UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

ESCOLA DE ENGENHARIA

DEPARTAMENTO DE ELETRÔNICA

ANÁLISE DA CODIFICAÇÃO LPC PARA SINAIS DE FALA

Autor: _____
André Bernardo Kutwak

Co-Orientador: _____
Prof. Sérgio Lima Netto

Co-Orientador: _____
Prof. Fernando Gil Vianna Resende Junior

Examinador: _____
Prof. Paulo Sergio Ramirez Diniz

DEL
Abril de 1999

Agradecimentos

Gostaria de agradecer a orientação recebida dos professores Sergio Lima Netto e Fernando Gil Vianna Resende Junior no decorrer da elaboração deste trabalho, bem como a colaboração dos alunos do grupo de voz da Universidade Federal do Rio de Janeiro.

Gostaria de agradecer também toda a paciência e apoio que tive da família e amigos para poder concluir este trabalho, que representa a conclusão de mais uma etapa em minha vida.

André Bernardo Kutwak

Resumo

Este trabalho trata sobre uma das formas mais básicas de codificação paramétrica de sinais de fala, que é o LPC (“linear prediction coding”). Neste tipo de codificação, segmentamos o sinal de fala original e extraímos parâmetros que representarão cada um dos segmentos. Esses parâmetros são os coeficientes LPC, o ganho, e o “pitch” no caso do sinal ser do tipo vozeado. Com isso consegue-se baixas taxas de transmissão apesar da baixa qualidade do sinal sintetizado.

Após uma rápida introdução, apresentaremos um capítulo sobre codificação por formato de onda, onde introduziremos conceitos como amostragem e quantização de um sinal analógico, bem como os principais padrões de codificadores por forma de onda.

No capítulo 3, introduziremos os aspectos teóricos da codificação LPC, bem como suas soluções. O sistema LPC implementado neste trabalho é detalhado neste capítulo com todos os seus blocos funcionais.

A principal contribuição deste trabalho encontra-se no capítulo 4, onde foram feitas diversas análises relativas à implementação do sistema LPC que nos permitiram chegar a algumas conclusões. Para isso, foram feitas diversas simulações no sistema implementado sob diferentes condições de funcionamento.

O capítulo 5 introduz um novo domínio denominado cepstral, onde é possível separarmos os componentes do sinal relativos à excitação e ao trato vocal. É mostrado como transformarmos os coeficientes LP em coeficientes cepstrais bem como algumas comparações entre as respostas obtidas a partir dessas duas formas de codificação.

Ao final, apresentamos as conclusões e contribuições deste trabalho, bem como o seu direcionamento futuro.

Palavras-chave

- Sinal de fala
- Codificação paramétrica
- Predição linear
- Estacionaridade
- “Pitch”

1. INTRODUÇÃO

Atualmente, é muito comum trabalharmos com sinais digitalizados. No caso de sinais de fala, podemos citar exemplos como os sistemas de telefonia digitais e o armazenamento de sinais de fala em CD (“compact disk”), HD (“hard disk”), ou outros tipos de mídias existentes. Esses sinais que são originalmente analógicos devem ser amostrados e quantizados em seu processo de digitalização.

Após a quantização devemos representar estes sinais através de bits. Para isso foram desenvolvidos diversos tipos de codificação de sinais. A codificação pode ser por formato de onda ou paramétrica. A codificação por formato de onda possui excelente qualidade na reprodução do sinal original, porém necessita de uma alta taxa de transmissão, que varia geralmente entre 16 e 64 kbps. Já na codificação paramétrica utiliza-se taxas de transmissão da ordem de 2,4 a 4,8 kbps, porém em alguns casos essa compressão diminui a qualidade do sinal..

O tipo mais simples de codificação digital por formato de onda é o PCM (“pulse code modulation”). Neste tipo de codificação representa-se o sinal original pelo valor de cada amostra em binário. A taxa de transmissão típica utilizada em telefonia digital utilizando-se a codificação PCM é de 64 kbps. Existem outros tipos de codificação por formato de onda que utilizam taxas mais baixas de transmissão. Podemos citar como exemplo o DPCM (“differencial pulse code modulation”) e o ADPCM (“adaptative differencial pulse code modulation”). No capítulo 1 poderemos ver com maiores detalhes os principais tipos de codificações por formato de onda.

Já na codificação paramétrica, extraímos parâmetros do sinal original que vão nos permitir reconstruir este sinal. Assim, ao invés de transmitirmos ou armazenarmos o sinal original, utilizamos apenas os parâmetros que o representam. Para isso, geralmente segmentamos o sinal original de modo a trabalharmos com pequenos segmentos do sinal que podem ser considerados estacionários.

A codificação LPC (“linear prediction coding”), tema deste trabalho, é o tipo mais simples de codificação paramétrica existente, e base para outros tipos de codificações paramétricas como por exemplo o RELP (“residual excited linear prediction”) e o CELP (“code excited linear prediction”). Na codificação LPC, segmenta-se o sinal original e faz-se a análise LPC sobre cada segmento, como será visto adiante. Os parâmetros calculados são os coeficientes LPC, o ganho, o tipo de sinal de voz (se é do tipo vozeado ou não vozeado) e o valor da freqüência fundamental no caso do segmento ser do tipo vozeado. Além de obtermos uma menor taxa de transmissão do que nas codificações por formato de onda, os coeficientes LPC também podem ser úteis em sistemas de reconhecimento de voz. Neste caso, os coeficientes calculados podem ser comparados aos coeficientes de sinais de referência para que seja feita uma associação entre o sinal disponível e algum dos sinais de referência. Através da codificação paramétrica, podemos também alterar características do sinal, como por exemplo a freqüência fundamental do sinal ou a entonação de uma frase através da manipulação dos parâmetros obtidos.

A grande desvantagem da codificação LPC é que o sinal gerado se apresenta robotizado com uma sensível perda de qualidade. Essa qualidade pode ser aumentada utilizando-se outros tipos de codificação paramétrica como o CELP. Neste caso, além de transmitirmos os coeficientes calculados, também transmitimos o índice de um "codebook" que contém vários tipos de sinais de excitação diferentes. Assim pode-se utilizar um sinal de excitação mais adequado ao segmento de fala que está sendo sintetizado melhorando a qualidade da síntese.

Este projeto trata especificamente do padrão LPC de codificação, mostrando como ele pode ser implementado e os diversos aspectos práticos e teóricos envolvidos em sua implementação e utilização.

O capítulo 1 traz um resumo das principais formas de codificação por forma de onda existentes, para que possamos fazer uma comparação entre as duas formas de codificação. Já o capítulo 2 descreve detalhadamente a codificação LPC e descreve o sistema LPC implementado através do software MATLAB, com todos os blocos que compõem o sistema, detalhando a função de cada um.

No capítulo 3, tratamos dos aspectos mais importantes envolvendo a implementação de um sistema LPC. Estes aspectos são vistos sob o ponto de vista teórico e prático. O capítulo contém o estudo dos seguintes itens: tamanho de segmento a ser utilizado, tipo de pulso glotal, ordem do sistema, tipo de janelamento, utilização da superposição e utilização de distintos métodos para a solução LPC (autocorrelação e covariância). Este capítulo é a maior contribuição deste trabalho, já que traz análises de um sistema implementado pelos alunos do grupo de processamento de fala que poderão ser muito úteis para futuros estudos e implementações na área de codificação de sinais de fala.

O capítulo 4 introduz o domínio cepstral, mostrando como levamos um sinal a este domínio e sua importância. Podemos ver também como transformamos coeficientes LPC em coeficientes cepstrais, que são muito utilizados em reconhecimento de sinais de fala. No capítulo 5 encontram-se as conclusões que puderam ser tiradas ao longo deste trabalho. Há também um apêndice, que conterá arquivos de sinal de voz explicitados no decorrer deste trabalho, além de uma descrição das rotinas implementadas e um guia de como utilizá-las.

2. CODIFICAÇÃO POR FORMATO DE ONDA

Os codificadores de sinais de fala podem ser divididos em três classes, os codificadores por forma de onda, os codificadores paramétricos e os codificadores híbridos. Os codificadores de forma de onda geralmente operam a altas taxas de transmissão e garantem alta qualidade do sinal de fala. Já os codificadores paramétricos operam a menores taxas mas tornam o sinal de fala robotizado. Os codificadores híbridos utilizam ambas as técnicas e geralmente levam a uma boa qualidade de sinal de fala com uma taxa de transmissão intermediária. Isso é ilustrado na figura 2.1, que mostra a variação da qualidade do sinal sintetizado em função da taxa de transmissão para cada uma dessas três classes de codificadores.

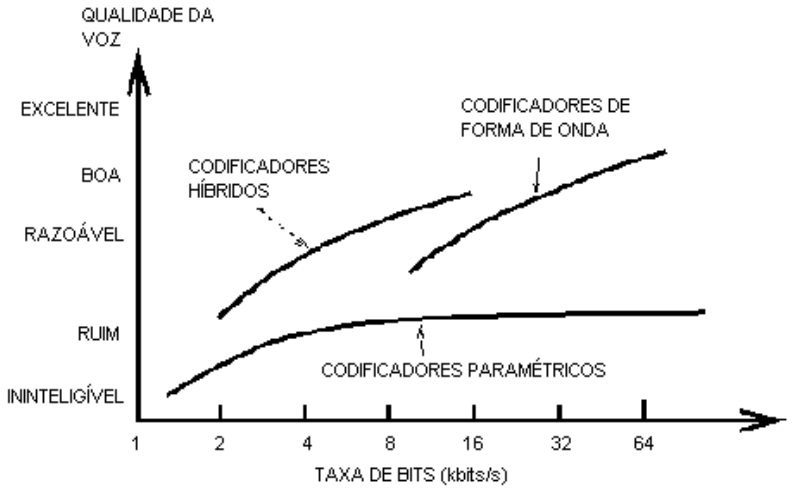

Figura 2.1: Qualidade da voz x Taxa de bits [8].

Na codificação por formato de onda, não há a necessidade de sabermos como o sinal foi gerado, já que não precisamos modelar o trato vocal. Por esse motivo, geralmente este tipo de codificador é de menor complexidade do que os outros. A taxa de transmissão deste tipo de codificador geralmente fica acima de 16 kbps. A codificação por formato de onda pode se dar tanto no domínio do tempo como no domínio da freqüência como poderemos ver adiante.

2.1 Teoria da Amostragem

Para convertermos um sinal analógico para o formato digital, devemos amostrá-lo no tempo. Para que não haja perda de informação, segundo o teorema de Nyquist, a freqüência de amostragem f_a deve ser pelo menos duas vezes maior do que a maior freqüência contida no espectro do sinal. Assim, normalmente passa-se o sinal a ser amostrado por um filtro passa-baixas a fim de limitar o seu espectro de freqüência na área de interesse da aplicação a ser feita.

No caso de sinais de fala, podemos utilizar um filtro com uma freqüência de corte de 3,4kHz e uma taxa de amostragem de 8kHz. Há diversos padrões de amostragem, que dependem do tipo de sinal a ser digitalizado. Por exemplo, um “cd-player” utiliza uma freqüência de amostragem de 44,1kHz, já que os sinais que compõem uma música possuem componentes de mais alta freqüência gerados por instrumentos musicais. Na figura 2.2 podemos ver os passos que devem ser tomados para a digitalização de um sinal analógico.

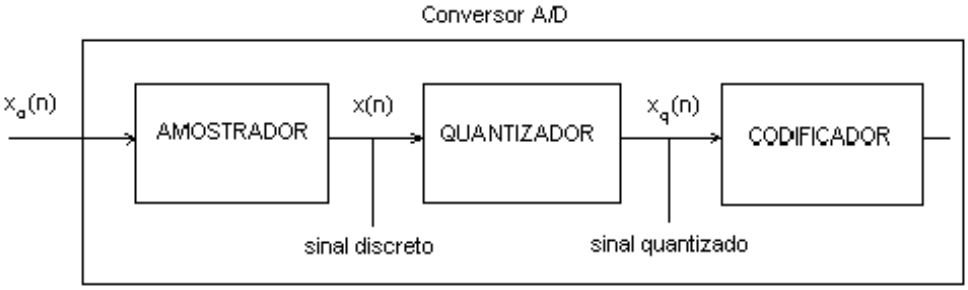

Figura 2.2: Digitalização de um sinal analógico [5].

2.2 Quantização

Como podemos ver no diagrama da figura 2.2, após amostrarmos o sinal original, devemos quantizá-lo de alguma maneira. Este processo introduz erros ao sinal, já que há uma limitação em se representar um sinal analógico por meio de bits, dada pelo número de bits utilizados para representar cada amostra do sinal. Assim, quanto mais bits forem utilizados, menor será o intervalo de quantização e conseqüentemente, menor será o erro introduzido.

A diferença entre a entrada não quantizada e a saída quantizada é o que chamamos de erro ou ruído de quantização que deve ser minimizado. Para isso, há algumas técnicas de quantização desenvolvidas e que se aplicam bem a determinados tipos de sinal como por exemplo, quantização uniforme, logarítmica, não uniforme e vetorial. O objetivo é fazer com que o tipo de quantização escolhida se adapte bem às estatísticas do sinal de fala de maneira que se consiga um desempenho ótimo, com a maior qualidade e menor taxa de bits possíveis.

2.2.1 Quantização Linear

Quantizadores lineares ou uniformes são aqueles em que os níveis de quantização do sinal amostrado são eqüidistantes sendo os mais simples de serem implementados. O problema da quantização linear, é que se tivermos um sinal de baixa amplitude e o intervalo de quantização não for suficientemente pequeno, introduziremos um erro muito grande ao sinal original.

Na figura 2.3 podemos ver a característica de transferência de um quantizador linear com um intervalo de quantização Δ assim como o erro obtido na quantização em função do nível do sinal de entrada.

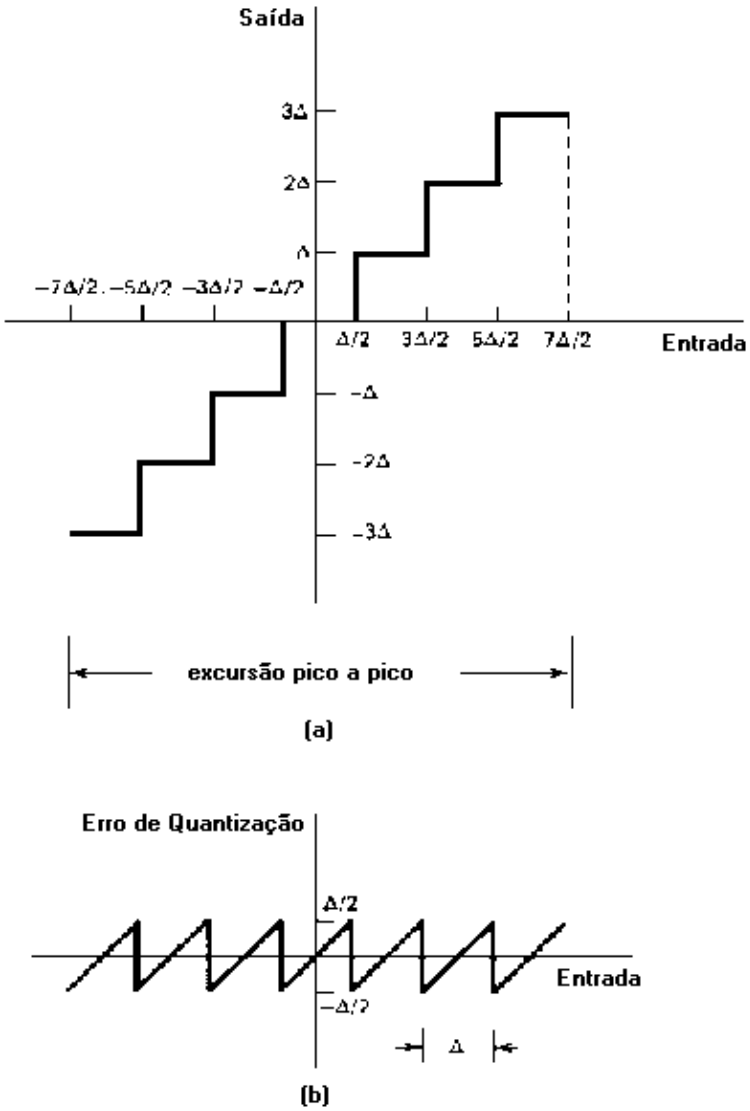

Figura 2.3: (a) Quantização uniforme; (b) Erro de quantização [5].

2.2.2 Quantização Logarítmica

Para alguns tipos de aplicações como por exemplo na quantização de sinais de fala, poderemos ter sinais que tem sua amplitude variando numa ordem de 1 para 1000. Para quantizarmos estes tipos de sinais precisamos trabalhar com um quantizador que cubra toda esta faixa. O que ocorre é que se não usarmos um grande número de bits para a quantização, o intervalo ficará muito grande e conseqüentemente teremos um erro de quantização grande.

Como em sinais de fala a probabilidade de termos amostras de baixas amplitudes é maior e o nosso aparelho auditivo possui uma percepção de amplitude logarítmica, dá-se uma maior ênfase aos sinais de baixa amplitude [5]. Para isso, utiliza-se intervalos de quantização não uniformes que crescem a medida que a amplitude do sinal aumenta.

Um método simples de obtermos isto é aplicar ao sinal um compressor com características logarítmicas antes da quantização. Então, o sinal comprimido pode ser uniformemente quantizado. Na saída do sistema o sinal é passado por um expansor, cuja característica de transferência é a inversa do compressor. Esta técnica é conhecida como "companding". As grandes vantagens são a simplicidade de implementação e a boa performance.

Há dois tipos de compressão muito utilizados, que são a lei μ e a lei A. A quantização do tipo "A-law" é utilizada no sistema PCM de telefonia europeu. Já a "μ-law" é utilizada no sistema PCM de telefonia americano, canadense e japonês. Estes dois tipos são bastante similares e as suas características de transferência são mostradas nas equações abaixo e na figura 2.4 para um valor de A=87,56 e de μ=255, que são os valores típicos utilizados em sistemas de telefonia. Usando esses esquemas é possível obter fala de qualidade utilizando 8 bits para representar cada amostra do sinal original.

$$\mu\text{-}law\text{: } |y| = \frac{\log(1+\mu|s|)}{\log(1+\mu)} \tag{2.1}$$

$$A\text{-}law\text{: } |y| = \frac{1+\log A|s|}{1+\log A} \tag{2.2}$$

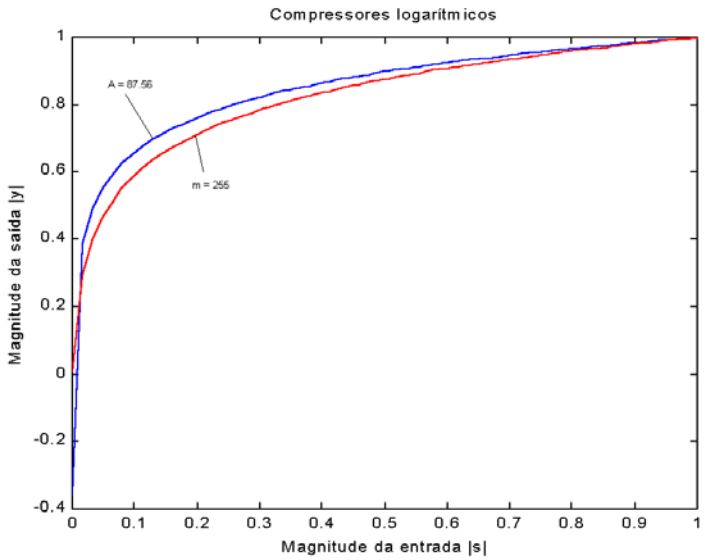

Figura 2.4: compressores do tipo μ-*law* e *A*-*law* [9].

2.2.3 Quantização Não Uniforme

Vimos que dependendo do tipo de sinal a ser quantizado, podemos utilizar compressores, de modo a diminuir o erro de quantização para sinais de baixa amplitude. Se pudermos obter a pdf (Função de Densidade de Probabilidade) do tipo de sinal a ser quantizado, podemos utilizar um quantizador que minimize o erro de quantização para aquele tipo de sinal conhecido. Isso é chamado de quantização não uniforme.

2.2.4 Quantização Vetorial

Os métodos até agora descritos quantizam cada amostra independentemente das outras, sendo chamados de quantização escalar. Um outro método é a quantização vetorial. Neste caso fazemos a quantização de um bloco do sinal original. Este tipo de quantização se mostra muito eficiente quando as amostras a serem quantizadas são estatisticamente dependentes uma das outras.

Na quantização vetorial, um segmento do sinal original, representado por um vetor N dimensional será comparado a vários outros vetores conhecidos. O vetor que mais se aproximar ao do sinal original representará o mesmo.

A quantização vetorial envolve algoritmos mais complexos e é mais sensível a erros de transmissão, já que o recebimento de um índice errado representa o erro de todo um segmento do sinal original, porém através dela consegue-se uma grande redução na taxa de transmissão.

2.3 PCM

A codificação PCM (“pulse code modulation”) é a forma mais simples de codificação de formato de onda. Consiste simplesmente em amostrar e quantizar as amostras do sinal original. Geralmente limita-se os sinais de fala na faixa de 4 kHz e amostra-se o sinal a 8 kHz, podendo essas freqüências assumir outros valores, dependendo da aplicação. Se utilizarmos uma quantização do tipo linear com 12 bits por amostra, obtemos fala de boa qualidade a uma taxa de 96 kbps. Esta taxa de transmissão pode ser reduzida se utilizarmos a quantização não uniforme.

Em sistemas comerciais de codificação de fala, utiliza-se a quantização logarítmica. Estes quantizadores produzem uma relação sinal ruído que é aproximadamente constante em uma faixa grande de níveis de entrada. Através deste tipo de quantização obtemos sinais de fala de excelente qualidade a uma taxa de 64 kbps, que é o padrão na telefonia digital.

A grande vantagem da codificação PCM é a sua pouca complexidade e alta qualidade obtida, a desvantagem é a alta taxa de transmissão que o impossibilita de ser utilizado em certos tipos de aplicações que necessitam de baixas taxas de transmissão.

2.4 DPCM

A codificação DPCM (“differencial pulse code modulation”) se utiliza do fato de que os sinais de fala apresentam uma alta correlação entre as sucessivas amostras para obter uma menor taxa de transmissão do que a codificação PCM. Esta alta correlação entre as sucessivas amostras possibilita a utilização de um preditor linear, para estimarmos o valor da amostra atual com base nas amostras passadas.

Assim, faz-se uma estimativa da amostra atual e compara-se esta estimativa com a amostra original, sendo transmitido apenas o sinal de erro entre esses dois sinais. Como esse sinal de erro possui amplitudes mais baixas do que a amostra original, podemos utilizar menos bits em sua representação, o que leva a codificação DPCM a uma menor taxa de transmissão do que a codificação PCM. Utilizando-se um compressor logarítmico e um quantizador de 4 bits para o sinal de erro a ser transmitido, a codificação DPCM resulta em alta qualidade de sinal de fala a uma taxa de 32 kbps.

Na figura 2.5 podemos ver o diagrama de um codificador DPCM e de um decodificador, que realiza o processo inverso ao codificador, somando o erro transmitido ao valor da estimativa realizada pelo preditor linear.

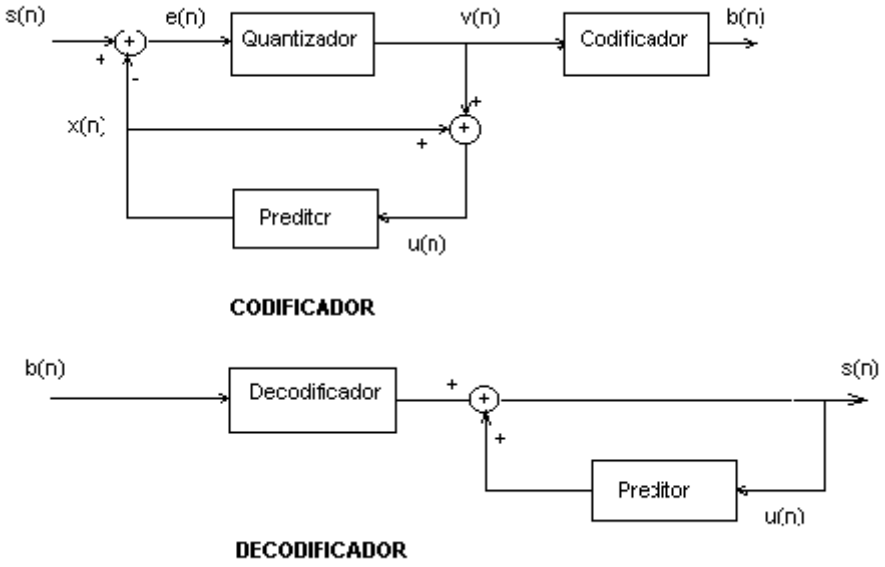

Figura 2.5: Codificação DPCM [5].

2.5 ADPCM

A codificação ADPCM (“adaptative differencial pulse code modulation”) apresenta vantagens em relação à codificação DPCM. Isto porque o preditor linear e o quantizador se adaptam ao sinal que está sendo codificado.

O quantizador tem o seu intervalo de quantização fixado em função do sinal de erro a ser quantizado, sendo este intervalo mudado dinamicamente. Para isso há uma lógica de controle que determina o intervalo de quantização ideal, tal que o erro inserido neste processo seja minimizado.

Já o preditor linear sofre alteração em seus coeficientes, já que os sinais de fala não são estacionários. Sendo assim são calculados os coeficientes tal que o erro da estimativa do sinal seja mínimo. Isso é feito através de uma lógica de controle que atua sobre o filtro de predição linear.

Este tipo de codificação é chamada de adaptativa, devido a esta propriedade de poder ter os seus parâmetros mudados em função do sinal a ser codificado. Na década de 80 o CCITT determinou o padrão G721 que é a codificação ADPCM a uma taxa de 32 kbps que oferece qualidade de reconstrução equivalente à codificação PCM a uma taxa de 64 kbps. Na figura 2.6 podemos ver o diagrama de um codificador e um decodificador ADPCM.

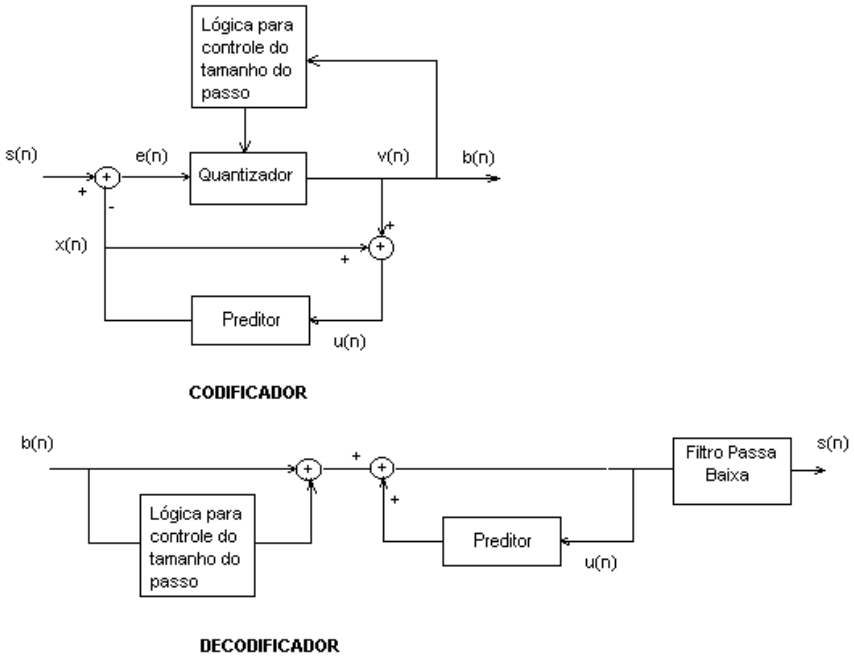

Figura 2.6: Sistema ADPCM [9].

2.6 Codificação por Sub-Banda

A codificação por sub-bandas é a técnica mais simples de implementação de codificação no domínio da freqüência. Em um codificador por sub-bandas, passa-se o sinal por um banco de filtros que separa o sinal em diversas bandas de freqüência. Em seguida, as sub-bandas são codificadas através de uma das formas descritas anteriormente no domínio do tempo. Na prática não é necessário utilizarmos muitas bandas. São utilizadas de 4 a 8 bandas sendo o sinal de cada banda codificado independentemente.

O número de bits atribuído a cada sub-banda pode ser variado de acordo com a importância perceptual da banda. No caso de sinais de fala, a maior parte da energia está concentrada em baixas freqüências, então utiliza-se um maior número de bits para as bandas de menor freqüência e um menor número para as bandas de alta freqüência. Com isso podemos dar uma pesagem diferente às faixas de freqüência na reconstrução do sinal [1]. Uma vantagem da codificação por sub-bandas, é que a quantização do ruído produzida em uma banda é confinada àquela banda.

Se utilizarmos por exemplo duas sub-bandas, uma taxa de amostragem de 8 kHz, 4 bits em codificação DPCM para a sub-banda de mais baixa freqüência e 2 bits para a sub-banda de freqüência mais alta, obtemos uma taxa de transmissão de 24 kbps. Na figura 2.7 podemos ver o diagrama de um codificador por sub-banda.

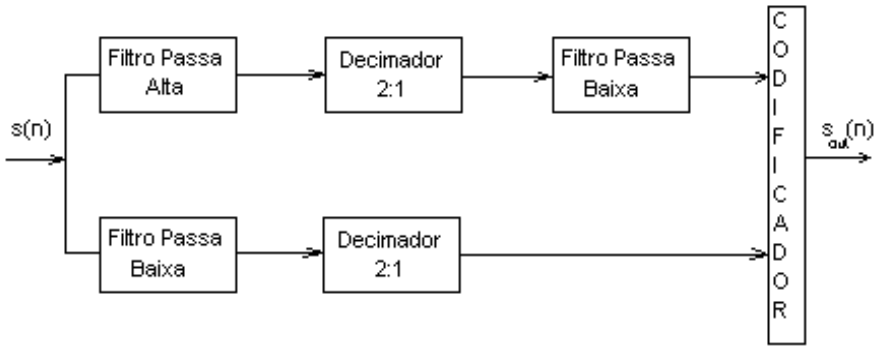

Figura 2.7: Exemplo de codificação por sub-bandas [9].

3. CODIFICAÇÃO LPC

A codificação LPC é o tipo mais simples de codificação paramétrica existente, sendo a base para outros tipos de codificações paramétricas mais complexas e eficientes. Neste tipo de codificação, dividimos o sinal em diversos segmentos da ordem de 10 a 30ms e fazemos a análise sobre cada um desses segmentos. Os parâmetros que devem ser extraídos de cada segmento para que possamos reconstruir esse sinal são os coeficientes do filtro de predição linear, o ganho, o fato do sinal de fala ser do tipo vozeado ou não e o valor da freqüência fundamental, denominado “pitch” para o caso de sinal do tipo vozeado.

3.1 Propriedades Básicas dos Sinais de Fala

Os sinais de fala, como dissemos anteriormente podem ser classificados como vozeados ou não vozeados. Essa classificação é de fundamental importância na codificação LPC, já que cada um deles terá um tipo de excitação diferente na síntese do sinal.

Os sinais de fala do tipo vozeado são aqueles que são gerados através da vibração de nossas cordas vocais como por exemplo todos as vogais que pronunciamos. Esses sinais têm como característica o fato de que possuem uma periodicidade bem definida, que é o “pitch”. Na figura 3.1 podemos ver um segmento de 30ms da vogal “a”.

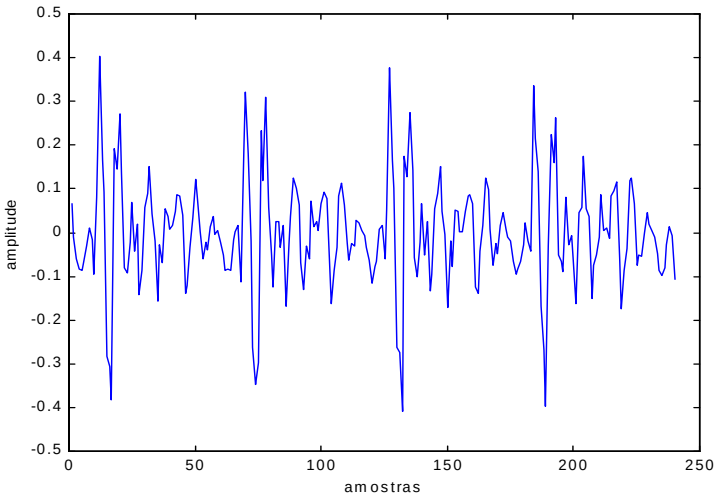

Figura 3.1: Sinal de fala do tipo vozeado.

Já os sinais do tipo não vozeados são gerados pela passagem de ar em alta velocidade através do trato vocal enquanto a glote está parcialmente aberta. Este tipo de sinal não apresenta quase nenhuma periodicidade. Na figura 3.2 podemos ver um segmento de 30ms de um sinal do tipo não vozeado.

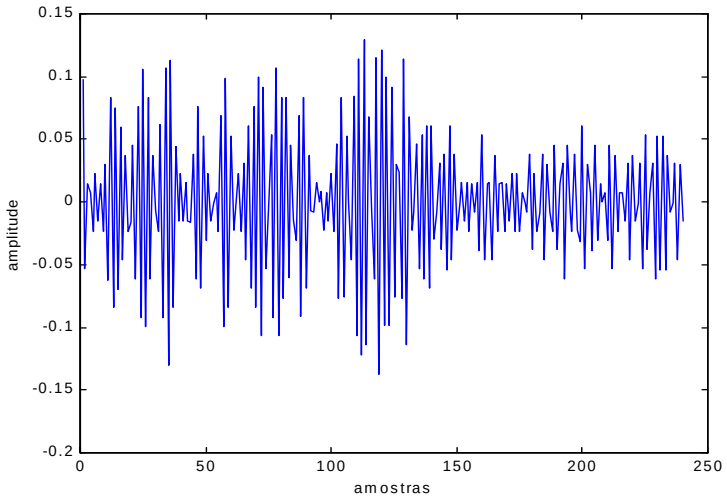

Figura 3.2: Sinal de fala do tipo não vozeado.

3.2 Modelo da Produção de Fala

Na codificação LPC, modelamos o sistema de produção de fala da seguinte maneira: para sinais de fala do tipo vozeado, excitamos um filtro que representa o modelo do trato vocal com pulsos glotais espaçados por um período de "pitch" e multiplicados por um ganho. Para sinais do tipo não vozeados, excitamos este mesmo filtro com ruído branco multiplicado por um ganho. Na saída deste filtro, obteremos o sinal de fala. Isto está ilustrado no diagrama da figura 3.3.

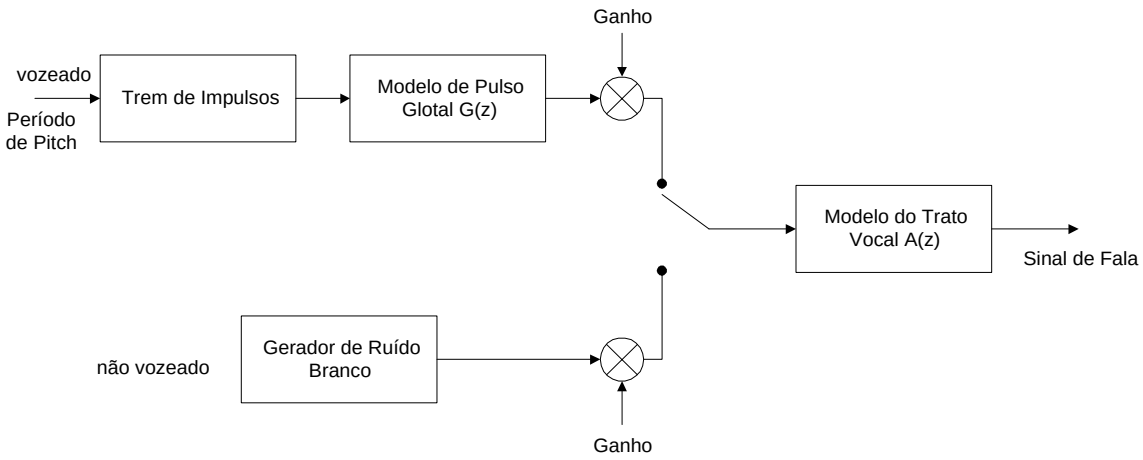

Figura 3.3: Modelo da produção de fala.

O modelo de predição linear assume que o sinal de fala é um processo auto-regressivo descrito por:

$$s(n) = \sum_{i=1}^{p} a_i s(n-i)\,, \tag{3.1}$$

onde *s(n)* é o sinal de fala, a_i são os coeficientes de predição e *p* é a ordem do preditor.

O modelo do trato vocal pode ser representado por um filtro de síntese "all-pole", dado pela seguinte função de transferência:

$$S(z) = \frac{g}{1 - A(z)} U(z)\,, \tag{3.2}$$

$$\text{onde } A(z) = \sum_{i=1}^{p} a_i z^{-i}\,. \tag{3.3}$$

Na figura 3.4, podemos ver o diagrama de um filtro de análise por predição linear. Neste caso, *s(n)* é o sinal de fala original, *p* é a ordem do sistema e *e(n)* é o sinal de erro.

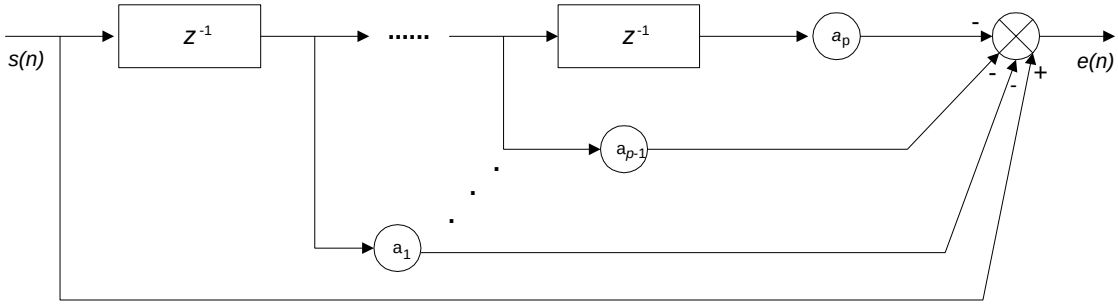

Figura 3.4: Filtro de análise por predição linear.

O sinal de erro *e(n)* é dado por:

$$e(n) = s(n) - \sum_{i=1}^{p} a_i s(n-i) \tag{3.4}$$

Na análise LPC, devemos encontrar o valor dos coeficientes a_i, tal que minimizemos uma função de custo do erro *e(n)*. Para isso, podemos utilizar o método da autocorrelação ou o método da covariância, que serão vistos mais adiante.

Como os sinais de fala não são sinais estacionários, devemos trabalhar com segmentos do sinal considerados aproximadamente estacionários, e atualizar os coeficientes do modelo do trato vocal para cada um desses segmentos.

3.3 Descrição de um Codificador LPC

O sistema LPC que será descrito, foi desenvolvido através do Software MATLAB. O sistema é constituído de vários módulos básicos, que realizam funções específicas e são chamados pelo módulo principal, que é o corpo do programa. Essa organização foi feita para que haja facilidade de compreensão do sistema por parte de outras pessoas, e para que haja maior facilidade na correção de erros e adaptações no sistema.

O sistema de codificação e síntese desenvolvido consiste de 11 blocos básicos, conforme a figura 3.5.

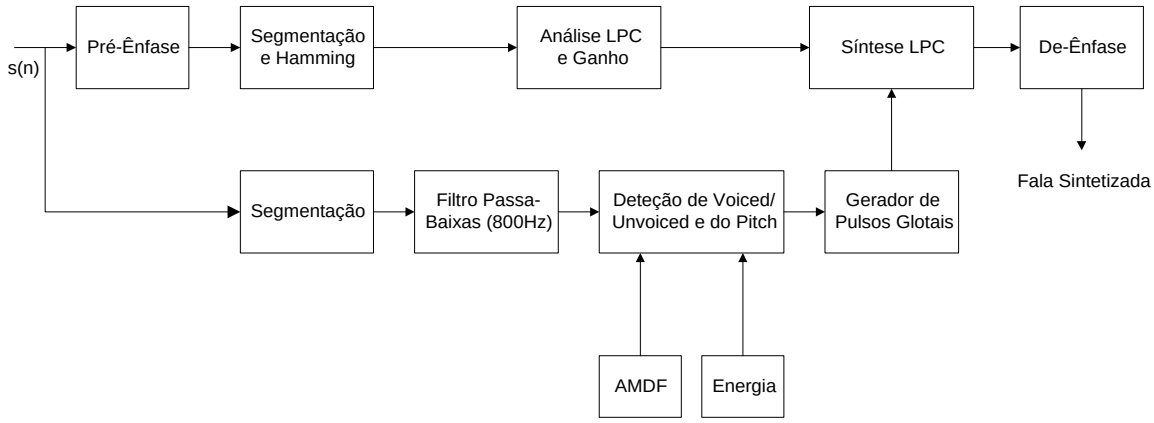

Fig. 3.5: Sistema de codificação e síntese LPC.

O funcionamento do sistema se dá da seguinte maneira: o sinal de voz amostrado *s(n)* passa pelo filtro de pré-ênfase, e é segmentado em janelas de tamanho determinado pelo usuário. Depois, cada um desses segmentos é multiplicado por uma janela de Hamming. Após esse tratamento dado ao sinal original, ele é processado por uma rotina que irá extrair os coeficientes LPC e o ganho de cada uma das janelas do sinal. A ordem do sistema, que determinará o número de coeficientes LPC é fornecida pelo usuário, sendo sua mudança bastante flexível.

Para o cálculo do “pitch”, o sinal não passa pela janela de Hamming, já que a mesma deforma o sinal no domínio do tempo, mudando as características do “pitch” de cada segmento de fala. Assim, no cálculo do “pitch”, o sinal segmentado em janelas passa por um filtro passa-baixa com freqüência de corte em 800 Hz, para a eliminação de sinais de alta freqüência, já que o “pitch” possui baixa freqüência (da ordem de 100 a 200 Hz). Após essa etapa, o sinal é processado para que possamos identificar se ele é do tipo vozeado ou não. Caso seja do tipo vozeado, deve-se determinar o valor do “pitch” do sinal contido nesta janela. Os algoritmos utilizados para isso, são o cálculo da energia do sinal e o AMDF, que serão mostrados e exemplificados adiante.

Depois do sinal ter passado por essas rotinas, teremos para cada janela do sinal original, um conjunto de parâmetros que irão possibilitar a reconstrução desse sinal de

forma aproximada. Esses parâmetros são os coeficientes LPC, o ganho, o tipo do sinal de fala (se vozeado ou não) e o valor do “pitch” (no caso de vozeado).

O módulo responsável pela síntese, recebe esses parâmetros e através de um algoritmo transforma-o novamente em sinal de fala. Para isso, utiliza-se um filtro em que seus coeficientes são os coeficientes LPC calculados e excitamos este filtro com ruído branco (no caso de sinal não vozeado), ou com pulsos glotais (no caso de sinal vozeado) que são gerados por uma outra rotina. A periodicidade da excitação deste filtro é determinada pelo valor do “pitch” calculado anteriormente. Depois que a síntese do sinal foi feita, o sinal passa por um filtro de de-ênfase. A seguir apresentaremos uma descrição mais sucinta de cada um desses blocos do sistema.

3.3.1 Pré-Ênfase

Entende-se por pré-ênfase a passagem do sinal por um filtro que concentra a energia relativa ao espectro de alta freqüência do sinal, reduzindo a sua componente DC. Isto é feito utilizando-se um filtro de primeira ordem que introduz um zero próximo a $\omega = 0$.

Emprega-se a pré-ênfase por dois motivos. Um deles é para a prevenção de instabilidade numérica. Isso porque como o sinal de fala é predominantemente composto de baixas freqüências, o modelo LP poderá resultar numa matriz de autocorrelação mal condicionada, podendo levar a instabilidades numéricas. Um filtro de primeira ordem neste caso diminui um pouco a correlação do sinal, diminuindo a possibilidade de instabilidade numérica [1].

O segundo motivo é que o componente de fase mínima do sinal glotal pode ser modelado por um filtro simples de 2 pólos próximos a z=1. Então a característica dos lábios com o seu zero perto de z=1, tende a cancelar os efeitos espectrais de um dos pólos glotais. Introduzindo, um segundo zero próximo a z=1, as contribuições espectrais da laringe e dos lábios seriam eliminadas, fazendo com que a análise LPC correspondesse apenas ao canal de fala humano. No entanto, apesar do filtro de pré-ênfase, o espectro de predição linear não fica totalmente livre dos efeitos da laringe e dos lábios [1].

O filtro de pré-ênfase é dado pela seguinte função de transferência:

$$H(z) = 1 - \mu z^{-1}, \tag{3.5}$$

onde μ varia tipicamente de 0,9 a 1,0 [1].

Na figura 3.6, podemos ver a resposta em magnitude do filtro de pré-ênfase para μ = 0,95.

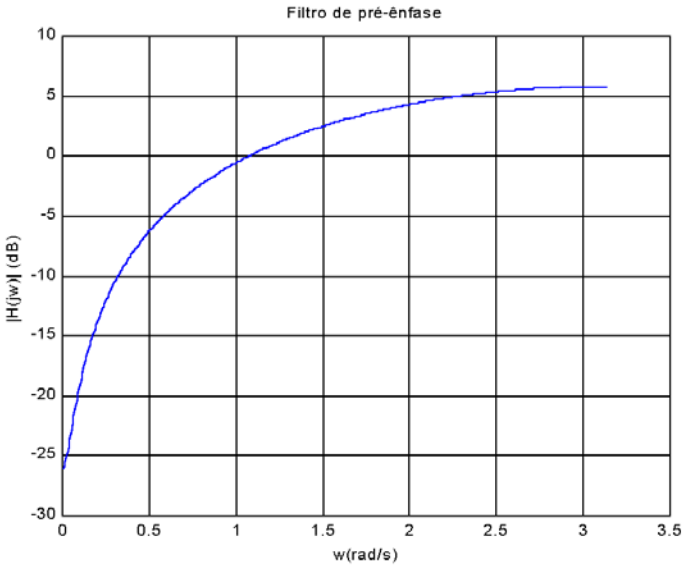

Figura 3.6: Resposta em magnitude do filtro de pré-ênfase com μ=0,95 [9].

3.3.2 Segmentação e Janelamento

Na codificação LPC, processamos o sinal original segmento a segmento. Cada um desses segmentos é analisado e dessa análise resultam parâmetros que representam esse segmento como um todo. Portanto, devemos assumir que este segmento é aproximadamente estacionário. Para isso devemos utilizar segmentos da ordem de 10 a 30ms como será visto no próximo capítulo.

Para segmentarmos o sinal original, devemos multiplicá-lo por uma janela com o tamanho de amostras que se deseja para cada segmento. Essa janela vai sendo deslocada de modo a obtermos vários segmentos que compõem o sinal original. Existem vários tipos de janelas que podemos usar para este fim. Algumas dessas janelas serão apresentadas a seguir.

- *Janela Retangular*

É o tipo de janela mais simples, sendo expressa pela seguinte função:

$$w(n) = \begin{cases} 1, \text{ para } 0<n\leq N \\ 0, \text{ para } n>N \end{cases} \qquad (3.6)$$

Esse tipo de janela gera uma grande descontinuidade no sinal, já que como pode ser visto na figura 3.7, este tipo de janela possui uma transição abrupta no início e no final da janela. Assim, apesar do sinal não ser distorcido no domínio do tempo por este tipo de janela, ele fica distorcido no domínio da freqüência devido a essas transições abruptas.

Este tipo de janelamento é utilizado para determinarmos o “pitch” do segmento de fala através do AMDF (“Average Magnitude Difference Function**”)** que será visto adiante. Isto porque o AMDF extrai a freqüência fundamental do segmento de fala através da função autocorrelação, e se deformamos a forma de onda no domínio do tempo, estaremos variando o valor da função autocorrelação do sinal.

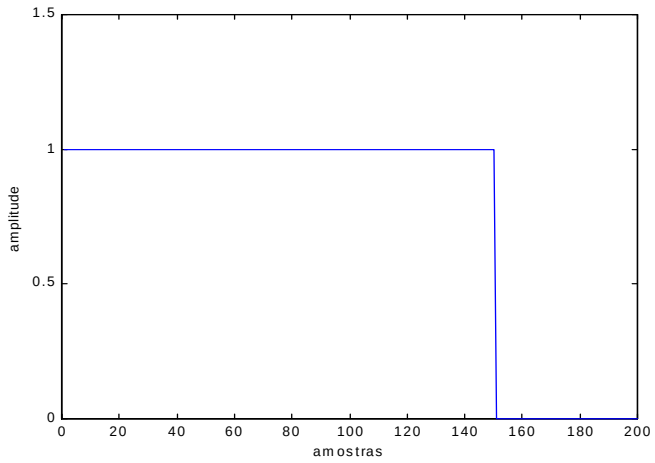

Figura 3.7: Janela retangular para *N*=150.

- *Janela do Tipo Hamming*

Na janela do tipo Hamming há uma atenuação gradativa nas extremidades, de forma a diminuir a descontinuidade do sinal. Isso faz com que o sinal seja distorcido no domínio do tempo, mas apresente menos distorções no domínio da freqüência. Na figura 3.8, podemos ver a janela do tipo Hamming. Ela é obtida através da seguinte expressão:

$$w(n) = \begin{cases} 0{,}54\text{-}0{,}46\cos\left(\dfrac{2\pi n}{N-1}\right), \text{ para } 0 \le n \le N\text{-}1 \\ 0, \text{ para } n \ge N \end{cases} \qquad (3.7)$$

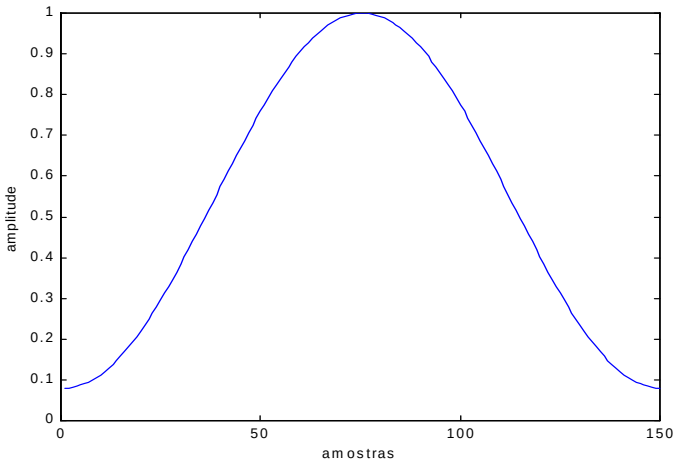

Figura 3.8: Janela do tipo Hamming para *N*=150.

- *Janela do Tipo Hanning*

É muito parecida com a janela do tipo Hamming, conforme podemos ver na figura 3.9, sendo definida como:

$$w(n) = \begin{cases} 0{,}5\text{-}0{,}5\cos\left(\dfrac{2\pi n}{N+1}\right), \text{ para } 0 \le n \le N\text{-}1 \\ 0, \text{ para } n \ge N \end{cases} \qquad (3.8)$$

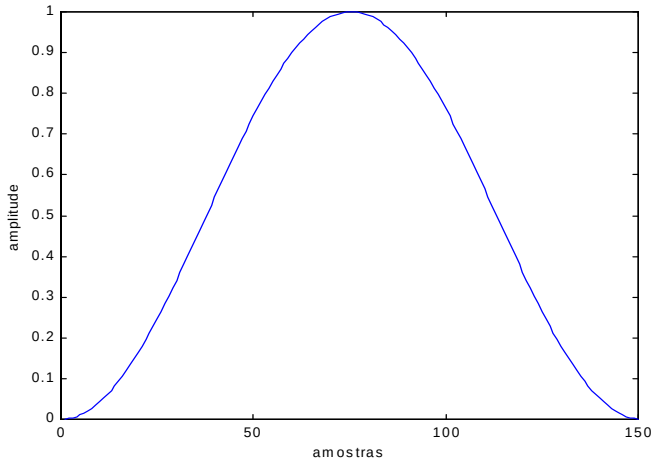

Figura 3.9: Janela do tipo Hanning para *N*=150.

- *Janela do Tipo Blackman*

A janela do tipo Blackman pode ser vista na figura 3.10, sendo definida como:

$$w(n) = \begin{cases} 0{,}42\text{-}0{,}5\cos\left(\frac{2\pi n}{N-1}\right) + 0{,}8\cos\left(\frac{4\pi n}{N-1}\right), \text{ para } 0 \le n \le N\text{-}1 \\ 0, \text{ para } n \ge N \end{cases} \quad (3.9)$$

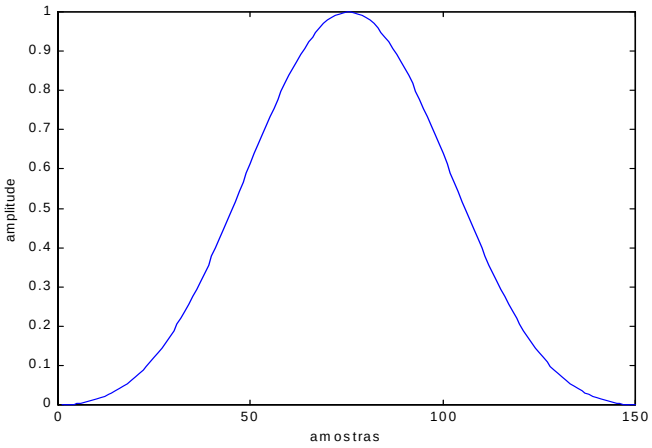

Figura 3.10: Janela do tipo Blackman para *N*=150.

- *Janela do Tipo Bartlett*

A janela do tipo Bartlett, que é quase idêntica à janela triangular, pode ser vista na figura 3.11, sendo definida como:

$$w(n) = \begin{cases} \frac{2(k+1)}{n-1}, \quad \text{para} \quad 1 \le k \le \frac{N+1}{2} \\ 2\text{-}\frac{2(k-1)}{n-1}, \quad \text{para} \quad \frac{N+1}{2} < k \le N \\ 0, \text{ para } n > N \end{cases} \quad (3.10)$$

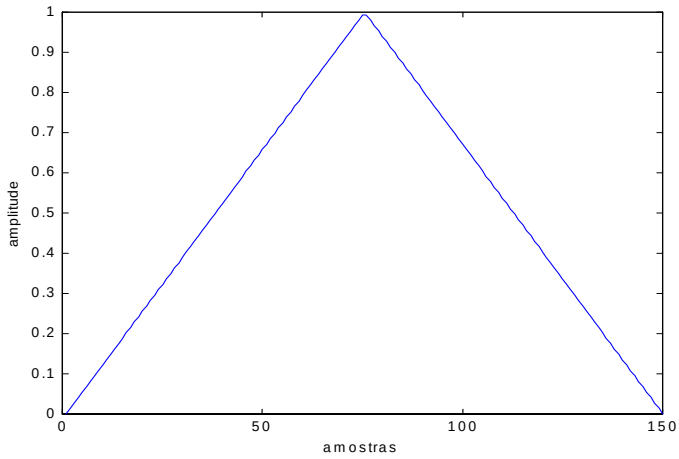

Figura 3.11: Janela do tipo Bartlett para *N*=150.

A segmentação pode ser feita com ou sem superposição. Se não utilizarmos superposição, cada segmento se iniciará na amostra posterior à última amostra do segmento anterior, conforme a figura 3.12.

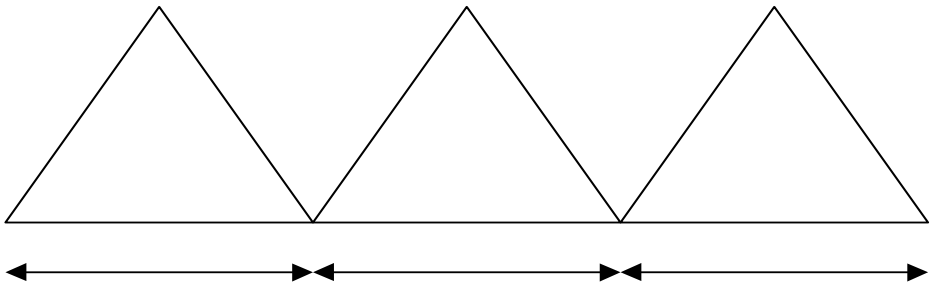

Figura 3.12: Segmentação sem superposição.

A utilização da segmentação poderá se dar com diferentes taxas, como por exemplo 50% ou 66,7%. Esta taxa é dada pela divisão entre o passo *I* e o tamanho do segmento *L*. Assim, se estivermos utilizando um segmento de tamanho 200 amostras e um passo de 100 amostras, estaremos utilizando uma taxa de superposição de 50%. Na figura 3.13 podemos ver como seria o janelamento com superposição.

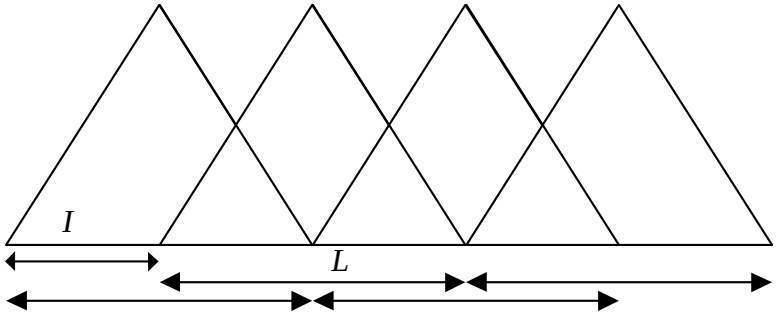

Figura 3.13: Segmentação com superposição.

No próximo capítulo, poderemos ver a influência do tipo de janelamento utilizado na codificação LPC, bem como a utilização ou não da superposição.

3.3.3 Análise LPC e Ganho

Como vimos anteriormente, na análise LPC estamos interessados em encontrar os valores dos *p* coeficientes, onde *p* é a ordem do filtro de predição linear, que minimizem uma função de custo do erro.

Em um preditor linear de ordem *p*, a amostra atual da sequência de fala é estimada através de uma combinação linear das *p* amostras passadas. Os parâmetros da predição linear são obtidos minimizando-se o erro médio quadrático da predição:

$$\frac{\partial \varepsilon}{\partial a_i} = 0\,,\ \ i = 1,2,...,p \tag{3.11}$$

onde $\varepsilon = E\left[e^2(n)\right] = E\left[(s(n) - y(n))^2\right]$.

Há pelo menos dois métodos para resolvermos essa minimização. Um deles é o da autocorrelação e o outro é o da covariância. Pelo método da autocorrelação, temos que esta minimização produz uma série de equações do tipo mostrado abaixo.

$$r_s(\eta) - \sum_{i=1}^{p} a_i r_s(\eta - i) = 0\,,\ \ \eta = 1,2,...p \tag{3.12}$$

onde $r_s(\eta) = E\left[s(n+\eta)s(n)\right]$ é a sequência de autocorrelação do segmento de fala.

A mesma equação sob a forma matricial é dada por :

$\boldsymbol{R_s a} = \boldsymbol{r_s}$, que equivale a:

$$\begin{bmatrix} r_s(0) & r_s(1) & r_s(2) & \Lambda & r_s(p-1) \\ r_s(1) & r_s(0) & r_s(1) & \Lambda & r_s(p-2) \\ r_s(2) & r_s(1) & r_s(0) & \Lambda & r_s(p-3) \\ \text{M} & \text{M} & \text{M} & & \text{M} \\ r_s(p-1) & r_s(p-2) & r_s(p-3) & \Lambda & r_s(0) \end{bmatrix} \begin{bmatrix} a(1) \\ a(2) \\ a(3) \\ \text{M} \\ a(p) \end{bmatrix} = \begin{bmatrix} r_s(1) \\ r_s(2) \\ r_s(3) \\ \text{M} \\ r_s(p) \end{bmatrix} \tag{3.13}$$

O valor dos coeficientes será dados por:

$$\boldsymbol{a} = \boldsymbol{R_s}^{-1}\boldsymbol{r_s} \tag{3.14}$$

Como a matriz $\boldsymbol{R_s}$ é do tipo Toeplitz, podemos invertê-la através da recursão de Levinson Durbin, o que torna a inversão muito mais rápida computacionalmente [1].

A seqüência de autocorrelação pode ser estimada através das *N* amostras do sinal de fala usando-se um estimador do tipo polarizado, não polarizado ou não polarizado com baixa variância. Os estimadores do tipo polarizado são desejáveis, uma vez que geralmente produzem polinômios de fase mínima e por este motivo foram utilizados em nossa implementação do sistema LPC [2].

Estimador do tipo polarizado:

$$\hat{r}_s(\eta) = \frac{1}{N-|\eta|} \sum_{i=0}^{N-|\eta|-1} s(n+|\eta|)s(n), \quad \eta = 1,2,...,\text{p} \tag{3.15}$$

Estimador do tipo não polarizado:

$$\tilde{r}_s(\eta) = \frac{1}{N} \sum_{i=0}^{N-|\eta|-1} s(n+|\eta|)s(n), \quad \eta = 1,2,...,\text{p} \tag{3.16}$$

Estimador do tipo não polarizado com baixa variância:

$$\hat{r}_s(\eta) = \frac{N-|m|}{N} \sum_{i=0}^{N-|\eta|-1} s(n+|\eta|)s(n), \quad \eta = 1,2,...,\text{p} \tag{3.17}$$

Pelo método da covariância, a minimização de ε, resulta na resolução da seguinte equação:

$\boldsymbol{\Phi}^{-1}\ \phi = \boldsymbol{a}$, que eqivale a:

$$\begin{bmatrix} \phi(1,1) & \phi(1,2) & \phi(1,3) & \Lambda & \phi(1,p) \\ \phi(2,1) & \phi(2,2) & \phi(2,3) & \Lambda & \phi(2,p) \\ \phi(3,1) & \phi(3,2) & \phi(3,3) & \Lambda & \phi(3,p) \\ \text{M} & \text{M} & \text{M} & & \text{M} \\ \phi(p,1) & \phi(p,2) & \phi(p,3) & \Lambda & \phi(p,p) \end{bmatrix}^{-1} \cdot \begin{bmatrix} \phi(1) \\ \phi(2) \\ \phi(3) \\ \text{M} \\ \phi(p) \end{bmatrix} = \begin{bmatrix} a(1) \\ a(2) \\ a(3) \\ \text{M} \\ a(p) \end{bmatrix} \tag{3.18}$$

onde a é o vetor dos coeficientes de predição linear

$$\phi[i,j] = \sum_{n=p+1}^{N} s(n-i)s(n-j)$$

$$\mathbf{\Phi} = \{\phi[1,0],...,\phi[p,0]\}$$

Apesar de $\mathbf{\Phi}$ não ser uma matriz do tipo Toeplitz, ela é uma matriz simétrica e pode ser decomposta em duas matrizes, uma triangular superior e outra triangular inferior. Assim, podemos resolver o sistema pelo método da substituição ("back substitution"). No próximo capítulo, poderemos ver as comparações entre a solução do problema LPC por cada um dos métodos aqui apresentados.

No modelo LPC o ganho é utilizado para que o sinal sintetizado possua a mesma energia do sinal original. O cálculo do ganho é realizado relacionando a energia na saída do filtro de análise LPC de cada segmento com a energia do segmento de sinal original. O ganho é uma função dos coeficientes da função autocorrelação do segmento de fala analisado e dos coeficientes do filtro de análise, sendo dado pela seguinte equação [1]:

$$g(n) = \left[r_s(0) - \sum_{i=1}^{p} a_i r_s(i) \right]^{\frac{1}{2}} \tag{3.19}$$

3.3.4 "Average Magnitude Difference Function" (AMDF)

A função AMDF é utilizada para o cálculo do "pitch" do sinal de fala. Ela calcula a diferença média entre o sinal original e o sinal deslocado de τ amostras. É definida como [1]:

$$AMDF(\tau) = \frac{1}{N} \sum_{j=1}^{k} | s(j) - s(j+\tau) |, \tag{3.20}$$

onde k é uma constante que deve ser arbitrada e N é o número de amostras de cada segmento de sinal de fala.

O valor da função AMDF exibirá pontos de mínimos quando o deslocamento τ for igual ou múltiplo do período de "pitch" (para sinais do tipo vozeado). Calculamos o valor da função AMDF para uma faixa de valores de τ na faixa entre 20 e 100, o que permite o cálculo de valores de "pitch" de 2,5ms a 12,5ms para uma freqüência de amostragem de

8kHz. Na figura 3.14 podemos ver o gráfico da função AMDF para a vogal “e” com uma freqüência de amostragem de 8kHz. Podemos ver que para valores de τ abaixo de 20 não calculamos o AMDF. Isso porque se utilizássemos τ igual ou próximo a zero, o valor do AMDF seria baixo podendo resultar numa falsa detecção do “pitch”.

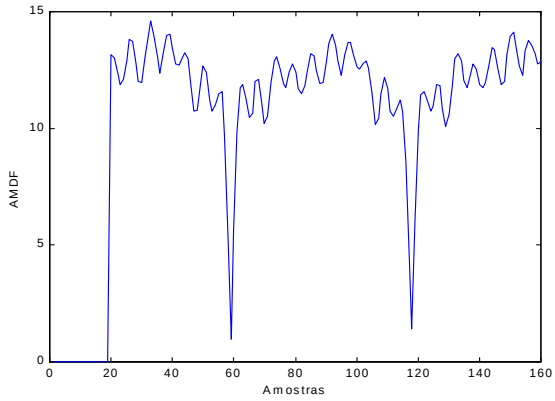

Figura 3.14: AMDF da vogal “e”.

3.3.5 Cálculo da Energia

A energia de um segmento de sinal de fala com *N* amostras pode ser calculada pela seguinte equação [1]:

$$E(n) = \sum_{k=1}^{N} s^2(k) \tag{3.21}$$

3.3.6 Detecção do “Pitch”

Para a detecção do valor do “pitch”, primeiramente passamos cada segmento de sinal de fala por um filtro passa baixa com uma freqüência de corte de 800Hz, já que o “pitch” possui baixa freqüência.

Depois, utilizamos os resultados obtidos no cálculo da energia do segmento e da função AMDF para decidirmos se o sinal é ou não do tipo vozeado e se for do tipo vozeado, calcularmos o valor do “pitch”. Na figura 3.15 podemos ver o gráfico da função AMDF para que possamos exemplificar como fazer a decisão do tipo de sinal e calcular o valor do “pitch” em sinais do tipo vozeado.

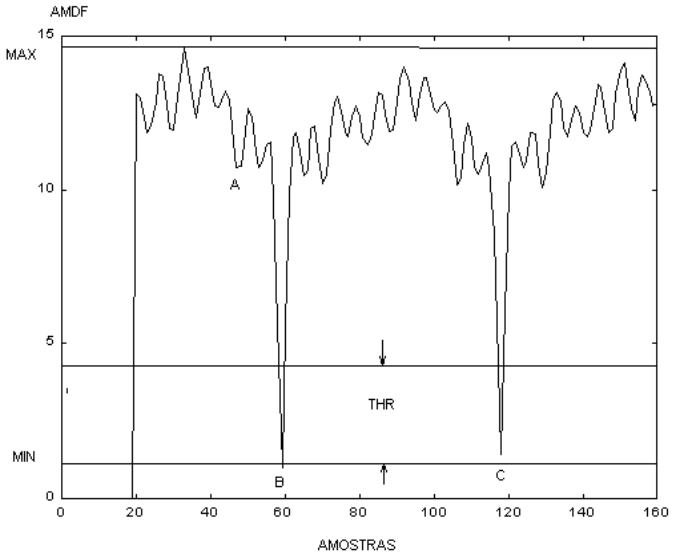

Figura 3.15: Gráfico da função AMDF.

Na figura 3.15, podemos ver que há mínimos locais e um mínimo global. Para determinarmos quais pontos são os candidatos a determinar o valor do "pitch", calculamos um limiar dado por:

$$thr = \frac{máx(AMDF) - \min(AMDF)}{k} \tag{3.22}$$

onde *k* é uma constante.

No nosso caso utilizamos um valor de *k*=8 obtido experimentalmente baseado em testes na rotina implementada. Então, o ponto que for o primeiro mínimo local que estiver abaixo do limiar definido por *min(AMDF)+thr* será candidato a determinar o valor do "pitch". Assim, na figura 3.15, o ponto B e o ponto C seriam candidatos a determinarem o valor do "pitch". Já o ponto a, não seria candidato, já que está acima do limiar definido.

Para fazermos a decisão se o sinal é do tipo vozeado ou não, implementamos o fluxograma mostrado a seguir, onde *rat* é definido como a divisão entre o valor máximo da função AMDF pelo valor do ponto candidato a determinar o "pitch", ou seja:

$$rat = \frac{\max(AMDF)}{\text{minvalue}(AMDF)} \tag{3.23}$$

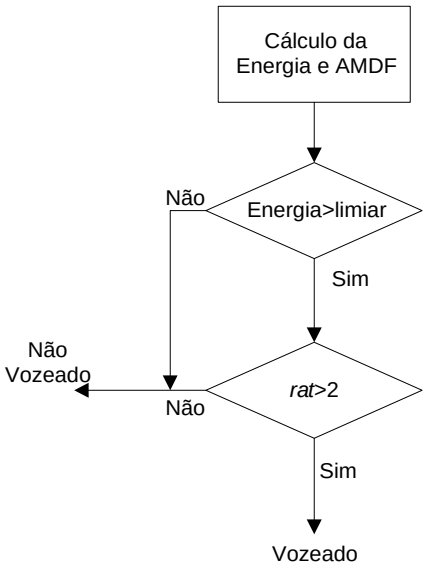

Figura 3.16: Algoritmo de decisão do tipo de sinal de fala.

O limiar de energia pode ser fixado arbitrariamente de acordo com testes práticos realizados. O limiar utilizado foi de 0,01. Se o segmento tiver uma energia abaixo deste limiar, o mesmo já é considerado do tipo não vozeado não necessitando continuarmos o algoritmo. O valor de *rat* também é obtido experimentalmente. No nosso caso utilizamos um valor de *rat* igual a 2.

Se o sinal for considerado do tipo vozeado, o período de "pitch" será dado pelo valor da abcissa do ponto definido anteriormente como minvalue (ponto B). No caso da figura 3.15, o período do "pitch" seria de aproximadamente $60T_s$, onde T_s é o período de amostragem.

3.3.7 Gerador de Pulsos Glotais

Como sabemos, nossa voz pode ser classificada como vozeada ou não vozeada. Para cada um desses casos teremos um tipo de excitação diferente no sistema LPC para obtermos a síntese da voz. Para o caso de sons não vozeados utiliza-se uma fonte de ruído branco com distribuição normal, média zero e variância 1. Já para os sons do tipo vozeado, utiliza-se pulsos glotais periódicos, que devem se aproximar do pulso glotal humano. Esses pulsos glotais devem se repetir a cada período de "pitch". Apresentaremos três tipos de pulsos glotais, que utilizamos no sistema implementado.

O primeiro pulso glotal é o mais simples de todos, sendo composto simplesmente de um trem de impulsos espaçados pelo período de "pitch", conforme mostra a figura 3.17.

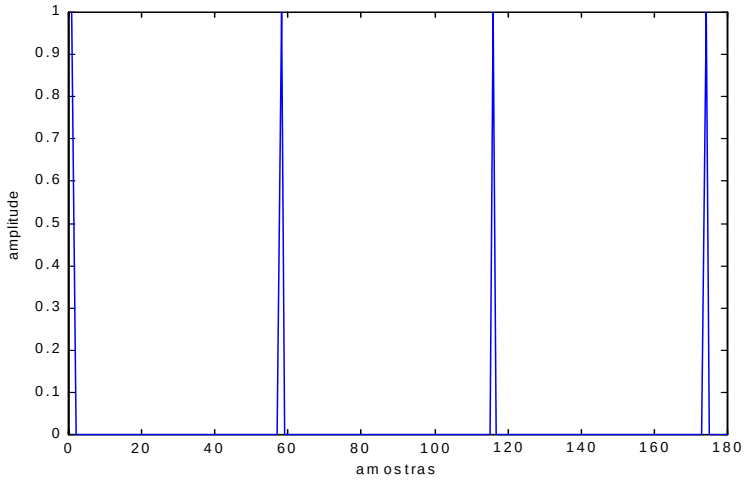

Figura 3.17: Forma do primeiro tipo de pulso glotal.

Nesse caso, o “pitch” é de 58 amostras a 8kHz, que equivale a 7,25ms.

Já o segundo tipo de pulso glotal, é definido pela equação [3]:

$$g(n) = \begin{cases} \frac{1}{2}[1-\cos(\pi \frac{n}{N_1})] & 0<n\leq N_1 \\ \cos(\pi (n-N_1)/2N_2) & N_1<n\leq N_1+N_2 \\ 0 & \text{outros valores de } n \end{cases} \quad (3.24)$$

Podemos ver na figura 3.18 uma seqüência deste tipo de pulso glotal para um valor de “pitch” de 58 amostras utilizando-se N_1 e N_2 iguais a 5.

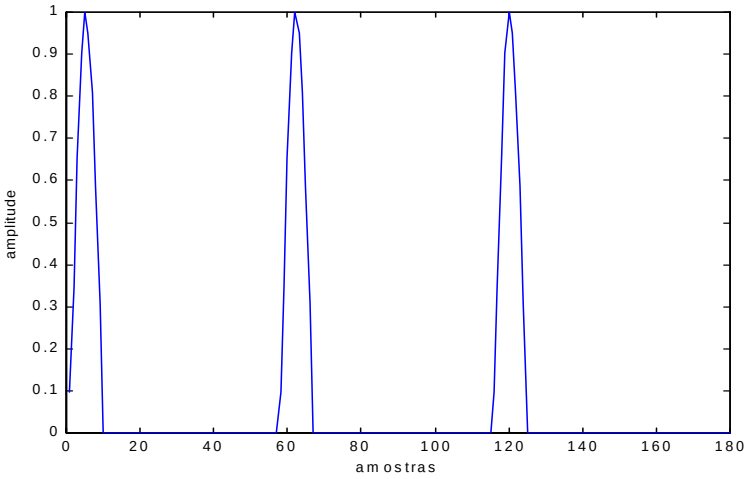

Figura 3.18: Seqüência do segundo tipo de pulso glotal.

Uma terceira forma de pulso glotal é obtida através da passagem de um trem de impulsos por um filtro *G(z)* dado pela seguinte equação [3]:

$$G(z) = \frac{-ae\ln(a)z^{-1}}{(1-az^{-1})^2} \tag{3.25}$$

onde *a* é uma constante a ser definida.

Na figura 3.19, podemos ver o terceiro tipo de pulso glotal para *a* = 0,4 e um "pitch" de valor *n*=58 amostras como nos exemplos anteriores.

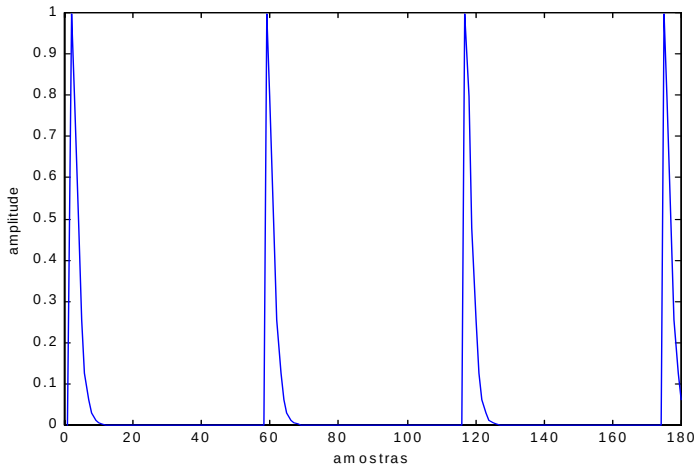

Figura 3.19: Forma do terceiro tipo de pulso glotal.

No próximo capítulo, será apresentada uma análise sobre a utilização desses diferentes tipos de pulsos glotais no codificador LPC.

3.3.8 Síntese LPC

A síntese LPC é feita com a utilização dos parâmetros calculados anteriormente. Cada um dos segmentos é sintetizado através da passagem do sinal de excitação através de um filtro "all-pole" com os coeficientes calculados na análise LPC. O sinal de saída será dado pela convolução entre o sinal de excitação e a resposta ao impulso do filtro de predição linear. O sinal de excitação poderá ser um dos pulsos glotais mostrados no caso de sinal do tipo vozeado ou ruído branco no caso de sinal do tipo não vozeado. A figura 3.20 ilustra o processo de síntese do sinal de fala. Podemos observar que a resposta do filtro que representa o trato vocal é bastante diferente para sinais do tipo vozeados e não vozeados apesar de utilizarmos o mesmo modelo para ambos os casos.

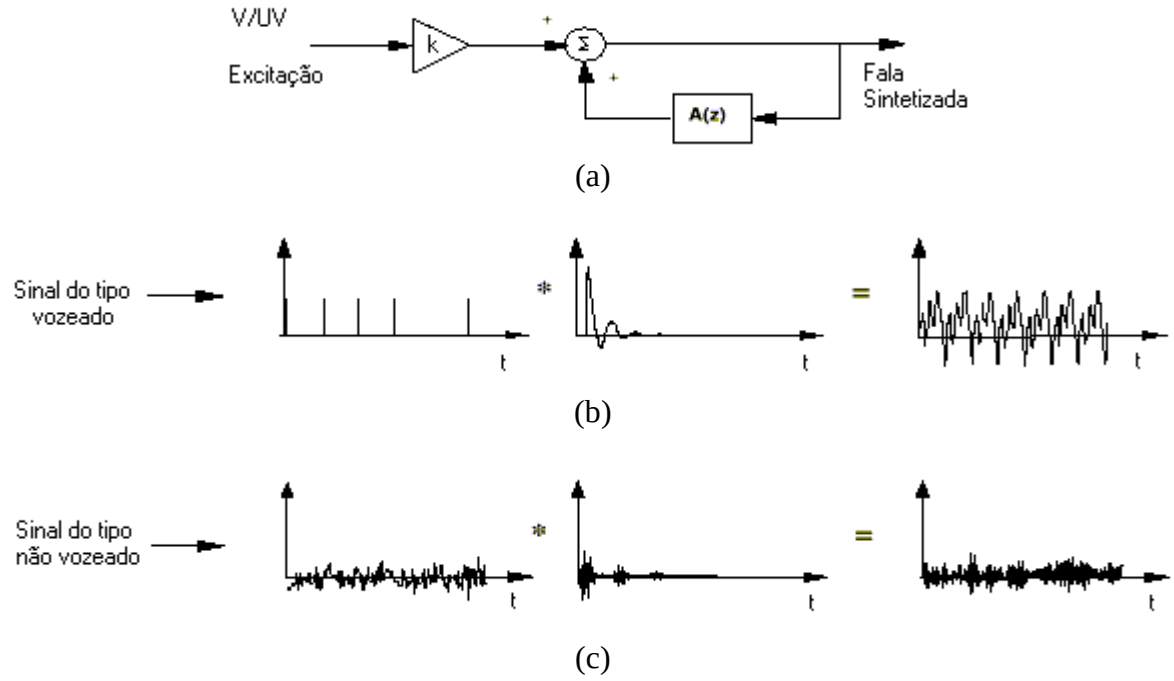

Figura 3.20: (a)Modelo do sistema de síntese, (b) Produção de voz do tipo vozeada, (c) Produção de voz do tipo não vozeada.

A rotina implementada, funciona da seguinte maneira. Quando fazemos a análise LPC, armazenamos os coeficientes de cada segmento em uma linha de uma matriz. Portanto esta matriz contém *p* colunas, onde *p* é a ordem do sistema e *nseg* linhas, onde *nseg* é o número de segmentos do sinal original. Além disso há uma matriz com o valor do ganho e uma outra com o valor do "pitch". Se o segmento possuir sinal do tipo não vozeado, o valor do "pitch" é igual a zero. Essas matrizes são passadas à rotina responsável pela síntese, que será performada através da passagem do sinal de excitação por um filtro do tipo "all-pole" com os coeficientes a_i.

3.3.9 De-Ênfase

A de-ênfase é a passagem do sinal sintetizado através de um filtro do tipo passa-baixa. Esse filtro serve apenas para cancelar os efeitos do filtro de pré-ênfase. O filtro de de-ênfase é dado pela seguinte função de transferência:

$$H(z) = \frac{1}{1 - \eta z^{-1}} \tag{3.26}$$

O parâmetro η não precisa ter o mesmo valor do parâmetro μ da pré-ênfase empregado. Valores de μ diferentes de η podem levar a melhores resultados. Normalmente utiliza-se um valor de η na faixa de 0,74 a 0,94 [4]. Utilizamos em nosso sistema um valor de 0,75.

Podemos ver na figura 3.21 a resposta na freqüência da magnitude do filtro de de-ênfase utilizando um valor de η de 0,75.

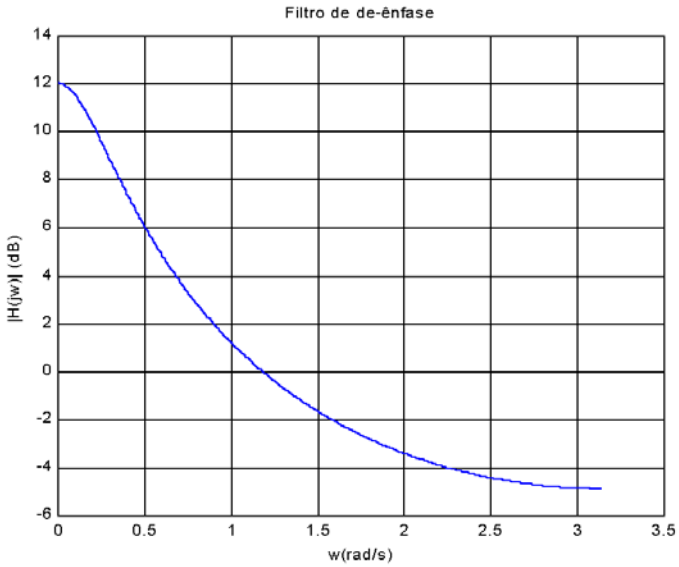

Figura 3.21. Resposta em magnitude do filtro de de-ênfase com μ=0,75 [9].

4. ANÁLISES DA CODIFICAÇÃO LPC

Como vimos anteriormente, um codificador LPC é composto de diversos blocos. A maneira de implementarmos estes blocos irá influenciar significativamente na qualidade do sistema. Assim, neste capítulo apresentamos um estudo de diversos aspectos que envolvem a implementação de um codificador LPC. Serão apresentados fundamentos teóricos, bem como simulações realizadas no sistema implementado para chegarmos a algumas conclusões.

4.1 Estudo do Tamanho do Segmento

Para utilizarmos o método de predição linear na análise e síntese de voz, devemos segmentar o sinal original em pequenas janelas, que devem ser processadas uma a uma. Isso porque os parâmetros calculados por predição linear representarão todo o segmento, e se o mesmo tiver uma duração grande, as características estatísticas do sinal dentro da própria janela irão variar, fazendo com que o sinal sintetizado fique bastante diferente do sinal original. Logo, esses segmentos devem ser aproximadamente estatisticamente estacionários para que o sistema funcione de maneira eficaz. Daí a necessidade de estudarmos a estacionaridade de um segmento de voz.

Os parâmetros calculados na análise LPC são os *p* coeficientes (onde *p* é a ordem do sistema), o ganho e o "pitch" (se o sinal for do tipo vozeado) de cada segmento. Esses parâmetros irão representar o segmento como um todo. Assim, se o segmento não for estacionário, o sinal sintetizado não terá sua forma de onda semelhante a do sinal original. Como exemplo podemos citar o caso em que o segmento for muito grande e houver uma variação do "pitch" do sinal. Nesse caso, o sistema LPC calculará um único valor de "pitch" que será usado para a síntese, não levando em consideração que o sinal teve suas características modificadas no decorrer do segmento.

Para evitarmos esses problemas, devemos estimar a faixa de tamanho de segmento, na qual podemos considerar o sinal como sendo estacionário. Devemos também considerar, que mesmo trabalhando com janelas adequadas, alguns segmentos podem, ainda assim, não serem estacionários em virtude de uma variação brusca nas características do sinal, entretanto na maioria das vezes poderemos considerar o segmento como sendo estacionário.

A questão da estacionaridade limita superiormente o tamanho do segmento, mas há também dois limites inferiores, que são dados pela resolução de freqüência do sinal e pela complexidade computacional resultante. A resolução de freqüência, pelo fato de que no caso do sinal ser do tipo vozeado, não podemos ter um segmento menor do que o período do "pitch", já que neste caso, não poderíamos determinar o seu valor de forma apropriada. A complexidade computacional decorre do fato de que quanto menor o segmento, maior a complexidade computacional, já que teremos que processar um número maior de segmentos.

Para verificarmos a estacionaridade de um segmento, utilizamos a função covariância, dada por:

$$\varphi(a,b;m) = \frac{1}{N} \sum_{n=m-N+1}^{m} s(n-a)s(n-b) \qquad (4.1)$$

Poderíamos ter utilizado outras funções, como a de autocorrelação por exemplo, mas optamos por basear as análises nessa função.

Assim, ao plotarmos a função acima, para diversos tamanhos de janela *N*, devemos verificar se o seu valor se mantém num valor próximo ao longo dos deslocamentos *a* e *b*. Se isto ocorrer, então entendemos que o segmento é aproximadamente estacionário, caso contrário não. Isto porque, se $\varphi(a,b;m)$ se mantiver aproximadamente constante ao longo dos deslocamentos de *a* e *b* significa que o sinal praticamente não variou estatisticamente, ou seja, podemos considerar este segmento como sendo aproximadamente estacionário.

Podemos fazer essa análise de maneira qualitativa e de maneira quantitativa. A maneira qualitativa, seria a análise visual dos gráficos obtidos da função covariância para diversos tamanhos de janela. A análise quantitativa será vista adiante.

A seguir podemos ver o gráfico de 3 segmentos distintos, calculados para diferentes valores de *N*. Estes segmentos foram extraídos de um arquivo no formato wav (count.wav), com freqüência de amostragem de 8kHz e amostragem de 16 bits. Os três segmentos distintos que serão analisados, tem o seu final (*m* na função covariância) nos pontos 26800, 16200 e 12150 respectivamente.

Começamos nossa análise para um segmento de 12,5 ms, já que o tamanho do segmento limita superiormente o valor máximo de “pitch” que pode ser calculado. Assim, consideramos que se utilizássemos segmentos menores do que 12,5 ms correríamos o risco de não conseguir calcular o valor do pitch. Como essa análise servirá de base para o tamanho do segmento utilizado no sistema LPC, não utilizamos valores de *N* menores do que 101.

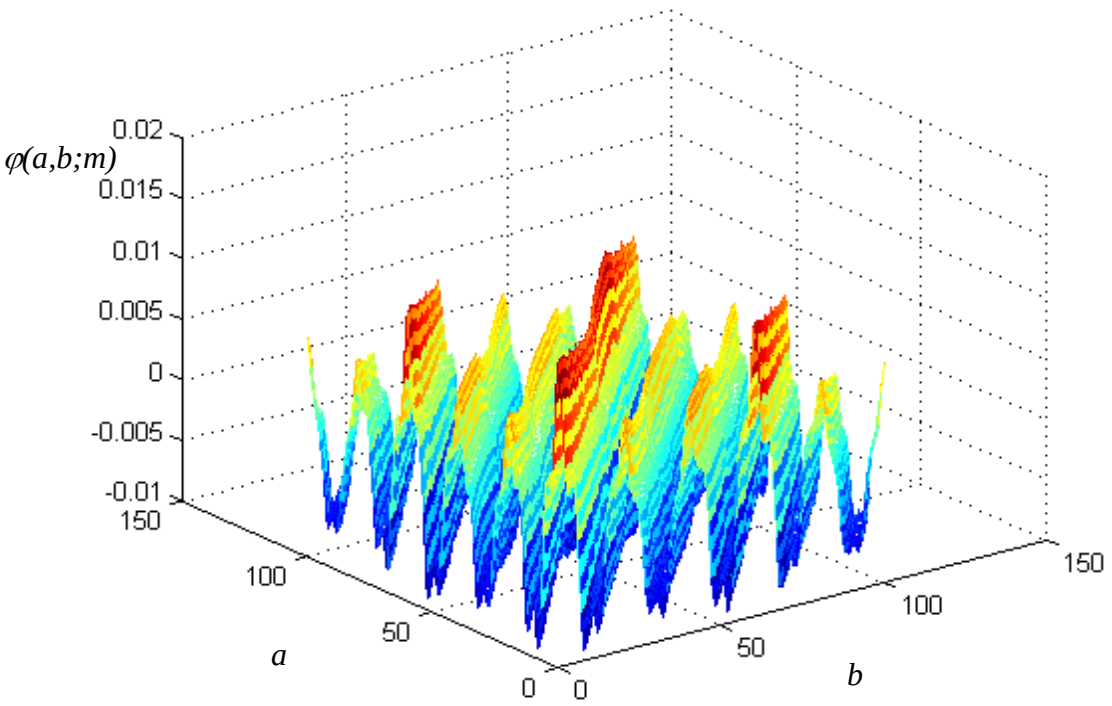

Figura 4.1: Função covariância para *N*=101 (12,5ms) – sinal 1.

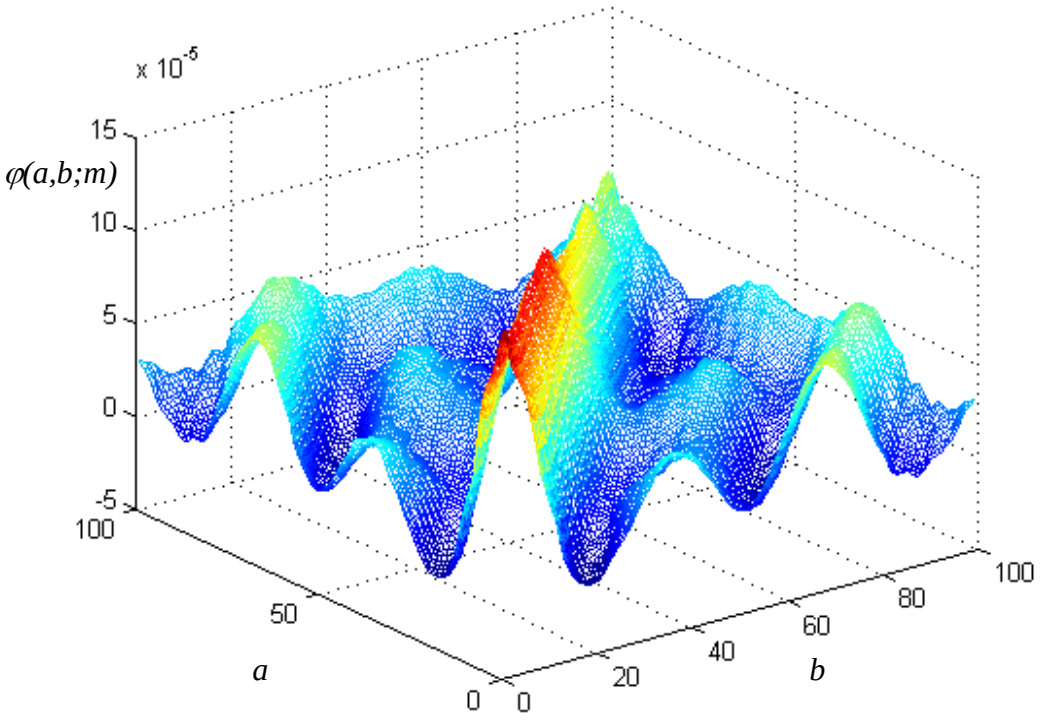

Figura 4.2: Função covariância para *N*=101 (12,5ms) – sinal 2.

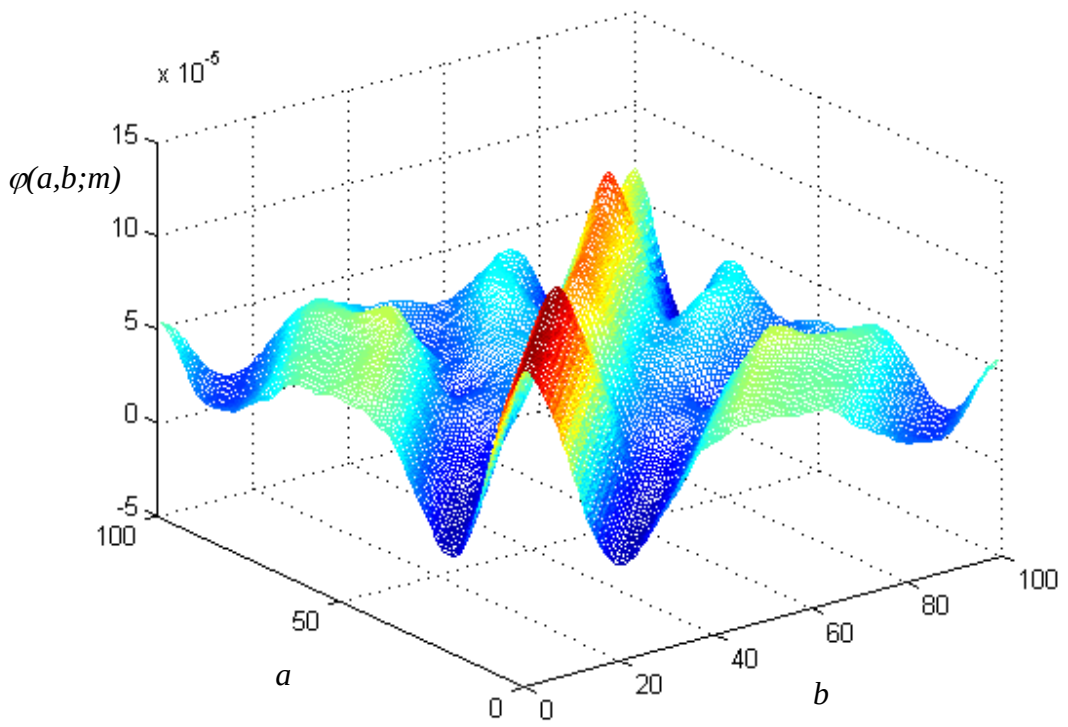

Figura 4.3: Função covariância para *N*=101 (12,5ms) – sinal 3.

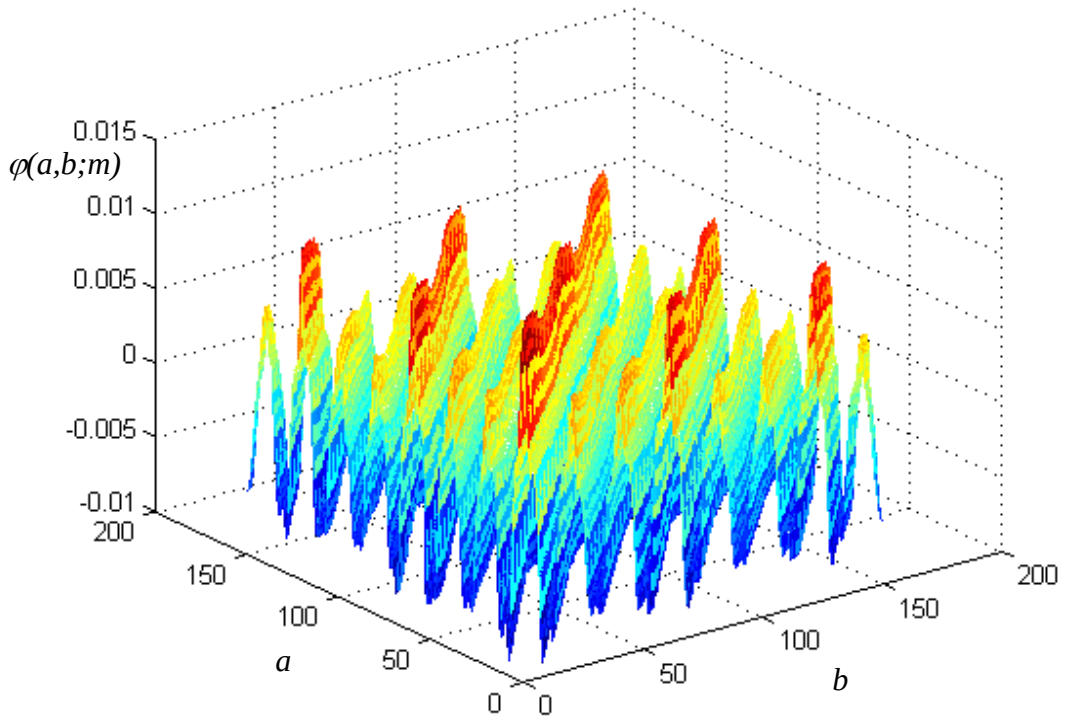

Figura 4.4: Função covariância para *N*=151 (18,75ms) – sinal 1.

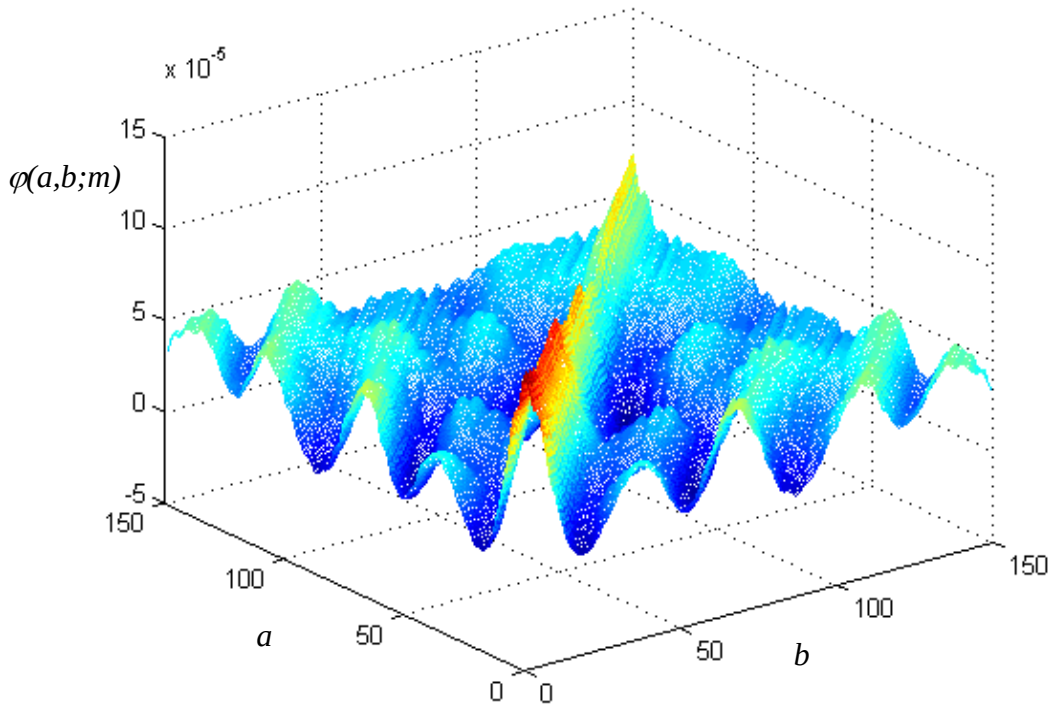

Figura 4.5: Função covariância para *N*=151 (18,75ms) – sinal 2.

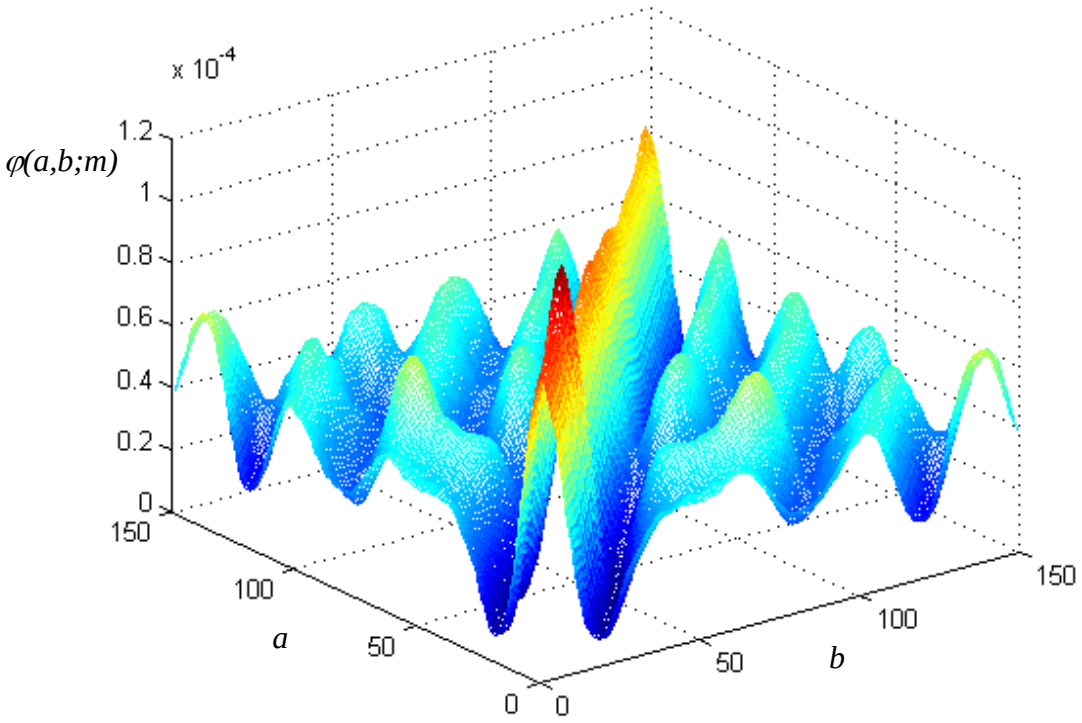

Figura 4.6: Função covariância para *N*=151 (18,75ms) – sinal 3.

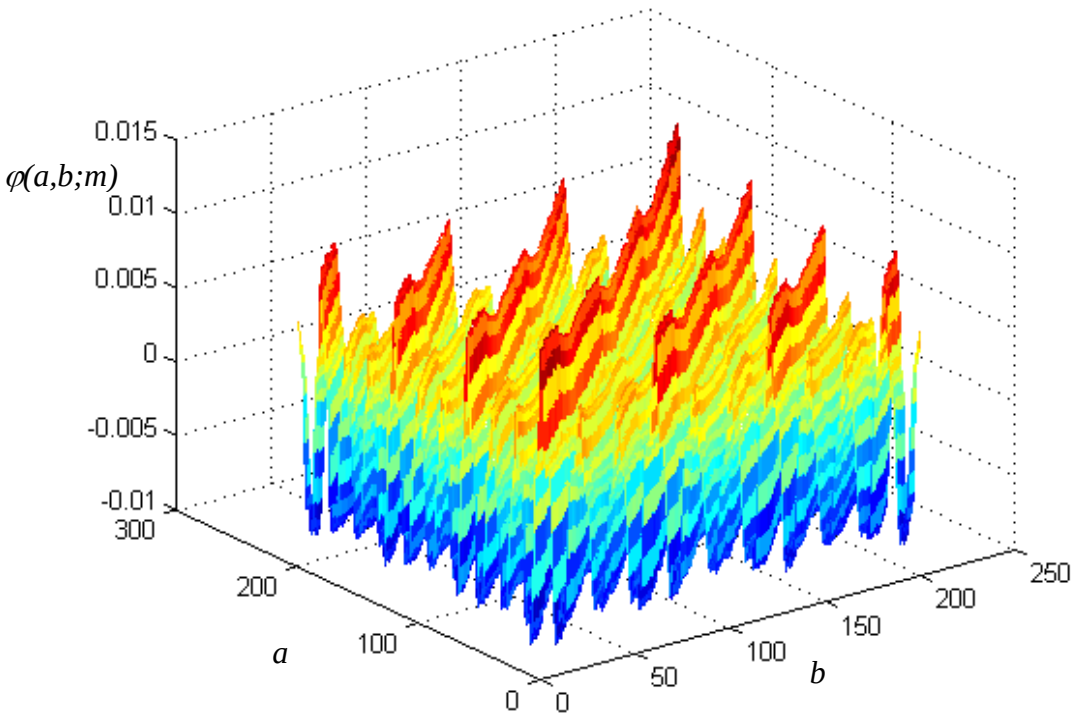

Figura 4.7: Função covariância para *N*=201 (25ms) – sinal 1.

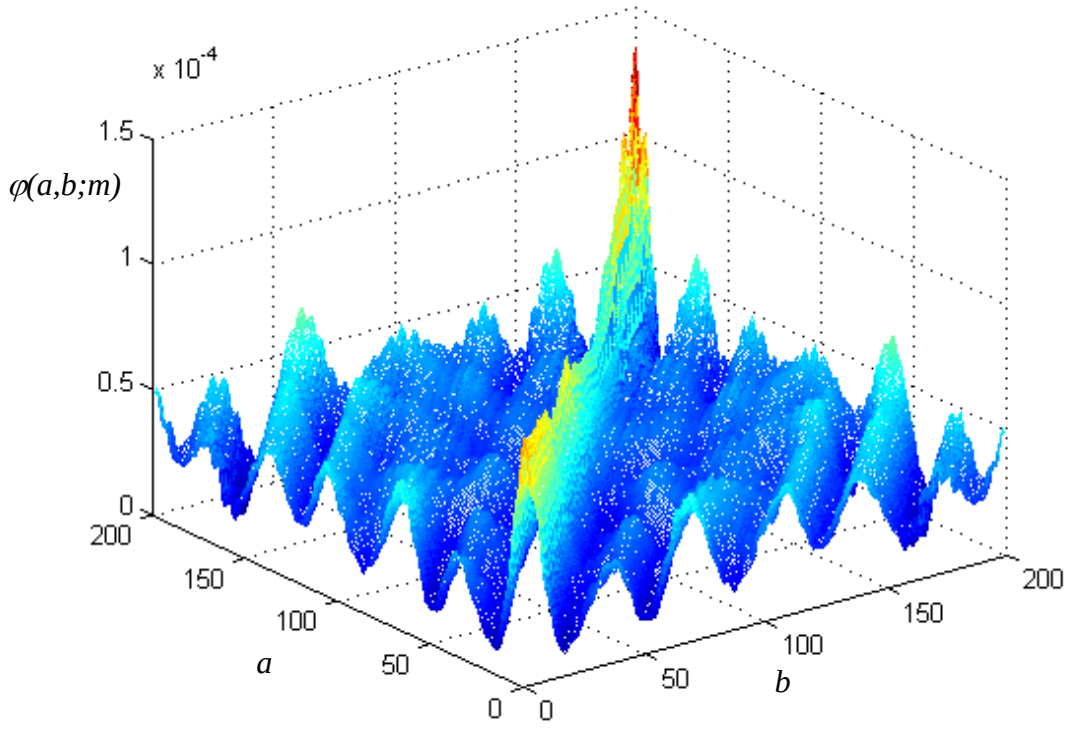

Figura 4.8: Função covariância para *N*=201 (25ms) – sinal 2.

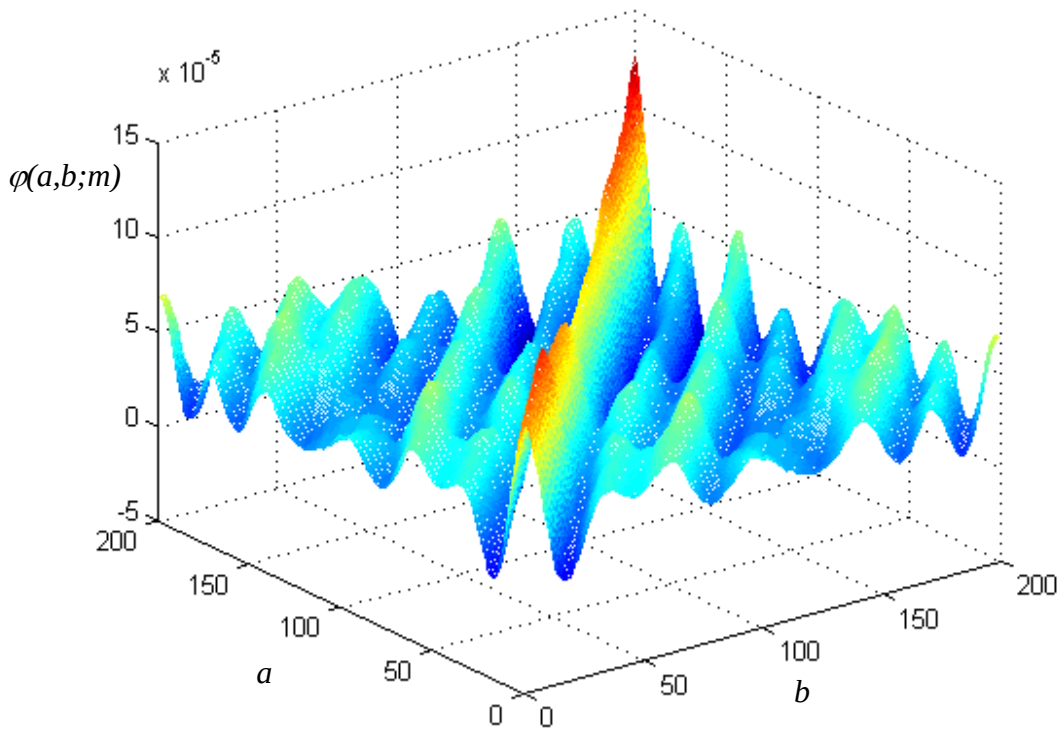

Figura 4.9: Função covariância para *N*=201 (25ms) – sinal 3.

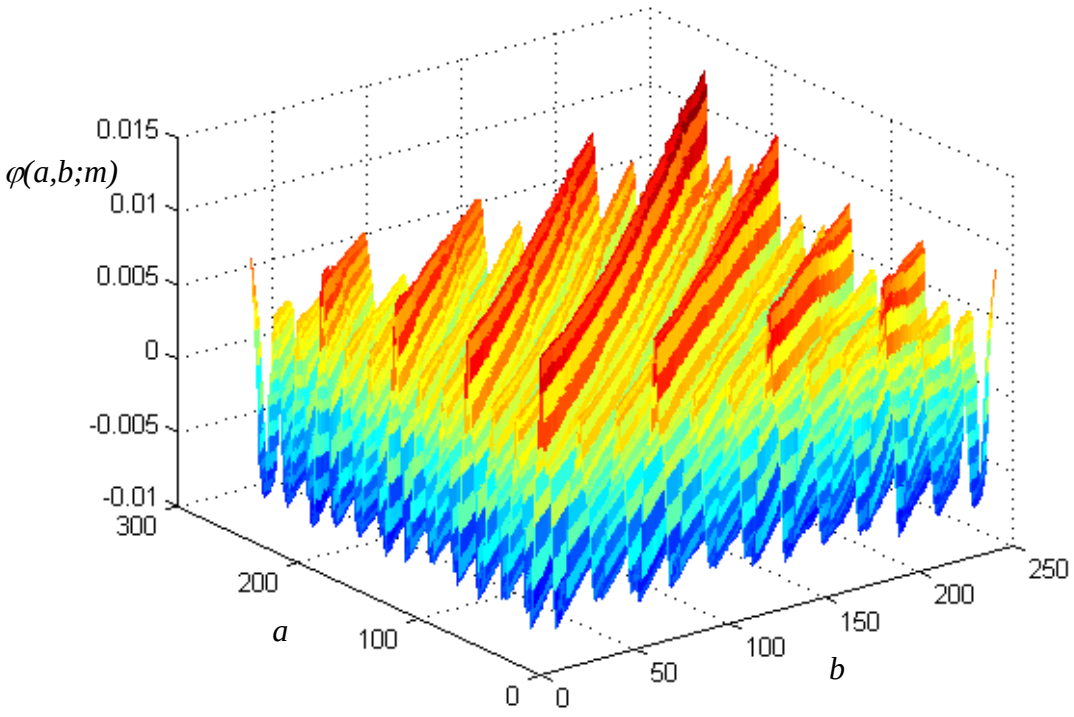

Figura 4.10: Função covariância para *N*=241 (30ms) – sinal 1.

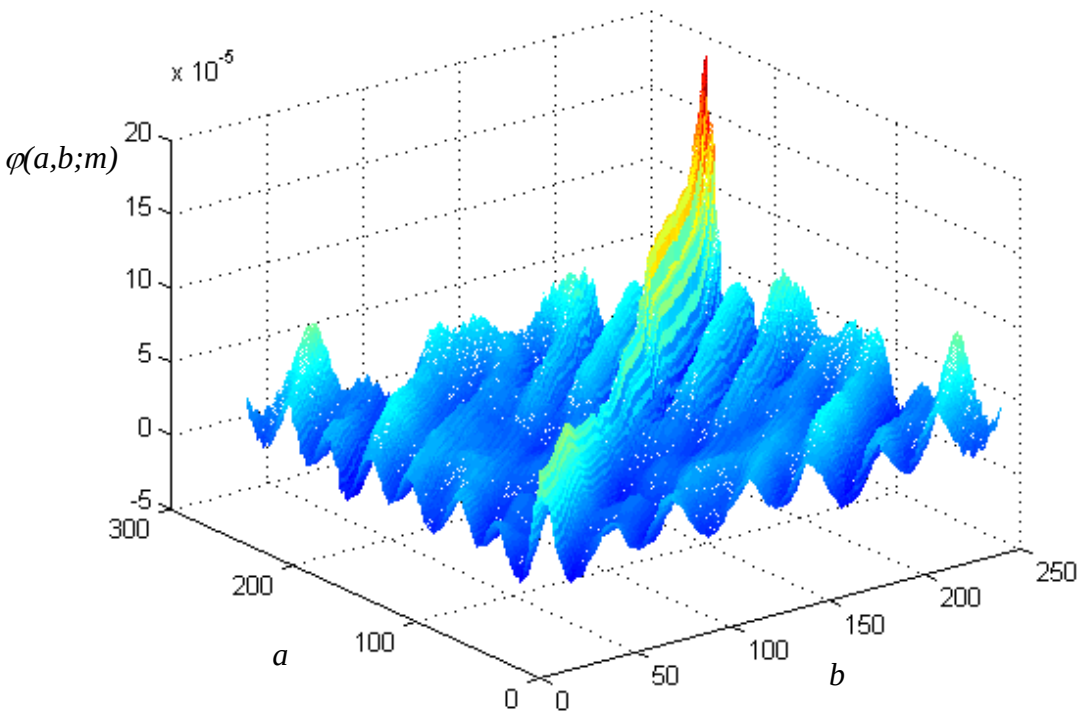

Figura 4.11: Função covariância para *N*=241 (30ms) – sinal 2.

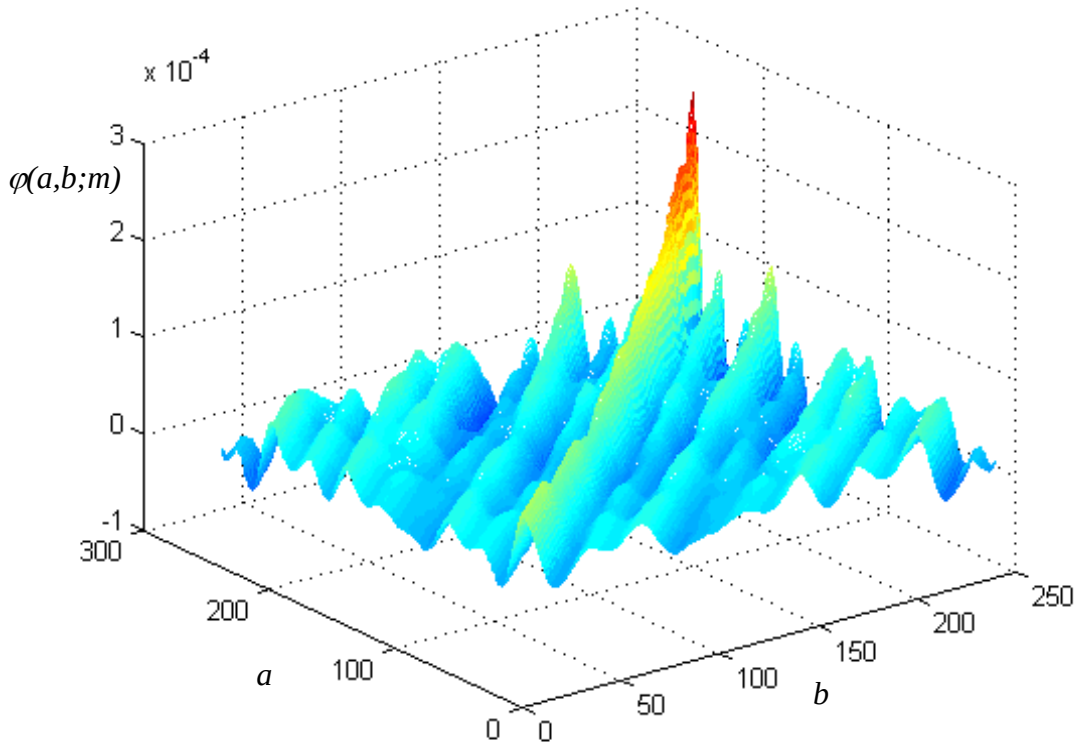

Figura 4.12: Função covariância para *N*=241 (30ms) – sinal 3.

Podemos observar nas figuras acima, com exceção do sinal 1, que quando utilizamos janelas de tamanho N=200, o sinal começa a se comportar de maneira não estacionária. Isso pode ser percebido verificando-se o valor da função covariância ao longo dos deslocamentos de a e b que passa a variar acentuadamente. Quando N=240 (30 ms), percebemos de maneira mais evidente que as propriedades de estacionaridade deixam de existir. Já com uma janela de tamanho N até 150, verificamos que podemos considerar o sinal como sendo estacionário. Já o sinal 1, aparentemente se mantém aproximadamente estacionário para todos os valores de N mostrados.

O problema da análise qualitativa, é que ela não quantiza nenhum valor sobre o nível de estacionaridade do segmento. Para isso, foi implementada uma análise quantitativa. Essa análise é feita da seguinte maneira:

- Ao calcularmos a função covariância dada por (4.1), obtemos uma matriz de covariância $\mathbf{\Phi}(a,b)$, onde a e b representam o deslocamento sofrido pelo sinal. Através dessas matrizes são gerados os gráficos exibidos anteriormente. Para avaliarmos o grau de estacionaridade do segmento, devemos calcular a variação exibida pelos sucessivos cortes desses gráficos. Cada um desses cortes é dado pelas diagonais da matriz. Como devemos comparar as diagonais da matriz e estas possuem tamanhos diferentes uma das outras, então extraímos parte das diagonais de mesmo tamanho e montamos duas sub-matrizes, **X** e **Y** conforme ilustrado nas figuras 4.13 e 4.14.

$$\mathbf{\Phi}(a,b) = \begin{vmatrix} \varphi(1,1) & \varphi(1,2) & \varphi(1,3) & \varphi(1,4) & \varphi(1,5) & \varphi(1,6) & \varphi(1,7) \\ \varphi(2,1) & \varphi(2,2) & \varphi(2,3) & \varphi(2,4) & \varphi(2,5) & \varphi(2,6) & \varphi(2,7) \\ \varphi(3,1) & \varphi(3,2) & \varphi(3,3) & \varphi(3,4) & \varphi(3,5) & \varphi(3,6) & \varphi(3,7) \\ \varphi(4,1) & \varphi(4,2) & \varphi(4,3) & \varphi(4,4) & \varphi(4,5) & \varphi(4,6) & \varphi(4,7) \\ \varphi(5,1) & \varphi(5,2) & \varphi(5,3) & \varphi(5,4) & \varphi(5,5) & \varphi(5,6) & \varphi(5,7) \\ \varphi(6,1) & \varphi(6,2) & \varphi(6,3) & \varphi(6,4) & \varphi(6,5) & \varphi(6,6) & \varphi(6,7) \\ \varphi(7,1) & \varphi(7,2) & \varphi(7,3) & \varphi(7,4) & \varphi(7,5) & \varphi(7,6) & \varphi(7,7) \end{vmatrix}$$

Figura 4.13: Matriz da função covariância

$$\mathbf{X} = \begin{vmatrix} \varphi(4,1) & \varphi(3,2) & \varphi(2,3) & \varphi(1,4) \\ \varphi(5,2) & \varphi(4,3) & \varphi(3,4) & \varphi(2,5) \\ \varphi(6,3) & \varphi(5,4) & \varphi(4,5) & \varphi(3,6) \\ \varphi(7,4) & \varphi(6,5) & \varphi(5,6) & \varphi(4,7) \end{vmatrix} \quad \text{e} \quad \mathbf{Y} = \begin{vmatrix} \varphi(4,2) & \varphi(3,3) & \varphi(2,4) \\ \varphi(5,3) & \varphi(4,4) & \varphi(3,5) \\ \varphi(6,4) & \varphi(5,5) & \varphi(4,6) \end{vmatrix}$$

Figura 4.14: Sub-matrizes de $\mathbf{\Phi}(a,b)$

- Calculamos a média dos elementos de cada uma das colunas das sub-matrizes, dada por:

$$média(j) = \frac{1}{I}\sum_{i=1}^{I} \mathbf{X}(i, j)\,, \tag{4.2}$$

onde *I* é o número total de linhas, *i* é o índice da linha, *j* é o índice de coluna e **X** é uma das sub-matrizes.

- Calcula-se então o erro relativo de cada ponto das sub-matrizes em relação à média de sua coluna. O erro relativo de cada ponto é dado por:

$$erro_{\mathbf{X}}(i, j) = \frac{\mathbf{X}(i, j) - média(j)}{média(j)} \tag{4.3}$$

- Calcula-se a média do erro relativo de todos os elementos da sub-matriz, dada por:

$$erro\_rel\_médio_{\mathbf{X}} = \frac{1}{IJ}\sum_{i=1}^{I}\left[\sum_{j=1}^{J} erro_{\mathbf{X}}(i, j)\right], \tag{4.4}$$

onde *J* é o número total de colunas da sub-matriz.

- Teremos então um erro relativo médio para cada sub-matriz. Então calculamos a média dos dois e obtemos o erro relativo médio total, dado por:

$$erro\_rel\_médio = \frac{erro\_rel\_médio_{\mathbf{x}} + erro\_rel\_médio_{\mathbf{y}}}{2}\,, \tag{4.5}$$

Esta análise nos dará um valor que representará a variação da função covariância em função do tamanho do segmento *N*. Se este valor for aumentando, é sinal que o segmento está se tornando cada vez menos estacionário. Os gráficos e a tabela apresentados a seguir mostram a variação deste erro relativo médio em função do tamanho de segmento *N* para cada um dos 3 sinais analisados anteriormente.

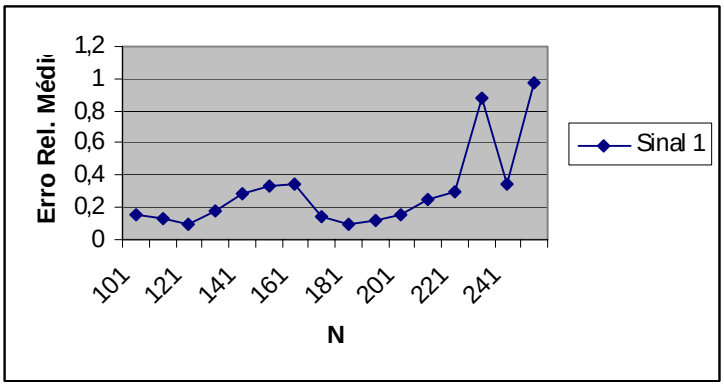

Figura 4.15: Erro relativo médio x *N* - sinal 1.

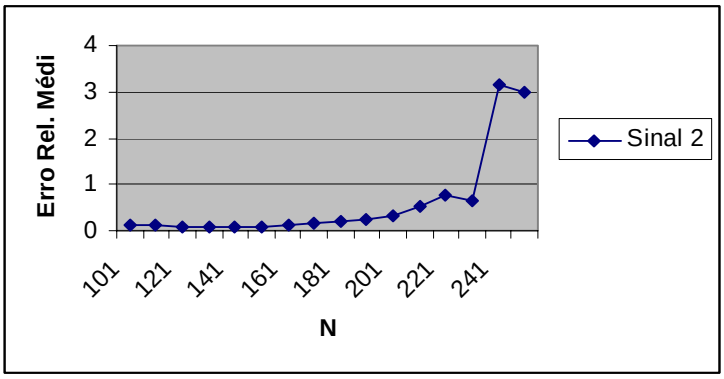

Figura 4.16: Erro relativo médio x *N* - sinal 2.

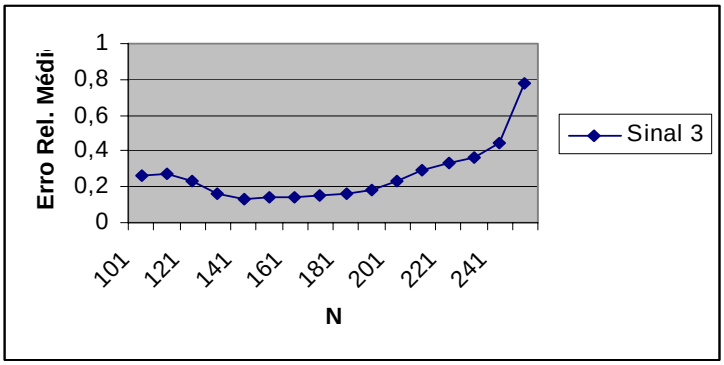

Figura 4.17: Erro relativo médio x *N* - sinal 3.

Tabela 4.1: Erro relativo médio x N - sinais 1, 2 e 3.

N	Erro Relativo		
	Sinal 1	Sinal 2	Sinal 3
101	0,15	0,14	0,26
111	0,13	0,12	0,27
121	0,09	0,08	0,23
131	0,18	0,08	0,16
141	0,28	0,07	0,13
151	0,33	0,09	0,14
161	0,34	0,13	0,14
171	0,14	0,17	0,15
181	0,09	0,22	0,16
191	0,12	0,26	0,18
201	0,16	0,32	0,23
211	0,25	0,51	0,29
221	0,3	0,78	0,33
231	0,88	0,63	0,36
241	0,34	3,17	0,44
251	0,98	2,99	0,78

Podemos ver através dos dados obtidos, que o erro relativo médio aumenta quando aumentamos o valor de *N*, indicando que o sinal vai se tornando menos estacionário com o aumento de *N*. Observamos que para *N* variando de 101 a 221, temos um valor de erro aproximadamente igual. Observamos também, que mesmo o sinal 1, que aparentemente se mantinha aproximadamente estacionário, teve o seu erro relativo médio aumentado com o aumento do valor de *N*. Isso mostra a importância de uma análise quantitativa, e não apenas qualitativa.

Consideramos que para o sistema LPC implementado, uma faixa adequada para o tamanho de segmento seria entre 100 (12,5 ms) e 220 (27,5 ms). Para confirmarmos isso, apresentamos a seguir duas tabelas com a comparação dos resultados obtidos no sistema LPC implementado para os diferentes tamanhos de janela. Para todos os tamanhos de janela, utilizou-se uma taxa de superposição de 50%.

Para que tenhamos uma idéia quantitativa, a respeito da qualidade do som sintetizado com e sem superposição, utilizaremos o erro médio quadrático, chamado de MSE (“mean square error”), que está sendo normalizado pela energia do sinal. O MSE é dado por:

$$\text{MSE} = \frac{1}{s^2(n)} \sum_{n=1}^{N} (y(n) - s(n))^2 \qquad (4.6)$$

onde *s(n)* é o sinal original, *y(n)* é o sinal sintetizado e *N* é o número de amostras existentes.

Uma outra medida importante para efeito de comparação é a da complexidade computacional do sistema com e sem superposição, que é possível ser estimada através do comando “flops”, do Matlab, que traz o número de operações com pontos flutuantes realizadas. Também deve ser levado em conta, a percepção que temos do som sintetizado, que é uma medida bastante subjetiva, mas importante. Para isso, anexaremos à documentação os arquivos gerados na síntese do sinal. Nas tabelas abaixo, classificaremos a qualidade do sinal sintetizado com sinais de mais e de menos, onde o número de sinais de mais é diretamente proporcional à qualidade do sinal sintetizado. Esta análise subjetiva não possui significado estatístico já que foi feita apenas por uma pessoa. Abaixo, apresentamos os resultados obtidos:

Tabela 4.2: Arquivo “aeiou.wav”.

N	Complex. Comp.	MSE	Qualidade
100	$115{,}82 \times 10^6$	1,48	+++
120	$107{,}40 \times 10^6$	1,40	+++
150	$97{,}06 \times 10^6$	1,42	++
180	$81{,}89 \times 10^6$	1,49	+
210	$70{,}95 \times 10^6$	1,42	-
240	$62{,}70 \times 10^6$	1,48	--

Tabela 4.3: Arquivo “teste.wav”.

N	Complex. Comp.	MSE	Qualidade
100	$38{,}9 \times 10^6$	1,24	+++
120	$35{,}20 \times 10^6$	1,37	+++
150	$31{,}33 \times 10^6$	1,36	++
180	$26{,}39 \times 10^6$	1,49	+
210	$22{,}95 \times 10^6$	1,65	-
240	$20{,}44 \times 10^6$	1,45	--

Podemos notar que o valor do MSE permanece praticamente igual para todos os valores de *N*, apesar de uma notável diferença na qualidade da síntese do sinal de fala. Isso nos leva a crer que muitas vezes o MSE não é um bom parâmetro para avaliarmos sinais de fala. Quanto a complexidade computacional, verifica-se um aumento ao diminuirmos o valor de *N*, como já era de se esperar. A qualidade é um fator bastante subjetivo, mas é bastante clara a piora da qualidade do som ao aumentarmos o valor de *N* para 210 ou para 240. O que ocorre para esses valores de *N*, é que o sinal fica cada vez mais robotizado. Os arquivos gerados se encontram em anexo denominados da seguinte maneira: “aeiou_XXX.wav” e “teste_XXX.wav”, onde “XXX” representa o valor de *N* utilizado.

Foram realizados três tipos de análise, dois deles teóricos, sendo que um qualitativo e outro quantitativo e outro experimental. A conclusão que chegou-se, é que para os tamanhos de segmentos testados, um tamanho de segmento na faixa de 100 (12,75ms) a 180 (22,5ms) amostras, para uma freqüência de amostragem de 8kHz apresenta melhores resultados. Isso porque para esses valores de *N*, obtemos uma boa qualidade na síntese de voz, como observa-se experimentalmente, às custas de uma complexidade computacional

média, e tendo uma boa resolução de freqüência, já que o período de “pitch” não excede o tamanho dessa janela. Essa faixa, está dentro do que esperávamos para que o segmento fosse considerado estacionário.

4.2 Estudo da Utilização da Superposição

Como pudemos ver, o sistema LPC, requer a divisão do sinal de fala, representado por um vetor, em janelas de curta duração de tempo (em torno de 10 a 30ms) que possam ser consideradas estacionárias. A análise dos parâmetros LPC, é então feita sobre cada janela resultante dessa segmentação. De modo análogo, o sinal de fala é reconstruído através da aglutinação de várias janelas sintetizadas. No entanto, se fizermos pura e simplesmente a junção de uma janela após a outra, sem utilizarmos nenhum elemento comum entre duas janelas consecutivas o sinal apresentará uma descontinuidade na transição entre as duas janelas.

A alternativa existente para tentarmos suavizar essa transição, é a utilização da superposição, ou “overlapping”. Nesse caso, duas janelas consecutivas possuem trechos comuns, onde será feita a análise dos parâmetros LPC. Da mesma forma, a síntese de duas janelas consecutivas também possuirá elementos comuns. Isso faz com que a análise do segmento seguinte seja semelhante a do segmento anterior, suavizando as transições geradas na síntese do sinal de fala.

Para fazermos a concatenação entre janelas consecutivas no momento da síntese, somamos os pontos comuns às duas janelas. Neste caso, se utilizarmos uma superposição de 50% do tamanho da janela, apenas a metade da primeira janela e metade da última, não seriam somadas a outra janela, o que representa um erro insignificante no sinal gerado.

O fator de superposição pode variar, podendo assumir valores como 50% (nesse caso, L=x e I=x/2), 66,7% (L=x e I=x/3), ou qualquer outro valor em que I seja um divisor de L. Um fator de 50% por exemplo, significa que se estivermos usando um tamanho de janela igual a L, então, a próxima janela se iniciará L/2 depois do início da janela atual.

A seguir, podemos comparar alguns valores obtidos no sistema LPC, para diferentes taxas de superposição. Para que tenhamos uma idéia quantitativa, a respeito da qualidade do som sintetizado com e sem superposição, utilizaremos o MSE. Avaliaremos também a complexidade computacional do sistema com e sem superposição, e a percepção que temos do som sintetizado, que é uma medida subjetiva, mas bastante importante.

Nos testes realizados, utilizamos segmentos de 20ms, ou seja, L=160, já que a freqüência de amostragem do nosso sinal é de 8 kHz. Apenas nos testes com taxa de superposição de 66,7%, tivemos que aumentar o valor de L para 162, já que neste caso L deve ser um valor múltiplo de 3. Podemos ver os resultados nas tabelas a seguir:

Tabela 4.4: Arquivo "aeiou.wav".

Taxa de Superposição	MSE	Complex. Comp.	Qualidade
0	2,08	$46{,}19 \times 10^6$	+
50%	1,55	$91{,}46 \times 10^6$	++
66,7%	1,31	$135{,}13 \times 10^6$	+++

Tabela 4.5: Arquivo "teste.wav".

Taxa de Superposição	MSE	Complex. Comp.	Qualidade
0	2,00	$14{,}96 \times 10^6$	+
50%	1,69	$29{,}51 \times 10^6$	++
66,7%	1,26	$43{,}49 \times 10^6$	+++

Através dos testes realizados, pudemos verificar que a superposição melhora sensivelmente a qualidade do sinal de voz sintetizado, no entanto, verifica-se um aumento considerável na complexidade computacional do sistema. Os arquivos gerados se encontram em anexo com a seguinte denominação: "aeiou_XX.wav" e "teste_XX.wav", onde "XX" representa a taxa de superposição utilizada. Podemos observar que o valor do erro dado pelo MSE diminuiu consideravelmente com a utilização da superposição.

Outra coisa que pode ser observada no sinal sintetizado é que tivemos uma variação no valor do "pitch" com a utilização da superposição que tornou o sinal mais agudo. Isso é explicado pelo fato de que não há sincronismo de "pitch" na concatenação das janelas, assim algumas vezes há duas excitações sucessivas separadas por um período menor do que o período do "pitch".

A partir desses dados, concluir que para o sistema LPC implementado e dentre as taxas de superposição testadas, se o fator tempo não for crítico, devemos usar uma taxa de superposição de 66,7%, chegando assim a uma boa qualidade de voz sintetizada. No entanto, se a qualidade do sinal não for tão importante, mas sim a inteligibilidade e o tempo de processamento, então devemos baixar a taxa de superposição para 50%, ou até mesmo não utilizarmos a superposição. Isso faz com que se perca qualidade, no entanto o sinal é perfeitamente inteligível, embora fique bastante robotizado.

4.3 Estudo da Utilização de Diferentes Tipos de Janelas

Para segmentarmos o sinal de fala original, devemos multiplicá-lo por uma janela com o tamanho de *N* amostras, como foi mostrado no capítulo anterior. Essa janela vai sendo deslocada de modo a obtermos vários segmentos que compõem o sinal original. Existem vários tipos de janelas que podemos usar para este fim sendo que algumas delas já foram apresentadas.

Aplicamos as janelas do tipo retangular, Hamming, Hanning, Blackman e Bartlett no sistema LPC utilizando como sinal original a vogal "a" amostrada a 8kHz, de modo a

verificarmos as variações na performance do sistema em função do tipo de janelamento utilizado. Utilizamos um tamanho de janela de 160 amostras, que equivalem a 20ms e uma taxa de superposição de 50%. Obtivemos os seguintes resultados:

Tabela 4.6: Arquivo "a_andre".

	Complex. Computacional	MSE	Qualidade
Retangular	$5{,}76 \times 10^6$	4,39	+
Hamming	$5{,}83 \times 10^6$	1,70	++
Hanning	$5{,}83 \times 10^6$	1,74	++
Blackman	$5{,}89 \times 10^6$	1,91	++
Bartlett	$5{,}79 \times 10^6$	1,84	++

Depois, fizemos o mesmo teste para o sinal contido no arquivo "teste.wav". Abaixo os resultados obtidos:

Tabela 4.7: Arquivo "teste.wav".

	Complex. Computacional	MSE	Qualidade
Retangular	$29{,}22 \times 10^6$	6,19	+
Hamming	$29{,}51 \times 10^6$	1,70	++
Hanning	$29{,}51 \times 10^6$	1,72	++
Blackman	$29{,}75 \times 10^6$	1,87	++
Bartlett	$29{,}32 \times 10^6$	1,81	++

Podemos observar através dos resultados obtidos que o valor do MSE se mantém em valores muito próximos para todos os tipos de janelamento com exceção da janela retangular que possui um erro bem mais alto. Em termos de qualidade do sinal sintetizado, é muito difícil percebermos alguma diferença na qualidade do sinal sintetizado. Apenas no janelamento do tipo retangular há uma pequena piora na qualidade do sinal sintetizado.

Já a complexidade computacional, permanece praticamente constante independente do tipo de janela utilizado. Sendo assim, utilizaremos no sistema implementado, a janela de Hamming, por possuir um menor valor de erro nos testes realizados. Podemos concluir que em nosso caso, a escolha do tipo de janelamento não é um fator crítico para o funcionamento do sistema. Apenas não é aconselhável a utilização do janelamento retangular que leva a resultados um pouco inferiores aos outros tipos de janelamentos.

Os arquivos gerados encontram-se em anexo sendo denominados como: "a_*.wav" e "teste_*.wav", onde '*' representa o nome da janela utilizada.

4.4 Estudo de Diferentes Técnicas Para a Solução do Problema LPC: Autocorrelação e Covariância

Conforme foi visto anteriormente, podemos calcular os coeficientes LPC através de duas formas distintas, que são a autocorrelação e a covariância. Nesta seção apresentaremos os resultados obtidos com ambos os métodos a fim de chegarmos a conclusões sobre a utilização de cada um dos métodos. Para isso, levaremos em conta aspectos como a resposta em freqüência dos coeficientes gerados por cada um dos métodos, o MSE entre o sinal gerado por cada um dos métodos e o sinal original, e a complexidade computacional associada a cada um dos métodos.

Na figura 4.18, podemos ver a resposta em magnitude para um filtro com os coeficientes calculados pelo método da autocorrelação e o espectro do segmento do sinal original calculado pela FFT (“fast fourier transform”) com 320 pontos. Utilizamos o sistema LPC com ordem 10. A resposta com o gráfico mais suavizado corresponde à resposta do filtro. Para esta análise, utilizou-se o som da vogal “a” amostrado a 8kHz e um segmento de 160 amostras, que equivale a 20ms.

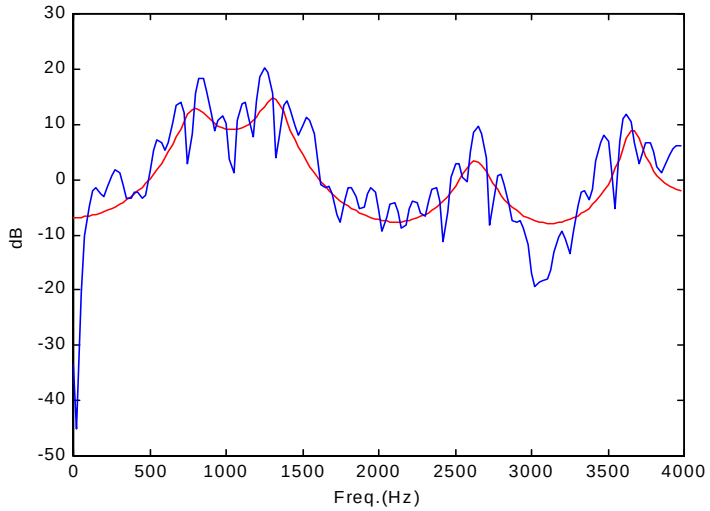

Figura 4.18: Resposta em magnitude utilizando a autocorrelação.

Já na figura 4.19, podemos ver a resposta em magnitude para um filtro com os coeficientes calculados pelo método da covariância, assim como o espectro do segmento do sinal original. Foi utilizado o mesmo sinal original para ambos os métodos.

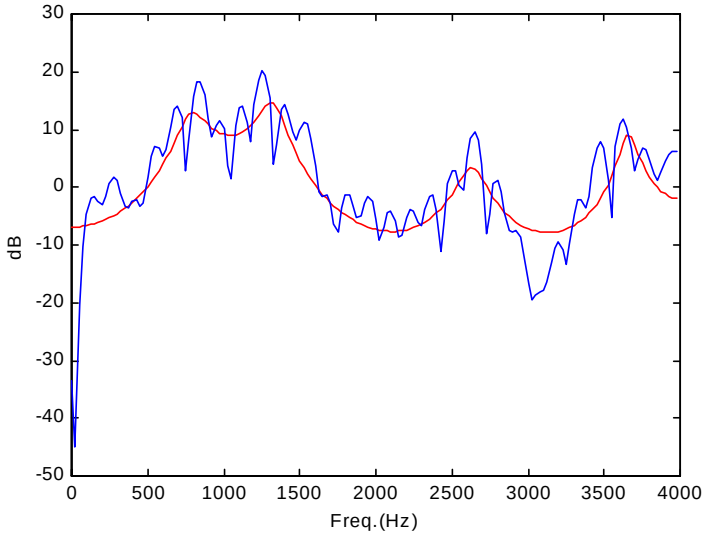

Figura 4.19: Resposta em magnitude utilizando a covariância.

Na figura 4.20, podemos ver a diferença entre a resposta na freqüência obtida pelos dois métodos.

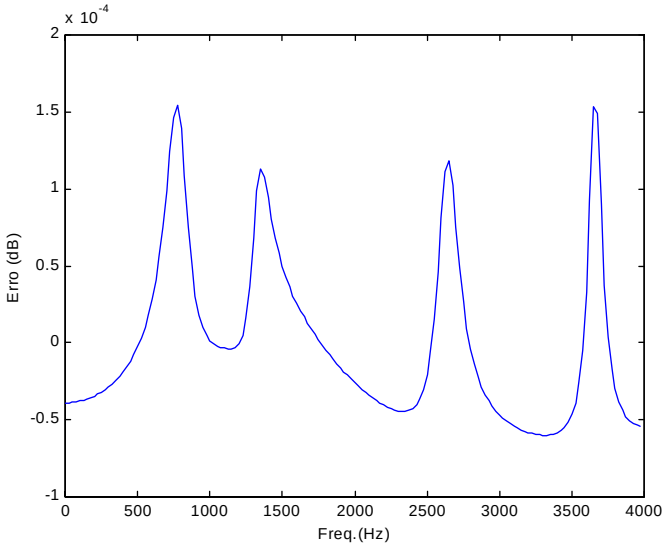

Figura 4.20: Diferença entre a resposta em magnitude obtida através dos dois métodos.

Conforme podemos notar, a diferença entre os resultados obtidos nos dois métodos é muito pequena sendo da ordem de 10^{-4}, de modo que com a utilização de ambos os métodos obtém-se a mesma qualidade na análise LPC.

Utilizou-se cada um dos métodos de análise LPC no sistema implementado para obtermos informações sobre a complexidade computacional e do erro (MSE) gerado por cada um deles. Para isso, utilizou-se o arquivo “teste.wav” amostrado a 8kHz, com o tamanho de segmento de 160 amostras, taxa de superposição de 50% e a ordem do sistema igual a 10. O método da autocorrelação utiliza a recursão de Levinson Durbin, enquanto

que o método da covariância utiliza o método de “back substitution” [1]. Obtiveram-se os seguintes resultados:

Tabela 4.8: Arquivo “teste.wav”.

	MSE	Complex. Computacional
Covariância	1,71	$30{,}54 \times 10^6$
Autocorrelação	1,69	$29{,}17 \times 10^6$

Podemos observar que o erro gerado pelos dois tipos de análise LPC são praticamente iguais como já era de se esperar devido à resposta em freqüência quase idêntica nos dois métodos apresentada anteriormente. Em relação à complexidade computacional, verifica-se que apesar do número de operações matemáticas no método da covariância ser apenas um pouco maior do que no método da autocorrelação, o método da covariância apresenta um tempo de execução aproximadamente quatro vezes maior do que o método da autocorrelação.

Devido ao fato de ambos os métodos apresentarem a mesma qualidade e o método da autocorrelação possuir um tempo de execução menor, optou-se pelo método da autocorrelação para a utilização no sistema LPC implementado.

4.5 Estudo da Ordem do Codificador LPC

Conforme descrito anteriormente, em um sistema LPC utilizamos um filtro de predição linear, que é responsável pela análise dos segmentos de fala. Se pudermos transmitir poucos coeficientes por cada segmento, conseguiremos uma baixa taxa de transmissão em nosso sistema LPC, além de reduzirmos a complexidade computacional do sistema.

O número de coeficientes é determinado pela ordem do filtro. O que ocorre, é que se trabalharmos com uma ordem muito baixa, a resposta do filtro não se aproxima do sinal original, já que o modelo se torna muito pobre. O objetivo desta seção então, é chegarmos a uma conclusão sobre um valor coerente para a ordem do filtro de predição linear.

Para podermos saber se a resposta do filtro é semelhante ao sinal original, devemos plotar no domínio da freqüência, tanto o sinal original como a resposta do filtro. Fazendo isso para diversos valores de p, onde p é a ordem do filtro, podemos saber como a resposta do filtro varia em função de sua ordem. Além disso, podemos calcular o erro entre a resposta do filtro e o sinal original para termos uma medida quantitativa a respeito da qualidade do filtro.

Para obtermos a transformada de um segmento do sinal original no domínio da freqüência, utilizamos a janela de Hamming, para não introduzirmos muita distorção no domínio da freqüência, já que é neste domínio que estaremos trabalhando.

Para esta análise, foi utilizado como sinal original, um segmento de 200 amostras do fonema /a/, gravado com uma freqüência de amostragem de 8kHz. Abaixo, podemos ver sobrepostos, o gráfico da resposta em magnitude do filtro de predição linear e o espectro do sinal original (figuras 4.21 à 4.27) calculado através da FFT com 400 pontos. Apresentamos diversos gráficos, cada um deles utilizando um valor distinto para a ordem do filtro.

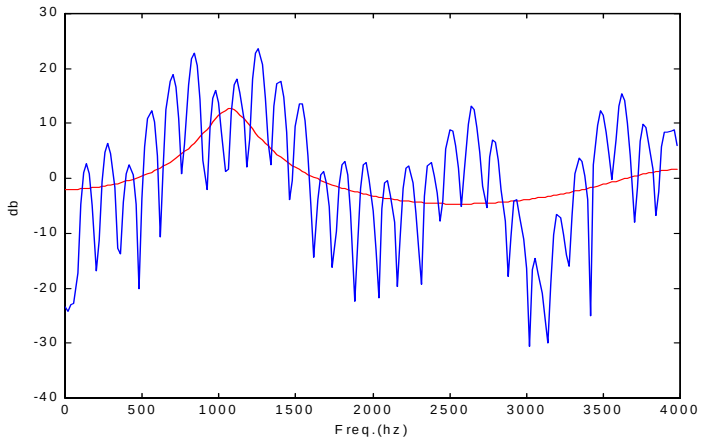

Figura 4.21: Ordem *M*=5.

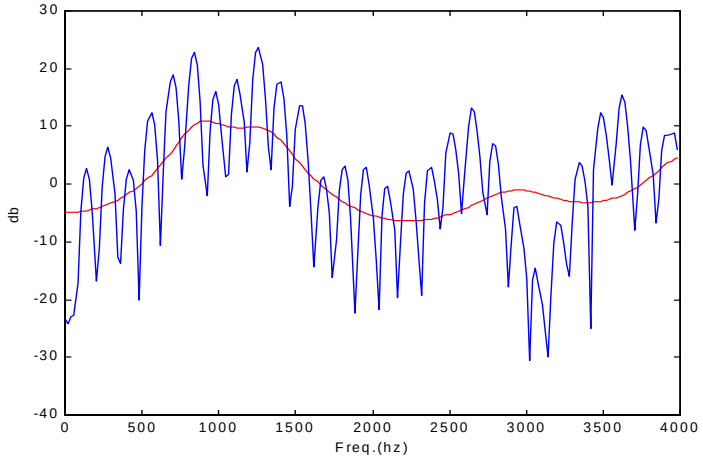

Figura 4.22: Ordem *M*=7.

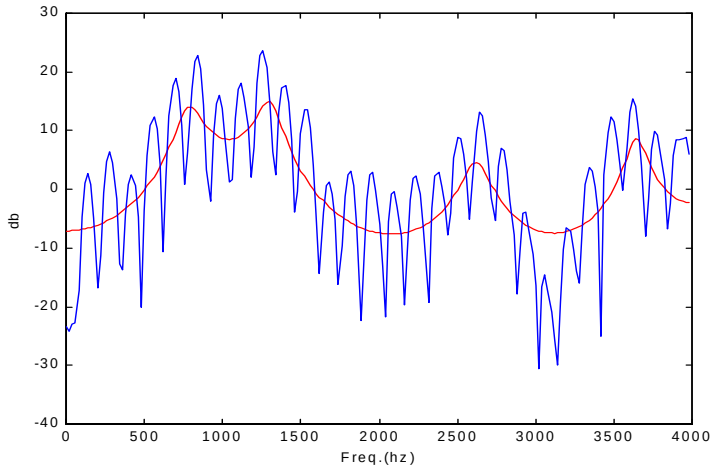

Figura 4.23 Ordem *M*=10.

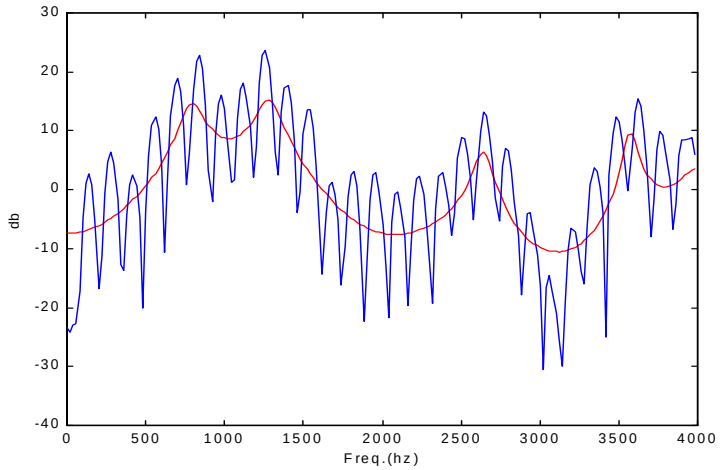

Figura 4.24: Ordem *M*=15.

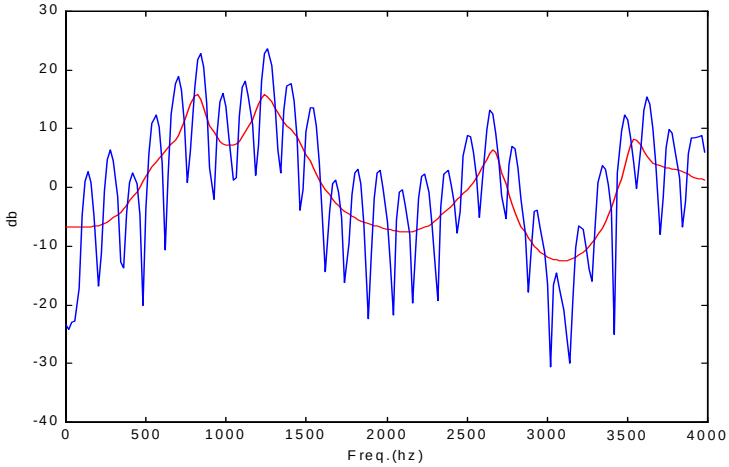

Figura 4.25: Ordem *M*=20.

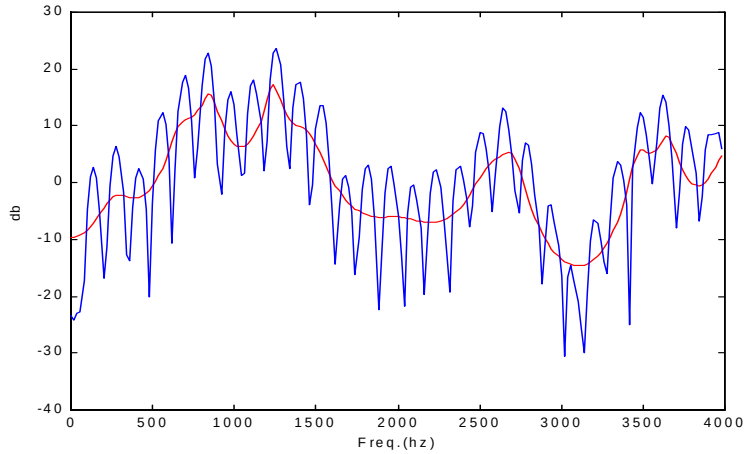

Figura 4.26: Ordem *M*=25.

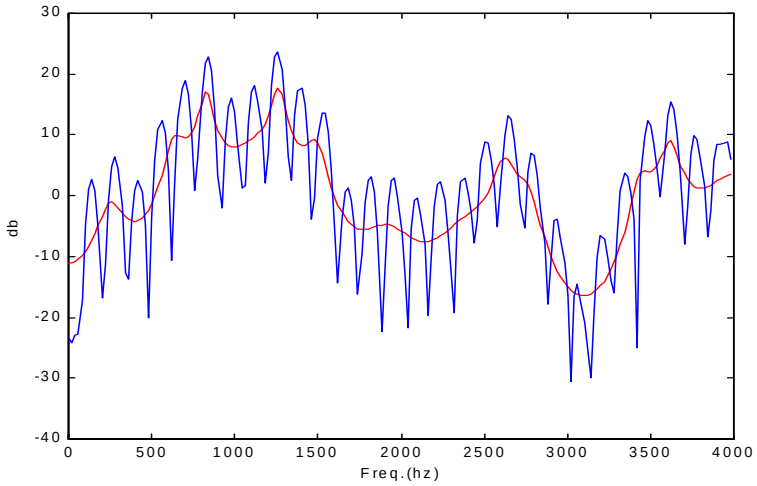

Figura 4.27: Ordem *M*=30.

Apresentamos a seguir, uma tabela que relaciona o erro entre a resposta em freqüência do filtro e a transformada do sinal no domínio da freqüência, para diferentes valores de ordem *M*. O erro é dado pelo somatório do valor absoluto do erro em cada ponto do gráfico, ou seja:

$$\text{Erro} = \text{sum}(\text{abs}(H(s)-S)) \qquad (4.2)$$

onde *H(s)* é a resposta em magnitude do filtro e *S* é a transformada de um segmento do sinal original no domínio da freqüência obtida pela FFT.

Tabela 4.9: Erro x Ordem do filtro.

Ordem do Filtro	Erro
5	2335,5
6	2335,6
7	2325,8
8	2158,1
9	2107,0
10	2069,6
11	2069,8
12	2071,8
13	2053,6
14	2049,8
15	2038,1
16	2029,6
17	2017,4
18	2015,6
19	2013,9
20	2013,3
22	1969,0
24	1950,3
26	1918,8
28	1914,7
30	1912,0
32	1912,5
34	1898,9
35	1898,7

No gráfico abaixo, podemos visualizar o erro obtido em função da ordem do filtro utilizada.

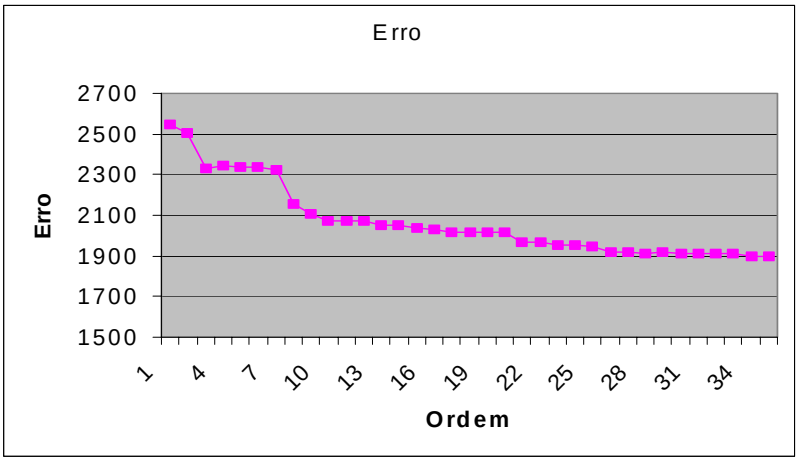

Figura 4.28: Gráfico do Erro obtido x Ordem.

A partir dos dados apresentados acima, podemos observar, que quando a ordem do filtro de predição linear passa de 10, não obtemos uma diminuição significativa no erro. Em contrapartida, para valores de 1 a 10, notamos uma significativa diminuição do erro quando incrementamos a ordem do filtro.

Outro aspecto que devemos considerar é que se aumentarmos muito a ordem do sistema, teremos que transmitir um número elevado de parâmetros para cada segmento, e o sistema LPC serve justamente para permitir baixas taxas de transmissão de sinais de fala. Além disso, quando aumentamos a ordem do sistema, estamos aumentando a sua complexidade computacional. Esses dois aspectos limitam superiormente a ordem do sistema.

Um outro estudo que apresentaremos, é como a variação da ordem do sistema influencia o sistema LPC implementado. Para isso, foi usado o arquivo "a_andre.wav", gerado com uma freqüência de amostragem de 8kHz. Rodamos o sistema com diversos valores de ordem *M*, e comparamos a complexidade computacional do sistema (flops), o erro do sinal sintetizado (MSE), e uma análise subjetiva do sinal sintetizado. Utilizamos o tamanho de segmento $L = 140$ (17,5 ms), e não utilizamos superposição para estes testes. A tabela abaixo mostra os resultados obtidos:

Tabela 4.10: Arquivo "a_andre.wav".

Ordem M	Complex. Comp. (Flops)	Erro (MSE)	Qualidade do Sinal
5	$3{,}23 \times 10^6$	2,20	-
8	$3{,}28 \times 10^6$	1,67	++
10	$3{,}31 \times 10^6$	1,72	++
15	$3{,}43 \times 10^6$	1,71	++
20	$3{,}59 \times 10^6$	1,73	++
30	$4{,}08 \times 10^6$	1,71	++

Como podemos observar, a complexidade computacional cresce um pouco quando aumentamos a ordem do sistema. Já o MSE, se mantém em níveis quase que constantes para valores de *M* maiores do que 8, sendo o valor do MSE maior para $M=5$.. Já a qualidade do sinal, é ruim quando utilizamos uma ordem $M=5$. Porém, para valores de *M* superiores a 8, obtemos aproximadamente a mesma qualidade de sinal sintetizado. Assim, considero que utilizar uma faixa de 8 a 15 para a ordem do sistema é o mais adequado.

Os arquivos gerados encontram-se em anexo denominados como: "a_XX.wav", onde XX representa a ordem do sistema utilizada.

4.6 Estudo dos Diferentes Tipos de Pulsos Glotais

Como sabemos, os sinais de fala podem ser classificados como vozeados ou não vozeados. Para cada um desses casos teremos um tipo de excitação diferente no sistema LPC para obtermos a síntese da fala. Para o caso de sons não vozeados utiliza-se uma fonte de ruído branco. Já para os sons do tipo vozeado, utiliza-se pulsos glotais periódicos, que devem se aproximar do pulso glotal humano. Esses pulsos glotais devem se repetir a cada período de "pitch". Utilizaremos os 3 tipos de pulsos glotais descritos no capítulo 3 para podermos chegarmos à conclusão de qual dentre eles gera melhores resultados.

Para a avaliação desses diferentes tipos de pulsos glotais, utilizamos cada um deles no sistema LPC implementado. O pulso do tipo 1 é mostrado na figura 3.17, o tipo 2 na figura 3.18 e o pulso do tipo 3 na figura 3.19. Foi feita uma análise quantitativa baseada no MSE entre o sinal original e o sinal sintetizado e uma análise qualitativa que é bastante subjetiva. A complexidade computacional também é comparada. A seguir apresento os resultados obtidos.

A tabela 4.11, apresenta os resultados obtidos através da utilização do som da vogal "a" como sinal original. A freqüência de amostragem é de 8kHz, o tamanho do segmento utilizado é de 160 amostras, foi utilizada uma taxa de superposição de 50% e a ordem do sistema é de 10.

Tabela 4.11: Arquivo "a_andre.wav".

	MSE	Complex. Computacional	Qualidade
Pulso Glotal 1	3,76	$5,76 \times 10^6$	++
Pulso Glotal 2	2,18	$5,77 \times 10^6$	++
Pulso Glotal 3	1,70	$5,83 \times 10^6$	+++

A tabela 4.12, apresenta os resultados obtidos para o arquivo "teste.wav", utilizando-se o sistema nas mesmas condições acima.

Tabela 4.12: Arquivo "teste.wav".

	MSE	Complex. Computacional	Qualidade
Pulso Glotal 1	3,89	$29,48 \times 10^6$	++
Pulso Glotal 2	2,96	$29,42 \times 10^6$	++
Pulso Glotal 3	1,70	$29,51 \times 10^6$	+++

Vemos que a síntese através dos três diferentes tipos de pulsos glotais qualitativamente geram resultados semelhantes, sendo que o pulso glotal do tipo 3 gera resultados um pouco melhores. Quanto ao MSE, o pulso glotal do tipo 3 é o que apresenta o melhor resultado para os dois sinais sintetizados. Em relação à complexidade computacional, não há mudanças significativas ao variarmos o tipo de pulso glotal.

A partir desses resultados adotaremos o pulso glotal do tipo 3 no sistema LPC implementado devido a pequena diferença qualitativa que este apresenta em relação aos outros dois tipos de pulsos glotais e ao melhor resultado quantitativo obtido.

Os arquivos gerados encontram-se em anexo, sendo denominados como: “a_tipoX.wav” e “tst_tipoX.wav”, onde X representa o tipo de pulso glotal utilizado segundo convencionado nesta seção.

Na figura 4.29, podemos visualizar um trecho do sinal original da vogal “i”, bem como a síntese performada com os diferentes tipos de pulso glotal. Podemos observar que o pulso glotal do tipo 2, que é obtido através da passagem do trem de impulso pelo filtro dado por (3.24), torna o sinal sintetizado mais suave.

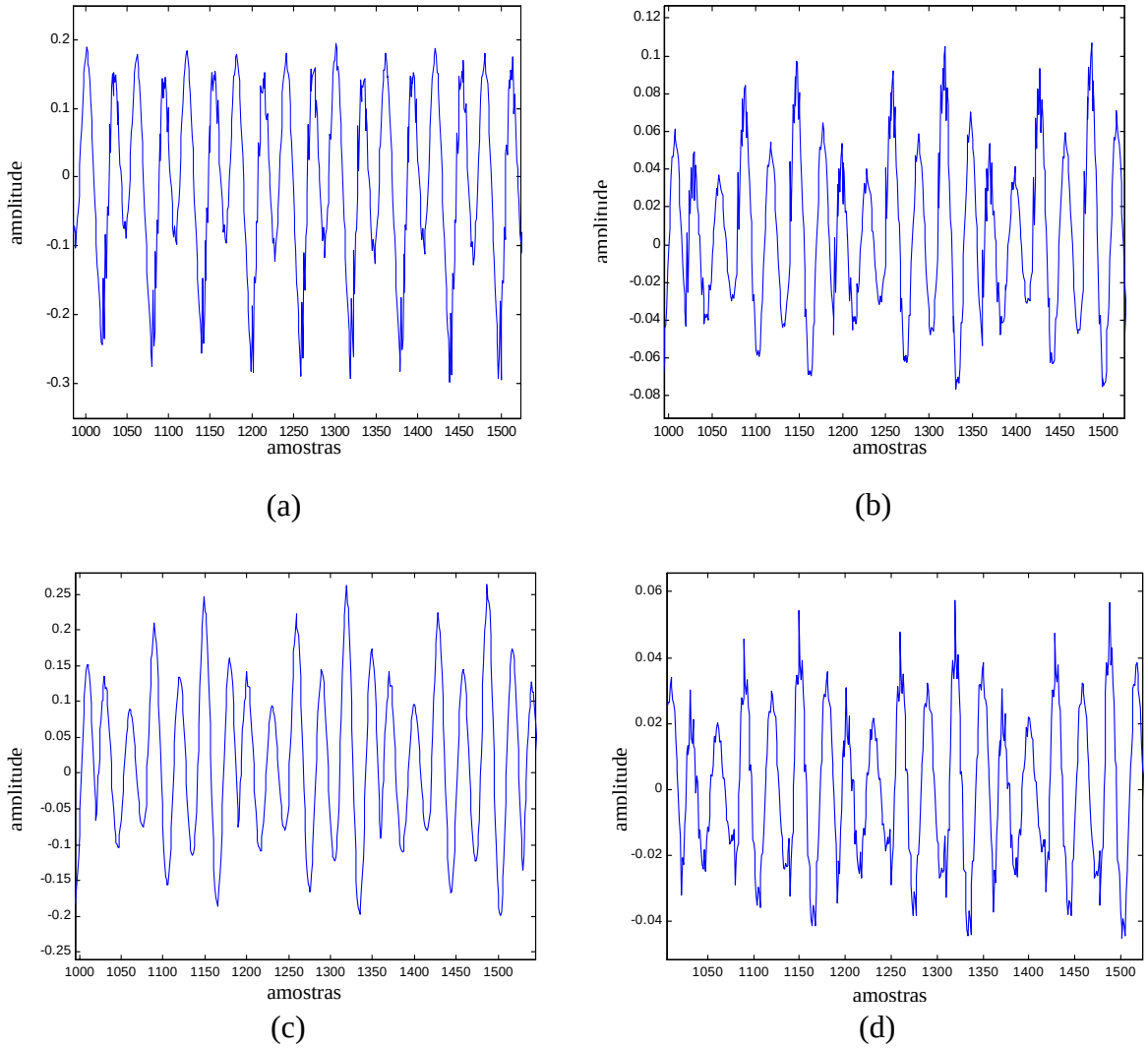

Figura 4.29: (a) Trecho original da vogal “i”; (b) Trecho da vogal “i” sintetizada com o pulso glotal do tipo 1; (c) Trecho da vogal “i” sintetizada com o pulso glotal do tipo 2; (d) Trecho da vogal “i” sintetizada com o pulso glotal do tipo 3.

Uma outra alternativa quanto ao tipo de excitação utilizado para sinais do tipo vozeado, é a excitação mista, em que adicionamos ruído branco aos pulsos glotais. Na figura 4.30, podemos ver como ficaria o sinal de excitação no caso de utilizarmos excitação mista utilizando o pulso glotal mostrado na figura 3.19 adicionado a um sinal de ruído com distribuição normal, média zero e variância 1.

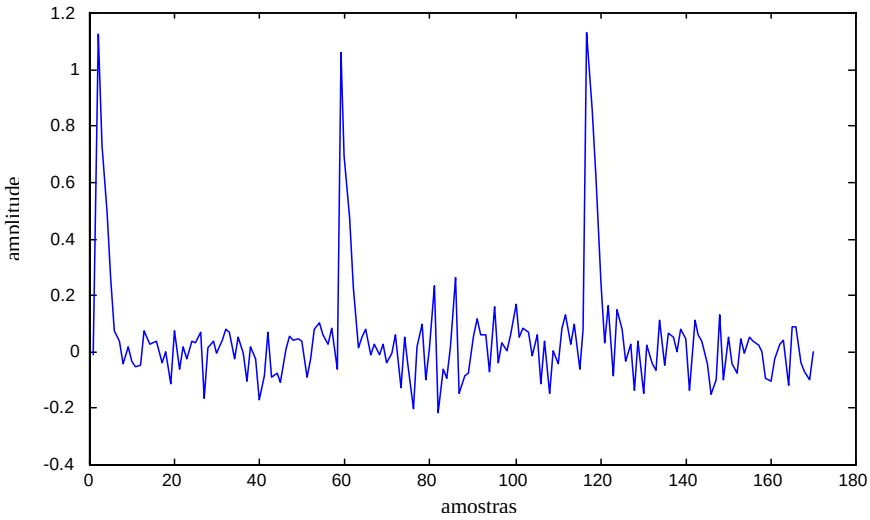

Figura 4.30: Forma de onda do pulso glotal utilizando excitação mista.

Para testarmos a utilização da excitação mista, utilizamos como sinal de excitação para segmentos do tipo vozeado o pulso glotal do tipo 3, adicionado a sinais de ruído com distribuição normal, média zero, e variância de 1, 0.5, 0.1 e 0.05 respectivamente. Para segmentos não vozeados, utilizamos como excitação este mesmo sinal de ruído. Utilizamos como sinal original o arquivo "a_andre.wav". Novamente utilizamos uma janela com 160 segmentos com uma taxa de superposição de 50%. Obtivemos como resultado os valores indicados na tabela abaixo.

Tabela 4.13: Arquivo "a_andre.wav".

	MSE	Complex. Computacional	Qualidade
Sem exc. Mista	1,70	$5{,}83 \times 10^6$	+++
Com exc. Mista/σ^2=1	1,71	$5{,}86 \times 10^6$	+++
Com exc. Mista/σ^2=0.5	1,69	$5{,}86 \times 10^6$	+++
Com exc. Mista/σ^2=0.1	1,70	$5{,}86 \times 10^6$	+++
Com exc. Mista/σ^2=0.05	1,70	$5{,}86 \times 10^6$	+++

Podemos observar que a utilização da excitação mista não trouxe nenhuma melhora na síntese de voz, com qualquer um dos ruídos testados acima. O valor do erro, expresso pelo MSE manteve-se praticamente constante assim como a qualidade do sinal sintetizado com e sem a utilização da excitação mista. Assim, optamos pela não utilização da excitação mista no sistema LPC implementado para sinais do tipo vozeado. Os arquivos gerados encontram-se em anexo sendo denominado como: "a_misto_XX.wav", onde "XX" representa a variância do ruído utilizado na excitação mista.

5. ANÁLISE CEPSTRAL

O sinal de voz é composto por uma seqüência de excitação convoluída com a resposta ao impulso do modelo do trato vocal. Nós temos acesso somente à saída deste modelo, no entanto muitas vezes é desejado que separemos esses dois componentes, principalmente no problema de reconhecimento de voz.

Geralmente, quando desejamos extrair uma parte do sinal, utilizamos algum tipo de filtro. Assim, se tivermos um ruído de alta freqüência em um sinal de baixa freqüência e desejarmos eliminar este ruído, podemos fazê-lo através de um filtro passa-baixas. Isto porque o sinal e o ruído estão em diferentes regiões do espectro.

O que ocorre no caso do sinal de voz, é que não conseguimos separar a seqüência de excitação $e(n)$ da resposta ao impulso do modelo do trato vocal $\theta(n)$ no domínio da freqüência, já que o sinal que temos é dado por:

$$s(n) = e(n) * \theta(n) \tag{5.1}$$

Assim, devemos representar o sinal de voz em um novo domínio, em que as duas componentes do sinal, $e(n)$ e $\theta(n)$, estejam combinadas linearmente e separadas uma da outra. Este domínio é o domínio cepstral, que será analisado neste capítulo.

O domínio cepstral pode ser dividido em cepstrum real e cepstrum complexo. A diferença entre os dois é que no cepstrum real, nós perdemos a informação sobre a fase do sinal, ou seja, todo sinal é de fase mínima. Já no cepstrum complexo, os coeficientes cepstrais possuem parte real e imaginária, mantendo a informação sobre a fase do sinal.

5.1 Cepstrum Real

A idéia do domínio cepstral como pudemos ver, é levar o sinal para um domínio onde a excitação e o modelo do trato vocal estejam separados e combinados linearmente de modo a poder separá-los.

Se levarmos o sinal para o domínio da freqüência, a convolução entre esses dois componentes do sinal se transformará em multiplicação, conforme a equação abaixo:

$$e(n) * \theta(n) \Leftrightarrow e(w)\ \theta(w) \tag{5.2}$$

Se depois disso aplicarmos o operador logarítmico a esse sinal, teremos que:

$$\log(e(w)\ \theta(w)) = \log(e(w)) + \log(\theta(w)) \tag{5.3}$$

Com isso, temos que o sinal obtido é uma combinação linear entre os dois componentes acima. Se aplicarmos a transformada inversa de Fourier ao sinal obtido na equação 5.3, levaremos o sinal a um novo domínio. Chamamos esse nosso domínio de

cepstral , e à nova freqüência de “quefrência”. Nesse novo domínio, os dois componentes do sinal de voz estarão em regiões diferentes do cepstrum (espectro no domínio cepstral), podendo ser separados um do outro através do processo de “liftering”, que seria a filtragem no domínio cepstral.

Assim o cepstrum real $c_s(n)$ de uma seqüência de um sinal de fala $s(n)$, é definido como:

$$c_s(n) = \Im^{-1}\{\log|\Im\{s(n)\}|\} = \frac{1}{2\pi}\int_{-\pi}^{\pi}\log|S(w)|e^{jwn}dw \qquad (5.4)$$

e

$$c_s(n) = c_e(n) + c_\theta(n) \qquad (5.5)$$

onde $c_e(n)$ corresponde à excitação e $c_\theta(n)$ corresponde ao modelo do trato vocal. Na figura 5.1 podemos ver um diagrama de blocos para a obtenção do cepstrum.

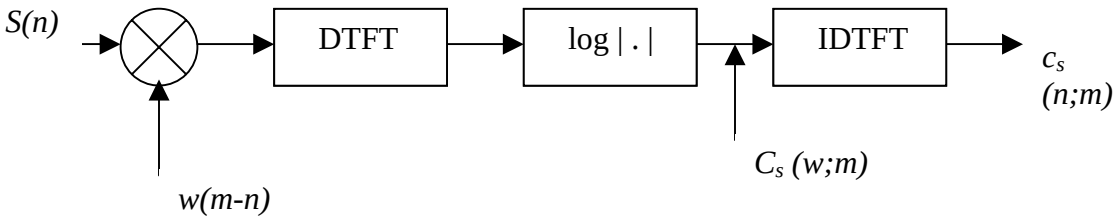

Figura 5.1: Obtenção do cepstrum real.

Na figura 5.2, podemos ver um segmento da vogal “a”, com 200 amostras e uma freqüência de amostragem de 8kHz, que será usado para o cálculo dos coeficientes cepstrais.

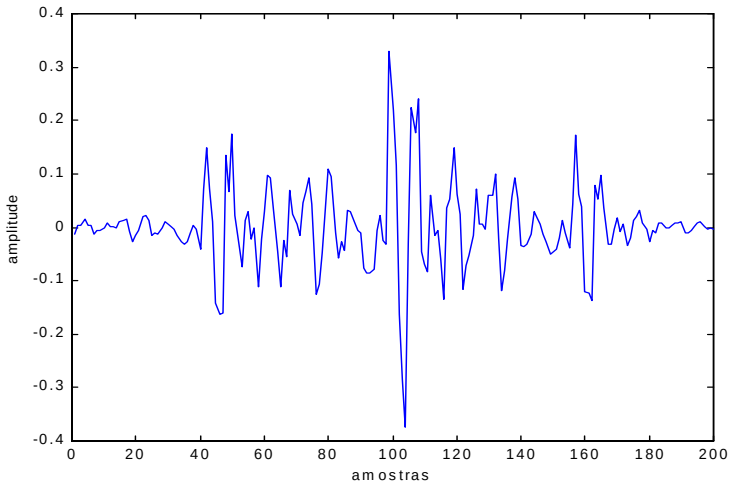

Figura 5.2: Segmento da vogal “a”.

Na figura 5.3, podemos ver a DTFT deste mesmo segmento. Este gráfico representa o espectro na freqüência do segmento da figura acima.

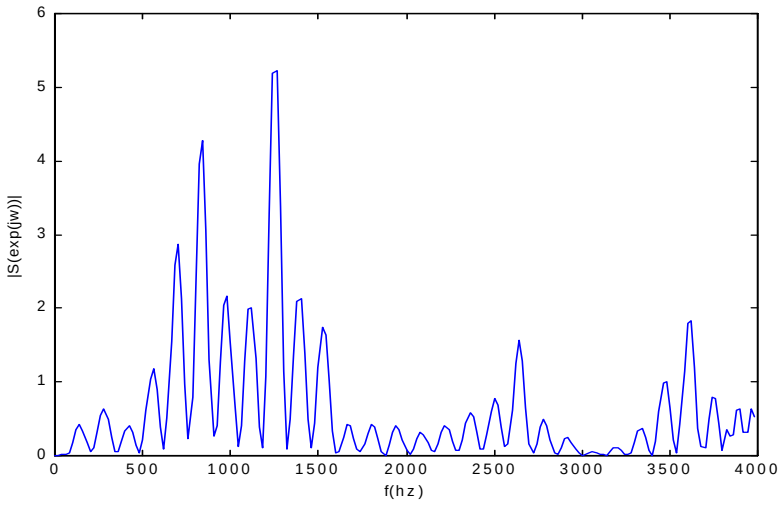

Figura 5.3: DTFT do segmento da figura 5.2.

Na figura 5.4, podemos ver como fica o sinal da figura 5.3, após a passagem pelo operador logarítmico.

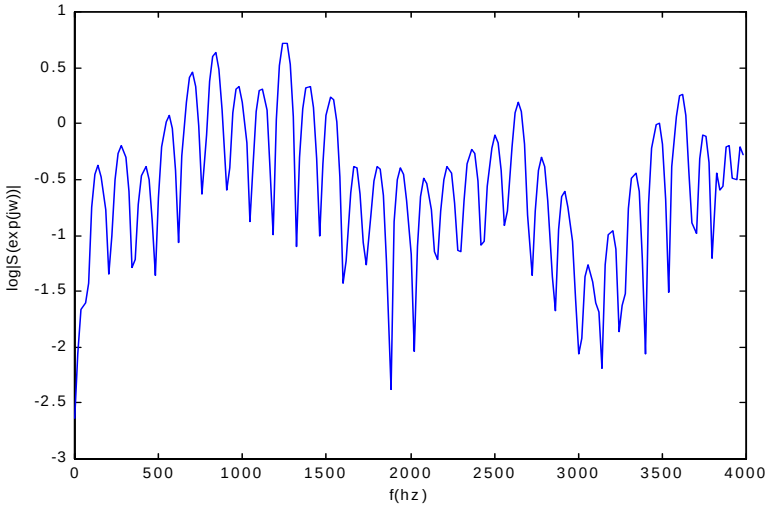

Figura 5.4: Aplicação do operador logarítmico ao sinal da figura 5.3.

Na figura 5.5, podemos finalmente observar o sinal no domínio cepstral. Esse sinal é obtido após aplicarmos a IDTFT ao sinal da figura 5.4. Como era de se esperar, podemos observar que a componente $c_\theta(n)$, que é a resposta ao impulso do trato vocal se encontra em baixas "quefrências", enquanto que $c_e(n)$, que é a componente da excitação se encontra em altas "quefrências". Com isso, pudemos ver a transformação sofrida pelo sinal até a passagem para o domínio cepstral.

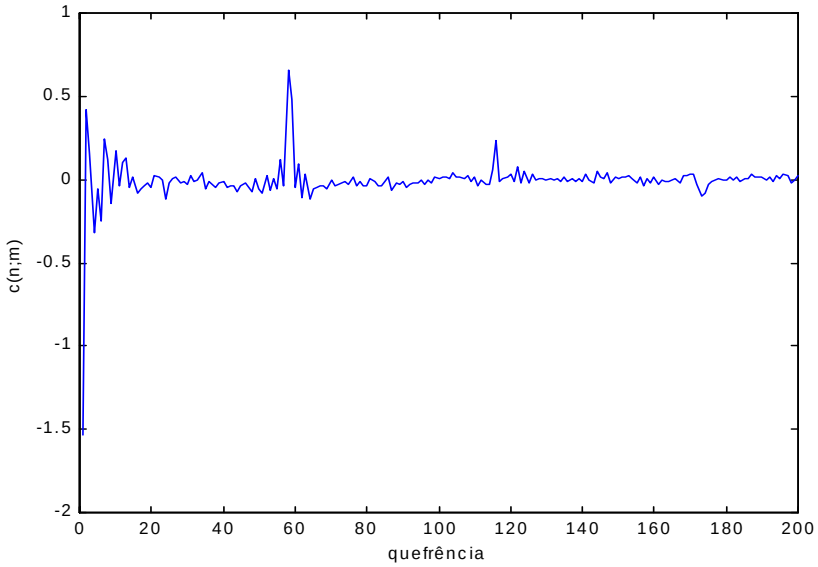

Figura 5.5: O sinal no domínio cepstral.

Através de $c_s(n)$, podemos estimar o pitch do segmento de voz que está sendo analisado. Para isso, basta observarmos o trem de impulsos que ocorre em altas "quefrências". Eles são os harmônicos da excitação. Assim, o pitch será dado pelo primeiro impulso que ocorrer no sinal no domínio cepstral. No caso da figura 5.5, isso ocorre aproximadamente na amostra 60, o que dá um período de pitch de 60/8000 = 7,5ms.

5.2 Processo de "Liftering"

O processo de "liftering" é análogo ao processo de filtragem que nós conhecemos, no entanto aplicado ao domínio cepstral. Este processo é utilizado para separarmos os componentes $c_e(n)$ e $c_\theta(n)$, já que o sinal obtido na saída do diagrama de blocos da figura 5.1 é $c_s(n) = c_e(n) + c_\theta(n)$.

Para isso, devemos multiplicar o sinal $c_s(n)$ por uma janela de curta duração, já que $c_\theta(n)$ se encontra em baixas "quefrências". A forma mais simples de se implementar essa janela seria multiplicando o sinal por uma janela retangular, porém pode-se usar outros tipos de janelas. Um tipo de janela muito usada está mostrada na figura 5.6. Esta tipo de "liftering" dá uma ênfase menor aos coeficientes que se encontram na extremidade do cepstrum. Isso é feito para normalizar a variância dos parâmetros [1].

$$l(n)=\begin{cases} 1+\frac{L}{2}\sin(\frac{\pi n}{L}), & n=0,1,...L \\ 0, & n > L \end{cases}$$

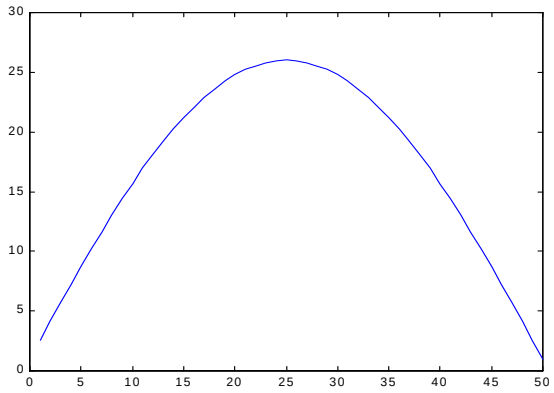

Figura 5.6: Tipo de janela utilizada para o processo de “liftering”.

Com a utilização do processo de “liftering”, podemos obter uma estimativa do formante de um segmento de voz. O que obtemos na verdade é o $\log|\theta(w)|$. Para isso, devemos aplicar o DTFT em $c_\theta(n)$, que é a componente de baixa “quefrência” de $c_s(n)$, conforme mostra o diagrama de blocos da figura 5.7.

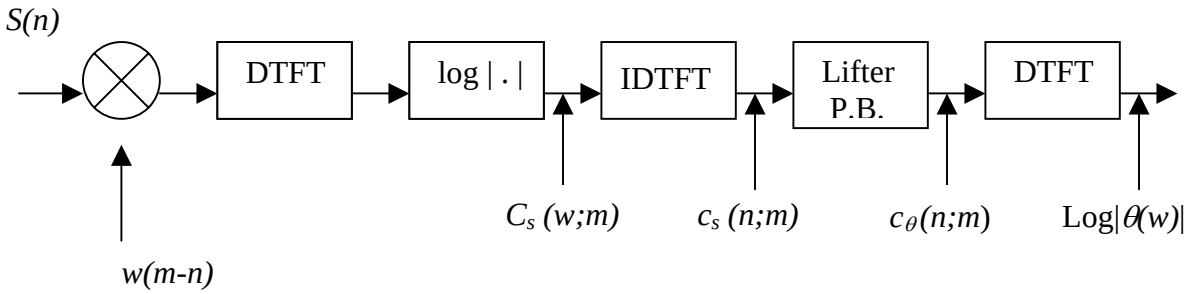

Figura 5.7: “Low-time liftering” para a obtenção de $\log|\theta(w)|$.

Na figura 5.8, podemos ver uma estimativa de $c_\theta(n)$, que foi obtido através de uma “liftragem” com uma janela retangular de tamanho L=30, a partir do sinal da figura 5.5, que é o $c_s(n)$.

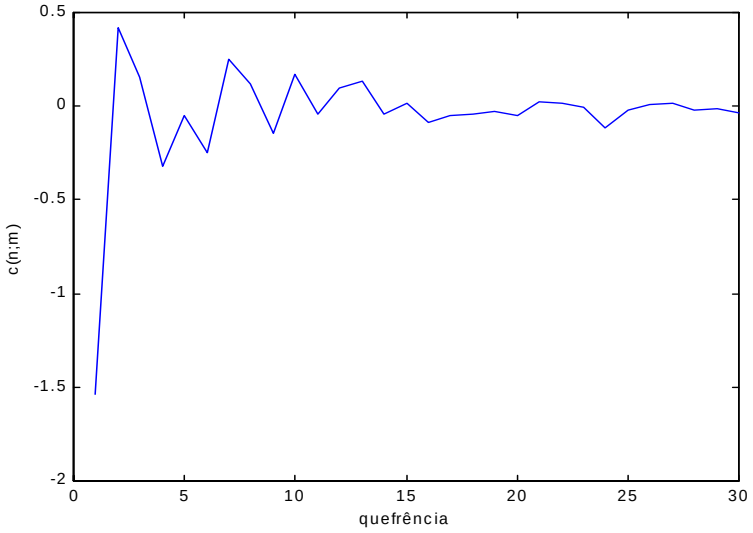

Figura 5.8: Estimativa de $c_\theta(n)$, através de "liftering".

Na figura 5.9, podemos observar o gráfico de $|\theta(w)|$ em db, obtido a partir dos coeficientes cepstrais $c_\theta(n)$ com o "liftrering" feito com uma janela retangular. Podemos comparar nesta mesma figura, o gráfico de $|\theta(w)|$ obtido através dos coeficientes LPC. Vale ressaltar que os dois gráficos não representam exatamente a mesma coisa, já que na análise cepstral, estamos retirando a componente da excitação, o que não acontece na análise LPC. Por isso não devemos esperar gráficos idênticos para as duas respostas. Utilizou-se neste caso 10 coeficientes em ambos os casos.

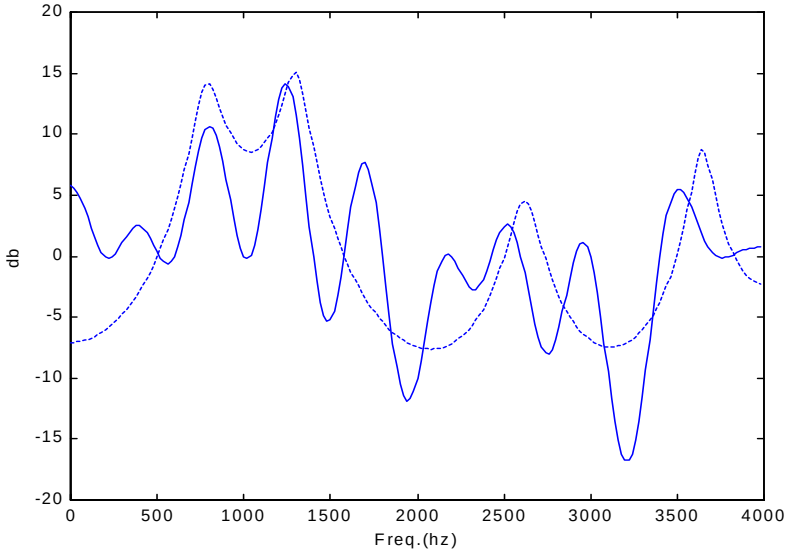

Figura 5.9: Gráfico de $|\theta(w)|$, onde a linha cheia representa a resposta obtida através dos coeficientes cepstrais e a linha pontilhada, a resposta obtida através dos coeficientes LPC.

Na figura 5.10 podemos novamente ver o gráfico de $|\theta(w)|$, porém desta vez, os coeficientes cepstrais sofreram um "liftering" com pesagem, utilizando-se para isso, a janela mostrada na figura 5.6.

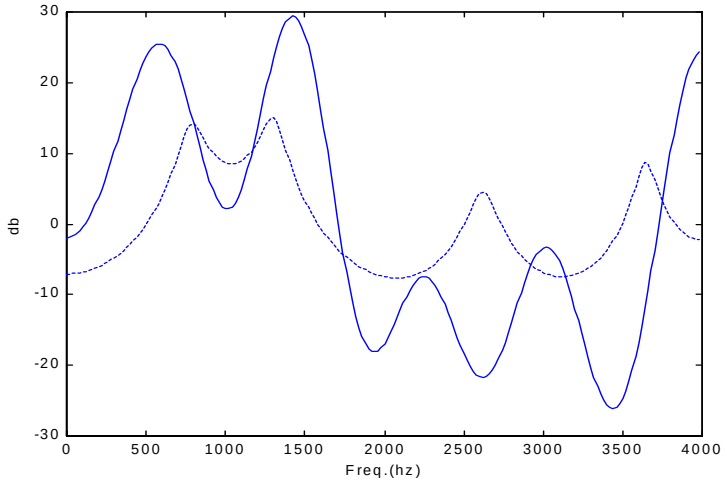

Figura 5.10: Gráfico de $|\theta(w)|$, onde a linha cheia representa a resposta obtida através dos coeficientes cepstrais e a linha pontilhada, a resposta obtida através dos coeficientes LPC.

Como podemos observar, a resposta obtida para $|\theta(w)|$ utilizando-se o "liftering" com pesagem, se aproxima mais da resposta obtida através dos coeficientes LPC.

5.3 Conversão LP – Cepstrum e Medidas de Distância

A análise por predição linear vem sendo utilizada largamente nos problemas de voz, devido ao seu fácil entendimento e implementação. Entretanto, os parâmetros LP carregam informações que degradam a performance de um sistema de reconhecimento de voz, principalmente no caso de um sistema independente de locutor, já que eles não conseguem representar apenas o modelo do trato vocal.

Esse problema é resolvido através dos coeficientes cepstrais como já pudemos ver. Como a obtenção dos coeficientes LP é muito simples, é de grande utilidade sabermos converter os coeficientes de uma representação para outra. Assim, temos que:

$$\gamma_\theta(n;m) = \begin{cases} \log \theta_0, & n=0 \\ -a(n;m) + \sum_{k=1}^{n-1} \frac{k}{n} \gamma_\theta(k;m)\, a(n-k;m), & n>0 \end{cases} \qquad (5.6)$$

onde $a(n;m)=0$ para $n \notin [1,M]$, em que M é a ordem do modelo LP e $\gamma_\theta(n;m) = 2c_\theta(n;m)$ para $n>0$

Para o problema de reconhecimento, podemos utilizar algumas medidas de distância entre coeficientes cepstrais, como a distância euclidiana, que pode ser usada na sua forma simples ou com pesagem, conforme as equações abaixo:

$$d_2[c_{\theta 1}(m), c_{\theta 2}(m)] = \sqrt{[c_{\theta 1}(m) - c_{\theta 2}(m)]'[c_{\theta 1}(m) - c_{\theta 2}(m)]} \quad (5.7)$$

$$d_{2w}[c_{\theta 1}(m), c_{\theta 2}(m)] = \sqrt{[c_{\theta 1}(m) - c_{\theta 2}(m)]'\Lambda^{-1}[c_{\theta 1}(m) - c_{\theta 2}(m)]} \quad (5.8)$$

onde: Λ é a matriz diagonal da covariância dos coeficientes cepstrais, sendo que os elementos que não estão na diagonal são ignorados.

A distância implementada por (5.8) é uma distância euclidiana com pesagem. Ela faz uma pesagem dos coeficientes de acordo com sua variância [1].

Na figura 5.11, podemos mais uma vez ver o gráfico em db de $|\theta(w)|$, obtido através dos coeficientes cepstrais. Porém desta vez, os coeficientes cepstrais utilizados foram obtidos através dos coeficientes LPC pela conversão dada por (5.6). Como sinal original, utilizamos o mesmo segmento utilizado anteriormente (vogal “a”).

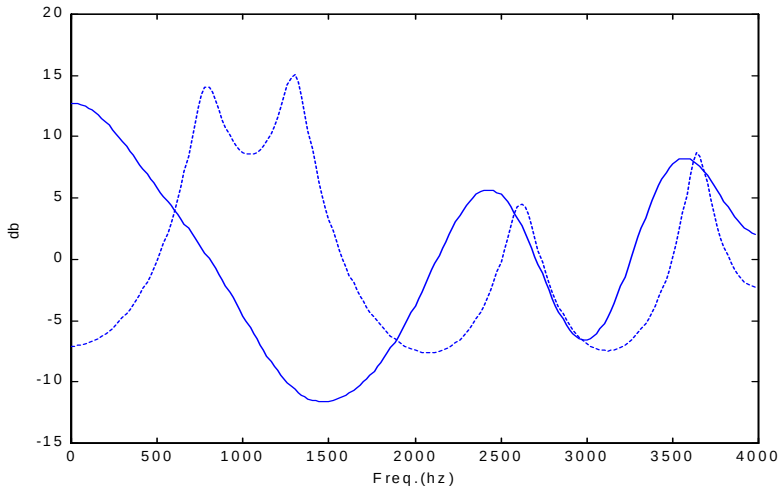

Figura 5.11: Gráfico de $|\theta(w)|$ obtido através dos coeficientes LPC, onde a linha cheia representa a resposta obtida através dos coeficientes cepstrais e a linha pontilhada, a resposta obtida através dos coeficientes LPC.

5.4 Mel-Cepstrum e Delta-Cepstrum

O mel-cepstrum e o delta-cepstrum são duas variações do cepstrum convencional. O delta-cepstrum é definido como:

$$\Delta c_s(n;m) = c_s(n;m + \delta Q) - c_s(n;m - \delta Q) \quad (5.9)$$

onde Q representa o número de amostras que a janela está sendo deslocada a cada novo "frame" e δ assume valores típicos como 1 ou 2.

Já o mel-cepstrum é baseado na observação de modelos psico-acústicos. Assim, observa-se que a freqüência percebida por nós não corresponde à freqüência real. Estabeleceu-se assim, uma nova escala de freqüência determinada mel. Assim, os coeficientes cepstrais são calculados para essa escala de freqüência e não para a escala convencional expressa em hertz. A freqüência na escala de mel, é dada por:

$$F_{mel} = 1125 \log(0{,}0016f + 1) \quad (5.10)$$

onde f é a freqüência real em hertz e F_{mel} é a freqüência em mel.

Na figura 5.12, podemos ver a correspondência entre a escala real de freqüência e a escala mel.

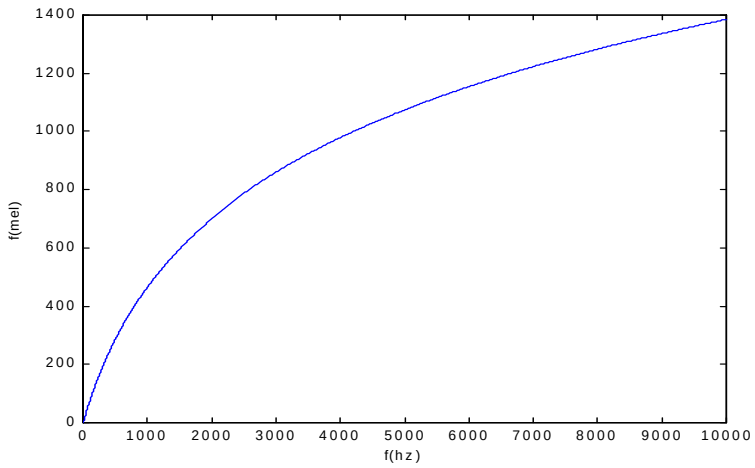

Figura 5.12: F_{hz} x F_{mel}.

6. CONCLUSÕES E DIRECIONAMENTO FUTURO

6.1 Conclusões

Foi implementado um codificador LPC para sinais de fala, e foram feitas análises de diversos parâmetros que envolvem esta implementação. Após as análises dos resultados obtidos podemos chegar a algumas conclusões:

- Pudemos constatar que segmentos de sinal de fala com um intervalo de tempo menor do que 20 a 30 ms podem ser considerados aproximadamente estacionários.

- A utilização da superposição na codificação LPC aumenta a qualidade do sinal sintetizado de forma considerável. Neste caso testamos 3 taxas diferentes de superposição e constatamos que quanto maior essa taxa, maior a qualidade do sinal.

- A utilização da superposição tornou o sinal de fala mais agudo, devido ao não sincronismo em relação ao "pitch" na concatenação dos segmentos adjacentes.

- A utilização dos diferentes tipos de janelamento não é um fator crítico na qualidade do sinal sintetizado no sistema LPC implementado, sendo que a janela do tipo Hamming apresentou um erro um pouco abaixo das outras.

- A solução LPC pelo método da autocorrelação e pelo método da covariância leva a uma mesma qualidade de síntese, contudo no algoritmo implementado o método da autocorrelação se apresentou mais veloz computacionalmente.

- Verificou-se que para qualquer ordem maior do que 8 utilizada no sistema LPC, não há melhora significativa na qualidade do sinal de fala.

- A utilização dos três diferentes tipos de pulsos glotais não alterou muito a qualidade do sinal de fala sintetizado no sistema LPC. Verificou-se que o terceiro tipo de pulso glotal apresentou resultados um pouco melhores do que os outros dois. Além disso, observamos que a utilização de excitação mista não trouxe melhoras na qualidade do sinal de fala.

- Verificamos que apesar da vantagem da codificação LPC ser uma forma comprimida da representação de sinais de fala, a qualidade do sinal sintetizado não é boa, sendo bastante inferior às codificações por formato de onda.

- Verificamos que no domínio cepstral há a separação entre formante e excitação. Isso torna os coeficientes cepstrais muito úteis no problema de reconhecimento de voz, já que a componente do sinal que mais varia entre diferentes interlocutores é a excitação.

6.2 Direcionamento Futuro

Uma limitação no sistema LPC implementado neste projeto é o fato de não haver sincronismo em função do “pitch” na concatenação de segmentos adjacentes. Isso deverá melhorar a qualidade do sinal sintetizado, não tornando o sinal mais agudo ao utilizarmos a superposição.

Outro ponto que deve ser aprofundado é verificar se há pulsos glotais mais adequados para a excitação de sinais de fala do tipo vozeado.

Esse trabalho também pode servir de base para o entendimento de codificação paramétrica para que possam ser implementados sistemas mais complexos e eficientes, como o RELP ou CELP, já que o LPC é a base de entendimento para esses codificadores.

BIBLIOGRAFIA

[1] J. R. Deller, J. G. Proakis e J. H. Hansen, *Discrete-Time Processing of Speech Signals. MacMillan,* 1993.

[2] A Spanias, *Speech Coding: A Tutorial Review*. 1995.

[3] J. P. Teixeira, "*Modelização paramétrica de sinais para aplicação em sistemas de conversão texto-fala*", Tese de mestrado, Universidade do Porto, 1995.

[4] T. P. Barnwell, III, K. Nayeby e C. H. Richardson, *Speech Coding: A Computer Laboratory Textbook*. J. Wiley & Sons, 1996.

[5] S. Haykin, *Digital Communications,* J. Wiley & Sons, 1988.

[6] S. Haykin, *Communication Systems*, J. Wiley & Sons, 1994.

[7] T. Robinson, *Speech Analysis*. 1998

[8] T. Robinson, *Speech Compression Techniques*. 1998

[9] R. C. de Lamare, "*Codificador CELP para Sinais de Fala em Língua Portuguesa*", Projeto Final de Curso, EE-UFRJ, 1999.

APÊNDICE

O sistema LPC implementado no “software” Matlab, é constituído pelas seguintes rotinas:

- Chamada.m – corpo principal do programa.
- Preenf.m – filtro de pré-ênfase.
- Seg.m – segmenta o sinal original.
- Alpc.m – calcula os coeficientes LPC e o ganho.
- Energia.m – calcula a energia de um segmento.
- Detpit.m – faz a detecção do “pitch” de um segmento.
- Amdf.m – calcula a função AMDF, sendo utilizada pela função Detpit.m.
- Slpcsuper.m – sintetiza o sinal de fala a partir dos parâmetros calculados.
- Deenf.m – filtro de de-ênfase.
- Mse.m – calcula o mse entre o sinal original e o sinal sintetizado

Para rodar o sistema, deve-se digitar no “prompt” do Matlab o seguinte comando:

chamada(*M,L,I,‘arquivo’*)

onde *M* é a ordem do sistema, *L* é o tamanho do segmento, *I* é o passo e *‘arquivo’* é o arquivo no formato “wav” com o sinal de fala a ser codificado.

Além das rotinas, encontram-se disponíveis os seguintes arquivos de som no formato “wav” correspondentes aos testes descritos neste trabalho:

- Os arquivos originais para a execução dos testes: “aeiou.wav”, “a_andre.wav” e “teste.wav”.
- “aeiou_XXX.wav” e “teste_XXX.wav”, onde “XXX” representa o tamanho de segmento utilizado no sistema.
- “aeiou_XX.wav” e teste_XX.wav”, onde “XX” representa a taxa de superposição utilizada no sistema.
- “a_*.wav” e “teste_*.wav”, onde “*” representa o nome da janela utilizada na segmentação do sinal original.
- “a_XX.wav”, onde XX representa a ordem do sistema utilizada.
- “a_tipoX.wav” e “tst_tipoX.wav”, onde X representa o tipo de pulso glotal utilizado como excitação do sistema.
- “a_misto_*”, onde “*” representa a variância do sinal de ruído utilizado na excitação mista.