempenho superior em termos de precisão;
- pode-se implementar característica de resposta em freqüência muito próxima da ideal;
- é tolerante à imprecisão de componentes;
- a filtragem adaptativa é relativamente simples de ser alcançada;
- os componentes elétricos necessários à sua implementação são facilmente encontrados e geralmente de baixo custo.

Os filtros digitais são dispositivos que possuem basicamente a função de filtrar algumas freqüências de um sinal. Mas isso pode ser feito de várias formas, como é discutido no próximo item.

### 3.1.3 Classificação dos Filtros

A diferenciação mais usual entre as diversas categorias de filtros se faz em termos da posição ocupada no *espectro* [36] pela faixa de freqüências que o filtro deve deixar passar, denominada *faixa passante*. Quando o espectro útil do sinal estiver situado entre os limites da faixa de passagem do sistema, o efeito da atenuação na transmissão é pouco significativo. Por isso, no

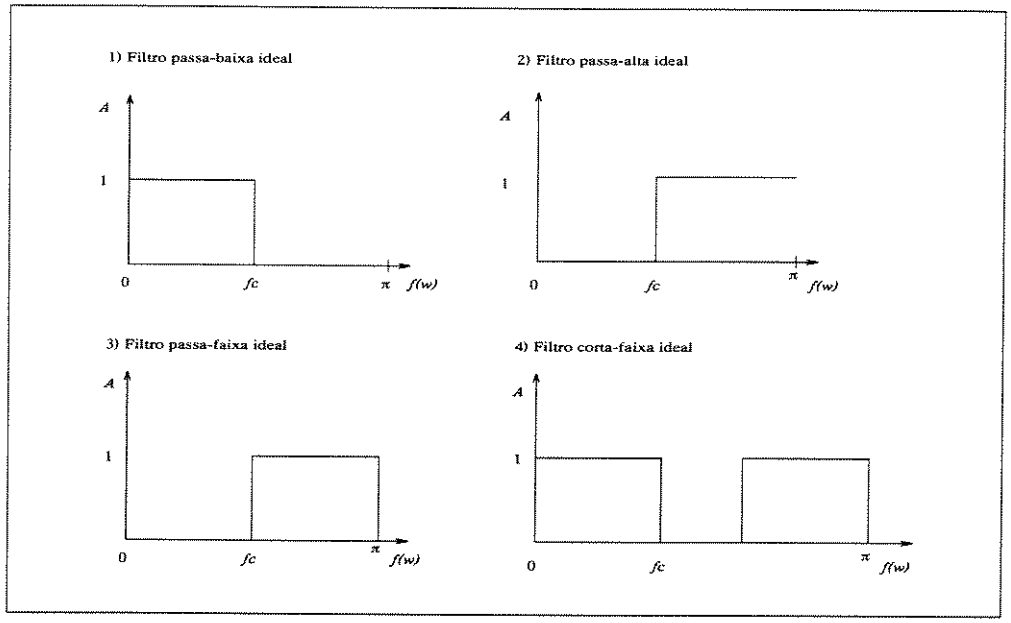


Figura 3.6: Curvas de filtros ideais - espectros unilaterais

projeto de filtros, é comum se analisar primeiro a largura de faixa do sinal e então selecionar, ou projetar, um filtro que deixe passar, no mínimo a faixa desejada. A seguir são descritas as categorias mais utilizadas de filtros:

1. **Filtros Passa-baixa:** São aqueles que deixam passar as freqüências de 0 Hz até um certo limite, rejeitando as freqüências que se situam além desse limite. Em geral servem de base para os projetos de outras modalidades de filtros. Por exemplo, os pares telefônicos que constituem as linhas urbanas se comportam como filtros passa-baixas. Os amplificadores convencionais na faixa de audio se comportam, para a maioria dos sinais digitais produzidos pelos equipamentos terminais, também como filtros passa-baixas.
2. **Filtros Passa-alta:** Possuem característica oposta à dos filtros passa-baixa, favorecendo a passagem das freqüências acima de um certo limite, e impedindo a passagem das que se situam abaixo do mesmo. Por exemplo, os guias de onda, na faixa de microondas, se comportam como filtros passa-altas.
3. **Filtros Passa-faixa:** Combinando, de uma certa forma, as características dos filtros passa-baixa e passa-alta, os filtros passa-faixa permitem a passagem das freqüências compreendidas em uma faixa delimitada por uma freqüência inferior e por uma superior, rejeitando as situadas abaixo do limite inferior ou acima do limite superior. Por exemplo, os canais multiplex telefônico e telegráfico se comportam

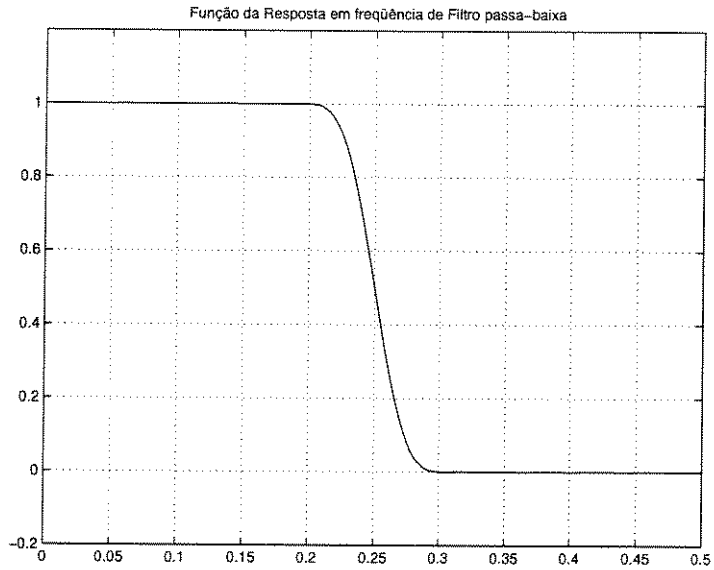


Figura 3.7: Aproximação da função da resposta em freqüência de um filtro passa-baixa

exatamente como filtros passa-faixas

4. **Filtros Corta-faixa:** Destinam-se a bloquear as freqüências compreendidas entre um limite inferior e um superior, dando passagem às demais freqüências dos espectro. Esses filtros apresentam características complementares aos filtros passa-faixa.

Os filtros descritos correspondem a modelos ideais que, pela presença de transições bruscas (descontinuidades), são irrealizáveis do ponto de vista físico. Esses modelos ideais, ilustrados na Figura 3.6, também possuem uma resposta plana na sua faixa útil. A realização física desses filtros pode ser feita de forma aproximada, dentro de uma faixa de tolerância.

A faixa útil do filtro passa-baixa ideal, representado pelo modelo (1) da Figura 3.6, vai de 0 até $fc$. Observe que a resposta em freqüência, que é o tema da próxima subseção nesse limite assume o valor 1 e, na faixa seguinte ($fc$ a $\infty$), assume valor zero.

### 3.1.4 A função da resposta em freqüência

A característica mais importante de um filtro é a sua resposta em freqüência, representada graficamente pela curva de atenuação versus freqüência (Figura 3.7).

Considere como entrada de um sistema linear invariante no tempo, como o descrito na seção 3.1.1, uma exponencial complexa com freqüência $\theta$, $e^{j\theta n} = \cos(\theta n) + j\sin(\theta n)$.

$$x(n) = e^{j\theta n}, \qquad -\infty < n < \infty. \tag{3.14}$$

Substituindo esta entrada na equação (3.13), a resposta para uma entrada exponencial na freqüência $\theta$ é dada por:

$$\begin{aligned} y(n) &= \sum_{m=-\infty}^{\infty} h(m)e^{j\theta(n-m)} \\ &= e^{j\theta n}\left[\sum_{m=-\infty}^{\infty} h(m)e^{-j\theta m}\right] \\ &= H(\theta)e^{j\theta n}. \end{aligned} \tag{3.15}$$

onde

$$H(\theta) = \sum_{m=-\infty}^{\infty} h(m)e^{-j\theta m}. \tag{3.16}$$

A função complexa de freqüência $\theta$, $H(\theta)$, é chamada de *função da resposta em freqüência de um sistema.* Note que, para os sistemas lineares invariantes no tempo, cada freqüência de entrada $\theta$ gera uma mesma freqüência na saída; mudam apenas a fase e a amplitude da exponencial complexa.

A função da resposta em freqüência (3.16) é periódica, com período igual a $2\pi$. Essa periodicidade é facilmente mostrada como segue:

$$\begin{aligned} H(\theta + 2\pi) &= \sum_{m=-\infty}^{\infty} h(m)e^{-j(\theta+2\pi)m} \\ &= \sum_{m=-\infty}^{\infty} h(m)e^{-j\theta m}e^{-j2\pi m} \\ &= H(\theta). \end{aligned} \tag{3.17}$$

A freqüência é denotada por $\theta$ em radianos por segundo ou por $f$ em *hertz* (ciclos por segundo). Tem-se:

$$\theta = 2\pi f. \tag{3.18}$$

A idéia de representar o comportamento de um sistema em termos de sua resposta à função exponencial complexa $e^{j\theta n}$ , pode ser generalizada por uma variável complexa qualquer $z$. Ou seja,

$$x(n) = z^n. \tag{3.19}$$

Nesse caso, em lugar da equação (3.15), tem-se::

$$y(n) = H(z)z^n, \tag{3.20}$$

onde

$$H(z) = \sum_{m=-\infty}^{\infty} h(m)z^{-m}. \tag{3.21}$$

$H(z)$, transformada $z$ [36] da resposta de impulso unitária $h(m)$, também chamada de *função de transferência do sistema.* Para essa representação ser válida é necessário que a somatória em (3.21) convirja *absolutamente* [30, 36]. A função da resposta em freqüência $H(\theta)$ é a função de transferência $H(z)$, avaliada sobre o círculo unitário $|z| = 1$ no plano complexo $z$.

A Figura 3.8 ilustra algumas formas de resposta em freqüência para filtros passa-baixa. A resposta em freqüência ideal mais simples é aquela cuja *faixa passante* se estende de 0 Hz até uma certa *freqüência de corte*, $fc$, e a *faixa de rejeição* se estende de $f = fc$ até $f = \pi$.

Em alguns casos, há um intervalo entre a faixa passante e a de rejeição, nos quais se deseja, ou não, a existência do sinal. Essa região é definida como *região de transição*, na qual a resposta ideal possui uma *função de transição* que conecta suavemente as respostas situadas na faixa passante e na faixa de rejeição (essa propriedade facilita a implementação de filtros reais com a mesma característica de resposta em freqüência).

A terceira possibilidade não define a resposta em freqüência na região de transição, chamada de *faixa de transição.* Os três casos considerados são estudados por diferentes métodos e critérios de projetos de filtros. Por isso,muitos autores particionam o projeto de um filtro digital em etapas distintas, conforme mostra a próxima subseção.

### 3.1.5 Fases do projeto de filtros

A ferramenta fundamental da filtragem digital é a aproximação da resposta em freqüência; trata-se da primeira fase das estapas de projeto de filtros, como caracterizado por Burrus, Serra e outros [30, 33, 36, 41, 34].

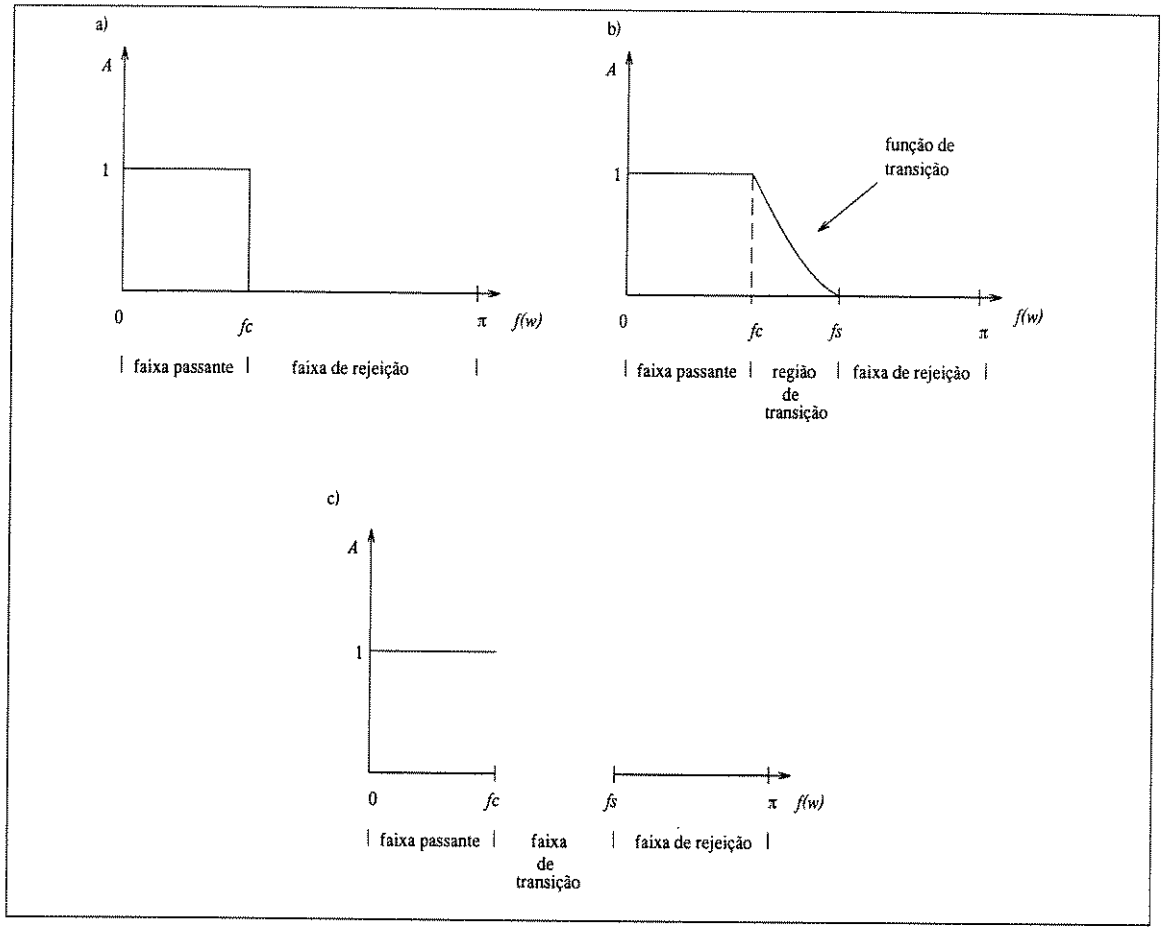


Figura 3.8: Respostas em freqüência de um filtro não recursivo passa-baixa ideal

Para Burrus [30] o processo de projeto de filtros é dividido em duas partes: solução do problema de aproximação e construção do filtro em si. O problema de aproximação trata da escolha dos parâmetros ou coeficientes da função de transferência do filtro para aproximar uma resposta ideal, ou desejada. Essa aproximação é freqüentemente realizada através do uso da resposta em freqüência. A parte do projeto que cuida da construção do filtro trata da escolha de uma estrutura adequada para implementar a função de transferência. Esta estrutura pode estar na forma de um diagrama de circuito, se o filtro é construído de componentes, ou um programa a ser usado por um computador ou um microprocessador dedicado a processamento de sinais.

Na etapa referente à resolução do problema de aproximação, deve-se especificar uma função de transferência e obter seus parâmetros através de quatro passos:

1. Escolha de uma função de resposta ideal (ou desejada), geralmente no domínio da freqüência;
2. Escolha de uma classe realizável de filtros (por exemplo, filtros não recursivos de comprimento $M + 1$);
3. Escolha da medida de qualidade da aproximação (por exemplo, o erro máximo no domínio da freqüência);
4. Seleção de um método ou algoritmo para encontrar a melhor função de tranferência do filtro (no caso deste trabalho, programação linear).

Ao contrário de Burrus [30], Serra [41] caracteriza o projeto de filtros em três fases. A fase inicial do projeto realiza a tarefa de aproximação, que tem por meta a construção das funções que descrevem (matematicamente) o comportamento que se espera do filtro. A fase seguinte, é a implementação, por meio da qual se configura o circuito do filtro e se calcula os valores de seus elementos. Na fase final, procede-se à análise do filtro implementado, fazendo-se as correções dos cálculos efetuados e, verificando-se a influência das perdas, e de outros fatores que afetam, em maior ou menor grau, o desempenho do filtro real.

No presente trabalho, busca-se resolver o problema de aproximação de forma que haja a maior fidelidade possível ao filtro ideal. Utiliza-se o método Simplex Revisado, objeto de estudo do próximo capítulo.

A próxima seção aborda o aspecto de recursividade em filtros digitais. Em seguida, discute-se a aplicação dos conceitos de aproximação de funções ao projeto de filtros digitais. A parte final deste capítulo faz uma breve revisão de trabalhos nas áreas de projeto de filtros e aproximações de funções.

## 3.2 Filtros Digitais Não Recursivos e Filtros Digitais Recursivos

Um filtro digital pode realizar apenas combinações lineares dos dados de entrada ou incluir manipulações dos dados de saída. No primeiro caso, os filtros são denominados não recursivos. Quando incluem o processamento dos dados de saída, os filtros são chamados recursivos.

A escolha entre usar um filtro não recursivo (FIR filters) ou recursivo (IIR filters) depende da aplicação e dos recursos disponíveis para a implementação.

| Casos | $H(\theta)$ |
|---|---|
| Caso 1 - $M$ ímpar<br>Resposta em freqüência simétrica | $\sum_{k=0}^{\frac{M-1}{2}} a_k \cos(\theta k)$ |
| Caso 2 - $M$ par<br>Resposta em freqüência simétrica | $\sum_{k=1}^{\frac{M}{2}} b_k \cos[\theta(k-\frac{1}{2})]$ |
| Caso 3 - $M$ ímpar<br>Resposta em freqüência assimétrica | $\sum_{k=1}^{\frac{M-1}{2}} c_k \sin(\theta k)$ |
| Caso 4 - $M$ par<br>Resposta em freqüência assimétrica | $\sum_{k=1}^{\frac{M}{2}} d_k \sin[\theta(k-\frac{1}{2})]$ |

Tabela 3.1: Simetria e comprimento para o filtro atingir fase linear

### 3.2.1 Filtros Digitais Não Recursivos

Os filtros digitais não recursivos são também chamados de *filtros digitais com resposta de impulso de duração finita* ou *FIR filters* (em inglês). Entre as boas características desses filtros, pode-se destacar:

- Filtros FIR permitem uma resposta de fase linear.
- Possuem estabilidade garantida [36].

A duração ou o comprimento da seqüência da resposta ao impulso destes filtros é, por definição, finita. Esses filtros são definidos pela fórmula linear:

$$y_n = \sum_{k=0}^{M} h_k x_{n-k}, \tag{3.22}$$

sendo que os coeficientes $h_k$ são as constantes do filtro, $x_{n-k}$ são os dados de entrada e $y_n$ os dados de saída. A equação (3.22) é uma forma particular da função de aproximação na forma geral linear (eq. (2.4)), definida no capítulo 2,

$$g_a(x) = \sum_{i=0}^{M} a_i r_i(x),$$

note que $h_k$ e $x_{n-k}$ (3.22) correspondem, respectivamente, a $a_i$ e $r_i$ na equação acima.

Existem quatro possibilidades de um filtro FIR alcançar a propriedade de fase linear. Essas possibilidades estão relacionadas ao comprimento do filtro ($M+1$, ou ordem $M$) e a

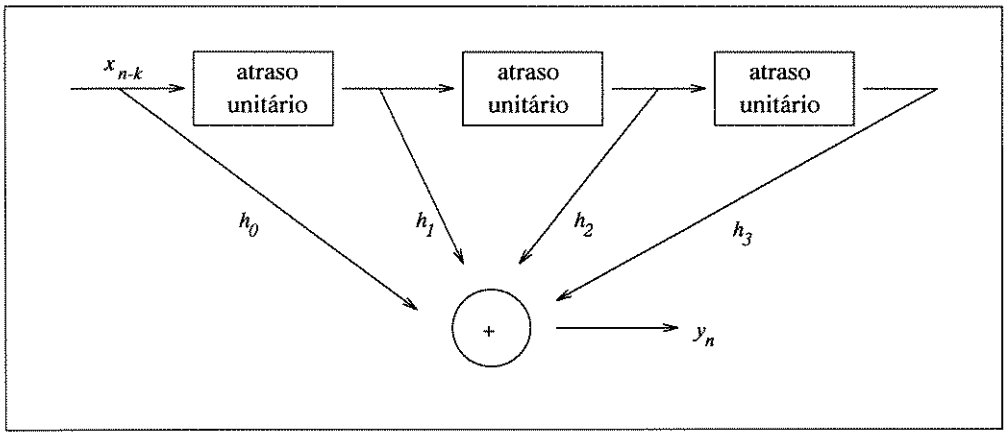


Figura 3.9: Modelo de um filtro digital não recursivo

sua simetria, conforme mostra a tabela 3.1. Comparando-se as funções da tabela 3.1 com a equação (3.22), observa-se que os temos $a_k$, $b_k$, $c_k$ e $d_k$ correspondem aos coeficientes $h_k$; os termos trigonométricos correspondem a $x_{n-k}$.

O problema básico do projeto de filtros não recursivos é a determinação do número de termos ($M$), e valores dos termos $h_k$, para que se alcance o efeito desejado. Existem vários métodos para projetar um filtro não recursivo em geral, na fase de aproximação são usadas três medidas de erro [30]. Uma das medidas é o erro quadrático médio da aproximação da resposta em freqüência; outra medida utilizada é o erro máximo sobre regiões especificadas da resposta em freqüência; a terceira medida é baseada na aproximação por séries de Taylor para a resposta desejada. O método baseado na primeira medida de erro é chamado de Aproximação por Mínimos Quadrados, o segundo é denominado Aproximação por Chebyshev e o terceiro Método de Butterworth.

A Figura 3.9 ilustra um filtro não recursivo para o qual $h_k = 0$ para $k < 0$ e para $k > 3$. A saída do filtro é dada por:

$$\begin{aligned} y_n &= \sum_{k=0}^{3} h_k x_{n-k} \\ &= h_0 x_n + h_1 x_{n-1} + h_2 x_{n-2} + h_3 x_{n-3}. \end{aligned}$$

### 3.2.2 Filtros Digitais Recursivos

Filtros Digitais Recursivos são também chamados de *filtros digitais com resposta de impulso de duração infinita* ou "*IIR filters*". Esses filtros são denominados recursivos porque seus dados de saída ($y_n$) são calculados a partir da entrada ($x_n$) e dados prévios de entrada e saída ($x_{n-k}$ e $y_{n-k}$, respectivamente). Isto é, a saída de um filtro digital recursivo é dada por:

$$y_n = \sum_{k=0}^{M} h_k x_{n-k} + \sum_{k=1}^{M} d_k y_{n-k}, \tag{3.23}$$

onde os coeficientes $h_k$ e $d_k$ são constantes para cada $k$.

Esses filtros são geralmente mais econômicos no tempo de execução e na necessidade de armazenamento de dados, comparando-os com os não recursivos. Porém, filtros recursivos provocam distorções de fase o que pode ser inaceitável em algumas aplicações.

Alguns métodos para o projeto de filtros digitais recursivos são baseados na conversão de Butterworth, Chebyshev (I e II) [30]. As características dessas aproximações são baseadas na combinação de séries de Taylor e aproximação de Chebyshev nas faixas passante e de rejeição.

A Figura 3.10 ilustra um filtro recursivo, construído a partir do filtro não recursivo representado na Fig. 3.9. Neste caso,

$$\begin{aligned} y_n &= \sum_{k=0}^{3} h_k x_{n-k} + \sum_{k=1}^{2} d_k y_{n-k} \\ &= h_0 x_n + h_1 x_{n-1} + h_2 x_{n-2} + h_3 x_{n-3} + d_1 y_{n-1} + d_2 y_{n-2}. \end{aligned}$$

## 3.3 Aplicação do problema de aproximação de funções no projeto de filtros digitais

A meta do problema de aproximação é obter os coeficientes $h_k$ e $d_k$, para $0 \leq k \leq M$, tal que a função da resposta em freqüência satisfaça as especificações do filtro a ser projetado. O número de métodos para aproximar funções é muito grande, assim como a possibilidade de empregá-los no problema da aproximação da resposta em freqüência de um filtro digital – a primeira fase do projeto de filtros (seção 3.1.5).

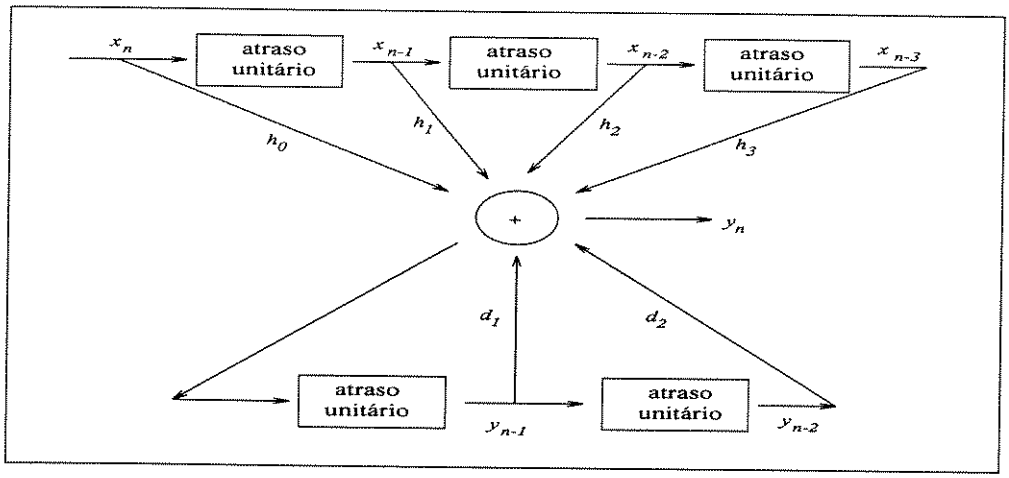


Figura 3.10: Modelo de um filtro digital recursivo

No projeto de filtros digitais os métodos de aproximação podem ser divididos em duas categorias, destinados, respectivamente aos filtros não recursivos (FIR) e aos filtros recursivos (IIR).

Este trabalho discute o uso de programação linear para aproximar um filtro FIR de fase linear, embora a aproximação de Chebyshev seja citada pela literatura consultada. Portanto, a seguir é dada uma breve apresentação da aproximação de Chebyshev.

### 3.3.1 Aproximação de Chebyshev

A teoria da aproximação de Chebyshev, quando aplicada ao problema de projetar um filtro, fornece algoritmos para encontrar os coeficientes de um filtro FIR de fase linear que tenha uma resposta em freqüência de valor mínimo para o erro máximo. Uma aproximação que minimize o erro máximo sobre um conjunto de freqüências é chamada uma *aproximação de Chebyshev.*

Esta aproximação é feita num padrão minimax, onde o erro existente entre as funções ideal e a aproximada é medido pela norma $L_\infty$. Ou seja, dada uma função ideal $H_{id}$, deseja-se minimizar o erro máximo produzido entre ela e a função de aproximação $H_{ap}$, definida abaixo:

$$H_{ap} = \sum_{i=0}^{M} a_i \cos(2\pi i\theta). \tag{3.24}$$

Sendo $M$ a ordem do filtro (ou $M+1$ o seu comprimento), $a_i$ os coeficientes do filtro e $\theta$ pontos do intervalo onde a freqüência é definida.

Usando-se o erro máximo absoluto ponderado definido no capítulo anterior (equação 2.61), tem-se:

$$\epsilon_{max} = \max_{\theta} \; w(x_\theta) \mid H_{id}(\theta) - H_{ap}(\theta) \mid . \tag{3.25}$$

Dessa forma, o problema de projetar um filtro digital FIR com fase linear como um problema de aproximação de Chebyshev, pode ser formulado matematicamente como $||E(\theta)||$,

$$||E(\theta)|| = \min_{coeficientes} \left[ \max_{\theta \in A} \; E(\theta) \right]. \tag{3.26}$$

Este problema, chamado de problema de aproximação de Chebyshev para os filtros FIR, minimiza o desvio máximo sobre um conjunto de freqüências. Ele conduz diretamente a uma caracterização de filtro ótimo em termos do teorema de alternação, que declara a existência de uma aproximação de Chebyshev ótima e única; o erro (ponderado) deste filtro ótimo possui, necessariamente, característica minimax. Este teorema, enunciado abaixo, pode ser encontrado em Burrus, Roberts e outros [30, 36, 34].

**Theorem 3 (Teorema de Alternação)** *Se $H_{ap}(\theta)$ é uma combinação linear de $M+1$ funções cossenos, isto é, se*

$$H_{ap}(\theta) = \sum_{i=0}^{M} a_i \cos(2\pi i \theta), \tag{3.27}$$

*então uma condição necessária e suficiente para que $H_{ap}(\theta)$ seja única e a melhor aproximação de Chebyshev ponderada para uma dada função contínua $H_{id}(\theta)$, sobre $A$, um subconjunto compacto sobre $(0, 0.5)$, é que a função erro ponderada $E(\theta) = w(\theta) \mid H_{id}(\theta) - H_{ap}(\theta) \mid$ exiba no mínimo $(M+2)$ freqüências extremas em $A$, isto é, devem existir $(M+2)$ pontos $\theta_i$ em $A$ tal que*

$$\theta_0 < \theta_1 < \ldots < \theta_M < \theta_{M+1}. \tag{3.28}$$

*Para estes pontos:*

$$E(\theta_i) = -E(\theta_{i+1}), \qquad i = 0, 1, 2, \ldots, M+1 \tag{3.29}$$

*e*

$$\mid E(\theta_i) \mid = \max_{\theta \in A} \; E(\theta). \tag{3.30}$$

Os $M+2$ pontos ordenados da lista (3.28) são os pontos onde o erro $E(\theta)$ tem seus valores extremos. O teorema de alternação fornece elementos para o reconhecimento de uma solução ótima, mas não mostra diretamente como escolher os coeficientes do filtro. Desta forma, dependo da interpretação deste teorema, tem sido criadas várias técnicas para obter essa solução ótima [36, 30, 33, 34].

## 3.4 Revisão da Literatura

O objetivo central deste trabalho está na proposta de aproximar uma função através de técnicas de programação linear, aplicado ao projeto de filtros digitais FIR com fase linear; especificamente, ao problema de aproximar a função da resposta em freqüência do filtro a ser projetado. A maioria dos trabalhos pesquisados tratam do projeto de filtros digitais. Seus enfoques principais são descritos a seguir.

### 3.4.1 Trabalhos em Aproximação de Funções e Projeto de Filtros

No início de 1958 JAMES E. KELLEY [23] formulou o problema de aproximação usando a norma $L_\infty$ e, especificando a função a ser aproximada $f(x)$ e a de aproximação $g_j(x)$, $0 \leq j \leq n$ como reais e contínuas sobre o intervalo fechado $S$ do eixo real, considerando o problema sobre um conjunto finito de pontos desse intervalo. Ele resolveu o problema através de polinômios, isto é, fazendo $g_j(x) = x^j$. Mesmo usando funções de formas especiais, assegurou a solução do problema para funções mais gerais.

KELLEY apresenta um algoritmo detalhado dos passos do Método Simplex que resolve a forma dual do problema de aproximação. Segundo o autor, sua principal vantagem é a possibilidade de aumentar os valores do grau do polinômio sem a necessidade de iniciar novamente o processo, se o grau usado anteriormente não produzir resultado satisfatório.

Em 1961, WARD [48] propôs o uso de programação linear no problema de aproximação polinomial, como uma forma eficiente de resolver esse tipo de problema. Ele visava usar essa proposta para introduzir a programação linear num curso de análise numérica.

A aproximação da função da resposta em freqüência de um filtro digital FIR com fase linear tem sido estudada desde o início da década de 70. Os autores propõem diferentes restrições para o problema, como também várias maneiras de abordá-lo.

TUFTS, RORABACHER e MOSIER [46], em 1970, estudaram formas para projetar filtros digitais de maneira simples. Esses autores usaram a transformada discreta de Fourier para algumas especificações de filtros e programação linear para outras.

HELMS [17] em 1971 estudou diversas técnicas através das quais se pode determinar os coeficientes de um filtro digital. Fez também um levantamento da técnica mais apropriada para as especificações do filtro. Entre esses métodos estão a programação linear (através do método

Simplex), programação não linear e programação inteira.

Uma pesquisa detalhada sobre o emprego de programação linear no projeto de filtros digitais FIR com fase linear foi realizada por RABINER [31, 32] em 1972. Ele formulou o problema restringindo as freqüências das faixas passante e de rejeição a limites superiores e inferiores para cada faixa. Também mostrou a possibilidade de empregar programação linear para projetar filtros com restrições simultâneas sobre as respostas em freqüência e no tempo, como ainda para projetar filtros FIR bidimensionais.

Em 1973, SCHAFER e RABINER [39] fizeram comparações entre os filtros digitais FIR e IIR definindo qual dos dois é o mais adequado para realizar interpolação. Os filtros digitais FIR foram considerados os mais apropriados, sobretudo por atingirem facilmente a característica de fase linear. Em seguida eles abordaram interpolações clássicas, como as interpolações linear e de Lagrange, cujas vantagens e desvantagens foram discutidas. Mencionaram ainda que através de técnicas de otimização linear é possível alcançar interpolações significativamente melhores – os testes realizados mostraram que o filtro ótimo projetado com programação linear foi melhor do que o obtido com Lagrange.

Técnicas para resolver o problema de aproximação que determina os coeficientes do filtro, através da minimização de uma medida de desempenho foram discutidas por RABINER, McCLELLAN e PARKS [34], em 1975. Em especial, esses autores estudaram a aproximação ponderada de Chebyshev e suas extensões, como por exemplo o algoritmo de trocas de Remez. Também abordaram as técnicas de programação linear para a solução deste problema de aproximação restrito, como havia sido descrito por RABINER [31, 32], em 1972. Entre as conclusões, consideraram programação linear como o método mais simples para resolver este problema quando restrições de resposta no tempo são adicionadas.

Em 1979, STEIGLITZ [42] usou programação linear para projetar filtros FIR cuja resposta em freqüência na faixa passante é monotonicamente decrescente (ou crescente); restrições extras no domínio da freqüência podem ser adicionadas ao problema, sem causar dificuldades na solução. Segundo o autor, a grande vantagem de programação linear sobre o algoritmo de trocas de Remez, por exemplo, está na generalidade inerente à formulação do problema. O Método das Duas Fases e o Simplex Revisado foram usados para resolver o Dual desse problema, definindo uma regra para evitar a degenerescência, (característica comum no dual).

SAMUELI [38] em 1988 apresentou um algoritmo melhorado de programação linear para o projeto de filtros digitais FIR. Ele procurou evitar um número grande de divisões do intervalo de interesse, sem prejudicar o resultado final. Para isso, um problema de programação linear foi caracterizado com uma única inequação para cada freqüência extrema na faixa de rejeição

da resposta. Como a posição das freqüências extremas na faixa de rejeição não são conhecidas *a priori*, ele usou uma aproximação iterativa, usando o menor número possível de pontos. A cada iteração do problema, os pontos de amostragem da freqüência no intervalo são ajustados de forma a coincidirem com os extremos calculados, na faixa de rejeição da resposta.

JOHNSON [21], em 1990, mostrou a possibilidade de encontrar uma solução básica factível para o problema de aproximação da resposta em freqüência, sem o uso de variáveis artificiais. Esta técnica reduz significativamente o número de iterações necessárias para chegar à solução ótima (comparando-se, por exemplo, com o método das duas fases). JOHNSON alterou o programa de STEIGLITZ [42] para eliminar a fase I e, onde necessário, introduziu restrições no domínio do tempo. Usou as mesmas especificações que STEIGLITZ [42] para projetar um filtro, comparando o desempenho do método criado por ele com o método das duas fases; obteve uma boa redução no número de iterações e no tempo gasto para a solução do problema.

STEIGLITZ, PARKS e KAISER [43], em 1992, usaram o algoritmo simplex, com duas fases, para encontrar um filtro com fase linear, e de comprimento mínimo, que atinja os limites prescritos sobre a resposta em freqüência. Para isso, maximizaram a distância da função resposta em freqüência às restrições de limite – segundo os autores, a maximização foi adotada para evitar a ocorrência de ciclagem no programa. O algoritmo desenvolvido é abrangente, criando a possibilidade de projetar filtros com especificações diferentes. Por exemplo, pode-se ajustar o tamanho das faixas de rejeição e passante, quando o comprimento do filtro é fixo. Restrições adicionais, como a monotonicidade da resposta para dar a característica de magnitude fixa, podem ser impostas nas faixas apropriadas de freqüência. O principal programa implementado é o METEOR.

Um algoritmo capaz de projetar filtros com coeficientes complexos ou reais foi proposto por BURNSIDE [6] em 1995. Esse autor usou uma variante do método Simplex, com a técnica criada por STEIGLITZ [42] para evitar a ciclagem. Filtros de ordem muito grande ($M = 1000$) foram projetados com esta técnica.

### 3.4.2 Comparação com a metodologia Desenvolvida

Estre trabalho pretende examinar a viabilidade do uso de programação linear para resolver problemas de aproximação de funções. Para isso, é realizado um estudo sobre o problema de projetar filtros digitais FIR com fase linear com o objetivo de utilizar PL para resolvê-lo. A seguir são realizadas algumas comparações entre os trabalhos citados e o desenvolvido.

KELLEY [23] e WARD [48] propuseram o uso de PL para o problema de aproximação de funções. Este trabalho além de propor PL para esse problema, aplica-o no problema de projetar filtros digitais.

TUFS, RORABACHER e MOSIER [46] e HELMS [17] fizeram um levantamento de qual técnica de aproximação é mais apropriada para determinadas especificações de filtro. Este trabalho enfatiza que o método Simplex é bom e flexível.

A formulação matemática, do problema de projeto de filtros FIR com fase linear, considerada neste trabalho é a mesma feita por RABINER [31, 32], STEIGLITZ [42], SAMUELI [38], JOHNSON [21] e STEIGLITZ, PARKS e KAISER [43].

STEIGLITZ [42] e STEIGLITZ, PARKS e KAISER [43], consideram restrições adicionais sobre a monotonicidade da resposta em freqüência do filtro. Essas restrições não são usadas nos estudos deste trabalho. No entanto a flexibilidade da abordagem adotada permite considerá-las sem maiores dificuldades. Esses autores iniciam o Método Simplex através do Método das Duas Fases; neste trabalho o método utilizado é o Big-M, como é descrito em maiores detalhes no capítulo 5. JOHNSON [21] mostrou a possibilidade de iniciar o Método Simplex sem o uso de variáveis artificiais.

Um problema comum apresentado pelo dual desse problema é a degenerescência. STEIGLITZ [42] apresenta, no seu trabalho de 1979 uma regra simples com bons resultados para driblar a ciclagem. Em outro trabalho STEIGLITZ, PARKS e KAISER [43] evitam a ciclagem ao optar por maximizar a distância da resposta em freqüência aos limites superiores e inferiores prescritos. Neste trabalho, o problema de ciclagem foi evitado através de uma regra simples, não encontrada na literatura consultada – a regra é detalhada no capítulo 5.

SAMUELI [38] observou que a matriz das restrições é mal condicionada quando o número de pontos discretizados no intervalo de interesse aumenta. Ele procurou contornar esse problema por redução nas divisões do intervalo sem sacrificar a resposta final. Algumas considerações sobre o mal condicionamento da matriz são discutidas no próximo capítulo.

STEIGLITZ, PARKS e KAISER [43] colocaram em domínio público os programas produzidos para o trabalho publicado em 1992. Esses programas foram usados nas comparações feitas no capítulo 5.

Maiores informações sobre o emprego do Método Simplex Revisado no problema de aproximação de funções, assim como as formulações do primal e dual, são apresentadas no próximo capítulo.

# Capítulo 4

# Utilização do Simplex Revisado em Aproximação de Funções

Neste capítulo será estudada uma das técnicas mais tradicionais e populares para a solução de problemas de otimização linear, o Método Simplex (Revisado). Em primeiro lugar haverá uma descrição histórica do surgimento do Método Simplex seguido de sua forma compacta, o Simplex Revisado. Na seqüência é feita a formulação do problema de aproximar a resposta em freqüência de um filtro digital, e, por último, a análise do método escolhido para a solução desse problema.

## 4.1 História da Programação Linear

O problema de programação linear foi estudado inicialmente por George B. Dantzig, em 1947, enquanto ele trabalhava para a Força Aérea do Estados Unidos, resultando no desenvolvimento do Método Simplex. O termo *Programação Linear* foi sugerido por T. C. Koopmans, em 1951, como uma alternativa para a forma inicial "Programação numa estrutura linear" [9].

Desde a origem desse método, muitas pessoas têm contribuído para o crescimento da programação linear através do desenvolvimento da sua teoria matemática, criando novos algoritmos, explorando novas aplicações e a usando como ferramenta para resolver problemas mais complexos, como programas discretos, programas não lineares, problemas combinatoriais, problemas de programação estocásticas e problemas de controle ótimo. O método Simplex possui, teoricamente, tempo de convergência exponencial em análise do pior caso (os algoritmos com

tempo de convergência exponencial são evitados, quando possível); mas na prática seu desempenho é muito bom (ou seja, a convergência na prática é muito melhor do que a convergência no pior caso). Em 1975, L. V. Kantorovich e T. C. Koopmans receberam o prêmio Nobel de Ciências Econômicas por "suas contribuições para a teoria de distribuição ótima de recursos [12]. Em 1979, L. G. Khachian provou que o Método dos Elipsóides [1] encontra uma solução ótima de um problema de programação linear num tempo polinomial, mas infelizmente a superioridade teórica desse método não pode ser realizada em aplicações práticas [12]. Em 1984, N. Karmarkar divulgou o Método Projetivo que além de ter um desempenho superior ao Método Simplex também possui um grande potencial para resolver problemas práticos de grande escala. Ele trabalha de forma diferente do Método Simplex, pois aproxima a solução através do interior da região factível, é uma aproximação de pontos interiores. Ultimamente, novos algoritmos de pontos interiores foram propostos a partir do conceito de trajetória central, que se tornou tema de interesse de muitos pesquisadores [9].

A concentração de pesquisas sobre os métodos de otimização linear, é motivado pelo grande número de problemas existentes nas áreas industrial, econômica, médica entre várias outras.

No próximo item discute-se a formulação padrão e as variantes de um problema de programação linear e como opera o Método Simplex (Revisado).

## 4.2 O Método Simplex

Um problema de programação linear na *forma padrão* pode ser escrito da seguinte maneira:

$$\begin{array}{lrcl} \min & z = c_1x_1 + c_2x_2 + \cdots + c_nx_n & & \\ \text{sujeito a} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & b_2 \\ & & \ddots & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & = & b_m \\ & x_1, x_2, \ldots, x_n & \geq & 0. \end{array} \tag{4.1}$$

O problema (4.1) também pode ser formulado usando a notação matricial, da forma:

$$\begin{array}{lrcl} \min & z = \mathbf{c}^t\mathbf{x} & & \\ \text{s. a} & \mathbf{Ax} & = & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0}, \end{array} \tag{4.2}$$

[1] Proposto por N. Z. Shor, D. B. Yudin e A. S. Nemirovskii

$\mathbf{c}^t$ representa um vetor transposto $\mathbf{c}^t = (c_1, \ldots, c_n)$; $\mathbf{x}$, $\mathbf{b}$ são vetores colunas

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \tag{4.3}$$

$\mathbf{A}_{m,n}$ uma matriz representada por:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ & \vdots & & \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}. \tag{4.4}$$

A solução ótima visa encontrar um vetor $\mathbf{x}$, entre todos os vetores factíveis, que minimiza a função objetivo e satisfaz as restrições do problema.

O problema de programação linear pode minimizar ou maximizar uma função objetivo linear e as restrições podem ser equações e/ou inequações, as quais são facilmente manipuladas para se transformarem de uma forma para outra. Essas manipulações são muito úteis na programação linear, principalmente para deixar o problema na forma padrão, exigida pelo Método Simplex.

A decomposição do sistema (4.2), em básico (B) e não básico (N), é necessária para o mecanismo do Método Simplex. Supondo que a matriz $\mathbf{A}_{mn}$ tenha posto completo (posto de $\mathbf{A} = m$) e $m < n$, pode-se selecionar $m$ colunas LI que formam uma matriz base $\mathbf{B}_{mm}$ não singular. Sem perda de generalidade a partição de $\mathbf{A}$ pode ser feita como:

$$\mathbf{A} = \left[\, \mathbf{B} \mid \mathbf{N} \,\right], \qquad \mathbf{x} = \begin{bmatrix} \mathbf{x}_B \\ \mathbf{x}_N \end{bmatrix}, \qquad \text{e} \qquad \mathbf{c} = \begin{bmatrix} \mathbf{c}_B \\ \mathbf{c}_N \end{bmatrix} \tag{4.5}$$

sendo $\mathbf{B} \in \Re^{mm}$ a matriz base descrita anteriormente, $\mathbf{N} \in \Re^{m(n-m)}$ a *matriz não básica*. As *variáveis básicas* são representadas pelo vetor $\mathbf{x}_B$ e as *variáveis não básicas* por $\mathbf{x}_N$. Reescrevendo o sistema (4.2), tem-se:

$$\mathbf{B}\mathbf{x}_B + \mathbf{N}\mathbf{X}_N = \mathbf{B} \qquad \Rightarrow \qquad \mathbf{I}_B\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N, \tag{4.6}$$

ao substituir (4.6) na função objetivo de (4.2), obtem-se:

$$z = \mathbf{c}_B^t\mathbf{x}_B + \mathbf{c}_N^t\mathbf{x}_N$$

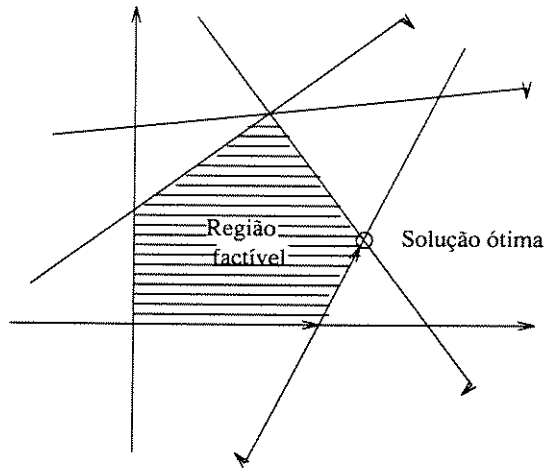


Figura 4.1: Interpretação geométrica de um problema de programação linear

$$
\begin{aligned}
&= \mathbf{c}_B^t(\mathbf{B}^{-1}\mathbf{b} - \mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N) + \mathbf{c}_N^t\mathbf{x}_N \\
&= \mathbf{c}_B^t\mathbf{B}^{-1}\mathbf{b} - \mathbf{c}_B^t\mathbf{B}^{-1}\mathbf{N}\mathbf{x}_N + \mathbf{c}_N^t\mathbf{x}_N \\
&= \bar{z} + \sum_{j \in I(N)} (c_j - z_j)x_j, \qquad (4.7)
\end{aligned}
$$

sendo $I(N)$ o conjunto dos índices das colunas não básicas e $z_j = c_B^t\mathbf{B}^{-1}\mathbf{a}_j$, com $\mathbf{a}_j$ representado a *j-ésima* coluna não básica da matriz $\mathbf{A}$.

Diz-se que o vetor $\mathbf{x}$ é uma *solução básica* de (4.2) quando em (4.6) $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$ e $\mathbf{x}_N = 0$; se $\mathbf{x}_B \geq 0$ então a solução básica é uma *solução básica factível* para o sistema.

O *Teorema Fundamental da Programação Linear* [4, 26] estabelece que, "se existe uma solução ótima para o problema, existirá uma solução básica ótima". Assim pode-se restringir a procura de soluções ótimas ao conjunto de soluções básicas factíveis.

A interpretação geométrica (figura 4.1) de um problema de programação linear leva a delimitar uma região, chamada *região factível*, através das restrições do problema que formam um conjunto convexo ($\wp = \{x \in \Re^n | \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq 0\}$) onde estão as suas soluções factíveis. Todo ponto extremo desse conjunto corresponde a uma solução básica factível; a solução básica factível ótima será alcançada quando o Método Simplex tiver "caminhado" pelos pontos extremos que melhoram a função objetivo, encontrando o ponto ótimo.

Deve-se enfatizar que o Método Simplex não precisa explorar todos os pontos extremos factíveis

para chegar à solução básica factível ótima. Por exemplo, num problema de minimização como o (4.2), ao partir de uma solução básica factível ele evolui iterativamente através dessas soluções, melhorando a função objetivo. A cada iteração a condição de otimalidade é verificada, isto é, se para cada variável não básica $(x_j)$ os $(c_j - z_j)$, $j \in I(N)$ de (4.7) são maiores ou iguais a zero (ou seja, aumentam o valor da função objetivo) então a solução corrente é ótima; caso contrário uma variável não básica que melhora valor da função objetivo se torna básica. Em seguida, são feitas as atualizações necessárias para a continuação deste processo iterativo.

Se o problema, na forma padrão, não possui uma base inicial factível $\mathbf{B}$ (geralmente a identidade), então é aplicado um mecanismo que encontra uma solução básica factível inicial que possibilitará o procedimento iterativo de buscar a solução ótima. Existem vários mecanismos que fazem isso, entre eles o Método das Duas Fases e o Método do Big-M [4]. Esses métodos introduzem variáveis artificiais no problema quando não existe uma solução inicial factível para começar o método Simplex. O processo do método Simplex é iniciado com essas variáveis artificiais fazendo parte da base inicial factível ($\mathbf{B}$), e termina quando elas não pertencem mais a essa base. Neste ponto do procedimento é encontrada uma base factível para o problema, que contém somente as variáveis originais. As variáveis artificiais são desconsideradas no algoritmo e o método Simplex é processado até encontrar a solução ótima para o problema.

### 4.2.1 O Algoritmo Simplex

Tendo encontrado uma base inicial factível com $x_B \geq 0$, $x_N = 0$, o Método Simplex pode ser iniciado. Ele é resumido na seqüência de passos abaixo.

- **Passo 1** - Resolva o sistema $\mathbf{B}\mathbf{x}_B = \mathbf{b}$.
  $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b} = \bar{\mathbf{b}}$, $\mathbf{x}_N = 0$ e $z = \mathbf{c}_B\mathbf{B}^{-1}\mathbf{b} = \mathbf{c}_B\mathbf{x}_B$.
- **Passo 2** - Calcule os custos relativos referente a todas as variáveis não básicas, ou seja:

$$\mathbf{c}_N^t - \mathbf{c}_B^t\mathbf{B}^{-1}\mathbf{N} = \sum_{j \in I(N)} (c_j - z_j)x_j. \tag{4.8}$$

  Verifique se existe algum $j$ tal que $c_j - z_j \leq 0$. Se não existir, pare; a solução é ótima. Senão, selecione uma variável $x_k$ para entrar na base sendo que,

$$c_k - z_k = \min_{j \in I(N)} \{c_j - z_j\}. \tag{4.9}$$

- **Passo 3** - Observe a coluna $k$ da matriz de restrições **A**. Se todos os elementos dessa coluna ( $\mathbf{y}_k$) forem menores ou iguais a zero, $\mathbf{y}_k \leq 0$ , pare; a solução é ilimitada. Se ao menos um elemento nessa coluna $k$ tal que $\mathbf{y}_k > 0$, prossiga.
- **Passo 4** - A variável $x_k$ deve entrar na base; a variável de bloqueio $x_r$ que deixará a base deve ser identificada através do teste da razão:
  $\frac{\bar{b}_r}{y_{rk}} = \min\{\frac{\bar{b}_i}{y_{ik}} \mid y_{ik} > 0\}$
- **Passo 5** - Atualize a matriz **A** efetuando operações de pivoteamento. O elemento $a_{rk}$ deve ser reduzido à unidade, enquanto $a_{ik}$, $i \neq r$ é reduzido a zero. Ou seja:

  `Quando` $i \neq r$ `faça`
  $\bar{a}_{ij} = a_{ij} - \frac{a_{rj}a_{ik}}{a_{rk}}$
  `Senão, faça`
  $\bar{a}_{rj} = \frac{a_{rj}}{a_{rk}}$.

  Atualize os custos relativos e o vetor **b**.
- **Passo 6** - Repita o algoritmo a partir do Passo 2 até o processo parar no Passo 2 ou 3.

O Método Simplex realiza (no Passo 3) uma grande quantidade de cálculos (e os armazena); uma boa parte desses cálculos não são usados para conclusão da iteração nos passos seguintes. O Método Simplex Revisado é um procedimento para a implementação dos passos do Método Simplex, que procura economizar cálculos e requisitos de memória.

A descrição dos passos e dos testes realizados pelo método Simplex Revisado será feita na próxima seção. O Simplex Revisado é o algoritmo empregado para aproximar a função da resposta em freqüência de um filtro digital.

## 4.3 Método Simplex Revisado (Primal)

O Método Simplex Revisado é uma técnica computacional eficiente que aplica as idéias principais do Método Simplex. . O Simplex Revisado realiza de forma mais eficiente os passos 2, 3 e 5 do algoritmo Simplex, como é descrito a seguir.

- **Passo 2** - Encontre o vetor das variáveis duais **w**, resolvendo o sistema $\mathbf{wB} = \mathbf{c}_B$. Calcule o custo relativo $c_j$ referente a todas as variáveis não básicas, $j \in I(N)$, da seguinte forma:

$c_j - z_j = c_j - \mathbf{w}\mathbf{a}_j$.

Verifique se existe algum $j$ tal que $c_j - \mathbf{w}\mathbf{a}_j < 0$. Se não existir, pare; a solução é ótima. Senão, selecione uma variável $x_k$ para entrar na base, cujo custo relativo é negativo (em geral o mais negativo entre todos).

- **Passo 3** - Resolva o sistema $\mathbf{B}\mathbf{y}_k = \mathbf{a}_k$. Se $\mathbf{y}_k \leq 0$ para todo $k$, pare, a solução é ilimitada. Se houver ao menos um elemento $k$ para que $\mathbf{y}_k > 0$, prossiga.
- **Passo 5** - Atualize $\mathbf{B}^{-1}$ e as variáveis básicas como segue:

```
Para i = 1 a m faça
    Para j = 1 a m faça
        Se i ≠ r
            B(i,j) ← B(i,j) − B(r,j)y(i)/y(r),
            b̄(i) ← b̄(i) − b̄(r)y(i)/y(r),
        Se i = r
            B(r,j) ← B(r,j)/y(r),
            b̄(r) ← b̄(r)/y(r),
        Fim (se)
    Fim (para j)
Fim (para i)
```

O maior esforço computacional desse método, a cada iteração, está no cálculo das variáveis duais ($\mathbf{w} = \mathbf{c}_B\mathbf{B}^{-1}$), na coluna atualizada ($\mathbf{y}_k = \mathbf{B}^{-1}\mathbf{a}_k$), no teste da razão $\frac{\bar{b}_r}{y_{kr}} = \min\{\frac{\bar{b}_i}{y_{ir}} | y_{ik} > 0\}$ e principalmente na atualização da matriz básica $\mathbf{B}^{-1}$.

## 4.4 Problema Dual

O problema de programação linear primal (4.2), definido pela matriz de restrições $\mathbf{A}$, o vetor do lado direito $\mathbf{b}$ e o vetor custo $\mathbf{c}$, possui outro problema de programação linear ao qual está associado, chamado *Problema Dual* [4], construído com o mesmo conjunto de dados $\mathbf{A}, \mathbf{b}, \mathbf{c}$.

A análise desses problemas são fundamentais na hora de resolvê-los, pois existe relações entre suas soluções como diz o teorema fundamental da dualidade [4].

**Theorem 4 (Teorema fundamental da dualidade)** *Com respeito aos problemas de programação linear primal e dual, apenas uma das afirmações é verdadeira.*

1. *Ambos problemas possuem soluções ótimas $x^*$ e $w^*$ com $cx^* = w^*b$.*
2. *Se um problema tem função objetivo com valor ilimitado, o outro problema deve ser infactível.*
3. *Ambos problemas são infactíveis.*

A teoria da dualidade é extensivamente estudada na literatura especializada em otimização [4, 9, 12, 28, 26].

Considere o problema de programação linear primal na sua forma padrão [2]:

$$\begin{array}{lrl} \min & z = \mathbf{c}^t\mathbf{x} & \\ \text{s. a} & \mathbf{Ax} & = \mathbf{b} \\ & \mathbf{x} & \geq \mathbf{0}, \end{array} \tag{4.10}$$

$\mathbf{A} \in \Re^{mn}, \quad \mathbf{c} \in \Re^n, \quad \mathbf{x} \in \Re^n, \quad \mathbf{b} \in \Re^m.$

De acordo com a teoria da dualidade, existe um problema dual correspondente a esse e através das relações existentes entre eles, o dual assume a forma:

$$\begin{array}{lrl} \max & z = \mathbf{b}^t\mathbf{w} & \\ \text{s. a} & \mathbf{A}^t\mathbf{w} & = \mathbf{c} \\ & \mathbf{w} & \text{irrestrito.} \end{array} \tag{4.11}$$

O vetor $\mathbf{w}$ é a *solução dual* se as restições do problema forem satisfeitas.

A idéia de aproximar uma função através de técnicas de programação linear, nesse caso o Simplex Revisado, visa explorar a estrutura do problema primal e dual resultantes para essa aplicação, procurando encontrar a forma mais adequada para buscar a solução ótima. A análise desses aspectos é o tema da próxima seção.

## 4.5 Caracterização do Problema

No capítulo 2 foram descritas algumas técnicas que são empregadas para resolver o problema de aproximação de funções, assim como algumas considerações (seção 2.5.4) sobre a aplicação

[2] Foi descrito em (4.2) e agora reescrito aqui para facilitar a visualização das alterações ocorridas com o dual.

de Métodos de Programação Linear para esse caso.

O problema de aproximar uma função usando, como definição de otimalidade, o erro máximo absoluto (otimização $L_\infty$), resulta num problema de programação linear, descrito em (2.65). Esta estrutura pode ser usada para aproximar funções contínuas e descontínuas, através de qualquer tipo de função de aproximação, como funções trigonométricas ou polinômios. Essa abrangência sobre o emprego de funções é uma característica bastante importante da programação linear, pois há a possibilidade de resolver problemas que os métodos tradicionais não conseguem; além de permitir que se "trabalhe" a formulação do problema, por exemplo, inserindo novas restrições, definindo pesos para diferentes regiões do intervalo de trabalho. Outra vantagem está na manipulação da própria técnica, que pode ser adaptada às particularidades do problema visando economia de cálculos, de espaço de armazenamento dos dados, enfim a eficiência do algoritmo.

Embora o problema de aproximar funções esteja presente em diversas áreas, este trabalho propõe resolver aproximações de funções via programação linear, com aplicação no projeto de filtros digitais.

Como foi visto no capítulo 3, a ferramenta fundamental do projeto de filtros digitais é a aproximação da resposta em freqüência. Para resolver esse problema, é essencial que sejam feitas algumas especificações como:

1. O filtro a ser projetado é Recursivo ou Não Recursivo?
2. Qual é o tipo de filtro com relação à faixa de freqüência filtrada?
3. Que tipo de resposta de fase deve possuir o filtro desejado?
4. Em que intervalo será feita a aproximação?
5. Qual é o intervalo da região de transição?
6. Haverá alguma restrição que limitará superior ou/e inferiormente a função da resposta em freqüência?
7. Se a questão anterior for afirmativa, em qual faixa existirá a restrição, na passante, de rejeição ou nas duas?
8. Outros...

Há muitos aspectos que podem ser abordados sobre esse problema, os quais são definidos de modo a atender uma necessidade do problema ou requisito do projetista. Para a formulação do problema estudado neste trabalho, considera-se as seguintes características:

1. Filtro Digital Não Recursivo, definido pela equação (3.22);

2. Filtro Digital Não Recursivo Passa-baixa;
3. Filtro Digital Não Recursivo Passa-baixa com fase linear;
4. Aproximação será realizada no intervalo $\{0, 0.5\}$;
5. Faixa de transição será compreendida entre $(0.2, 0.3)$ — Programa SemIntervalo;
6. Região de transição compreendida entre $(0.25 - 10^{-8}, 0.25 + 10^{-8})$ — Programa ComIntervalo.

Para resposta em freqüência simétrica, de um filtro passa-baixa, foi escolhida como função de interpolação uma série de cossenos com intervalo normalizado (conforme a tabela 3.1 no capítulo 3). Desta forma, escreve-se a função de aproximação (2.4) como:

$$g_{\mathbf{a}}(x) = \sum_{i=0}^{M} a_i \cos(2\pi x i). \tag{4.12}$$

Como o problema é considerado de forma discretizada, os pontos são selecionados no intervalo de trabalho pela definição (2.42) apresentada no capítulo 2. O modelo definido em (2.65) é agora reescrito, representando matematicamente o problema primal da aproximação da resposta em freqüência desse filtro:

$$\begin{array}{l} \min\ Z \\ \text{s. a} \\ \left.\begin{array}{rcl} -\sum_{i=0}^{M} a_i \cos(2\pi x_k i) - Z.1 & \leq & -f(x_k) \\ \sum_{i=0}^{M} a_i \cos(2\pi x_k i) - Z.1 & \leq & f(x_k) \end{array}\right\} \quad k = 0, 1, \ldots, N \\ a_i,\ i = 0, 1, \ldots, M \ \textit{livres}. \end{array} \tag{4.13}$$

Observe que o número de colunas da matriz de restrições é $(M + 2)$ e o número de linhas é $(2N + 2)$. O número de pontos discretos no intervalo de interesse é $(N + 1)$.

Se $N$ for aumentado de forma indefinida o número de restrições crescerá significativamente, o que gera um gasto de memória excessivo com cálculos. Isso torna conveniente a análise do problema Dual correspondente:

$$\begin{array}{ll} \max & -\sum_{k=0}^{N} f(x_k) v_k + \sum_{j=0}^{N} f(x_j) v_{N+j+1} \\ \text{s. a} & \\ & -\sum_{k=0}^{N} \cos(2\pi x_k i) v_k + \sum_{j=0}^{N} \cos(2\pi x_j i) v_{N+j+1} = 0,\ i = 0, 1, \ldots, M \\ & -\sum_{k=0}^{2N+1} v_k = 1 \\ & v_k \leq 0,\ \forall k. \end{array} \tag{4.14}$$

Nessa versão (4.14) do problema dual as variáveis de decisão são negativas. Para colocar o problema na forma padrão, torna-se necessária a substituição dessas variáveis.

Fazendo-se $v = -u$, tem-se:

$$u_i \geq 0. \tag{4.15}$$

Após as substituições das variáveis, tem-se o seguinte problema dual:

$$\begin{array}{ll} \max & \sum_{k=0}^{N} f(x_k)u_k - \sum_{j=0}^{N} f(x_j)u_{N+j+1} \\ \text{s. a} & \\ & \sum_{k=0}^{N} \cos(2\pi x_k i)u_k - \sum_{j=0}^{N} \cos(2\pi x_j i)u_{N+j+1} = 0, \ i = 0, 1, \ldots, M \\ & \sum_{k=0}^{2N+1} u_k = 1 \\ & u_k \geq 0, \ \forall k. \end{array} \tag{4.16}$$

Nos próximos itens, faz-se uma análise mais detalhada dos problemas primal (4.13) e dual (4.16), Procurando-se destacar suas características e propriedades. A abordagem desses aspectos é realizada separadamente, iniciando-se com o problema primal.

### 4.5.1 Características do Problema Primal

O problema primal (4.13), possui muitas características que influenciam no desempenho do Método Simplex Revisado, na estrutura, etc.

Como esse problema não está na forma padrão, deverão ser adicionadas $(2N + 2)$ variáveis de folga. Essas variáveis representam o valor necessário, maior ou igual a zero, que cada uma das inequações pertencentes às restrições precisam para se tornarem equações, isto é, para que a somatória do lado direito seja de mesmo valor do apresentado no lado esquerdo. Sendo elas adicionadas ao sistema, formam uma base inicial factível, não sendo necessária a presença de variáveis artificiais para iniciar o processo. Mesmo assim, o tamanho do sistema aumentou em $(2N + 2)$ colunas e se $N$, o número de pontos selecionados no intervalo de trabalho, tiver um valor elevado o aumento será considerável.

Além disso, a grade de pontos selecionados nesse intervalo se torna cada vez mais densa à medida que $N$ cresce. Segundo Samueli [38], este fato pode provocar instabilidade numérica; o alto número de restrições no problema pode, também, causar dificuldades numéricas nas operações de pivoteamento do Método Simplex Revisado.

### 4.5.2 Características do Problema Dual

O problema dual apresentado em (4.16) possui $(M+2)$ restrições e já está na forma padrão. Devido ao fato dele não apresentar uma base inicial factível, deve-se adicionar variáveis artificiais para iniciar o Método Simplex Revisado.

Esse é um problema degenerado, isto é, $(M+1)$ das $(M+2)$ variáveis básicas possuem, no início, valor zero em (4.16). Isso implica que existirão muitos empates no passo 3 do método, o teste da razão. Steiglitz [42], resolveu esse problema escolhendo o maior pivô entre todos aqueles que se encontram nos empates ocorridos no teste da razão. A solução usada para resolvereventuais problemas de convergência devido a empates no passo 4 será discutida no próximo capítulo.

O problema dual é fácil de ser implementado e oferece economia no armazenamento dos dados e nos cálculos realizados pelo Simplex. Além disso, o número necessário de variáveis artificiais que serão introduzidas no sistema é $(M+2)$, ou seja, um valor bem menor que a quantidade de variáveis de folga necessárias no primal. Tanto no dual como no primal há o problema de considerar matriz de restrições mal comportada, conforme é discutido a seguir.

### 4.5.3 Características da Matriz de Restrições

A matriz $\mathbf{A}$, gerada pelas restrições do problema, possui duas características muito importantes, que devem ser avaliadas no instante de resolver o problema.

Samueli [38] verificou que a matriz é mal condicionada quando o filtro é de alta ordem (alto valor de $M$). Entretanto, através dos testes realizados, foi constatado que a matriz de restições só será mal condicionada, à medida que a ordem do filtro $(M)$ cresce, se o problema considerado não definir a aproximação sobre a faixa de transição, como foi mostrado na Figura 3.8 (c). Ou seja, quando no primal/dual não existir as linhas/colunas correspondentes às restrições/variáveis compreendidas na faixa de transição. Caso contrário, quando todos os pontos do intervalo são avaliados, inclusive os da região de transição (Fig. 3.8 (b)), a matriz é muito bem condicionada, mesmo que corresponda a um filtro de alta ordem. O número de condição dessa matriz gira em torno de $\approx 1,45$, o que significa, teoricamente, que a solução encontrada para o sistema é confiável.

Do ponto de vista teórico, um sistema que contém uma matriz mal condicionada pode ou não apresentar distorção nos resultados. Foi observado, no projeto de filtros, um caso em que a

matriz de restrições é mal condicionada, mas "a priori" não há como afirmar que o resultado obtido a partir dessa matriz sofre desvios devido à esse problema. De fato, os resultados encontrados nos testes realizados no capítulo 5 foram bons.

Analisando o problema de projetar um filtro digital sob o aspecto físico, é mais adequado considerar a função da resposta em freqüência apenas sobre a faixa passante e a de rejeição (Fig. 3.8 (b)), mesmo que sob o aspecto computacional exista o problema de realizar cálculos com uma matriz mal condicionada. Neste trabalho, o intervalo de interesse é considerado de ambas as formas.

## 4.6 Análise do Simplex Revisado

O uso de técnicas de programação linear para resolver o problema de aproximação de funções é muito atraente devido à sua versatilidade e maleabilidade. Mesmo não apresentando uma "solução fechada" [3] como várias técnicas apresentam, e como preferem e defendem alguns autores, as vantagens verificadas pelo uso de Programação Linear nesse caso são evidentes e devem ser consideradas.

No problema de projetar filtros digitais, as qualidades do projeto por filtros digitais são observadas a todo momento, principalmente no instante de manipular o modelo matemático que pode ser adequado à diversas situações. Algumas dessas possíveis adaptações do modelo são descritas a seguir.

Rabiner [31] citou que, ao usar o algoritmo de maior passo de descida, as restrições sobre a faixa passante não podem ser mantidas, mas através do uso da Programação Linear, relações entre as faixas passante e de rejeição podem ser estabelecidas. Segundo ele, existem outras vantagens no uso de PL:

- A convergência do procedimento é garantida dentro de um número fixo de iterações;
- Freqüências críticas da resposta desejada podem ser exatamente especificadas;
- O método pode ser usado em problemas de projeto com parâmetros muito variados;
- Finalmente, com a existência de técnicas de Programação Linear Inteira, o problema de projeto pode ser combinado com o problema de avaliar os coeficientes para projetar filtros ótimos com um comprimento prescrito.

[3] Ver capítulo 2

A programação linear permite flexibilidade na modelagem do problema de aproximar a função da resposta em freqüência, pois é possível introduzir ou retirar restrições, e mesmo fixar parâmetros (coeficientes do filtro) em valores determinados.

A abrangência da programação linear permite, por exemplo:

1. Aproximar funções descontínuas (caso dos filtros digitais);
2. Usar série de cossenos ou senos como função de aproximação (caso dos filtros digitais);
3. Usar como funções de aproximação polinômios, splines, para aproximar outros tipos de funções;
4. Interferir num modelo buscando atingir uma meta determinada, etc.

O algoritmo usado para projetar filtros digitais não recursivos com fase linear será descrito no próximo capítulo.

# Capítulo 5

# Implementação e análise do trabalho

Este capítulo tem o propósito de discutir as implementações do algoritmo; mostrar e comparar os resultados computacionais obtidos com o trabalho METEOR [43], buscando avaliar os desempenhos com relação aos objetivos propostos.

A idéia de usar o Método Simplex Revisado, para resolver o problema do projeto de filtros é atraente pela possibilidade de usar na implementação do método as características do problema. Embora o algoritmo do Simplex Revisado tenha sido descrito no capítulo anterior, é necessário revê-lo, mostrando as adaptações às particularidades do problema. Além das alterações, foram feitas várias versões do algoritmo, as quais interpretam o mesmo problema sob pontos de vista diferentes.

## 5.1 Considerações sobre a implementação

Inicialmente foram analisadas as características do problema procurando enquadrá-las de forma coerente com o algoritmo do Simplex Revisado. Isso foi feito com o intuito de construir um único programa que realizasse todas as tarefas ao mesmo tempo, ou seja, gerasse os dados do problema e o resolvesse.

O programa solicita ao usuário dois parâmetros responsáveis pela formação dos dados do problema. Essas informações de entrada dizem respeito ao comprimento do filtro $(M+1)$ e ao número de divisões que o intervalo deverá possuir $(N)$. Isto é, são as especificações do filtro que se deseja projetar. No final do processo, o programa devolve as variáveis duais que

atingiram o valor ótimo e a função objetivo que estima o erro absoluto máximo.

Outro aspecto a ser considerado é o fato do problema dual (eq. 4.16) não possuir uma base inicial factível, o que torna necessária a introdução de variáveis artificiais e o uso de algum método para iniciar o processo de otimização. O método escolhido para realizar essa tarefa foi o Big-M [28, 4], que se mostrou adequado à aplicação, além de ser facilmente implementado.

Esse problema é altamente degenerado, pois o vetor do lado direito possui todos os elementos iguais a zero (com excessão do último que possui valor igual a unidade (eq. 4.16). Para resolver a degenerescência do problema foi utilizado um processo simples que consiste em definir, inicialmente, todos os elementos da última coluna da matriz inversa da base com valor um, gerando um vetor solução inicial com componemtes iguais a um, por exemplo:

$$\begin{bmatrix} 1\,0\,0\,0\,1 \\ 0\,1\,0\,0\,1 \\ 0\,0\,1\,0\,1 \\ 0\,0\,0\,1\,1 \\ 0\,0\,0\,0\,1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}. \tag{5.1}$$

Como pode ser observado é apenas um artifício para evitar uma possível ciclagem do método [4], e não produz nenhuma alteração no valor da solução. Essa alternativa é diferente da usada por Steiglitz [42], Johnson [21] e por Kaiser, Parks e Steiglitz no trabalho METEOR [43].

Maiores detalhes da implementação são descritos a seguir.

### 5.1.1 Estruturas usadas no programa

A linguagem de programação usada é o Pascal 6.0, e a maioria das variáveis do algoritmo são usadas com precisão dupla, procurando um resultado mais preciso.

Conforme explicações anteriores, o programa gera os dados do problema a partir de dois parâmetros: o comprimento do filtro $(M + 1)$ e o número de divisões do intervalo $(N)$; esse parâmetros são a base dos cálculos dos componentes do problema.

Segundo a formulação do problema dual (eq. 4.16), pode-se observar que cada coluna é formada com a informação de um ponto específico do intervalo discretizado, e que a mesma coluna se repete com os sinais trocados. Através dessa observação, decidiu-se agrupar os dados referentes

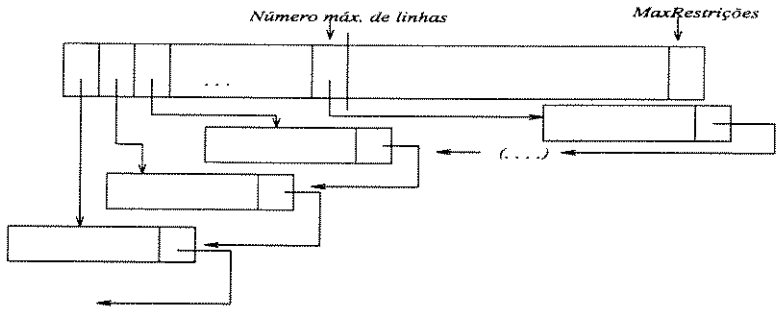


Figura 5.1: Estrutura usada para armazenar a base

a essas colunas (inclusive as artificiais). Para isso, foi usado um registro com os seguintes campos:

1. Informação se a variável é ou não básica;
2. Custo da função objetivo correspondente;
3. Ponto discreto do intervalo referente à coluna;
4. Se a coluna é positiva, negativa ou artificial;
5. Ponteiro que indica o próximo registro do problema.

Os elementos utilizados pelo Simplex Revisado na solução do problema, possuem estruturas distintas, descritas a seguir:

- A base ($\mathbf{B}$) é armazenada num vetor (veja Fig. 5.1) de dimensão igual ao número máximo de restrições do problema (MaxRestrições); cada elemento desse vetor é um ponteiro para um registro, como o descrito há pouco.
- A inversa da base ($\mathbf{BI}$) é armazenada num vetor (Fig. 5.2) de dimensão MaxRestrições, sendo que cada elemento desse vetor é um ponteiro para outro vetor de dimensão MaxRestrições que possui o conteúdo das linhas da matriz inversa da base.
- A solução básica ($\mathbf{x}_b$) é indicada por um ponteiro e armazenada num vetor de dimensão MaxRestrições, conforme a Figura 5.3.

Essa forma de armazenar as informações evita que sejam reservados grandes espaços para a matriz de restrições, e para a matriz inversa da base.

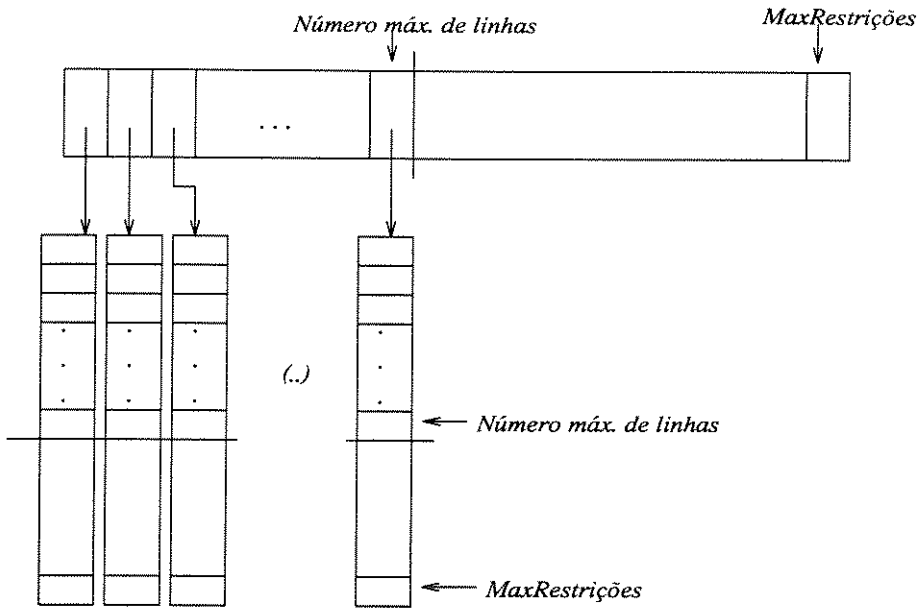


Figura 5.2: Estrutura usada para armazenar a inversa da base

Pode-se verificar que a matriz das restrições não é armazenada por inteiro, isto é, não são armazenadas suas linhas nem suas colunas, mas apenas as informações necessárias para gerá-las.

Em seguida, é descrita a maneira com que o algoritmo trabalha com as informações da matriz, armazenadas nos registros descritos.

### 5.1.2 Leitura dos dados do problema

Durante a realização do Simplex Revisado, o método utiliza as colunas da matriz de restrições em duas situações: na hora de selecionar a variável que entrará na base e de selecionar aquela que a deixará. Mesmo assim, não é preciso calcular explicitamente as colunas da matriz, pois nos dois casos elas equivalem a calcular um polinômio complexo.

Desta forma, ao invés de calcular as colunas e após isso efetuar o produto com outros vetores (realizar um produto interno), optou-se por calcular o polinômio complexo que esquivale à essas duas ações. Quando a coluna em questão se refere às variáveis artificiais, o procedimento é mais simples porque as primeiras $(M+1)$ colunas são da forma $\mathbf{e_i}$ o que dispensa cálculos, e a última coluna $(M+2)$ equivale a somar todos os elementos do vetor pelo qual ela seria

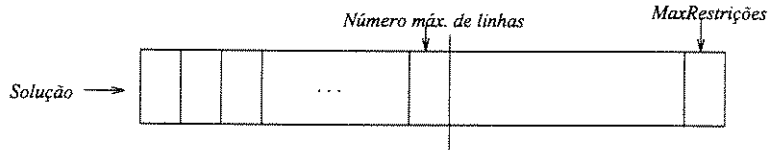


Figura 5.3: Estrutura usada para armazenar a solução

multiplicada (devido ao critério adotado para evitar a ciclagem no método).

Para facilitar o entendimento desse processo, os passos do algoritmo são detalhados em seguida:

- **Passo 1:** O algoritmo requisita informações, ao usuário, sobre o comprimento do filtro e o número de divisões do intervalo.
- **Passo 2:** A partir das informações do Passo 1, o algoritmo:
  1. Calcula os pontos de discretização do intervalo e aloca, em registros, as informações básicas sobre a coluna que cada ponto discreto representa.
  2. Constroe a inversa da base, preparando-a, para evitar a ciclagem do problema.
  3. Gera o vetor de solução, assim como a base inicial que aponta, no começo, para registros que contém as informações das variáveis artificiais do problema.
- **Passo 3:** É neste ponto que o método Simplex Revisado se inicia. Ele calcula $\mathbf{w} = \mathbf{c}_B\mathbf{B}^{-1}$ e para cada variável não básica calcula:

$$c_k - z_k = \min_{j \in I(N)} \{c_j - z_j\},$$

ou seja,

$$c_k - z_k = \min_{j \in I(N)} \{c_j - \mathbf{w}\mathbf{a}_j\}, \tag{5.2}$$

sendo que $\mathbf{w}\mathbf{a}_j$ pode ser calculado como um polinômio complexo, pois $\mathbf{a}_j$ representa uma coluna com a seguinte estrutura:

$$\mathbf{a}_j = \begin{cases} \cos(2\pi x_j i), & i = 0, 1, \ldots, M \\ 1, & i = M + 1 \end{cases}$$

$$j = 0, 1, \ldots, 2N + 1.$$

Observe que o produto $\mathbf{wa}_j$ pode ser calculado conforme o algoritmo:

```
Se x_j for uma var. artificial faça
   Se x_j não representa a coluna (M + 2) da inversa da base faça
      rr ← w(i);
   Senão faça
      rr ← w(M + 2);
      Para i = 1 a M + 1 faça
            rr ← rr − w(i);
      Fim (Para)
   Fim (Se)
Fim (Se)
Se x_j não representa uma var. artificial faça
   rr ← 0;
   ri ← 0;
   α ← 2 * π * x(j);
   xr ← cos(α);
   xi ← sin(α);
   Para i = (M + 1) downto 1 faça
         tr ← rr * xr − ri * xi;
         ri ← rr * xi + ri * xr;
         rr ← tr + w(i);
   Fim (Para)
   Se a coluna em questão for positiva faça
         rr ← w(M + 2) + rr;
   Fim (Se)
   Se a coluna em questão for negativa faça
         rr ← w(M + 2) − rr;
   Fim (Se)
Fim (Se).
```

- **Passo 4:** Se $c_k - z_k$, calculado no passo anterior for maior ou igual a zero pare; a solução é ótima. Senão calcule $\mathbf{y}_k = \mathbf{B}^{-1}\mathbf{a}_k$; neste ponto observe que o produto interno de cada linha de $\mathbf{B}^{-1}$ com a coluna $\mathbf{a}_k$, pode ser efetuado com o mesmo procedimento descrito no Passo 3, bastando substituir a variável $w(i)$ por $\mathbf{B}_i^{-1}(j)$.
- **Passo 5:** Se $\mathbf{y}_k$ for menor ou igual a zero pare; a solução é ilimitada. Senão faça o teste da razão:

$$\frac{\bar{b}_r}{y_{rk}} = \min\{\frac{\bar{b}_i}{y_{ik}} : y_{ik} > 0\}. \tag{5.3}$$

- **Passo 6:** Atualize os dados conforme foi descrito na página 49.

Esse algoritmo representa o método Simplex Revisado com a adaptação necessária para a solução do problema. Porém, observou-se a possibilidade de incrementar o algoritmo com o uso de algum tipo de busca unidimensional, cujos detalhes são dados a seguir.

## 5.2 Variações do Programa

Antes de entrar em maiores detalhes sobre o uso de busca unidimensional no programa é importante esclarecer que, *a priori*, há duas versões básicas do algoritmo: uma que considera todo o intervalo de trabalho e outra que não considera a faixa de transição (Figuras 3.8 (b) e(c), respectivamente). Foram usadas essas duas formas devido ao comportamento da matriz de restrições, descrito no capítulo 4. Segundo Kreyszig [24], não há distorção na solução de um sistema *real* se a matriz possuir número de condição até o valor 30, mesmo assim é interessante analisar os resultados produzidos pelos dois algoritmos.

Pelo fato dos dados serem provenientes de um intervalo discretizado, e analisando atentamente o algoritmo do Simplex Revisado, é facil verificar que no Passo 3, talvez o maior valor encontrado para a função custo, $c(x_{max})$, entre os pontos discretos não significa que ele é o maior entre $(x_{max} - \varepsilon, x_{max} + \varepsilon)$. Ou seja, pode existir um ponto muito próximo ao escolhido, e não pertencente ao conjunto de pontos discretos do intervalo, que produza um valor maior para a função custo reduzido, observe a Figura 5.4.

Portanto, neste passo, optou-se por uma busca unidimensional no intervalo gerado por $(x_{max} - \varepsilon, x_{max} + \varepsilon)$, sendo $\varepsilon$ um valor menor que a distância entre dois pontos (discretos) consecutivos do intervalo, para verificar se a função custo eleita anteriormente é realmente a que possui valor máximo.

Devido à existência das duas versões básicas do algoritmo, e da possibilidade de incluir busca unidimensional no programa, foram realizadas seis versões diferentes que resolvem o mesmo problema, e se baseiam nos seguintes aspectos:

- **ComIntervalo** - Considera todo o intervalo de trabalho, ou seja, inclui a região de transição;
- **SemIntervalo** - Exclui os pontos compreendidos na faixa de rejeição;

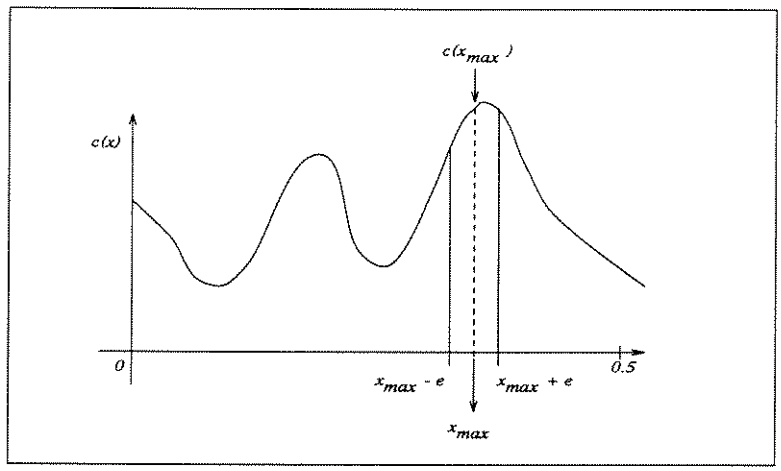


Figura 5.4: Gráfico da função custo

- **ComIntervaloÁurea** - Considera todo o intervalo de trabalho e aplica busca unidimensional através de Seção Áurea [26];
- **ComIntervaloBezier** - Considera todo o intervalo de trabalho e aplica busca unidimensional através de Interpolação de Bezier [37];
- **SemIntervaloÁurea** - Exclui os pontos compreendidos na faixa de rejeição e aplica busca unidimensional através de Seção Áurea [26];
- **SemIntervaloBezier** - Exclui os pontos compreendidos na faixa de rejeição e aplica busca unidimensional através de Interpolação de Bezier [37].

Essas versões do programa são abordadas na seqüência, assim como algumas considerações sobre os resultados que elas apresentaram.

### 5.2.1 Abordando todo o intervalo de trabalho

Através da Figura 3.8 foram observados os três casos freqüentemente considerados, estudados por diferentes métodos e critérios que visam projetar um filtro.

Embora a literatura mostre que o caso mais empregado é aquele cuja função da resposta em freqüência não é definida na região de transição, Kaiser, Parks e Steiglitz [43], Samueli [1] [38],

[1] O autor fez observações sobre o mal condicionamento desta matriz de restrições.

| *Desvio Máximo da Aproximação* | | | |
|---|---|---|---|
| $M$ | $N = 100$ | $N = 50$ | $N = 20$ |
| 75 | 6.169e-003 | 4.090e-003 | — |
| 71 | 6.253e-003 | 4.325e-003 | — |
| 67 | 6.964e-003 | 5.669e-003 | — |
| 65 | 6.965e-003 | 5.670e-003 | — |
| 61 | 7.470e-003 | 6.830e-003 | — |
| 57 | 7.765e-003 | 7.513e-003 | — |
| 51 | 8.461e-003 | 8.076e-003 | — |
| 45 | 1.046e-002 | 1.042e-002 | — |
| 39 | 1.325e-002 | 1.306e-002 | — |
| 31 | 1.494e-002 | 1.451e-002 | 9.420e-003 |
| 21 | 2.596e-002 | 2.564e-002 | 2.445e-002 |
| 11 | 5.076e-002 | 5.062e-002 | 4.690e-002 |
| 05 | 1.130e-001 | 1.130e-001 | 1.130e-001 |

Tabela 5.1: Soluções encontradas pelo programa ComIntervalo usando $eps = 0.05$

estuda-se também, filtros com definição da região de transição. A função da resposta em freqüência desses filtros proporcionam matrizes de restrições bem comportadas.

A principal diferença entre os algoritmos básicos, além das diferentes maneiras de considerar o intervalo, é o modo de calcular a função objetivo (que no dual corresponde à função da resposta em freqüência). A versão do programa ComIntervalo segue exatamente os passos do algoritmo descrito na seção 5.1.2, mas a função da resposta em freqüência é considerada com valor unitário na faixa $[0, fc - eps]$, assume os valores de uma reta entre $(fc - eps, fc + eps)$ e zero no restante do intervalo $[fc + eps, 0.5]$, sendo que a freqüência de corte corresponde a $fc = 0.25$ e $eps$ foi testado com dois valores; $eps = 10^{-8}$ e $eps = 0.05$ ($eps$ é o parâmetro que define o tamanho do intervalo da região de transição). Esse algoritmo fornece uma seletividade maior para o filtro a ser projetado para o valor de $eps = 10^{-8}$, pois a região de transição é bastante estreita.

Para assegurar o resultado alcançado com esse programa foi feita uma versão do primal, com as mesmas especificações do dual, no *Matlab*. A implementação usa a ferramenta *lp* do toolbox de otimização. Essa função resolve um problema primal de programação linear.

As tabelas 5.1 e 5.2 possuem as soluções encontradas por esse programa (ComIntervalo) para as especificações de $M$ (ordem do filtro) e $N$ (número de divisões do intervalo) com valores

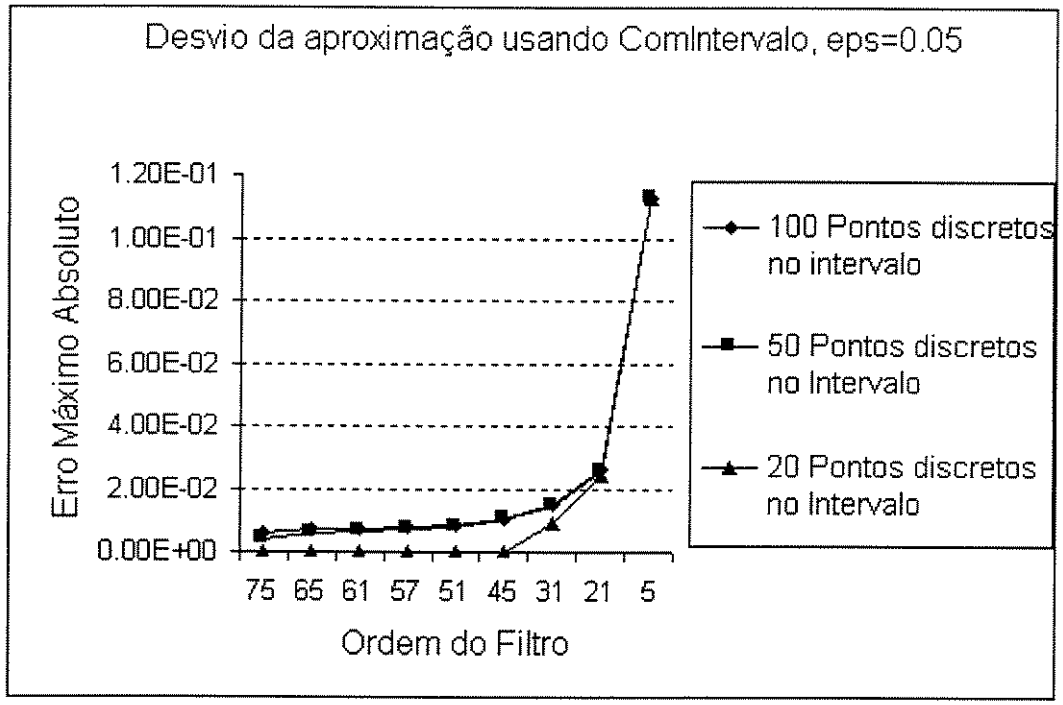


Figura 5.5: Gáfico das soluções encontradas pelo programa ComIntervalo usando $eps = 0.05$

de folga $eps = 0.05$ e $eps = 10^{-8}$, respectivamente. Os resultados das tabelas 5.1 e 5.2 sã ilustrados através dos gráficos 5.5 e 5.6.

Analisando os resultados, percebe-se que o refinamento nas especificações (como aumentar a ordem do filtro, $M$, e o número de pontos discretos do intervalo, $N$), não produz melhoras significativas para a função da resposta em freqüência. Os resultados obtidos pelo primal (versão *Matlab*) e dual (versão em Pascal) foram idênticos.

A seguir, são estudadas as outras versões do programa.

### 5.2.2 Excluindo a faixa de transição

Esta é a aplicação mais comum do problema de projeto de filtros. Como foi citado anteriormente, é o caso mais usado na literatura consultada.

O fato de desprezar os pontos que pertencem à faixa de transição (Figura 5.7) implica na

| *Desvio Máximo da Aproximação* | | | |
|---|---|---|---|
| $M$ | $N = 100$ | $N = 50$ | $N = 20$ |
| 75 | 9.169e-002 | 1.218e-002 | — |
| 65 | 1.048e-001 | 2.105e-002 | — |
| 61 | 1.109e-001 | 2.734e-002 | — |
| 57 | 1.251e-001 | 3.417e-002 | — |
| 51 | 1.528e-001 | 5.046e-002 | — |
| 45 | 1.673e-001 | 6.273e-002 | — |
| 31 | 2.371e-001 | 1.165e-001 | 1.038e-002 |
| 21 | 2.884e-001 | 1.731e-001 | 3.885e-002 |
| 05 | 4.257e-001 | 3.464e-001 | 2.308e-001 |

Tabela 5.2: Soluções encontradas pelo programa ComIntervalo usando $eps = 10^{-8}$

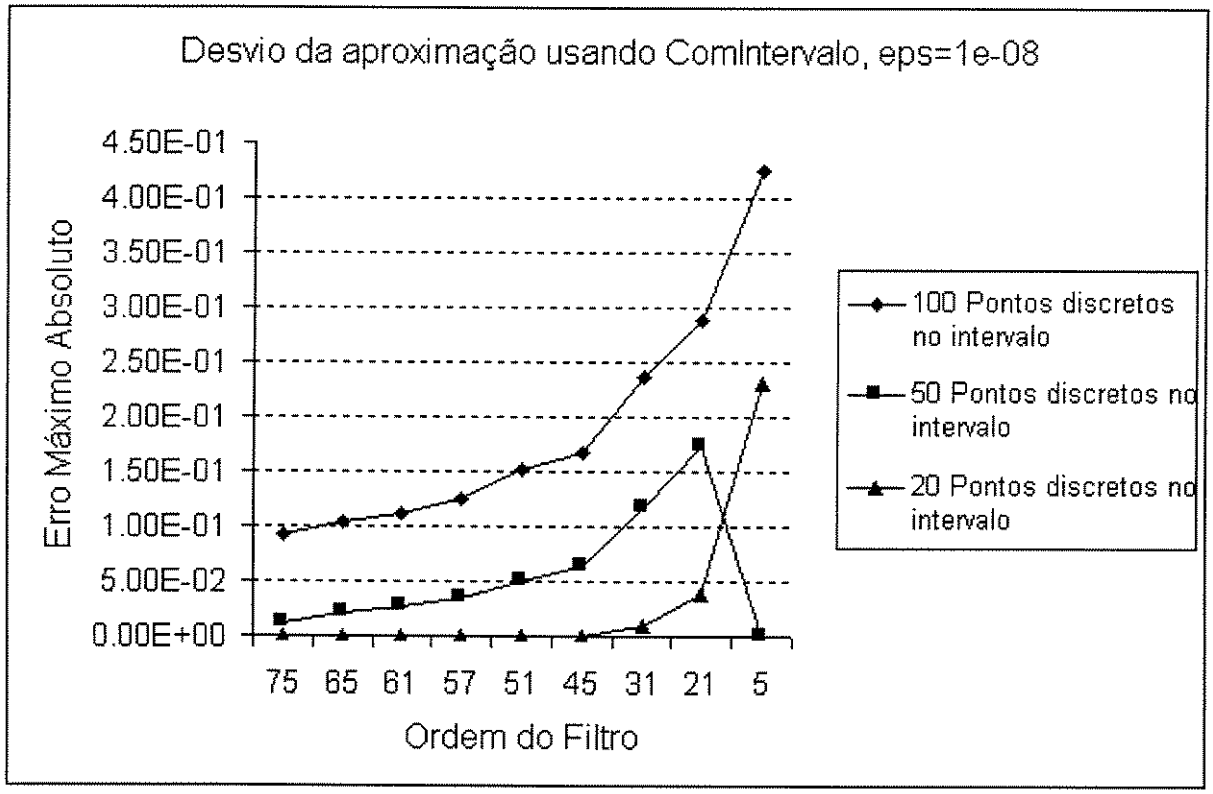


Figura 5.6: Gáfico das soluções encontradas pelo programa ComIntervalo usando $eps = 10^{-8}$

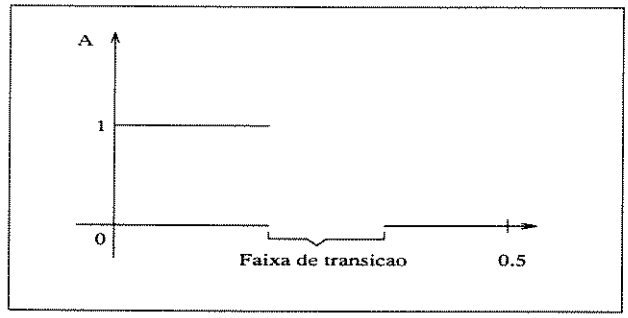


Figura 5.7: Filtro Digital Passa-baixa desconsiderando a região de transição

exclusão de linhas (quando é considerado o problema primal) ou colunas (quando é o dual) da matriz de restrições. Obeserva-se que esse *corte* torna a matriz mal condicionada — o número de condição desse tipo de matriz aumentou à medida que a ordem do filtro ($M$) cresceu.

O algoritmo que descreve esta interpretação do problema (programa SemIntervalo) é praticamente o mesmo do caso ComIntervalo. A principal diferença é na geração dos dados. Nesse caso é preciso eliminar os pontos discretos indesejáveis. Discretiza-se todo o intervalo em $(N+1)$ amostras; se para os pontos, pertencentes ao conjunto gerado pela divisão do intervalo entre $[0, fc - eps]$, a função da resposta em freqüência possui valor unitário; para os pontos discretos pertencentes ao intervalo $[fc + eps, 0.5]$ a função objetivo assume valor zero; para o restante do intervalo a função não é definida. Os valores usados, neste programa, para a freqüência de corte foi $fc = 0.25$; para a folga, $eps = 0.05$.

Para este caso também foi realizado um programa primal, no *Matlab*, para a averigüação dos resultados (que foram os mesmos), o qual gera os dados deste problema e resolve o sistema com a função *lp* definida na página 65. Além deste programa do *Matlab*, foi possível estabelecer comparações de resultados com o programa Meteor, descrito no artigo de Steiglitz, Parks e Kaiser [43] — os autores deixaram o programa em domínio público (obtido por *ftp*). O Meteor é detalhado na próxima seção (5.4), onde se encontram as comparações das soluções em conjunto com as respectivas análises.

Os resultados encontrados pelo programa SemIntervalo foram melhores que os alcançados pelo ComIntervalo, obviamente por este último analisar o erro inclusive na região de descontinuidade do problema, o que não ocorre no primeiro. Vale lembrar que ao se projetar um filtro, deseja-se que ele proporcione ganho próximo a um nas freqüências que ele deixa passar, e ganho zero nas freqüências que ele não deixa passar.

Observe, pelas soluções obtidas, que os desvios não sofrem alterações consideráveis ao aumentar

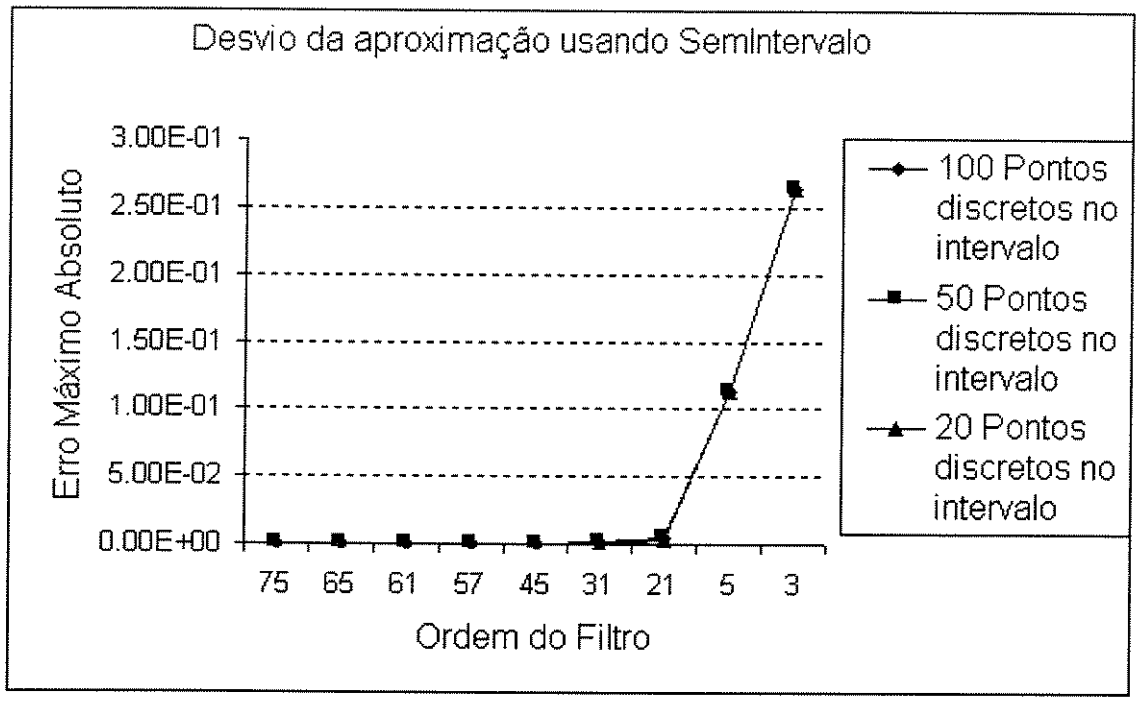


Figura 5.8: Gáfico das soluções encontradas pelo programa SemIntervalo

| *Desvio Máximo da Aproximação* | | | |
|---|---|---|---|
| $M$ | $N = 100$ | $N = 50$ | $N = 20$ |
| 75 | 4.245e-007 | 5.910e-009 | — |
| 71 | 9.880e-007 | 4.060e-008 | — |
| 67 | 2.071e-006 | 1.872e-007 | — |
| 65 | 2.071e-006 | 1.872e-007 | — |
| 61 | 4.354e-006 | 7.144e-007 | — |
| 57 | 9.469e-006 | 2.374e-006 | — |
| 51 | 3.885e-005 | 1.867e-005 | — |
| 45 | 8.050e-005 | 4.755e-005 | — |
| 39 | 3.319e-004 | 2.654e-004 | — |
| 31 | 1.287e-003 | 1.256e-003 | 1.785e-004 |
| 21 | 5.444e-003 | 5.439e-003 | 3.642e-003 |
| 11 | 5.076e-002 | 5.062e-002 | 4.690e-002 |
| 05 | 1.130e-001 | 1.130e-001 | 1.130e-001 |

Tabela 5.3: Soluções encontradas pelo programa SemIntervalo

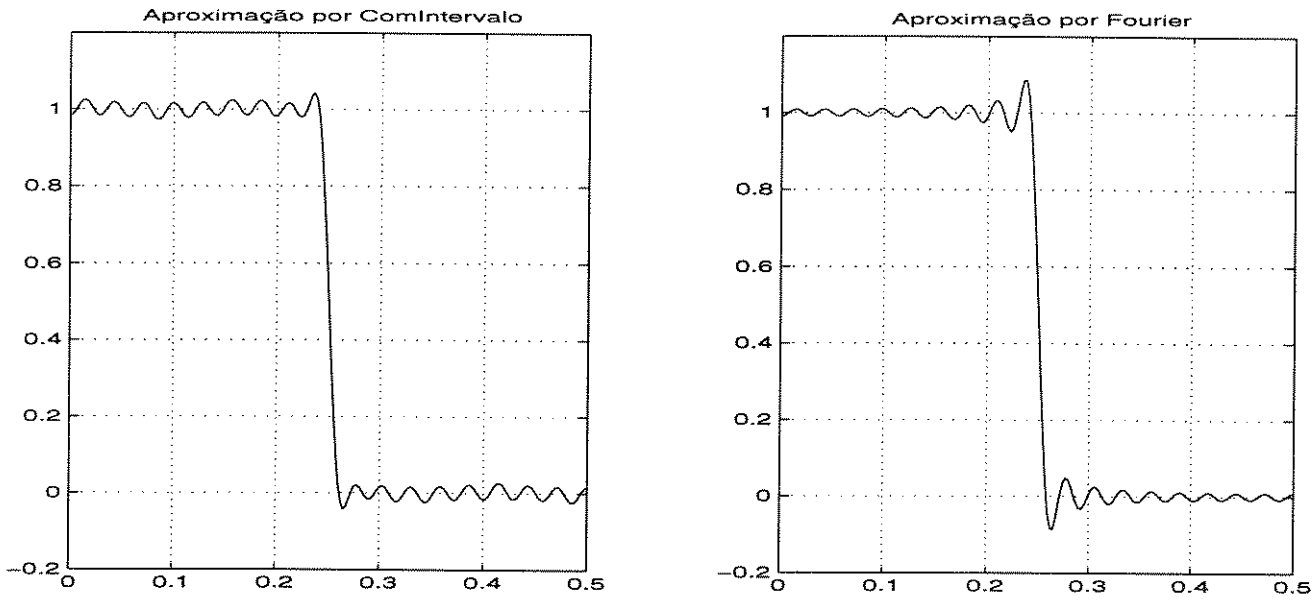


Figura 5.9: Filtros projetados por ComIntervalo e Fourier com ordem M=71 e N=50

o número de pontos discretos no intervalo, mas sim quando são permitidas ordens maiores para o filtro. Entretanto, conforme a tabela 5.3 e o gráfico 5.8, não é preciso aumentar muito o comprimento do filtro $(M + 1)$ para conseguir bons resultados. Portanto, nem sempre é necessário que o filtro possua alta ordem para desempenhar bem a sua tarefa.

Os tópicos descritos a seguir avaliam a introdução de uma busca unidimensional no programa.

### 5.2.3 Aplicação de Buscas Unidimensionais

A proposta do uso de busca unidimensional surgiu com a intenção de melhorar os resultados sem precisar contar com muitos pontos discretos. Usando essa idéia, uma amostra não pertencente ao conjunto de pontos discretos também poderia ser avaliada (por meio da busca unidimensional).

Conforme a descrição da página 63, esse método é incluído no Passo 3 do algoritmo simplex. Verifica-se os pontos (não discretos) vizinhos daquele que fornece a função custo relativo, com valor máximo para o conjunto discreto, procurando se alguma dessas amostras vizinhas produz um valor ainda maior, para a função custo reduzido.

Os métodos da Seção Áurea [26] e o método que usa Curvas de Bezier [37] foram implementados, individualmente, em cada um dos programas básicos: ComIntervalo e SemIntervalo. Foram escolhidos esses dois métodos porque Seção Áurea é simples de ser realizado e Bezier devido ao seu bom desempenho no trabalho desenvolvido por Said e Dobgenski [37].

A implementação de buscas unidimensionais, de fato, levou a melhores soluções. No entanto, as melhoras foram sempre modestas (na ordem de $10^{-5}$ e $10^{-6}$). Devido ao pequeno aumento da precisão da solução, não faz sentido expor os resultados.

## 5.3 Comparação com Fourier

Foi feito um programa, no *Matlab*, para projetar um filtro digital FIR passa baixa, com fase linear através de séries de Fourier. Esse programa é bastante simples, pois usa apenas a série de Fourier para aproximar o filtro desejado. A figura 5.9 mostra a diferença entre os filtros projetados por ComIntervalo e Fourier. Observa-se que o filtro projetado com ComIntervalo possui o desvio máximo na aproximação um pouco menor do que o apresentado pela série de Fourier. Entretanto, para filtros de baixa ordem a aproximação por Fourier é de melhor qualidade. Foram analisadas, em conjunto, apenas as aproximações por ComIntervalo e Fourier por causa da forma como o intervalo é considerado, por completo.

## 5.4 Resultados Obtidos

Conforme as descrições anteriores, os programas ComIntervalo e SemIntervalo são eficientes na tarefa de aproximar a função da resposta em freqüência de um filtro digital. Embora alguns resultados conseguidos com esses programas já tenham sido mostrados, é importante observá-los em conjunto com resultados de outros autores. Como foi possível ter acesso aos programas (Form e Meteor) usados no trabalho desenvolvido por Steiglitz, Parks e Kaiser [43], os resultados provenientes desses programas e de SemIntervalo são comparados.

Na revisão de literatura (capítulo 3, página 41), foi citado o trabalho referente à esses programas, e as especificações que ele aceita para projetar um filtro. O programa SemIntervalo, desenvolvido neste trabalho, se enquadra numa das três opções oferecidas por Meteor [43].

O programa Form pede informações ao usuário sobre as especificações do filtro, como, por exemplo, a extensão da faixa passante e de rejeição, número de divisões que terá o intervalo

| *Desvio Máximo da Aproximação* | | | | | | |
|---|---|---|---|---|---|---|
| $M$ | $N = 100$ | | $N = 50$ | | $N = 20$ | |
| | SemIntervalo | Meteor | SemIntervalo | Meteor | SemIntervalo | Meteor |
| 75 | 4.245e-007 | 8.408e-007 | 5.910e-009 | 7.968e-005 | — | — |
| 67 | 2.071e-006 | 2.071e-006 | 1.872e-007 | 1.872e-007 | — | — |
| 65 | 2.071e-006 | 4.355e-006 | 1.872e-007 | 7.167e-007 | — | — |
| 61 | 4.354e-006 | 9.469e-006 | 7.144e-007 | 2.374e-006 | — | — |
| 57 | 9.469e-006 | 1.943e-005 | 2.374e-006 | 6.950e-006 | — | — |
| 51 | 3.885e-005 | 3.885e-005 | 1.867e-005 | 1.867e-005 | — | — |
| 45 | 8.050e-005 | 1.684e-004 | 4.755e-005 | 1.159e-004 | — | — |
| 39 | 3.319e-004 | 3.319e-004 | 2.654e-004 | 2.654e-004 | — | — |
| 31 | 1.287e-003 | 1.287e-003 | 1.256e-003 | 1.256e-003 | 1.785e-004 | 1.785e-004 |
| 21 | 5.444e-003 | 1.135e-002 | 5.439e-003 | 1.100e-002 | 3.642e-003 | 9.950e-003 |
| 11 | 5.076e-002 | 5.076e-002 | 5.062e-002 | 5.062e-002 | 4.690e-002 | 4.690e-002 |
| 05 | 1.130e-001 | infactível | 1.130e-001 | infactível | 1.130e-001 | infactível |
| 03 | 2.639e-001 | 2.639e-001 | 2.639e-001 | 2.639e-001 | 2.639e-001 | 2.639e-001 |

Tabela 5.4: Soluções encontradas por SemIntervalo e Meteor

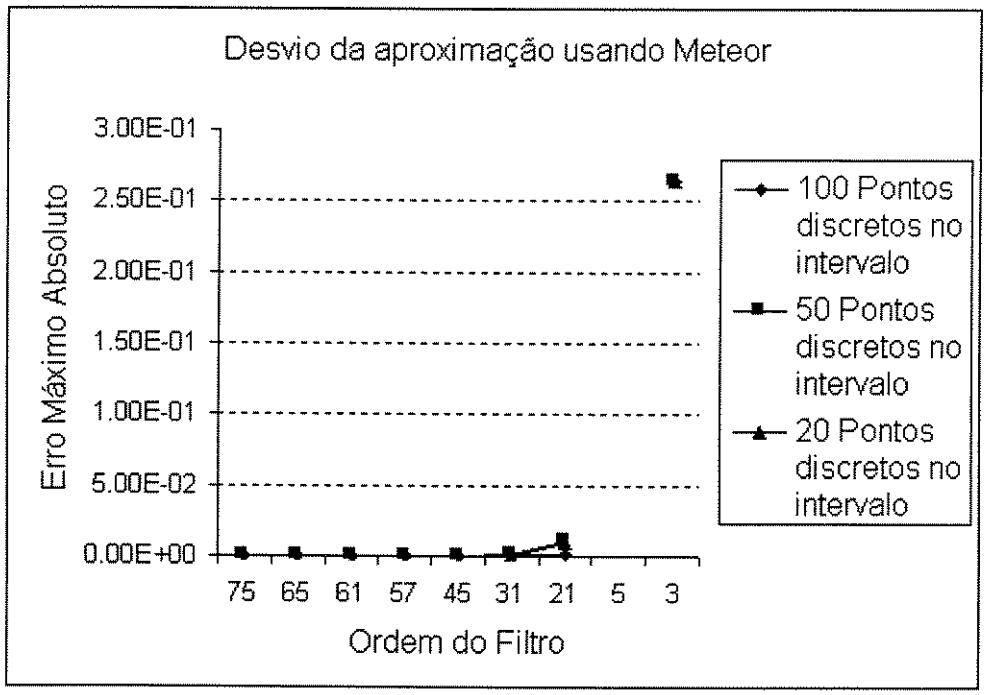


Figura 5.10: Gáfico das soluções encontradas pelo programa Meteor

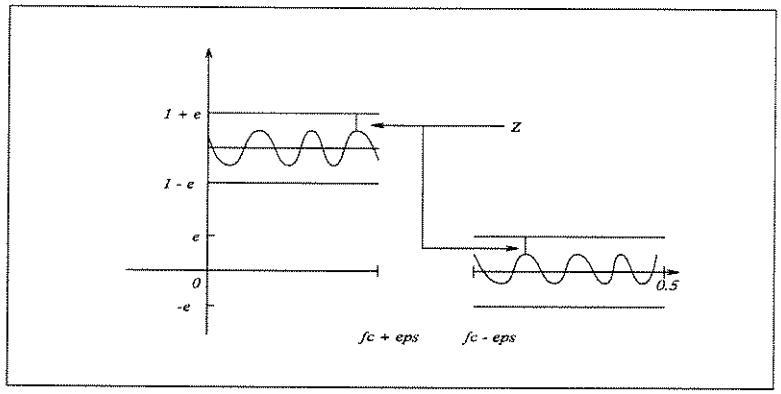


Figura 5.11: Filtro Digital Passa-baixa desconsiderando a região de transição

(o programa gera um arquivo de saída com essas informações).

O programa Meteor requisita os dados de entrada, fornecidos por Form, e resolve a aproximação através do método simplex revisado, usando a fase I do método para gerar uma solução factível, e a fase II para encontrar a solução ótima. Utilizando os dados de entrada, ele constroe a matriz de restrições, o vetor de soluções, a função objetivo e os armazena em matriz e vetores, respectivamente. Para realizar os passos do Simplex é construído o *tableau* de pivoteamento, característico do método, que também é armazenado numa matriz. Os autores afirmam que deram prioridade ao tempo de execução, em relação ao espaço de armazenamento das informações. No programa SemIntervalo, adotou-se a alternativa contrária. Assim, as soluções com SemIntervalo demoram um pouco mais para serem alcançadas, se comparado ao Meteor. Como a diferença de tempo é pequena (menos de 1 minuto), ela não faz parte das observações, as quais são centradas nos resultados sobre o tamanho do desvio máximo ocorrido para aproximar a função da resposta em freqüência. O modo como é feita a divisão do intervalo e a seleção dos pontos é o mesmo nos programas SemIntervalo e Meteor. Naturalmente, a divisão do intervalo e a seleção de pontos é a mesma, para evitar distorção na comparação dos resultados — além disso, esses dois fatores determinam o comportamento da matriz.

A tabela 5.4 mostra os resultados alcançados com os programas SemIntervalo e Meteor, para aproximar a função da resposta em freqüência de um filtro digital, seguindo as mesmas especificações. Para esses resultados foi considerado um filtro não recursivo passa-baixa, com fase linear; o comprimento do filtro e o número de divisões do intervalo são definidos individualmente (para cada filtro).

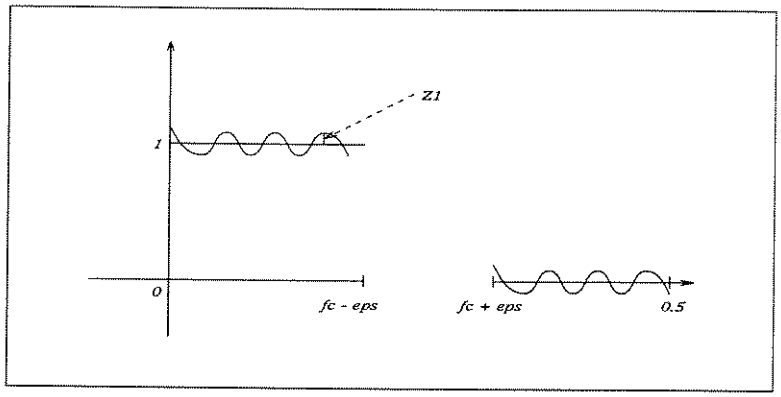


Figura 5.12: Filtro Digital Passa-baixa desconsiderando a região de transição

### 5.4.1 Análise dos Resultados

Considerando os resultados apresentados na tabela 5.4, verifica-se que o programa SemIntervalo foi um pouco mais preciso na aproximação do que o Meteor, para os filtros de alta ordem. Quando a ordem do filtro é pequena as soluções foram praticamente as mesmas, com exceção do caso em que $M = 5$, pois o Meteor não conseguiu resolver este problema.

As pequenas diferenças entre as soluções obtidas com os dois programas ocorreram, provavelmente pela formulação e armazenamento dos dados do problema, já que o método usado, por ambos para resolver o problema foi o mesmo. A Figura 5.11 ilustra a objetivo de Meteor, que maximiza a distância da função de aproximação da resposta em freqüência (denominada $Z$ na Figura 5.11), até o seu limite superior/inferior — essas informações foram descritas na revisão de literatura (capítulo 3).

O programa SemIntervalo minimiza o erro máximo produzido pela função de aproximação (denominado $Z1$ na Figura 5.12), tornando desnecessário o uso das restrições de limite superior e inferior usadas pelo Meteor. Essa idéia é representada na Figura 5.12.

A formulação adotada em SemIntervalo é a mais encontrada na literatura consultada, embora essa forma gere um problema dual degenerado. No entanto como discutiu-se anteriormente, os problemas de ciclagem podem ser contornados. A formulação usada por Meteor evita a ciclagem.

A seguir, são mostrados alguns gráficos que ilustram as funções encontradas (por SemIntervalo e Meteor) para a aproximação da resposta em freqüência. Apresenta-se também, gráficos com o erro absoluto, dos resultados produzidos por esses programas.

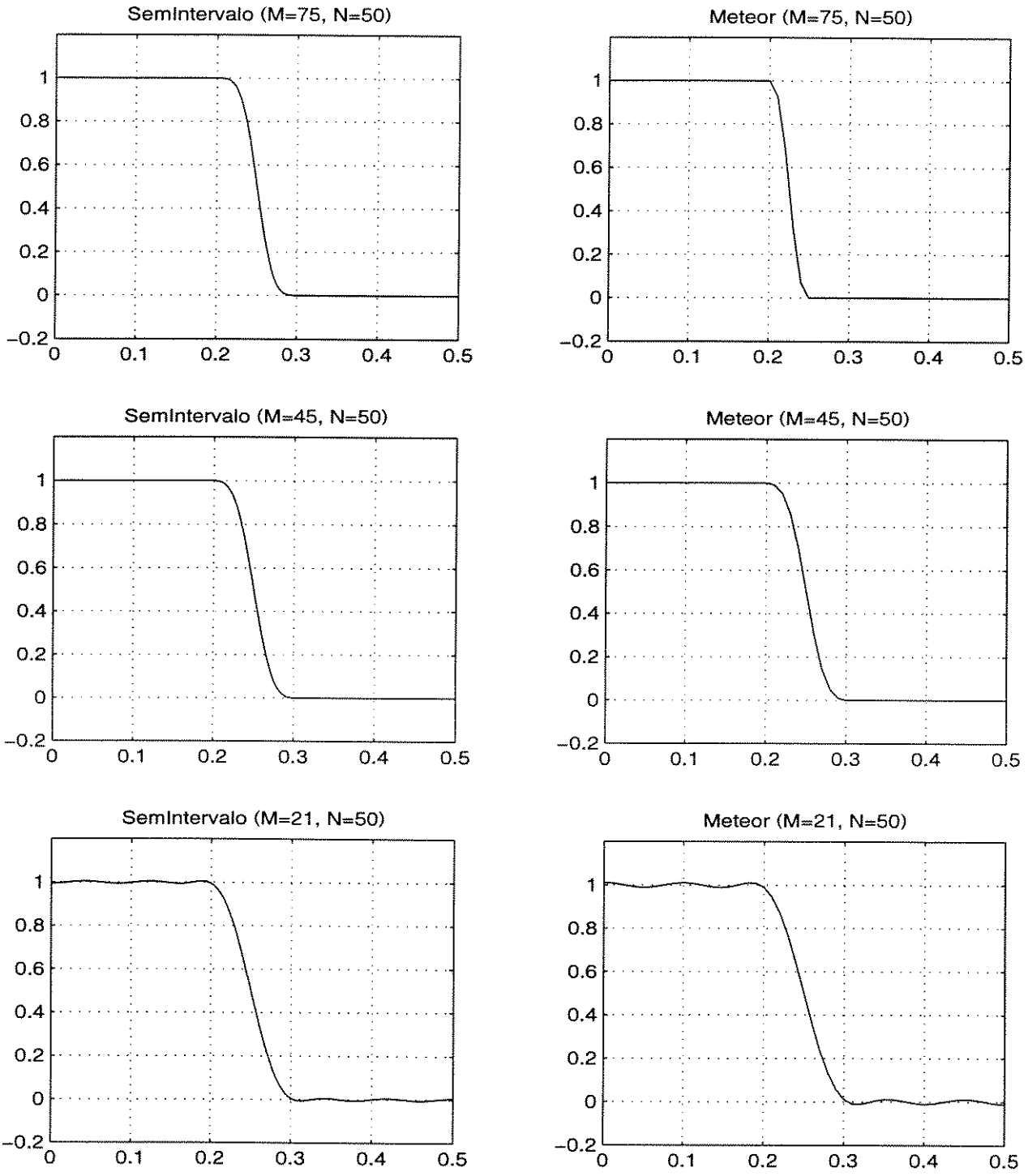


Figura 5.13: Filtros Digitais Passa Baixa projetados por SemIntervalo e Meteor

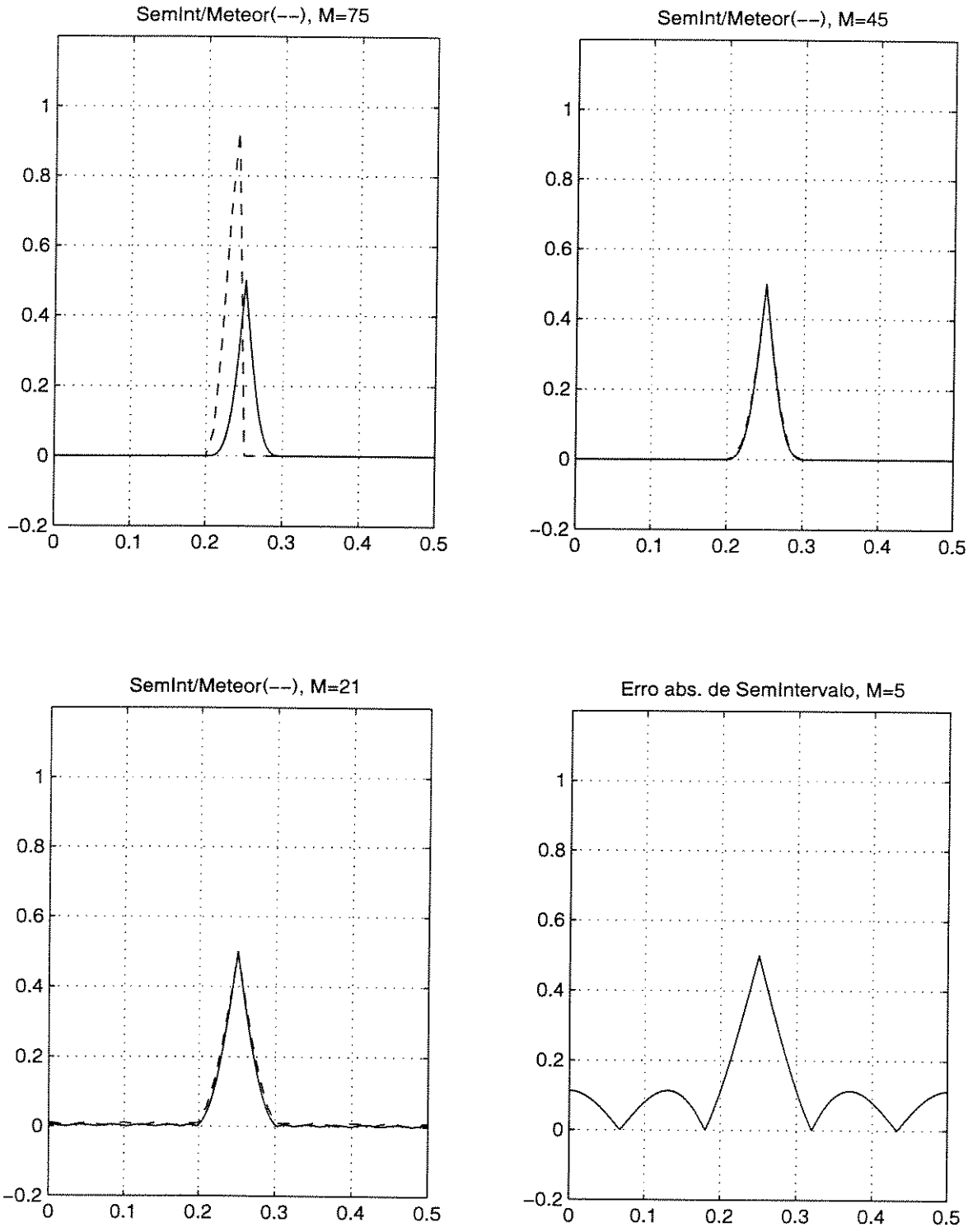


Figura 5.14: Erro absoluto entre os filtros projetados por SemIntervalo e Meteor

# Capítulo 6

# Conclusões

Uma das principais propostas deste estudo é mostrar a possibilidade de usar programação linear em problemas de aproximação de funções. A viabilidade desta aplicação foi mostrada através da solução do problema de projetar filtros digitais FIR, passa baixa, com fase linear. As funções da resposta em freqüência, obtida usando essa técnica, foi bem próxima da função ideal.

O capítulo 5 mostra a diferença entre o uso do método Simplex Revisado e aproximação através de séries de Fourier, para projetar um filtro digital de mesmas especificações. Através dos gráficos que representam as aproximações das funções da resposta em freqüência (Figura 5.9), é fácil observar que a aproximação feita por séries de Fourier possui um erro acentuado próximo à região de transição — essa é uma característica das aproximações por séries de Fourier. Como o erro dos extremos é acentuado, a aproximação feita por séries de Fourier é menos adequada do que a aproximação obtida por SemIntervalo (que usa o método Simplex Revisado). De fato, as aproximações obtidas com o programa SemIntervalo foram muito próximas das ideais.

Os teste realizados com o programa Meteor e SemIntervalo tiveram, praticamente, os mesmos resultados. A principal diferença entre eles é dada pela forma de implementação do método Simplex Revisado, pois Meteor não se preocupa em usar pouco espaço para armazenar os dados; SemIntervalo é o oposto, neste sentido. Embora os algoritmos tenham enfoques diferentes quanto à forma de implementação, os resultados encontrados por eles são muito bons, reforçando que a idéia de que os métodos de Programação Linear são uma boa alternativa para a aproximação de funções. Nesse sentido, também pode ser estudado o uso de métodos de pontos interiores [1, 22].

O trabalho realizado por Johnson [21], em 1990, foi recentemente analisado e incorporado ao estudo desta tese. Portanto, não houve tempo disponível para realizar testes com o algoritmo que ele desenvolveu.

Segundo as observações que Johnson fez em seu trabalho, ele fez testes comparativos com os problemas que Steiglitz [42] usou num estudo em 1979, e obteve melhora significativa no número de iterações necessárias para resolver o problema (e conseqüentemente no tempo). Mas, não obteve melhora no resultado do erro entre a função da resposta em freqüência e a desejada.

Seguindo essa mesma linha foi realizado, nesta pesquisa, um teste com as especificações de filtro que Steiglitz usou em [42]. Foi constatado que o erro obtido por SemIntervalo foi um pouco menor que o encontrado pelo algoritmo usado por Steiglitz. Portanto, é provável que o trabalho desenvolvido por Johnson tenha melhores resultados com relação ao tempo de execução do programa, se comparado ao SemIntervalo, e soluções bem próximas.

# Referências Bibliográficas


[1] I. Adler, N. K. Karmarkar, M. G. C. Resende and G. Veiga, "An Implementation od Karmarkars Algorithm for Linear Programming", *Maths Programming*, vol.44, 297-335, 1989.

[2] A. Antoniou, *Digital Filter: Analysis and Design*, McGraw-Hill, New York, 1979.

[3] N. S. Bakhvalov, *Numerical Methods*, Mir Publishers, Moscou, 1977.

[4] M. S. Bazaraa, J.J. Jarvis and H. D. Sherali, *Linear Programming and Network Flows*, John Wiley & Sons, New York, 1990.

[5] Ia. S. Bugrov and S. M. Nikolski, *Matemática para Engenharia, Tópicos de Análise*, vol. 3, Editora Mir Moscovo, URSS, 1987.

[6] D. Burnside and T. W. Parks, "Optimal Design of FIR Filters with the Complex Chebyshev Error Criteria", *IEEE Transactions on Signal Processing*, vol.43, n.3, 605-616, 1995.

[7] C. Cunha, *Métodos Numéricos para as Engenharias e Ciências Aplicadas*, Editora da Unicamp, Campinas, 1993.

[8] G. Dahlquist and A. Björck, *Numerical Methods*, Prentice-Hall, New Jersey, 1974.

[9] G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, New Jersey, 1963.

[10] P. J. Davis, *Interpolation & Approximation*, Dover Publications Inc., New York, 1975.

[11] R. A. DeVore and G. G. Lorentz, *Constructive Approximation*, Springer-Verlag, New York, 1993.

[12] S. C. Fang and S. Puthenpura, *Linear Optimization and Extensions: Theory and Algorithms*, Prentice Hall, New Jersey, 1993.

[13] C. E. Fröberg, *Numerical Mathematics*, Cummings Publishing Company Inc., California, 1985.

# Apêndice A

# Sistemas de Haar

Define-se: $\Phi = \{\phi_1, \ldots, \phi_n\}$ um conjunto de funções contínuas reais ou complexas sobre um espaço topológico de Hausdorff $A$, sendo que o espaço topológico compacto de Hausdorff é um subconjunto compacto (fechado e limitado) de $\Re^n$, é um sistema de Haar se as seguintes condições são satisfeitas:

1. $A$ contém no mínimo $n$ pontos;
2. Cada $P = a_1\phi_1 + \ldots + a_n\phi_n$, que não possui todos os coeficientes iguais a zero, tem no máximo $n-1$ zeros distintos sobre $A$.

O espaço $X_n$ gerado por $\phi_1, \ldots, \phi_n$ é um espaço de Haar. Os elementos de $X_n$ são chamados polinômios (ou $\Phi$-polinômios). Outra base de $X_n$, $\Psi : \psi_1, \ldots, \psi_n$, também é um sistema de Haar. Em particular segue desta definição que as funções de um sistema de Haar são linearmente independentes.

**Theorem 5** *Para um sistema de Haar, existe um único polinômio de melhor aproximação para cada função contínua.*

**Theorem 6 (A. Haar (1918) )** *Seja $\phi_1, \ldots, \phi_n$ funções contínuas linearmente independentes, definidas sobre um espaço compacto de Hausdorff $A$, que contém no mínimo n pontos. Se cada função contínua tem somente um polinômio de melhor aproximação, então $\phi_1, \ldots, \phi_n$ é um sistema de Haar.*

[14] P. E. Gill, W. Murray and M. H. Wright, *Pratical Optimization*, Academic Press, San Diego, 1989.

[15] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, London, 1990.

[16] R. W. Hamming, *Digital Filters*, Prentice-Hall, California, 1983.

[17] H. D. Helms,"Digital Filters with Equiripple or Minimax Response", *IEEE transactions on Audio and Electroacoustics*, vol AU-19, n. 1, 87-93, 1971.

[18] A. G. Holzman, *Operations Research Support Methodology*, Marcel Decker, New York, 1979.

[19] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, London, 1985.

[20] L. B. Jackson, *Digital Filters and Signal Processing*, Kluwer Academic Publishers, Boston, 1989.

[21] A.T. Johnson Jr., "Optimal Linear Phase Digital Filter Design by One-Phase Linear Programming", *IEEE Transactions on Circuits and Systems*, vol.37, n.3, 554-558, 1990.

[22] N. K. Karmarkar, "A New Polynomial-Time Algorithm for Linear Programming", *Combinatorica*, vol.4, 373-395, 1984.

[23] J.E. Kelley, "An application of linear programming to curve fitting, *J. Soc. Indust. Appl. Math.*, vol.6, 15-22, 1958.

[24] E. Kreyszig, *Advanced Engineering Mathematics*, John Wiley & Sons, New York, 1993.

[25] P. J. Laurent, *Approximation et Optimisation*, Hermann, Paris, 1972.

[26] D. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley, New York, 1984.

[27] G. Meinardus, *Approximation of Functions Theory and Numerical Methods*, Springer-Verlag, New York, 1967.

[28] K. G. Murty, *Linear Programming*, John Wiley & Sons, New York, 1983.

[29] R. G. Parker and R. L. Rardin, *Discrete Optimization*, Academic Press, San Diego, 1988.

[30] T. W. Parks and C. S. Burrus, *Digital Filter Design*, John Wiley & Sons, New York, 1987.

[31] L. R. Rabiner,"Linear Program Design of Finite Impulse Response (FIR) Digital Filters", *IEEE transactions on Audio and Electroacoustics*, vol AU-20, n..4, 280-288, 1972.

[32] L. R. Rabiner,"The Design of Finite Impulse Response Digital Filters using Linear Programming Techniques", *The Bell System Techinical Journal*, 1177-1198, 1972.

[33] L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*, Prentice-Hall, New Jersey, 1975.

[34] L. R. Rabiner, J. H. McClellan and T. W. Parks,"FIR Digital Filter Design Techniques using Weighted Chebyshev Approximation", *Proceeding of the IEEE*, vol 63, n.4, 595-610, 1975.

[35] A. P. Ricieri, *Construindo a Série de Fourier*, Edições Prandiano, São Paulo, 1988.

[36] R. A. Roberts and C. T. Mullis, *Digital Signal Processing*, Addison-Wesley, California, 1987.

[37] A. Said and J. Dobgenski, "Curvas de Bezier para Otimização", *Anais dos Resumos Estendidos VII CLAIO e XXVIII SBPO*, vol 1, 46-51, 1996.

[38] H. Samueli, "On the Design of Optimal Equiripple FIR Digital Filters for Data Transmission Applications", *IEEE Transactions on Circuits and Systems*, vol.35,n.12, 1542-1546, 1988.

[39] R. W. Schafer and L. R. Rabiner, "A Digital Signal Processing Approach to Interpolation", *Proceeding of the IEEE*, vol 61, n.6, 692-702, 1973.

[40] M. H. Schultz, *Spline Analysis*, Prentice-Hall, New Jersey, 1973.

[41] C. P. Serra, *Teoria e Projeto de Filtros*, Cartgraf, Campinas, 1983.

[42] K. Steiglitz, "Optimal Design of FIR Digital Filters with Monotone Passband Response", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. ASSP 27, n.6, 643-649, 1979.

[43] K. Steiglitz, T. W. Parks and J. F. Kaiser, "METEOR: A Constraint-Based FIR Filter Design Program", *IEEE Transactions on Signal Processing*, vol. 40, n.8, 1901-1909, 1992.

[44] F. J. Taylor, *Digital Filter Desing Handbook*, Marcel Dekker, New York, 1983.

[45] T. J. Terrell, *Introduction to Digital Filters*, John Wiley & Sons, New York, 1988.

[46] D. W. Tufts, D. W. Rorabacher and W. E. Moiser, "Designing Simple, Effective Digital Filters", *IEEE transactions on Audio and Electroacoustics*, vol AU-18, n.2, 142-158, 1970.

[47] S. Zionts, *Linear and integer programming*, Prentice Hall, New Jersey, 1974.

[48] L. E. Ward Jr, "Linear Programming and Approximations Problems", *Classroon Notes*, 46-53, 1961.