Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação

ESTUDO DE ALGORITMOS DE QUANTIZAÇÃO VETORIAL APLICADOS A SINAIS DE FALA

Ricardo Paranhos Velloso Violato

Dissertação de Mestrado apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos para obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: Engenharia de Computação.

Orientador: Fernando José Von Zuben

Campinas, SP
2010

Este exemplar correspondente à redação final da Dissertação/Tese
defendida por: Ricardo Paranhos Velloso Violato
e aprovada através da Comissão Julgada em: 08 / 07 / 2010
Fernando José Von Zuben
orientador

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

V811e

Violato, Ricardo Paranhos Velloso
Estudo de algoritmos de quantização vetorial aplicados a sinais de fala / Ricardo Paranhos Velloso Violato. – Campinas, SP: [s.n.], 2010.

Orientador: Fernando José Von Zuben.
Dissertação de Mestrado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Codificação de voz. 2. Sistemas de processamento da fala. 3. Algoritmos - Processamento de dados. 4. Aprendizado de computador. 5. Inteligência artificial - Processamento de dados. I. Von Zuben, Fernando José. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Título em Inglês:	Study of vector quantization algorithms applied to speech signals
Palavras-chave em Inglês:	Speech coding, Speech processing systems, Algorithms - Data processing, Learning computer, Artificial intelligence - Data processing
Área de concentração:	Engenharia de Computação
Titulação:	Mestre em Engenharia Elétrica
Banca Examinadora:	Sarajane Marques Peres, Romis Ribeiro de Faissol Attux
Data da defesa:	08/07/2010
Programa de Pós Graduação:	Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato: Ricardo Paranhos Velloso Violato

Data da Defesa: 8 de julho de 2010

Título da Tese: "Estudo de Algoritmos de Quantização Vetorial Aplicados a Sinais de Fala"

Prof. Dr. Fernando José Von Zuben (Presidente): Fernando José Von Zuben

Profa. Dra. Sarajane Marques Peres: Sarajane Marques Peres

Prof. Dr. Romis Ribeiro de Faissol Attux: Romis Attux

Resumo

Este trabalho apresenta um estudo comparativo de três algoritmos de quantização vetorial, aplicados para a compressão de sinais de fala: k-médias, NG (do inglês *Neural-Gas*) e ARIA. Na técnica de compressão utilizada, os sinais são primeiramente parametrizados e quantizados, para serem armazenados e/ou transmitidos. Para recompor o sinal, os vetores quantizados são mapeados em quadros de fala, que são, por sua vez, concatenados, através de uma técnica de síntese concatenativa. Esse sistema pressupõe a existência de um dicionário (*codebook*) de vetores-padrão (*codevectors*), os quais são utilizados na etapa de codificação, e de um dicionário de quadros, que é utilizado na etapa de decodificação. Tais dicionários são gerados aplicando-se um algoritmo de quantização vetorial junto a uma base de treinamento. Em particular, deseja-se avaliar o algoritmo imuno-inspirado denominado ARIA e sua capacidade de preservação da densidade da distribuição dos dados. São testados também diferentes conjuntos de parâmetros para identificar aquele que produz os melhores resultados. Por fim, são propostas modificações no algoritmo ARIA visando ganho de desempenho tanto na preservação de densidade quanto na qualidade do sinal sintetizado.

Palavras-chave: codificação de fala, quantização vetorial, algoritmo imuno-inspirado, preservação de densidade.

Abstract

This work presents a comparative study of three algorithms for vector quantization, applied for the compression of speech signals: k-means, NG (Neural-Gas) and ARIA. In the compression technique used, the signals are first parameterized and quantized to be stored and/or transmitted. To reconstruct the signal, the quantized vectors are mapped into speech frames, which are concatenated through a concatenative synthesis technique. This system assumes the existence of a dictionary (*codebook*) of reference vectors (*codevectors*), which is used in the coding step, and a dictionary of frames, which is used in the decoding step. These dictionaries are generated by applying a vector quantization algorithm within a training database. In particular, we want to evaluate the immune-inspired algorithm called ARIA and its ability to preserve the density of data distribution. Different sets of parameters are also tested in order to identify the one that produces the best results. Finally, modifications to the ARIA algorithm are proposed aiming at obtaining gain in performance in both the preservation of density and the quality of the synthesized signal.

Keywords: speech coding, vector quantization, immune-inspired algorithm, density preservation.

Agradecimentos

Ao meu orientador, Prof. Fernando José Von Zuben, sou grato pela orientação.

Aos colegas de pós-graduação e do CPqD, pelas críticas e sugestões.

A minha família, pelo apoio durante esta jornada.

À FEEC, por propiciar acesso às instalações e por toda a infra-estrutura para o desenvolvimento da pesquisa.

Ao CPqD, pela disponibilização da base de dados de fala.

Sumário

Lista de Figuras

Lista de Tabelas

Trabalhos Publicados Pelo Autor

1. Ricardo P. V. Violato, Fernando J. Von Zuben, Flávio O. Simões, Mário Uliani Neto, Edson J. Nagle, Fernando O. Runstein, Leandro de C. T. Gomes. "Agrupamento Sensível à Densidade para a Quantização de Sinais de Fala". 30º *Congresso Ibero-Latino-Americano de Métodos Computacionais em Engenharia* (CILAMCE 2009), Armação de Búzios, Rio de Janeiro, Brasil, 08-11 novembro 2009.

2. Ricardo P. V. Violato, Alisson G. Azzolini, Fernando J. Von Zuben. "Antibodies with Adaptive Radius as Prototypes of High-Dimensional Datasets". *Proceedings of the 9th International Conference on Artificial Immune Systems* (ICARIS'2010), Lecture Notes in Computer Science, vol. 6209, pp. 158-170, 2010.

3. Alisson G. Azzolini, Ricardo P. V. Violato, Fernando J. Von Zuben. "Density Preservation and Vector Quantization in Immune-Inspired Algorithms". *Proceedings of the 9th International Conference on Artificial Immune Systems* (ICARIS'2010), Lecture Notes in Computer Science, vol. 6209, pp. 33-46, 2010.

Capítulo 1

Introdução

Por ser a mais simples e natural forma de comunicação do ser humano, a fala sempre despertou um grande interesse científico. No que tange à engenharia, o objetivo das pesquisas na área de processamento de fala era, inicialmente, desenvolver sistemas de armazenamento do sinal e sistemas de comunicação por voz à distância, ou telefonia. Tais sistemas primordiais eram todos analógicos, ou seja, a própria forma de onda do sinal era transmitida e/ou armazenada.

Com o avanço tecnológico, esses sistemas analógicos migraram para sistemas digitais, em virtude das vantagens que a representação digital do sinal oferece. A digitalização é um processo no qual os sinais de fala, que são originalmente analógicos, são amostrados e quantizados, passando a ser representados por uma sequência de bits. A recente expansão dos computadores e dos sistemas de comunicação digitais vem tornando o uso de sinais de fala digitalizados cada vez mais comum.

Esse desenvolvimento só foi possível graças ao estudo de meios eficientes para transmitir, armazenar e parametrizar o sinal de fala. Esse mesmo avanço tecnológico também provocou o surgimento de outras áreas de estudo, por exemplo, a conversão texto-fala. Conversores texto-fala, mais conhecidos por sua sigla em inglês TTS (*text-to-speech*), são sistemas que sintetizam o sinal de fala correspondente à leitura de um texto.

Sistemas TTS têm sido aplicados em diversas áreas, mas sempre norteados pelo objetivo de facilitar a comunicação do homem com os computadores. Dado que os computadores apresentam grande capacidade de armazenar e processar informações, na presença de um conversor texto-fala essas informações podem ser disponibilizadas para um usuário através da fala. Exemplos de aplicações são inúmeros, dos quais se destacam: consultas por telefone (e-mail, catálogos, informações bancárias, horários de voos, calendário esportivo etc.), auxílio ao aprendizado da língua, sistemas de navegação e auxílio a deficientes visuais e vocais. A evolução da tecnologia complementar à conversão, ou seja, o reconhecimento de fala, permitirá que nossa interação com as máquinas se torne cada vez mais rápida e natural, ampliando o leque de aplicações imagináveis.

Isto aponta para uma popularização dos sistemas TTS. Os sistemas atuais de conversão texto-fala que buscam sintetizar fala de alta naturalidade são baseados em uma técnica conhecida como síntese concatenativa por seleção de unidades. Tais sistemas dependem de uma grande base de sinais de fala pré-gravados, o que pode dificultar ou até mesmo inviabilizar certas aplicações.

Neste trabalho, é apresentada uma técnica de codificação de sinais de fala que visa a compressão de bases de sinais, como as empregadas nos conversores texto-fala. Nessa técnica, os sinais são primeiramente divididos em quadros, e, em seguida, são parametrizados e quantizados, para serem armazenados e/ou transmitidos. Para recompor o sinal, os vetores quantizados são mapeados em quadros de fala, que são, por sua vez, concatenados, através de uma técnica de síntese concatenativa.

Dois aspectos desse processo são analisados no presente trabalho: é feito um estudo dos atributos do sinal de fala utilizados na etapa de parametrização, bem como uma análise de algoritmos de quantização vetorial empregados, é claro, na etapa de quantização.

Para o funcionamento desse sistema, é necessária a existência de um dicionário (*codebook*) de vetores-padrão (*codevectors*), os quais são utilizados na etapa de codificação, e de um dicionário de quadros, que é utilizado na etapa de decodificação. Tais dicionários são gerados aplicando-se um algoritmo de quantização vetorial junto a uma base de treinamento.

Em quantização vetorial, o objetivo é representar certa distribuição de dados utilizando um número de protótipos significativamente menor que o número de dados. No processo de quantização, um vetor qualquer é associado a um protótipo, que é uma versão quantizada e, portanto, aproximada do vetor original. A quantização vetorial é uma forma de compressão de dados, pois apenas os protótipos precisam ser armazenados, ao invés da base de dados inteira.

Neste trabalho, são testados três algoritmos de quantização vetorial: *k*-médias, NG (do inglês *Neural-Gas*) e ARIA (da sigla em inglês para *Adaptive Radius Immune Algorithm*). A este último é dedicada atenção especial e são propostas duas modificações para melhorar o seu desempenho. Ao associar os dados a antígenos e os protótipos (vinculados ao processo de quantização) a anticorpos, o algoritmo ARIA representa um paradigma alternativo de quantização vetorial, o que justifica o seu emprego e a comparação de desempenho com outras propostas.

O ARIA é um algoritmo de agrupamento de dados pertencente à classe dos Sistemas Imunológicos Artificiais, isto é, sua criação foi inspirada no sistema imunológico natural. Sistemas imunológicos artificiais são definidos como mecanismos computacionais compostos por metodologias de processamento de informação, inspiradas no sistema imune natural, visando a solução de problemas do mundo real.

A apresentação conceitual deste trabalho começa justamente pelo sistema imunológico natural, no Capítulo 2. Nesse mesmo capítulo, também é definido o conceito de sistemas imunológicos artificiais e são apresentadas características gerais comuns a esses sistemas e o escopo de sua utilização.

Em seguida, o Capítulo 3 trata da quantização vetorial, onde também são descritos os algoritmos utilizados neste trabalho, bem como métodos para avaliar a qualidade da solução fornecida por esses algoritmos.

O Capítulo 4 aborda o tema da síntese de fala, especialmente a síntese concatenativa, e utiliza como pano de fundo a conversão texto-fala, que também é apresentada nesse capítulo.

O sistema de codificação do sinal de fala empregado neste trabalho é descrito no Capítulo 5, que inclui uma explicação das técnicas de processamento de sinais necessárias para sua implementação. Além disso, este capítulo descreve os parâmetros do sinal de fala testados e, ainda, métodos de avaliação da qualidade do sinal de fala sintético produzido.

Isto posto, no Capítulo 6, são apresentados os resultados obtidos. O capítulo começa com a descrição da base de dados utilizada. Em seguida, relata os resultados auferidos com diferentes conjuntos de parâmetros, em termos da qualidade do sinal de fala gerado. Após determinar o conjunto de parâmetros que levou ao melhor resultado, o desempenho dos algoritmos de quantização é comparado. A partir daí, são propostas modificações no algoritmo ARIA, que aprimoraram sua performance. Por último, é avaliada a relação entre a qualidade do sinal de fala sintetizado e as medidas de qualidade da quantização vetorial (erro de quantização e entropia relativa), e o efeito das modificações propostas para o ARIA em relação a seu desempenho mediante essas medidas.

As considerações finais fazem parte do Capítulo 7, onde são apresentadas as conclusões e perspectivas futuras do trabalho realizado, procurando mostrar as contribuições feitas e delineando sugestões para o prosseguimento da pesquisa desenvolvida.

Capítulo 2

Sistemas Imunológicos: Do Natural aos Artificiais

2.1 Introdução

Todos os seres vivos têm a habilidade de resistir a agentes causadores de doenças. A natureza dessa resistência varia de uma espécie para outra e é uma função da complexidade do organismo. Os mamíferos, particularmente os humanos, desenvolveram sistemas imunológicos altamente sofisticados que interagem com outros sistemas do corpo (como o sistema nervoso e o sistema endócrino) para manter a vida (de Castro, 2006). Neste capítulo são apresentados alguns conceitos básicos referentes apenas ao sistema imunológico dos animais vertebrados, mais especificamente dos mamíferos, pois se trata do principal objeto da pesquisa imunológica e da mais importante fonte de inspiração para os sistemas imunológicos artificiais.

Algumas características do sistema imune despertaram, principalmente a partir dos anos de 1990, o interesse de engenheiros e cientistas da computação, que se inspiraram nelas para desenvolver métodos computacionais conhecidos hoje como sistemas imunológicos artificiais. Sistemas imunológicos artificiais são mecanismos computacionais compostos por metodologias de processamento de informação, inspiradas no sistema imune natural, visando a solução de problemas do mundo real (Dasgupta, 1998). Juntamente com a inteligência de enxame, os sistemas imunológicos artificiais são um dos campos de estudo mais recentes dentro da computação inspirada na natureza (de Castro, 2006).

Este capítulo apresenta uma descrição breve do sistema imunológico natural e das propriedades que serviram de inspiração para a criação de algoritmos computacionais (Seção 2.2). Em seguida, é definido o conceito de sistemas imunológicos artificiais, são apresentadas características gerais comuns a esses sistemas, o escopo de sua utilização e, por fim, descreve-se como algumas de suas

propriedades podem ser empregadas para quantização vetorial (Seção 2.3).

2.2 Sistema Imunológico Natural

Dentre os mecanismos de defesa do organismo pode-se incluir barreiras físicas, como a pele, e bioquímicas, derivadas das condições fisiológicas de funcionamento do corpo, como o pH. Caso essas barreiras não sejam eficazes no combate a um agente invasor, entra em ação o sistema imunológico (Figura 2.1).

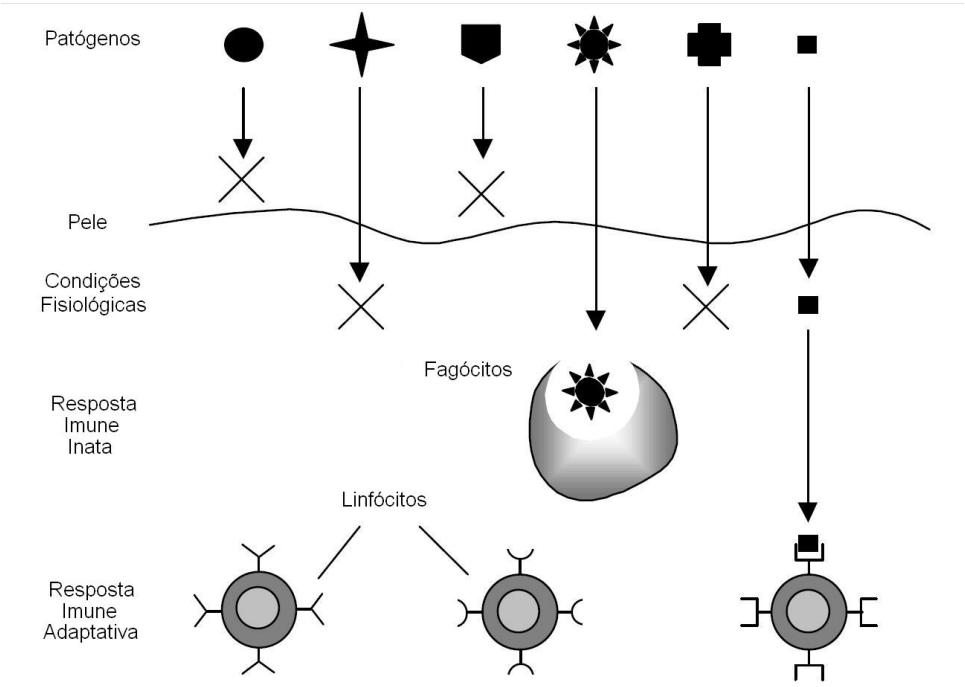


Fig. 2.1: **Mecanismos de defesa do organismo: barreiras físicas e fisiológicas (algumas vezes consideradas parte da resposta imune inata), resposta imune inata e resposta imune adaptativa. Figura extraída de (de Castro, 2001), com permissão do autor.**

O sistema imunológico é capaz de reconhecer e combater agentes invasores, também chamados de patógenos, sendo um mecanismo de auto-identificação (identificação do próprio indivíduo), responsável pela manutenção da integridade física e homeostase do organismo (de Castro, 2001). Sua complexidade pode ser comparada à do cérebro em muitos aspectos, e, assim como o sistema nervoso central, cada indivíduo possui o seu próprio sistema imunológico, com capacidades e vulnerabilidades particulares. Gêmeos idênticos, portanto nascidos com o mesmo código genético, desenvolvem sistemas imunológicos diferentes, do mesmo modo como eles desenvolvem cérebros diferentes. O

sistema imunológico de cada pessoa armazena o histórico único de vida do indivíduo, pois esse sistema, como o cérebro, se organiza através de sua experiência de vida (Cohen, 2004).

Os mecanismos de reconhecimento e combate, inerentes ao sistema imunológico, reagem a padrões moleculares chamados de antígenos, presentes em agentes invasores. Uma vez que o sistema imunológico reconheça um antígeno de um dado patógeno, a resposta imunológica é ativada, buscando eliminar esse invasor potencialmente perigoso (Janeway et al., 2001).

Existem dois tipos de respostas imunes: a inata e a adaptativa. O sistema imune inato é capaz de atuar em uma vasta variedade de agentes invasores, sem que tenha sido exposto previamente a eles. Já o sistema imune adaptativo é específico para cada antígeno e precisa se adaptar, como seu próprio nome diz, e aprender a reconhecer um agente desconhecido para poder combatê-lo. Uma vez combatido um patógeno, o sistema consegue memorizar seu padrão molecular (antígeno) e ter uma resposta mais rápida e eficiente em uma futura invasão.

Em seu trabalho, de Castro e Timmis (2002) enumeram e discutem algumas das propriedades do sistema imunológico, muitas delas evidentemente interessantes sob a perspectiva computacional, das quais destacam-se: reconhecimento de padrões, manutenção de diversidade, memória, tolerância a ruído, remoção de redundância, aprendizagem por reforço e auto-organização.

O foco principal neste capítulo será o sistema imune adaptativo, que foi a principal inspiração para a criação dos Sistemas Imunológicos Artificiais (de Castro e Timmis, 2002). Há muitas teorias que buscam explicar o funcionamento do sistema imune adaptativo, mas duas delas formam a base do algoritmo utilizado nesse trabalho: o princípio da Seleção Clonal (Burnet, 1959) e a Teoria da Rede Imunológica (Jerne, 1974). Deve-se destacar que algumas teorias que serão apresentadas não são completamente aceitas pelos imunologistas, mas isso não impede que sejam aproveitadas como inspiração para produzir algoritmos computacionais para a solução de problemas.

2.2.1 Aspectos Históricos

A imunologia é uma ciência relativamente nova. Sua origem é atribuída a Edward Jenner, que em 1796 descobriu o processo de inocular indivíduos sadios com cepas atenuadas de agentes causadores de doença, a fim de obter proteção natural do organismo contra a enfermidade, o que é conhecido desde então como vacinação.

Quando Jenner introduziu a vacinação, possivelmente nada sabia a respeito dos agentes infecciosos que causam doenças. Foi somente mais tarde, no século XIX, que Robert Koch provou que as doenças infecciosas eram causadas por microorganismos, cada um responsável por uma enfermidade ou patologia. Reconhecem-se atualmente quatro grandes categorias de microorganismos causadores de doenças ou patógenos (Janeway et al., 2001): os vírus, as bactérias, os fungos e outros organismos eucarióticos (formados por células com núcleo separado do citoplasma) relativamente grandes

ou complexos, coletivamente chamados de parasitas.

Faltava ainda compreender quais mecanismos eram responsáveis pela proteção do corpo contra esses patógenos. A imunidade inata foi descoberta pelo imunologista russo Elie Metchnikoff, que verificou que muitos microorganismos podiam ser ingeridos e digeridos por células chamadas de macrófagos, em um processo que recebeu o nome de fagocitose. A resposta imune adaptativa foi descoberta posteriormente por Emil von Behring e Shibasaburo Kitasato. Eles identificaram no soro de indivíduos vacinados a presença de certas substâncias, que foram então denominadas de anticorpos, que se ligavam especificamente aos agentes infecciosos (Janeway et al., 2001).

Desde então, o entendimento do sistema imune avançou bastante e muitas teorias surgiram para tentar explicar como ocorre a resposta imune, como se dá a interação dos patógenos com o sistema imune, qual a relação entre a imunidade inata e a imunidade adaptativa e como esses sistemas interagem com o restante do organismo.

2.2.2 O Sistema Imune Inato

A imunidade inata é responsável pela primeira linha de defesa do hospedeiro contra patógenos. Os microorganismos encontrados diariamente na vida de um indivíduo normal causam doenças apenas ocasionalmente. Em sua maioria eles são detectados e destruídos em questão de horas pelos mecanismos de imunidade inata (Janeway et al., 2001). Como indica o nome, esses mecanismos existem no organismo antes mesmo de um encontro com um agente infeccioso, e são rapidamente ativados por ele.

A imunidade inata é dita não-específica, pois é capaz de reconhecer estruturas comuns a muitos patógenos. Na realidade, seus mecanismos são capazes de discriminar entre células do hospedeiro e superfícies de patógenos, ou seja, entre próprio e não-próprio. A imunidade inata também é dita não-adaptativa, pois o sistema imune inato não desenvolve memória e respostas secundárias, ou seja, não se adapta para produzir uma resposta mais rápida e eficaz a um novo encontro com o mesmo agente invasor (Pinchuk, 2002).

Pode-se dizer que a imunidade inata é inicialmente realizada por uma barreira física, formada pelo epitélio que separa as superfícies internas e externas do corpo, ou seja, a pele e as mucosas dos tratos respiratórios, gastrintestinal e reprodutivo. Internamente, a imunidade inata é composta por células especializadas em reconhecer e eliminar microorganismos, incluindo os leucócitos (neutrófilos e macrófagos), as células matadoras naturais NK (do inglês *Natural Killers*) e alguns linfócitos T e B (descritos na Seção 2.2.3) com pouca especificidade aos receptores dos antígenos. Fazem parte também do sistema imune inato algumas proteínas e peptídeos circulantes que exercem um papel antimicrobiano e formam o chamado sistema do complemento (Pinchuk, 2002).

O sistema imune inato usa uma diversidade de receptores que reconhecem e respondem aos pa-

tógenos. Aqueles que reconhecem diretamente a superfície dos patógenos frequentemente ligam-se a padrões repetitivos, que são característicos das superfícies microbianas, mas que não são encontrados nas células do hospedeiro. Alguns desses receptores estimulam diretamente a fagocitose, enquanto outros são produzidos como moléculas secretadas que promovem a fagocitose dos patógenos pela ativação do complemento (Janeway et al., 2001).

O sistema do complemento é um sistema de proteínas plasmáticas que provê a primeira resposta imune inata. Quando moléculas desse sistema se ligam a certos patógenos, elas ajudam na eliminação deles através de dois processos. O primeiro é um processo pelo qual proteínas do complemento rompem a membrana no patógeno, formando poros na sua superfície e resultando na sua destruição. O segundo é a opsonização, que consiste no recobrimento de um patógeno por proteínas que sinalizam para os macrófagos que esse complexo deve ser fagocitado (Segel e Cohen, 2001).

Resultados da diferenciação de monócitos (que circulam no sangue), após estes migrarem para os tecidos, os macrófagos são literalmente grandes células comedoras. Eles circulam por tecidos de todo o corpo ingerindo e digerindo antígenos. Além disso, também são capazes de estimular a ação dos linfócitos T (ver Seção 2.2.3), atuando como células apresentadoras de antígenos (APC - do inglês *Antigen-Presenting Cell*).

Outro tipo de leucócitos, além dos macrófagos, os neutrófilos são os elementos celulares mais numerosos e importantes da resposta inata. Da mesma forma que os macrófagos, eles têm receptores de superfície para constituintes comuns de bactérias e complemento e são as principais células que englobam e destroem os microorganismos invasores (Janeway et al., 2001). Uma diferença importante entre os macrófagos e os neutrófilos é que os primeiros têm um tempo de vida muito maior do que os segundos, chegando a viver semanas depois de atuar em um local de inflamação (Pinchuk, 2002).

Através da detecção de uma menor quantidade de uma molécula chamada MHC de classe I (do inglês *Major Histocompatibility Complex*), as células matadoras naturais (NK) reconhecem principalmente células infectadas por vírus ou tumores e induzem a apoptose (morte celular programada). As moléculas MHC são fundamentais para a resposta imune adaptativa e são descritas na Seção 2.2.3.

Os mecanismos da imunidade inata estão envolvidos nas fases iniciais de uma infecção e podem ser eficazes na sua eliminação. Entretanto, alguns patógenos desenvolveram estratégias que lhes permitem, em algumas ocasiões, esquivarem-se ou dominarem os mecanismos da defesa imune inata e estabelecer um foco infeccioso a partir de onde eles podem se disseminar. Nessas circunstâncias, a resposta imune inata tem um papel crucial de controle dos principais aspectos da resposta adaptativa, através do reconhecimento dos microorganismos infecciosos e da indução dos sinais necessários para a ativação da resposta imune adaptativa. Os linfócitos antígeno-específicos da resposta imune adaptativa são ativados por moléculas co-estimuladoras que são induzidas nas células do sistema imune inato durante sua interação com os patógenos.

2.2.3 O Sistema Imune Adaptativo

Somente quando as defesas inatas do hospedeiro são sobrepujadas, evadidas, ou dominadas, é necessária uma resposta imune adaptativa ou induzida (Janeway et al., 2001). As principais células envolvidas na resposta imune adaptativa são os linfócitos B e T. Assim como todas as células do sangue, incluindo, portanto, as células do sistema imune, ambos surgem na medula óssea, mas apenas os linfócitos B ali se diferenciam, enquanto os linfócitos T migram para o timo para sofrer seu processo de amadurecimento. É o local de maturação que deu origem aos seus nomes: B, do inglês *bone marrow*, e T, de timo (em inglês *thymus*). Uma vez completada sua maturação celular, os dois tipos de linfócitos entram na corrente sanguínea, migrando para os órgãos linfóides periféricos, que são os linfonodos, o baço e os tecidos linfóides associados às mucosas, como as amígdalas, as placas de Peyer e o apêndice cecal. A medula óssea e o timo são considerados órgãos linfóides centrais (Janeway et al., 2001). A Figura 2.2 mostra a localização anatômica desses órgãos.

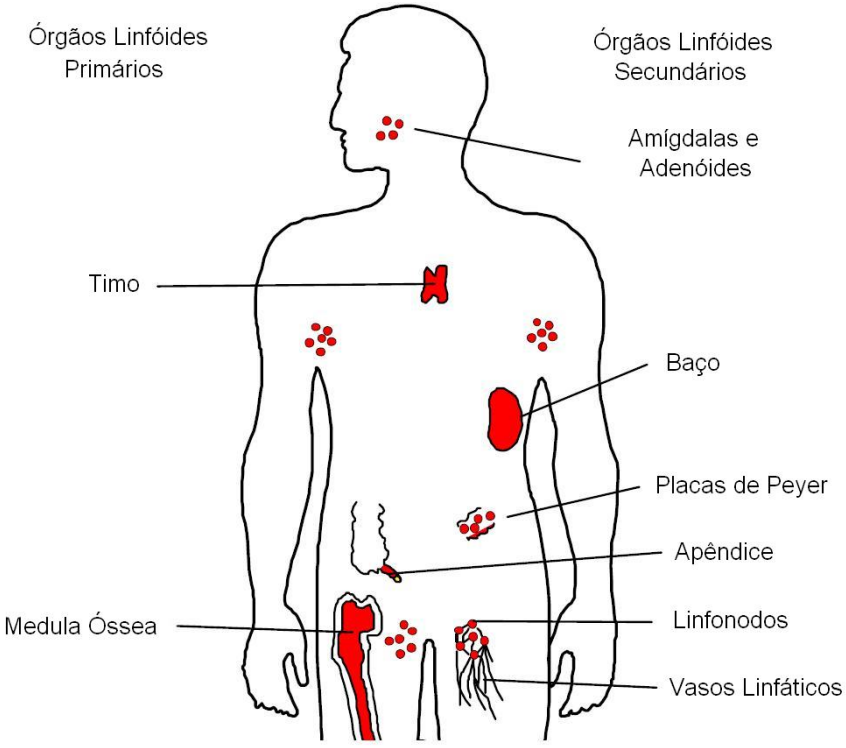


Fig. 2.2: **Órgãos linfóides no corpo humano. O timo e a medula óssea são os órgãos primários ou centrais. Os órgãos secundários ou periféricos são: o baço, as placas de Peyer, o apêndice, as amígdalas e os linfonodos. Figura extraída de (de Castro, 2001), com permissão do autor.**

A indução da resposta imune adaptativa começa quando um patógeno é ingerido por uma célula

dendrítica imatura, que apresenta receptores na sua superfície capazes de reconhecer estruturas comuns de vários patógenos. A função das células dendríticas, contudo, não é primeiramente destruir os patógenos, mas levar os antígenos patogênicos para órgãos linfóides periféricos e apresentá-los aos linfócitos T. Quando isso ocorre, diz-se que ela amadureceu em uma célula apresentadora de antígeno (APC - do inglês *Antigen-Presenting Cell*).

Outras células, além das dendríticas, podem desempenhar o papel de apresentadoras de antígenos, como os macrófagos (ver Seção 2.2.2) e os linfócitos B (de Castro e Timmis, 2002), e todas elas fazem isso através de uma molécula chamada MHC (do inglês *Major Histocompatibility Complex*), pois os receptores das células T, ou TCR (do inglês *T-cell receptor*), só são capazes de reconhecer antígenos quando unidos a uma molécula MHC. As células APC capturam proteínas antigênicas e processam-nas, ingerindo e digerindo o antígeno. Isso causa a fragmentação do antígeno em moléculas menores chamadas de peptídeos. São esses peptídeos que se ligam a uma molécula MHC, formando um complexo MHC/peptídeo que é então apresentado na superfície da célula (Figura 2.3).

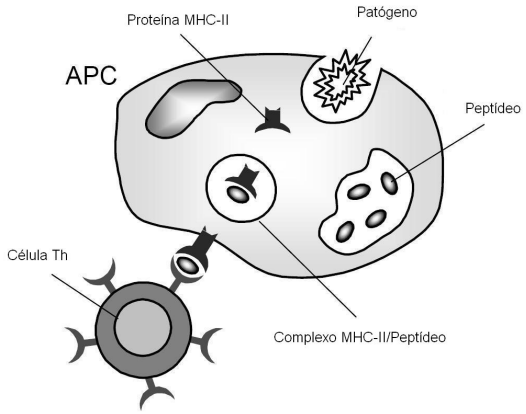


Fig. 2.3: **Célula APC captura um patógeno, fragmenta-o em peptídeos que se ligam a uma proteína MHC de classe II. O complexo formado por essa união é exposto na superfície da célula, onde pode ser reconhecido pelo linfócito T auxiliador. Figura extraída de (de Castro, 2001), com permissão do autor.**

Há duas classes de moléculas MHC. A MHC de classe I é encontrada em todas as células do corpo, enquanto a MHC de classe II é exclusiva das células do sistema imune. Há também dois tipos de linfócito T, os matadores Tk (do inglês *killer T-cells*) e os auxiliadores Th (do inglês *helper T-cells*). Os linfócitos T matadores identificam peptídeos antigênicos ligados à MHC de classe I. Já as células T auxiliadoras interagem com antígenos ligados à MHC de classe II. A MHC de classe I é especializada em apresentar patógenos intracelulares e a MHC de classe II é especializada em apresentar patógenos recolhidos do meio extracelular.

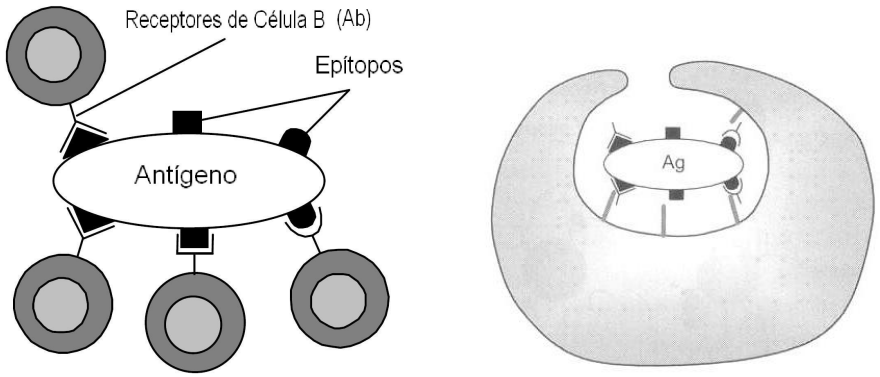
Os linfócitos Tk só são ativados quando identificam um complexo MHC/peptídeo juntamente com um sinal co-estimulante do sistema imune inato. Se uma célula Tk é ativada, ela destruirá a célula hospedeira infectada (por isso esse tipo de célula recebeu o nome de matadora) de uma das seguintes formas: ou ativando o mecanismo de apoptose (morte celular programada) da célula hospedeira, ou criando furos na membrana da célula, ou secretando moléculas tóxicas. Como todas as células do corpo produzem MHC de classe I, as células Tk podem identificar e eliminar qualquer célula do corpo que esteja infectada.

As células Th recebem esse nome, pois elas auxiliam a atividade das células APC através de um sinal co-estimulante, desencadeando a resposta imune na célula apresentadora. Por exemplo, macrófagos são estimulados a destruir o que estiver dentro de suas vesículas e células B serão estimuladas a se proliferar e diferenciar (Segel e Cohen, 2001). São essas células também as responsáveis pela tolerância ao próprio, ou seja, para que o sistema imune não seja ativado pelo próprio hospedeiro.

Assim como as células T, as células B também possuem receptores em sua superfície, chamados então de BCR (do inglês *B-cell receptor*). O reconhecimento de um antígeno ocorre quando esses receptores se ligam a uma molécula presente na superfície do antígeno, chamada epítopo. A intensidade com que essa ligação acontece é chamada de afinidade e, quando a afinidade do BCR com certo epítopo excede um certo limiar, a célula B fica ativa e ela secreta seus receptores na forma de anticorpos. Cada célula B produz um único tipo de anticorpo, por isso ela é dita monoespecífica. Os antígenos, no entanto, podem apresentar uma série de epítopos, ou seja, diferentes anticorpos podem reconhecer o mesmo antígeno (Figura 2.4(a)). Anticorpos ligados aos epítopos de um patógeno apresentam dois efeitos: primeiro, eles opsonizam o patógeno, sinalizando para outras células que elas podem ingerí-lo e processá-lo (Figura 2.4(b)); e segundo, eles neutralizam a atuação do patógeno, impedindo que esses se liguem às células do hospedeiro.

As células B estão relacionadas a três importantes propriedades do sistema imune: adaptação, memória e tolerância ao próprio. Para compreender esses conceitos, vamos detalhar melhor como essas células são capazes de identificar os mais diversos antígenos (epítopos) e como ocorre de fato a ativação dessas células.

Quando uma célula B é ativada, ela migra para um linfodo, onde a resposta imune adaptativa se desenvolve. No linfonodo, as células B ativadas produzem vários clones através de divisão celular. Entretanto, essa clonagem está sujeita a uma forma de mutação chamada de hipermutação somática, em que as taxas de mutação são muito maiores (até 9 ordens de grandeza maiores) do que as observadas na divisão celular comum. As taxas de variabilidade são inversamente proporcionais à sua afinidade ao antígeno em questão, aumentando as chances de que os clones apresentem estruturas de receptores diferentes do progenitor e, portanto, diferentes afinidades com o epítopo. As novas células B têm a oportunidade de se ligar aos epítopos presentes no linfonodo e, dentre as novas



(a) (b)

Fig. 2.4: **Identificação e eliminação de um antígeno. 2.4(a) Um antígeno com três diferentes epítopos, sendo reconhecido por receptores (anticorpos Ab) de três diferentes células B. Figura extraída de (de Castro, 2001), com permissão do autor. 2.4(b) Um fagócito ingerindo o antígeno marcado com anticorpos. Figura extraída de (de Castro, 2001), com permissão do autor.**

células geradas, as que possuem maior afinidade com o antígeno são selecionadas e as demais suprimidas, mecanismo esse conhecido como maturação de afinidade. Este processo de expansão clonal, hipermutação e seleção das células com receptores mais bem adaptados é denominado Seleção Clonal (Burnet, 1959, 1978). Pode-se dizer que o princípio da seleção clonal apresenta características semelhantes ao processo de evolução das espécies através da seleção natural, no qual as chances de sobrevivência e reprodução de indivíduos mais bem adaptados ao ambiente é maior do que a daqueles menos adaptados, favorecendo assim a evolução da espécie como um todo.

As células B selecionadas se diferenciam em células B plasmáticas, especializadas em secretar anticorpos, ou em células B de memória (Figura 2.5). Se o mesmo patógeno é encontrado no futuro, essa população já adaptada de células B de memória pode responder de forma muito mais rápida à invasão, pois o procedimento de maturação de afinidade desencadeado no primeiro encontro pode durar dias. Essa propriedade é explorada pelo processo de vacinação, em que o organismo é exposto ao antígeno atenuado, que não causa a doença em si, mas produz uma resposta imune, que gera células B de memória.

As células B são ativadas, na verdade, pela presença de dois sinais estimulantes: um deles ocorre quando elas identificam um patógeno (sinal I) e o outro é fornecido por uma célula Th (sinal II). Isso porque a célula Th precisa verificar o reconhecimento feito pela célula B. Como descrito anteriormente, a célula B apresenta o antígeno à célula Th, através de MHC de classe II. Se a célula Th se liga ao complexo MHC/peptídeo, ela fornece o sinal II, ativando a célula B (seleção positiva de células B). A Figura 2.6 resume esse processo.

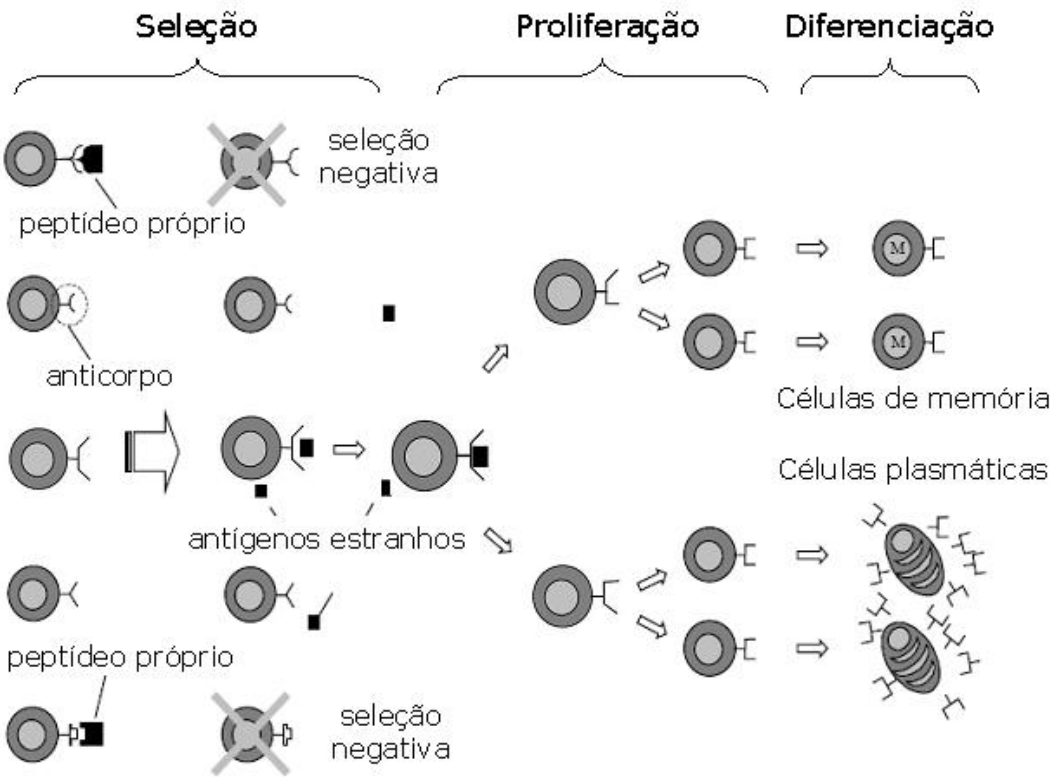


Fig. 2.5: **Princípio da Seleção Clonal. Células B que reconhecem o antígeno proliferam, através de clonagem com hipermutação somática, em que a taxa de variabilidade é inversamente proporcional à sua afinidade. Aquelas com maior afinidade ao antígeno sobrevivem e se diferenciam em células plasmáticas ou células de memória. Figura extraída de (de Castro, 2001), com permissão do autor, e adaptada.**

O amadurecimento dos linfócitos T no timo preserva aqueles que reconhecem uma molécula MHC do próprio hospedeiro (seleção positiva), enquanto elimina aquelas que reconhecem peptídeos próprios ligados a MHC própria (seleção negativa), o que torna as células eficientes em detectar agentes invasores e, ao mesmo tempo, tolerantes ao próprio. Portanto, a célula Th não produzirá o sinal II caso uma célula B reconheça o próprio hospedeiro. Se uma célula B recebe o sinal I sem a presença do sinal II, ela deve morrer. É esse mecanismo que permite o sistema imune ser tolerante ao próprio hospedeiro.

Foi visto até agora o que se pode considerar um modelo de sistema imune que discrima entre próprio e não-próprio. Há outras duas abordagens que serão salientadas. A Teoria do Perigo, proposta por Matzinger (Matzinger, 1994, 2002), sugere que o sistema imune é capaz de reconhecer sinais de *stress* ou danos ao organismo, ou seja, o sistema está mais preocupado em previnir a destruição do que em identificar invasores.

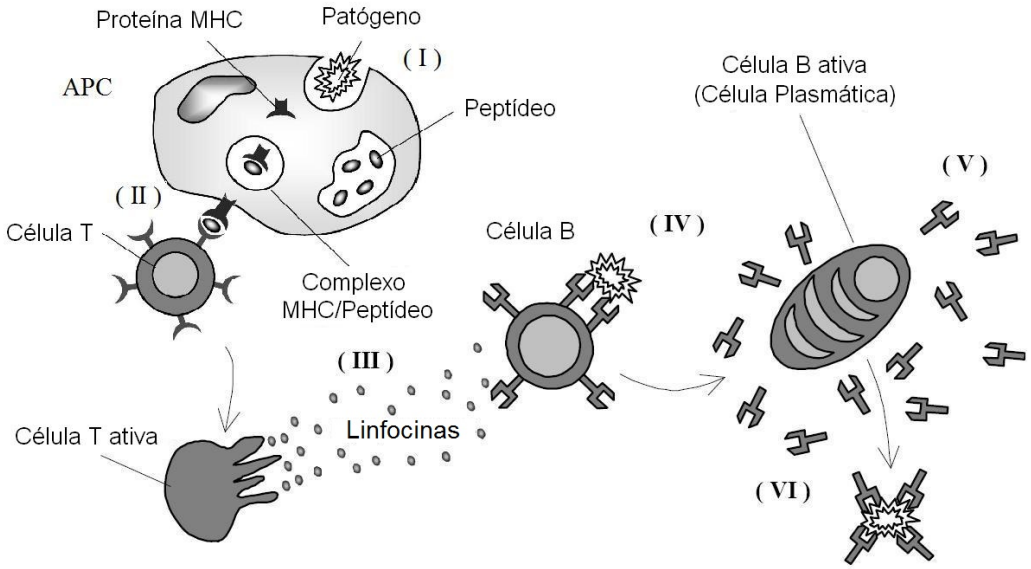


Fig. 2.6: **Uma célula APC reconhece um patógeno, ingere-o e digere-o (I). O patógeno é fragmentado em peptídeos que se ligam a uma molécula MHC. O complexo formado é apresentado a uma célula T (II). O reconhecimento do complexo ativa a célula T, que então libera sinais químicos que estimulam as células B (III). As células B reconhecem antígenos no meio (IV) e, uma vez ativadas, essas célula se diferenciam em células plasmáticas, que secretam anticorpos (V). Os anticorpos identificam e neutralizam os patógenos (VI). Figura extraída de (de Castro, 2001), com permissão do autor, e adaptada.**

Essa abordagem é fundamentada no fato de que, apesar de existirem muitas entidades não-próprias que são perigosas, como bactérias e vírus, há também células do próprio que são perigosas, como tumores, e entidades não-próprias que não são perigosas, como certas bactérias benéficas. A questão é como o sistema imune discrimina entre o que é perigoso e o que não é. Simplificadamente, essa teoria propõe que as células APC são estimuladas por sinais de alarme obtidos de células danificadas, desencadeando a resposta imune.

Outra teoria menos ortodoxa é a Teoria da Rede Imunológica, formalizada por Jerne (Jerne, 1974). Diferentemente da teoria da seleção clonal, que sugere um sistema imune composto por elementos (células e moléculas) discretos em repouso e que são estimulados por antígenos não-próprios, a teoria da rede imunológica propõe que as células e moléculas do sistema imune presentes no organismo são capazes de reconhecer não só antígenos, mas também umas às outras. Dessa forma, o sistema imunológico pode ser visto como uma enorme e complexa rede, onde cada componente reconhece e é reconhecido por outros elementos, e interfere e sofre interferência desses outros elementos. Esta relação entre componentes faz com que a rede imunológica atinja um estado de equilíbrio vinculado

à concentração dos seus componentes. Dessa forma, são alterações neste equilíbrio dinâmico que provocam as respostas imunes. Segundo essa óptica, as principais características do sistema imunológico, como aprendizado, memória, manutenção de diversidade, são consideradas propriedades emergentes, consequências dos mecanismos de regulação que mantém a rede em equilíbrio (de Castro e Timmis, 2002).

Apesar de muito popular após sua publicação, a teoria de Jerne se encontra em descrédito entre os imunologistas desde os anos de 1990, devido à falta de evidências empíricas que comprovem seus mecanismos de operação.

2.3 Sistemas Imunológicos Artificiais

Podem ser encontradas na literatura (de Castro, 2001; de Castro e Timmis, 2002; Dasgupta, 1998; Starlab, 2008; Timmis, 2000) muitas definições para sistemas imunológicos artificiais. Neste trabalho, é apresentada apenas a definição de de Castro e Timmis (2002), que buscou compilar outras definições em uma forma compacta e única:

Sistemas Imunológicos Artificiais (SIAs) são sistemas adaptativos, inspirados na imunologia teórica e em funções, princípios e modelos imunológicos, que são aplicados à resolução de problemas.

Em seu trabalho (de Castro e Timmis, 2002), os autores destacam que o termo imunologia teórica refere-se a todos os mecanismos, princípios, modelos e teorias matemáticas e não-matemáticas usados para descrever o funcionamento do sistema imune. Também deve ser ressaltado que a maioria dos sistemas imunológicos artificiais usa apenas algumas ideias do sistema imune, valendo-se de altos níveis de abstração.

Segundo um processo conhecido como engenharia imunológica (de Castro, 2001), para desenvolver sistemas imunológicos artificiais, é necessário definir os seguintes elementos básicos (de Castro, 2006; de Castro e Timmis, 2002): uma representação para os componentes do sistema; um conjunto de mecanismos para avaliar a interação desses componentes com o ambiente e uns com os outros; e procedimentos de adaptação que governem a dinâmica do sistema, ou seja, algoritmos imunológicos.

Segundo de Castro (2006), pode-se dividir os algoritmos imunológicos em cinco classes principais:

- Modelos de medula óssea;
- Algoritmo de seleção negativa;
- Algoritmo de seleção clonal;

- Modelos de rede imunológica contínua;
- Modelos de rede imunológica discreta.

Os modelos de medula óssea são usados para gerar populações de células e moléculas imunes para serem utilizadas em um SIA ou outras abordagens populacionais, como os algoritmos genéticos. Os algoritmos de seleção negativa são usados para definir um conjunto de detectores para desempenhar principalmente detecção de anomalia. Os algoritmos de seleção clonal são usados para gerar um repertório de células imunes que apresentam alta afinidade a padrões antigênicos. O algoritmo controla a expansão, a variação genética e a seleção das células. Os modelos de rede imunológica contínua são usados para simular uma rede imunológica dinâmica em um ambiente contínuo, enquanto os modelos de rede imunológica discreta são usados para um ambiente discreto.

Os sistemas imunológicos artificiais superam algumas das dificuldades encontradas em outras abordagens populacionais, destacando-se as seguintes vantagens:

- São inerentemente capazes de manter a diversidade da população;
- O tamanho da população a cada geração é automaticamente definido de acordo com a demanda da aplicação;
- Soluções ótimas locais tendem a ser simultaneamente preservadas, quando localizadas.

Apesar de existirem abordagens evolutivas que também apresentam essas características, é importante ressaltar que nos SIAs essas vantagens são fruto de propriedades inerentes ao funcionamento do sistema. Com essas propriedades, os SIAs apresentam uma vasta gama de aplicações, destacando-se as seguintes áreas: reconhecimento de padrões, análise de dados (classificação, agrupamento, quantização), busca e otimização e detecção de falhas e anomalias. Em de Castro e Timmis (2002), são apresentados exemplos de algoritmos e aplicações dessas e de outras áreas.

Para realizar uma tarefa de quantização vetorial, que é o objetivo deste trabalho, é estudado um algoritmo denominado ARIA (*Adaptive Radius Immune Algorithm*) (Bezerra et al., 2005), inspirado no sistema imunológico e que explora os conceitos do princípio da seleção clonal e da teoria da rede imunológica. Em quantização vetorial, deseja-se representar um conjunto de dados de entrada com um número reduzido de protótipos (ver Capítulo 3).

Interpretando os dados de entrada como antígenos e os protótipos como anticorpos, deseja-se um método em que os anticorpos identifiquem os antígenos da melhor maneira possível. Pode-se criar um mecanismo em que, dado um certo posicionamento de protótipos, eles se adaptem para melhor representar os dados e, caso seja necessário, que o número de protótipos aumente. Ora, pode-se

ver que há uma forte relação entre esse mecanismo e o processo de expansão clonal, hipermutação somática e seleção das células mais bem adaptadas, que caracteriza a seleção clonal.

Para evitar a proximidade excessiva de dois ou mais anticorpos, o que poderia levar a um crescimento indesejável de sua quantidade, pode-se medir tal proximidade e remover anticorpos que estejam muito próximos, segundo algum critério. Ao não permitir que os protótipos aproximem-se uns dos outros, pretende-se evitar um desperdício de recursos, pois tais protótipos estariam desempenhando quase o mesmo papel, ou seja, reconhecendo dados muito parecidos. Estabelecer uma comunicação entre elementos do sistema imunológico entre si, e não apenas deles com os antígenos, é o que foi proposto na teoria da rede imunológica para explicar a resposta imune, e pode servir de inspiração para criar um mecanismo de controle da distribuição e do número de protótipos.

Esses procedimentos constituem o esqueleto do algoritmo ARIA, que está descrito em detalhes na Seção 3.2.3.

Capítulo 3

Quantização Vetorial

3.1 Introdução

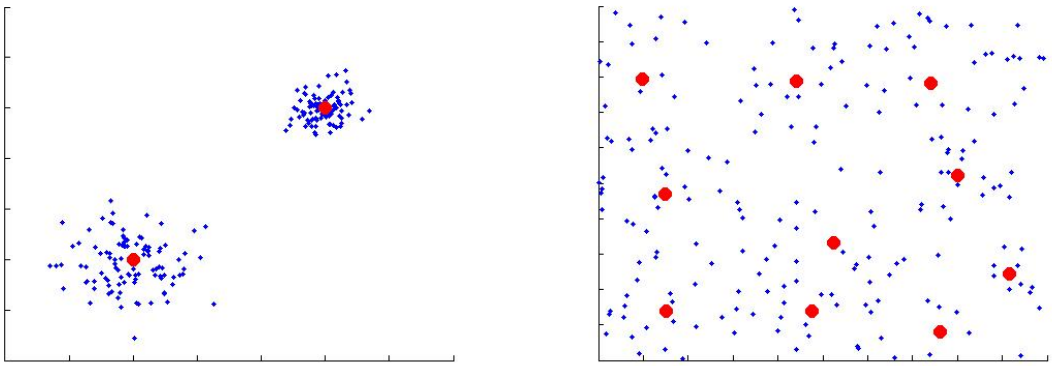
Dada a ampliação da nossa capacidade de gerar e armazenar informações, as bases de dados estão cada vez maiores e se tornam de difícil tratamento, caso não haja um procedimento automático para sua análise. Nesse sentido, técnicas de agrupamento de dados têm se tornado um tópico muito importante dentro de uma área mais ampla conhecida como mineração de dados.

Agrupamento de dados (ou clusterização) é o processo de agrupar um conjunto de dados em classes ou grupos (*clusters*), de forma que amostras do mesmo grupo apresentem alta similaridade entre si e tenham pouca similaridade com amostras de outros grupos (Duda et al., 2001). Dessa forma, um grupo é definido como uma coleção de dados que são similares aos dados do mesmo grupo e diferentes daqueles de outros grupos. Um conjunto de dados com essas características pode ser tratado coletivamente como uma entidade só e, portanto, o agrupamento de dados pode ser considerado como uma forma de compressão de dados.

Diferentemente do procedimento de classificação, que requer uma base de dados em que as amostras já tenham seus rótulos definidos, para servir de exemplos para o treinamento de um classificador, o agrupamento de dados não necessita de uma base de dados rotulada. Por isso, a classificação é considerada uma técnica de aprendizado supervisionado e o agrupamento de dados uma técnica de aprendizado não-supervisionado ou de aprendizado por observação.

Com o agrupamento de dados, podem-se identificar regiões densas e esparsas no espaço dos dados e, assim, descobrir padrões de distribuição gerais e relações interessantes entre os atributos dos dados (Han e Kamber, 2006). Por isso, essa área de pesquisa está presente nas mais diversas aplicações de todas as ciências (exatas, humanas e biológicas), incluindo pesquisa de mercado, reconhecimento de padrões, processamento de imagem, processamento de voz, prevenção de fraudes e análise genética.

Em quantização vetorial o objetivo é representar certa distribuição de dados utilizando um número



(a) (b)

Fig. 3.1: **Exemplos de posicionamento de protótipos para 3.1(a) agrupamento de dados e 3.1(b) quantização vetorial. Os protótipos obtidos pelo agrupamento poderiam ser utilizados para representar (quantizar) as amostras de tal grupo.**

de protótipos significativamente menor que o número de dados. O papel desses protótipos é aproximar estatisticamente o conjunto de dados original. Imagine que haja um conjunto de dados de entrada formado por N vetores de dimensão d. A quantização vetorial consiste em mapear esses dados em outro conjunto de $M < N$ protótipos, também de dimensão d (Simões et al., 2008). Daí advém o nome dessa técnica, pois, no processo de quantização, os vetores do conjunto de entrada, assim como qualquer outro novo dado (vetor), são associados a um protótipo, que é uma versão quantizada e, portanto, aproximada do vetor original.

Ao definir conjuntos de dados similares, as técnicas de agrupamento de dados podem ser utilizadas para quantização vetorial, de forma que todos os dados de um grupo passam a ser representados por um protótipo, por exemplo, o centroide do grupo. Na Figura 3.1, visualizam-se exemplos de posicionamento de protótipos para agrupamento de dados (Figura 3.1(a)) e para quantização vetorial (Figura 3.1(b)). Claramente, poder-se-iam utilizar os protótipos que caracterizam os grupos na Figura 3.1(a) para quantizar suas amostras, obtendo assim uma representação mais compacta dos dados.

A quantização vetorial é uma forma de compressão de dados, pois apenas os protótipos precisam ser armazenados, ao invés da base de dados inteira. O conjunto de protótipos divide o espaço dos dados em regiões que cada um representa (ver Figura 3.2), de forma que qualquer dado é mapeado pela região em que ele se encontra e apenas o índice dessa região é suficiente para quantizar o dado em um protótipo (Cherkassky e Mulier, 1998).

Há muitas técnicas para agrupamento de dados e podem-se encontrar na literatura classificações para essas técnicas. Por exemplo, em Han e Kamber (2006) elas são divididas em cinco categorias:

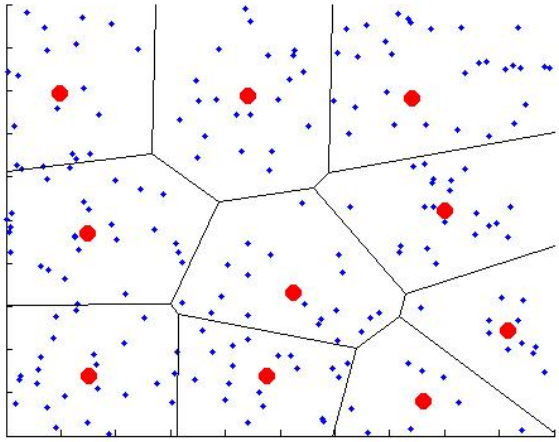


Fig. 3.2: **Partições obtidas pela quantização vetorial.**

métodos de partição, métodos hierárquicos, métodos baseados em densidade, métodos baseados em *grid* e métodos baseados em modelos. No entanto, o objetivo de todas elas é conceitualmente o mesmo: gerar partições dos dados. O que muda é apenas como essas partições são geradas e organizadas. Maiores detalhes (com exemplos de algoritmos e de aplicações de métodos das cinco categorias) podem ser encontrados em Han e Kamber (2006).

Os métodos de partição podem ser subdivididos em duas categorias: *hard clustering* e *fuzzy clustering*. Nos de partição rígida (*hard clustering*), cada amostra pertence a um e apenas um grupo. Já nas técnicas baseadas em partições nebulosas (*fuzzy clustering*), cada amostra pode estar associada a vários grupos, com um certo grau de pertinência a cada grupo. Uma restrição comum às duas categorias é que cada grupo deve conter pelo menos uma amostra.

Os métodos hierárquicos também podem ser subdivididos em duas categorias (Cherkassky e Mulier, 1998): aglomerativa (*bottom up*) ou divisiva (*top down*). Em um método hierárquico aglomerativo, inicialmente cada amostra é considerada um grupo e progressivamente os grupos são unidos, até que todas as amostras pertençam a um só grupo, chamado de nó raiz. Em um método hierárquico divisivo, inicialmente todas as amostras pertencem a um único grupo (nó raiz) e recursivamente os grupos pais são subdivididos em grupos filhos. Nos métodos hierárquicos, as relações entre os grupos resultantes são representadas por uma árvore ou um dendrograma.

A maioria dos métodos de partição agrupa as amostras baseando-se na distância entre elas. Por isso, esses métodos são eficientes apenas em encontrar grupos com formato hiper-esférico. Nos métodos baseados em densidade, os grupos são definidos por regiões com uma densidade de dados maior do que a observada em sua vizinhança. Tais métodos conseguem, portanto, identificar grupos

de qualquer formato.

3.2 Algoritmos Empregados em Quantização Vetorial

Nesta seção, são apresentados os três algoritmos para agrupamento de dados (ou quantização vetorial) que foram utilizados neste trabalho. O primeiro deles é o *k*-médias, uma das mais populares heurísticas usadas para agrupamento de dados ou quantização. Em seguida, é descrita uma rede neural, chamada de Neural-Gas (NG), desenvolvida para quantização vetorial. O outro algoritmo avaliado é um algoritmo imunológico (ARIA), que foi criado para agrupamento de dados, mas que, para realizar essa tarefa, primeiramente executa um processo de quantização.

Obviamente, existem inúmeros algoritmos para agrupamento de dados e inúmeros algoritmos para quantização vetorial, sendo que, como descrito anteriormente, um mesmo algoritmo pode ser empregado para qualquer uma dessas duas tarefas. O algoritmo ARIA é uma evolução da aiNet (de Castro e Von Zuben, 2001) e um dos objetivos deste trabalho é avaliar seu desempenho em uma aplicação específica, buscando identificar limitações em sua operação e propor modificações que aprimorem seu desempenho.

Para validar os resultados, decidiu-se compará-lo com o algoritmo padrão *k*-médias e com uma outra classe de algoritmo bio-inspirado, a rede neural NG. Inicialmente pretendia-se testar os mapas auto-organizáveis de Kohonen. No entanto, em teste preliminares, seu desempenho não foi satisfatório. O NG foi então escolhido por ter uma implementação simplificada disponível juntamente com o *toolbox* do SOM. O algoritmo *k*-médias foi escolhido por se tratar de um algoritmo de referência. Nas próximas seções, esses três algoritmo são descritos detalhadamente.

3.2.1 *k*-médias

Considere o seguinte problema de agrupamento de dados: dado um conjunto de n pontos no espaço d-dimensional (padrões de entrada) e um inteiro k, posicionar um conjunto de k pontos, chamados de centroides ou protótipos, também no espaço d-dimensional, de forma a minimizar a distância quadrática média de cada padrão ao protótipo mais próximo.

Não existe um algoritmo exato para resolver esse problema, mas várias heurísticas já foram propostas para a obtenção uma solução aproximada. Dentre essas heurísticas, a mais popular é o algoritmo de Lloyd (MacQueen, 1967; Lloyd, 1982), que ficou conhecido como algoritmo *k*-médias. O algoritmo *k*-médias funciona da seguinte forma:

N: número de padrões de entrada
k: número de protótipos
C: conjunto de padrões representado por cada protótipo

1 Inicialize os *k* protótipos;
2 **While** critério de convergência não for atingido **do**:
    2.1 **For** i = 1 **to** N **do**:
        2.1.1 Identifique o protótipo mais próximo do padrão de entrada i;
        2.1.2 Atualize C;
    end
    2.2 Reposicione cada protótipo no centroide do subconjunto de padrões associados a ele;
end

Os protótipos podem ser inicializados, por exemplo, amostrando aleatoriamente os padrões de entrada. O critério de convergência pode ser certo número pré-determinado de iterações ou então a ausência de modificações nos grupos.

O algoritmo *k*-médias funciona bem para problemas em que os grupos são compactos, bem separados e com formato hiper-esférico. Além de ser bastante simples, a complexidade computacional do *k*-médias é aproximadamente linear, o que torna esse algoritmo uma boa opção para problemas de larga escala (Xu e Wunsch II, 2008). No entanto, deve-se destacar também duas limitações do algoritmo: o número de grupos precisa ser definido previamente e o resultado do algoritmo depende da inicialização dos protótipos.

O primeiro problema pode ser resolvido executando-se o algoritmo com diferentes valores de *k* e utilizando algum índice de validação de agrupamentos para identificar qual *k* produziu a melhor solução. Para contornar o segundo problema, existem métodos baseados em algum tipo de busca local, capazes de fazer com que o algoritmo escape de mínimos locais (Kanungo et al., 2002, 2004). Obviamente essas soluções implicam em um aumento do custo computacional do procedimento completo.

Para uma aplicação de quantização vetorial, definir previamente o valor *k* geralmente não é um problema. Pelo contrário, quase sempre se conhece ou se deseja um certo número de protótipos, pois este número está diretamente relacionado com o fator de compressão da quantização. Além disso, os índices de validação de agrupamentos perdem seu sentido, já que o objetivo não é mais identificar grupos de dados, mas sim produzir uma representação mais compacta dos dados. Repare que a qualidade da quantização, ao menos em um possível sentido, está diretamente associada ao objetivo do *k*-médias: minimizar a distância quadrática média de cada dado ao protótipo mais próximo. A Seção 3.4 trata desse assunto.

3.2.2 NG - Neural-Gas

O algoritmo Neural-Gas (NG) é uma rede neural desenvolvida para compressão de dados através de quantização vetorial. Esse algoritmo pode ser visto como uma variação dos mapas auto-organizáveis de Kohonen (SOM - Self-Organizing Map) (Kohonen, 1982), em que se emprega uma rede mais flexível capaz de (*i*) quantizar conjuntos de dados topologicamente heterogêneos e (*ii*) identificar as similaridades entre os padrões de entrada sem a necessidade de predefinir a topologia da rede (Martinetz e Schulten, 1991; Martinetz et al., 1993).

No algoritmo NG, para cada padrão de entrada, os neurônios (protótipos) têm seus pesos sinápticos adaptados na direção desse padrão em função de sua distância a esse estímulo de entrada. Quanto mais distante, menor o passo de adaptação, segundo um *ranking* de distâncias de cada neurônio da rede ao padrão de entrada (o neurônio mais próximo é o primeiro e o mais distante é o último no *ranking*). Essa relação é determinada segundo uma queda exponencial, que, dessa forma, define a vizinhança que é influenciada a cada estímulo recebido.

Em seguida, é criada uma conexão entre os dois neurônios mais próximos desse padrão de entrada (que podem-se chamar vencedor e segundo colocado), caso ela não exista, e inicializa uma idade para essa conexão em zero. Caso a conexão já exista, sua idade é reinicializada em zero. Depois se incrementa a idade de todas as conexões ligadas ao neurônio vencedor. Por fim, as conexões que atingem uma idade superior a um certo limiar são removidas.

Então, repetem-se essas etapas para todos os padrões de entrada por um certo número de iterações, sendo que a cada padrão de entrada o passo máximo permitido e a vizinhança são reduzidos, levando à convergência dos neurônios. A forma como esses decaimentos ocorrem está descrita em detalhes na Seção 6.3. A cada iteração, a ordem em que os padrões de entrada são apresentados à rede é alterada, sendo sempre definida aleatoriamente. Resumidamente, o pseudo-código do algoritmo NG é o seguinte:

C: matriz de conexões
T: matriz de idade das conexões
W: matriz com os pesos sinápticos dos neurônios
N: número de dados de entrada
k: número de neurônios
max_{it}: número de iterações
$idade_{max}$: idade máxima permitida para as conexões

1 Inicialize $idade_{max}$, max_{it}, W, C e T

2 **For** it = 1 **to** max_{it} **do**:

2.1 Crie uma sequência aleatória $rand$ contendo os números de 1 a N, sem repetição;

2.2 **For** i = 1 **to** N **do**:

2.2.1 Ordene os k neurônios de acordo com sua distância ao padrão de entrada $rand(i)$;

2.2.2 Adapte os neurônios na direção do padrão de entrada, sendo que quanto mais distante menor o passo de adaptação (ver Equação 6.4);

2.2.3 Inicialize uma conexão entre o neurônio vencedor e o segundo melhor colocado;

2.2.4 Inicialize a idade dessa conexão em 0;

2.2.5 Incremente a idade de todas as conexões do neurônio vencedor;

2.2.6 Remova as conexões do neurônio vencedor que atingiram a idade máxima permitida;

2.2.7 Reduza o passo de adaptação máximo permitido e o "tamanho" da vizinhança (ver Equações 6.1, 6.2 e 6.3).

end

end

Uma implementação simplificada do NG pode ser encontrada no *toolbox* do SOM (SOM toolbox, 2008). Ela funciona da mesma forma do algoritmo original, mas não gera a matriz de conexões e, consequentemente, também não há matriz de idade das conexões [1], ou seja, não estão implementados os passos 2.2.3 a 2.2.6. Neste trabalho, esta implementação do *toolbox* foi utilizada.

Perceba que, diferentemente do SOM, em que a vizinhança é definida a priori por uma rede ligando os protótipos, no NG a influência da vizinhança é determinada pela distância no espaço dos dados.

3.2.3 ARIA - Adaptive Radius Immune Algorithm

O ARIA (Bezerra et al., 2005) é um algoritmo de agrupamento de dados pertencente à classe dos Sistemas Imunológicos Artificiais (SIAs) (de Castro e Timmis, 2002). Muitos algoritmos imunológicos para agrupamento de dados são baseados na redução da redundância presente nos dados, seguida do agrupamento propriamente dito (de Castro e Von Zuben, 2001; Timmis e Neal, 2001). A maioria desses algoritmos funciona posicionando um número reduzido de protótipos nas regiões mais

[1] Repare que a informação das matrizes C e T pode ser condensada em apenas uma matriz, na qual o valor zero indica que não há conexão entre os neurônios e valores positivos indicam a idade das conexões existentes. O pseudo-código apresentados aqui é baseado em (Martinetz e Schulten, 1991).

representativas dos dados, produzindo assim uma representação parcimoniosa desses dados (etapa de redução de redundância). Técnicas de partição podem então ser usadas para agrupar os protótipos resultantes, formando clusters (etapa de agrupamento).

A fase de redução de redundância tem, portanto, um papel muito importante nesse processo, pois é através dela que se busca eliminar o ruído presente nos dados e evidenciar as fronteiras entre os grupos, diminuindo a complexidade do problema. No entanto, deve-se destacar que a remoção inadequada de redundância pode ser prejudicial para a qualidade da solução, pois é justamente a redundância que provê o conhecimento (Stibor e Timmis, 2007). Afinal, um conjunto de dados só forma um grupo quando seus elementos compartilham atributos semelhantes e são, portanto, redundantes entre si.

O ARIA foi desenvolvido a partir do algoritmo aiNet (de Castro e Von Zuben, 2001), buscando superar algumas de suas limitações, principalmente em problemas em que (*i*) os clusters estão muito próximos uns dos outros, (*ii*) a densidade de dados varia de um cluster para outro e (*iii*) quando há sobreposição entre os cluster ou suas fronteiras não estão bem definidas (*fuzzy borders*). Nessas situações, dado que o posicionamento dos anticorpos (protótipos) gerado pela aiNet não leva em consideração a informação de densidade presente nos dados, as distâncias relativas entre os protótipos não corresponde à distância relativa entre os dados, o que pode levar a uma identificação incorreta dos clusters. O que diferencia o ARIA é justamente sua capacidade de preservar a densidade da distribuição dos dados, ou seja, este algoritmo consegue posicionar mais protótipos onde há maior concentração dos dados.

Para compreender o funcionamento do ARIA, considere os dados como sendo antígenos e os protótipos como sendo os anticorpos. O objetivo do algoritmo é produzir um conjunto de anticorpos que reconheça adequadamente os antígenos. Para isso, são aplicados o princípio da seleção clonal (Burnet, 1959) e a teoria da rede imunológica (Jerne, 1974), em um procedimento iterativo com três etapas principais (Bezerra et al., 2005):

1. *Maturação de Afinidade*: os antígenos (dados) são apresentados aos anticorpos (protótipos), que sofrem hipermutação para melhor reconhecer os antígenos (interação antígeno-anticorpo).

2. *Expansão Clonal*: os anticorpos que reconhecem antígenos a uma distância maior do que seu raio de atuação são clonados e a população aumenta.

3. *Supressão da Rede*: a interação dos anticorpos é quantificada e, caso um anticorpo reconheça outro, um deles é removido da população (interação anticorpo-anticorpo).

Para facilitar o entendimento do algoritmo, é apresentado primeiramente o seu pseudocódigo e, em seguida, será explicado cada um de seus passos.

Ag: antígenos
Ab: anticorpos
E: raio que define a vizinhança para o cálculo da densidade local
max_{it}: número de iterações
N: número de dados de entrada (antígenos)
n: tamanho inicial da população de anticorpos
R: vetor com o raio de cada anticorpo
r: raio mínimo empregado na atualização de R
μ: taxa de mutação
c: constante de decaimento geométrico da taxa de mutação

1 Inicialização de variáveis (Ab, E, n, R, max_{it}, r, μ, c)
2 **For** it = 1 **to** max_{it} **do**:
    2.1 **For** i = 1 **to** N **do**:
        2.1.1 Selecione o anticorpo Ab que melhor reconhece o antígeno Ag_i;
        2.1.2 Mute Ab com taxa μ na direção de Ag_i;
    end
    2.2 Elimine os $Ab's$ que não foram estimulados;
    2.3 Clone os $Ab's$ que reconhecem $Ag's$ localizados a
        uma distância maior do que seu raio;
    2.4 Calcule a densidade local para cada Ab;
    2.5 Atualize R;
    2.6 Aplique a supressão da rede de anticorpos;
    2.7 Atualize E;
    2.8 Reduza a taxa de mutação μ;
end

No Passo 1, os parâmetros do algoritmo são definidos. Além do número de iterações (max_{it}), o projetista deve definir a população inicial de anticorpos (Ab), seus raios iniciais (R), o raio inicial (E), o raio mínimo (r), a taxa de mutação inicial (μ) e a constante de decaimento ($c < 1$). Repare que há um número elevado de parâmetros que devem ser definidos a priori, o que é uma dificuldade comum em técnicas bio-inspiradas.

A operação do algoritmo está toda definida dentro do laço iterativo do Passo 2. O Passo 2.1 representa a etapa de maturação de afinidade. Os antígenos são apresentados um a um aos anticorpos e aquele com maior afinidade (menor distância euclidiana) é selecionado e sofre uma mutação com taxa μ_{it} na direção do antígeno, como descrito na Equação 3.1.

$$Ab'_i = Ab_i + \mu_{it}\ rand\ \ (Ag_j - Ab_i) \tag{3.1}$$

onde Ab_i é o anticorpo vencedor antes da mutação, Ab'_i após a mutação, Ag_j é o antígeno em questão, μ_{it} é a taxa de mutação e $rand$ é um número gerado aleatoriamente segundo uma distribuição uniforme entre 0 e 1.

Pode acontecer de um certo anticorpo não vencer para nenhum antígeno, ou seja, para nenhum dos antígenos ele é o anticorpo com maior afinidade. Esse anticorpo, portanto, não está contribuindo para o reconhecimento dos antígenos e deve ser eliminado para evitar o desperdício de recursos. Isso ocorre no Passo 2.2.

Os anticorpos possuem um raio de atuação inversamente proporcional à densidade local de dados na região do anticorpo. Caso a distância do anticorpo com maior afinidade a um certo antígeno seja maior do que seu raio (R_i) de atuação, esse anticorpo é escolhido para sofrer clonagem. Essa é a etapa de expansão clonal, descrita no Passo 2.3. Os clones são cópias dos anticorpos originais mutadas na direção do antígeno que desencadeou o processo de clonagem. Um único anticorpo pode reconhecer vários antígenos que satisfaçam essa condição, mas é permitido gerar apenas um clone por anticorpo. Com isso, espera-se que o crescimento da rede seja suave, tornando o processo de auto-organização da rede mais estável.

Em seguida, é calculada a densidade local de cada anticorpo (Passo 2.4). A densidade é definida como o número de dados na vizinhança do anticorpo, definida pelo raio E. Com os valores de densidade, calcula-se o raio (R_i) de atuação de cada anticorpo (Passo 2.5) segundo a Equação 3.2.

$$R_i = r\left(\frac{den_{\max}}{den_i}\right)^{\frac{1}{\dim}} \tag{3.2}$$

onde r é o raio mínimo, den_i é a densidade local de dados do anticorpo i, $den_{\max}$ é a maior densidade local de dados de um anticorpo na iteração e dim é a dimensão dos dados. Observe que um anticorpo posicionado na região mais densa, ou seja, que reconhece o maior número de antígenos a uma distância E, terá um raio de valor igual a r e todos os outros terão raio maior do que esse valor. Note também que essa fórmula não implica que o raio seja inversamente proporcional à densidade local, mas sim que o hipervolume de uma hiperespera o seja.

Por fim, ocorre a supressão da rede (Passo 2.6), em que o limiar de supressão é o próprio raio (R_i) de atuação dos anticorpos. Assim, caso a distância entre dois anticorpos seja menor do que o raio de atuação de um deles, aquele com maior raio é removido da rede. Calculando o valor médio dos raios de todos os anticorpos da rede no final de cada iteração, obtém-se o raio (E) que define a vizinhança para o cálculo de densidade local (Passo 2.7).

Por último, a taxa de mutação é reduzida segundo a fórmula dada pela Equação 3.3:

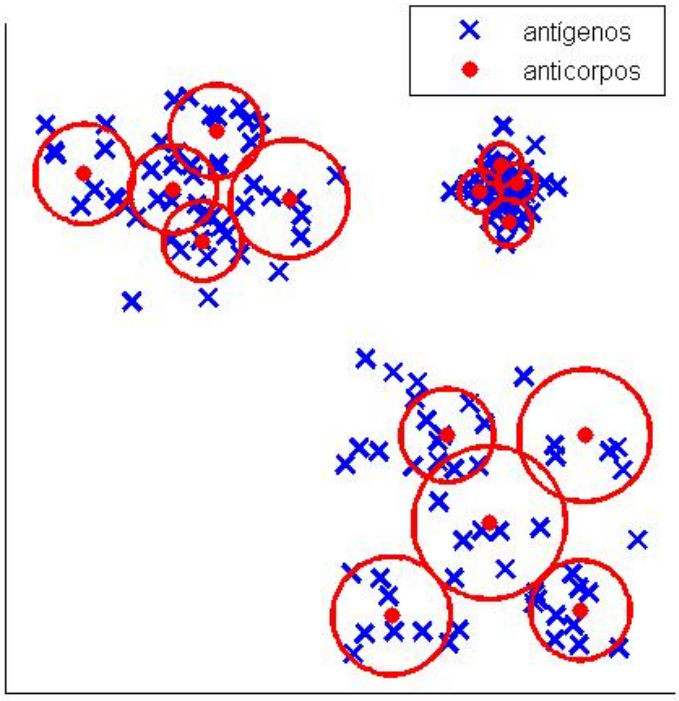


Fig. 3.3: **Exemplo do posicionamento e dos raios de atuação dos anticorpos.**

$$\mu_{it+1} = \mu_{it}\ c \tag{3.3}$$

Essa redução força a convergência da rede de anticorpos, pois a cada iteração eles se movimentarão menos e seus clones ficarão mais próximos a eles, forçando a supressão de um deles. Dessa forma o tamanho e o posicionamento da população se estabilizam.

O algoritmo consegue preservar a densidade local dos dados, permitindo que os anticorpos posicionados nas regiões mais densas fiquem mais próximos, pois possuem um raio menor. Já nas regiões mais esparsas, seus raios tendem a ser maiores e a distribuição de anticorpos tende a ser esparsa também. A Figura 3.3 mostra um exemplo em duas dimensões do posicionamento e dos raios de atuação dos anticorpos obtidos ao final da execução do algoritmo para um conjunto de dados gerados aleatoriamente a partir de três distribuições de probabilidade gaussianas com variâncias diferentes, produzindo 50 pontos cada uma.

Repare que o algoritmo foi capaz de gerar um posicionamento adequado dos protótipos e que os

raios [2] são inversamente proporcionais à densidade local dos dados, como era esperado.

3.3 Tabela Comparativa

Aqui é apresentada uma tabela comparando os algoritmos qualitativamente.

Tab. 3.1: **Comparação entre os algoritmos *k*-médias, Neural-Gas e ARIA.**

	k-médias	NG	ARIA
Número de protótipos	fixo e definido a priori	fixo e definido a priori	auto-ajustável
Sensível à densidade	não	não	sim
Bio-inspirado	não	sim	sim
Sensibilidade à inicialização	forte	média	fraca
Custo Computacional	baixo	médio	alto
Sensibilidade a mínimos locais	forte	média	fraca

3.4 Formas de Avaliar a Qualidade da Quantização

A quantização vetorial é o processo de mapear dados em protótipos. Essa aproximação introduz um erro na representação dos dados de entrada, denominado erro de quantização vetorial (Gray e Neuhoff, 1998).

O erro de quantização de uma amostra de entrada n_i é dado pela distância entre tal amostra e o protótipo m_j que a representa, definido por $m_j = q(n_i)$, como mostra a Equação 3.4.

$$Q_i = d(n_i, q(n_i)) \tag{3.4}$$

onde $d(\cdot, \cdot)$ é uma métrica que fornece uma medida da distância entre dois vetores, por exemplo, a distância euclidiana [3].

Para avaliar o resultado de um algoritmo de quantização, pode-se empregar o erro de quantização médio Q_N (Equação 3.5), ou seja, a média do erro de quantização individual de cada amostra n_i, tomadas todas as N amostras de entrada.

$$Q_N = \frac{1}{N}\sum_{i=1}^{N} Q_i = \frac{1}{N}\sum_{i=1}^{N} d(n_i, q(n_i)) \tag{3.5}$$

[2] Formalmente, para um caso em duas dimensões, as áreas dos círculos são inversamente proporcionais à densidade local dos dados e não os raios.

[3] O algoritmo *k*-médias, ao posicionar protótipos no centroide do conjunto de dados que eles representam, minimiza, de fato, a distância euclidiana ao quadrado.

Pode-se afirmar que o erro de quantização é a forma mais simples de avaliar a qualidade da quantização vetorial. Contudo, há situações em que o erro de quantização pode não ser uma medida adequada da qualidade da quatização.

Suponha, por exemplo, um cenário simples em que há dois grupos de dados bem distintos, com o mesmo número de amostras, gerados a partir da amostragem de duas funções gaussianas, com médias e variâncias diferentes, como mostra a Figura 3.4

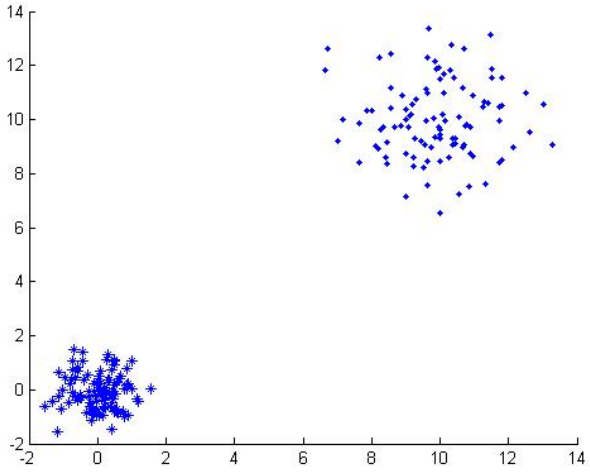


Fig. 3.4: **Dois grupos de dados bem distintos, com 100 amostras cada, gerados a partir da amostragem de duas funções gaussianas, sendo uma delas com média 0 e variância 0,5 e a outra com média 10 e variância 2,0.**

Uma distribuição de protótipos, que busque minimizar o erro de quantização, posicionaria mais protótipos sobre o grupo menos denso (produzido pela gaussiana com variância maior), onde as distâncias são naturalmente maiores.

No entanto, como há o mesmo número de amostras nos dois grupos, poderia ser interessante posicionar também o mesmo número de protótipos nas regiões definidas por cada grupo. Dessa forma, se considerarmos a similaridade entre a distribuição dos dados e a dos protótipos, seria obtida uma solução melhor, pelo menos no sentido de número de protótipos por amostra, junto a cada grupo. Existem, portanto, dois objetivos diferentes e muitas vezes conflitantes (Azzolini et al., 2010): minimizar o erro de quantização ou maximizar a similaridade entre as distribuições.

Para avaliar tal similaridade, outra medida de qualidade da quantização é sugerida: a entropia relativa, ou divergência de Kullback-Leibler (Kullback, 1959). Suponha que se conhecem as funções densidade de probabilidade $p_N(x)$ e $p_M(x)$, das quais teriam sido amostrados, respectivamente, os N dados de entrada e os M protótipos. Então, é possível medir a dissimilaridade $H(N, M)$ entre essas

duas distribuições usando a entropia relativa (Fukunaga e Hayes, 1989), dada na Equação 3.6.

$$H(N, M) = \int \ln \left[\frac{p_N(x)}{p_M(x)} \right] p_N(x)\, dx \tag{3.6}$$

Quanto mais distintas forem as distribuições, maior será o valor de $H(N, M)$, sendo 0 o resultado quando as duas distribuições são iguais. A Equação 3.6 da entropia pode ser re-escrita como $E\{\ln[p_M(x)/p_N(x)]\}$, onde a esperança é tomada em relação a $p_N(x)$. Substituindo, então, a esperança pela média amostral, chega-se a uma aproximação para a entropia relativa, mostrada na Equação 3.7 (Fukunaga e Hayes, 1989).

$$\hat{H}(N, M) = \frac{1}{N} \sum_{i=1}^{N} \ln \left[\frac{p_N(x_i)}{p_M(x_i)} \right] \tag{3.7}$$

Entretanto, geralmente as funções $p_N(x)$ e $p_M(x)$ não são conhecidas e, portanto, precisam ser estimadas. Há algumas maneiras de estimar funções densidade de probabilidade, das quais duas serão apresentadas (Silverman, 1986): o método KNN (do inglês *k-nearest neighbour*) e o método do estimador de núcleo (do inglês *kernel estimator*). Segundo o método KNN, a estimativa é feita da seguinte forma:

$$\hat{p}(x) = \frac{k}{[d_k(x)]^d} \tag{3.8}$$

onde $d_k(x)$ é a distância entre x e seu k-ésimo vizinho mais próximo e d é a dimensão do espaço. Com isso, obtém-se uma medida de número de amostras em um certo volume, de tal forma que o parâmetro k define o tamanho da vizinhaça que é empregada nesta estimativa. O estimador de núcleo é definido por:

$$\hat{p}(x) = \frac{1}{N} \sum_{i=1}^{N} K(x - x_i) \tag{3.9}$$

no qual a função de núcleo K deve satisfazer a seguinte condição:

$$\int_{-\infty}^{\infty} K(\mathbf{x}) d\mathbf{x} = 1 \tag{3.10}$$

Esse método já foi utilizado para avaliar a qualidade da compressão do algoritmo aiNet, predecessor do ARIA, por Stibor e Timmis (2007). Em seu trabalho, os autores empregaram como função de núcleo a função gaussiana multivariada, dada pela Equação 3.11.

$$\mathrm{K}(\mathbf{x}) = \frac{1}{(2\pi)^{l/2} h^l} \exp\left(-\frac{\|\mathbf{x}\|^2}{2h^2} \right) \tag{3.11}$$

A chamada largura de banda h controla o tamanho da vizinhança, enquanto a função de núcleo determina o formato dessa influência.

Resumindo, a medida de qualidade chamada de entropia relativa é dada por:

$$\hat{H}(N, M) = \frac{1}{N} \sum_{i=1}^{N} \ln \left[\frac{\hat{p}_N(x_i)}{\hat{p}_M(x_i)} \right] \tag{3.12}$$

Essa medida atende ao critério de posicionamento de protótipos adotado pelo ARIA, baseado no conceito de preservação de densidade, segundo o qual deseja-se posicionar mais protótipos onde há mais dados (ou, no caso do exemplo, o mesmo número de protótipos onde há o mesmo número de dados).

Tanto o erro de quantização quanto a entropia relativa serão utilizados para avaliar o desempenho dos algoritmos de quantização descritos na Seção 3.2. É importante ressaltar que a medida mais adequada a se usar depende da aplicação em que a quantização está inserida. Por exemplo, na aplicação abordada neste trabalho, em que sinais de fala serão quantizados, pode-se medir a qualidade da quantização avaliando-se a qualidade do sinal de fala produzido (ver Capítulo 5). No entanto, os algoritmos de quantização operam no espaço dos dados que são fornecidos a eles e, portanto, deve-se saber que critério perseguir neste espaço, para que a qualidade da aplicação seja a melhor possível. Em outras palavras, deve-se buscar uma medida de qualidade da quantização que seja correlacionada com a medida de qualidade da aplicação.

Capítulo 4

Síntese de Fala

4.1 Introdução

As chamadas ciências da fala englobam diversas abordagens do estudo da fala, abrangendo áreas como a engenharia, a física, a linguística, a psicologia experimental e cognitiva, a fisiologia da fala e a informática (Simões, 1999).

No que diz respeito à engenharia, há várias frentes de estudo no que se passa a chamar de processamento de fala. O conceito de processamento de fala esteve primeiramente relacionado quase sempre à codificação de fala, que estuda meios eficientes para transmitir, armazenar e parametrizar o sinal de fala. Com o avanço tecnológico, outras áreas de estudo surgiram, das quais destacam-se (*i*) o reconhecimento de fala, que a partir de um sinal de fala busca obter a descrição textual do que foi falado, (*ii*) a síntese de fala, que utiliza mecanismos artificiais para a produção de um sinal de fala, e (*iii*) o reconhecimento de locutor, que pretende identificar, a partir de um sinal de fala, quem o pronunciou, dentre outras áreas (reconhecimento de língua, sistemas de tradução automática etc).

A conversão texto-fala pode ser vista como um caso particular da síntese de fala, em que o sinal de fala é produzido a partir de um texto. Note que se pode dizer que a conversão texto-fala realiza a operação inversa do reconhecimento de fala.

Este capítulo tem por objetivo introduzir os sistemas de conversão texto-fala, mais especificamente a etapa de síntese de fala baseada na concateção de trechos de sinais de fala. Cabe mencionar que a técnica a ser proposta no Capítulo 5 pode ser utilizada para compressão da base de fala utilizada em tais sistemas. Além disso, vendo a situação por outro ângulo, pode-se afirmar que o sistema descrito no Capítulo 5 para a codificação de sinais de fala está baseado em síntese concatenativa, que, portanto, merece ser detalhada. O Capítulo 4 está organizado da seguinte forma: a Seção 4.2 é dedicada a uma breve introdução à conversão texto-fala e a Seção 4.3 trata da última etapa da conversão, a síntese de fala.

4.2 Aspectos Gerais da Conversão Texto-Fala

Os conversores texto-fala, também conhecidos como TTS (*Text-To-Speech*), são sistemas que produzem fala sintética correspondente à leitura de um texto (Latsch, 2005). De maneira geral, a tarefa da síntese de fala a partir de texto pode ser dividida em duas etapas distintas, realizadas em sequência: a primeira etapa, correspondente à análise do texto, consiste em obter a representação fonológica da mensagem a partir de sua forma ortográfica; a etapa de síntese, por sua vez, é responsável pela geração do sinal acústico associado à representação fonológica obtida na etapa anterior (Simões, 1999). O diagrama da Figura 4.1 mostra de forma simplificada os passos necessários para executar a conversão.

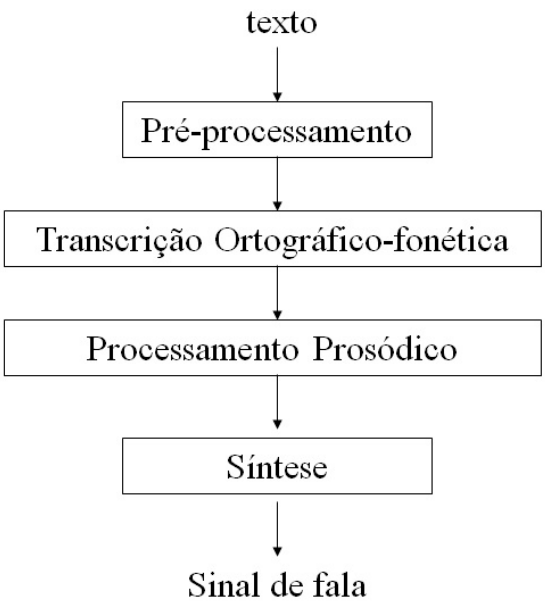


Fig. 4.1: **Passos da conversão texto-fala.**

A etapa de pré-processamento busca a formatação do texto para representar em sua forma textual dígitos, siglas, abreviaturas e símbolos especiais. A partir do texto formatado, executa-se a transcrição ortográfico-fonética. Essa etapa consiste em encontrar a sequência correta de fonemas que representa cada uma das palavras contidas no texto. Vários são os desafios dessa etapa, pois muitas letras, por exemplo, tem pronúncias variadas (observe a letra *x* das seguintes palavras: tórax, xícara, exame, próximo), muitas palavras de mesma ortografia podem pertencer a classes gramaticais diferentes (o piloto / eu piloto), com pronúncias diferentes, e há também casos de palavras de mesma ortografia e classe gramatical, mas com pronúncias diferentes dependendo do contexto (por exemplo, a palavra sede, que se pronuncia *sêde* ou *séde* dependendo do contexto).

Mas a representação fonológica isolada não é suficiente para a obtenção de uma fala sintetizada

de qualidade. Deve-se ainda determinar o acento lexical das palavras, o ritmo e a entonação da frase e a duração das pausas. Estes são chamados de parâmetros prosódicos e permitem a geração de fala de forma mais natural.

A última etapa do processo de conversão texto-fala é a síntese do sinal propriamente dita, que será tratada com mais detalhes na próxima seção. O papel do módulo de síntese consiste em obter o sinal acústico a partir da representação fonético-prosódica obtida nas etapas anteriores (Simões, 1999).

4.3 Síntese de Fala

Os sistemas de síntese de fala podem ser divididos em quatro grandes grupos: a síntese por regras, a síntese articulatória, a síntese paramétrica e a síntese concatenativa.

O princípio de funcionamento do sintetizador por regras é baseado no modelo fonte-filtro da teoria acústica de produção da fala. Segundo esse modelo, o sinal de fala produzido pelo aparelho fonador humano corresponde ao resultado da passagem de uma fonte de excitação (que pode ser sonora, não sonora ou mista) por um filtro, cuja função de transferência é determinada pela configuração instantânea do trato vocal. Ao fornecer um modelo adequado da fonte de excitação ao sintetizador, pode-se supor que ele é capaz de produzir sinal de fala na sua saída, desde que o modelo seja capaz de simular a função de transferência do trato vocal humano. Em outras palavras, a qualidade do sinal sintético gerado depende do modelamento correto do processo de filtragem e também da fonte de excitação (Simões, 1999). O primeiro sintetizador de sucesso baseado nessa abordagem foi proposto por Klatt (1980), e foi aperfeiçoado nos anos seguintes.

O funcionamento de um sistema de síntese articulatória baseia-se na construção de um modelo físico o mais realista possível do aparelho fonador humano, capaz de mimetizar a dinâmica dos diversos articuladores no processo de produção da fala. As posições desses articuladores (língua, mandíbula, lábios, osso hióide, véu palatino etc.) correspondem às variáveis do modelo. Pode-se afirmar que essa abordagem foi a que obteve menos sucesso em aplicações reais, estando até hoje restrita ao ambiente acadêmico.

Recentemente, uma abordagem que vem se destacando é a síntese paramétrica. Essa técnica utiliza uma base de fala gravada para treinar modelos estatísticos, geralmente utilizando HMM (*Hidden Markov Models*), os quais geram parâmetros de fala, que por sua vez são utilizados para a síntese de um sinal. Essa abordagem gera fala sintética de boa qualidade, mas ainda inferior à obtida pela técnica de síntese concatenativa, que será descrita a seguir. Apesar disso, a chamada síntese HMM tem recebido bastante atenção da comunidade científica da área, pois ela apresenta duas vantagens principais em relação à síntese concatenativa (Benesty et al., 2008). Primeiro, ela necessita de muito menos memória e, segundo, com essa técnica é mais fácil realizar as tarefas de modificação e transformação

de voz, que constituem outro tema de pesquisa atual.

Atualmente, a maioria dos sistemas de síntese de fala que buscam alta qualidade ainda utiliza a técnica de síntese concatenativa. Na síntese concatenativa, o sinal de fala é gerado através da concatenação de trechos gravados de fala, como será detalhado na Seção 4.3.1.

Embora a conversão texto-fala forneça uma boa ilustração de sua utilidade, síntese de fala não está restrita a esse universo. A maioria dos sistemas atuais de comunicações celulares ou pela Internet utiliza alguma técnica de síntese para a codificação e transmissão do sinal de fala, diferentemente da telefonia fixa clássica, em que a própria forma de onda do sinal de fala era transmitida, apenas com uma limitação de frequência (até 3.4 kHz). O objetivo dessas técnicas é quase sempre a redução da banda de transmissão, conferindo certa robustez às condições de cada sistema específico e preservando a naturalidade da fala.

4.3.1 Síntese Concatenativa de Fala

A ideia por trás da síntese concatenativa é gerar um sinal de fala artificial a partir da concatenação de segmentos gravados de fala natural. Tais segmentos devem ser selecionados a partir de um inventário de unidades previamente construído, e o conteúdo desse inventário deve ser tal que seja possível sintetizar todas as sequências fonéticas possíveis de serem realizadas dentro de uma determinada língua (Simões, 1999).

Para tanto, é necessário decidir quais serão as unidades básicas utilizadas para concatenação. As unidades podem variar desde simples quadros, passando por fones (o conceito de fone será definido no Capítulo 5), difones, trifones, polifones, sílabas, palavras e até mesmo conjuntos de palavras. Essa decisão define como será o banco de gravações contendo as unidades desejadas. Obviamente, quanto maiores forem as unidades concatenadas, menor será o número de concatenações necessárias e mais natural será a fala sintetizada. Infelizmente, supondo que será armazenada uma realização de cada unidade, a utilização de unidades maiores implica em um maior número de unidades no inventário (por exemplo, uma língua é composta geralmente de algumas dezenas de fones, mas o número de palavras é imenso), o que aumenta seu custo de geração e armazenamento.

A princípio, pode parecer interessante utilizar fones como os blocos constituintes básicos, pois eles são poucos e com eles seria possível recriar qualquer sequência de sons. No entanto, essa solução leva a sinais de qualidade muito ruim, muitas vezes sequer inteligíveis. Isso porque as características de um fone são fortemente influenciadas pelos fones adjacentes (contexto fonético), um fenômeno conhecido como co-articulação.

Uma escolha mais razoável são os difones. Um difone é um segmento de fala que se inicia no meio de um certo fone e termina no meio do fone seguinte. Com isso, pretende-se incluir os efeitos da co-articulação dentro da unidade, evitando a ocorrência de descontinuidades nas concatenações.

Ainda assim, há sons em que os efeitos da co-articulação se estendem por mais do que um fone ou que tem uma forte característica dinâmica, tal que não há uma região boa para se fazer o corte. Por isso, a utilização de difones também leva a sinais degradados. Nesses casos é interessante utilizar unidades maiores, como trifones ou sílabas.

A utilização de unidades mistas, chamadas genericamente de polifones (como difones, trifones e sílabas), é aplicada até os dias de hoje em sistemas de síntese comerciais. A ideia é manter o tamanho do inventário de unidades pequeno, mas cobrir a maior parte dos sons da língua, respeitando sua dinâmica e incluindo seus efeitos de co-articulação. Essa técnica é capaz de gerar um sinal de fala inteligível, mas de pouca naturalidade.

A síntese a partir de um inventário fixo, ou seja, em que está armazenada apenas uma instância de cada possível unidade (como as técnicas baseadas em difones ou polifones), apresenta dois problemas principais (Benesty et al., 2008). Primeiro, essa estratégia leva à gravação de unidades hiper-articuladas, capazes de serem utilizadas na maioria dos contextos, mas que portanto não é específica a nenhum deles. Segundo, são necessárias técnicas de processamento de sinais para adaptar tais unidades para cada caso e esse processamento também causa alguma degradação.

Gerar um sinal de fala sintética de ótima qualidade (natural e inteligível) só foi possível a partir da síntese baseada em seleção de unidades, em que são armazenadas várias instâncias de cada unidade. Em tempo de execução, dada uma sequência fonético-prosódica alvo, um algoritmo de seleção escolhe, a partir de uma base de gravações extensa, a melhor sequência de unidades acústicas para representá-la. As unidades escolhidas ainda podem ser difones, apesar de fonemas poderem ser utilizados como alternativa para o caso em que o difone com as características desejadas não está disponível. Também é possível que palavras ou até mesmo frases inteiras possam ser utilizadas, diminuindo o número de concatenações.

Para que tais sistemas produzam fala sintética de qualidade satisfatória (inteligibilidade e naturalidade próximas às da fala humana), é necessário que o banco de gravações contenha diversos exemplos de um grande número de contextos fonéticos (Hentz e Seara, 2009). Para obter essa diversidade, geralmente esses bancos contêm algumas horas de gravações, ou até mesmo dezenas de horas, o que implica em centenas de megabytes de memória ocupados.

Uma vez gravados, os sinais de fala que comporão o banco precisam ser processados, em um procedimento chamado de segmentação e transcrição fonética, no qual o sinal de fala será divido em unidades às quais será associado o devido fonema. Essa etapa pode ser automatizada, mas quase sempre é feita uma revisão manual por um especialista.

A Figura 4.2 ilustra o processo de conversão texto-fala baseado em síntese concatenativa.

Apesar do recente avanço das tecnologias de armazenamento de dados, que permitiram reduzir seus custos a níveis muito baixos, há situações em que o tamanho da base de gravações pode ser um

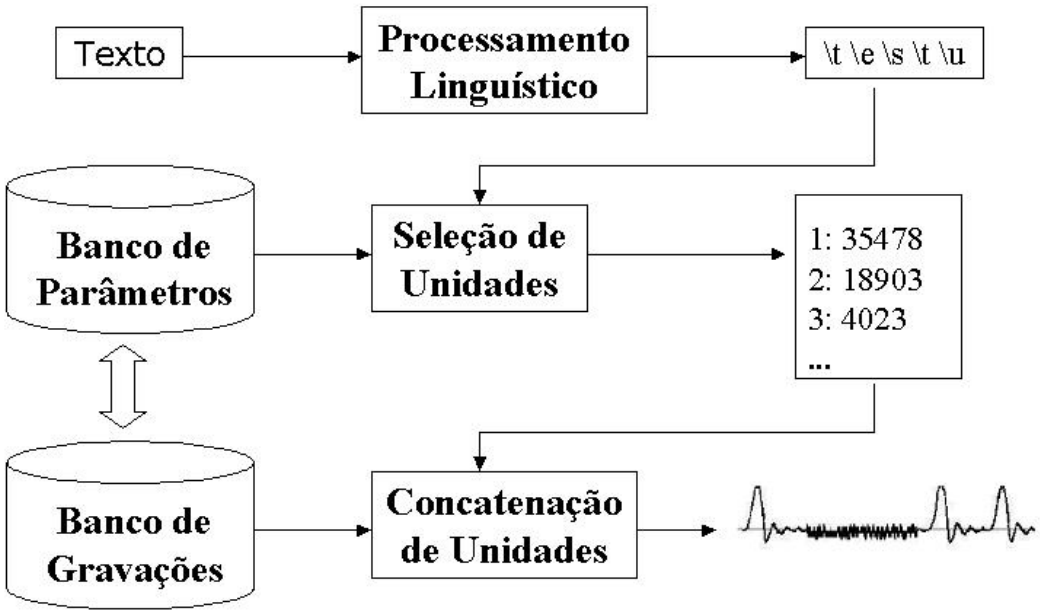


Fig. 4.2: **Etapas da conversão texto-fala baseada em síntese concatenativa.**

problema, como em dispositivos móveis (celulares, *smartphones* e PDA's), receptores de TV digital e em qualquer situação em que é necessária a transmissão da base (por exemplo, para instalação remota de sistemas de conversão texto-fala). Neste trabalho, é implementado um sistema de quantização vetorial que pode ser empregado para a compressão de bases de sinais de fala gravados.

Além de bases de fala empregadas em sistemas de conversão texto-fala baseados em síntese concatenativa por seleção de unidades, há outras aplicações que requerem grande espaço para armazenamento de sinais de fala pré-gravados. Um *talking book*, que é uma versão falada equivalente à versão escrita de um livro, por exemplo. Outras aplicações, como um sistema de assistência ao aprendizado de língua, dicionários eletrônicos e enciclopédias eletrônicas, são aplicações potenciais para o sistema avaliado neste trabalho, pois também necessitam armazenar grandes quantidades de sinais de fala (Lee e Cox, 2001) ou podem estar associados a um sistema de conversão TTS. Além disso, cabe destacar que os testes realizados neste trabalho foram feitos a partir do conteúdo parcial de uma base de gravações desenvolvida para um sistema TTS comercial.

Capítulo 5

Codificação de Fala

5.1 Introdução

Recentemente, viu-se uma expansão dos computadores e dos sistemas de comunicação digitais, tornando o uso de sinais de fala digitalizados cada vez mais comum. Em tais sistemas, o sinal de fala é, portanto, representado por uma sequência de bits. A maior vantagem dessa representação binária é que a informação pode ser recuperada perfeitamente (sem distorção) após atravessar um canal ruidoso, além de não perder sua qualidade ao passar por diversas partes de diferentes sistemas de transmissão [1].

No entanto, uma representação digital do sinal de fala sempre vai apresentar erros de quantização, os quais são mais reduzidos quanto maior o número de bits, fazendo com que tais sistemas fiquem complexos e caros. Considerando, por exemplo, uma taxa de amostragem de 8 kHz, e que cada amostra tenha uma precisão de 16 bits (o que pode ser considerada uma precisão suficiente para a quantização adequada das amostras de voz), seria necessário uma taxa de: (8000 amostras/segundo) * (16 bits/amostras) = 128kb/s. Isso encorajou o desenvolvimento de diversos métodos de codificação de fala, buscando formas mais eficientes para a transmissão e o armazenamento de sinais de fala digitais.

Historicamente, a tecnologia de codificação de fala foi dominada por codificadores baseados em predição linear. Para obter um sinal de qualidade boa, a maioria desses codificadores pode ser definida como "aproximadores" de forma de onda. Dentre estes, os codificadores CELP (*code-excited linear prediction*) (Schoroeder e Atal, 1985) e suas variações - VSELP (Gerson e Jasiuk, 1990), LD-CELP (Chen, 1989; Chen et al., 1992), ACELP (Adoul et al., 1987; Laflamme et al., 1990), CS-CELP (Kataoka et al., 1993), CS-ACELP (Salami et al., 1998), PSI-CELP (Miki et al., 1993), RCELP (Kleijn et al., 1993), eX-CELP (Gao et al., 2001) - se tornaram, a partir de sua criação na década de

[1] Essas vantagens existem, supondo que o sistema tenha sido projetado apropriadamente.

80, a técnica de codificação dominante.

Para taxas acima de 4kb/s, os codificadores de forma de onda baseados no codificador CELP são capazes de produzir fala de boa qualidade. Para taxas abaixo desse valor, a maioria dos codificadores busca modelar apenas as características perceptuais mais importantes, tipicamente através da codificação de parâmetros do modelo de predição linear do sinal de fala (Benesty et al., 2008). Por isso, esses codificadores são chamados de codificadores paramétricos.

Este capítulo descreve o sistema de codificação de fala empregado neste trabalho, o qual segue uma abordagem diferente da abordagem desses codificadores e é baseada em um paradigma de síntese concatenativa (Lee e Cox, 2001), geralmente empregada em sistemas de conversão texto-fala. Tal codificador pode ser útil, por exemplo, em aplicações que requerem o armazenamento e/ou a transmissão de grandes quantidades de sinais de fala pré-gravados. Este é o caso de sistemas de conversão texto-fala (Seção 4.2) baseados em síntese concatenativa, que dependem de uma extensa base de sinais de fala gravados para produzir resultados de qualidade elevada (Seção 4.3.1).

Uma questão importante é definir um método de avaliação do desempenho do sistema de codificação empregado, quanto à qualidade do sinal de fala produzido. Duas formas tradicionais para avaliação da qualidade de sinais de fala são a avaliação subjetiva (inspeção auditiva) e avaliação objetiva (medida gerada por software). A avaliação subjetiva consiste do uso de métodos padronizados para geração de notas de avaliação de qualidade por avaliadores humanos, sendo esta bastante dispendiosa em termos de tempo e requisitos de infra-estrutura. A avaliação objetiva, por sua vez, substitui os avaliadores humanos por um algoritmo cuja função é modelar o comportamento desses avaliadores, através da utilização de modelos psicoacústicos, que levam em conta diversas características peculiares do aparelho auditivo humano.

A compressão de uma extensa base de fala será utilizada para avaliar o desempenho dos algoritmos de quantização vetorial apresentados no Capítulo 3, uma vez que representam um cenário desafiador para tais algoritmos, com uma grande quantidade de dados de alta dimensão. A avaliação se dará de duas formas distintas, uma baseada na qualidade do sinal de fala produzido e outra baseada na distribuição de protótipos gerada, sendo que esta última está descrita na Seção 3.4 e a primeira está descrita na Seção 5.6. Deve-se ressaltar que ao avaliar a qualidade do sinal de fala produzido, avalia-se o sistema como um todo (parâmetros, algoritmo de quantização, técnica de concatenação etc.) e não apenas a eficiência do algoritmo de quantização vetorial aplicado.

A organização do capítulo é a seguinte: a Seção 5.2 introduz os conceitos básicos de processamento de sinais utilizados. A Seção 5.3 mostra como é construído um *codebook* de quadros de fala (chamado de dicionário de quadros), através de quantização vetorial, e a Seção 5.4 descreve como tal dicionário pode ser usado para a codificação de um sinal de fala, visando a compressão de uma base de sinais. Na Seção 5.5, serão apresentados alguns dos principais parâmetros extraídos dos sinais de

fala e que são usualmente empregados em seu processamento. Por fim, a Seção 5.6 descreve técnicas de avaliação da qualidade de um sinal de fala, que são necessárias para avaliar o desempenho do sistema descrito.

5.2 Processamento do Sinal de Fala

Um sinal de fala é produzido a partir da passagem do ar pelo aparelho fonador humano. As características do sinal de fala em um dado instante dependem da configuração momentânea do trato vocal do falante, ou seja, da abertura dos lábios e da mandíbula, da posição da língua, da taxa de vibração das pregas vocais etc. Ao proferir uma sentença, o falante modifica continuamente a configuração de seu trato vocal, de forma a produzir uma sequência de sons que transmite uma mensagem ao ouvinte. Essa sequência de sons é composta por unidades básicas denominadas fones. Pode-se definir um fone como um trecho do sinal de fala, cujas características acústicas seguem um determinado padrão (Simões et al., 2008).

Pode-se dividir os sinais de fala em três categorias: os vozeados, os não-vozeados e aqueles com características híbridas entre as dos sinais vozeados e as dos não-vozeados.

Nos trechos de sinal de fala chamados de vozeados (do inglês *voiced*), ocorre a vibração das pregas vocais. Percebe-se, nesse caso, que o sinal de fala apresenta uma característica quase periódica, em que a frequência do sinal produzido está diretamente relacionado à taxa de vibração das pregas vocais (Figura 5.1a). Já nos trechos não-vozeados do sinal (do inglês *unvoiced*), não ocorre vibração das pregas vocais e o sinal apresenta característica totalmente aperiódica, assemelhando-se a um sinal de ruído (Figura 5.3a).

Embora as características acústicas do sinal de fala variem continuamente ao longo do tempo, é possível analisá-las de forma discreta. Supondo que o sinal de fala é estacionário se considerarmos períodos de tempo suficientemente pequenos, este sinal é subdividido em trechos de curta duração, chamados quadros (*frames*), cujas características podem ser consideradas praticamente constantes.

A forma mais comum de divisão do sinal é adotar quadros de tamanho fixo, por exemplo, 10 ou 20 ms. No entanto, neste trabalho utiliza-se uma abordagem que trata de forma diferente os trechos vozeados dos não-vozeados, com o auxílio das marcas de *pitch*. Nos trechos do sinal com característica vozeada, as marcas de *pitch* são posicionadas nos picos do sinal de fala. Nos trechos não-vozeados, as marcas de *pitch* são posicionadas em instantes igualmente espaçados no tempo, no caso 10 ms. Define-se um quadro como sendo o trecho de sinal centrado em uma marca de *pitch*, iniciando-se na marca anterior e terminando na marca seguinte. Outras duas definições emergem dessa: o período esquerdo do quadro, espaço de tempo entre a marca inicial e a central, e o período direito, espaço de tempo entre a marca central e a final. Percebe-se que com essa forma de divisão

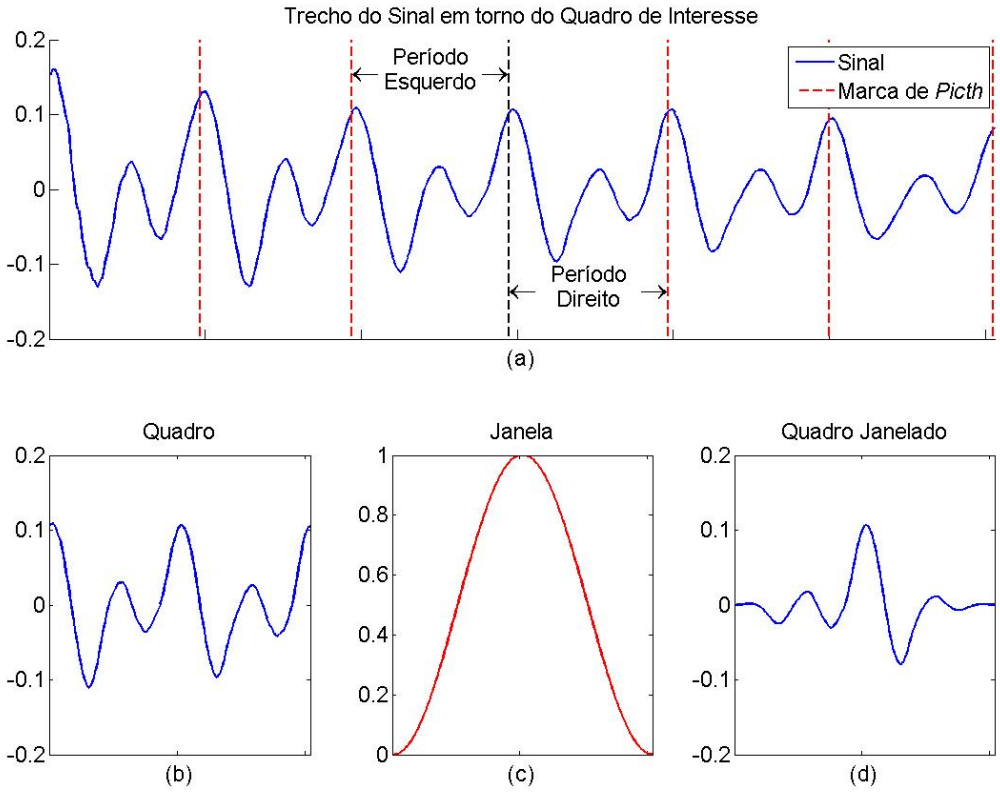


Fig. 5.1: **Etapas do processamento do sinal de fala empregado neste trabalho. (a) Trecho de Sinal Vozeado. (b) Quadro. (c) Janela com o mesmo número de amostras do quadro, obtida pela concatenação de duas janelas Hanning. (d) Quadro Janelado.**

do sinal há sobreposição entre quadros consecutivos, pois as amostras do período esquerdo de um certo quadro coincidem com as amostras do período direito do quadro imediatamente anterior. É importante notar também que o quadro pode ser assimétrico em relação à marca de *pitch* central, ou seja, seus períodos direito e esquerdo podem ser diferentes.

Entretanto, antes de serem analisados, os quadros passam ainda por uma etapa de janelamento. O janelamento pode ser entendido como um procedimento para limitar a análise de um sinal a apenas um certo trecho. Nesse sentido, a divisão do sinal em quadros, descrita anteriormente, nada mais é do que um processo de janelamento utilizando uma função retangular de amplitude unitária e limitada pelas marcas de *pitch*. No entanto, a função retangular não é uma escolha interessante para esta aplicação, como ficará mais claro a seguir.

No processo de janelamento de quadros adjacentes, as janelas são posicionadas de forma que a primeira amostra de uma janela coincida com a amostra da marca central da janela anterior e a última

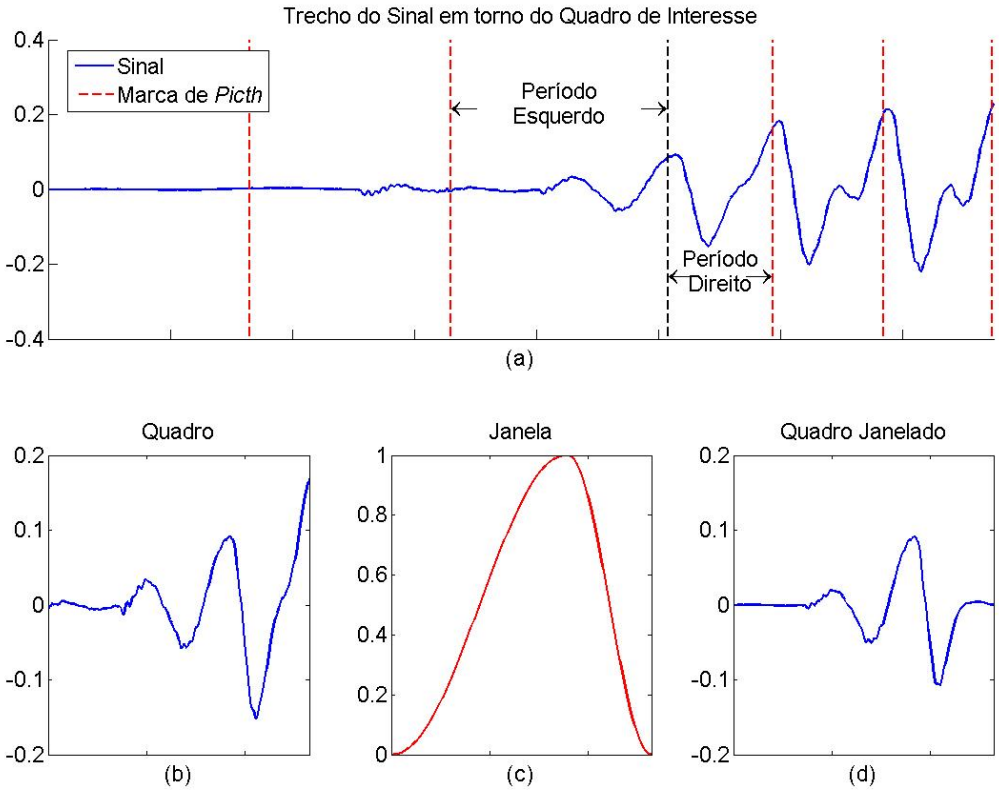


Fig. 5.2: **Etapas do processamento do sinal de fala empregado neste trabalho. (a) Trecho de Sinal Híbrido de Transição. (b) Quadro. (c) Janela com o mesmo número de amostras do quadro, obtida pela concatenação de duas janelas Hanning. (d) Quadro Janelado.**

amostra dessa mesma janela coincida com a amostra da marca central da janela seguinte. Ao somar as amostras de janelas consecutivas posicionadas dessa maneira, em um processo chamado PSOLA (*Pitch-Synchronous Overlap and Add*) (Moulines, 1990), deseja-se obter uma sequência de amostras de valor unitário constante. Essa propriedade permite reconstruir o sinal original sem distorção a partir de seus quadros janelados. Para isso, basta posicionar os quadros janelados conforme descrito anteriormente e em seguida fazer o *overlap and add* dos mesmos (Simões et al., 2008).

As funções mais comuns utilizadas no janelamento de quadros de sinais de fala são as janelas de Hamming e de Hanning. Nesse trabalho utiliza-se a janela de Hanning (Figuras 5.1c, 5.2c, 5.3c), definida pela Equação 5.1, onde *N* é o número de amostras da janela (Boll, 1979; Makhoul e Wolf, 1972).

$$\omega(n) = \frac{1}{2} \cdot \left[1 - \cos\left(2\pi \cdot \frac{n}{N-1}\right)\right] \rightarrow 0 \leq n \leq N-1 \tag{5.1}$$

Além de conseguir reconstruir o sinal original, outra vantagem de se utilizar uma janela com decaimento nas laterais é evitar a inserção de conteúdo de alta frequência na análise espectral dos quadros, que decorreria da tentativa de modelar o degrau no início e no fim de um quadro obtido com uma janela retangular.

As funções que definem as janelas são, em geral, simétricas em relação ao seu centro. Dada a maneira como se definiu um quadro neste trabalho, estes podem ser assimétricos e, para realizar o janelamento, constroi-se então uma janela assimétrica a partir da concatenação de duas metades de janelas de tamanhos diferentes, definidos pelo período esquerdo e pelo período direito do quadro em questão. Assim, caso se deseje fazer o janelamento de um quadro de período esquerdo N_1 e período direito N_2, a primeira metade da janela requerida corresponde à metade esquerda de uma janela de Hanning de tamanho $2N_1$, e a segunda metade corresponde à metade direita de uma janela de Hanning de tamanho $2N_2$. Essa janela (assimétrica, se $N_1 \neq N_2$) é definida pela expressão 5.2.

$$\omega(n) = \begin{cases} \frac{1}{2} \cdot \left[1 - \cos\left(2\pi \cdot \frac{n}{2N_1 - 1}\right)\right] & \rightarrow \quad 0 \leq n \leq N_1 \\ \\ \frac{1}{2} \cdot \left[1 - \cos\left(2\pi \cdot \frac{n - N1 + N2 - 1}{2N_2 - 1}\right)\right] & \rightarrow \quad N_1 \leq n \leq N_2 + N_1 - 1 \end{cases} \tag{5.2}$$

Ao multiplicar-se o sinal de fala por uma janela posicionada na marca de *pitch* central do quadro sob análise, obtém-se o quadro janelado, cujas amostras correspondem às amostras originais do quadro com atenuação crescente em direção às bordas. São esses quadros janelados que constituem a unidade básica de análise neste trabalho.

As Figuras 5.1, 5.2 e 5.3 ilustram esse processo de manipulação do sinal de fala para a obtenção dos quadros janelados. Nessas figuras, há um trecho de sinal de fala e as marcas de *pitch* (a), destacando-se um quadro de interesse (b), a janela utilizada (c) e o quadro janelado obtido (d). A Figura 5.1 exemplifica um quadro de um trecho vozeado do sinal, como se pode perceber por sua característica periódica. A Figura 5.2 mostra um exemplo de um quadro de transição do silêncio (não-vozeado) para uma vogal (vozeada), onde se observa uma característica híbrida. Note como o quadro é nitidamente assimétrico, com período esquerdo maior do que o período direito, e como isso se reflete na janela utilizada. Na Figura 5.3, há um exemplo de quadro não-vozeado.

Por fim, os quadros janelados são submetidos a um processo de parametrização, no qual cada quadro passa a ser representado por um conjunto de parâmetros. Uma série de atributos pode ser calculada a partir de um sinal de fala e, na Seção 5.5, serão descritos alguns dos principais parâmetros extraídos dos sinais de fala e que são usualmente empregados em diversas aplicações (síntese de fala, reconhecimento de fala etc.).

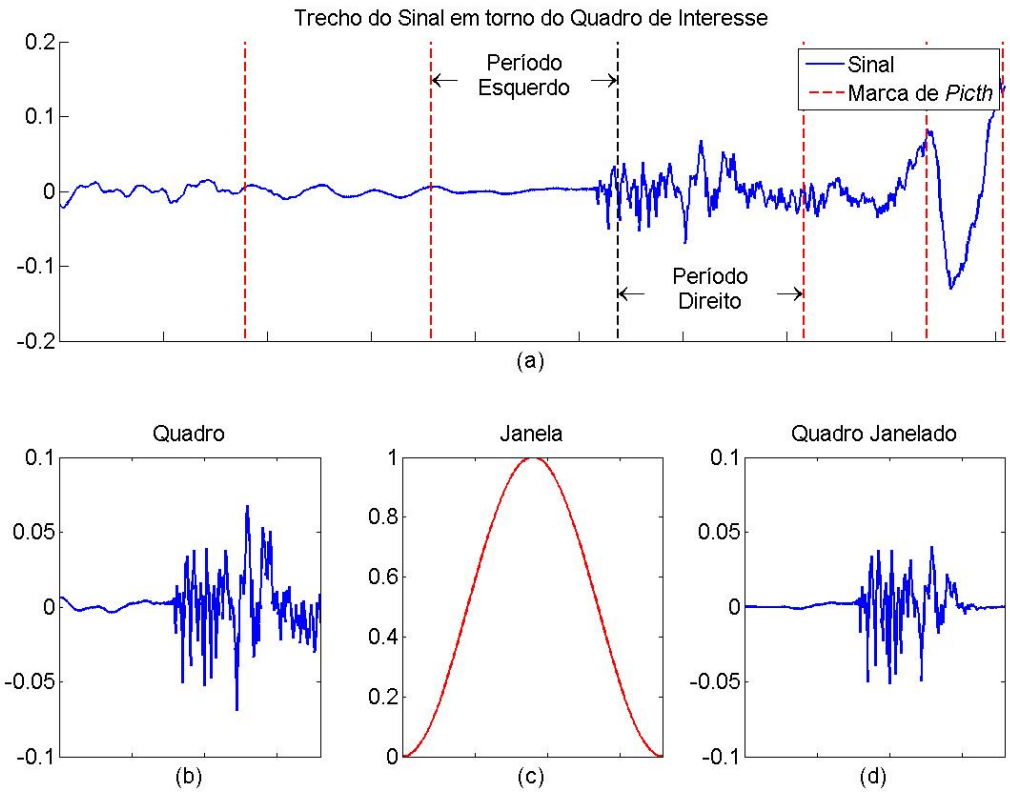


Fig. 5.3: **Etapas do processamento do sinal de fala empregado neste trabalho. (a) Trecho de Sinal Não-Vozeado. (b) Quadro. (c) Janela com o mesmo número de amostras do quadro, obtida pela concatenação de duas janelas Hanning. (d) Quadro Janelado.**

5.3 Geração do Codebook

Suponha que exista uma base de gravações de sinais de fala utilizada em um sistema de síntese de fala (Capítulo 4), a qual se deseja compactar. Chamada de base de treinamento, ela será utilizada para a geração do dicionário de quadros e de seu correspondente *codebook*.

Os sinais de fala presentes na base de treinamento são submetidos ao processo descrito na Seção 5.2, no qual os sinais são subdivididos em quadros, janelados e, por fim, parametrizados. Para todos os quadros, o conjunto de parâmetros calculados é o mesmo, de forma que cada quadro gera um vetor de atributos de mesma dimensão. Assim, a base de sinais de fala é convertida em uma matriz de vetores de atributos. Chamar-se-á de base de treinamento tanto a base de sinais de fala quanto sua correspondente matriz de vetores de atributos, sendo que através do contexto é possível distinguir quando se está falando de uma ou de outra.

Uma vez calculados os parâmetros dos quadros, um algoritmo de agrupamento de dados (Capítulo 3) pode ser utilizado para agrupar (quantizar) os quadros, de acordo com a proximidade (segundo alguma métrica) entre seus vetores de parâmetros. O resultado da execução desses algoritmos é um conjunto de protótipos que representa o conjunto de dados de entrada (base de treinamento), separando-os em grupos diferentes. Deseja-se que os quadros pertencentes a um mesmo grupo, ou seja, representado pelo mesmo protótipo, sejam semelhantes entre si, de modo que o protótipo seja um bom representante de todos eles.

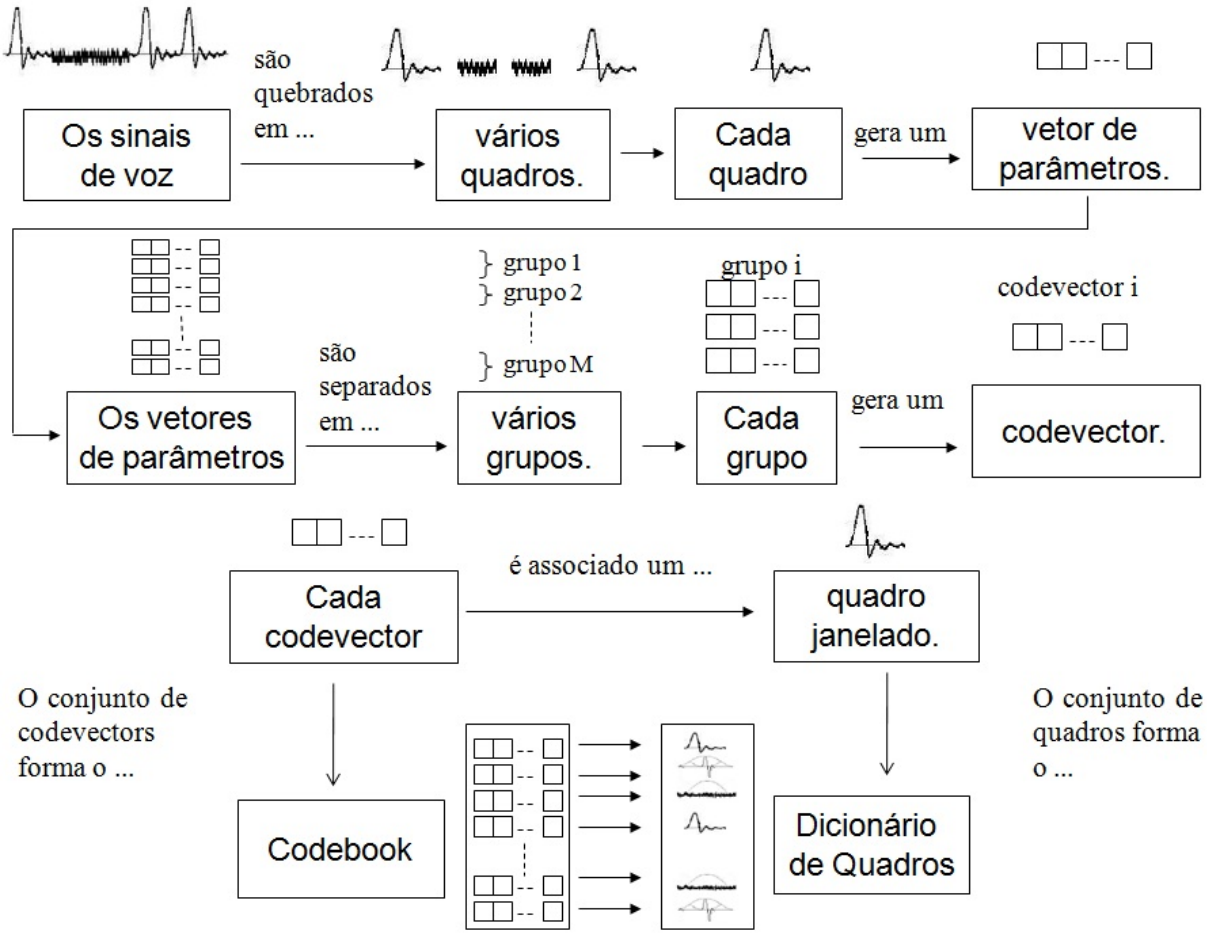
Deseja-se associar aos protótipos um quadro, por motivos que serão explicados na próxima seção. Como o protótipo de cada grupo não necessariamente coincide com a posição de um vetor de treinamento existente, é escolhido como representante do grupo o vetor de treinamento mais próximo ao protótipo desse grupo, a fim de associá-lo ao quadro que o originou. Designam-se esses vetores de *codevectors*. O conjunto dos *codevectors* forma o *codebook*, ao qual relaciona-se o *dicionário de quadros*, constituído pelos quadros que originaram os *codevectors*. Em um sistema de compressão, o *dicionário de quadros* será utilizado para a reconstrução das formas de onda, na etapa de decodificação do sinal de fala, enquanto o *codebook* será utilizado na etapa de codificação desse sistema, descrito na próxima seção. A Figura 5.4 ilustra essas etapas.

5.4 Compressão do Sinal de Fala

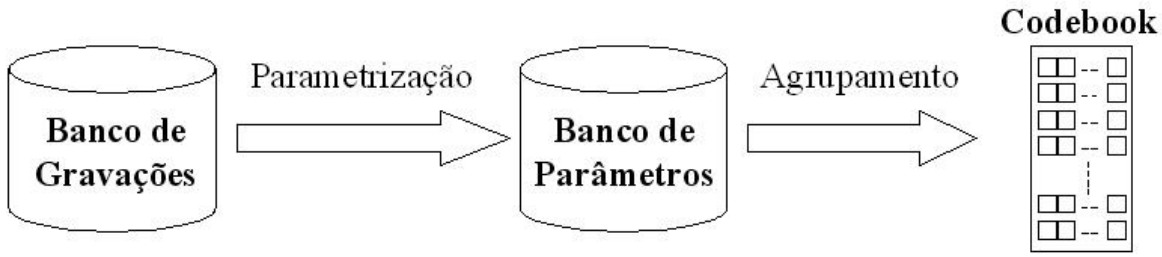
Uma vez gerados o *codebook* e o dicionário de quadros, é possível utilizá-los para codificar qualquer sinal de fala do mesmo locutor [2], inclusive a própria base de treinamento. Para isso, o sinal que se deseja codificar deve passar pelo mesmo processo de divisão em quadros, janelamento e parametrização descrito na Seção 5.2, através do qual o sinal de fala é transformado em uma sequência de vetores de parâmetros. Para cada vetor de parâmetros, varre-se então o *codebook* em busca do *codevector* mais próximo, de forma que a sequência de quadros é mapeada em uma sequência de índices do *codebook*. Esse processo de codificação está ilustrado na Figura 5.5(a).

Para a reconstrução (decodificação) do sinal, a sequência de índices é remapeada em uma sequência de quadros, utilizando os quadros do dicionário. Essa sequência de quadros é então concatenada através da técnica de síntese de fala, conhecida como PSOLA (*Pitch-Synchronous Overlap and Add*), como ilustra a Figura 5.5(b). Além dos índices, também é necessário ter/receber no decodificador a informação de frequência fundamental e energia de cada quadro, cujas finalidades estão descritas a seguir. Portanto, o codificador também deve extrair essas informações do quadro original para disponibilizá-las.

[2] Na realidade, nada impede de se usar o *codebook* e o dicionário de quadros para codificar sinais de fala produzidos por outros locutores, embora a qualidade do sinal e a identidade do locutor tendam a ficar severamente comprometidas.



(a)



(b)

Fig. 5.4: **Etapas do processo de geração do *codebook*. 5.4(a) Etapas detalhadas. 5.4(b) Processo resumido.**

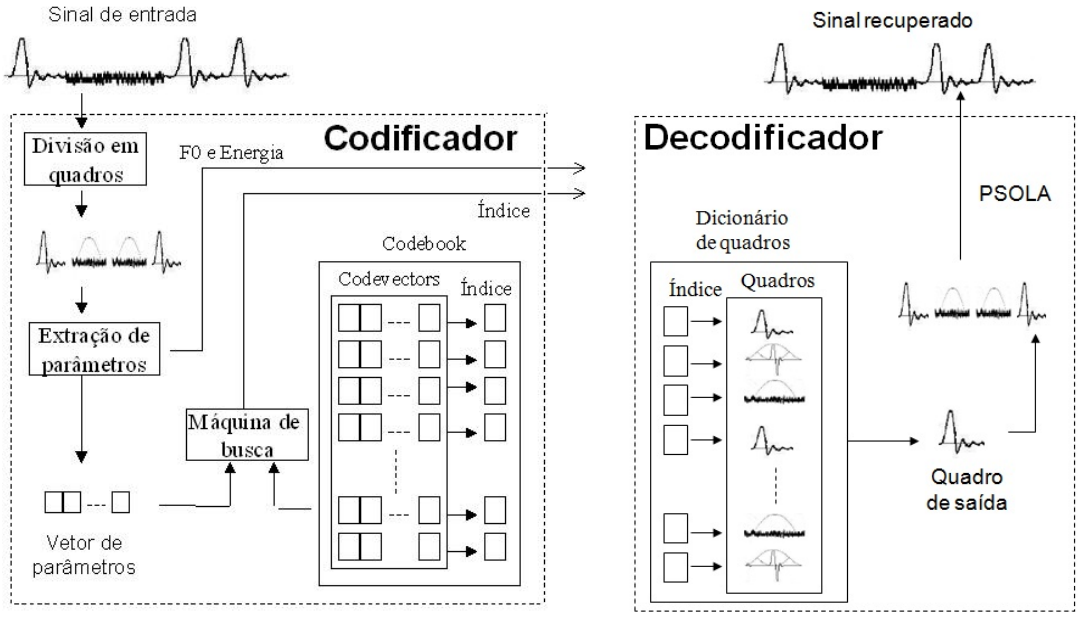


Fig. 5.5: **Processo de codificação e decodificação do sinal de fala. 5.4(a) Codificação do sinal de fala utilizando o codebook. 5.4(b) Decodificação do sinal de fala utilizando o dicionário de quadros e os índices obtidos na etapa de codificação.**

Na técnica PSOLA, os quadros são posicionados de forma a preservar os períodos de *pitch* do sinal original da seguinte forma: o centro do quadro do dicionário (marca de *pitch* central) é posicionado de forma que coincida com a marca de *pitch* central do quadro original. Dado que esses quadros (o original e o advindo do dicionário), não têm necessariamente o mesmo tamanho (o período esquerdo de um quadro não é igual ao período direito do quadro seguinte), a sobreposição de quadros consecutivos pode ser maior ou menor do que 50%. Durante essa operação de *overlap and add*, faz-se também um ajuste de amplitude dos quadros a fim de preservar a energia do sinal original.

Caso o objetivo seja compactar uma base de fala, ela própria é usada como base de treinamento e, em seguida, ela passa pelo processo de codificação descrito acima, de tal modo que a base comprimida é constituída por:

- Dicionário de quadros;
- Sequência de índices de mapeamento dos quadros originais em quadros do dicionário;
- Frequência Fundamental dos quadros originais;
- Energia dos quadros originais.

Por fim, deve-se destacar que, para recuperar o sinal propriamente dito, este deve ser sintetizado em tempo real, pois ele fica armazenado em sua forma codificada (sequência de índices + frequência fundamental + energia).

No caso da codificação de um sinal qualquer, o decodificador deve conter apenas o dicionário de quadros e os outros itens são gerados pelo codificador (que contém o *codebook*) e transmitidos e/ou armazenados.

Neste trabalho, foi implementado desde a etapa de processamento do sinal de fala (janelamento e parametrização), passando pela geração do *codebook* e do dicionário de quadros (quantização vetorial) até a reconstrução do sinal (síntese concatenativa), partindo de uma base de sinais de fala, disponibilizada juntamente com a marcação de *pitch* dos sinais. Ou seja, foram desenvolvidos os módulos mostrados nas Figuras 5.4 e 5.5, sendo que foram feitos estudos quanto aos parâmetros utilizados, os quais serão descritos na próxima seção, e quanto ao algoritmo de quantização empregado para a seleção dos quadros do dicionário.

5.5 Parâmetros do Sinal de Fala

A parametrização é um método utilizado para extrair a informação que interessa de um sinal, dada determinada aplicação. Na maioria das situações, trabalhar diretamente com a forma de onda de um sinal pode ser inviável ou levar a resultados pobres. A parametrização do sinal, portanto, busca formas mais eficientes de interpretá-lo para alguma finalidade, qualquer que seja (armazenamento, transmissão, codificação etc).

Na literatura, podem ser encontradas descrições de inúmeros atributos extraídos de sinais de fala (Davis e Mermelstein, 1980; Picone, 1993). Dependendo da aplicação desejada, o uso de determinado atributo, ou um conjunto deles, se mostra mais interessante. Por exemplo, para reconhecimento de fala, deseja-se utilizar atributos capazes de armazenar informações que contribuam na discriminação do que foi falado, independente de quem falou. Já em uma aplicação de reconhecimento de locutor, deseja-se exatamente o contrário, ou seja, atributos que armazenem informações capazes de diferenciar cada locutor, independente do que foi falado [3].

Como foi dito nas seções anteriores, a etapa de parametrização desempenha um papel fundamental no sistema proposto, visto que a quantização da base de dados de fala ocorre no espaço dos parâmetros. A escolha de atributos capazes de discriminar os quadros de fala de forma eficiente é, portanto, de suma importância para a eficiência do sistema.

Para isso, os atributos devem conter informações que diferenciem um determinado quadro de fala

[3] Há também o caso do chamado reconhecimento de locutor dependente de texto, em que a informação do que foi falado também faz parte do reconhecimento.

de outro, de acordo com a percepção humana de seu som, uma vez que os quadros do dicionário irão substituir os quadros originais de um sinal, e deseja-se que tal codificação seja a mais imperceptível possível para o usuário. Nesta seção, são apresentados apenas os atributos mais usuais e de particular interesse para a aplicação proposta. Não é intenção deste texto fazer uma descrição completa e detalhada de todos os atributos de fala já propostos na literatura.

5.5.1 Parâmetros Extraídos no Domínio do Tempo

A partir da forma de onda no tempo, alguns parâmetros úteis podem ser extraídos. Dentre eles destacam-se a energia, a potência, taxa de cruzamentos de zeros, os já mencionados período esquerdo e período direito, e coeficientes de predição linear (LPC - *linear prediction coefficients*).

A energia é definida como a soma do quadrado das amostras do sinal e pode ser útil para, por exemplo, detectar quadros de silêncio, os quais normalmente apresentam energia muito menor do que aquela associada aos outros quadros, ou para normalizar o sinal, de forma que todos os quadros apresentem energia unitária. A potência é simplesmente energia por unidade de tempo, ou seja, a energia divida pela duração do quadro.

A taxa de cruzamentos de zeros é o número de vezes que amostras consecutivas trocam de sinal, de negativo para positivo ou vice-versa. Outro parâmetro conceitualmente parecido é o número de inflexões do sinal, dado pelo número de vezes que sua primeira derivada em relação ao tempo troca de sinal. Tais parâmetros podem ajudar a distinguir entre quadros de sons vozeados e não-vozeados, haja vista que estes últimos apresentam um comportamento mais ruidoso, de maneira que suas formas de onda têm um número muito maior de cruzamentos de zeros e/ou inflexões do que os quadros de sons vozeados.

O período esquerdo e o período direito já foram descritos na Seção 5.2, mas repetem-se aqui para completude da descrição. Considerando que um quadro é definido como o segmento de sinal que se inicia em uma marca de *pitch*, é centrado na marca seguinte e termina na marca posterior à central (ver Figuras 5.1, 5.2 e 5.3), o período esquerdo é definido como o número de amostras (ou o espaço de tempo) entre a marca central e a inicial, e o período direito tem a mesma definição, mas entre a marca central e a final. Esses períodos podem ser usados para obter a frequência fundamental F_0, no caso dos quadros vozeados, pois nesses casos as marcas de *pitch* estão espaçadas de um período fundamental (supondo que a marcação de *pitch* foi corretamente efetuada).

Os Coeficientes LPC e LSF

A predição linear é talvez a forma mais comum de análise do sinal de fala (Atal, 2006). A predição linear é uma técnica de separação fonte/filtro que assume um modelo simples de produção de fala,

o qual considera que o sinal de fala é o resultado da passagem de um sinal de entrada por um filtro linear. Normalmente há interesse particular no filtro, pois dado o filtro e o sinal de saída (o sinal de fala), pode-se chegar ao sinal de entrada, muitas vezes visto como o sinal de erro da filtragem (resíduo) (Rabiner e Schaffer, 1976). Como descrito na Seção 5.2, considera-se que o sinal de fala em um quadro é estacionário e, portanto, o filtro é invariante no tempo para cada quadro.

Esta análise recebe o nome de predição linear, pois considera que cada amostra do sinal de fala pode ser aproximada (*predita*) a partir de uma combinação *linear* de amostras passadas. Os pesos dados às amostras passadas nesta combinação são os chamados coeficientes de predição linear (LPC - do inglês *Linear Prediction Coefficients*) e definem o filtro em questão, cuja ordem é determinada pelo número de amostras passadas utilizadas (Markel e Gray Jr., 1976). Quanto maior a ordem, melhor é a predição. Chamando de $s(n)$ um dado sinal de fala, $\hat{s}(n)$ sua aproximação e de M a ordem do filtro utilizado:

$$\hat{s}\,(n) = \sum_{i=1}^{M} a_i \; s(n-i) \tag{5.3}$$

onde a_i são os coeficientes do filtro de predição linear.

Existem alguns algoritmos para estimar os coeficientes LPC [4] de forma a minimizar o erro de predição, dos quais se destaca o algoritmo de Levinson-Durbin (Levinson, 1947; Durbin, 1959, 1960). Mas para este trabalho, o método utilizado não importa e, por isso, eles não serão descritos.

No entanto, sabe-se que os coeficientes LPC são inapropriados para quantização, devido a sua faixa dinânica relativamente grande e por que a quantização pode transformar um filtro LPC estável em um filtro instável (Song e Juang, 1993). Mas é possível representar os coeficientes LPC de outras maneiras. Uma representação bastante usada por sua robustez à quantização são os chamados coeficientes LSF (do inglês *Line Spectral Frequency*), também chamados de LSP (do inglês *Line Spectral Pairs*), introduzidos por (Itakura, 1975). Sem entrar no mérito matemático, os coeficientes LSF apresentam as seguintes propriedades (Hentz e Seara, 2009; Song e Juang, 1984):

- Têm faixa dinâmica limitada, o que os torna mais adequados para a quantização;
- Erros de quantização não tornam o filtro instável;
- Parâmetros LSF podem ser interpolados.

[4] Apesar do termo "coeficientes" já estar presente na sigla LPC, a expressão "coeficientes LPC" será utilizada para fazer referência a eles, pois é a forma usualmente empregada e soa mais natural para o leitor.

5.5.2 Parâmetros Extraídos no Domínio da Frequência

Geralmente, a análise de um sinal de fala ocorre no domínio da frequência. Até mesmo alguns dos parâmetros calculados no domínio do tempo podem ser também obtidos ou analisados no domínio da frequência.

Para isso, inicialmente é calculado o espectro de frequência do sinal, através da transformada discreta de Fourier (DFT - *discrete Fourier transform*), que na prática quase sempre é implementada com o algoritmo FFT (*fast Fourier transform*). O espectro da DFT é complexo e pode ser representado por suas partes real e imaginária ou por sua magnitude e fase. Sabe-se que o ouvido humano não é sensível à fase e, por isso, a magnitude do espectro é a representação mais adequada para o processamento de fala no domínio da frequência (Taylor, 2009).

A sensibilidade do ouvido humano é aproximadamente logarítmica, ou seja, uma multiplicação na amplitude do sinal produz apenas um crescimento aditivo na intensidade sonora percebida. Portanto, é comum representar a amplitude em uma escala logarítmica. Usualmente emprega-se o chamado *log do espectro de potência*, isto é, o logaritmo do quadrado da magnitude do espectro de frequência. A Figura 5.6 mostra o log do espectro de potência do quadro janelado da Figura 5.1.

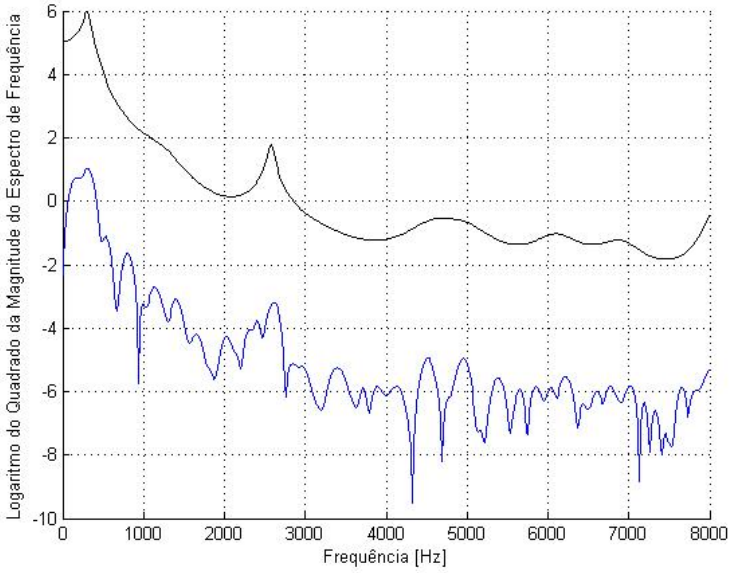


Fig. 5.6: **Logaritmo do quadrado da magnitude do espectro do quadro da Figura 5.1 e sua respectiva envoltória espectral.**

A partir da Figura 5.6, pode-se visualizar três conceitos importantes no processamento de fala: a frequência fundamental e suas harmônicas, os formantes e a envoltória espectral. Na Figura 5.6, nota-se uma série de picos, igualmente espaçados na frequência. São as harmônicas, múltiplas da

frequência fundamental do sinal, que pode ser estimada pela diferença entre duas harmônicas consecutivas. Os picos mais globais no espectro são os chamados formantes. Fisicamente, os formantes são as frequências de ressonância do trato vocal e a frequência fundamental é a taxa de vibração das pregas vocais. O contorno do espectro de potência, algo como "ligar" os picos do sinal, formando uma curva, é denominado a envoltória espectral e também está mostrado na figura. A envoltória foi obtida a partir da resposta em frequência do filtro LPC de ordem 20.

Outra peculiaridade do ouvido humano é sua resposta em frequência. Estudos revelaram que a resposta em frequência do ouvido humano é não-linear e empiricamente foram determinadas escalas mais convenientes para representar a frequência. Duas dessas escalas mais conhecidas são as escalas *mel* (Stevens et al., 1937) e *Bark* (Zwicker e Fastl, 1990). O mapeamento da escala linear (em Hertz) para a escal mel de frequência é dado por:

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) = 1127 \log_e\left(1 + \frac{f}{700}\right) \tag{5.4}$$

onde m é o valor da frequência na escala mel e f é o valor da frequência em Hertz. Neste trabalho, será utilizada a escala mel, motivo pelo qual não será apresentado aqui o mapeamento para a escala Bark, o qual pode ser facilmente encontrado na literatura (Zwicker e Fastl, 1990; Zwicker, 1961). A Figura 5.7 mostra o resultado deste mapeamento (até a frequência de 16 kHz), na qual pode-se observar que até 1000 Hz a relação é praticamente linear e depois ela segue uma curva logarítmica.

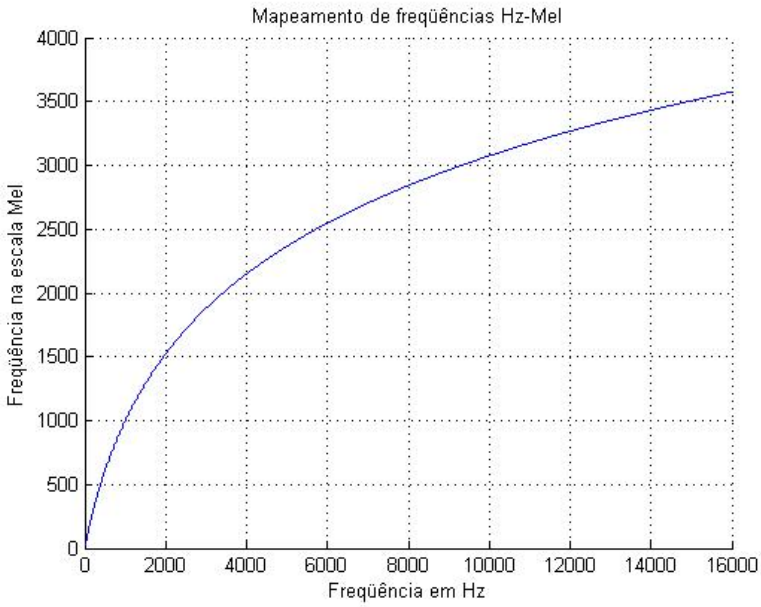


Fig. 5.7: **Mapeamento de frequências nas escalas linear (em Hertz) e mel, segundo a fórmula da Equação 5.4.**

Os Coeficientes Mel

Particularmente, dado que a frequência fundamental já foi extraída de alguma forma, interessa apenas o contorno do espectro, ou seja, a envoltória espectral. Mais do que isso, já que a resposta do ouvido humano pode ser aproximada de uma maneira melhor na escala mel, deseja-se obter uma informação condizente com nossa percepção de frequências. Agrupar o conteúdo espectral de frequências próximas leva a uma estimativa do contorno e, se a largura de banda de cada um desses grupos de frequência obedecer à escala mel, aproxima-se da reposta do ouvido (Picone, 1993). Uma forma eficiente de obter e armazenar essa informação é com a chamada análise por banco de filtros (Taylor, 2009).

Imagine que o sinal de fala passa por um banco de filtros passa-faixa, onde cada filtro define uma banda crítica, espaçados uniformemente na escala mel. O número de filtros utilizados deve ser suficiente para produzir uma boa estimativa da envoltória espectral e eles devem cobrir a largura de banda do sinal, isto é, para um sinal amostrado a, digamos, 16 kHz, precisa-se de filtros até a frequência de 8 kHz (obedecendo ao teorema da amostragem - (Oppenheim e Schaffer, 1989)). Além disso, para que haja preservação da energia, a soma das respostas em frequência dos filtros deve ser sempre unitária. Um formato simples geralmente empregado para respeitar essa condição é o triangular. A Figura 5.8 ilustra tal banco de filtros, cujos valores de frequência central podem ser encontrados em (Picone, 1993).

Por fim, pode-se calcular a energia do sinal resultante de cada filtragem, ou melhor, o logaritmo da energia. O resultado dessa operação é o que chamaremos de *coeficientes mel*, sendo esta uma nomenclatura empregada neste trabalho e não encontrada na literatura. A Figura 5.9 ilustra tal resultado para o sinal da Figura 5.6.

Resumindo, o conteúdo espectral na escala mel, aqui chamado de coeficientes mel, é definido como o logaritmo na base 10 da energia contida no sinal após passar por um filtro de banda-crítica. Matematicamente, portanto, os coeficientes mel são dados por:

$$coeficiente\ mel^{(n)} = \log_{10} \sum_{k=-\infty}^{\infty} |(X(k) \cdot H_n)|^2 \tag{5.5}$$

onde $coeficiente\ mel^{(n)}$ designa o n-ésimo coeficiente mel, $X(k)$ é a transformada de Fourier do sinal de entrada $x(t)$ (um quadro janelado) e H_n é a resposta em freqüência do n-ésimo filtro de banda-crítica, mostrados na Figura 5.8.

Por último, pode-se empregar uma normalização de energia, simplesmente dividindo cada coeficiente mel pela soma de todos eles.

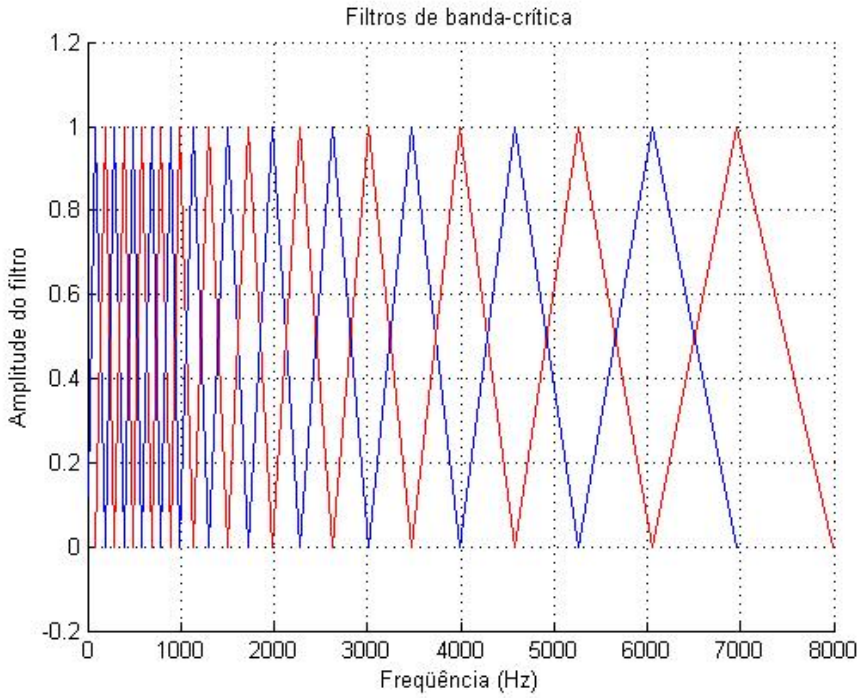


Fig. 5.8: **Banco de filtros triangulares linearmente espaçados na escala mel. A figura mostra a escala de frequência em Hertz, para facilitar a compreensão do comportamento do banco.**

Os Coeficientes MFCC

O último conjunto de parâmetros do sinal de fala que será tratado aqui são os chamados coeficientes mel-cepstrais (MFCC - mel-frequency cepstral coefficients). O termo *cepstro* foi cunhado a partir da inversão da primeira metade da palavra espectro (a primeira letra *e* de espectro foi suprimida), ou, em inglês, *cepstrum* a partir de spectrum (Borget et al., 1963).

Os coeficientes mel-cepstrais são a transformada discreta de cosseno (DCT - *Discrete Cosine Transform*) do logaritmo na base 10, da energia do sinal resultante da filtragem do sinal original, por um dos filtros de banda-crítica na escala mel, descritos na seção anterior (Davis e Mermelstein, 1980). Ou seja, os coeficientes mel-cepstrais são a transformada de cosseno dos coeficientes mel. Matematicamente, os MFCC são dados por:

$$MFCC^{(n)} = \sum_{k=1}^{K} m(k) \cdot \cos\left[n \cdot (k - 0.5) \cdot \frac{\pi}{K}\right] \tag{5.6}$$

onde $MFCC^{(n)}$ designa o n-ésimo coeficiente mel-cepstral, $m(k)$ é o $k-$ésimo coeficiente mel e K é o número de filtros de banda-crítica utilizados. Repare que para $(n) = 0$ o MFCC é a própria energia do sinal e que é possível calcular quantos coeficientes se desejar, sendo usual o emprego dos

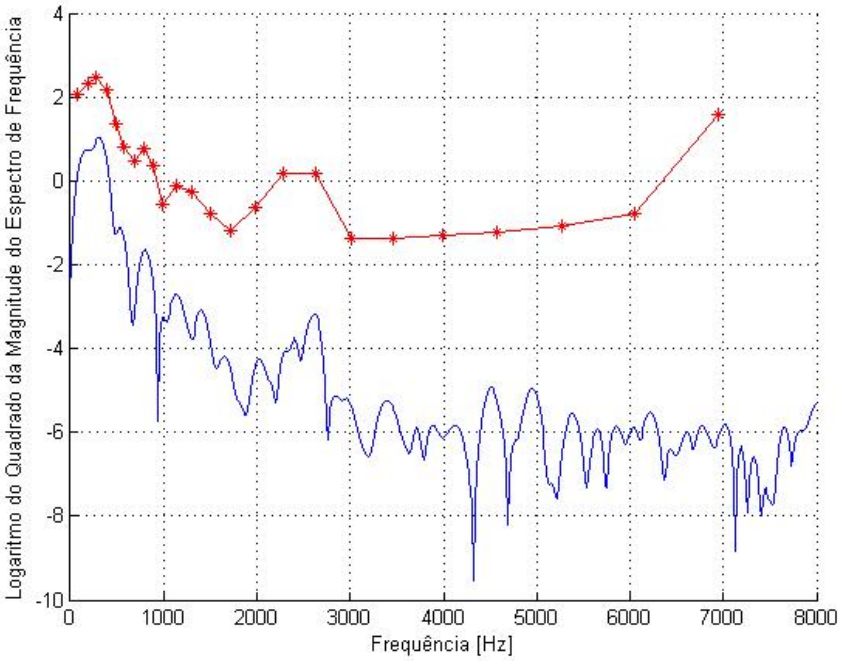


Fig. 5.9: **Coeficientes mel do quadro da Figura 5.1. Para efeito de visualização, foi feita uma correção de amplitude nos coeficientes, mas interessa apenas no seu formato.**

12 primeiros coeficientes seguintes à energia.

Portanto, os coeficientes MFCC são uma transformação matemática, que se alega ser capaz de realizar uma boa separação fonte/filtro, além de produzir coeficientes estatisticamente independentes, o que é de particular interesse em algumas aplicações. Por isso, esses coeficientes são provavelmente, ao lado dos coeficientes LPC, os mais utilizados nas mais variadas aplicações de processamento de fala.

5.6 Métodos de Avaliação

Dada a crescente expansão dos sistemas de comunicação vista nos últimos anos, com a telefonia fixa, a telefonia móvel e a internet, todos envolvendo a fala como o único ou um dos principais meios de interação, desenvolver métodos práticos para avaliar a qualidade da fala se tornou imprescindível, pois o sucesso de qualquer tecnologia, seja ela um equipamento de rede, um método de codificação de fala, um terminal do usuário etc, depende fortemente da qualidade do sinal de fala percebida pelo usuário final.

A qualidade da fala é resultado de um processo psicoacústico complexo de percepção humana.

Ao ouvir um sinal de fala, uma pessoa estabelece uma relação entre o que foi ouvido e o que seria esperado ou o ideal (um conceito interno de cada indivíduo), produzindo uma percepção individual de qualidade. Portanto, a qualidade de um sinal de fala é uma medida subjetiva, ou seja, pessoas diferentes avaliam de forma diferente a qualidade de um mesmo sinal.

O que está sendo chamado aqui de *qualidade da fala*, na verdade é composto por vários fatores, ou dimensões perceptuais. As dimensões mais comuns são inteligibilidade, naturalidade, nível de ruído etc. Entretanto, somente em aplicações específicas é comum o uso de apenas um desses fatores individualmente. Menos comum ainda é o uso de uma métrica multidimensional que comporte várias dessas dimensões, dada a complexidade de se definir tal métrica. Geralmente emprega-se uma métrica única, que seja capaz de reproduzir a percepção geral do que então se chama de qualidade da fala.

A forma mais óbvia de estimar a qualidade da fala é pedir para um grupo de pessoas ouvir amostras do sinal e dar nota para a qualidade percebida, a partir das quais pode-se então determinar a qualidade "média" do sinal. Fica claro que esta abordagem, denominada de teste subjetivo, é bastante custosa e demorada, quase sempre inviável em um ambiente de crescente demanda por avaliações em campo e em tempo real.

Por essas dificuldades em empregar testes subjetivos, foram desenvolvidas medidas objetivas de avaliação, baseadas em algoritmos computacionais, que tentam inferir objetivamente um julgamento subjetivo que é a qualidade da fala, buscando aproximar seus resultados dos que seriam obtidos em um teste subjetivo.

Além da separação nessas duas categorias (teste subjetivos ou testes objetivos), os métodos de avaliação da qualidade da fala podem ser divididos em outros dois conjuntos, considerando-se a disponibilidade ou não do sinal original (sinal de referência), para compará-lo com o sinal após os processos de codificação e/ou transmissão, gerando medidas absolutas ou relativas.

Dentre os métodos de avaliação subjetiva, o mais comum é a medida absoluta dada na escala MOS (*mean opinion score*), normalizado pela ITU-T, em sua norma de referência P.800 (P.800, 1996). Nesse método, é pedido a um conjunto de ouvintes que eles avaliem uma série de sinais de fala, dando notas à qualidade de cada um dos sinais apresentados, de acordo com a escala MOS mostrada na Tabela 5.1. Repare que não há um sinal de referência para comparação. A nota MOS[5] de determinado sistema, como o próprio nome diz, é o resultado da média das notas dos ouvintes para todos os sinais apresentados. Por ser bastante simples, este método é também muito popular.

Os métodos de avaliação objetiva geralmente utilizados são medidas relativas, ou seja, dependem da existência de um sinal de referência para comparação. A classe mais simples de algoritmos é composta de métodos de comparação da forma de onda do sinal no domínio do tempo, como é o caso

[5]Chama-se o valor do MOS de um sistema de *nota MOS*, apesar do termo *nota* já estar embutido na sigla em inglês, pois soa mais natural para o leitor dessa forma.

Tab. 5.1: **Notas na escala MOS.**

Qualidade	Nota
Excelente	5
Boa	4
Razoável	3
Ruim	2
Péssima	1

da relação sinal-ruído (SNR - *signal-to-noise ratio*). Medidas baseadas no domínio da frequência, como a distorção espectral (SD - *spectral distortion*) são também simples de implementar e, além disso, apresentam maior correlação com testes subjetivos, sendo, por isso, mais aceitas como uma medida de qualidade.

No entanto, a maioria dos métodos de avaliação objetiva de qualidade da fala está baseada, atualmente, no que se pode chamar de domínio perceptual ou psicoacústico. Tais métodos buscam imitar os processos de percepção e de avaliação humanos. Tal processo envolve a resposta do sistema auditivo humano, cujo modelo já se considera bem definido na literatura, mas também existe um componente cognitivo, mais complexo e cujo modelo não se encontra tão bem desenvolvido.

A avaliação é realizada através da determinação de uma distância perceptual entre o sinal que se deseja avaliar e o sinal de referência e, em seguida, criando uma função, geralmente não-linear, que mapeie esta distância em uma medida de qualidade da fala. A fim de obter um estimador para a nota MOS, é necessário normalizar o resultado para a escala MOS, que varia de 1 a 5.

Os algoritmos mais conhecidos de avaliação objetiva da qualidade da fala baseados em modelos psicoacústicos de percepção são: BSD (*Bark Spectral Distance*) (Wang et al., 1992), PSQM (*Perceptual Speech Quality Measure*) (Beerends e Stemerdink, 1994), PAQM (*Perceptual Audio Quality Measure*) (Beerends e Stemerdink, 1992), PEAQ (*Perceptual Evaluation of Audio Quality*) (Thiede et al., 2000), PAMS (*Perceptual Analysis Measurement System*) (Rix e Hollier, 2000), MNB (*Measuring Normalizing Blocks*) (Voran, 1999a,b) e PESQ (*Perceptual Evaluation of Speech Quality*) (Rix et al., 2001).

O algoritmo PESQ é o mais recente deles e é fruto da combinação dos algoritmos PAMS e PSQM99 (uma versão atualizada e estendida do PSQM), tornando-se uma recomendação da ITU-T (P.862, 2001). Além disso, diferentemente das técnicas anteriores, PESQ é capaz de predizer, com boa correlação, a qualidade subjetiva de um sinal de fala em uma ampla gama de condições, como distorções de codificação, ruído e perda de pacotes (Rix et al., 2001).

Por isso, o algoritmo escolhido para ser utilizado neste trabalho foi o PESQ. Inicialmente, a norma de referência do PESQ era voltada para a avaliação de sinais de voz na faixa de telefonia (faixa de freqüência até 4 kHz). Posteriormente, foi produzida uma outra implementação de referência,

construída especialmente para trabalhar com espectro de banda larga (até 7 kHz) (P.862.2, 2007).

Foi utilizada a implementação do algoritmo PESQ disponibilizada na página web da ITU (ITU, 2008). A Figura 5.10, mostra como o algoritmo é utilizado para avaliar o sistema.

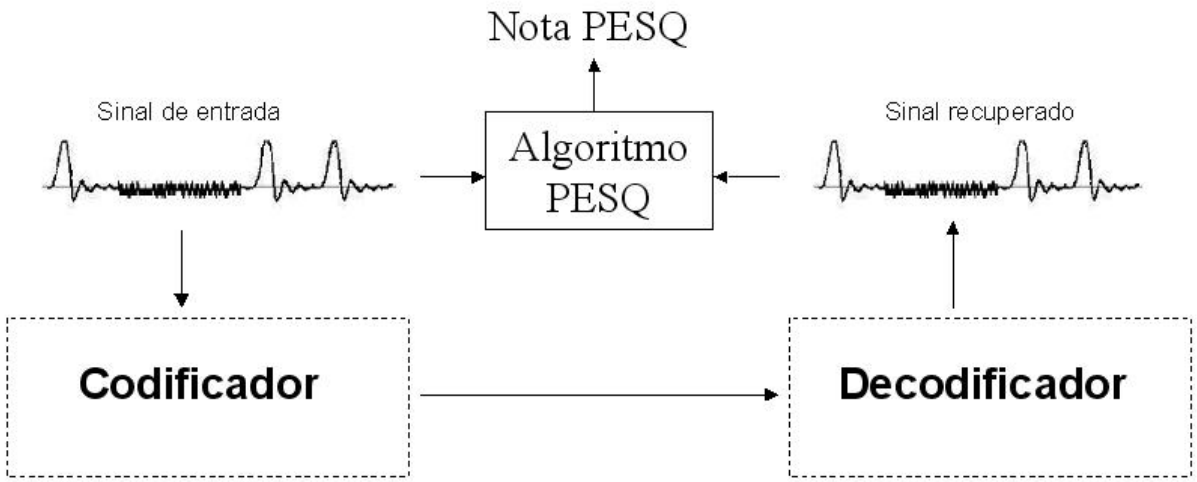


Fig. 5.10: **O algoritmo PESQ recebe como entradas os sinais original e degradado, obtido após a codificação e a decodificação, e fornece uma nota da qualidade do sinal degradado.**

Capítulo 6

Resultados

6.1 Descrição dos Dados Utilizados e dos Testes Realizados

O principal objetivo deste trabalho é avaliar o desempenho de alguns algoritmos de quantização, aqueles descritos no Capítulo 3, em uma aplicação específica e desafiadora: a quantização de sinais de fala, como descrito no Capítulo 5.

Para isso, obviamente precisa-se de uma base de sinais de fala gravados. Tal base foi fornecida pela Fundação Centro de Pesquisa e Desenvolvimento em Telecomunicações (CPqD) e, sendo uma base proprietária, não está disponível para consulta. A base disponibilizada é constituída por 450 frases, gravadas em estúdio por uma locutora profissional, amostradas a 16 kHz e digitalizadas com 16 bits por amostra, no formato PCM linear. Essas frases foram geradas de forma a apresentar ampla riqueza fonética. Juntamente com os sinais de fala, foram disponibilizados arquivos com a marcação de *pitch* de cada sinal, de forma que os quadros de fala já estavam definidos. A Tabela 6.1 resume as características dos sinais gravados.

Tab. 6.1: **Características do sinal de fala.**

Local de Gravação	Laboratório de gravação do CPqD
Locutora	Rosana Lee
Taxa de Amostragem	16 kHz
Número de bits por amostra	16
Formato	PCM linear

As 450 frases perfazem um total de aproximadamente 26 minutos de gravação e ocupam um total de 49,7 megabytes de memória. Primeiramente, as frases foram separadas em dois conjuntos distintos: 400 frases foram usadas para treinamento, ou seja, para a geração do *codebook* e do dicionário de quadros, conforme descrito na Seção 5.3, e as 50 frases restantes foram usadas para teste, em

que o *codebook* resultante foi utilizado para codificar os sinais da base de teste e o dicionário de quadros foi utilizado para decodificá-los, conforme descrito na Seção 5.4. A Tabela 6.2 sumariza as características dessa base, bem como dos conjuntos de treinamento e de teste.

Tab. 6.2: **Características da base de fala utilizada.**

	Treinamento	Teste	Total
Número de frases	400	50	450
Número de quadros	194.437	23.867	218.304
Duração (minutos)	~ 23	~ 3	~ 26
Memória ocupada (MB)	44,2	5,5	49,7

Conforme observado no Capítulo 5, nada impede que se utilize a própria base de treinamento para os testes.

Dois testes são usados como referência de desempenho para os algoritmos de quantização. No primeiro deles, além dos algoritmos descritos no Capítulo 3, um método mais trivial de escolha dos quadros que comporão o dicionário foi testado: a simples escolha aleatória. O outro teste é utilizar a base de treinamento inteira como *codebook*. Os próximos parágrafos explicam a razão para esses testes servirem de referência.

Dado que a base de treinamento completa sempre apresenta mais riqueza fonética do que sua versão quantizada [1], espera-se que ela produza resultados melhores. O papel dos algoritmos de quantização é justamente escolher os quadros de forma a promover a menor perda de qualidade possível, em comparação com o resultado obtido utilizando a base inteira. Como neste trabalho os dicionários de quadros serão muito pequenos em relação ao tamanho da base de treinamento, não se espera detectar informação puramente redundante (que resultaria em um empate de qualidade). Deseja-se apenas fazer a melhor escolha de quadros possível, pois sempre haverá degradação.

Nesse sentido, a escolha aleatória representa um limiar inferior, pois acredita-se que os algoritmos de quantização sejam capazes de selecionar quadros mais representativos do universo disponível, conduzindo assim a resultados melhores. Isso só não aconteceria em três situações: se a distribuição dos dados fosse realmente aleatória; se o algoritmo de quantização tiver um comportamento distorcido, escolhendo na verdade os quadros menos representativos; ou se as distâncias no espaço dos parâmetros utilizados não estabelecerem correlação com a qualidade do sinal obtido, ou em outras palavras, ainda que os vetores estejam próximos no espaço dos parâmetros, os quadros que eles representam não sejam perceptualmente parecidos.

[1] Tal afirmação supõe que a base tenha sido projetada adequadamente e que, portanto, quanto mais arquivos de voz, maior a riqueza presente, ou seja, considera-se que não há informação puramente redundante, cuja remoção não interfira na qualidade. É claro que, em um universo de 200 mil quadros, caso apenas um seja removido, por exemplo, provavelmente a qualidade do sistema não sofrerá variação significativa.

Dessas três hipóteses, a primeira pode ser automaticamente descartada, pois sabe-se que os dados não são aleatórios, mas sim fruto de locuções realizadas fisicamente. As outras duas serão avaliadas neste trabalho.

Com as 400 frases de treinamento foram montados quatro conjuntos. O primeiro deles é composto pelas frases de 1 a 50, o segundo pelas frases de 1 a 100, o terceiro pelas frases de 1 a 200 e o quarto pelas frases de 1 a 400, ou seja, todas as frases de treinamento. A Tabela 6.3 descreve as características de cada um desses conjuntos.

Tab. 6.3: **Características da base de fala utilizada.**

Conjunto	1	2	3	4
Número de frases	50	100	200	400
Número de quadros	28.146	51.899	99.984	194.437
Duração (minutos)	~ 3	~ 6	~ 12	~ 23
Memória ocupada (MB)	6,4	11,7	22,8	44,2

Sabe-se que em 400 frases há mais riqueza de informação, mas também mais redundância, e pretende-se com isso avaliar a influência do tamanho da base de treinamento nos resultados. Para isso, é necessário que os *codebooks* gerados a partir de cada conjunto tenham o mesmo tamanho, isolando, dessa forma, o efeito do crescimento da base de treinamento. Para este teste, foi escolhido um número de *codevectors* igual a 500. A razão desta escolha será apresentada abaixo.

Já para avaliar a influência do tamanho do *codebook*, para cada um dos conjuntos de treinamento, foram gerados *codebooks* com diferentes quantidades de *codevectors*. Mais do que isso, deseja-se testar o resultado obtido com diferentes taxas de compressão, para diferentes bases de treinamento. Considera-se aqui, como taxa de compressão, a relação entre o número de quadros usados no treinamento e o número de quadros no dicionário de quadros (número de *codevectors*). Com o intuito de realizar esses testes, definiu-se duas taxas de compressão fixas: 100 e 200 vezes. Obviamente, segundo a definição de taxa de compressão apresentada acima, para que ela seja fixa, se o tamanho da base de dados de treinamento dobra, o tamanho do codebook deve dobrar também.

O motivo da escolha desses valores (100, 200 e 500) leva em conta o seguinte fato: aproximando o número de quadros em cada conjunto de treinamento para 25, 50, 100 e 200 mil, respectivamente, o número de quadros dos dicionários, respeitando as taxas de compressão de 100 e 200 vezes, será de 125, 250, 500, 1000 e 2000, dependendo da configuração desejada, conforme apresenta a Tabela 6.4. Dessa forma, para os conjuntos 2 e 3, consegue-se atender dois requisitos: taxa de compressão fixa e número de *codevectors* fixo. Essa não é a única escolha que causa esse efeito, mas é uma das possíveis e é a que foi adotada.

Resumindo, com o conjunto de testes descrito na Tabela 6.4, testam-se:

Tab. 6.4: **Configuração dos testes realizados.**

Configuração Utilizada		Teste Realizado		
Número de Frases	Número de *Codevectors*	Taxa de Compressão Fixa		Número de *Codevectors* fixo 500
		100 vezes	200 vezes	
50	125		x	
	250	x		
	500			x
100	250		x	
	500	x		x
200	500		x	x
	1000	x		
400	500			x
	1000		x	
	2000	x		

- Tamanhos de *codebook* diferentes, para uma mesma base de treinamento;
- Taxas de compressão fixas, para diferentes bases de treinamento;
- Tamanho de *codebook* fixo, para diferentes bases de treinamento.

Finalizando a descrição dos dados utilizados e dos testes realizados, essas dez configurações foram avaliadas para todos os algoritmos, sendo que, para cada uma delas, o algoritmo foi executado 5 vezes, a fim de obter resultados médios. Além disso, foram testados diferentes conjuntos de parâmetros na composição dos vetores. Ou seja, cada algoritmo é executado 50 vezes para o conjunto de parâmetros em que ele é testado.

6.2 Avaliando Diferentes Conjuntos de Parâmetros

Primeiramente, buscou-se identificar o conjunto de parâmetros que levaria aos melhores resultados. Como descrito na Seção 5.5, os parâmetros de fala mais usuais, empregados em diversas áreas do processamento de fala, são os coeficientes LPC e os MFCC. Portanto, eles foram candidatos avaliados inicialmente. Além desses, os coeficientes LSF, obtidos a partir dos LPC, considerados mais robustos à quantização, também foram testados.

Baseado em valores frequentemente encontrados na literatura (Taylor, 2009), decidiu-se utilizar um número de 12 coeficientes MFCC, 20 coeficientes LPC e, portanto, também 20 coeficientes LSF (afinal, estes são em mesmo número que os LPC). Não foi feita uma análise variando-se esses valores.

Para testar esses três conjuntos de parâmetros, decidiu-se não utilizar todos os algoritmos, pois demandaria muito tempo e o objetivo é definir um conjunto ótimo de parâmetros, dentre os testados. Inicialmente não se objetiva comparar o desempenho dos algoritmos. Sendo o *k*-médias o mais simples dos algoritmos a serem testados, ele foi escolhido para esses testes. Para balizar os resultados, como descrito na seção anterior, também empregou-se a escolha aleatória de protótipos (limite inferior) e utilizou-se a base de treinamento inteira como dicionário de quadros (limite superior).

O algoritmo foi configurado para rodar por 50 iterações, suficientes para a convergência do algoritmo, e os protótipos iniciais foram escolhidos aleatoriamente entre os dados de entrada.

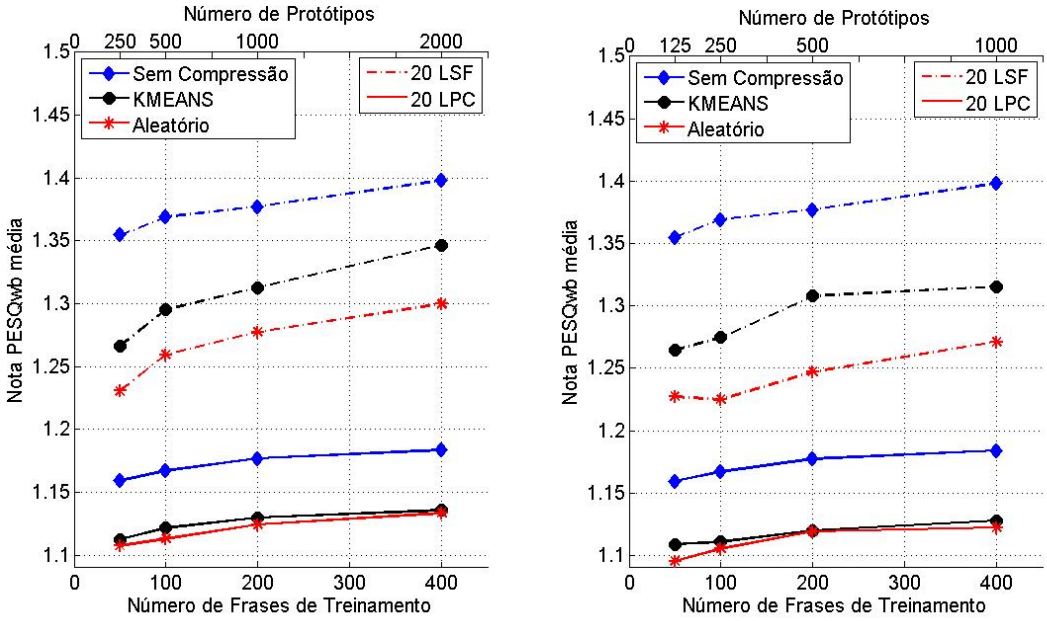
Primeiramente, testou-se se os parâmetros LSF produzem resultados melhores dos que os LPC. As Figuras 6.1(a), 6.1(b) e 6.1(c) apresentam os resultados obtidos para os testes descritos na Tabela 6.4. Nesses gráficos também foram incluídas as curvas de desempenho da escolha aleatória de *codevectors* e as curvas denominadas "Sem Compressão", que indicam os resultados obtidos quando se empregou a base de treinamento inteira como *codebook*. Estas últimas são idênticas em todos os gráficos, pois elas só dependem do número de frases usado no treinamento.

Nesses gráficos, o eixo vertical apresenta a nota PESQ média obtida com a reconstrução das 50 frases de teste utilizando os diversos *codebooks* produzidos a partir das frases de treinamento. Na legenda desse eixo aparece "PESQwb" para deixar explícito que foi empregada a implementação do algoritmo PESQ para sinais de banda larga (ou *wideband* em inglês, de onde vem o índice "wb").

Nessas figuras e nas subsequentes (Figuras 6.2, 6.3 e 6.5), para facilitar a identificação de cada uma das curvas, a cor das linhas e dos marcadores e o formato dos marcadores de cada ponto distinguem os algoritmos. Dessa forma, mesmo em uma impressão em preto-e-branco, o marcador é suficiente para a discriminação correta dos algoritmos. Já o tipo de linha que interpola os pontos serve para discriminar o tipo de parâmetro utilizado. Na Figura 6.1, por exemplo, o algoritmo *k*-médias aparece de preto, com um círculo (•) como marcador, a escolha aleatória de protótipos aparece de vermelho, com um asterisco (*) como marcador, e o caso sem compressão aparece de azul, com um losango de marcador. Os resultados para os parâmetros LPC utilizam uma linha contínua para ligar os pontos, enquanto os resultados para os parâmetros LSF utilizam uma linha não-contínua, que intercala traço e ponto.

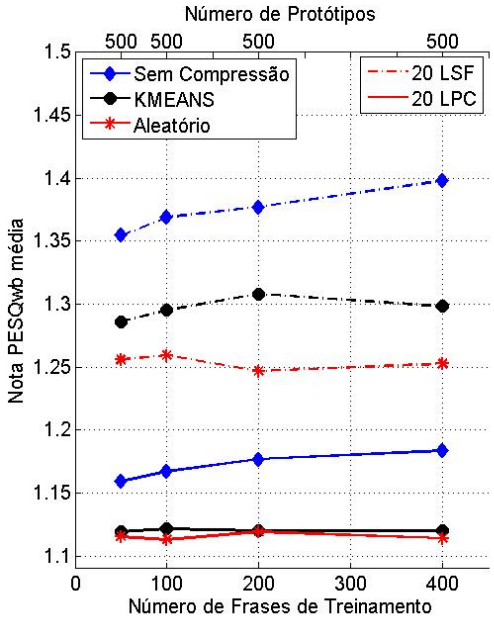
Analisando, primeiramente, apenas os gráficos referentes às taxas de compressão fixas (Figuras 6.1(a) e 6.1(b)), pode-se observar uma tendência de crescimento em todas as curvas apresentadas. Este comportamento era esperado, pois a manutenção da taxa de compressão implica em um aumento do número de quadros no dicionário, uma vez que a base de treinamento também cresce [2]. Com mais quadros disponíveis para a quantização do sinal, melhor tende a ser a qualidade do sinal sintetizado.

[2]Lembre-se que se denomina de taxa de compressão a relação entre o número de quadros da base de treinamento e o número de quadros do dicionário de quadros.



(a) Taxa de compressão de 100 vezes.

(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.1: **Resultado do *k*-médias, utilizando dois conjuntos de parâmetros distintos: LPC (—) e LSF (- · -).**

Por esse mesmo motivo, percebe-se que os resultados para a taxa de compressão de 200 vezes são piores dos que os obtidos para a taxa de compressão de 100 vezes, onde há o dobro do número de quadros.

Observando, agora, o gráfico referente ao caso em que o tamanho do *codebook* foi mantido fixo, com 500 *codevectors* (Figura 6.1(c)), pode-se admitir que as curvas são quase horizontais. Conclui-se que, apesar da maior disponibilidade de vetores para se decidir quais irão compor o *codebook*, o algoritmo não selecionou um conjunto de *codevectors* melhor. Portanto, pode-se afirmar que, dado o tamanho limitado de *codebook* exigido, não foi possível melhorar sua qualidade, mesmo quando havia maior diversidade de opções para escolha.

Focando a análise no desempenho do algoritmo *k*-médias, chama a atenção o fato de o algoritmo ter praticamente empatado com a escolha aleatória, quando se aplicaram os coeficientes LPC. Para esse conjunto de parâmetros, a utilização da base de treinamento completa propiciou um ganho de qualidade apenas modesto. Juntamente com o fato de a qualidade obtida ter sido bastante ruim, isso sugere que tais coeficientes não são aptos a discriminar corretamente os quadro de fala, ao menos para a aplicação testada.

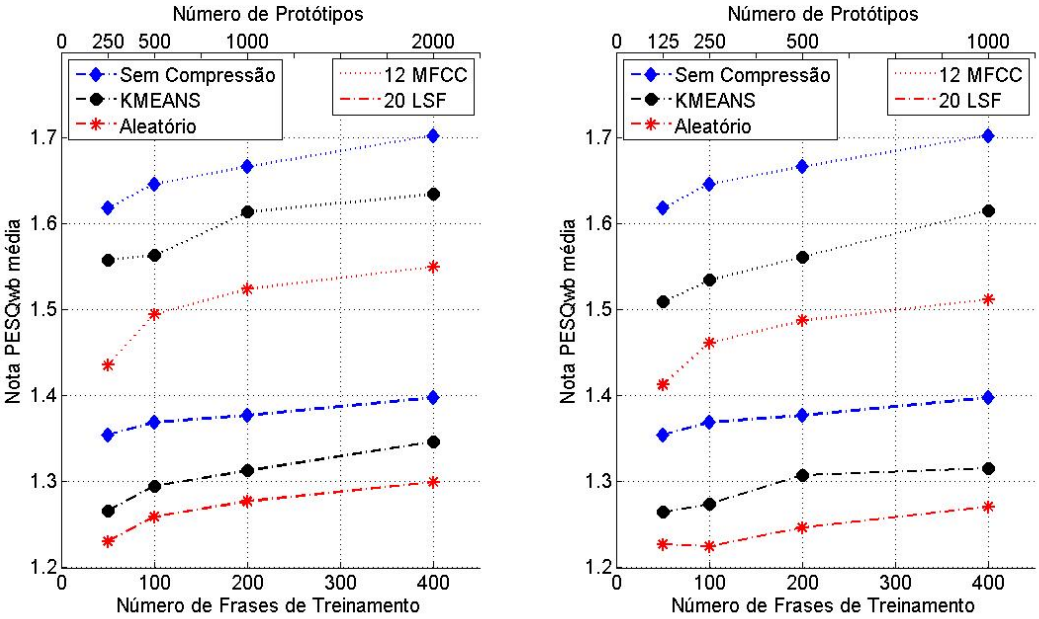
Quando se utilizou os coeficientes LSF, o desempenho do algoritmo *k*-médias foi diferente, revelando que o algoritmo foi capaz de escolher vetores mais representativos do universo disponível, o que refletiu em um ganho de qualidade comparado com a escolha aleatória. Nesta aplicação, o papel dos algoritmos de quantização vetorial é tentar obter uma curva o mais próxima possível da curva "Sem Compressão", enquanto o estudo de técnicas de processamento de sinais atuaria para tentar "puxar" essas curvas para cima. O foco deste trabalho está na análise dos algoritmos.

Por fim, comparando os dois tipos de parâmetros avaliados, os resultados gerados com os coeficientes LSF foram superiores aos gerados com os coeficientes LPC. Isso era esperado, dada a reconhecida maior robustez dos coeficientes LSF à quantização.

Observe, agora, os resultados dos coeficientes LSF comparados com os resultados dos coeficientes MFCC, mostrados nas Figuras 6.2(a), 6.2(b) e 6.2(c). Foi mantida a mesma relação de cores, marcadores e linhas da figura anterior (Figura 6.1), exceto pelas linhas utilizadas nas curvas de resultado dos coeficientes MFCC, que são pontilhadas.

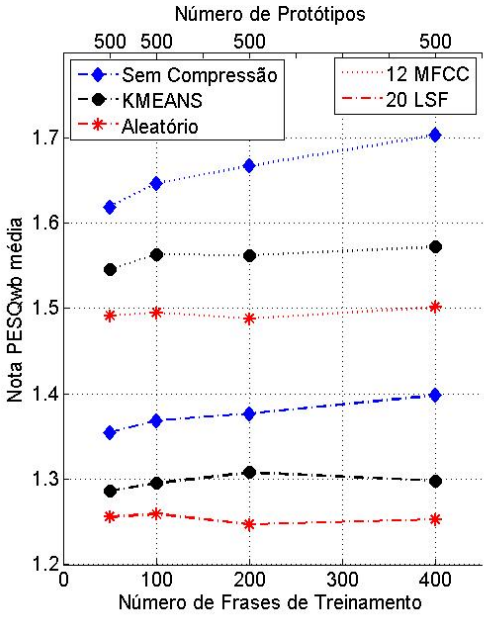
A análise feita para os resultados apresentados na Figura 6.1 também é válida para os resultados obtidos com os coeficientes MFCC. O destaque, neste caso, fica para o fato de a utilização destes coeficientes ter produzido resultados superiores aos produzidos pelos coeficientes LSF.

Os coeficientes MFCC são largamente utilizados em aplicações de reconhecimento de fala, em que já se mostraram muito eficientes (Taylor, 2009; Benesty et al., 2008). Daí, infere-se que eles são capazes de discriminar com certa precisão os sons de cada quadro. Portanto, também era esperado que eles serviriam para a aplicação aqui testada, na qual deseja-se trocar quadros de um sinal da forma



(a) Taxa de compressão de 100 vezes.

(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.2: **Resultado do *k*-médias, utilizando dois conjuntos de parâmetros distintos: MFCC (· · ·) e LSF (- · -).**

mais imperceptível.

Contudo, os resultados obtidos com esses parâmetros mais tradicionais (LPC, LSF e MFCC), foram muito pobres. Isso serviu de motivação para testar os coeficientes mel (ver Seção 5.5.2), bem menos frequentemente encontrados na literatura. O número de coeficientes mel utilizado é definido pela taxa de amostragem do sinal. Neste caso, os sinais foram amostrados a 16 kHz e na faixa de frequência até 8 kHz há 24 filtros de banda-crítica (Picone, 1993), ilustrados na Figura 5.8. Portanto, foram calculados 24 coeficientes mel para cada quadro.

As Figuras 6.3(a), 6.3(b) e 6.3(c) comparam a qualidade obtida com os coeficientes mel apenas com as curvas obtidas com os MFCC, pois foi o melhor resultado dentre os três conjuntos de parâmetros testados inicialmente.

Mais uma vez, observa-se um ganho de qualidade com o novo conjunto de parâmetros testado. Contudo, diferentemente do comportamento observado anteriormente (na comparação dos coeficientes LPC, LSF e MFCC), esse ganho não foi tão grande, de tal forma que algumas curvas se sobrepõem, dificultando, mas não impedindo, sua visualização. Ao final desta seção, serão apresentados gráficos comparando os conjuntos de parâmetros testados para cada método individualmente ("Sem Compressão", escolha aleatória e *k*-médias), facilitando a visualização da evolução dos resultados com a mudança dos parâmetros empregados (ver Figura 6.6).

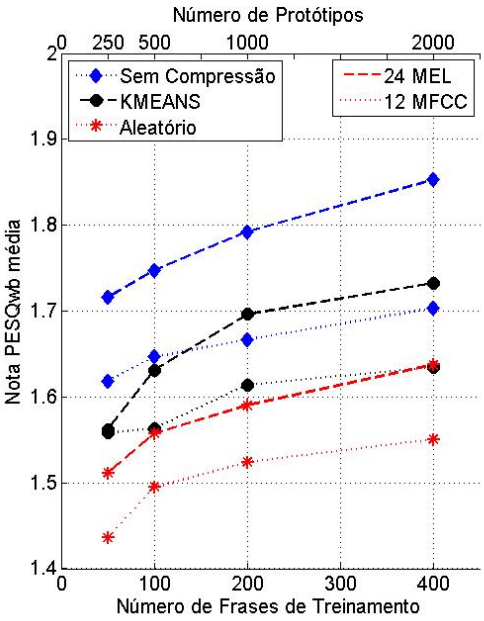
Conclui-se, dados os resultados apresentados nas Figuras 6.1, 6.2 e 6.3, que os 24 coeficientes mel foram os atributos que levaram ao resultado de melhor qualidade.

Adicionalmente, após observar que os coeficientes mel apresentaram o melhor resultado, acrescentou-se a esse vetor de parâmetros outras informações consideradas importantes: a energia (En) e a frequência fundamental (F_0). Na realidade, não foram inseridas exatamente a energia e o F_0. A extração de F_0 de um sinal é bastante complexa e é objeto de muita pesquisa. Neste trabalho, utilizou-se a marcação de *pitch* fornecida, como uma estimativa da frequência fundamental, e decidiu-se então pelo período esquerdo (PE) do quadro como uma aproximação de F_0. Note que o conceito de frequência fundamental só pode ser relacionado a quadros vozeados, mas, como nos trechos não vozeados a marcação de *pitch* segue o mesmo padrão de espaçamento fixo entre as marcas, esse fato não interfere no resultado [3].

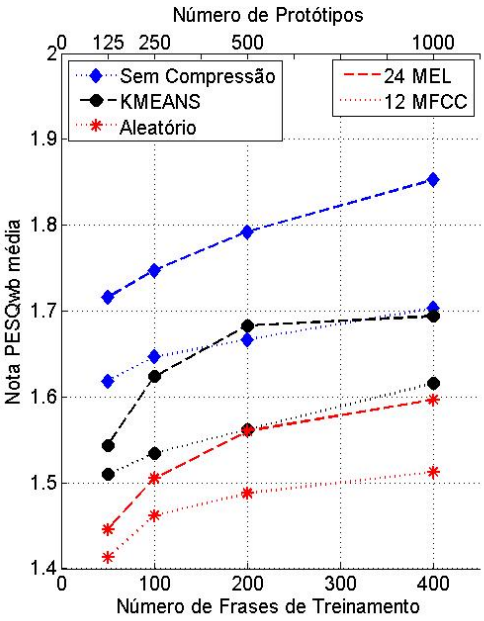
Para a inserção desses novos parâmetros, deve-se considerar o problema da sua escala numérica. O período esquerdo é dado em número de amostras, que apresenta valores muito maiores do que os dos coeficientes mel. Já a energia apresenta valores mais próximos aos dos coeficientes mel, afinal, estes são a energia do sinal após passar por um dos filtros de banda-crítica.

A forma mais simples de contornar essas difereças seria normalizar os dados, ou seja, fazer com que a média fosse nula e a variância unitária para cada um dos parâmetros, quando tomados da

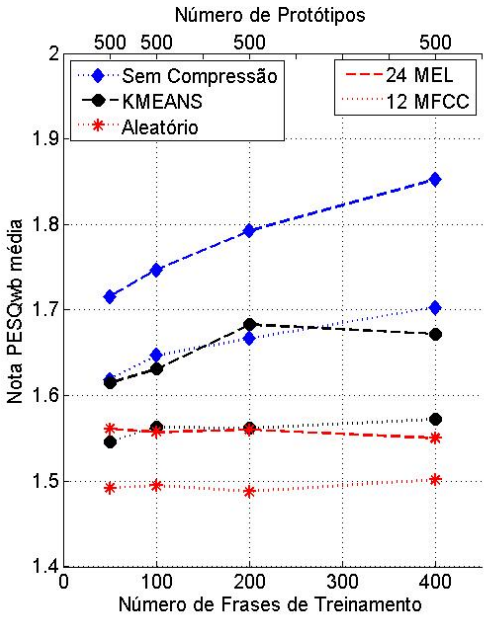
[3] Considera-se que a marcação de *pitch* foi feita adequadamente.



(a) Taxa de compressão de 100 vezes.

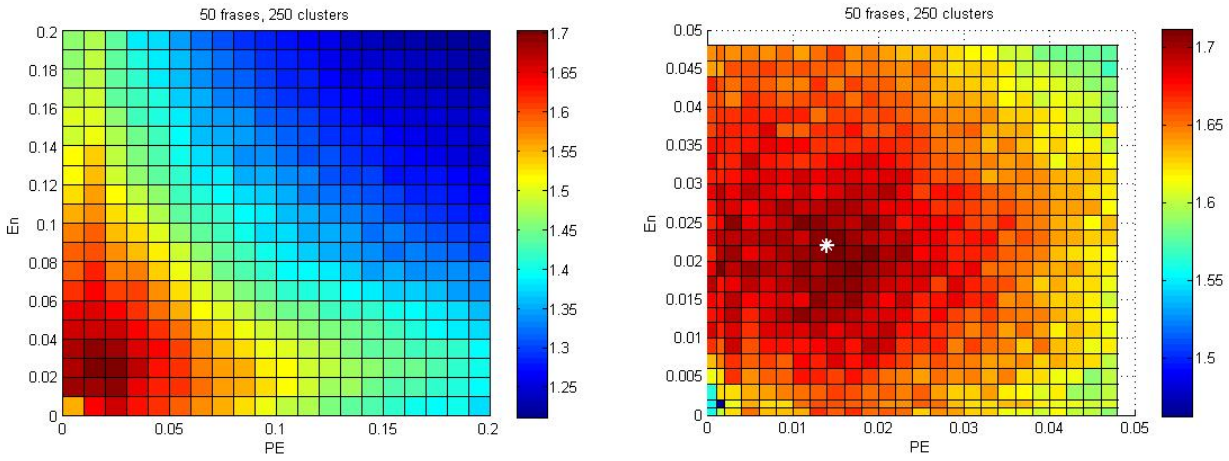


(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.3: **Resultado do *k*-médias, utilizando dois conjuntos de parâmetros distintos: MFCC (· · ·) e MEL (- - -).**



(a) (b)

Fig. 6.4: **Resultado da otimização dos pesos de normalização da energia (En) e do período esquerdo (PE), utilizando a configuração de 50 frases de treinamento e *codebook* com 250 protótipos. 6.4(a) Resultado em uma região larga. 6.4(b) Zoom na melhor região.**

base inteira. Inicialmente testou-se normalizar apenas os coeficientes mel, para avaliar o efeito desse procedimento no resultado. Tal teste revelou resultados piores do que os obtidos sem a normalização. Esse opção foi, então, descartada e decidiu-se normalizar apenas a energia e o período esquerdo e, em seguida, multiplicá-los por pesos que foram ajustados para otimizar o resultado.

Essa estratégia poderia ser aplicada para todos os parâmetros, mas nesse caso, ao invés de otimizar apenas dois pesos, seria necessário otimizar 26, o que é muito mais desafiador. Como os valores dos coeficientes mel já apresentam uma relação que se deseja preservar, não há motivo para investir nessa tarefa.

Os dois pesos foram otimizados através de uma busca "exaustiva", variando-se ambos os valores dentro de certos limites. Esses limites também foram definidos empiricamente, com alguns testes iniciais. O que se está chamando de busca "exaustiva" é, na realidade, uma busca em *grid*, pois no espaço contínuo uma busca literalmente exaustiva implica em testar os infinitos valores possíveis em qualquer intervalo dado. Por isso, dados os limites, definiu-se um tamanho de passo para variar o valor dos pesos. Uma vez detectada a melhor região, esse passo foi reduzido, formando uma espécie de *zoom* para encontrar o ótimo. A Figura 6.4 ilustra o resultado obtido, onde a cor indica a nota PESQwb obtida: quanto mais vermelho, maior a nota, quanto mais azul, menor a nota. Os pesos ótimos encontrados estão marcados na figura com um asterisco branco e são 0,014 para o período esquerdo e 0,022 para a energia.

É importante observar que essa otimização dos pesos de normalização foi realizada em apenas

uma das configurações testadas, na qual se utilizavam 50 frases de treinamento e *codebook* com 250 protótipos. Essa decisão foi tomada devido ao alto custo computacional dessa otimização, já que para cada avaliação de uma dupla de pesos, foram executadas 5 simulações a fim de obter uma média. Depois, esses mesmos pesos foram aplicados em todas as outras configurações avaliadas.

As Figuras 6.5(a), 6.5(b) e 6.5(c) mostram o ganho de qualidade obtido com a inserção desses dois parâmetros.

Analisando as curvas "Sem Compressão", percebe-se o grande potencial de ganho auferido com a inserção desses dois atributos (En e PE). Entretanto, tanto a escolha aleatória quanto o algoritmo *k*-médias não foram capazes de explorar esse potencial, produzindo sim ganhos de qualidade, mas de menor monta.

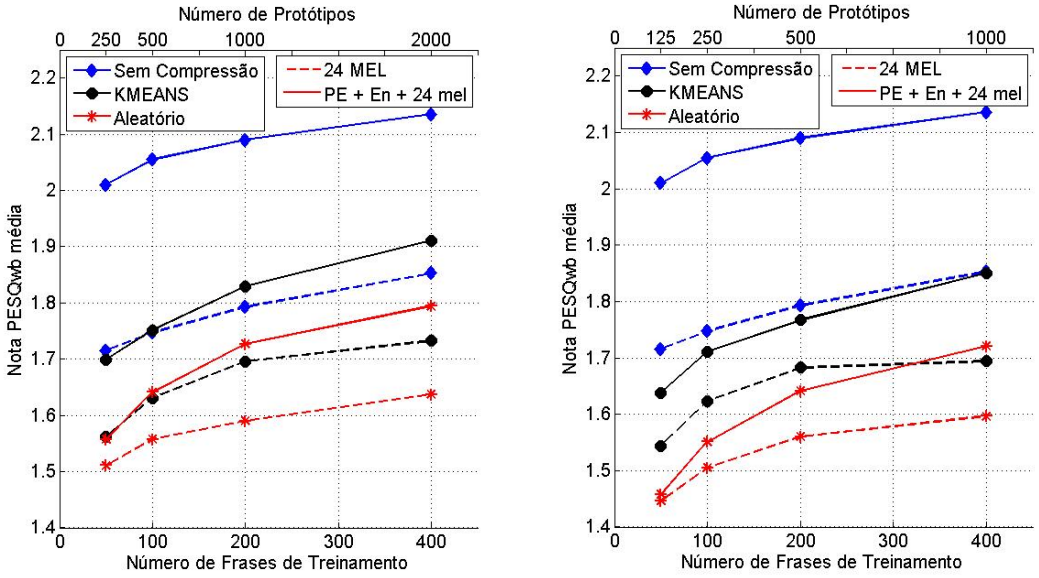
Por fim, a Figura 6.6 apresenta esses mesmos resultados agrupando agora as curvas dos diferentes parâmetros em uma mesma figura, para cada um dos métodos testados ("Sem Compressão", escolha aleatória e *k*-médias). Através desses gráficos, fica mais fácil visualizar a evolução obtida. Apenas o gráfico da Figura 6.6(a) não aparece na mesma escala para a nota PESQwb que os demais gráficos.

Apesar de a leitura ficar comprometida, as figuras foram feitas intencionalmente pequenas, para que todos os gráficos ficassem na mesma página, facilitando a comparação. Repetindo, aqui não há informação nova, uma vez que os valores estão todos presentes nas figuras anteriores dessa seção. O objetivo dessa figura é permitir a visualização da evolução obtida, envolvendo todos os parâmetros testados.

Concluindo essa seção, o vetor de parâmetros constituído de energia e período esquerdo, normalizados e ponderados, mais os 24 coeficientes mel, produziu os melhores resultados e foi empregado para avaliar o desempenho dos outros algoritmos. Este é o tema da próxima seção.

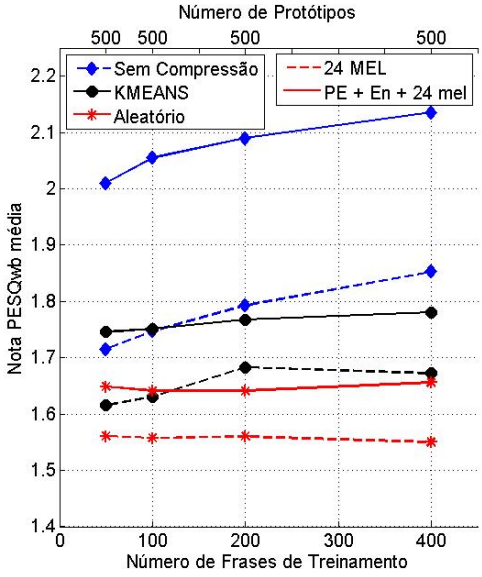
6.3 Avaliando Diferentes Algoritmos de Quantização Vetorial

Na seção anterior, foram avaliados diferentes conjuntos de parâmetros, com o propósito de identificar qual produziria os melhores resultados. Para aqueles testes, foi utilizado apenas o algoritmo *k*-médias. Agora já se sabe que os 24 coeficientes mel, juntamente com o período esquerdo e a energia, devidamente normalizados e ponderados, provêem melhor qualidade aos sinais de fala sintetizados. Nesta seção, analisa-se o desempenho dos outros algoritmos de quantização vetorial, aplicados unicamente a este conjunto de parâmetros "ótimo". Os algoritmos avaliados estão descritos no Capítulo 3 e são, além do *k*-médias, o ARIA e o NG (*Neural-Gas*).



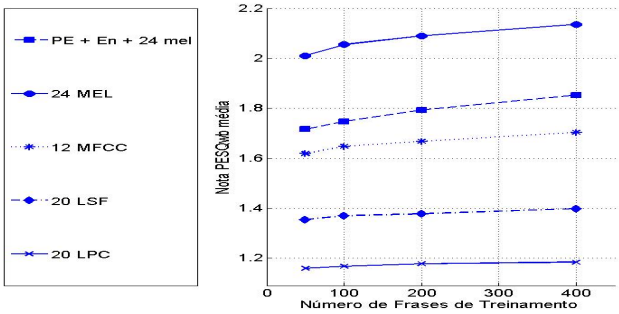
(a) Taxa de compressão de 100 vezes.

(b) Taxa de compressão de 200 vezes.

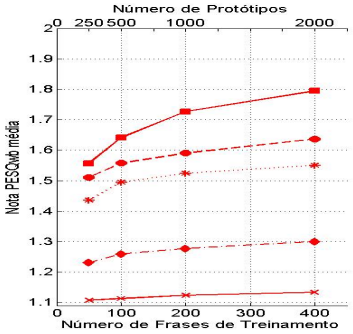


(c) Número de protótipos fixo (500).

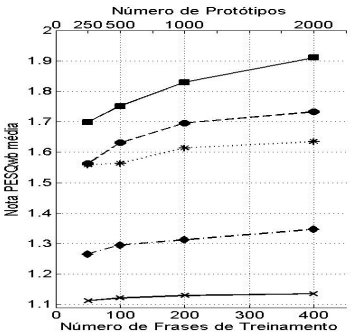
Fig. 6.5: **Resultado do *k*-médias, utilizando dois conjuntos de parâmetros distintos: MEL (- - -) e PE + En + MEL(—).**



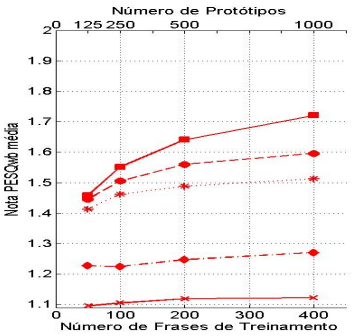
(a) “Sem Compressão”.



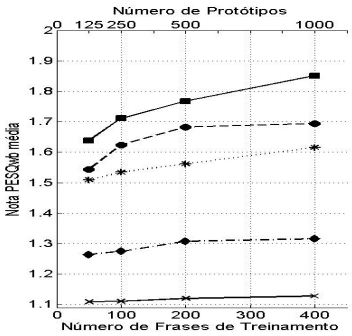
(b) Escolha aleatória de protótipos, para uma taxa de compressão fixa de 100 vezes.



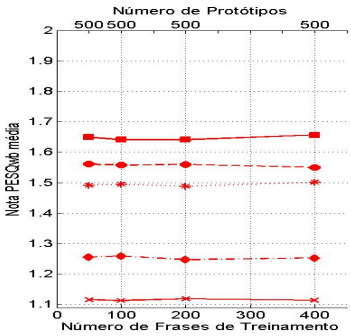
(c) Algoritmo *k*-médias, para uma taxa de compressão fixa de 100 vezes.



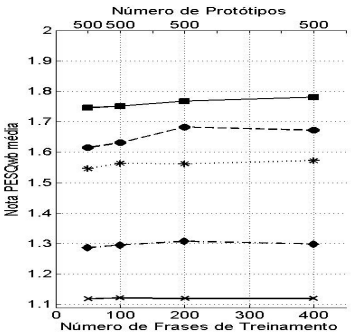
(d) Escolha aleatória de protótipos, para uma taxa de compressão fixa de 200 vezes.



(e) Algoritmo *k*-médias, para uma taxa de compressão fixa de 200 vezes.



(f) Escolha aleatória de protótipos, para número de protótipos fixo (500).



(g) Algoritmo *k*-médias, para número de protótipos fixo (500).

Fig. 6.6: **Resultado comparativo dos conjuntos de atributos testados (20 LPC, 20 LSF, 12 MFCC, 24 MEL e PE + En + 24 MEL).**

6.3.1 Configuração do NG

Para a operação do algoritmo NG é necessário definir três decaimentos. Foram utilizadas as equações disponíveis no próprio código do NG, no SOM Toolbox, para definir esses decaimentos, as quais estão copiadas aqui, nas equações 6.1, 6.2 e 6.3. A Equação 6.1 descreve o decaimento do passo que cada neurônio dá na direção do dado, conforme seu *ranking* de distância a esse dado. Já a Equação 6.2 descreve o decaimento da intensidade com que esse primeiro decaimento ocorre, ou seja, os neurônios mais mal ranqueados passam a deslocar cada vez menos. A Equação 6.3 descreve o decaimento do passo máximo permitido. A atualização da posição de um neurônio é definida pela Equação 6.4. Vale destacar que λ, α e, portanto, h são atualizados para cada padrão de entrada apresentado à rede e não apenas a cada nova iteração.

$$h = e^{\frac{-ranking}{\lambda(i)}} \tag{6.1}$$

$$\lambda(i) = \lambda_0 \left(\frac{0,01}{\lambda_0}\right)^{\frac{i-1}{L_{tr}}} \tag{6.2}$$

$$\alpha(i) = \alpha_0 \left(\frac{0,005}{\alpha_0}\right)^{\frac{i-1}{L_{tr}}} \tag{6.3}$$

$$Neurônio_k = Neurônio_k + \alpha(i)\ h\ (padrao\ de\ entrada - Neurônio_k) \tag{6.4}$$

As constantes λ_0 e α_0 são valores definidos pelo usuário e a constante L_{tr} (de "comprimento do treinamento", do inglês *training length*) é calculada a partir da multiplicação do número de dados de treinamento pelo número de iterações do algoritmo, que também é definido pelo usuário. O índice i é incrementado a cada padrão de entrada apresentado à rede e, por isso, vai de 1 a L_{tr}. Ou seja, a cada padrão de entrada, é utilizado um valor diferente tanto de λ quanto de α, sempre menores que o anterior.

Para entender o efeito dos três decaimentos em conjunto, considere, por exemplo, a relação entre o passo dado pelo neurônio vencedor, para um certo padrão de entrada em uma certa iteração, e o passo do segundo melhor neurônio. Por definição, o *ranking* do neurônio vencedor é definido como sendo 0 e, portanto, h será sempre igual a 1 para o neurônio vencedor. O valor de h será sempre menor para o segundo melhor colocado, cujo *ranking* é 1, pois é isso que descreve a Equação 6.1. O fato do valor de λ também decair, conforme descreve a Equação 6.2, implica que, para cada novo padrão de entrada apresentado à rede, o valor de h do segundo colocado será cada vez menor.

O tamanho do passo dado, quando a posição do neurônio for ser atualizada, sofre ainda outro decaimento, pois o valor de h é multiplicado por α, que também decai (este é o terceiro decaimento)

a cada novo padrão de entrada apresentado à rede. Ou seja, o passo dado pelo neurônio vencedor, por exemplo, para o primeiro padrão apresentado à rede, na primeira iteração será, na verdade, igual a α_0 e cada vez menor daí em diante.

O mesmo código sugere valores para as constantes λ_0 e α_0, sendo que, para a constante λ_0, o valor é dado pela metade do número de neurônios, e, para a constante α_0, o valor é $0,5$. Neste trabalho, foram utilizadas exatamente essas configurações, exceto pelo valor de α_0 que foi colocado em $0,25$. Esses valores foram determinados em testes preliminares e levaram a resultados satisfatórios, como será visto na Seção 6.3.3.

Completando a descrição da configuração adotada, o número de iterações foi definido em 15, pois esse número se mostrou suficiente para a convergência do algoritmo (o erro de quantização se estabilizava). Na inicialização, assim como foi feito para o *k*-médias, os neurônios foram escolhidos aleatoriamente entre os padrões de entrada.

6.3.2 Configuração do ARIA

Descrevem-se, agora, as configurações utilizadas para o algoritmo ARIA. Todas elas foram determinadas empiricamente. A população inicial de anticorpos foi, novamente, escolhida aleatoriamente dentre os antígenos, do mesmo modo como fora feito para os algoritmos *k*-médias e NG. Mas, no caso do ARIA, o número de anticorpos é auto-ajustável e, portanto, não é necessário utilizar uma população inicial com o número de protótipos desejado. Na realidade, é interessante começar com poucos anticorpos, para que o algoritmo se adapte e gere o número adequado, produzindo mais anticorpos onde for mais necessário, e removendo-os onde eles não estão contribuindo. O tamanho da população inicial empregado foi $n = 20$.

A taxa de mutação inicial foi $\mu = 1$ e sua redução foi iniciada logo na primeira iteração, com uma constante de decaimento geométrico $c = 0,95$. O raio E, o qual define a vizinhaça para o cálculo da densidade local de dados, foi inicializado com valor igual a $2\ r$. A constante r é o raio mínimo permitido aos anticorpos e seu valor depende da configuração do teste em questão. Os valores utilizados serão apresentados adiante. Os raios iniciais dos anticorpos, que também dependem do valor de r, foram determinados calculando-se a densidade local inicial deles, utilizando o valor de E inicial, e então foi empregada a fórmula de cálculo do raio em função da densidade local, dada na Equação 3.2.

O critério de parada adotado para o ARIA foi de duas uma: ou era atingido o número máximo de iterações, definido em $max_{it} = 60$, ou o algoritmo convergia antes disso e era terminado. Esse procedimento não está descrito na versão original do ARIA, que adotava unicamente como critério de parada o número máximo de iterações.

O critério de convergência aplicado foi avaliar se o tamanho da rede de anticorpos havia se esta-

bilizado e se a movimentação dos anticorpos também. A movimentação da rede foi calculada pela média da diferença da posição dos anticorpos antes e depois da etapa de maturação de afinidade. Considerou-se que tanto o tamanho da população quanto sua movimentação haviam se estabilizado caso seu valor, ao final da iteração corrente, estivesse próximo o suficiente da média de seus valores nas últimas iterações. Essa média foi tomada das últimas 4 iterações e incluiu o valor atual, e o valor de "próximo o suficiente" foi definido em 10 para o tamanho da população e em $0,001$ para a movimentação da rede, lembrando que esses valores foram determinados empiricamente.

Isso quer dizer que, por exemplo, se o tamanho x_t da população de anticorpos ao final de certa geração t for igual à média y dos tamanhos da população nas últimas 4 iterações e do seu valor atual ($y = 1/5\ (x_t + x_{t-1} + x_{t-2} + x_{t-3} + x_{t-4})$), mais ou menos 10, ou seja, $y - 10 \leq x_t \leq y + 10$, o critério de convergência do tamanho da população foi atingido. Para que o fim do algoritmo seja determinado, é necessário que os dois critérios de convergência sejam atingidos. Para evitar uma parada prematura inesperada, também foi imposto um número mínimo de 30 gerações.

Antes de apresentar os resultados obtidos, mais uma consideração deve ser feita a respeito do ARIA. O valor do raio mínimo r influencia no tamanho final da população de anticorpos, pois é ele que controla o valor dos raios dos anticorpos, que por sua vez são usados nos processos de expansão clonal e supressão. Ora, como são buscados tamanhos de *codebook* diferentes, partindo de bases de tamanhos diferentes, o valor de r deve ser ajustado para cada caso.

No entanto, não há um mecanismo que estime a priori o resultado que será obtido e, por isso, foram necessárias simulações com diversos valores de r para se chegar ao número desejado. Mais do que isso, o algoritmo se mostrou bastante instável, pois empregando a mesma configuração inicial, muitas vezes os resultados foram populações de tamanhos bastante diferentes. Assim, atingir o tamanho exato de *codebook* seria bastante custoso. Felizmente não é necessário que esse valor seja exato, afinal um *codebook* com tamanho de 2000 quadros e outro de 1970, por exemplo, não são significativamente diferentes.

Lembrando que se quer obter uma média de cinco resultados, foram executadas quantas simulações do ARIA fossem necessárias, para que o tamanho final da população de anticorpos fosse, em cinco casos, igual ao número de protótipos desejado, $\pm 5\%$. A Tabela 6.5 mostra os intervalos aceitáveis, o valor dos raios mínimos r empregados em cada configuração de teste para atingir o número de protótipos dentro desse intervalo e o valor médio dos tamanhos de *codebook* obtidos.

6.3.3 Resultados

Finalmente, apresentam-se, na Figura 6.7, os resultados obtidos pelos algoritmos *k*-médias, NG e ARIA, pela escolha aleatória de protótipos e pelo *codebook* formado pelas bases de treinameto inteiras. Nos gráficos, a curva de cada algoritmo foi feita com um trio diferente de: (*i*) cor, (*ii*)

Tab. 6.5: **Número médio de protótipos obtido pelo algoritmo ARIA e raio mínimo r utilizado, para cada configuração de teste.**

Número de Frases de Treinamento	Número de Protótipos				r Utilizado
	Mínimo	Desejado	Máximo	Obtido	
50	118	125	132	125,4	0,0102
50	237	250	263	246,4	0,0087
50	475	500	525	513,2	0,0080
100	237	250	263	250,0	0,0085
100	475	500	525	509,6	0,0079
200	475	500	525	494,6	0,0078
200	950	1000	1050	1027,4	0,0072
400	475	500	525	494,8	0,0075
400	950	1000	1050	1015,4	0,0071
400	1900	2000	2100	1970,2	0,0067

formato do marcador e (*iii*) linha que interpola os pontos.

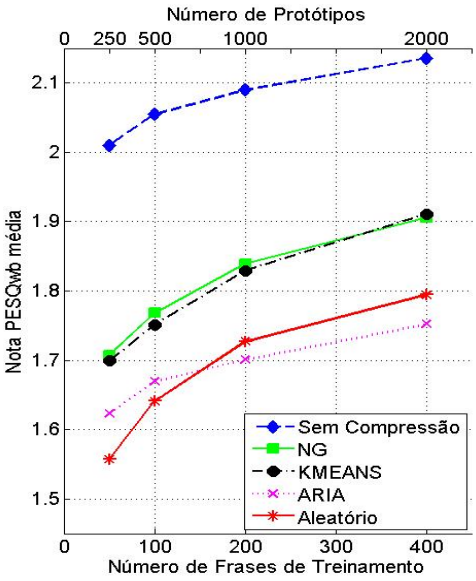
Comparando o desempenho do algoritmo *k*-médias com o algoritmo NG, pode-se afirmar que houve quase um empate, com o algoritmo NG ligeiramente melhor. Interessante que um algoritmo simples como o *k*-médias leva a resultados tão bons quanto o de outro mais elaborado. Mas é importante destacar que não foi feito um estudo elaborado dos parâmetros do algoritmo NG para otimizar seu resultado.

Analisando a Figura 6.7(c), repara-se que há uma tendência de crescimento das curvas desses dois algoritmos. Isso implica que, diferentemente do que foi observado nos resultados com os primeiros conjuntos de parâmetros, neste caso, os algoritmos foram capazes de explorar melhor a maior diversidade de dados, decorrente do aumento da base de treinamento, na escolha de quadros para compor o dicionário. Esse comportamente já poderia ter sido notado para o algoritmo *k*-médias, no caso do conjunto de parâmetros formado apenas pelos 24 coeficientes mel e, em menor escala, no caso dos 12 MFCC (ver Figura 6.3(c)).

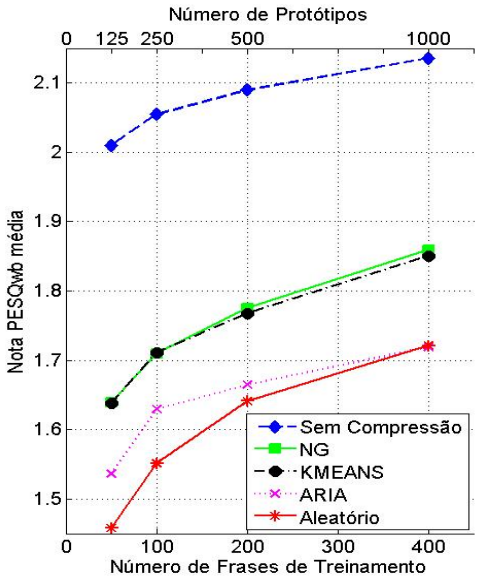
O destaque negativo fica por conta do resultado do ARIA, que apresentou um resultado distante do obtido pelos outros algoritmos e muito próximo do obtido pela escolha aleatória, chegando, em algumas situações, a perder para a escolha aleatória. Conforme destacado no início deste capítulo, isso significa que o ARIA, ao invés de escolher dados representativos do universo disponível, escolheu dados pouco representativos para compor o *codebook*.

Investigou-se, então, o que poderia estar causando esse comportamento distorcido. Analisando a distribuição do número de dados representado por cada protótipo, foram gerados os histogramas da Figura 6.8.

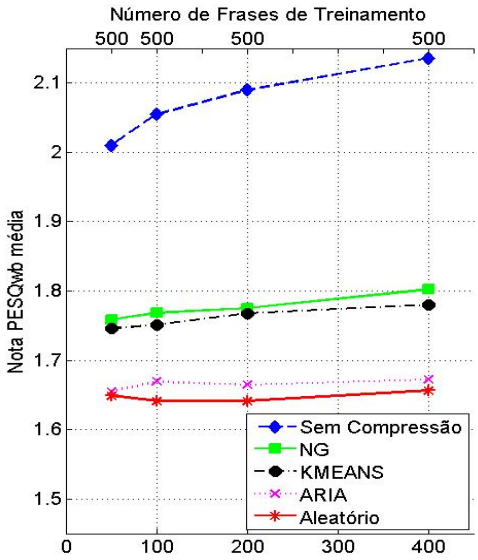
Apresenta-se, aqui, o resultado para apenas uma configuração de teste, mas essa análise foi feita



(a) Taxa de compressão de 100 vezes.



(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.7: **Resultado dos algoritmos *k*-médias, NG e ARIA, da escolha aleatória de protótipos e do *codebook* formado pelas bases de treinamento inteiras ("Sem Compressão"), utilizando o conjunto de parâmetros PE + En + 24 mel.**

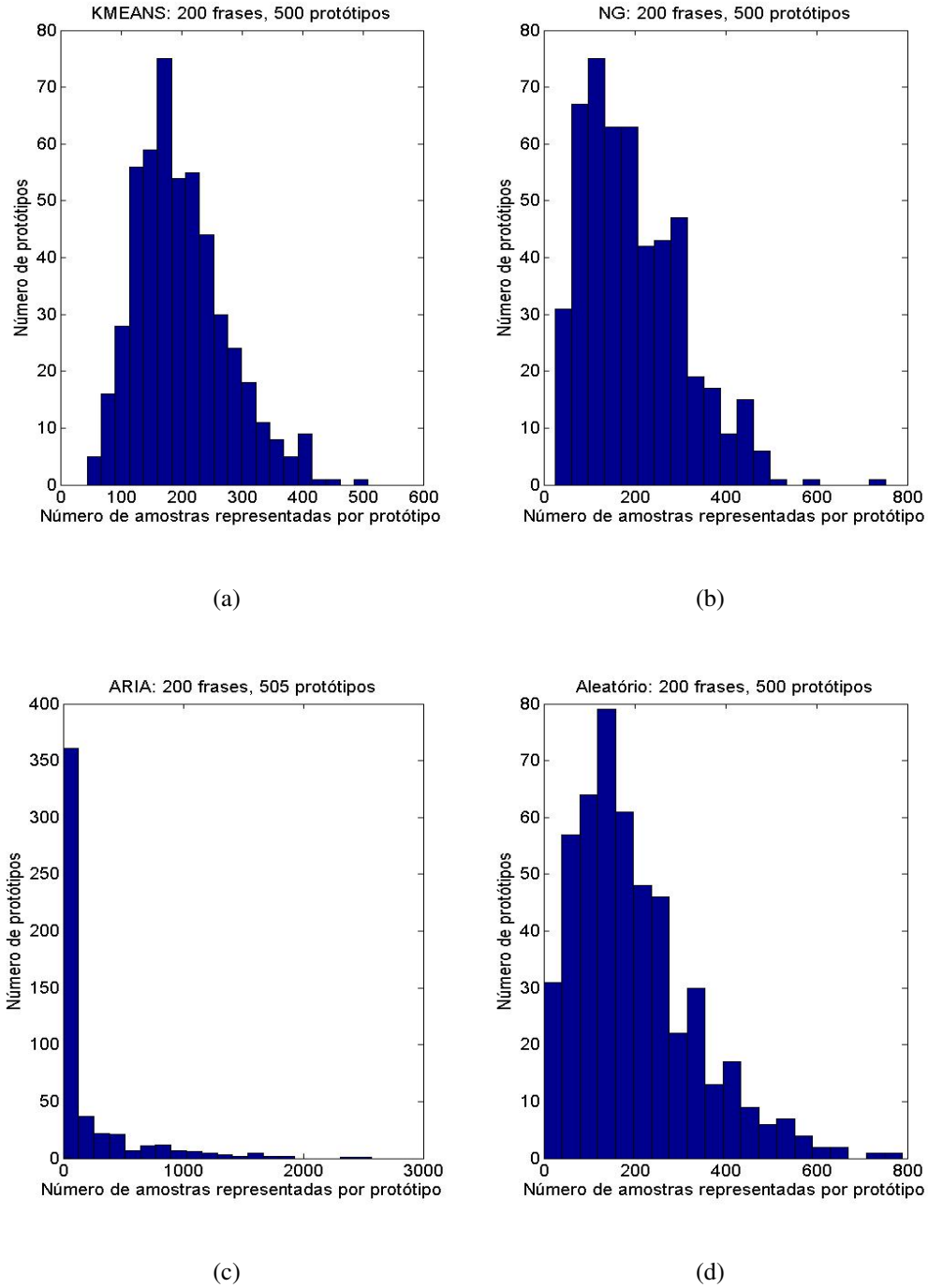


Fig. 6.8: **Histograma do número de amostras de entrada que cada protótipo representa, para a configuração de teste com 200 frases de treinamento e *codebook* com 500 *codevectors*, para os algoritmos: 6.8(a) *k*-médias. 6.8(b) NG. 6.8(c) ARIA. 6.8(d) Escolha aleatória.**

para todas as configurações e os resultados foram equivalentes. Dentre as 5 simulações disponíveis de cada algoritmo, utilizou-se aquela que resultou na maior nota PESQwb.

Nitidamente, o ARIA produziu uma distribuição anômala de protótipos, muito diferente da dos outros algoritmos. Percebe-se um número grande de protótipos representando poucas amostras, indicado pela barra alta à esquerda do histograma, e um número pequeno de protótipos representando muitas amostras, indicado pela longa "cauda" no histograma, com várias barras baixas, à direita (ver Figura 6.8(c)). Essa característica se configura como o oposto do que se esperava obter, dada a anunciada sensibilidade do ARIA à densidade relativa dos grupos. Uma possível explicação está na alta dimensão do espaço (dimensão = 26) em que se encontram os dados, que alterou de forma inesperada a sensibilidade dos anticorpos à densidade local. Adicionalmente, a análise dos valores dos raios dos protótipos mostrou que os muitos protótipos representando poucas amostras encontravam-se nas regiões menos densas, pois apresentavam raios grandes quando comparados ao valor médio da população, enquanto os poucos protótipos representando muitas amostras estavam nas regiões mais densas (raios pequenos) (Violato et al., 2009).

Dados esses resultados, foram propostas modificações no algoritmo ARIA, que serão descritas nas próximas seções.

6.4 Primeira Proposta - Modificação no Cálculo do Raio

Na seção anterior, ficou claro que o ARIA apresentou grandes dificuldades frente aos testes realizados, chegando, em algumas situações, a produzir resultados de qualidade inferior à obtida até mesmo pela escolha aleatória de protótipos. Os histogramas forneceram uma indicação do que poderia estar causando essa degradação, revelando uma distribuição distorcida do número de amostras representadas por cada protótipo.

Para contornar esse problema, precisava-se de um método que estimulasse a supressão de anticorpos que representassem poucos dados e de um método que estimulasse a clonagem de anticorpos que representassem muitos dados. Surgiu então uma ideia que implementaria esses dois métodos com uma única modificação no algoritmo. Foi proposta uma nova fórmula para o cálculo do raio de cada anticorpo, dada pela Equação 6.5 (Violato et al., 2009).

$$R_i = r \left(\frac{den_{\max}}{den_i} \right)^{\frac{\kappa}{\dim}} \tag{6.5}$$

Dessa forma, quanto maior o valor de κ, maior será o crescimento do raio dos anticorpos posicionados em regiões menos densas, estimulando a ocorrência de supressão [4]. Além disso, essa proposta

[4] Atenção, não confundir este κ com o k no algoritmo k-médias, que indica o número de centroides utilizado.

admite a utilização de um raio mínimo menor, permitindo que mais anticorpos se posicionem nas regiões mais densas. Com isso, combatem-se as duas distorções citadas anteriormente. Uma desvantagem dessa nova fórmula é que se insere no algoritmo um novo parâmetro a ser configurado pelo usuário, para cada conjunto de parâmetros a se testar.

Então, para algumas configurações de teste, variou-se o valor de κ, com o intuito de estudar o efeito dessa variação no resultado final do algoritmo. Foi analisada a influência tanto na nota PESQwb quanto no histograma e, em todos os casos, chegou-se a resultados similares aos mostrados na Figura 6.9, que ilustra, novamente, o resultado quando foi usada a base de treinamento com 200 frases e *codebooks* com aproximadamente 500 *codevectors*.

Se o leitor reparar no histograma apresentado na Figura 6.9 para o ARIA com $\kappa = 1$, ou seja, sua formulação original, irá perceber que ele é diferente do mostrado na Figura 6.8(c). Isso ocorre, pois esses resultados foram obtidos em um momento anterior da pesquisa, quando o conjunto de parâmetros empregado ainda não incluia a energia e, no qual a normalização do período esquerdo era feita de outra forma. A ordem da apresentação dos resultados visa uma maior organização, a fim de facilitar o acompanhamento do texto, e não segue necessariamente a ordem cronológica dos fatos. Mas ressaltamos que isso não invalida o resultado obtido.

Comparando as notas PESQ obtidas para as configurações testadas, percebeu-se que os melhores resultados foram obtidos na maioria dos casos para $\kappa = 5$ e, em alguns casos, para $\kappa = 4$. Na Figura 6.9, repara-se que a nota PESQwb praticamente estabilizou para esses valores. Esse resultado foi suficiente para que se decidisse não fazer testes com valores maiores de κ. Repare que, para $\kappa = 5$, uma vez que $dim = 26$:

$$\frac{k}{\dim} = \frac{5}{26} \cong \frac{1}{5} \cong \frac{1}{\sqrt{26}} \tag{6.6}$$

Dada essa aproximação, utilizou-se a fórmula da Equação 6.7 no cálculo do raio dos anticorpos (Violato et al., 2009), com o propósito de gerar resultados para todas as configurações de teste, a fim de compará-los com os resultados previamente obtidos.

$$R_i = r \left(\frac{den_{\max}}{den_i} \right)^{\frac{1}{\sqrt{dim}}} \tag{6.7}$$

Essa fórmula evita a inserção do parâmetro κ no algoritmo e, ao menos para os dados específicos desse trabalho, é uma aproximação válida. Na Figura 6.10, são mostrados os resultados para as taxas de compressão fixas (100 e 200 vezes) e para os *codebooks* de tamanho fixo igual a 500.

Para chegar a esses resultados, os valores do raio mínimo r empregados foram diferentes dos utilizados na formulação original do algoritmo e estão na Tabela 6.6.

Como era esperado, os valores de r na Tabela 6.5 são maiores do que os valores da Tabela 6.6.

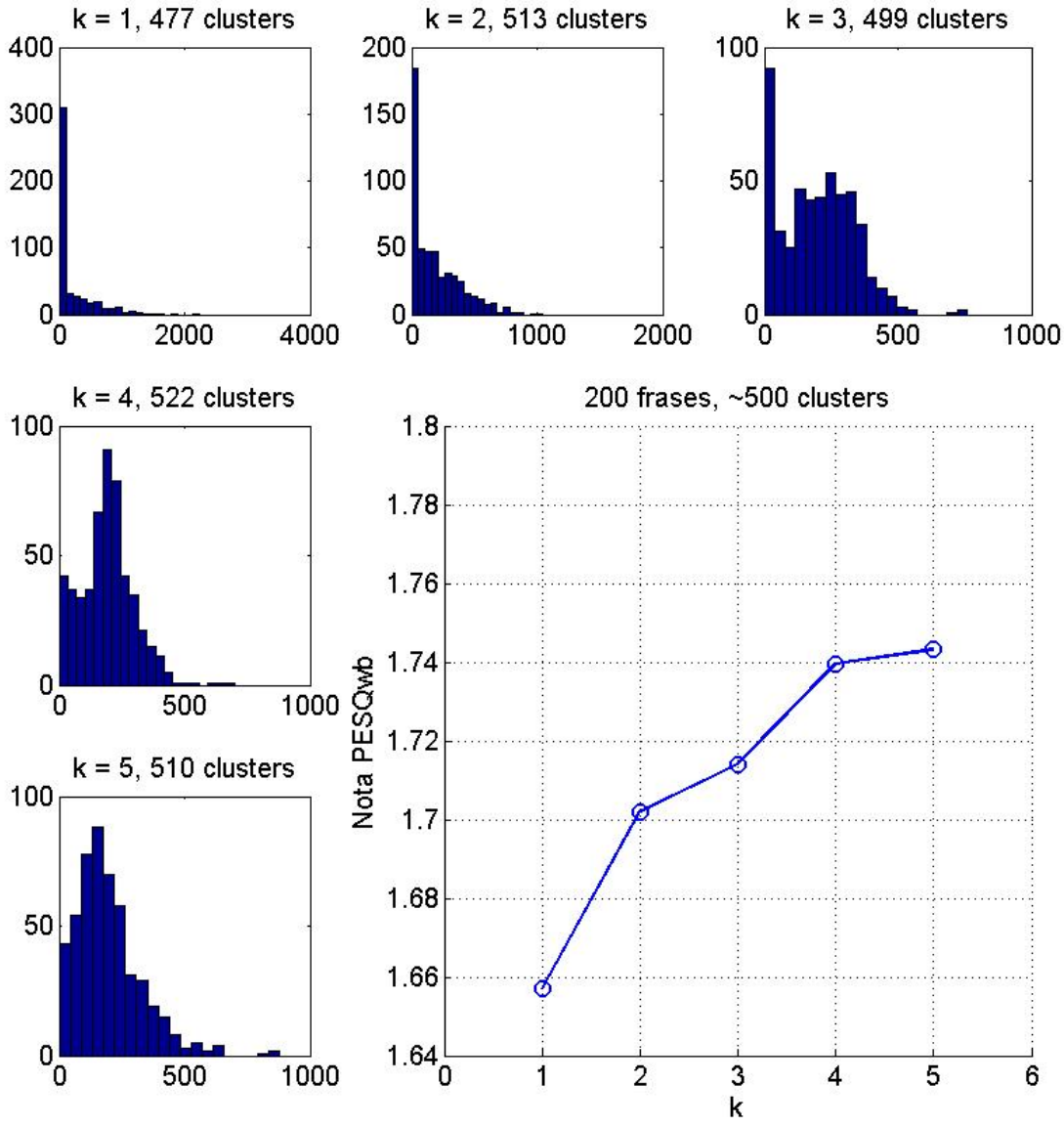
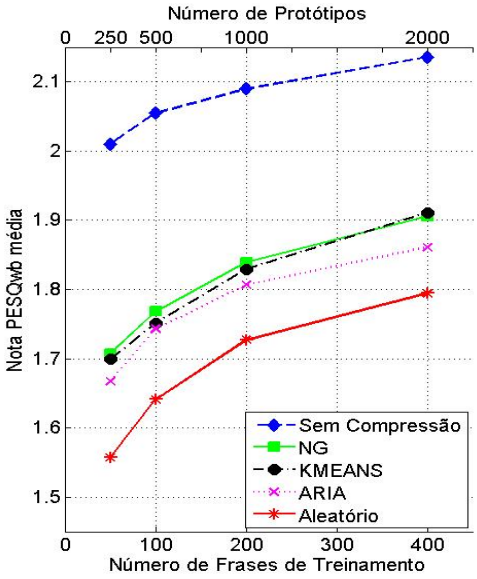
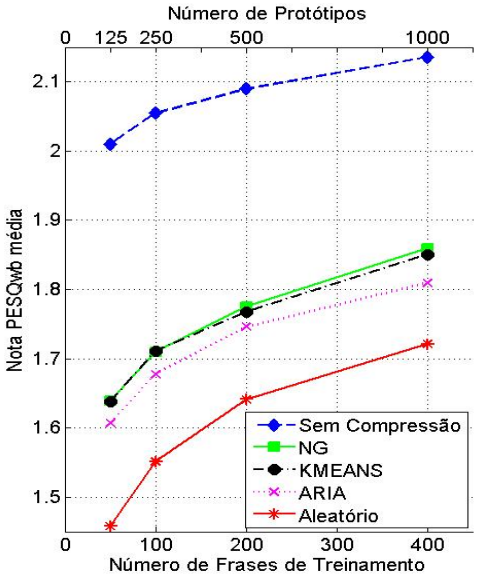


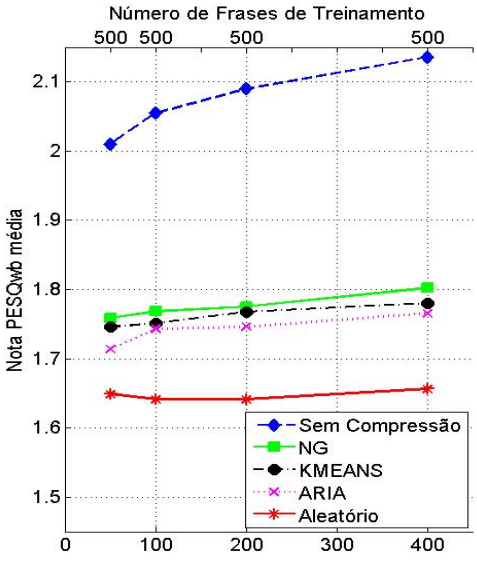
Fig. 6.9: **Resultados obtidos ao empregar a fórmula 6.5 no algoritmo ARIA, para diferentes valores de κ. A base de treinamento empregada continha 200 frases e o tamanho dos *codebooks* produzidos foi aproximadamente 500.**



(a) Taxa de compressão de 100 vezes.



(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.10: **Resultado dos algoritmos *k*-médias, NG e ARIA (utilizando a Equação 6.7 para cálculo do raio dos anticorpos), da escolha aleatória de protótipos e do *codebook* formado pelas bases de treinamento inteiras ("Sem Compressão"), utilizando o conjunto de parâmetros PE + En + 24 mel.**

Tab. 6.6: **Número médio de protótipos obtido pelo algoritmo ARIA, utilizando a Equação 6.7 para cálculo do raio dos anticorpos, e raio mínimo r utilizado.**

Número de Frases de Treinamento	Número de Protótipos				r
	Mínimo	Desejado	Máximo	Obtido	Utilizado
50	118	125	132	125,4	0,005380
50	237	250	263	245,8	0,004680
50	475	500	525	496,2	0,004255
100	237	250	263	247,8	0,004500
100	475	500	525	489,2	0,004020
200	475	500	525	502,6	0,003750
200	950	1000	1050	1000,4	0,003455
400	475	500	525	494,6	0,003550
400	950	1000	1050	994,0	0,003240
400	1900	2000	2100	2000,4	0,003010

Afinal, esse era um dos objetivos dessa proposta.

Comparando, agora, as Figuras 6.7 e 6.10, percebe-se claramente a evolução da qualidade do resultado do algoritmo ARIA. No entanto, o ARIA ainda apresenta um desempenho levemente inferior ao dos algoritmos *k*-médias e NG, por motivos que ficarão evidentes nas próximas seções.

6.5 Segunda Proposta - Modificação no Cálculo da Densidade

Dado que, mesmo após a modificação na fórmula de cálculo do raio dos anticorpos, o algoritmo ARIA continuava perdendo para os algoritmos NG e *k*-means, procurou-se identificar o que mais poderia estar provocando esse desempenho inferior. O objeto de estudo passou a ser, então, o método empregado para a estimativa de densidade local.

No algoritmo ARIA, a densidade local de antígenos na vizinhança do anticorpo é estimada contando-se o número de antígenos nesta vizinhança. A vizinhança é igual para todos os anticorpos, definida pela constante E, o que fornece uma medida relativa de densidade para cada anticorpo da população. Este método de estimativa de densidade é conhecido como método do histograma (Silverman, 1986).

Entretanto, o método do histograma pode não ser eficiente para dados de dimensão elevada. Para ilustrar esse problema, imagine o seguinte cenário, similiar ao apresentado na Seção 3.4: considere um conjunto de dados no espaço d-dimensional, obtido pela amostragem de duas distribuições gaussianas. O centro de uma das gaussianas é $\mu_1 = (0_1, 0_2 \cdots 0_d)$ e o centro da outra é $\mu_2 = (10_1, 10_2 \cdots 10_d)$. As duas têm matrizes de covariância diagonal, com a mesma variância em todas as dimensões, mas uma delas tem desvio-padrão igual ao dobro da outra, ou seja, $\sigma_1 = 1$ and $\sigma_2 = 0.5$. O mesmo número de pontos $N = 1000$ é amostrado de cada distribuição, formando

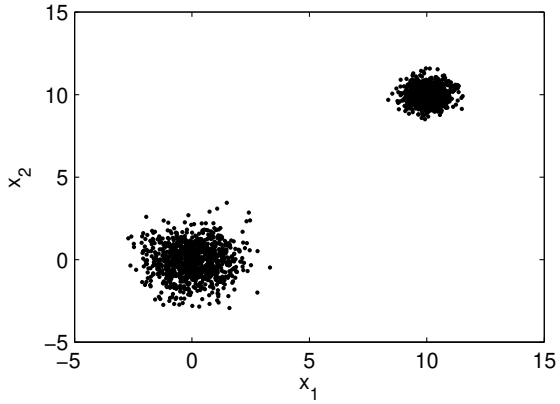


Fig. 6.11: **Dois grupos de dados bem distintos, obtidos a partir de duas distribuições gaussianas, amostradas 1000 vezes cada uma.**

dois grupos bem distintos, grupos 1 e 2, respectivamente, como mostra a Figura 6.11.

Suponha que se deseja posicionar apenas um protótipo (anticorpo) para representar cada grupo, exatamente em seu centro. De acordo com o algoritmo ARIA, seus raios devem ser proporcionais à densidade local de antígenos. A densidade ρ estimada na vizinhança de certo ponto x é dada por:

$$\rho(\mathbf{x}) = \frac{k}{V(r)} \tag{6.8}$$

onde k é o número de pontos dentro da hiperesfera de raio r centrada em $\mathbf{x}$ e $V(r)$ é o volume dessa hiperespera. O volume de uma hiperesfera de raio r no espaço d-dimensional é proporcional à d-ésima potência de r (Stibor et al., 2006):

$$V(r) = c\, r^d \tag{6.9}$$

No cálculo dos raios, interessa a relação de densidades. Para estimar a relação de densidade $\frac{\rho_1(\mu_1)}{\rho_2(\mu_2)}$ usando a Equação 6.8, pode-se fixar ou o número de pontos k (método KNN - *k-nearest neighbours*) ou o volume $V(r)$ (método do histograma) (Silverman, 1986). Para utilizar o método KNN, posiciona-se então uma hiperespera de raio r_1 no centro do grupo 1 e uma hiperesfera de raio r_2 no centro do grupo 2. Caso se escolham r_1 e r_2 proporcionais aos desvios-padrão das gaussianas $\left(\frac{r_1}{r_2} \equiv \frac{\sigma_1}{\sigma_2} = 2\right)$, haverá o mesmo número esperado de pontos $k_1 = k_2 = k$ dentro de cada hiperesfera. Usando as Equações 6.8 e 6.9, obtém-se:

$$\frac{\rho_1(\mu_1)}{\rho_2(\mu_2)} = \frac{k}{V(r_1)}\frac{V(r_2)}{k} = \frac{k}{cr_1^d}\frac{cr_2^d}{k} = \left(\frac{r_2}{r_1}\right)^d = 2^{-d} \tag{6.10}$$

No entanto, o algoritmo ARIA estima a densidade fixando r, ou $V(r)$ (método do histograma), e

Tab. 6.7: **Número médio de protótipos obtido pelo algoritmo ARIA, utilizando o método KNN com $k = 100$ para estimação de densidade e a fórmula original para cálculo do raio dos anticorpos, e raio mínimo r utilizado.**

Número de Frases de Treinamento	Número de Protótipos				r
	Mínimo	Desejado	Máximo	Obtido	Utilizado
50	118	125	132	125,4	0,005200
50	237	250	263	247,2	0,004440
50	475	500	525	493,6	0,004230
100	237	250	263	248,6	0,002375
100	475	500	525	505,6	0,002140
200	475	500	525	495,4	0,001485
200	950	1000	1050	995	0,001380
400	475	500	525	499,6	0,001281
400	950	1000	1050	1010	0,000960
400	1900	2000	2100	2011	0,000898

não k, obtendo:

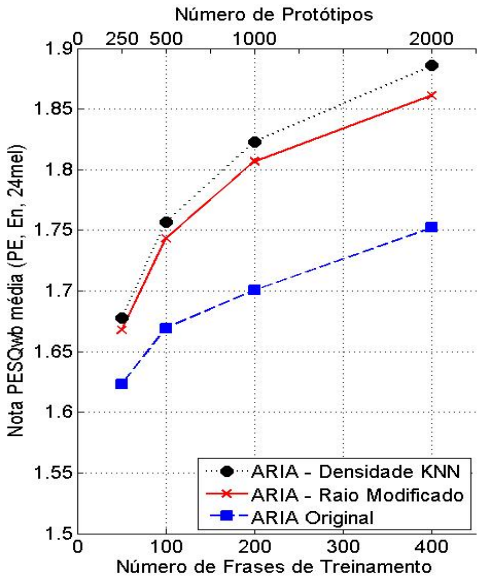
$$\frac{\rho_1(\mu_1)}{\rho_2(\mu_2)} = \frac{k_1}{V(r)} \frac{V(r)}{k_2} = \frac{k_1}{k_2} \tag{6.11}$$

Para que a estimativa do ARIA seja correta, basta que $\frac{k_1}{k_2} \approx 2^{-d}$. Isso parece bastante simples, mas se a dimensão d for grande, essa estimativa se torna praticamente impossível de funcionar. Por exemplo, no caso dos dados utilizados nesse trabalho, que têm dimensão $d = 26$, seriam necessários no mínimo 2^{26} pontos (mais de 60 milhões).

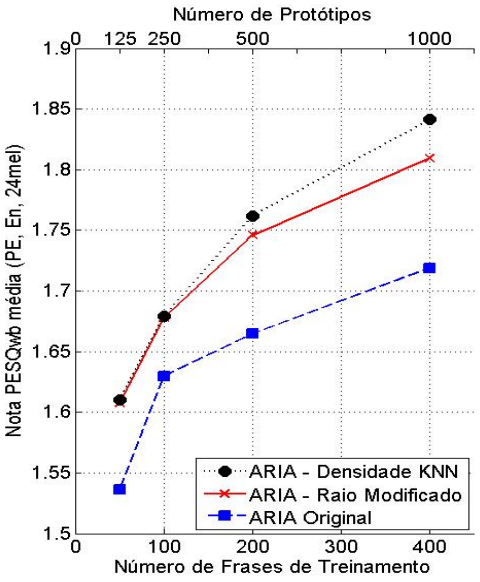
Portanto, o verdadeiro problema no algoritmo não está na fórmula de cálculo do raio, mas sim no método de estimação de densidade. A solução apresentada na seção anterior apenas mascarava o real problema, servindo como um paliativo. Assim, ao invés de fixar o volume da hiperesfera através do raio de vizinhança E (método do histograma), passou-se a utilizar o método KNN, fixando k, e retomou-se a fórmula original de cálculo do raio dos anticorpos (Violato et al., 2010).

A Figura 6.12 compara as três versões do algoritmo ARIA, sendo que para o método KNN a densidade foi estimada com $k = 100$. Os valores de raio mínimo empregados em cada caso, bem como o número médio de protótipos obtido, estão mostrados na Tabela 6.7.

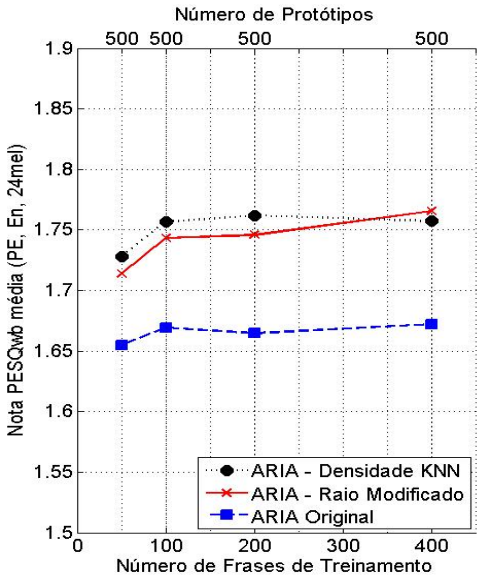
Nitidamente, a solução apresentada nesta seção obteve os melhores resultados, empatando em dois casos e perdendo em apenas um dos dez testes feitos. Na comparação com os outros algoritmos, infelizmente o algoritmo ARIA ainda perde tanto para o algoritmo *k*-médias quanto para o algoritmo NG (na realidade, em apenas um teste o resultado do algoritmo ARIA supera o do *k*-médias). Não é mostrada uma figura com esta comparação devido à semelhança que esta teria com a Figura 6.10, somente com as curvas do algoritmo ARIA mais próximas das curvas do *k*-médias.



(a) Taxa de compressão de 100 vezes.



(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.12: **Resultado das três versões do algoritmo ARIA, utilizando o conjunto de parâmetros PE + En + 24 mel.**

Na seção 6.6, é explicado por que o algoritmo ARIA, que agora é seguramente capaz de preservar a densidade, continua com desempenho pior que o dos outros algoritmos.

6.6 Relação entre a Nota PESQ, o Erro de Quantização e a Entropia Relativa

Até o momento, só se avaliou a qualidade dos algoritmos de quantização vetorial, através da nota PESQwb média das 50 frases de teste, sintetizadas a partir dos dicionários de quadros formados por quadros selecionados da base de treinamento pelos algoritmos em questão.

Nessa seção, verifica-se se há correlação entre essa medida de qualidade do sinal de fala e as medidas de qualidade da quantização dos dados, isto é, o erro de quantização e a entropia relativa das distribuições de dados e protótipos, conforme descrito na Seção 3.4.

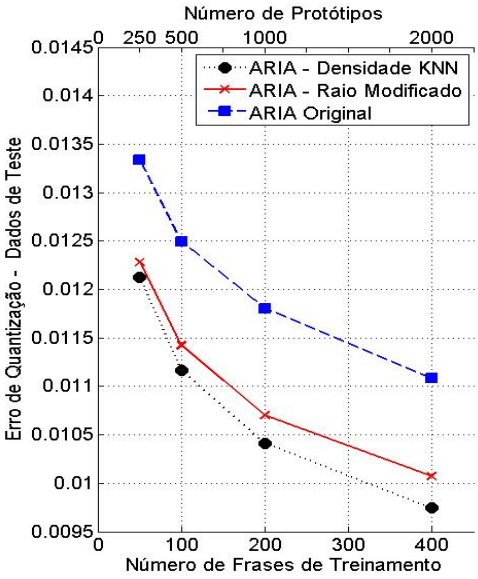
Começando pelo erro de quantização, pode-se calculá-lo tanto em relação aos dados de treinamento quanto em relação aos dados de teste. Como as notas PESQwb são referentes aos dados de teste, calcula-se o erro de quantização em relação aos dados de teste também. Na Figura 6.13, são apresentados resultados comparando o algoritmo ARIA em sua versão original com as versões propostas.

Assim como para a nota PESQwb, a modificações no algoritmo ARIA propostas neste trabalho levaram a erros de quantização menores. E, também como para a nota PESQwb, a modificação no método de estimação de densidade foi a que produziu as melhores soluções. Na Figura 6.14, compara-se a versão do algoritmo ARIA que produziu os melhores resultados com os outros algoritmos, isto é, NG e *k*-médias, com a escolha aleatória de protótipos e com o teste "Sem Compressão".

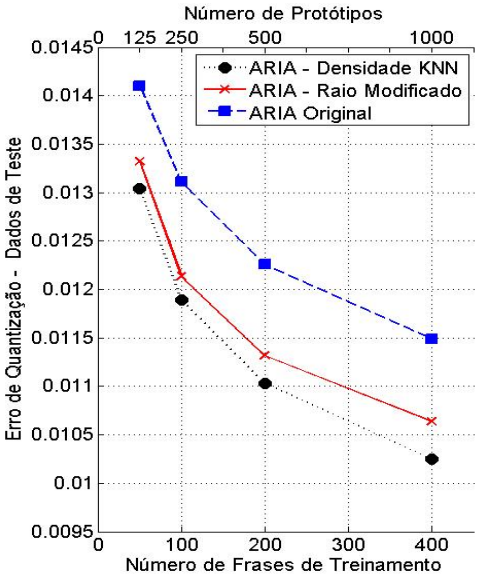
Apesar da escala prejudicada, é possível perceber que o algoritmo NG obteve os menores erros de quantização, seguido pelo *k*-médias e depois pelo ARIA. Esse comportamento é equivalente ao observado para os resultados de nota PESQwb desses algoritmos. Com isso, já se pode afirmar que existe correlação entre essas medidas.

Para conhecer a natureza dessa correlação, na Figura 6.15 é mostrado um gráfico da nota PESQwb vs. o erro de quantização, incluindo o resultado de todos os algoritmos. Os 10 pontos de cada algoritmo são referentes às 10 configurações de testes diferentes adotadas, exceto, é claro, para os pontos "Sem Compressão", que são apenas 4, um para cada tamanho de base de treinamento adotada.

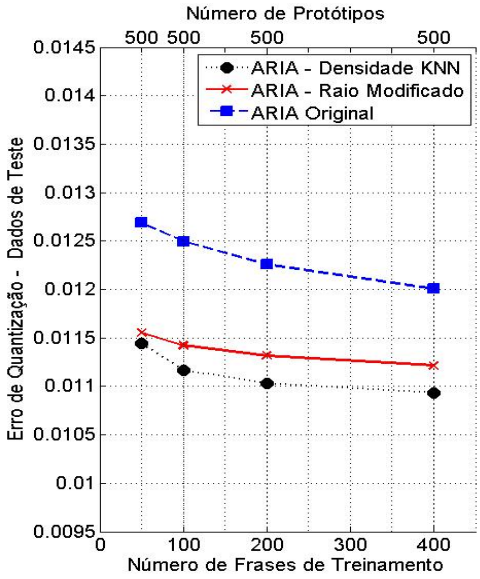
O gráfico da Figura 6.15 deixa claro que há correlação linear entre a nota PESQwb e o erro de quantização. Além disso, os gráficos das Figuras 6.13 e 6.14 mostram que o erro do ARIA diminuiu, após as alterações na fórmula de cálculo do raio e no método de estimação de densidade, mas continua com resultados piores do que os outros algoritmos. É importante destacar que o objetivo do ARIA é produzir uma distribuição de protótipos que respeite a densidade dos dados, e não minimizar o



(a) Taxa de compressão de 100 vezes.

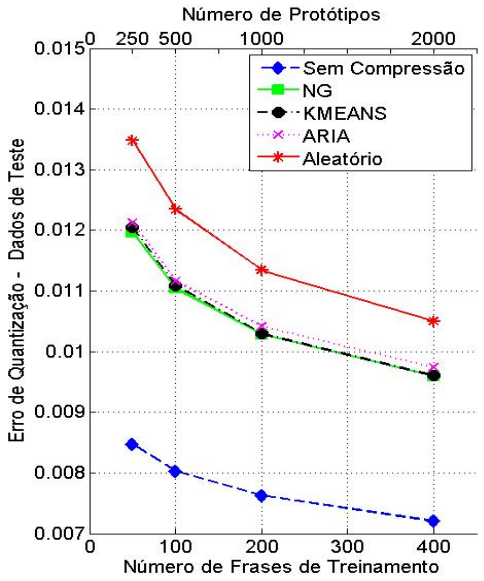


(b) Taxa de compressão de 200 vezes.

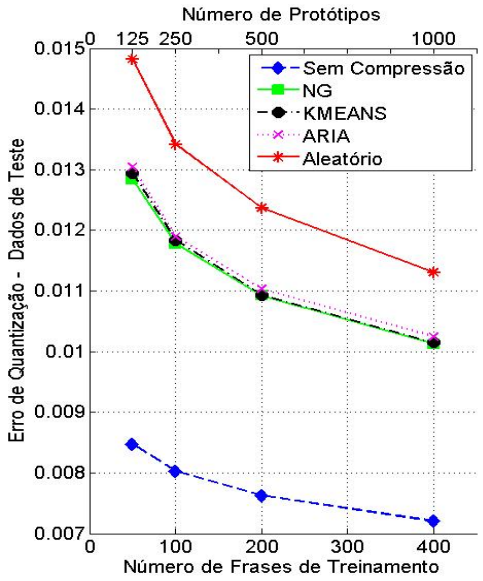


(c) Número de protótipos fixo (500).

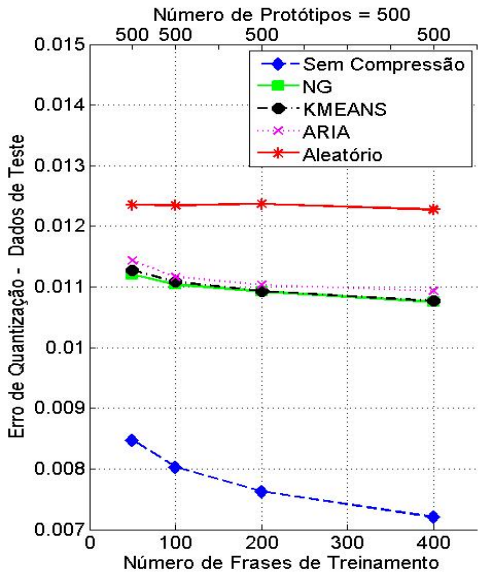
Fig. 6.13: **Erro de quantização médio das três versões do algoritmo ARIA.**



(a) Taxa de compressão de 100 vezes.



(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.14: **Erro de quantização médio (em relação aos dados de teste) dos algoritmos *k*-médias, NG e ARIA (utilizando o método KNN para estimação de densidade), da escolha aleatória de protótipos e do *codebook* formado pelas bases de treinamento inteiras ("Sem Compressão").**

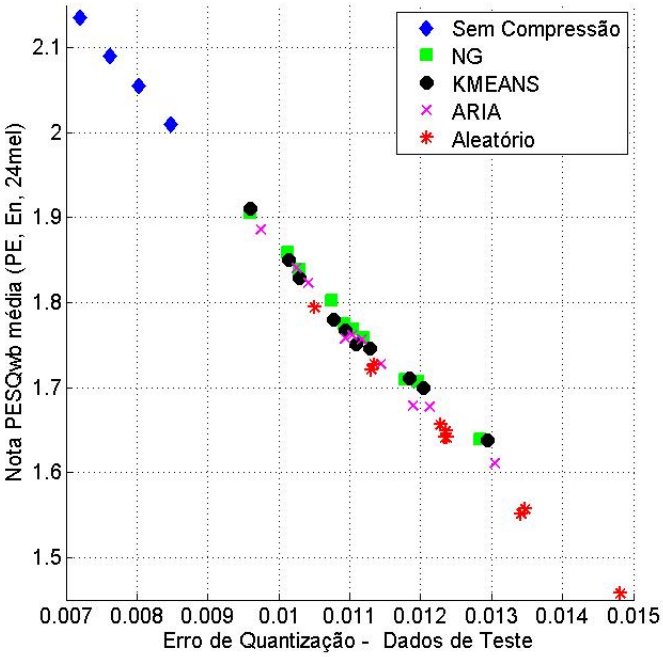


Fig. 6.15: **Relação entre a nota PESQwb média das frases de teste e o erro de quantização dos dados de teste. Nesse gráfico, aparecem os resultados de todos os algoritmos aplicados a todas as configurações de teste utilizadas.**

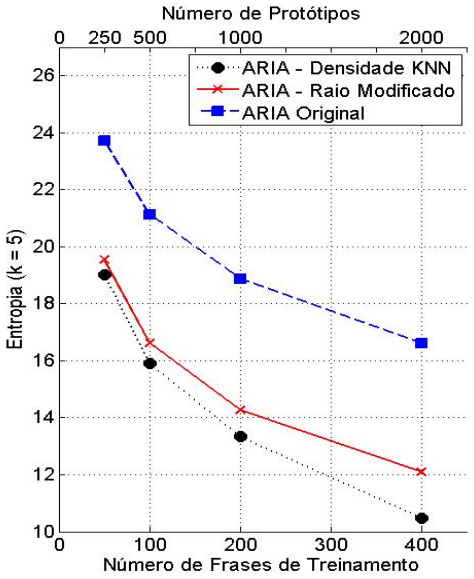
erro de quantização. Por isso, investigou-se se o ARIA supera os outros algoritmos na questão da distribuição dos protótipos. A entropia relativa, que indica a proximidade entre as distribuições de dados e protótipos, dará uma resposta a essa pergunta.

Caso a resposta seja afirmativa, isso implica que, para esta aplicação, não é interessante selecionar protótipos que respeitem a densidade, mas sim selecioná-los de forma que o erro de quantização seja o menor possível.

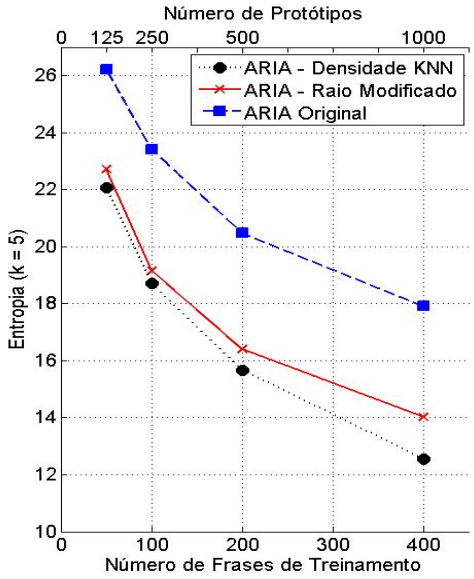
Para esse teste, decidiu-se usar o método KNN por dois motivos. Primeiro porque o parâmetro k (tamanho da vizinhança) é mais intuitivo e, por isso, mais fácil de regular, do que a largura de banda h do método do estimador de núcleo. Segundo porque, em testes preliminares, o estimador de núcleo forneceu resultados piores.

Para essa avaliação, empregaram-se diferentes valores de k (do método KNN) para se estimar a entropia relativa das distribuições: $k = 5$, $k = 20$, $k = 50$ e $k = 100$. Primeiro compara-se apenas o resultado do ARIA original com o resultado das modificações propostas neste trabalho. Tal resultado é apresentado na Figura 6.16. Apenas o caso $k = 5$ é mostrado, pois os resultados para os outros valores de k foram equivalentes.

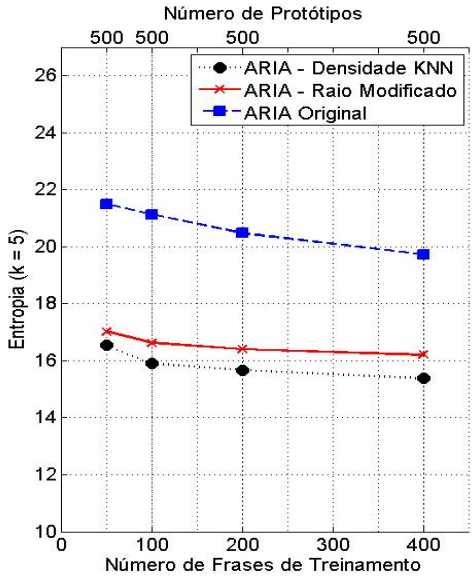
Vê-se que, também segundo essa medida, as modificações propostas aperfeiçoaram o resultado e



(a) Taxa de compressão de 100 vezes.



(b) Taxa de compressão de 200 vezes.



(c) Número de protótipos fixo (500).

Fig. 6.16: **Erro de quantização médio das três versões do algoritmo ARIA.**

que a modificação no método de estimação de densidade foi a que surtiu mais efeito, uma vez que, quanto mais próximo de zero o valor da entropia, mais similares são as distribuições comparadas.

Na Figura 6.17, apresentam-se os resultados dos algoritmos NG, *k*-médias, ARIA (apenas da versão com o método KNN para estimação de densidade) e da escolha aleatória, novamente apenas para $k = 5$.

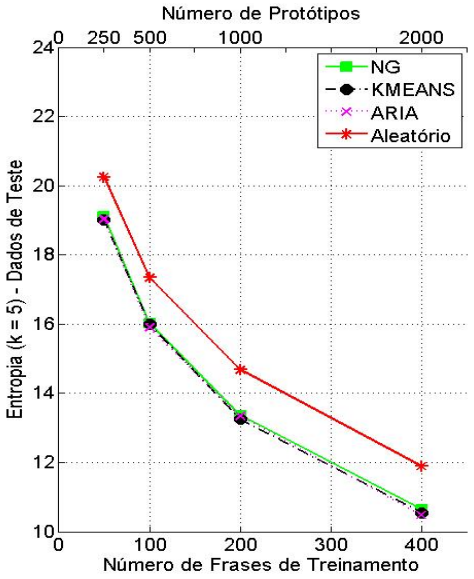
Mais uma vez, pode-se ver que há correlação da entropia relativa com a nota PESQwb, pois as curvas apresentam comportamento semelhante ao observado para o erro de quantização. A principal diferença é que nesse caso, o ARIA apresentou os melhores resultados, seguido pelo *k*-médias e depois pelo NG.

Para melhor visualização e compreensão da natureza da correlação entre as medidas, tal qual foi feito para o erro de quantização, na Figura 6.18 são mostrados gráficos de nota PESQ vs. Entropia relativa, agora para os diferentes valores de k avaliados.

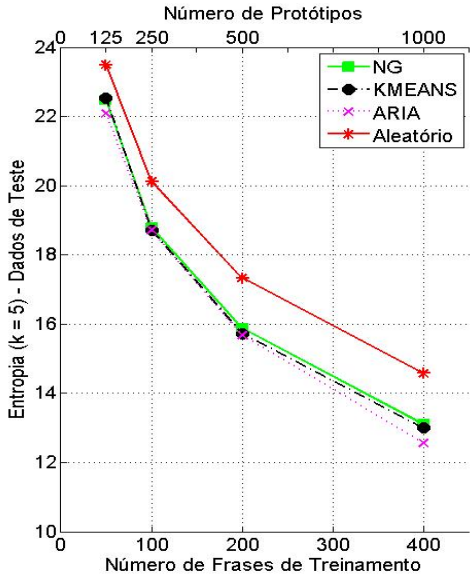
Da Figura 6.18, pode-se concluir que (*i*) a escolha aleatória produz o pior resultado, (*ii*) *k*-médias é ligeiramente melhor que o NG, pois seus pontos estão um pouco mais à esquerda (menor entropia relativa), diferente da nota PESQwb, em que o NG é que é ligeiramente superior ao *k*-médias, pois seus pontos estão um pouco mais acima (ver também Figuras 6.7 e 6.10), e (*iii*) o ARIA, apesar de na maioria das situações produzir distribuições melhores ou tão boas quanto *k*-médias (entropia relativa menor ou igual), levou a notas PESQwb sempre piores.

Observando as curvas da nota PESQwb vs. Entropia relativa (Figura 6.18), percebe-se que as medidas são correlacionadas, pelo menos quando se considera cada algoritmo individualmente. Ou seja, uma distribuição de protótipos gerada por certo algoritmo que produz uma entropia relativa menor em relação aos dados de teste, sempre leva a uma nota PESQwb maior, para todos os k's testados. No entanto, quando se consideram algoritmos diferentes, isso nem sempre é verdade, pois repare que há pontos mais acima (nota PESQwb maior) e mais à direta (entropia maior) do que outros. Por isso, pode-se dizer que a correlação entre a nota PESQwb e o erro de quantização é "mais forte" do que sua correlação com a entropia relativa entre as distribuições, uma vez que, para o erro de quantização, os pontos de todos os algoritmos são colineares.

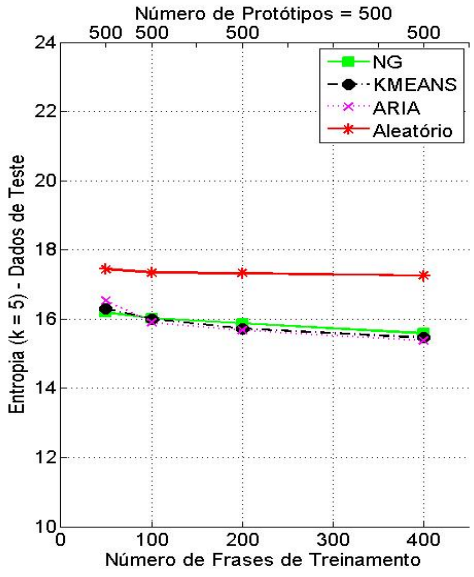
Com essa análise, também se justifica por que o ARIA, mesmo após as modificações que melhoraram seu desempenho, ainda perde para os outros algoritmos na aplicação deste trabalho: o algoritmo posiciona protótipos que, apesar de levar à menor entropia relativa, não geram o menor erro de quantização, refletindo em uma nota PESQwb pior.



(a) Taxa de compressão de 100 vezes.

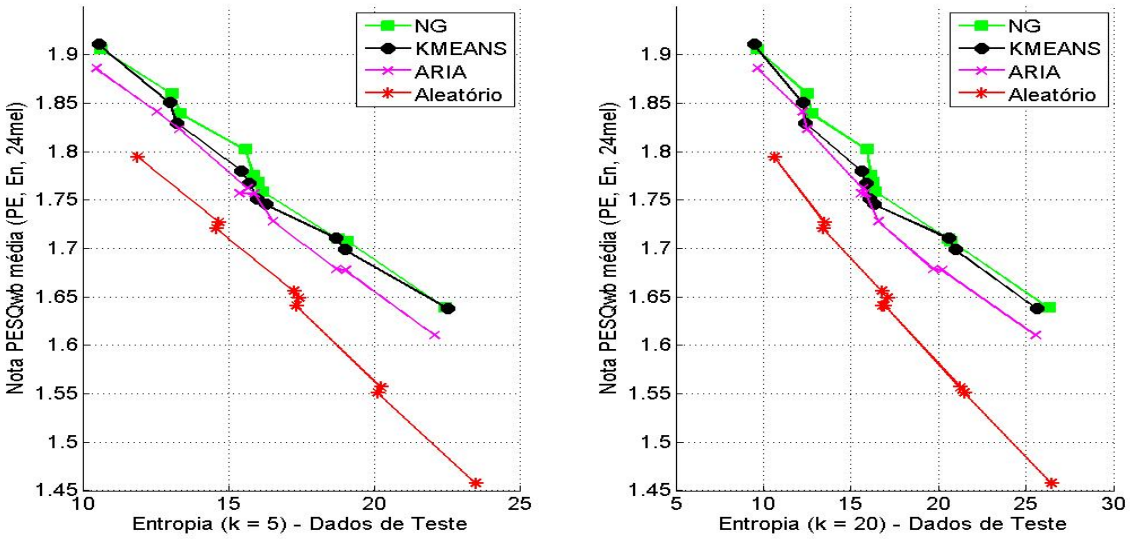


(b) Taxa de compressão de 200 vezes.



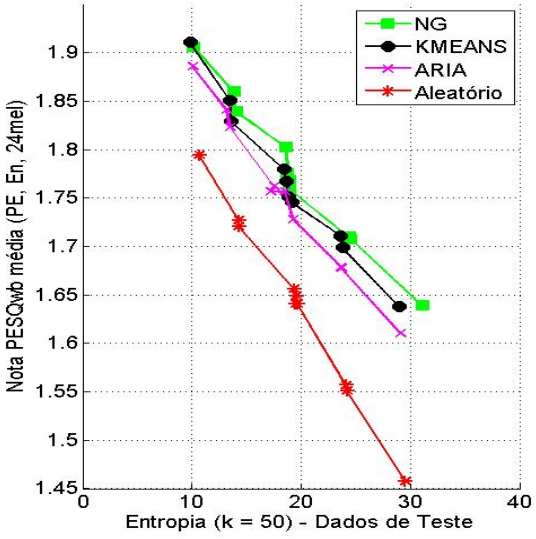
(c) Número de protótipos fixo (500).

Fig. 6.17: **Erro de quantização médio das três versões do algoritmo ARIA.**

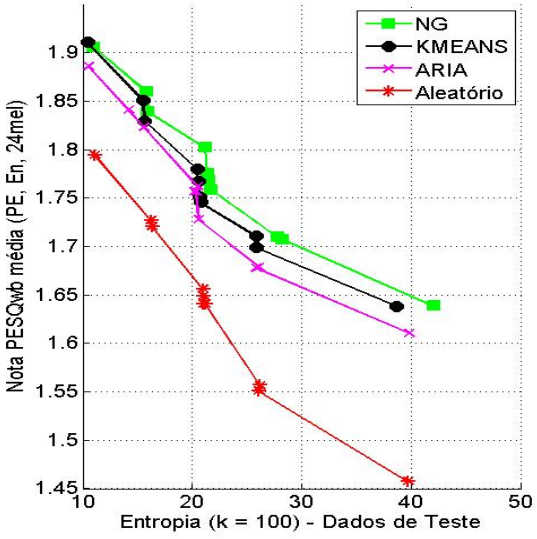


(a)

(b)



(c)



(d)

Fig. 6.18: **Resultado da relação da nota PESQwb e da entropia relativa entre as distribuições dos dados de teste e dos protótipos, produzidos pelo algoritmos NG, *k*-médias, ARIA (versão com o método KNN para estimação de densidade) e escolha aleatória de protótipos, estimada com o método KNN, para diferentes valores de k. 6.18(a) $k = 5$. 6.18(b) $k = 20$. 6.18(c) $k = 50$. 6.18(d) $k = 100$.**

Capítulo 7

Conclusão

Este trabalho realizou um estudo de algoritmos de quantização vetorial aplicados na compressão de sinais de fala. Na técnica de compressão utilizada, os sinais são primeiramente divididos em quadros, depois são janelados e, em seguida, são parametrizados e quantizados, para serem armazenados e/ou transmitidos. Para recompor o sinal, os vetores quantizados são mapeados em quadros de fala, que são, por sua vez, concatenados através de uma técnica de síntese concatenativa, conhecida como PSOLA.

Um dos estudos feitos neste trabalho foi a avaliação de diferentes conjuntos de atributos do sinal de fala, usados na etapa de parametrização. Os parâmetros testados foram os coeficientes LPC, os coeficientes LSF, os coeficientes MFCC, os coeficientes mel, a energia e a frequência fundamental. Concluiu-se que os melhores resultados eram obtidos quando se empregavam os coeficientes mel, associados à energia e ao período esquerdo (que carrega a informação de frequência fundamental). A magnitude desses dois últimos atributos foi alterada, em um processo de normalização e ponderação, no qual os pesos utilizados foram otimizados.

Uma vez definido o conjunto de atributos do sinal que conduziram aos melhores resultados, foi feito outro estudo, dessa vez envolvendo diferentes algoritmos de quantização vetorial. Foram avaliados os algoritmos *k*-médias, NG (*Neural-Gas*) e ARIA (*Adaptive Radius Immune Algorithm*). Em uma avaliação considerando unicamente a qualidade do sinal de fala sintetizado como medida de desempenho, os algoritmos *k*-médias e NG tiveram resultados equivalentes, enquanto o ARIA apresentou os piores resultados.

A causa desses resultados inferiores foi descoberta: na contramão da esperada preservação de densidade, que o algoritmo propunha e mostrava ser capaz de obter para dados de baixa dimensão, o posicionamento de protótipos realizado pelo ARIA revelou-se distorcido, o que pode ser associado à elevada dimensão dos dados. Este comportamento inesperado foi detectado pela primeira vez neste trabalho. Determinado o motivo, foram propostas duas modificações simples no algoritmo, uma delas

alterando a forma com que os raios dos anticorpos eram calculados e a outra mudando o método de estimação de densidade local de dados na vizinhança de cada anticorpo, sendo esta última a mais eficiente.

Tais modificações foram implementadas e levaram o ARIA a melhorar consideravelmente seu desempenho, mas ainda perdia para os outros dois algoritmos em questão. Com isso, também foi investigada neste trabalho a relação entre a qualidade do sinal produzido, dada pela nota PESQwb, e duas medidas de avaliação da qualidade da quantização: o erro de quantização e a entropia relativa, utilizada para avaliar a similaridade entre as distribuições de dados e de protótipos. Cabe mencionar que a entropia relativa está diretamente vinculada à preservação de densidade.

O erro de quantização mostrou-se fortemente correlacionado à nota PESQwb, enquanto a entropia relativa também apresentou certa correlação, mas não tão destacada quanto a do erro de quantização. Isso explica o porquê do desempenho inferior do ARIA para esta aplicação. O objetivo do ARIA é a preservação da densidade na distribuição dos protótipos, o que resulta em entropias relativas menores, não correspondendo necessariamente à minimização do erro de quantização. Por isso, o ARIA também perdeu para os outros algoritmos no quesito erro de quantização, apesar de superá-los no quesito entropia relativa. Entretanto, deve-se ressaltar que as modificações propostas foram capazes de aprimorar a resposta do algoritmo em relação a ambas as medidas de qualidade da quantização.

Dados esses resultados, conclui-se que não é recomendável a utilização do ARIA para esta aplicação. Além disso, dos três algoritmos testados, o ARIA é o mais custoso computacionalmente. O custo computacional é um fator importante na análise de algoritmos, mas não foi abordado neste trabalho, porque a quantização vetorial é uma etapa *offline* na aplicação. Assim, seria justificável o uso de um algoritmo computacionalmente mais caro, caso ele levasse a resultados melhores.

Por fim, do ponto de vista da aplicação, os valores absolutos de nota PESQwb conseguidos pela técnica descrita são relativamente ruins, tornando inviável sua utilização em ferramentas comerciais, exceto em situações que não exijam muita qualidade, ou que os recursos de memória disponível sejam realmente pequenos.

Enfrentar essa limitação é a principal sugestão para a continuidade deste trabalho. Repare que não foi empregada nenhuma técnica de processamento de sinais para reduzir as distorções introduzidas pela concatenação de quadros, os quais muitas vezes não casam. As amostras dos quadros são simplesmente sobrepostas e somadas.

Além disso, a qualidade do sinal sintetizado poderia ser melhorada através de uma escolha da sequência de quadros do dicionário que leve em consideração algum tipo de custo de concatenação. Lembre-se que, neste trabalho, há apenas o que se pode chamar de custo de substituição. Em outras palavras, na hora da escolha de um quadro, é avaliada apenas a distância entre o vetor do quadro original e os *codevectors*, mas não a distância entre vetores consecutivos.

Seria interessante também avaliar diferentes locutores (masculinos e femininos), para verificar o desempenho do sistema com vozes mais graves ou agudas, e trabalhar com sinais amostrados a 8 kHz, o que reduz imediatamente o tamanho do dicionário de quadros pela metade e atenua algumas degradações de alta frequência.

Com isso, espera-se obter uma melhora da qualidade do sinal produzido, independentemente da eficiência do algoritmo de quantização vetorial utilizado.

Outro ponto em que há margem para trabalhos futuros é o estudo de outros parâmetros do sinal de fala, ou de diferentes combinações dos parâmetros já apresentados aqui, buscando um vetor de atributos mais representativo e altamente correlacionado com a qualidade do sinal produzido.

No que diz respeito aos algoritmos de quantização, pode-se procurar algoritmos mais competentes na minimização do erro de quantização ou aperfeiçoar os algoritmos descritos neste trabalho, de forma que a qualidade do sinal gerado a partir do *codebook* se aproxime cada vez mais da qualidade obtida com o emprego desta técnica, mas sem compressão.

Quanto ao ARIA, já foi proposta outra melhoria visando torná-lo eficiente para a minimização do erro de quantização (Azzolini et al., 2010), envolvendo o método empregado no cálculo do seu raio. O efeito dessa melhoria no desempenho do ARIA junto às aplicações consideradas nesta pesquisa ainda precisa ser avaliado.

Outras etapas do algoritmo ARIA podem ser modificadas. O modelo de treinamento sequencial pode ser substituído por um treinamento em batelada. O mecanismo de clonagem pode ser mais eficiente, evitando a proliferação exagerada de anticorpos. Assim como o mecanismo de supressão, que se baseia em uma rede de anticorpos totalmente conectada, o que é bastante custoso, poderia ser revisto para, por exemplo, empregar outro tipo de rede.

Conclui-se, por fim, que ainda há espaço para muitas melhorias no sistema descrito neste trabalho tanto no que diz respeito ao processamento de sinais quanto no que diz respeito à quantização vetorial, o que indica boas perspectivas futuras para a linha de pesquisa.

Referências Bibliográficas

J.-P. Adoul, P. Mabilleau, M. Delprat, e S. Morisette. Fast CELP Coding Based on Algebraic Codes. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 12, pp. 1957–1960, April 1987.

B. S. Atal. The History of Linear Prediction. In *IEEE Signal Processing Magazine*, vol. 23, pp. 154–161, March 2006.

A. G. Azzolini, R. P. V. Violato, e F. J. Von Zuben. Density Preservation and Vector Quantization in Immune-Inspired Algorithms. In *Proceedings of the 9th International Conference on Artificial Immune Systems (ICARIS'2010), Lecture Notes in Computer Science*, vol. 6209, pp. 33–46, July 2010.

J. G. Beerends e J. A. Stemerdink. A Perceptual Audio Quality Measure Based on a Psychoacoustic Sound Representation. *Journal of the Audio Engineering Society*, vol. 40, no. 12, pp. 963–974, December 1992.

J. G. Beerends e J. A. Stemerdink. A Perceptual Speech-Quality Measure Based on a Psychoacoustic Sound Representation. *Journal of the Audio Engineering Society*, vol. 42, no. 3, pp. 115–123, March 1994.

J. Benesty, M. M. Sondhi, e Y. Huang, editors. *Springer Handbook of Speech Processing*. Springer, 2008.

G. B. Bezerra, T. V. Barra, L. N. de Castro, e F. J. Von Zuben. Adaptive Radius Immune Algorithm for Data Clustering. In C. Jacob, M. L. Pilat, P. J. Bentley, e J. Timmis, editors, *Proceedings of 4th International Conference on Artificial Immune Systems (ICARIS-2005)*, vol. 3627 of *Lecture Notes in Computer Science*, pp. 290–303. Springer-Verlag, August 2005.

S. Boll. Suppression of Acoustic Noise in Speech Using Spectral Subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 27, no. 2, pp. 113–120, April 1979.

B. P. Borget, M. J. R. Healy, e J. W. Tukey. The Quefrency Alanysis of Times Series for Echos: Cepstrum, Pseudo-Autocovariance, Cross-Cepstrum, and Saphe Cracking. In M. Rosenblatt, editor, *Proceedings of the Symposium on Time Series Analysis*, pp. 209–243. Wiley, 1963.

F. M. Burnet. *The Clonal Selection Theory of Acquired Immunity*. Vanderbilt University Press, Nashville, TN, 1959.

F. M. Burnet. Clonal Selection and After. In G. I. Bell, A. S. Perelson, e G. H. Pimgley Jr, editors, *Theoretical Immunology*, pp. 63–85. Marcel Dekker Inc., 1978.

J.-H. Chen. A Robust Low-Delay CELP Speech Coder at 16kb/s. In *Proceedings of IEEE Global Telcommunications Conference*, vol. 2, pp. 1237–1241, 1989.

J.-H. Chen, R. V. Cox, Y.-C. Lin, N. S. Jayant, e M. J. Melchner. A Low-Delay CELP Coder for the CCITT 16kb/s Speech Coding Standard. *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 5, pp. 830–849, June 1992.

V. Cherkassky e F. Mulier. *Learning From Data: Concepts, Theory, and Methods*. Wiley-Interscience, 1998.

I. R. Cohen. *Tending Adam's Garden: Evolving the Cognitive Immune Self*. Academic Press, 2004.

D. Dasgupta. *Artificial Immune Systems and their Applications*. Springer-Verlag, 1998.

S. Davis e P. Mermelstein. Comparision of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, August 1980.

L. N. de Castro. *Engenharia Imunológica: Desenvolvimento e Aplicação de Ferramentas Computacionais Inspiradas em Sistemas Imunológicos Artificiais*. Tese de Doutorado, UNICAMP, Maio 2001.

L. N. de Castro. *Fundamentals of Natural Computing: Basic Concepts, Algorithms and Applications*. Chapman & Hall/CRC, 2006.

L. N. de Castro e J. Timmis. *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer-Verlag, 2002.

L. N. de Castro e F. J. Von Zuben. aiNet: An Artificial Immune Network for Data Analysis. In H. A. Abbass, R. A. Sarker, e C. S. Newton, editors, *Data Mining: A Heuristic Approach*, chapter 12, pp. 231–259. Idea Group Publishing, 2001.

R. O. Duda, P. E. Hart, e D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2nd edition, 2001.

J. Durbin. Efficient Estimation of Parameters on Moving-Average Models. *Biometrika*, vol. 46, no. 3-4, pp. 306–316, 1959.

J. Durbin. The Fitting of Time-Series Models. *Revue de l'Institut International de Statistique*, vol. 28, no. 3, pp. 233–243, 1960.

K. Fukunaga e R. R. Hayes. The Reduced Parzen Classifier. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 11, no. 4, pp. 423–425, April 1989.

Y. Gao, A. Benyassine, J. Thyssen, H. Su, e E. Shlomot. eX-CELP: A Speech Coding Paradigm. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 689–692, 2001.

I. A. Gerson e M. A. Jasiuk. Vector Sum Excited Linear Prediction (VSELP) Speech Coding at 8 kbps. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 461–464, 1990.

R. M. Gray e D. L. Neuhoff. Quantization. *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2325–2383, October 1998.

J. Han e M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2nd edition, 2006.

A. H. Hentz e R. Seara. Compressão de Bancos de Fala para Sistemas de Síntese Concatenativa de Alta Qualidade. In *XXVII Simpósio Brasileiro de Telecomunicações (SBrT)*, 2009.

F. Itakura. Line Spectrum Representation of Linear Predictive Coefficients of Speech Signals. *Journal of the Acoustical Society of America*, vol. 57, no. 1, pp. 35, 1975.

ITU. Website acessado em agosto. http://www.itu.int, 2008.

C. A. Janeway, P. Travers, M. Walport, e M. Shlomchik. *Imunobiologia: O Sistema Imune na Saúde e na Doença*. Artmed, 2001.

N. K. Jerne. Towards a Network Theory of the Immune System. In *Ann. Immunol. Inst. Pasteur*, no 1-2 in 125C, pp. 373–389, January 1974.

T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, e A. Y. Wu. An Efficient k-means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, July 2002.

T. Kanungo, D. M. Mount, N. Netanyahu, C. Piatko, R. Silverman, e A. Y. Wu. A Local Search Approximation Algorithm for k-Means Clustering. *Computational Geometry: Theory and Applications*, vol. 28, no. 2-3, pp. 89–112, June 2004.

A. Kataoka, T. Moriya, e S. Hayashi. An 8-kbit/s Speech Coder Based on Conjugate Structure CELP. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 592–595, 1993.

D. H. Klatt. Software for a Cascade/Parallel Formant Synthesizer. *Journal of the Acoustical Society of America*, vol. 67, no. 3, pp. 971–995, March 1980.

W. B. Kleijn, P. Kron, L. Cellario, e D. Sereno. A 5.85 kb/s CELP Algorithm for Cellular Applications. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 596–599, 1993.

T. Kohonen. Self-Organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, January 1982.

S. Kullback. *Information Theory and Statistics*. John Wiley & Sons, 1959.

C. Laflamme, J.-P. Adoul, H. Y. Su, e S. Morisette. On Reducing Computational Complexity of Codebook Search in CELP Coder through the use of Algebraic Codes. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 177–180, 1990.

V. L. Latsch. Construção de Banco de Unidades para Síntese de Fala por Concatenação no Domínio Temporal. Dissertação de Mestrado, UFRJ, Abril 2005.

K.-S. Lee e R. V. Cox. A Very Low Bit Rate Speech Coder Based on a Recognition-Synthesis Paradigm. *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 5, pp. 482–491, July 2001.

N. Levinson. The Wiener RMS (Root Mean Square) Error Criterion in Filter Design and Prediction. *Journal of Mathematics and Physics of the Massachusetts Institute of Technology*, vol. 25, no. 4, pp. 261–278, 1947.

S. P. Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, March 1982.

J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. Fifth Berkeley Symp. Math. Statistics and Probability*, pp. 281–296, 1967.

J. Makhoul e J. Wolf. *Linear Prediction and the Spectral Analysis of Speech*, pp. 172–185. Bolt, Beranek, and Newman Inc., 1972.

J. D. Markel e A. H. Gray Jr. *Linear Prediction of Speech*. Springer, 1976.

T. M. Martinetz, S. G. Berkovich, e K. J. Schulten. "Neural-gas" Network for Vector Quantization and its Application to Time-Series Prediction. *IEEE Transactions on Neural Networks*, vol. 4, no. 4, pp. 558 – 569, July 1993.

T. M. Martinetz e K. J. Schulten. A "Neural-Gas" Network Learns Topologies. In T. Kohonen, K. Mäkisara, O. Simula, e J. Kangas, editors, *Artificial Neural Networks*, pp. 397–402. Elsevier, North-Holland, Amsterdam, 1991.

P. Matzinger. Tolerance, Danger and the Extended Family. *Annual Review of Immunology*, vol. 12, pp. 991–1045, April 1994.

P. Matzinger. The Danger Model: A Renewed Sense of Self. *Science*, vol. 296, no. 5566, pp. 301–305, April 2002.

S. Miki, K. Mano, H. Ohmuro, e T. Moriya. Pitch Synchronous Innovation CELP (PSI-CELP). In *Proceedings of Eurospeech Conference*, pp. 261–264, 1993.

E. Moulines. *Algorithmes de Codage et de Modification des Paramètres Prosodiques pour la Synthèse de la Parole à partir du Texte*. Tese de Doutorado, École National Supérieure des Télécommunications, February 1990.

A. V. Oppenheim e R. W. Schaffer. *Discrete-Time Signal Processing*. Prentice-Hall, 1989.

ITU-T Recommendation P.800. Methods for Subjective Determination of Transmission Quality, 1996.

ITU-T Recommendation P.862. Perceptual Evaluation of Speech Quality (PESQ): An Objective Method for End-to-End Speech Quality Assessment of Narrow Band Telephone Networks and Speech Codecs, February 2001.

ITU-T Recommendation P.862.2. Wideband Extension to Recommendation P.862 for the Assessment of Wideband Telephone Networks and Speech Codecs, November 2007.

J. Picone. Signal Modeling Techniques in Speech Recognition. *Proceedings of the IEEE*, vol. 81, no. 9, pp. 1215–1247, September 1993.

G. Pinchuk. *Theory and Problems of Immunology*. USA: McGraw-Hill, 2002.

L. R. Rabiner e R. W. Schaffer. *Digital Processing of Speech Signal*. Prentice-Hall, 1976.

A. Rix, J. Beerends, M. Hollier, e A. Hekstra. Perceptual Evaluation of Speech Quality (PESQ) - A New Method for Speech Quality Assessment of Telephone Networks and Codecs. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 73–76, 2001.

A. Rix e M. Hollier. The Perceptual Analysis Measurement System for Robust end-to-end Speech Quality Assessment. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1515–1518, June 2000.

R. Salami, C. Laflamme, J. P. Adoul, A. Kataoka, S. Hayashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, e Y. Shoham. Design and Description of CS-ACELP: a Toll Quality 8 kb/s Speech Coder. *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 2, pp. 116–130, March 1998.

M. R. Schoroeder e B. S. Atal. Code-Excited Linear Prediction (CELP): High Quality Speech at Very Low Bit Rates. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 937–940, March 1985.

L. A. Segel e I. Cohen, editors. *Design Principle for the Immune System and Other Distributed Autonomous Systems*. Oxford University Press, 2001.

B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Number 26 in Monographs on statistics and applied probability. Chapman & Hall, 1986.

F. O. Simões. Implementação de um Sistema de Conversão Texto-Fala para o Português do Brasil. Dissertação de Mestrado, UNICAMP, Maio 1999.

F. O. Simões, M. Uliani Neto, J. B. Machado, E. J. Nagle, F. O. Runstein, e L. C. T. Gomes. Speech Compression Using Vector Quantization and Unsupervised Neural Networks. In *Brain Inspired Cognitive Systems (BICS 2008) - Fourth International ICSC Symposium on Biologically Inspired Systems (BIS 2008)*, 2008.

F. K. Song e B.-H. Juang. Line Spectrum Pair (LSP) and Speech Data Compression. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 9, pp. 37–40, March 1984.

F. K. Song e B.-H. Juang. Optimal Quantization of LSP parameters. *IEEE Transactions on Speech and Audio Processing*, vol. 6, pp. 15–24, 1993.

Starlab. Website acessado em agosto. users.pandora.be/richard.wheeler1/ais/inn.html, 2008.

S. Stevens, J. Volkman, e E. Newman. A Scale for the Measurement of the Psychological Magnitude of Pitch. *Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, January 1937.

T. Stibor e J. Timmis. An Investigation on the Compression Quality of aiNet. In *Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence (FOCI 2007)*, pp. 495–502, 2007.

T. Stibor, J. Timmis, e C. Eckert. On the use of hyperspheres in artificial immune systems as antibody recognition regions. In H. Bersini e J. Carneiro, editors, *Proceedings of 5th International Conference on Artificial Immune Systems (ICARIS-2006)*, vol. 4163 of *Lecture Notes in Computer Science*, pp. 215–228. Springer-Verlag, September 2006.

P. Taylor. *Text-to-Speech Synthesis*. Cambridge University Press, 2009.

T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerends, e C. Colomes. PEAQ - The ITU Standard for Objective Measurement of Perceived Audio Quality. *Journal of the Audio Engineering Society*, vol. 48, no. 1/2, pp. 3–29, February 2000.

J. Timmis. *Artificial Immune Systems: A Novel Data Analysis Technique Inspired by the Immune Network Theory*. Tese de Doutorado, Department of Computer Science, University of Whales, September 2000.

J. Timmis e M. Neal. A resource Limited Artificial Immune System for Data Analysis. *Knowledge Based Systems*, vol. 14, no. 3-4, pp. 121–130, June 2001.

SOM toolbox. Website acessado em agosto. http://www.cis.hut.fi/projects/somtoolbox, 2008.

R. P. V. Violato, A. G. Azzolini, e F. J. Von Zuben. Antibodies with Adaptive Radius as Prototypes of High-Dimensional Datasets. In *Proceedings of the 9th International Conference on Artificial Immune Systems (ICARIS'2010), Lecture Notes in Computer Science*, vol. 6209, pp. 158–170, July 2010.

R. P. V. Violato, F. J. Von Zuben, F. O. Simoes, M. Uliani Neto, E. J. Nagle, F. O. Runstein, e L. C. T. Gomes. Agrupamento Sensível à Densidade para a Quantização de Sinais de Fala. In 30º *Congresso Ibero-Latino-Americano de Métodos Computacionais em Engenharia (CILAMCE 2009)*, Novembro 2009.

S. Voran. Objective Estimation of Perceived Speech Quality, Part I: Development of the Measuring Normalizing Block Technique. *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 4, pp. 371–382, July 1999a.

S. Voran. Objective Estimation of Perceived Speech Quality, Part II: Evaluation of the Measuring Normalizing Block Technique. *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 4, pp. 383–390, July 1999b.

S. Wang, A. Sekey, e A. Gersho. An Objective Measure for Predicting Subjective Quality of Speech Coders. *IEEE Journal on Selected Areas in Communications*, vol. 10, no. 5, pp. 819–829, June 1992.

R. Xu e D. C. Wunsch II. Recent Advances in Cluster Analysis. *International Journal of Intelligent Computing and Cybernetics (IJICC)*, vol. 1, no. 4, pp. 484–508, 2008.

E. Zwicker. Subdivision of the Audible Frequency Range into Critical Bands. *Journal of the Acoustical Society of America*, vol. 33, no. 2, pp. 248, February 1961.

E. Zwicker e H. Fastl. *Psychoacoustics, Facts and Models*. Springer Verlag, 1990.