Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação
Departamento de Comunicações

Algoritmos Genéticos para Interpolação Não-Linear de Imagem e Decodificação de Códigos Lineares

Fabbryccio A. C. M. Cardoso

Orientador :
Prof. Dr. Dalton Soares Arantes

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do Título de Mestre em Engenharia Elétrica.

Campinas, SP - Brasil
1998

RESUMO

Investiga-se neste trabalho a aplicação de algoritmos genéticos (AGs) ao problema da otimização de filtros não-lineares bidimensionais, baseados em redes neurais, para interpolação de imagens. Aplica-se também os AGs ao problema da decodificação de códigos corretores de erros, onde se verifica quão importante é a representação cromossômica e a estrutura do problema. Um algoritmo genético padrão foi inicialmente aplicado na otimização desses dois problemas. Introduzindo-se informações adicionais relativas à estrutura do problema, modificações neste algoritmo foram então propostas e implementadas, com o objetivo de melhorar a sua eficiência. No problema da interpolação de imagens, uma informação importante que deve ser preservada são as bordas da imagem. Estudou-se uma forma de utilizar tal informação na amplificação de imagens, visando uma boa qualidade subjetiva. Um algoritmo interpolador não-linear, que explora intensamente as características de bordas, foi implementado e os resultados comparados com as técnicas convencionais.

ABSTRACT

The object of this dissertation is to study and apply Genetic Algorithms (GAs) to the optimization of bidimensional non-linear filters, based on neural networks, for image interpolation. The GAs are also applied to the decoding of binary error-correcting codes, where the important issues of chromosome representation and problem structure are emphasized. A standard genetic algorithm is initially applied to both of these problems, in order to assess its performance. By introducing additional information about the structure of the problems, modifications are then proposed and implemented in order to increase the efficiency of the standard GA. For example, in the image interpolation problem one key aspect is the preservation of image borders. By preserving borders in the amplified image, subjective picture quality is greatly improved. A specific algorithm that exploits these characteristics is implemented and the results compared with those of traditional techniques.

AGRADECIMENTOS

Acredito firmemente no princípio da dualidade aplicado à vida. Da mesma forma que não há o dia sem a noite ou não existe o calor sem o frio, não há realizações se não houver trabalho, dedicação e disciplina. Acabo de cumprir mais uma meta da minha carreira e da minha vida – a conclusão de uma proposta de tese. Foi um período curto, apenas dois anos, mas que representou um acúmulo de conhecimento e experiência que eu jamais havia experimentado antes. A sensação de dever cumprido por ter conseguido realizar algo do qual me orgulho muito, esta tese, é indescritível. Por estes motivos, agradeço de todo o meu coração e de minha alma aos meus pais, principalmente à minha mãe, todo o apoio que me deram através do seu amor, do seu incentivo e da sua dedicação, seja me incentivando, me auxiliando e até mesmo me financiando. Também agradeço ao meu orientador Prof. Dr. Dalton S. Arantes, pessoa cuja inteligência, esforço e atenção a mim dedicados, foram fundamentais para que esta tese fosse terminada com êxito.

Também quero fazer uma homenagem aos amigos Edmilson da Silva Moraes, Marcelo Menezes de Carvalho, Mylene Christine Q. de Farias, Luis Rômulo Mendes, Rene Togni del Pietro, Fernando Cesar Comparsi de Castro e Maria Cristina Felippetto de Castro pelos momentos de descontração e companheirismo, além das inúmeras discussões sobre diversos temas, muitos deles bastante frutíferos. Ao Edmilson, sou ainda muito grato pelas valiosas colaborações a este trabalho.

De fato, estes dois anos foram maravilhosos porque, além de tudo que já descrevi, foi neste período que eu conheci a engenheira Alexandra Maria Rodrigues, que hoje é a minha esposa e meu grande amor. A ela, devo grandes momentos de alegria e ternura, que espero retribuir ao longo de toda a minha vida.

Não poderia deixar de lembrar das agências de fomento à pesquisa, que vêm promovendo o desenvolvimento científico e tecnológico do país. Entre as quais, o CNPq, que financiou o desenvolvimento desta tese de mestrado.

Aos Professores Max H. M. Costa, Márcio Luiz de A. Netto, Fábio Violaro e João Baptista T. Yabu-uti, agradeço as valiosas contribuições e sugestões. Ao CPqD da TELEBRÁS e à Unicamp, agradeço as oportunidades de me familiarizar com áreas avançadas da pesquisa de vídeo/imagem digital.

Acima de tudo, agradeço a Deus, meu Pai, que me sustenta e me guia por caminhos seguros.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1. Objetivos da Tese

Dois temas são abordados nesta tese : interpolação de imagens e decodificação de códigos lineares. Apesar de serem dois tópicos bastante diferentes, eles têm em comum a utilização de algoritmos genéticos, seja para a otimização de um filtro não-linear para interpolação de imagens, seja para encontrar uma palavra código, de um código linear, que é a mais próxima de um vetor binário recebido no que se refere à métrica de Hamming, após transmissão sobre o Canal Binário Simétrico.

Interpolação espacial é uma técnica importante que é freqüentemente utilizada na ampliação de imagens ou, simplesmente, na recuperação da imagem original a partir da sua versão dizimada. São muitas as aplicações práticas desta técnica. Por exemplo, ampliação de vídeo/imagem para monitores de vídeo, televisão de alta definição, scanners ópticos, impressoras de alta resolução, sistemas de transmissão progressiva de imagens, etc. O mercado de microcomputadores (PCs) e estações de trabalho tem crescido bastante nos últimos anos, em razão da utilização de tecnologias avançadas na confecção de CPUs, barramentos de entrada/saída (PCI por exemplo) de altíssima velocidade, discos rígidos de grande capacidade (acima de 4Gbytes), amplos monitores com altas taxas de varredura, bem como de placas aceleradoras de vídeo com elevado poder de processamento gráfico, que suportam altas resoluções de imagem a taxas de 24 bits/pixel ou maiores. Estes sistemas nos tornam aptos a adquirir, armazenar, processar e visualizar imagens de altíssima

resolução. Com toda esta evolução, os dispositivos digitais passaram a incorporar funções de ampliação de imagem diretamente em seus "chips" gráficos. Entretanto, os algoritmos por eles empregados são em geral baseados em técnicas lineares de interpolação que resultam em imagens interpoladas de baixa qualidade. Nesta tese, a ampliação espacial é investigada através do estudo de técnicas não-lineares de interpolação para melhorar a qualidade psico-visual da imagem final interpolada.

De fato, uma contribuição importante deste trabalho está no desenvolvimento e implementação de duas técnicas não-lineares, que melhoram a resolução da imagem interpolada a partir da predição, baseada em modelos de multi-resolução, de informações de alta freqüência. Algoritmos Genéticos e Redes Neurais bidimensionais são utilizados na construção do modelo de predição usando apenas dois níveis de resolução da imagem a ser interpolada. É também utilizado um modelo de predição que enfatiza as características de bordas da imagem, evitado serrilhamento e degradação de bordas. Resultados são apresentados e comparados com ampliações obtidas por técnicas lineares.

Outra contribuição deste trabalho é a aplicação de Algoritmos Genéticos ao problema da decodificação de códigos corretores de erros. A validação desta técnica de decodificação evolucionária é conduzida via simulações que, devido aos interessantes resultados obtidos, apontam para soluções ainda pouco exploradas na decodificação de códigos lineares.

1.2. Motivação

Muitas técnicas de interpolação convencional têm sido utilizadas para aumentar a resolução espacial de uma imagem. Alguma destas incluem replicação de pixel, interpolação bilinear, métodos baseados em splines, etc. Entretanto, estas técnicas têm um desempenho pobre no que se refere ao aspecto subjetivo da imagem interpolada, visto que elas tendem a borrar a imagem ou introduzir artefatos na forma de serrilhamento na imagem interpolada.

A tendência recente tem sido implementar estas técnicas de interpolação em "hardware" para expansão de imagens em tempo real. Isto é particularmente importante para vídeo digital, onde grandes larguras de banda são requeridas. Por exemplo, técnicas de escalonamento de vídeo usam uma largura de banda adicional para transmitir informa-

ções de alta freqüência num sistema de serviço diferenciado. Técnicas de interpolação não-linear permitem a reprodução de vídeo em mais alta resolução sem a necessidade desta largura de banda adicional. Neste caso, as informações de alta freqüência, em vez de transmitidas, são estimadas. Dispositivos de aquisição de imagens, tais como "scanners", também utilizam técnicas de interpolação de modo a aumentar a resolução óptica através de "software". Portanto, está justificada a necessidade e a importância de algoritmos robustos para interpolação de imagens visando, principalmente nos aspectos subjetivos, uma qualidade final melhor do que aquela obtida por métodos lineares.

Já a decodificação de códigos lineares é um problema em que a complexidade da busca cresce rapidamente com o comprimento do código utilizado, ou seja, ocorre o que é chamado de *explosão combinatorial*. Existem métodos que conseguem encontrar excelentes soluções para este problema. Entretanto, exige-se, nestes casos, uma certa riqueza no que se refere às estruturas algébricas do código utilizado, o que resulta em algoritmos de decodificação bastante específicos. Isto justifica a necessidade de se desenvolver algoritmos de decodificação mais robustos, que possam ser empregados na decodificação de códigos sem estrutura algébrica, mas com melhores propriedades de distância mínima.

1.3. Interpolação Não-Linear de Imagens

O problema da interpolação de imagens é *mal condicionado*, porque existe uma infinidade de imagens expandidas que ao serem decimadas geram a mesma imagem. Deste modo, a idéia por trás dos métodos de interpolação aqui desenvolvidos, está na criação de modelos que levem a uma determinada imagem expandida a partir da preservação do conteúdo de freqüência original e adição de informações de mais alta freqüência. Isto deve ser desenvolvido, a fim de realçar certos aspectos, tais como as bordas da imagem, ou para melhorar a resolução em freqüência da imagem expandida, obtendo assim imagens menos borradas do que as obtidas por métodos tradicionais.

Numa primeira etapa, são utilizadas redes neurais como filtros bidimensionais não-lineares, para extrair, a partir de duas resoluções da imagem original (ela própria e a sua versão dizimada), um modelo de expansão que seja capaz de enriquecer a imagem interpolada. Estes filtros são otimizados para atender uma medida objetiva de qualidade.

Nesta tese, foi utilizado o erro quadrático médio (EQM). Devido à estrutura não-linear do filtro empregado, sua otimização, com relação ao EQM, leva a uma superfície contínua, derivável e multimodal. As características de continuidade e existência de derivadas de primeira e segunda ordem da superfície de erro são importantes, pois nos habilitam a utilizar métodos de otimização baseados no gradiente. Todavia, a multimodalidade, ou seja, o fato da superfície de erro ser composta de muitos vales e picos, torna necessária a utilização de algoritmos que não fiquem presos em pontos de ótimo local. Com isto em mente, neste trabalho utilizamos algoritmos genéticos (AGs), assim como algoritmos híbridos que associam métodos de gradiente com AGs, para a otimização dos filtros.

Numa segunda etapa, implementamos um modelo de expansão que pode ser utilizado em qualquer imagem, sem a necessidade de treinamento. Este modelo de expansão consiste em interpolar a imagem de forma adaptativa, considerando informações de bordas locais. A idéia é utilizar operações lineares, sempre discriminando, dentro de uma janela de convolução, duas regiões, cujas fronteiras são as bordas. Mais ainda, esta estratégia nos habilita a controlar o grau de degradação desejado para as bordas e, ao mesmo tempo, evita que ocorra serrilhamento nas mesmas.

1.4. Decodificação Genética de Códigos Lineares

Nesta tese, utilizamos algoritmos genéticos também para implementar um decodificador para o caso mais geral onde se conhece apenas a matriz geradora $\mathbf{G}$ do código utilizado. Entretanto, as manipulações cromossômicas usuais, definidas pelos operadores genéticos e realizadas diretamente sobre as palavras código, podem gerar palavras que não pertencem ao código utilizado. O método aqui proposto utiliza os conjuntos de informação para garantir que toda possível solução seja uma palavra código, isto é, esteja dentro do espaço de busca. Isto é feito representando cada solução por dois cromossomos, sendo que um deles é um conjunto de informação e o outro, um vetor qualquer $\mathbf{x}$, de comprimento igual ao da palavra código, que pode estar, ou não, dentro do espaço de busca. Desta forma, os operadores de seleção, cruzamento e mutação são aplicados sobre os indivíduos para que a população evoluça na direção do vetor recebido, de modo a decodificá-lo corretamente.

Uma variante desta estratégia, que aumenta significativamente o desempenho do algo-

ritmo, consiste em manter o vetor $\mathbf{x}$ sempre numa vizinhança do vetor recebido e aplicar as operações genéticas também aos conjuntos de informação.

1.5. Organização da Tese

Neste capítulo, apresentamos ao leitor o tema da tese e o que motivou tal escolha. Definimos quais os problemas a serem atacados e, através de um breve resumo dos métodos aqui desenvolvidos e implementados, apresentamos as possíveis soluções para os mesmos. A seguir, descrevemos sucintamente cada capítulo desta tese, com o objetivo de facilitar a sua leitura.

No Capítulo 2, apresentamos as propriedades que tornam os algoritmos genéticos tão importantes no que se refere ao problema de busca e otimização. Descrevemos a estrutura básica desses algoritmos, qual o papel dos operadores genéticos utilizados nessa estrutura, assim como as possíveis variações na estrutura e nos operadores de modo a superar certas limitações existentes no algoritmo básico. Um exemplo de implementação de um algoritmo genético é desenvolvido passo a passo para um problema simples, com o objetivo de facilitar o seu entendimento.

No Capítulo 3, damos uma visão geral sobre as diversas técnicas, lineares e não-lineares, de interpolação de imagens. Também é feita uma revisão de redes neurais e a sua utilização como filtro não-linear bidimensional de fase zero. Apresentamos ainda, um algoritmo genético (AG), seguido de uma demonstração formal de convergência, bem como alguns esquemas híbridos envolvendo AGs e métodos de otimização baseados em gradiente. Todos estes esquemas apresentados visam a otimização do filtro não-linear bidimensional. Para finalizar, são apresentados dois esquemas de interpolação. O primeiro é um sistema interpolador baseado em redes neurais e o segundo é um interpolador adaptativo para preservação de bordas.

No Capítulo 4, são apresentados os resultados experimentais obtidos dos esquemas do Capítulo 3. São comparados os diferentes algoritmos de otimização – gradiente conjugado, algoritmo genético e esquemas híbridos AG-Gradiente – implementados para otimizar o filtro não-linear bidimensional. Também são apresentadas imagens expandidas pelos dois esquemas de interpolação aqui desenvolvidos, e validada a hipótese de que a rede neural é

capaz de representar o modelo de interpolação através de apenas dois níveis de resolução da imagem a ser expandida.

No Capítulo 5, apresentamos a estrutura do AG utilizado na decodificação de códigos lineares, assim como a representação cromossômica adotada e os diferentes operadores genéticos implementados. Resultados de simulação são obtidos com o objetivo de validar o método e comparar a eficiência dos diferentes esquemas de decodificação por AGs.

As conclusões deste trabalho são apresentadas no Capítulo 6, assim como as várias possibilidades de trabalhos futuros originadas a partir desta tese.

Capítulo 2

Algoritmos Genéticos

2.1. Introdução

A maioria dos problemas científicos pode ser formulada como um problema de busca e otimização, podendo ser visto como uma *caixa preta* contendo entradas, saídas e um determinado número de parâmetros a serem determinados. Deseja-se então determinar tais parâmetros de acordo com uma função objetivo $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, sendo que n é o número de entradas e m é o número de saídas. Um exemplo de tal função objetivo é o erro quadrático médio entre a saída do sistema, para uma determinada entrada, e a saída desejada.

Neste caso, os parâmetros podem ser obtidos considerando apenas problemas de minimização. Dado uma função $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ e um espaço de busca $S \subseteq \mathbb{R}^h$, o problema de otimização pode ser formulado como

$$\text{minimizar } f(\mathbf{s}) \,|\mathbf{s} \in S \,.$$

Em termos de busca, o mesmo problema pode ser escrito na forma

$$\text{encontrar } \mathbf{s}^* \,|f(\mathbf{s}^*) \leq f(\mathbf{s}) \,, \;\; \forall \mathbf{s} \in S \,.$$

Portanto, existe uma dualidade entre as duas soluções, de modo que um problema de otimização pode ser considerado um problema de busca e vice-versa. Neste trabalho, os dois termos são utilizados indistintamente.

O interessante do problema de busca ou de otimização é a diversidade na forma da

função objetivo, que pode ser derivável, ou não, uni- ou multi-dimensional e uni- ou multimodal, ou seja, com muitos vales e picos.

Existem basicamente três métodos de otimização : os numéricos, os enumerativos e os probabilísticos. Existe também uma quantidade enorme de técnicas híbridas.

A categoria de algoritmos que chamamos de métodos numéricos engloba praticamente todos os algoritmos tradicionais, que se baseiam no gradiente, para ajustar os parâmetros do sistema recursivamente. A idéia básica é ter um ponto de partida qualquer e calcular o ponto seguinte com base no gradiente local. Também podemos olhar para a concavidade da função objetivo na vizinhança do ponto de referência e calcular o ponto seguinte a partir das segundas derivadas parciais, assim como das primeiras. Normalmente, as derivadas parciais são aproximadas localmente em cada iteração. Métodos de gradiente-conjugado, quase-Newtonianos e estratégia de métrica variável utilizam esta idéia de aproximação iterativa das derivadas parciais. Uma restrição fundamental é que a busca realizada por tais algoritmos é local. Esta é uma característica dos métodos de gradiente, que são ineficientes quando a função objetivo é multimodal. De fato, se tais métodos forem empregados num espaço de busca multimodal, os algoritmos convergirão para um ótimo local que dependerá do ponto de partida utilizado. Estes algoritmos também são incapazes de lidar com problemas em que a função objetivo é não diferenciável e/ou descontínua. A grande maioria das funções relacionadas com problemas extraídos do mundo real são descontínuas, ruidosas e/ou multimodais. Portanto, é de se esperar que métodos com restrições de continuidade e existência de derivada não são apropriados para todos os possíveis problemas de busca e otimização e sim para um domínio de problemas bastante limitado.

Existe uma grande variedade de esquemas enumerativos. A idéia é bastante intuitiva e basicamente imita o tipo de busca que nós humanos realizamos no dia a dia : examinam ponto a ponto o espaço de busca atrás do ótimo global. Apesar da simplicidade deste tipo de algoritmo ser bastante atrativa, tal esquema deve ser desconsiderado por uma simples razão : ineficiência quando o espaço de busca é extremamente grande ou infinito. Mesmo os esquemas enumerativos de *programação dinâmica* tornam-se ineficientes no que se refere ao tamanho e à complexidade do espaço de busca.

Já os esquemas probabilísticos, são métodos que se utilizam da escolha probabilística como ferramenta em um processo de busca dirigida. Um exemplo desta técnica é a chamada *têmpera simulada* que utiliza processos aleatórios para auxiliar na busca por estados

de energia mínima. Outro exemplo são os *algoritmos genéticos* (AGs) [4], [7], [27], [28].

AGs são processos de busca aleatória estruturada, que imitam o processo de evolução biológica baseado nas leis da seleção natural e da genética. O algoritmo inicia com uma coleção de estimativas codificadas (*indivíduos*) da solução procurada para o sistema. Esta representação por indivíduos é uma particularidade do algoritmo que não trabalha diretamente no espaço de busca e sim na sua versão parametrizada. A cada indivíduo está associada uma medida da qualidade relativa da estimativa associada à solução desejada para o sistema. A esta medida é dado o nome de *adequabilidade* e à coleção de estimativas de *população*. Em cada geração, ou seja, iteração do algoritmo, indivíduos são selecionados com base na sua adequabilidade para gerar novos indivíduos que formarão a base da próxima geração. A analogia com o processo biológico sugere que tal procedimento conduzirá a soluções eficientes para problemas não lineares complexos.

No aspecto teórico, o AG é um dos paradigmas e métodos englobados pela *Computação Evolucionária* (CE). Outros métodos são a Programação Evolucionária (PE), Estratégias Evolucionárias (EEs), Programação Genética (PG) e Sistemas Classificadores (SCs). Os SCs e a PG podem ser vistos como aplicações de AGs, enquanto a PE e as EEs foram desenvolvidas independentemente dos AGs. Devido à flexibilidade e simplicidade de implementação, além da eficácia na realização de busca global num espaço n-dimensional com muitos picos e vales, os Algoritmos Genéticos são as mais difundidas e estudadas das técnicas de CE.

A partir da discussão realizada sobre os métodos de otimização, concluímos que os métodos de busca convencionais não são robustos. Obviamente esta conclusão é um tanto vaga, visto que não foi feita uma análise exaustiva sobre o grande número de métodos convencionais existentes. Entretanto, a falta de robustez de tais métodos não implica que os mesmos não sejam úteis. De fato, os esquemas convencionais mencionados e um incontável número de métodos híbridos envolvendo tais métodos têm sido utilizados com sucesso em muitas aplicações específicas. À medida que problemas mais complexos são atacados, outros esquemas de busca vão surgindo. Quanto aos AGs, são métodos mais robustos de busca e cuja faixa de aplicações é bem mais ampla do que a dos demais métodos. Por outro lado, para problemas de otimização convexa ou para otimização local de funções contínuas, os métodos baseados no gradiente podem ser muito eficientes. Por este motivo, é comum criar métodos híbridos, envolvendo AGs e métodos convencionais,

para melhorar a eficiência dos AGs de acordo com o problema tratado. Nesta tese são desenvolvidos esquemas híbridos para melhorar a eficiência do AG em aplicações de processamento de sinais (otimização de filtros não-lineares), em particular para interpolação de imagens.

2.2. Características de um Algoritmo Genético

De modo geral, podemos citar quatro características fundamentais dos AGs que os diferenciam das técnicas convencionais, tornando-os algoritmos mais robustos. Primeiro, AGs não trabalham diretamente no espaço de busca e sim num espaço de soluções codificadas. Segundo, a busca realizada pelos AGs é feita a partir de uma população de pontos, e não a partir de um ponto isolado. Terceiro, a única informação utilizada pelos AGs é um valor, calculado a partir de uma função objetivo, associado a cada ponto da população, não necessitando deste modo o uso de derivadas ou qualquer outra informação auxiliar. Finalmente, os AGs usam regras associadas a transições probabilísticas, e não regras determinísticas.

Os AGs requerem um conjunto de estimativas codificadas da solução do problema de otimização, onde toda possível solução é codificada como uma ou mais seqüências de símbolos de algum alfabeto finito A. O mais comum é utilizar o alfabeto binário $A = \{0, 1\}$. Não existe uma regra pré-estabelecida de como tal codificação deva ser feita; de fato, o método de codificação dependerá de cada problema. De acordo com a metáfora biológica usualmente empregada, cada possível solução codificada é chamada de *indivíduo*. Cada indivíduo pode ser representado por um ou mais *cromossomos*, que são as seqüências de símbolos citadas acima. Normalmente, a cada indivíduo é associado apenas um cromossomo, de modo que, quando isto ocorre, os dois termos são utilizados indistintamente.

Muitas técnicas requerem informações auxiliares além daquela representada pelos valores da função objetivo. Por exemplo, métodos baseados no gradiente necessitam das derivadas para maximizar ou minimizar a função objetivo, e outros procedimentos de busca local, como as técnicas de otimização combinatorial, requerem acesso à maioria, se não a todos, os parâmetros tabulados. Ao contrário destes métodos, os AGs não necessitam de nenhuma informação auxiliar, mas apenas dos valores associados à função

objetivo. Por este motivo, diz-se que os algoritmos genéticos são cegos e é esta cegueira que os tornam tão robustos. É verdade que diferentes problemas têm diferentes formas de informações auxiliares, e a não utilização destas informações pode tornar os AGs menos eficientes, em termos de performance (convergência mais lenta), quando comparados com métodos projetados especificamente para um determinado problema. Daí a importância de se alterar o AG padrão, imbutindo nele métodos tradicionais que exploram tais características auxiliares, com o objetivo de acelerar a convergência.

Diferentemente de muitos métodos, os AGs usam regras de transição probabilística para guiar sua busca. Isto não significa que os AGs sejam ingênuas buscas aleatórias ; pelo contrário, os AGs tentam dirigir as buscas para regiões do espaço onde é mais provável encontrar a solução do problema de otimização.

Juntas, estas quatro características – espaço de busca codificado, busca a partir de uma população de pontos, cegueira com relação às informações auxiliares e a utilização de operadores probabilísticos – contribuem para a robustez dos algoritmos genéticos e resultam em vantagens sobre as técnicas usualmente empregadas.

2.3. A Estrutura Básica de um AG

O objetivo desta seção é dar uma visão geral do AG padrão no que se refere aos seus operadores e à sua estrutura básica. Não entraremos em detalhes de implementação, assim como não analisaremos os operadores genéticos em profundidade, mas apenas ilustraremos a função de cada operador.

Antes de tudo, a primeira coisa a ser feita é a definição de uma representação cromossômica para o problema. Em muitos problemas, isto pode ser feito, determinando, primeiro, qual o espaço de valores sobre os quais os parâmetros estão definidos. Em seguida, este espaço deve ser quantizado, representando vetorialmente, através de um número finito de bits, cada valor decimal em uma base binária.

Uma vez definida a representação cromossômica, o próximo passo é gerar um conjunto de soluções-candidatas que irá constituir a população inicial para o AG. Sobre esse conjunto de pontos são aplicados os operadores genéticos. A função desses operadores é alterar os elementos da população, de modo que ela evolua na direção do ótimo global.

Além disso, o tamanho da população é alterado para mais ou para menos de acordo com o operador aplicado. Porém, o tamanho básico, no início e no final de cada iteração, é sempre o mesmo. Como uma referência à evolução Darwiniana, cada iteração de um AG é chamada de geração. Isto não implica, porém, que todos os individuos da próxima geração sejam *filhos* da geração anterior, porque pode haver indivíduos que devido à sua alta adequabilidade (aptidão) sobrevivam nas gerações seguintes.

Não há uma regra geral no que se refere à inicialização da população. Pode-se gerar, a partir de um gerador de números aleatórios, N indivíduos que integrarão a população inicial. Também pode-se fazer uso de algum processo heurístico para iniciar a população. Porém, seja qual for o método de inicialização, é importante ter em mente que, do mesmo modo que no caso biológico, a evolução está diretamente relacionada com a *diversidade* da população. De fato, uma população diversificada implica que os seus indivíduos constituem uma boa amostragem do espaço de busca.

Outro aspecto da implementação do AG, que pode provocar dúvidas, é a condição de finalização do algoritmo. Esta não é uma questão trivial quando a função objetivo é intrinsicamente multimodal. O AG deveria parar quando encontrasse o ótimo global. Entretanto, tal ponto ótimo é desconhecido. Para contornar este problema, pode-se estabelecer que, se o AG ficar preso em um determinado ponto durante um certo número de gerações, o algoritmo deve ser finalizado. Entretanto, isto é perigoso porque estes pontos podem não corresponder ao ponto ótimo global. Por causa deste critério de parada, o AG pode convergir para um ponto sub-ótimo, apesar de se poder provar a convergência do AG para o ponto ótimo global.

A avaliação em um algoritmo genético é fundamental, pois é ela que atribui ao indivíduo um valor que expressará a sua adequabilidade relativa no processo de evolução. A partir da analogia com o processo biológico, esta medida refletiria o quanto cada indivíduo está adequado (adaptado) ao meio ambiente em que vive. Por este motivo, tal medida é chamada de adequabilidade. Portanto, a avaliação irá medir a capacidade que um indivíduo tem de sobreviver ao longo das gerações futuras, gerando descendentes semelhantes a ele. Em muitos problemas esta medida é o próprio valor da função objetivo que se quer otimizar.

Na Figura 2.1 é ilustrado o algoritmo com três operadores básicos, que são *seleção*, *cruzamento* e *mutação*. Este algoritmo revela bons resultados em muitos problemas prá-

ticos.

O operador seleção, em um AG, tenta imitar o processo de seleção natural da evolução Darwiniana. Isto significa que indivíduos são eliminados pela adversidade do meio (competição, predadores, doenças e outros obstáculos), deixando apenas os indivíduos mais fortes, ou seja, aqueles mais adequados ao meio. Aqueles que sobreviverem irão se reproduzir, passando seu material genético adiante. Observe que tal operador reduz o tamanho da população original, digamos de N para D indivíduos. Entretanto, não se está simplesmente escolhendo os D melhores indivíduos, porque não é isso que ocorre na natureza. Assim, com o objetivo de preservar a diversidade da população, um indivíduo fraco pode vir a sobreviver ou, ao contrário, um indivíduo forte pode vir a morrer. De fato, na natureza o indivíduo mais forte tem maior probabilidade de sobrevivência do que um indivíduo mais fraco. Com isto em mente, deve-se associar, a cada indivíduo da população, uma probabilidade de sobreviver (ser selecionado), que será calculada de acordo com a sua adequabilidade.

A partir das probabilidades de sobrevivência de cada indivíduo, constroi-se um processo de seleção análogo a uma roleta. Neste processo, cada indivíduo está associado um *slot* na roleta e o tamanho de cada *slot* está diretamente associado com a adequabilidade. Na Figura 2.2, mostra-se uma distribuição hipotética para um tamanho de população igual a quatro. Deste modo, para se selecionar D indivíduos, basta rodar a roleta D vezes.

Após o operador seleção ter sido aplicado à população P, o tamanho da mesma é reduzido de N para D. Assim, a partir de P obtém-se uma subpopulação de tamanho $D < N$. O passo seguinte é aplicar o operador cruzamento para que novos indivíduos, semelhantes aos pais, sejam gerados e restaurem o tamanho da população base, ou seja, uma população de tamanho N.

Cruzamento é um operador que imita o processo biológico de reprodução sexuada. A idéia básica é realizar a troca de material genético entre dois indivíduos para gerar um ou mais filhos. Um procedimento simples de cruzamento, apresentado na Figura 2.3 e empregado ao caso em que um indivíduo é representado por apenas um cromossomo, pode ser desenvolvido em dois passos. Primeiro, um par de indivíduos é escolhido aleatoriamente da população. Em seguida, os dois cromossomos são recombinados da seguinte forma : uma posição inteira k é gerada aleatoriamente com distribuição uniforme no intervalos $[1, l-1]$, sendo l o comprimento do cromossomo. Como ilustração, suponha o seguinte

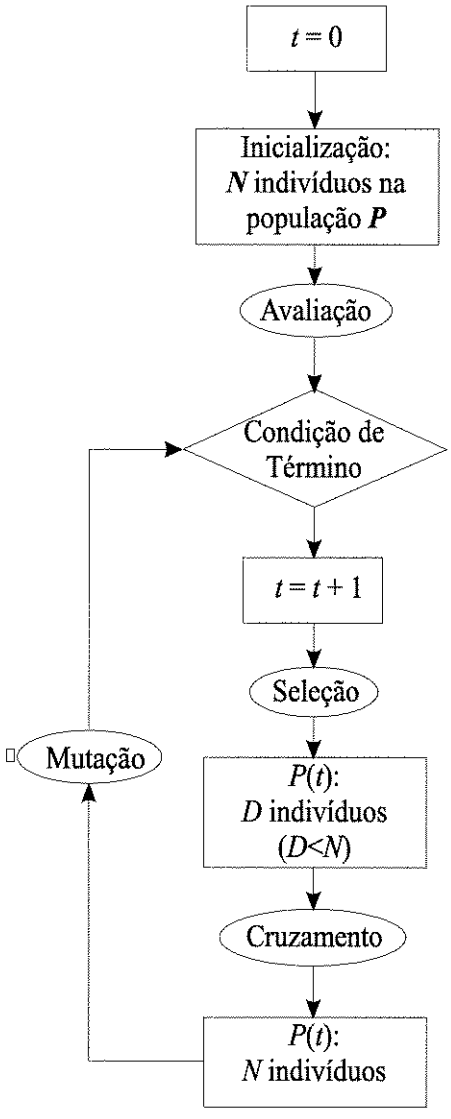

Figura 2.1: Estrutura básica de um AG com três operadores : seleção, cruzamento e mutação.

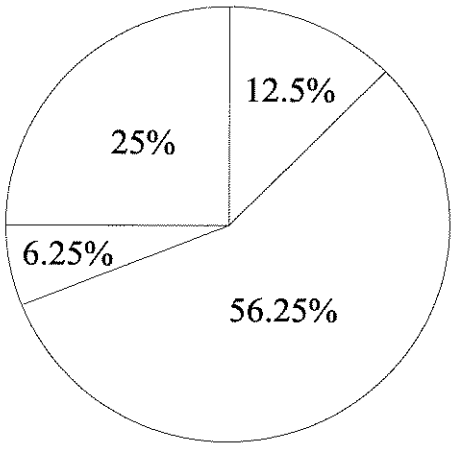

Figura 2.2: Exemplo de roleta utilizada pelo operador seleção. Neste exemplo, a população é de tamanho quatro e os valores dos slots são hipotéticos.

par de cromossomos :

$$C_1 = 11101\,|1011$$
$$C_2 = 00001\,|0001$$

Neste caso, a posição k escolhida aleatoriamente entre 1 e 8 foi $k = 4$, como indicado acima. O resultado da recombinação é :

$$C_3 = 11101\,|0001$$
$$C_4 = 00001\,|1011$$

O operador cruzamento realiza então a *recombinação cromossômica* com o objetivo de explorar a bagagem genética da população para encontrar o melhor indivíduo possível. Visto que o tamanho da população é finito, todas as possíveis recombinações de indivíduos da população irá cobrir uma região finita do espaço de busca, cujo tamanho dependerá do tamanho da população e da diversidade da mesma. Por este motivo, a recombinação é equivalente a uma busca local porque, qualquer que seja o indivíduo gerado por tal operação, ele se encontrará dentro desta região finita do espaço de busca. Portanto, se o ponto ótimo estiver fora desta região, ele jamais poderá ser encontrado apenas com os operadores seleção e cruzamento. O operador *seleção* tem a função de reduzir a carga genética da população de modo a reduzir a região de busca local realizada pelo operador cruzamento. Essa restrição da região de busca local é feita de forma inteligente, ou seja,

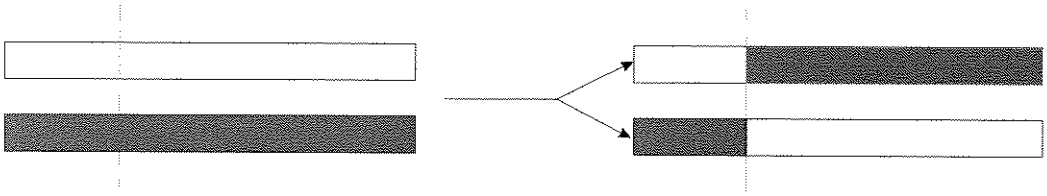

Figura 2.3: Recombinação de dois cromossomos pais para gerar dois cromossomos filhos.

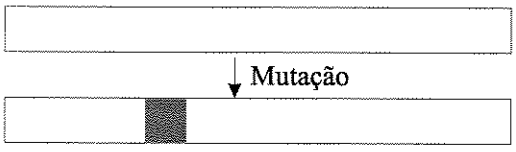

Figura 2.4: Processo de mutação.

ela é feita de modo que a região de busca resultante continue englobando o ponto de ótimo existente nesta região que pode ser ou não o ótimo global.

Quando o operador mutação é aplicado em uma população, todos os cromosssomos da população serão percorridos, posição por posição, havendo uma probabilidade p_{mut} de se alterar o valor associado à posição atual. O processo é ilustrado com um cromossomo na Figura 2.4. A alteração aleatória de indivíduos implica em uma modificação da bagagem genética original da população, de modo a alterar a região de busca da recombinação. Portanto, a mutação faz com que o AG tenha maior mobilidade, e não fique preso a uma única região do espaço, podendo alcançar o ótimo global mesmo que ele não se encontre na região de busca definida pela população inicial e pelas operações de seleção e cruzamento. Em outras palavras, a mutação explora novas regiões do espaço, realizando uma busca global. Outra função do operador mutação consiste em renovar a carga genética perdida devido ao operador seleção, ampliando a região da busca local a ser realizada pelo operador cruzamento.

2.4. Problemas Práticos do AG Padrão

Como a seleção é feita com base na probabilidade de cada indivíduo sobreviver, pode-se perder o melhor indivíduo da população, o que tornaria a convergência bem mais lenta. Além disso, este método permite a seleção de um indivíduo já selecionado. Isto

faria com que se perdesse informação genética importante, visto que é totalmente possível que apenas indivíduos ruins sejam selecionados. Por outro lado, o mais provável, após a seleção, é a multiplicidade de cópias do melhor indivíduo, devido a sua alta adequabilidade com relação aos demais. Isto implicaria em *erros de amostragem* na população resultante. Trata-se da seleção repetida de um mesmo indivíduo gerando um número de cópias bem maior do que o esperado, sendo que o número esperado de cópias de um mesmo indivíduo é dado multiplicando-se a sua probabilidade de seleção pelo tamanho da população. Erros de amostragem podem resultar na perda de informação genética importante e resultados de baixa qualidade devido à convergência prematura do AG para uma solução sub-ótima. Em outra seção, são discutidas formas de melhorar o operador, mas já podemos adiantar que uma forma de contornar este problema é realizar a seleção sem repetição. Seleção com repetição é o caso mais simples, pois a roleta é implementada apenas uma vez por geração para gerar a subpopulação que irá se reproduzir e restaurar a população base. Se na subpopulação selecionada for proibida a ocorrência de indivíduos repetidos, então, cada vez que um indivíduo é selecionado, o *slot* correspondente àquele indivíduo é retirado e a roleta é reimplementada. Isto é, a distribuição de probabilidade é renormalizada. Nesta tese, adotamos esta roleta variável.

Outro problema relacionado com a seleção é a *perda de diversidade* da população. *Diversidade* é um conceito abstrato que indica o grau em que as mais diversas regiões do espaço de busca estão representadas na população, ou seja, quão variada é a população. Este problema ocorre, por exemplo, quando se seleciona apenas os melhores indivíduos da população, que embora sejam diferentes, levam a uma região (definida pela recombinação destes indivíduos) do espaço de busca dominada apenas por um ótimo local. Entretanto, a perda de diversidade é desejada quando a região, gerada por todas as possíveis recombinações dos indivíduos da população, contiver o ótimo global.

Visto que o ato de selecionar indivíduos da população implica em perda de material genético e, conseqüentemente, numa redução da região alcançável pela recombinação dos indivíduos da população, então pode-se afirmar que se não houver mutação ou se a probabilidade de mutação for pequena, independentemente do método de seleção adotado, ocorrerá perda de diversidade. Isto significa que diferentes métodos de seleção podem ser comparados entre si no que se refere a erros de amostragem.

A avaliação dos indivíduos determinará o grau de discriminação dos indivíduos na

população. Por exemplo, dados dois indivíduos diferentes, se a nota dada ao melhor indivíduo relativa ao outro for muito mais alta significa que a probabilidade dele ser selecionado também é bem mais alta, ou seja, a discriminação é maior. Portanto, quanto maior for a discriminação entre os indivíduos, maior será a perda de diversidade. Por este motivo, afirma-se que a perda de diversidade numa população irá depender do método de avaliação adotado.

O operador mutação atenua o problema da perda de diversidade, visto que este operador amplia a região de busca local a ser realizada pela recombinação. Todavia, não é possível resolver o problema da perda de diversidade através da mutação sem que isso afete o dempenho do AG, visto que isso poderia quebrar o balanço existente entre seleção-recombinação e mutação.

De fato, o balanço entre seleção-recombinação e mutação é um dilema em AGs. Como já foi dito anteriormente, recombinação realiza um tipo de busca localizada, visto que utiliza a bagagem genética da população para gerar novos indivíduos; a seleção restringe a região de busca local da recombinação, visto que material genético é descartado; e a mutação torna possível ao AG explorar novas regiões do espaço de busca, ampliando e deslocando a região da busca local a ser realizada pela recombinação. Portanto, muita seleção-recombinação acelera a convergência do algoritmo, mas pode tornar a busca demasiadamente localizada, ao passo que muita mutação pode tornar a busca demasiadamente desordenada.

Até aqui vimos que os erros de amostragem e a perda de diversidade levam o AG a uma convergência prematura para uma solução que não é o ótimo global. Além disso, o ótimo global pode não ser encontrado devido à natureza do problema que é enganadora. Problemas enganadores surgem principalmente devido à representação binária dos cromossomos. Nesses problemas existem fragmentos cromossômicos com alta adequabilidade que juntos, em um único cromossomo, representam uma solução sub-ótima. Uma estratégia para contornar este problema é aplicar uma transformação no espaço de busca de modo que o problema se torne mais fácil para o AG.

Tendo em vista que os AGs operam com várias "frentes de busca", caracterizadas pelos indivíduos da população, o cálculo da função de adequabilidade pode se tornar o ponto crítico na implementação do algoritmo. Embora os operadores genéticos sejam sempre de muito baixa complexidade, a avaliação dos cromossomos pode demandar um grande pro-

cessamento para certos problemas. Por exemplo, em processamento de sinais, a avaliação pode envolver um banco de filtros aplicados a certos sinais, como imagens, voz, etc. Neste caso, para cada cromossomo numa geração, é necessário filtrar o sinal a partir deste banco de filtros. Uma solução para este problema pode estar na implementação paralela do AG. Em vez de se ter uma grande população evoluindo ao longo das gerações, ter-se-ía várias populações pequenas evoluindo paralelamente. Obviamente, estas populações terão de interagir de alguma forma. Um estudo bastante interessante das diferentes possibilidades de se implementar um AG paralelo é apresentado em Goldberg [4].

2.5. Outros Modelos Populacionais

Nesta Seção são apresentadas várias alternativas ao modelo populacional do AG padrão, que apenas produz uma nova população e substitui a anterior sem impor nenhuma condição a este processo. Obviamente, a imposição de condições à geração de uma nova população, para substituir a anterior, aumentará a complexidade do algoritmo, sendo necessário um tempo maior de execução para o mesmo número de gerações.

Uma condição que poderia ser imposta é a **não existência de duplicatas** na população. Esta é uma medida que atenua o problema da rápida perda de diversidade de um AG simples. Há inumeras maneiras de se implementar esta condição. Uma delas seria a geração aleatória de indivíduos para substituir as duplicatas, que poderia ser encarada como uma imigração de indivíduos, vindos de outras regiões do espaço de busca. Poder-se-ía ainda mudar o valor de uma ou mais posições, aleatoriamente escolhidas, do cromossomo a ser substituído, o que corresponderia a um tipo de mutação forçada. Outra forma, que foi implementada nesta Tese, é a utilização de uma roleta de comprimento variável, para evitar que um indivíduo já selecionado seja selecionado novamente.

Também poder-se-ía impor a condição de que o melhor indivíduo fosse mantido na população seguinte. Esta é uma condição razoável, porque evita que tal indivíduo seja perdido durante o processo de seleção, e é conhecida como **estratégia elitista**. Isto significa aumento na velocidade de convergência do algoritmo sem perda de diversidade. Uma implementação desta estratégia consiste em gerar uma nova população a partir do par de operadores seleção-recombinação. Neste caso, antes de substituir a população

antiga, deve-se investigar se o melhor indivíduo da nova população é melhor ou não do que o melhor indivíduo da população anterior; se não for melhor, substitui-se o pior da nova população pelo melhor da população anterior.

Como já foi dito anteriormente, em um AG padrão, uma nova população é gerada a partir da aplicação do par de operadores seleção-recombinação. Com este procedimento, pais com adequabilidade maior do que a dos filhos são perdidos. Este problema pode ser facilmente contornado se a ordem dos operadores for invertida. Primeiro aplica-se o operador recombinação direto na população base (N pais) para gerar uma população com N filhos. Pais e filhos são unidos em uma só população com tamanho $2N$ e, em seguida, é aplicado o operador seleção para gerar a nova população. A desvantagem deste método é que ele requer o dobro de avaliações de indivíduos por geração, e é a avaliação que em geral eleva o custo do AG. Este procedimento é conhecido como **modelo de geração contínua** e é ilustrado na Figura 2.5.

Outra forma de evitar a perda de diversidade de uma população é impor que todos os indivíduos da população, exceto o mais adequado, sejam eliminados e no lugar deles sejam gerados aleatoriamente outros indivíduos. Esta técnica é chamada de **extinção e imigração**. Extinção porque em uma população com N indivíduos, $N-1$ são eliminados, e imigração porque outros $N-1$ indivíduos imigram de outras regiões do espaço de busca para preencher a população. A extinção e imigração devem ocorrer quando não houver melhora na adequabilidade do melhor indivíduo num determinado número de gerações. Esta é uma técnica simples que não aumenta a complexidade do algoritmo.

2.6. Métodos de Avaliação

Em geral, não é uma estratégia inteligente utilizar diretamente a função objetivo como uma medida de adequabilidade, mesmo quando a função a ser otimizada seja real e seus valores sempre positivos. O mapeamento *função objetivo* $f(x) \rightarrow$ *função adequabilidade* $a(x)$ é fundamental, e deve ser feito com muito cuidado.

Devido ao método de seleção, a função adequabilidade deve ter certas características. Primeiro, os valores de adequabilidade devem ser sempre positivos. Uma forma de sempre se obter valores positivos de adequabilidade é somar, ao valor da função objetivo dos

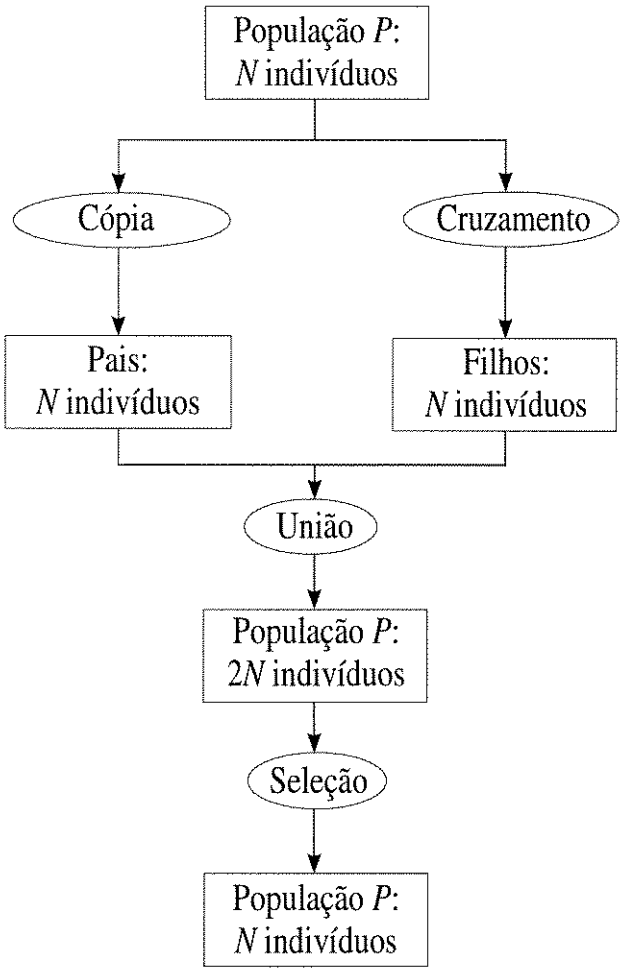

Figura 2.5: Modelo de geração contínua. A população da geração seguinte é obtida a partir da aplicação do operador seleção numa população constituída por pais e filhos, sendo que os filhos foram obtidos a partir dos pais por recombinação.

indivíduos, o valor absoluto $C_{\min}$ obtido a partir do menor valor que a função objetivo assume na população, isto é,

$$a(x) = f(x) + C_{\min}.$$

Segundo, é interessante que a adequabilidade de um determinado indivíduo seja tal que, quanto mais alta, maiores sejam as chaces de sobrevivência. Portanto, se o problema de otimização for a minimização de uma função objetivo, a função adequabilidade pode ser obtida multiplicando os valores da função objetivo por -1. Entretanto, este procedimento não garante que todos os possíveis valores de adequabilidade sejam positivos. A idéia é transformar o problema de minimização em um problema de maximização. Por este motivo, a seguinte transformação é comumente utilizada

$$a(x) = C_{\max} - f(x),$$

onde $C_{\max}$ pode ser escolhido como sendo o maior valor de $f(x)$ entre os indivíduos da população a ser avaliada.

Definimos *pressão seletiva* como a capacidade de discriminação dos indivíduos de uma população com relação à probabilidade de seleção. Por exemplo, se os indivíduos de uma população têm probabilidades de seleção semelhantes, então diz-se que a pressão seletiva é baixa, caso contrário, se os piores indivíduos têm uma probabilidade de seleção bem menor do que a dos melhores, então diz-se que a pressão seletiva é alta.

A pressão seletiva está implicitamente relacionada com a diversidade da população. Alta pressão seletiva tende a fazer a diversidade cair rapidamente, levando a população a estagnar em poucas gerações. Por outro lado, uma baixa pressão seletiva pode tornar o processo de busca proibitivamente lento.

Portanto, em vez de se utilizar uma pressão seletiva constante ao longo de todas as gerações, pode-se torná-la adaptativa. Nas primeiras gerações utiliza-se uma baixa pressão seletiva, de modo que mesmo indivíduos de baixa qualidade relativa tenham alguma chance de sobreviver. Quando a população já estiver madura, depois de muitas gerações, é interessante aumentar a pressão seletiva, porque os valores de adequabilidade de tais indivíduos já estarão muito próximos entre si.

Neste trabalho apresentamos quatro técnicas para controlar a pressão seletiva ao longo das gerações : *escalamento linear* ; *graduação* ; *corte* σ; e *nicho e compartilhamento.*

2.6.1. Escalamento Linear

Seja $f(\cdot)$ a função objetivo, que chamaremos de adequabilidade crua, e $a(\cdot)$ a função de adequabilidade efetiva. O mapeamento que leva a adequabilidade crua à adequabilidade efetiva, $a(\cdot) = f(\cdot)$, não é eficiente nos casos em que a adequabilidade crua assume valores semelhantes entre os indivíduos da população. Também, não é eficiente nos casos em que surgem indivíduos excepcionalmente ruins ou bons, quando comparados com o restante da população. No primeiro caso, quando as adequabilidades dos indivíduos da população são semelhantes (variância baixa), a população já está estagnada, ou seja, próximo de um ponto ótimo. Portanto, é interessante aumentar a pressão seletiva de modo que, a partir da recombinação genética, a convergência para o ótimo seja acelerada.

O mapeamento linear $f(\cdot) \rightarrow a(\cdot)$, definido pela equação

$$a(\cdot) = \alpha f(\cdot) + \beta, \tag{2.1}$$

tem por objetivo controlar de forma efetiva a pressão seletiva exercida sobre a população.

Quando os valores da função objetivo, ou adequabilidade crua, são muito próximos entre os indivíduos da população, a população já está madura e indica uma proximidade de um ótimo que pode ser ou não o ótimo global. Se a adequabilidade crua fosse utilizada, todos os indivíduos da população teriam praticamente a mesma chance de sobrevivência, tornando muito lenta a convergência para o ponto ótimo. Além disso, dependendo da natureza do problema, as adequabilidades cruas dos indivíduos podem ser semelhantes, mesmo tratando-se de indivíduos distantes entre si no espaço de busca. Neste caso, é interessante aumentar a pressão seletiva, acelerando a convergência do AG, de modo que uma busca local mais refinada seja feita. Uma maneira de aumentar a pressão seletiva é utilizar um mapeamento linear, como mostrado na Figura 2.6, que aumente a variância da adequabilidade crua sem alterar seu valor médio, distingüindo melhor os indivíduos para seleção.

Por outro lado, quando a adequabilidade crua na população possuir uma alta variância, não é interessante usá-la diretamente, porque a pressão seletiva seria alta. A idéia é baixar a pressão seletiva a partir de uma contração linear, diminuindo a variância da adequabilidade crua mas mantendo a adequabilidade média, como é mostrado na Figura 2.7. Isto é feito para permitir que mesmo indivíduos com baixa qualidade relativa pos-

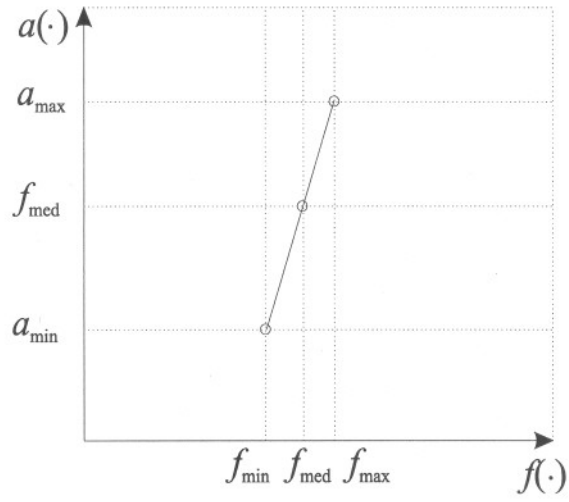

Figura 2.6: Escalamento linear que aumenta a pressão seletiva, preservando a adequabilidade média.

sam sobreviver. Por exemplo, na população pode-se ter indivíduos com qualidade relativa alta, cujo material genético delimita uma região do espaço de busca que não engloba o ótimo global, mas ao mesmo tempo existir um indivíduo com baixa qualidade relativa que pode levar o AG ao ótimo global, a partir da recombinação de seu material genético com o dos outros indivíduos. O importante é a carga genética da população como um todo, porque será ela quem delimitará a região do espaço de busca local a ser explorada pela recombinação.

Existe ainda a possibilidade de aparecer um indivíduo cuja adequabilidade relativa seja muito maior ou menor do que a dos demais indivíduos. No segundo caso, não haveria problemas porque a chance de sobrevivência de tal indivíduo é baixa e não afetaria a pressão seletiva. O problema está no primeiro caso, em que a adequabilidade relativa de um indivíduo é muito alta, porque afetaria a pressão seletiva. Neste caso, as probabilidades de sobrevivências, $p_i = a_i / \sum_j a_j$, dos demais indivíduos seriam semelhantes e bem menores do que a deste super-indivíduo, implicando em perda rápida de diversidade. Para contornar este problema, deve-se mapear a adequabilidade crua máxima para um valor que seja, digamos, um múltiplo da adequabilidade média, $a_{\max} = \gamma f_{med}$. Isto é ilustrado na Figura 2.8.

Com base nos argumentos acima, pode-se utilizar um escalamento linear que realiza o

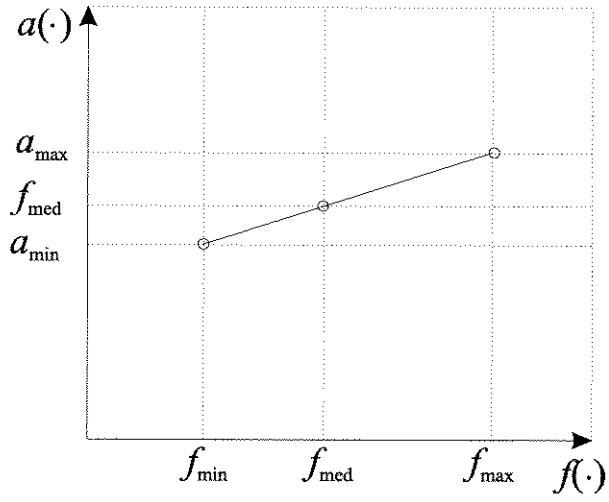

Figura 2.7: Escalamento linear (contrativo) a ser utilizado quando a variância da adequabilidade crua é alta, diminuindo a pressão seletiva.

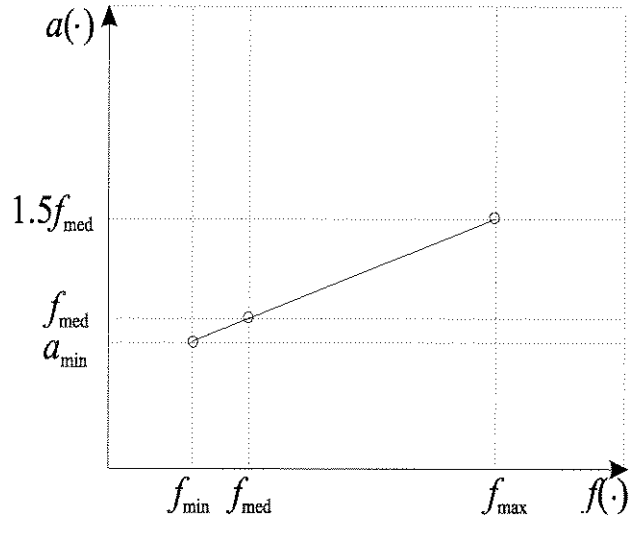

Figura 2.8: Escalamento a ser realizado quando surge um indivíduo muito acima da média. Aqui fixamos $\gamma = 1,5$.

seguinte mapeamento :

$$\begin{aligned} f_{\text{med}} &\rightarrow f_{\text{med}} \\ f_{\text{max}} &\rightarrow \gamma f_{\text{med}} \end{aligned}$$

Deste modo, a partir da equação da reta, podemos escrever

$$\frac{a(\cdot) - f_{med}}{f(\cdot) - f_{med}} = \frac{\gamma f_{med} - f_{med}}{f_{\text{max}} - f_{med}} \tag{2.2}$$

Desenvolvendo (2.2) na forma da Equação 2.1, obtemos :

$$\begin{cases} \alpha = \frac{(\gamma-1) f_{med}}{f_{\text{max}} - f_{med}} \\ \beta = f_{med} \frac{f_{\text{max}} - \gamma f_{med}}{f_{\text{max}} - f_{med}} \end{cases} \tag{2.3}$$

Observe que este mapeamento controla efetivamente a pressão seletiva, atendendo aos casos ilustrados nas Figuras 2.6, 2.7 e 2.8. Por exemplo, no caso em que a variância da adequabilidade crua na população é baixa,

$$<< 2(a_{\text{max}} - a_{\text{med}}) = 2 f_{\text{med}}(\gamma - 1),$$

para $\gamma = 1,5$, a variância da adequabilidade efetiva torna-se maior do que a da adequabilidade crua, aumentando a pressão seletiva; no caso em que a variância da adequabilidade crua é alta, digamos bem maior do que f_{med} (a máxima variância da adequabilidade efetiva para $\gamma = 1,5$), estamos reduzindo a variância e, deste modo, a pressão seletiva. Portanto, a variância máxima da adequabilidade efetiva é fixa e igual a $2 f_{\text{med}}(\gamma - 1)$. Mais ainda, ela representa um limiar, que se adapta à população ao longo das gerações, abaixo do qual a adequabilidade crua da população é aumentada e vice-versa.

Entretanto, o mapeamento proposto insere um inconveniente que é ilustrado na Figura 2.9. Na situação em que surge um indivíduo muito abaixo da média, este mapeamento pode tornar a adequabilidade destes indivíduos negativa. Uma solução para este problema, é realizar outro mapeamento linear que leve a adequabilidade crua mínima, f_{min}, para uma adequabilidade zero, mas que mantenha a adequabilidade média. Em outras palavras, pode-se propor um escalamento linear que realiza o seguinte mapeamento

$$\begin{aligned} f_{\text{med}} &\rightarrow f_{\text{med}} \\ f_{\text{min}} &\rightarrow 0, \end{aligned}$$

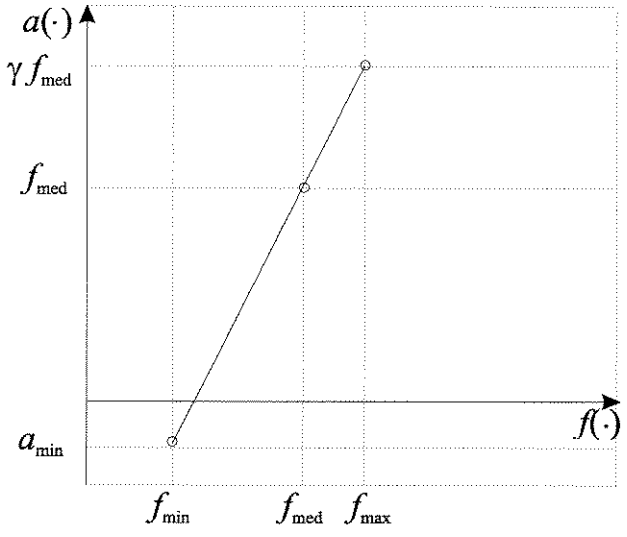

Figura 2.9: Exemplo do caso em que o mapeamento linear escolhido leva a valores negativos de adequabilidade.

que deverá ser utilizado quando $a_{\min}$, no mapeamento da Equação 2.2, assumir valores negativos. Deste modo, a partir da equação da reta, o escalamento linear alternativo pode ser escrito como :

$$\frac{a(\cdot)}{f(\cdot) - f_{\min}} = \frac{f_{\text{med}}}{f_{\text{med}} - f_{\min}}. \tag{2.4}$$

Neste caso, as constantes α e β são então dadas por

$$\begin{cases} \alpha = \frac{f_{\text{med}}}{f_{\text{med}} - f_{\min}} \\ \beta = -\frac{f_{\min} f_{\text{med}}}{f_{\text{med}} - f_{\min}} \end{cases} \tag{2.5}$$

Para resumir tudo o que foi dito acima, os coeficientes α e β da transformação (2.1) devem ser calculados como segue.

Se

$$f_{\min} \geq \frac{f_{\max} - \gamma f_{med}}{\gamma - 1}, \tag{2.6}$$

então

$$\begin{cases} \alpha = \frac{(\gamma-1) f_{med}}{f_{\max} - f_{med}} \\ \beta = f_{med} \frac{f_{\max} - \gamma f_{med}}{f_{\max} - f_{med}}. \end{cases}$$

Caso contrário,

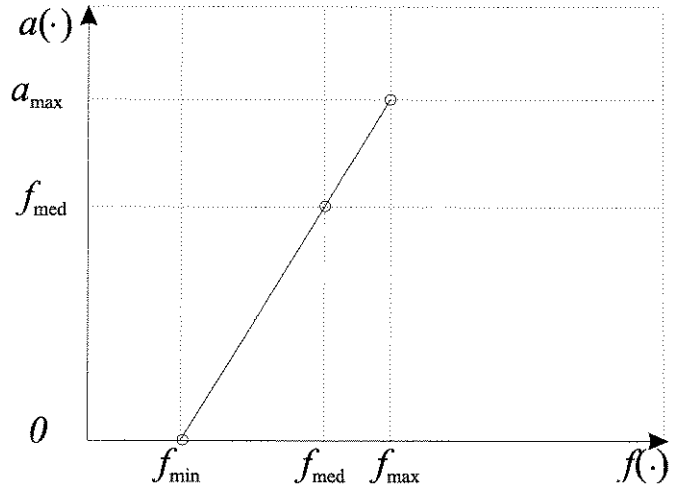

Figura 2.10: Mapeamento a ser utilizado quando o mapeamento original leva a valores negativos de adequabilidade.

$$\begin{cases} \alpha = \frac{f_{med}}{f_{med} - f_{\min}} \\ \beta = -\frac{f_{\min} f_{med}}{f_{med} - f_{\min}} \end{cases}$$

2.6.2. Graduação

Na graduação, a medida de adequabilidade é parcialmente dissociada da função objetivo (adequabilidade crua). Os cromossomos são dispostos em um vetor em ordem crescente, de acordo com o valor da função objetivo; ou seja, os cromossomos são graduados e os valores de adequabilidade são atribuídos de acordo apenas com a ordem. Esta estratégia é interessante porque evita variações indesejáveis da função objetivo.

2.6.3. Corte σ

De modo a evitar que indivíduos muito ruins sejam reproduzidos por erro de amostragem, um método simples é *cortar* todos os pontos cuja adequabilidade seja inferior a um limiar fixo menor do que a média, especificado em termos do desvio padrão (geralmente representado pela letra σ). Por exemplo, pode-se atribuir adequabilidade igual a zero para

todos os indivíduos com adequabilidade menor do que a média menos duas vezes o desvio padrão.

2.6.4. Nicho e Compartilhamento

O processo seletivo na maior parte dos AGs é bastante severo : todos os indivíduos competem entre si, mas os mais fortes da população são privilegiados. Na natureza, contudo, espécies distintas e que tenham diferentes papéis ecológicos não competem entre si e podem conviver em um mesmo ambiente.

Essa idéia levou ao conceito de compartilhamento, em que a adequabilidade dos indivíduos é modificada de tal modo a diminuir o nível de competição entre indivíduos que estejam distantes no espaço de busca. Por exemplo, pode-se usar

$$a(x_i) = \frac{f(x_i)}{\sum_{j=1}^{N} \lambda(\|x_i - x_j\|)},$$

onde $\|x - y\|$ é a distância Euclidiana entre os pontos x e y, e λ é uma função de compartilhamento, que deve ser decrescente em relação à distância entre os pontos. É comum que λ seja definida de modo a resultar igual a 1 para a menor distância entre dois indivíduos da população e 0 para a maior distância.

Na otimização de uma função multimodal, o compartilhamento resulta na formação de subpopulações de indivíduos concentrados nos picos da função. Sem compartilhamento, a tendência do AG seria convergir para um único pico, não necessariamente o mais alto.

Contra este método pesam os fatos de que o cálculo das distâncias requer considerável tempo de processamento ($O(N^2/2)$), e a prática demonstra que as subpopulações, embora próximas aos ótimos locais, geralmente não convergem para eles.

2.7. Operadores Genéticos Melhorados

Existe um grande número de operadores específicos propostos para resolver um dado problema. Nesta seção, estamos interessados em operadores de uso geral que sirvam como métodos alternativos aos operadores empregados no AG simples. Com isto em mente, apresentamos outros métodos de seleção e cruzamento. Entre eles, a roleta sem repetição

que foi proposta por esta tese.

2.7.1. Seleção com Baixo Erro de Amostragem

A forma de seleção que propomos nesta tese, como alternativa à roleta tradicional, é a implementação de uma roleta variável. A roleta tradicional tem comprimento 1 e seus "slots" têm comprimentos correspondentes às probabilidades de seleção de cada indivíduo. Esta roleta é fixa e, portanto, um indivíduo selecionado pode ser escolhido novamente. Para evitar este problema, retiramos o "slot" correspondente ao indivíduo já selecionado, ou seja, fazemos sua probabilidade igual a zero. Isto faz com que o comprimento da roleta seja decrescente, variando de 1, quando nenhum indivíduo foi selecionado ainda, até zero, quando todos os indivíduos foram selecionados. A utilidade desta estratégia é tão maior quanto menor for o tamanho da população, porque, ao evitar duplicatas, faz com que não haja erros de amostragem, mas perde a capacidade de simular o modelo biológico de reprodução assexuada.

Existem outros modelos de seleção, baseados no cálculo do número esperado de cópias de cada indivíduo, que é feito a partir da sua probabilidade de sobrevivência e do tamanho da população. Entretanto, estes modelos de seleção não são muito úteis em populações pequenas, porque, ao se permitir cópias de um mesmo indivíduo, permite-se também uma maior perda de diversidade da população.

2.7.2. Cruzamento

Uma primeira variante de cruzamento se refere ao pareamento de cromossomos, ou seja, em vez de sortearmos aleatoriamente os pares para recombinação, podemos pareá-los de acordo com o grau de adequabilidade. A idéia é parear indivíduos com adequabilidades semelhantes, ou seja, dividir a população em dois grupos – os mais adequados e os menos adequados – e realizar o pareamento dentro de cada grupo.

Outras variantes podem ser usadas na recombinação. A recombinação multiponto utiliza a recombinação tradicional com mais de um ponto de corte (Figura 2.11). Já a recombinação uniforme, utiliza uma seqüência binária auxiliar, aleatoriamente gerada com

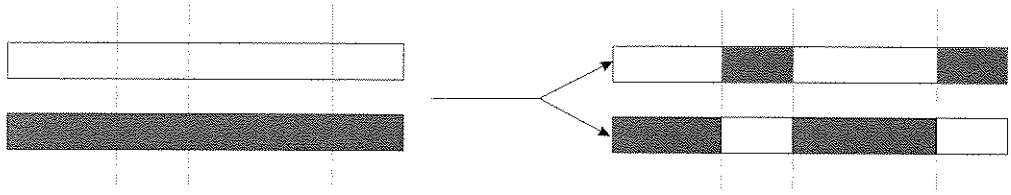

Figura 2.11: Cruzamento multiponto.

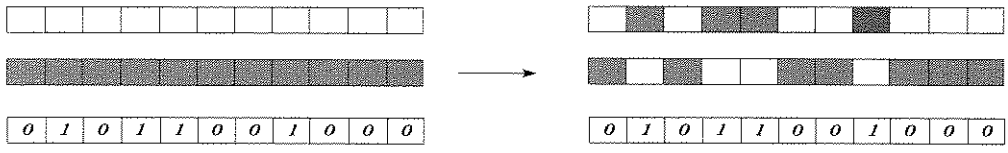

Figura 2.12: Cruzamento Uniforme.

mesmo comprimento dos cromossomos pais, para gerar dois cromossomos filhos a partir da troca dos bits, de mesma posição, dos cromossomos pais : para cada posição da seqüência auxiliar, se for encontrado um "1" realiza-se a troca, caso contrário não se faz nada. A recombinação uniforme é ilustrada na Figura 2.12.

2.8. AGs Híbridos

AGs são algoritmos que em geral não levam em conta a estrutura do problema a ser otimizado. A única informação necessária é o valor da função objetivo calculado nos pontos de interesse. É por este motivo que os AGs são tão robustos. Entretanto, quando existem informações específicas, pode ser vantajoso considerar um AG híbrido. Algoritmos genéticos podem ser combinados com várias técnicas de busca, especificamente projetadas para o problema, de modo a formar um híbrido que explore tanto a perspectiva global do AG como a convergência local da técnica de busca específica.

Otimização local de uma função contínua de uma ou mais variáveis é um problema muito bem conhecido, e inúmeras técnicas baseadas em gradiente estão disponíveis para o cálculo do ótimo local nestes problemas. Para desenvolver um AG híbrido para uma função objetivo multimodal e derivável, podemos simplesmente combinar as técnicas de busca local com o algoritmo genético. Mas, devido às características da busca local de um problema específico, vão existir diferentes técnicas híbridas de acordo com cada problema

ou classe de problemas. Em outras palavras, ao implementar uma técnica híbrida, estamos sacrificando alguma generalidade para utilizar informações específicas do problema com o objetivo de aumentar a eficiência do AG.

Há diversos modos de se implementar um AG híbrido. Uma maneira, talvez a mais óbvia, seja executar o AG puro para obter uma estimativa inicial e então aplicar uma técnica de busca local sobre um certo percentual de pontos da última geração. Uma outra maneira é embutir uma técnica de busca local dentro do AG, a partir da criação de um novo operador. Este operador selecionaria, por exemplo, o melhor indivíduo da população, converteria sua representação cromossômica para sua representação original, aplicaria um número de iterações da busca local ao ponto escolhido e o converteria em seguida para a representação cromossômica. No Capítulo 3, apresentamos três esquemas híbridos aplicados na otimização de um filtro não-linear.

Uma forma de se implementar busca local baseada em gradiente, sem implicar em perda de generalidade do AG, pode ser obtida através da alteração nos k bits de um cromossomo, com o objetivo de melhorar sua adequabilidade. É intuitiva a analogia desta estratégia com o método do gradiente, que adiciona um $\Delta\vec{x}$ ao parâmetro a ser otimizado com o objetivo de se maximizar (ou minimizar) uma função objetivo. O algoritmo *k-bits* pode ser implementado a partir dos seguintes passos :

1. Selecione um ou mais dos melhores indivíduos da população na geração atual;
2. Percorrendo bit a bit a seqüência cromossômica, realize sucessivas mudanças de um bit, retendo as alterações que melhoram a adequabilidade do indivíduo ou indivíduos selecionados;
3. No final da varredura, devolva estes indivíduos à população e continue o fluxo normal do algoritmo genético.

2.9. Um Exemplo de Implementação

Nesta seção pretendemos ilustrar como as idéias apresentadas podem ser utilizadas para minimizar uma função objetivo $f(x)$, a partir do AG da Figura 2.1, preservando em cada geração o melhor indivíduo. A função escolhida é dada por

$$f(x) = -\frac{\sin(4\pi x)}{4\pi x} \tag{2.7}$$

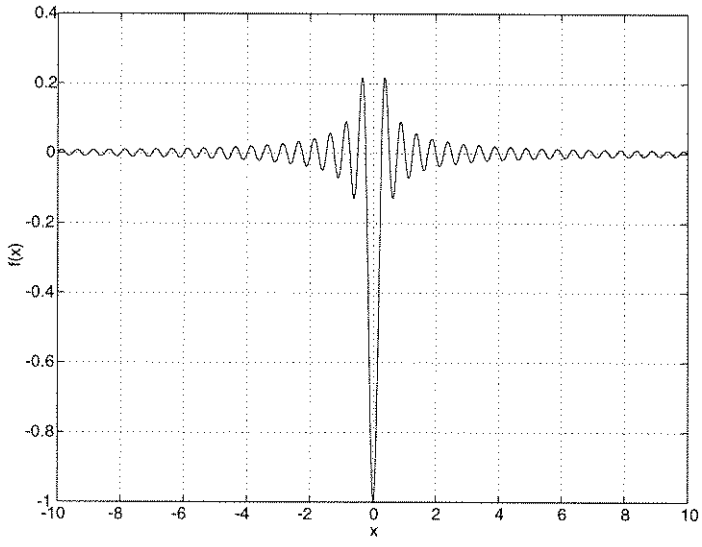

Figura 2.13: Função $f(x)$ que se deseja minimizar.

definida sobre o intervalo $-10 \leq x \leq 10$, sendo que o ponto de mínimo global ocorre em $x = 0$ conforme mostrado na Figura 2.13. Esta é uma função interessante de se analisar pelo fato de ser multimodal com muitos pontos de máximo e de mínimo locais, o que implica que puros métodos de gradiente não devem ser capazes de encontrar o ótimo global na maioria das tentativas.

2.9.1. Representação Cromossômica

Cada possível amostra x do espaço de busca é representada como uma seqüência de bits de comprimento finito L. Conseqüentemente, o espaço de busca terá de ser quantizado, sendo que o passo de quantização corresponderá à precisão desejada. Escolhemos $L = 32$ bits, o que correspnde a um passo de quantização de $20/2^{32} = 4,656612873077 \times 10^{-9}$, ou seja, uma precisão de nove casas decimais.

Uma vez definido o comprimento, em bits, do cromossomo, o próximo passo é mapear o espaço de busca para um espaço de soluções codificadas. A idéia é mapear o intervalo contínuo $-10 \leq x \leq 10$ para um intervalo discreto $0 \leq \bar{x} \leq 2^L - 1$. Computacionalmente, x é do tipo ponto flutuante de 64 bits (tipo *double* em C) e $\bar{x}$ é um inteiro sem sinal de 64 bits (*unsigned long* em C), sendo que utizamos apenas os 32 bits mais significativos do

unsigned long. Desta forma, o mapeamento $x \longrightarrow \bar{x}$ é definido por

$$\bar{x} = \left\lfloor \frac{x - x_{\min}}{x_{\max} - x_{\min}} \left(2^L - 1\right) + 0,5 \right\rfloor \tag{2.8}$$

e o mapeamento inverso $\bar{x} \longrightarrow x$ por

$$x = \frac{\bar{x}}{2^L - 1} \left(x_{\max} - x_{\min}\right) + x_{\min}. \tag{2.9}$$

Não é necessário converter $\bar{x}$ para a base binária se a linguagem de programação utilizada suportar operadores binários, caso contrário será necessário definir um vetor de comprimento L e, em seguida, obter os componentes deste vetor convertendo-se $\bar{x}$ para a base 2. Nesta tese, todos os algoritmos foram implementados em C, que possui operadores binários, tais como *and*, *or* e *xor*. Estes operadores atuam bit a bit nos inteiros envolvidos. Por exemplo, $7\,and\,8 = 0$, $7\,or\,8 = 15$, $7\,xor\,9 = 14$.

2.9.2. Inicialização

Tendo em vista que o AG é um conjunto de operadores atuando sobre uma população de indivíduos (cromossomos) de tamanho $N = 31$, a primeira etapa do algoritmo é a geração de cromossomos para a população inicial, de modo que os indivíduos gerados sejam uma representação, a mais variada possível, do espaço de busca. Com isto em mente, a população de cromossomos pode ser obtida gerando-se aleatoriamente, com distribuição uniforme, inteiros no intervalo $0 \leq \bar{x} \leq 2^L - 1$.

2.9.3. Condição de Término

Visto que o ponto de ótimo não é conhecido, a melhor estratégia é investigar se, depois de um certo número de gerações L_t, não houve alteração no melhor indivíduo ou a alteração foi menor do que uma dada precisão. No algoritmo implementado L_t é feito igual a 20.

2.9.4. Avaliação

A primeira etapa da avaliação consiste em calcular a função adequabilidade do indivíduo x_i, com $i \in [1 \cdots N]$, sendo N o tamanho da população. A função adequabilidade $a(x)$ deve ser sempre positiva e, quanto maior for o seu valor, mais próximo o indivíduo x deve estar da solução global. Com isto em mente, a adequabilidade é dada por

$$a(x_i) = f_{\max} - f(x_i),$$

sendo $f_{\max}$ o máximo valor que $f(x)$ assume na população avaliada. Também, podemos embutir no cálculo de $a(x_i)$ o escalamento linear definido pela equação 2.1 :

$$a(x_i) = \alpha (f_{\max} - f(x_i)) + \beta \tag{2.10}$$

Uma vez calculada a função adequabilidade $a(x_i)$, o próximo passo é calcular a probabilidade de seleção de cada indivíduo i, dada por

$$p(x_i) = \frac{a(x_i)}{\sum_{j=1}^{N} a(x_j)}. \tag{2.11}$$

2.9.5. Seleção

A técnica de seleção utilizada é a da roleta e o número de indivíduos selecionados é igual a $D = 15$. Contudo, a roleta é alterada de modo que não haja duplicata entre os individuos selecionados. O algoritmo é apresentado a seguir :

1. Define-se o inteiro D, como sendo o número de indivíduos a serem selecionados.
2. Define-se um vetor rol cujo número de elementos é igual ao tamanho da população.
3. Define-se o vetor p com N elementos, sendo que $p[i] = p(x_i)$.
4. Define-se um vetor S com D elementos.
5. Para $k = 1$ até $k = D$, faça :
   (a) $rol[1] = p[1]$
   (b) Para $i = 2$ até $i = N$, faça :
      i. $rol[i] = rol[i-1] + p[i]$

(c) fim de Para

(d) Se $rol[N]$ igual a zero, então termine o algoritmo porque não há mais o que ser selecionado.

(e) Gera-se um número aleatório, $prob$, uniformemente distribuído no intervalo de 0 a $rol[N]$;

(f) $selecionado = 0$;

(g) Para $i = 1$ até $i = N$, faça :

i. $selecionado = selecionado + (prob > rol[i])$;

(h) fim de Para

(i) $S[k] = x_{selecionado}$; ou seja, tome o indivíduo selecionado e armazene-o no vetor S.

(j) $p[selecionado] = 0$;

6. fim de Para.

Em vez de ficar renormalizando a distribuição de probabilidade de seleção à medida que indivíduos vão sendo selecionados, a idéia embutida no algoritmo consiste em gerar um número aleatório, $prob$, no intervalo de 0 até $\sum_{i=1}^{N} p(x_i)$. Se $\sum_{i=1}^{M-1} p(x_i) < prob < \sum_{i=1}^{M} p(x_i)$, então seleciona-se o M-ésimo indivíduo da população e iguala-se sua probabilidade de seleção a 0. Feito isto, não há como este indivíduo ser selecionado novamente, pois $\sum_{i=1}^{M-1} p(x_i) = \sum_{i=1}^{M} p(x_i)$. Observe que o intervalo sobre o qual $prob$ deve ser gerado, vai sendo diminuído à medida que indivíduos vão sendo selecionados. Por exemplo, $\sum_{i=1}^{N} p(x_i)$ é sempre igual a 1 na seleção do primeiro indivíduo, mas, como a probabilidade do indivíduo selecionado é feita igual a 0, então $\sum_{i=1}^{N} p(x_i)$ torna-se cada vez mais próximo de zero até que, quando não há mais o que ser selecionado, $\sum_{i=1}^{N} p(x_i)$ torna-se igual a zero. O N-ésimo elemento do vetor rol fornece o somatório das probabilidades de seleção dos indivíduos.

2.9.6. Cruzamento

O operador cruzamento consiste em agrupar pares distintos de indivíduos, de modo que, a partir da recombinação, dois outros indivíduos (filhos) sejam gerados. Empregou-

se aqui apenas recombinação aleatória de um ponto. Neste processo, escolhe-se um ponto de corte aleatoriamente, e efetua-se a troca dos segmentos correspondentes. Uma vez escolhidos dois indivíduos $P[pai]$ e $P[mãe]$ da populção P, o algoritmo de recombinação opera da seguinte forma :

1. Gera-se aleatoriamente um ponto de corte s entre 1 e L, sendo L o comprimento da seqüência de bits.
2. $mask1 = 2^s - 1$;
3. $mask2 = 2^L - 1 - mask1$;
4. $filho1 = (P[pai] \text{ and } mask1) \text{or} (P[mãe] \text{ and } mask2)$;
5. $filho2 = (P[pai] \text{ and } mask2) \text{or} (P[mãe] \text{ and } mask1)$;

Por exemplo, para $L = 4$ e $s = 1$, seja

$$\begin{aligned} P[pai] &= 11_{10} = (101|1)_2 \\ P[mãe] &= 4_{10} = (010|0)_2 \end{aligned}$$

Então :

$$\begin{aligned} mask1 &= 1_{10} = 0001_2 \\ mask2 &= 14_{10} = 1110_2 \\ filho1 &= (11_{10} \text{ and } 1_{10}) \text{ or } (4_{10} \text{ and } 14_{10}) \\ &= 1_{10} \text{ or } 4_{10} = 5_{10} = (010|1)_2 \\ filho2 &= (11_{10} \text{ and } 14_{10}) \text{ or } (4_{10} \text{ and } 1_{10}) \\ &= 10_{10} \text{ or } 0_{10} = 10_{10} = (101|0)_2 \end{aligned}$$

2.9.7. Mutação

O processo de mutação ocorre após o cruzamento. Escolhemos como probabilidade de mutação de bit $p_{mut} = 0,01$, de modo que, em média, $p_{mut} \times L \times N = 0,01 \times 32 \times 31 = 9,92$ bits são alterados a cada geração. Visto que a mutação é puramente aleatória, o cromossomo resultante poderá ter ou não maior valor de adequabilidade. O algoritmo de mutação é apresentado abaixo :

1. Define-se P como um vetor de tamanho N, cujos elementos (inteiros sem sinal) são os indivíduos da população.
2. Define-se $mask$ como um inteiro sem sinal, cujos bits indicarão quais bits, de um determinado indivíduo, serão trocados,
3. Para $i = 1$ até $i = N$, faça :
   (a) $mask = 0$;
   (b) Para $k = 1$ até $k = L$, faça :
      i. Gera-se um número aleatório, $prob$, entre 0 e 1 ;
      ii. Se $prob < p_{mut}$, faça :
         A. $mask = mask + 2^{k-1}$;
      iii. fim de Se.
   (c) $P[i] = P[i] \operatorname{xor} mask$;
4. fim de Para.

Como exemplo, suponha que $L = 4$ e que $prob$ tenha sido menor que p_{mut} no algoritmo acima, para $k = 1$ e $k = 3$. Suponha também que o i-ésimo indivíduo seja $P[i] = 15_{10} = 1111_2$. Deste modo, obtemos :

$$\begin{aligned} mask &= 2^0 + 2^2 = 5_{10} = 0101_2 \\ P[i] &= 15_{10} \operatorname{xor} 5_{10} = 10_{10} = 1010_2 \end{aligned}$$

Em outras palavras, trocou-se os bits 1 e 3 de $P[i]$.

2.9.8. Resultados

A Figura 2.14 mostra os valores de $f(\cdot)$, dados pela Equação 2.7, para o melhor indivíduo da população. Como pode ser visto nesta figura, o AG implementado teve sucesso em obter o mínimo global de $f(\cdot)$ em poucas gerações. Terminado o algoritmo, decodifica-se este melhor indivíduo de volta ao espaço de busca original. No caso desta simulação, o melhor indivíduo obtido foi

$$x = -3,8559 \times 10^{-5}.$$

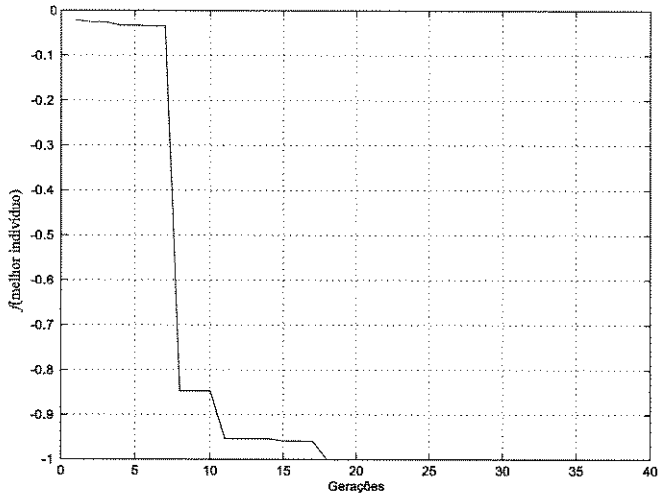

Figura 2.14: Valores de $f(\cdot)$, ao longo das gerações, para o melhor indivíduo da população.

No total, as 38 gerações corresponderam a $38 \times 31 = 1178$ indivíduos averiguados dos 2^{32} possíveis indivíduos, representando $2,74 \times 10^{-5}\%$ do espaço de busca. Na verdade o algoritmo convergiu em 18 gerações, mas teve de esperar por 20 gerações devido ao critério de parada adotado.

Capítulo 3

Interpolação Não Linear de Imagens

3.1. Introdução

Seja para armazenamento ou transmissão, o processo de dizimação é um modo efetivo de reduzir a informação espacial em uma imagem digital. O processo inverso, chamado de interpolação, serve para recuperar a imagem original a partir da sua versão dizimada ou, siplesmente, para ampliar uma imagem digital. Quanto às aplicações, a interpolação é necessária em codificadores baseados na análise de multiresolução, ampliação de imagem/vídeo para visualização em computadores ou TV de alta definição, scanners ópticos, impressoras de alta resolução, conversão de formato de imagem/vídeo, etc.. Para gerar mapas precisos da superfície da Terra, cartografistas devem expandir pequenas regiões de imagens de satélite. Em imagens médicas, fatias de tomografia computadorizada e raios-X podem necessitar ampliação na busca de anomalias. Fotografias de reconhecimento militar devem ser expandiadas, de forma acurada, para mostrar detalhes escondidos de fábricas de armamentos e pistas de aterrissagem.

Com métodos usuais de interpolação, tais detalhes da imagem original tenderiam a parecer menos importantes. Isto diminui a utilidade da imagem expandida no que se refere à precisão dos detalhes. Algumas destas técnicas incluem replicação de pixel, interpolação bilinear e métodos baseados em spline. Entretanto, tais técnicas freqüentemente têm um desempenho muito pobre quanto à qualidade subjetiva, visto que tendem a suavizar a imagem nas regiões descontínuas ou criar artefatos.

O teorema da amostragem especifica qual é a máxima freqüência que pode ser retida no

Figura 3.1: Similaridade entre duas escalas de uma mesma imagem.

sinal amostrado, ou reamostrado digitalmente, sem que ocorra superposição no seu espectro de freqüência. Tal limite é chamado de freqüência de Nyquist e corresponde à metade da freqüência de amostragem utilizada. Isto impõe um limite fundamental no conteúdo de freqüência de um sinal digital disponível para interpolação, visto que o mesmo é limitado em banda. Em outras palavras, dizimação e interpolação não são exatamente processos inversos, porque para dizimar (ou reamostrar com uma freqüência mais baixa) é necessário jogar fora informações de alta freqüência (detalhes) que não podem ser recuperadas na interpolação. Isto significa que é possível aumentar a taxa de um sinal digital mas não a sua resolução. Entretanto, pode-se contornar este problema criando-se um modelo de reconstrução do sinal original a partir do sinal dizimado, ou seja, um modelo que diga como prever as mais altas freqüências, que foram perdidas, a partir das baixas freqüências.

Neste trabalho, propomos que o modelo de reconstrução seja aprendido a partir de um filtro FIR bidimensional não-linear, inspirado nas redes neurais. Neste modelo, se X é a imagem a ser interpolada e $D(X)$ a sua versão dizimada, o preditor (interpolador) será treinado a partir do par $[D(X), X]$ e, em seguida, utilizado para interpolar a imagem X. Em outras palavras, o interpolador é ensinado a mapear $D(X)$ em X e utilizado posteriormente para expandir X em uma versão ampliada $A(X)$. O que sugere que tal estratégia pode dar certo, é a similaridade existente entre as diversas escalas da imagem, conforme é mostrado na Figura 3.1. De fato, comprovamos, através de simulações, a validade de tal estratégia. Neste caso o filtro não-linear é otimizado com algoritmos genéticos.

Em uma segunda etapa, implementamos um outro interpolador não-linear, cujo modelo de reconstrução explora as características de bordas da imagem. As bordas são uma característica relevante da imagem devido a sua importância para os estágios iniciais

do processamento visual humano. Além disso, verifica-se, ao dizimar várias vezes uma imagem, que as bordas são mantidas, mesmo em uma resolução mais baixa. Podemos afirmar então que a estrutura de construção de uma imagem, a partir de sua escala inferior, está intimamente relacionada com as bordas. Neste caso, o modelo de interpolação é adaptativo localmente às bordas da imagem e não necessita de uma etapa de otimização como no método descrito anteriormente.

3.2. Interpolação Linear de Imagens

As técnicas lineares de interpolação de imagens podem ser classificadas em dois tipos : interpolação no domínio da freqüência e interpolação no domínio do espaço. A interpolação no domínio do espaço pode ainda ser dividida em interpolação por filtragem e interpolação de nível de cinza. As técnicas mais utilizadas são aquelas que manipulam os pixels diretamente no domínio do espaço. Também são estas últimas que dão os melhores resultados. O objetivo desta seção é apresentar ambas as técnicas de interpolação linear para tornar evidente as vantagens dos métodos não-lineares sobre as técnicas lineares. Com isto em mente, apresentamos inicialmente o teorema da amostragem e, em seguida, a conceituação dos dois métodos de interpolação, bem como as técnicas de interpolação de imagens mais utilizadas.

3.2.1. O Teorema da amostragem e sua aplicação a imagens

Seguiremos aqui, a mesma estrutura de exposição utilizada por Ahmed [8]. Primeiro desenvolvemos o teorema para o caso unidimensional (1D). Depois, estendemos o teorema para o caso bidimensional (2D).

Teorema 3.1 *Se um sinal $x_a(t)$ não contém nenhuma freqüência mais alta do que Ω_c, ele pode ser completamente determinado tomando-se suas amostras em intervalos igualmente espaçados de T. O intervalo de amostragem T é dado por*

$$T \leq 2\pi \frac{1}{2\Omega_c},$$

sendo T dado em segundos e Ω_c em radianos por segundo.

Apresentamos aqui algumas passagens da prova deste Teorema, a fim de fixarmos a notação e enfatizar alguns aspectos importantes para a interpolação. Com isto em mente, considere a representação de um sinal analógico $x_a(t)$ por sua transformada de Fourier

$$X_a(j\Omega) = \int_{-\infty}^{\infty} x_a(t)\, e^{-j\Omega t} dt \tag{3.1a}$$

$$x_a(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X_a(j\Omega)\, e^{j\Omega t} d\Omega. \tag{3.1b}$$

Se $x[n]$ representa uma seqüência discreta obtida por amostragem de $x_a(t)$ em intervalos igualmente espaçados de T, então podemos utilizar a Equação 3.1b para escrever :

$$x[n] = x_a(nT) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X_a(j\Omega)\, e^{j\Omega nT} d\Omega \tag{3.2}$$

A partir da transformada de Fourier, obtemos

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X\left(e^{j\omega}\right) e^{j\omega n}, \tag{3.3}$$

sendo $X(e^{j\omega})$ a transformada de Fourier discreta de $x[n]$. O próximo passo é então relacionar as transformadas de Fourier do sinal analógico e a do sinal discreto. Isto é feito a partir da comparação entre as Equações 3.2 e 3.3. Mas primeiro, é necessário escrever a Equação 3.2 na forma da Equação 3.3, mudando seus limites de integração a partir de manipulações matemáticas. Com isto em mente, escrevemos a integral no intervalo $[-\infty, \infty]$ como somas de integrais sobre intervalos $2\pi/T$:

$$x[n] = \frac{1}{2\pi} \sum_{r=-\infty}^{\infty} \int_{(2r-1)\pi/T}^{(2r+1)\pi/T} X_a(j\Omega)\, e^{j\Omega nT} d\Omega$$

Esta expressão é equivalente a

$$x[n] = \frac{1}{2\pi} \int_{-\pi/T}^{\pi/T} \left(\frac{1}{T} \sum_{r=-\infty}^{\infty} X_a\left(j\frac{\omega}{T} + j\frac{2\pi r}{T} \right) \right) e^{j\omega n} d\omega. \tag{3.4}$$

Comparando 3.4 com 3.3, obtemos :

$$X\left(e^{j\omega}\right) = \frac{1}{T} \sum_{r=-\infty}^{\infty} X_a\left(\frac{j(\omega + 2\pi r)}{T} \right) \tag{3.5}$$

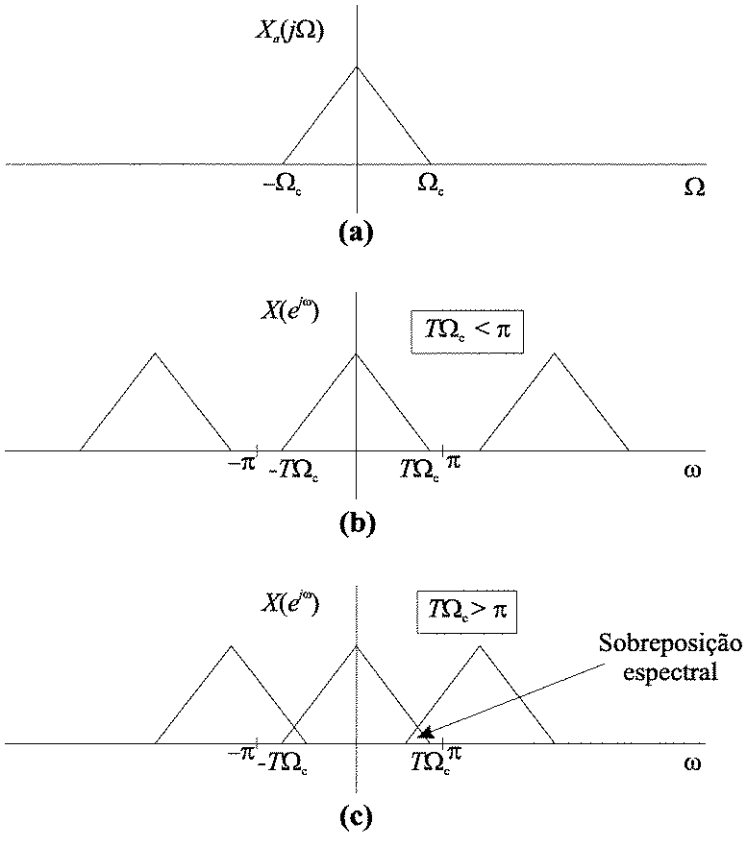

Figura 3.2: Espectro de freqüência de um sinal analógico e de sua versão amostrada.

A Equação 3.5 é interpretada graficamente na Figura 3.2. A partir desta figura, concluímos que

$$T \leq 2\pi \frac{1}{2\Omega_c}.$$

Teorema 3.2 *O teorema da amostragem bidimensional estabelece que uma função $f_a(x, y)$, limitada em freqüência, pode ser completamente recuperada de suas amostras se as linhas e colunas forem amostradas independentemente com intervalos entre as amostras dados por*

$$T_H \leq 2\pi \frac{1}{2\Omega_H}, \quad T_V \leq 2\pi \frac{1}{2\Omega_V} \quad , \tag{3.6}$$

sendo Ω_V e Ω_H as máximas freqüências em radiano por segundo nas direções horizontal e vertical, respectivamente.

A demonstração deste teorema é idêntica à demonstração do teorema anterior e não é feita aqui.

O teorema da amostragem impõe uma restrição importante : o sinal deve ter banda finita para ser reconstruído perfeitamente. Entretanto, em imagens isto em geral não ocorre, apesar do seu espectro cair rapidamente. Portanto, é necessário utilizar um filtro passa-baixas antes de amostrar um sinal para limitá-lo em freqüência. Se isto não for feito, qualquer que seja o valor escolhido para a freqüência de amostragem, irá ocorrer sobreposição espectral ("aliasing"). O mesmo ocorre quando se dizima uma imagem, quando é sempre necessário um filtro passa-baixas ("anti-aliasing") antes de diminuir a resolução da imagem. Conseqüentemente, sempre que se dizima uma imagem joga-se fora informações de alta freqüência que não mais poderão ser recuperadas.

3.2.2. Interpolação no Domínio da Freqüência

Esta técnica é uma aplicação direta do teorema da amostragem. Dada uma imagem $N \times N$, primeiro obtém-se sua transformada DFT. Em seguida, adiciona-se zeros como informação de mais alta freqüência. Isto é mostrado na Figura 3.3. Finalmente, calcula-se a DFT inversa para se obter a imagem interpolada. Observe que não se adicionou nenhuma informação espectral, porque ela não é conhecida. Mais ainda, esta infomação espectral é de difícil estimação visto que a DFT não apresenta nenhuma localização espacial.

Por este motivo, pode-se afirmar que as transformadas wavelets são mais adequadas do que a DFT no que se refere à estimação da informação perdida. Para ser mais claro, devido à localização e ao suporte compacto das funções de base, as sub-bandas da transformada wavelet correspondem a detalhes, em um dado nível de resolução, da imagem a ser interpolada. Um dos aspectos mais importantes de uma imagem, no que se refere ao sistema visual humano, são as bordas. É interessante notar que as bordas de uma imagem se manifestam em diferentes níveis de resolução, ou sub-bandas da transformada wavelet, de modo que elas também devem se manifestar nas sub-bandas de mais alta freqüência que se quer estimar. O resultado é que se pode então dobrar o tamanho da imagem adicionando-se informações que foram estimadas a partir da informação existente. O difícil, neste caso, é encontrar um modelo que realize esta estimação, visto que as sub-bandas são ortogonais entre si. Por outro lado, a similaridade entre as sub-bandas tem sido usada

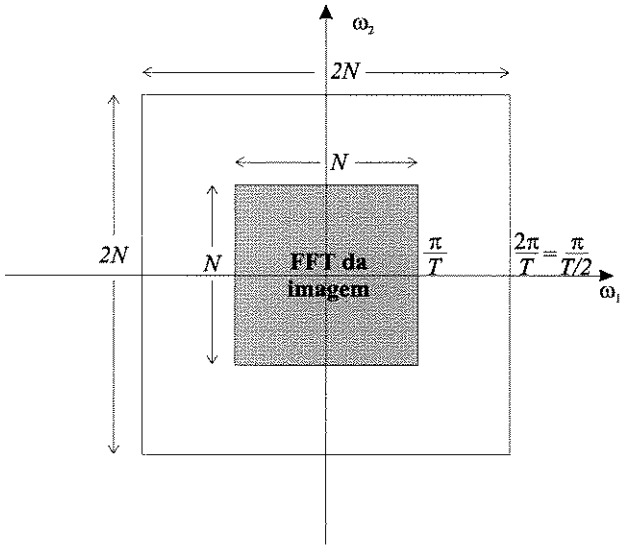

Figura 3.3: Adição de zeros à DFT para obter o espectro da imagem expandida. Neste exemplo, adicionou-se zeros ao espectro com o objetivo de dobrar o tamanho da imagem.

com sucesso na compressão de imagens Shapiro [22] e Said [23].

A desvantagem da técnica que utiliza a DFT é a distorção da informação original. A DFT trata o sinal no domínio espacial como sendo periódico, provocando o surgimento de descontinuidades que podem se manifestar ao longo de todo o espectro do sinal. Uma solução para este problema é a utilização da DCT (Transformada de Cossenos Discreta) no lugar da DFT. A DCT também vê o sinal como sendo periódico, entretanto ela impõe simetria par ao sinal no domínio do espaço, e isto substitui as descontinuidades por pontos de inflexão, onde a derivada não existe. Apesar disto, este efeito é menos danoso do que as descontinuidades.

3.2.3. Interpolação por Filtragem

Esta técnica é bastante simples. Inicialmente, insere-se amostras na imagem a ser interpolada entre as amostras conhecidas. O usual é a inserção de zeros, visto que isto implica em uma compressão do espectro da imagem pelo mesmo fator de ampliação que é desejado para a mesma. Devido à periodicidade do espectro, é necessário utilizar uma filtragem passa-baixas para evitar que a mesma informação nas baixas freqüências se repita nas mais altas freqüências (em torno de π). Na Figura 3.4, ilustramos, a partir

de um exemplo unidimensional, o efeito destas operações (realizadas no espaço) sobre o espectro do sinal. Para mais detalhes, consulte Oppenheim [20].

Na aplicação desta técnica a imagens, podemos supor separabilidade entre linhas e colunas, de modo que as técnicas unidimensionais podem ser aplicadas diretamente a imagens. Entretanto, esta estratégia não é interessante porque o filtro resultante bidimensional tem simetria quadrada, ou seja, apenas as direções vertical e horizontal são tratadas de forma idêntica pelo filtro passa-baixas. Obtém-se uma melhora qualitativa considerável se o filtro for projetado diretamente no plano bidimensional com simetria circular na freqüência.

Aqui também não se adiciona informação alguma ao espectro do sinal. O objetivo é preservar ao máximo a informação original, melhorando-se o filtro passa-baixas de modo a compensar seus desvios com relação ao passa-baixas ideal.

3.2.4. Interpolação de Nível de Cinza

Suponha uma imagem f com coordenadas (x, y) e uma imagem g com coordenadas $(\hat{x}, \widehat{y})$ relacionadas entre si pela seguinte transformação geométrica

$$\hat{x} = 2x \tag{3.7a}$$

$$\widehat{y} = 2y \tag{3.7b}$$

Esta transformação gera uma imagem $g\,(\hat{x}, \widehat{y})$ cujo tamanho é o dobro do tamanho da imagem $f\,(x, y)$. Por causa da transformação realizada, apenas os valores de pixel da imagem g que envolvem coordenadas ímpares estão indefinidos. Isto ocorre porque o mapeamento inverso das coordenadas pares, na imagem g, leva a valores de coordenada fracionários na imagem f e, devido à imagem f ser digital, somente são conhecidos os valores de pixel em coordenadas inteiras. Inferir quais os valores de nível de cinza daquelas posições (ímpares na imagem g e fracionárias na imagem f), com base somente nos valores de pixel das coordenadas inteiras de f, torna-se então necessário. A técnica utilizada para alcançar este objetivo é chamada de interpolação de nível de cinza.

O esquema mais simples para interpolação de nível de cinza é baseado na abordagem do vizinho mais próximo. Este método, que também é chamado de interpolação de ordem zero ou replicação de pixel, é ilustrado na Figura 3.5. Primeiro, realiza-se o mapeamento

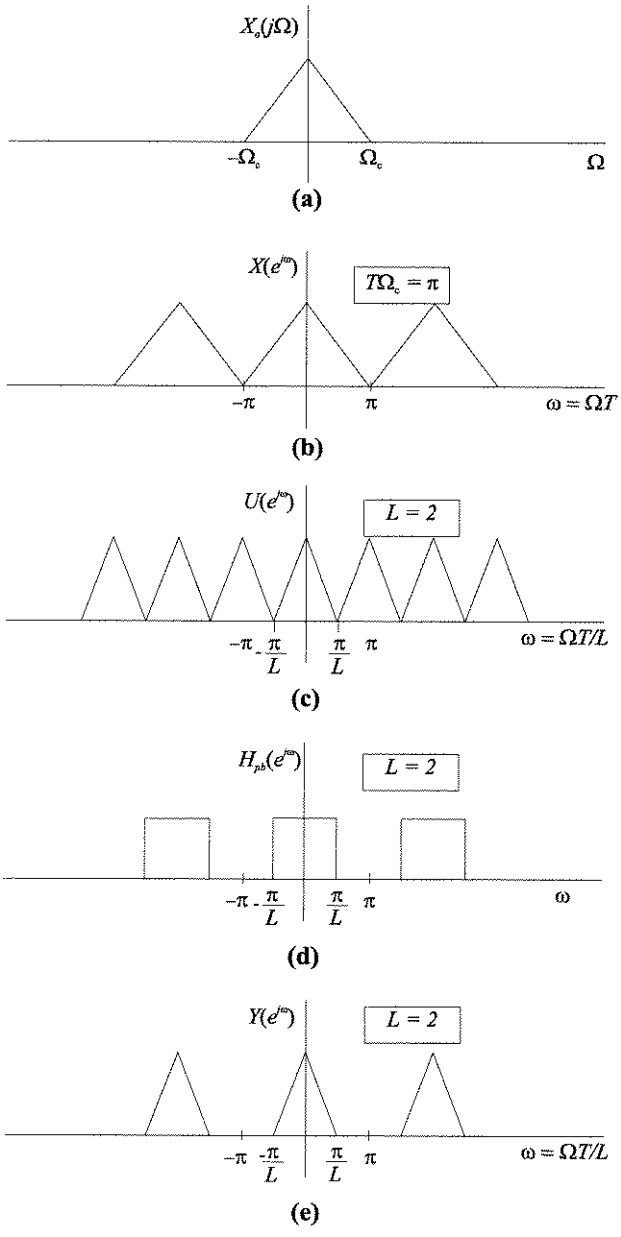

Figura 3.4: Efeito da interpolação linear unidimensional, por filtragem no domínio espacial, sobre o conteúdo espectral do sinal. Em (a), $X_a(j\Omega)$ é o espectro do sinal analógico. Em (b), $X(e^{j\omega})$ é o espectro do sinal discreto, obtido a partir do analógico com a freqüência de amostragem mínima possível. Em (c), $U(e^{j\omega})$ é o espectro do sinal discreto em (b), $x[n]$, após ter tido suas amostras intercaladas com $L-1$ zeros. Observe que há uma compressão do espectro de modo que se faz necessário o uso do filtro passa-baixas em (d). Uma vez filtrado $u[n]$ com $h[n]$, obtemos o sinal $y[n]$ com uma freqüência de amostragem aumentada L vezes, ou seja, obtemos o sinal ampliado L vezes.

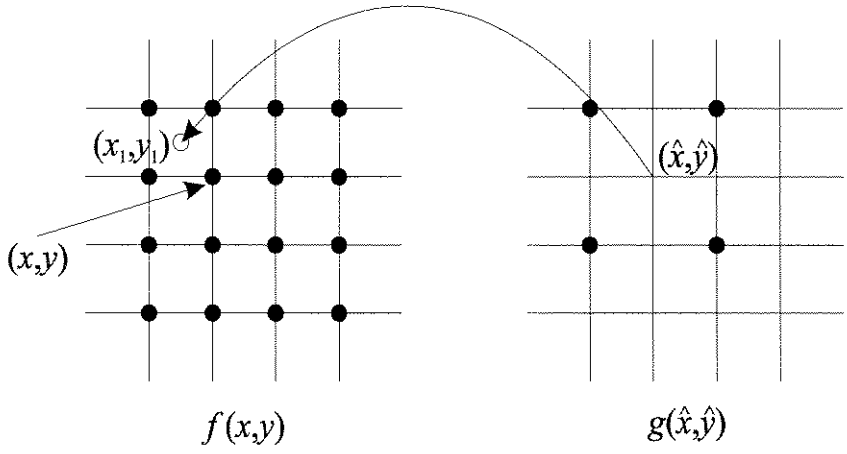

Figura 3.5: Interpolação de nível de cinza baseada no conceito de vizinho mais próximo. As posições $(\hat{x}, \hat{y})$ da imagem g em que os valores de pixel estão indefinidos, correspondem a posições da imagem f cujas coordenadas (x_1, y_1) são fracionárias. Arredondando (x_1, y_1), obtém-se (x, y) e $g(\hat{x}, \hat{y}) = f(x, y)$.

inverso das coordenadas ímpares $(\hat{x}, \widehat{y})$ nas coordenadas fracionárias (x_1, y_1), a partir das Equações (3.7a) e (3.7b). Segundo, arredonda-se as coordenadas fracionárias para as coordenadas inteiras (x, y) em f . Finalmente, faz-se $g(\hat{x}, \widehat{y}) = f(x, y)$.

Embora a interpolação de ordem zero seja de fácil implementação, este método tem a desvantagem de produzir artefatos indesejáveis, tais como distorções na continuidade de bordas em imagens de fina resolução. Resultados mais suaves podem ser obtidos por meio de técnicas mais sofisticadas, tais como *interpolação bicúbica*, que adapta uma superfície do tipo $\sin(x)/x$ sobre um número muito grande de vizinhos (digamos, 16) com o objetivo de obter uma estimativa suave do nível de cinza em qualquer ponto desejado. Esta é uma técnica que reduz bastante o efeito de bloco, mas apresenta custo alto do ponto de vista computacional e torna a imagem interpolada borrada. Um compromisso razoável é utilizar a interpolação bilinear, que é computacionalmente mais simples, borra menos a imagem, e gera efeitos de bloco na imagem final menores do que a replicação de "pixels".

A abordagem dada pela interpolação bilinear utiliza os níveis de cinza dos quatro vizinhos mais próximos. A idéia é mapear as coordenadas $(\hat{x}, \widehat{y})$ da imagem g, cujos valores de pixel estão indefinidos, pois resultam em coordenadas fracionárias $(x_{\text{frac}}, y_{\text{frac}})$

na imagem f e, em seguida, tomar os quatro vizinhos mais próximos com coordenadas inteiras (x_1, y_1), (x_2, y_2), (x_3, y_3) e (x_4, y_4), para adaptar uma superfície quadrática de intensidade de pixel passando por estes quatro pontos, na forma

$$\begin{aligned} f(x_1, y_1) &= ax_1 + by_1 + cx_1y_1 + d \\ f(x_2, y_2) &= ax_2 + by_2 + cx_2y_2 + d \\ f(x_3, y_3) &= ax_3 + by_3 + cx_3y_3 + d \\ f(x_4, y_4) &= ax_4 + by_4 + cx_4y_4 + d, \end{aligned}$$

onde os quatro coeficientes da superfície são facilmente determinados da solução deste sistema com quatro equações e quatro incógnitas. Uma vez solucionado este sistema para os coeficientes a, b, c e d, calcula-se o valor do pixel na coordenada fracionária

$$f(x_{\text{frac}}, y_{\text{frac}}) = ax_{\text{frac}} + by_{\text{frac}} + cx_{\text{frac}}y_{\text{frac}} + d$$

e torna-se

$$g(\hat{x}, \hat{y}) = f(x_{\text{frac}}, y_{\text{frac}}).$$

3.3. Interpolação Não-Linear de Imagens

Ao invés de se utilizar técnicas lineares ou bilineares, pode-se interpolar uma imagem com filtros não-lineares, que normalmente dão melhores resultados subjetivos. Desta forma, esta seção apresenta o estado-da-arte em interpolação não-linear de imagens.

Um esquema de interpolação não-linear para enriquecer a resolução de imagens paradas é apresentado por Jensen [10]. Este algoritmo é baseado em um modelo de fonte que enfatiza a integridade visual das bordas detectadas, e incorpora um operador adequado para as bordas. Uma pequena vizinhança sobre cada pixel na imagem de baixa resolução é primeiro mapeado, tendo as bordas como referência, para um espaço contínuo. Uma aproximação quantizada em dois níveis serve como um modelo local, no qual uma grade de amostragem de mais alta resolução pode então ser superposta. Visto que esta janela irá correr pixel por pixel através da imagem, é inevitável que localmente haja sobreposição de janelas. Neste caso, deve-se tirar a média dos valores das janelas concorrentes para suavizar os erros. O resultado é uma imagem de melhor resolução e com bordas notadamente realçadas.

No sistema de interpolação proposto por Hong [9], uma parte dos coeficientes da DCT é utilizada para extrair informações de bordas em uma determinada janela da imagem. Cinco diferentes tipos de bordas são identificados através do uso apropriado dos coeficientes DCT. Cada tipo de borda determina qual técnica de interpolação deve ser aplicada para a região local correspondente. Depois que o tipo de borda é determinado, a interpolação bilinear e a replicação de pixel são combinadas e desenvolvidas simultaneamente. Finalmente, cinco diferentes filtros passa-baixas Gaussianos reduzem adaptativamente as descontinuidades que resultam desta técnica de interpolação.

Schultz [14] por sua vez, coloca o problema dentro de uma abordagem estatística, utilizando estimação de máximo a posteriori. A idéia básica é que interpolação de imagens é um problema mal-condicionado, visto que existe um conjunto virtualmente infinito de imagens que podem ser expandidas a partir dos dados originais. Dois problemas são tratados em separado : primeiro, para a interpolação de uma imagem livre de ruídos; depois, para uma imagem corrompida por ruído gaussiano. Problemas de minimização de funcional são definidos para ambos os casos, os quais são então resolvidos para formar campos de parâmetros estimados que correspondem às imagens expandidas. Os funcionais propostos são convexos, de modo que métodos eficientes de otimização basados em gradiente podem ser empregados na minimização.

Wong [12] desenvolveu um filtro espaço-escala não-linear baseado na teoria da informação e na mecânica estatística. Para cada pixel na imagem, uma vizinhança ponderada é utilizada na clusterização. O centro do cluster (ou janela) determina a saída do filtro. O filtro é governado por um parâmetro de escala simples que dita a extensão da vizinhança de cada pixel a ser utilizada na clusterização. Isto, juntamente com a característica local do sinal, provê mecanismos para remoção de ruído impulsivo, atenuação de ruídos não impulsivos e preservação de bordas. Wong [12] utilizou tal filtro não-linear para interpolação dentro de um dos mais populares sistemas de multiresolução (Pirâmide Laplaciana), e demonstrou sua aplicação na geração de um sistema de multiresolução não-linear.

Resumindo, o estado-da-arte em interpolação não-linear de imagens, consiste em dar uma abordagem baseada em um modelo estrutural, onde a informação de alta freqüência pode ser predita com base no conteúdo de freqüência da imagem a ser interpolada, sendo que o foco da maioria dos modelos utilizados são as bordas da imagem. Outros algoritmos de interpolação não-linear foram desenvolvidos por Wang [17], Vleeschauwer [18], Martinez

[19], Herodotou [11], Lakshmanan [15], Zeng [16] e Onural [13].

3.4. Redes Neurais

O objetivo desta seção é apresentar uma visão geral sobre as redes neurais e suas possíveis formas de implementação, visando sua utilização na interpolação de imagens.

Uma rede neural (RN) é um modelo projetado para simular o modo como o cérebro realiza uma determinada tarefa ou função de interesse. Uma RN pode ser implementada utilizando-se componentes eletrônicos ou simulada via software em um computador digital. Para alcançar boa performance, a rede emprega uma interconexão maciça de unidades básicas de processamento conhecidas como neurônios. Deste modo, pode-se definir uma rede neural como sendo um processador composto de células básicas (neurônios), organizadas em uma estrutura maciçamente paralela, cuja função é o armazenamento do conhecimento pela experiência. O conhecimento é adquirido pela rede através de um processo de aprendizagem (treinamento) e é armazenado na rede pelas interconexões entre os neurônios. Estas interconexões são conhecidas como pesos sinápticos.

O procedimento utilizado no treinamento da rede é chamado de *algoritmo de aprendizagem*. Sua função é modificar os pesos sinápticos para atender um objetivo de projeto. Usualmente o treinamento é supervisionado, ou seja, para cada sinal aplicado à entrada, é imposta uma saída desejada. Desta forma, os pesos sinápticos são corrigidos de modo a minimizar o erro médio entre a saída desejada e a saída obtida. Ao par entrada-saída utilizado na fase de treinamento, dá-se o nome de amostra. Uma técnica de aprendizagem consiste em repetir o treinamento para muitas amostras, até que a rede alcance um estado estacionário, ou seja, não haja mais mudanças significativas nos pesos sinápticos, ou o erro médio entre a saída desejada e a saída obtida atinja seu valor mínimo. Deste modo, a rede aprende com as amostras a construir um mapeamento entre a entrada e a saída da rede.

A unidade básica de uma rede neural é o *neurônio*, que é um dispositivo não linear. Conseqüentemente, uma rede neural constituída pela interconexão de vários neurônios, também é não-linear. Não-linearidade é uma propriedade importante, principalmente se o mecanismo físico responsável pela geração do sinal de entrada é inerentemente não-linear.

3.4.1. Modelo de Neurônios

Um neurônio é a unidade básica de processamento da informação, que é fundamental na operação da rede neural. A Figura 3.6(a) mostra o modelo de um neurônio típico. São três os elementos básicos de um neurônio : um conjunto de sinápses, um somador e uma função de ativação. Observe que o modelo da Figura 3.6(a) também inclui um limiar que é externamente aplicado à função de ativação. Na Figura 3.6(b), tal limiar é interpretado como um peso sináptico. Em termos matemáticos, pode-se descrever um neurônio a partir do seguinte par de equações

$$v_k = \sum_{j=1}^{p} w_{kj} x_j \tag{3.8}$$

e

$$y_k = \varphi\left(v_k - \theta_k\right), \tag{3.9}$$

onde $x_1, x_2, \ldots, x_p$ são as entradas ; $w_{k1}, w_{k2}, \ldots, w_{kp}$ são os pesos sinápticos do neurônio k ; v_k é a combinação linear das entradas ; θ_k é o limiar ; $\varphi(\cdot)$ é a função de ativação e y_k é o sinal de saída do neurônio. O uso do limiar tem o efeito de aplicar uma transformação afim à saída v_k do combinador linear no modelo da Figura 3.6, como mostrado por $u_k = v_k - \theta_k$.

A função de ativação, denotada por $\varphi(\cdot)$, define a saída de um neurônio em termos do nível de atividade em sua entrada. Pode-se pensar em diversos tipos de função de ativação, sendo que a mais comum é a sigmóide. Ela é definida como uma função estritamente crescente, que exibe suavidade (é derivável em todos os pontos) e propriedades assintóticas. Esta última propriedade é importante para que o nível da saída do neurônio esteja sempre dentro de um intervalo finito. Um exemplo de sigmóide é a função logística, definida por

$$\varphi(v) = \frac{1}{1 + \exp(-av)}, \tag{3.10}$$

onde a é o parâmetro de inclinação da sigmóide.

A função de ativação definida por (3.10) varia de 0 a +1. É muitas vezes necessário que a função de ativação seja simétrica em torno da origem, por exemplo, variando-se de -1 a $+1$. Deste modo, subtraindo 0,5 de (3.10) e multiplicando em seguida por 2, obtém-se a função *signum*, que nada mais é do que a função tangente hiperbólica. Variando

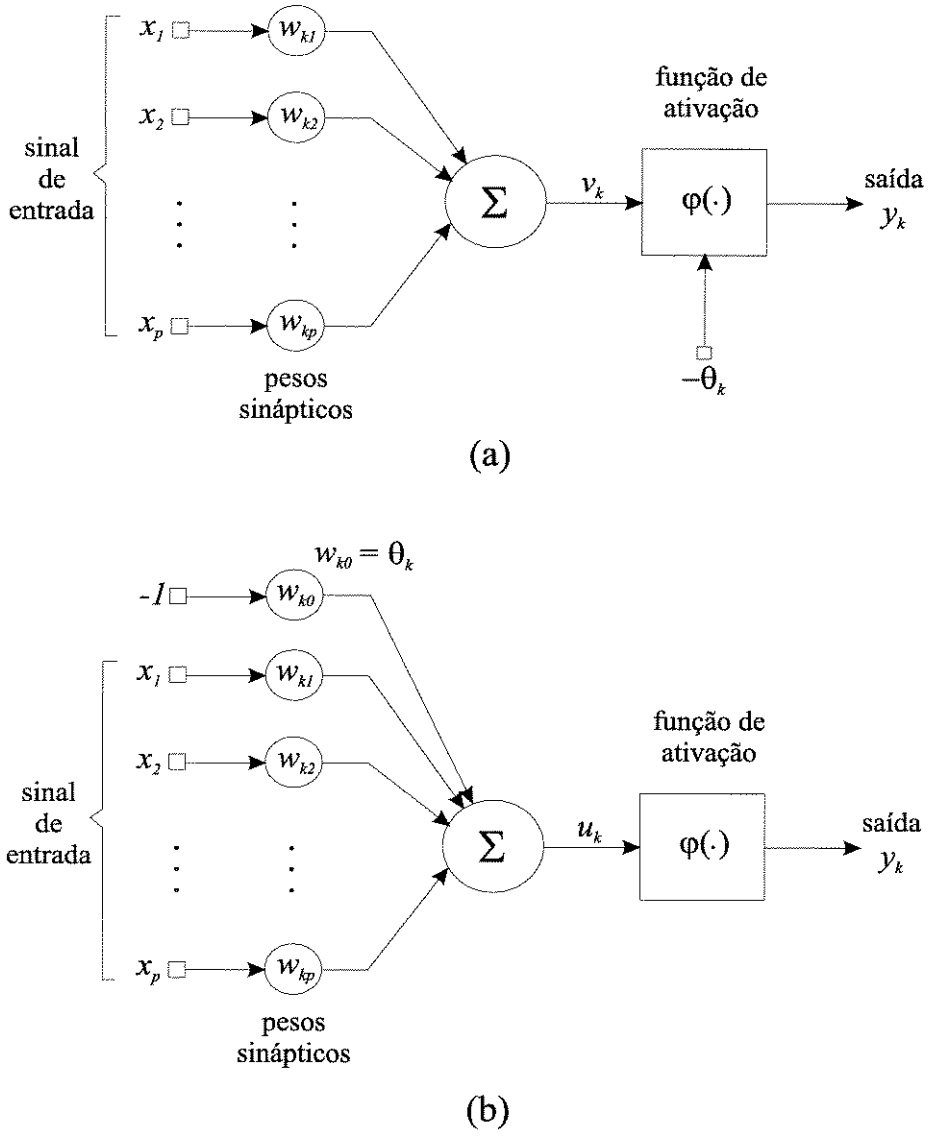

Figura 3.6: Modelo Não-Linear de um Neurônio.

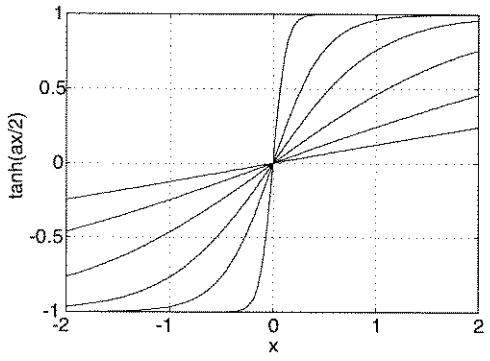

Figura 3.7: Exemplos de sigmóides para diferentes valores da constante a.

o parâmetro a, pode-se obter *signums* com diferentes inclinações, como é mostrado na Figura 3.7.

$$\varphi(v) = \tanh\left(\frac{av}{2}\right) = \frac{1 - \exp(-av)}{1 + \exp(av)}. \tag{3.11}$$

3.4.2. Arquiteturas de Rede

Na seção anterior, falamos sobre a unidade básica de uma rede neural : o neurônio. Também dissemos que uma rede neural consiste basicamente da interconexão maciça destas unidades básicas. Nesta seção, descrevemos as possíveis arquiteturas de redes, isto é às formas como as redes neurais podem ser estruturadas. De modo geral, podemos identificar quatro classes diferentes de arquitetura de redes : redes de camada única, redes multicamadas, redes recorrentes e redes com estrutura reticulada (*lattice*).

Uma rede de camada única tem este nome porque existe apenas uma camada de neurônios. Também existe uma camada de nós de entrada, mas esta camada não é considerada porque nenhuma operação é feita ali. Além disso, esta rede não possui realimentação, ou seja, toda informação aplicada em sua entrada se propaga diretamente até a saída. Esta rede, ilustrada na Figura 3.8, é também chamada de Perceptron.

Em uma rede multicamadas ou perceptron multicamadas, a propagação da informação também é direta, ou seja, sem realimentação. O que a difere da rede de camada única é, como o próprio nome diz, a existência de outras camadas de neurônios entre a camada de nós de entrada e a camada de neurônios de saída. Tais camadas são chamadas de *camadas*

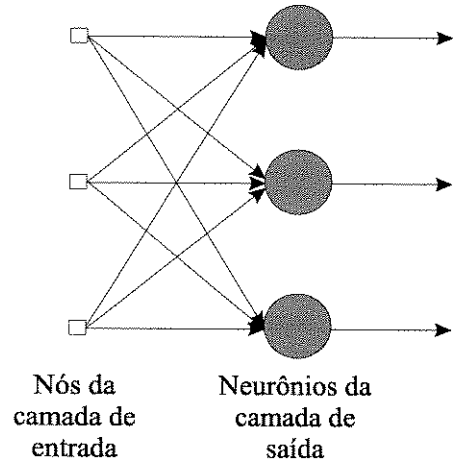

Figura 3.8: Rede de camada única.

escondidas, porque não fazem parte da interface do usuário, como o fazem as camadas de entrada e saída. Pela adição de uma ou mais camadas escondidas, a rede se torna apta a extrair estatísticas de mais alta ordem, adquirindo uma perspectiva global, apesar de sua conectividade local.

Em uma rede de camada única, é através dos nós de entrada que o sinal de ativação é adquirido e propagado para a camada de neurônios de saída. No caso de uma rede multicamadas, o sinal de ativação de uma camada são as saídas da camada anterior, ou seja, uma rede multicamadas nada mais é do que o cascateamento de redes de camada única. Na Figura 3.9 é mostrada a estrutura de uma rede multicamadas para o caso de uma camada escondida. Esta rede, em particular, é dita ser 6-4-2, visto que contém seis nós de entrada, quatro neurônios escondidos e dois neurônios de saída. Esta rede também é dita *totalmente conectada,* visto que cada nó em uma camada da rede é conectado a cada outro nó da camada adjacente posterior. Quando alguma conexão da rede inexiste, diz-se que a rede é *parcialmente conectada.*

As redes recorrentes são muito semelhantes às redes multicamadas, a não ser pela presença de realimentação. Ao se introduzir uma realimentação, também se introduz um atrasador que é um elemento de memória. Na Figura 3.10 é mostrado um exemplo de rede recorrente obtida pela realimentação dos neurônios de saída na rede da Figura 3.9.

Uma rede com estrutura em *lattice* é uma rede cujos neurônios estão distribuídos ao longo de um arranjo multidimensional, cada um dos quais alimentado pelos mesmos nós

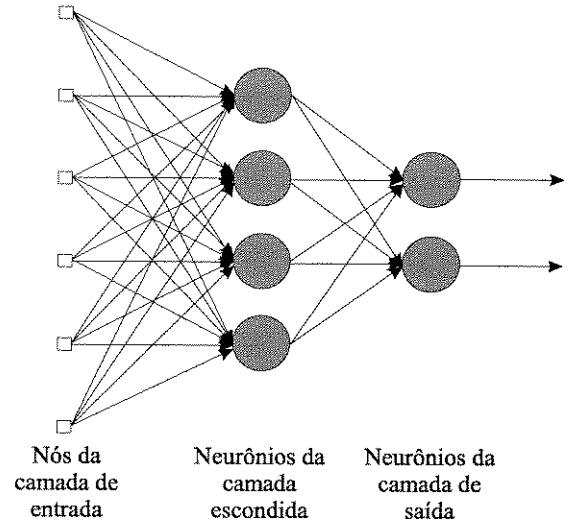

Figura 3.9: Rede neural multicamadas com uma camada escondida.

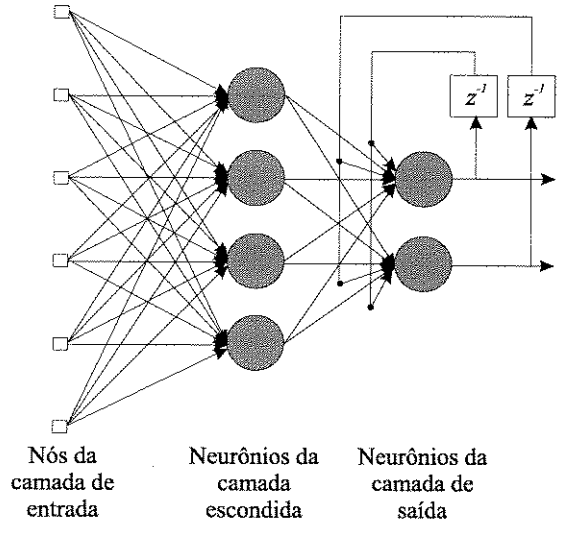

Figura 3.10: Rede recorrente com uma camada escondida.

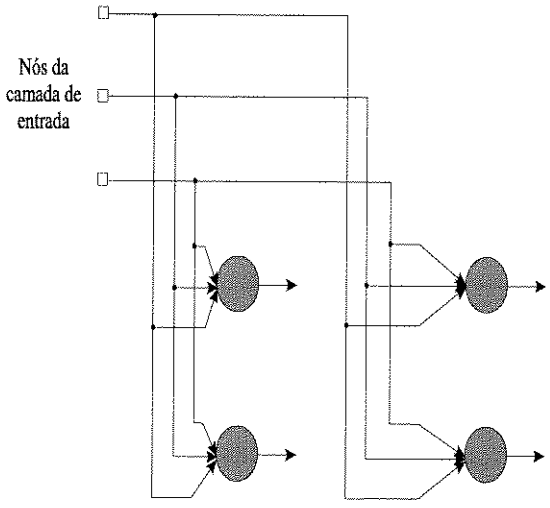

Figura 3.11: Uma rede "lattice" bidimensional 2 por 2.

de entrada. A saída da rede será um arranjo multidimensional constituído pelas saídas dos neurônios da rede. Na Figura 3.11 é apresentado um exemplo de rede com estrutura em *lattice* bidimensional 2×2.

3.5. A Rede Perceptron Multicamadas como Filtro Não-Linear Aplicado a Imagens

Nesta seção é apresentada a estrutura do filtro interpolador não-linear bidimensional e o modo de obtê-la a partir de uma rede neural. Em seguida, adaptamos este filtro não-linear bidimensional para o caso de imagens. Basicamente, impomos simetrias nos coeficientes do filtro de modo que o mesmo tenha fase zero. Estas simetrias nos parâmetros do filtro também são importantes para reduzir a complexidade do algoritmo de treinamento.

Primeiro vamos analisar o caso de um filtro FIR unidimensional e sua associação com uma rede neural. Um filtro FIR com N coeficientes é mostrado na Figura 3.12(a). Analisando tal filtro, observamos que a região demarcada pode ser vista como uma rede neural linear, como na Figura 3.12(b), constituída de N entradas, um neurônio linear (sem a função de ativação) e uma saída. Se substituirmos esta rede linear por uma rede

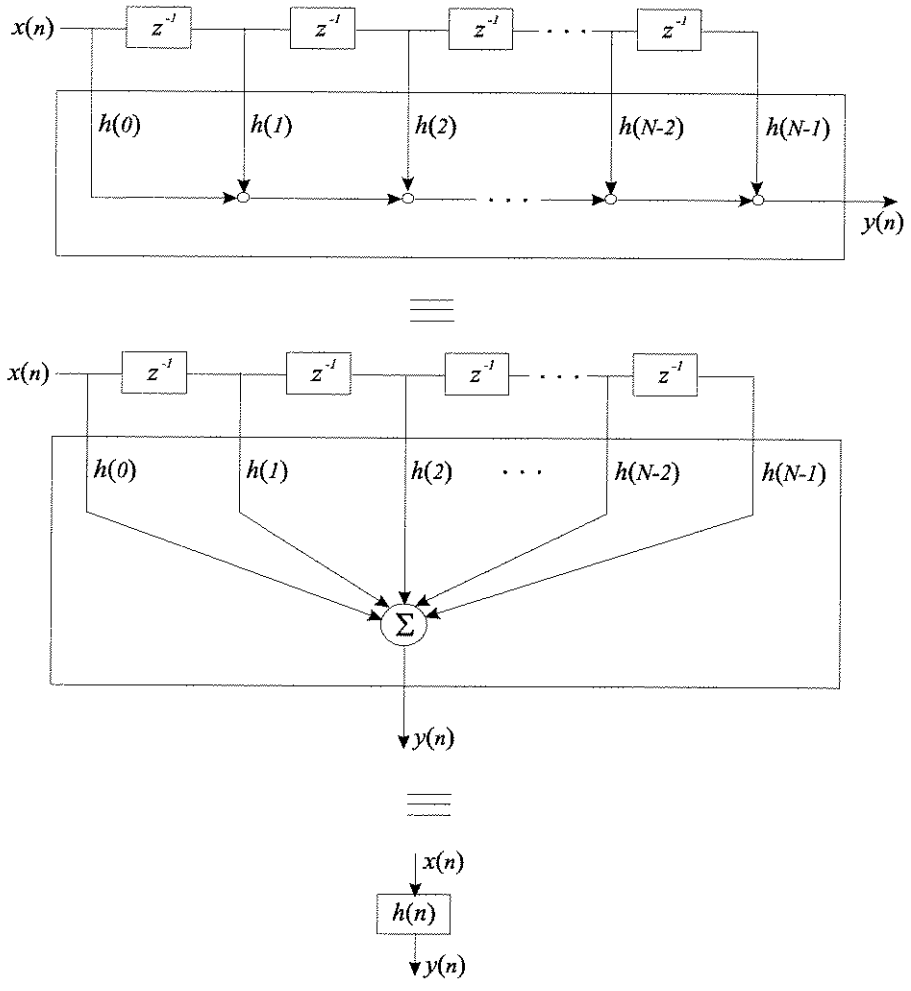

Figura 3.12: Três representações de um mesmo filtro FIR com N coeficientes. A região demarcada pode ser vista como a mais simples rede neural possível, ou seja, uma rede que contém apenas um neurônio linear com N entradas e uma saída.

neural mais complexa, como é mostrado na Figura 3.13, teremos um filtro não-linear unidimensional implementado a partir de uma rede neural. A partir da Figura 3.13, se os atrasos do filtro não-linear forem embutidos na rede neural, sua primeira camada escondida pode ser vista como um banco de filtros lineares em paralelo, sendo que cada filtro seria seguido de uma função de ativação.

Se é desejado que o filtro não-linear tenha fase linear ou zero, é necessário que cada filtro, na primeira camada escondida, tenha fase linear. Em outras palavras, a simetria $h(n) = h(N-1-n)$ deve ser imposta para cada um destes filtros. Um filtro de fase linear é importante em processamento digital de imagens, a fim de evitar que haja distorções de

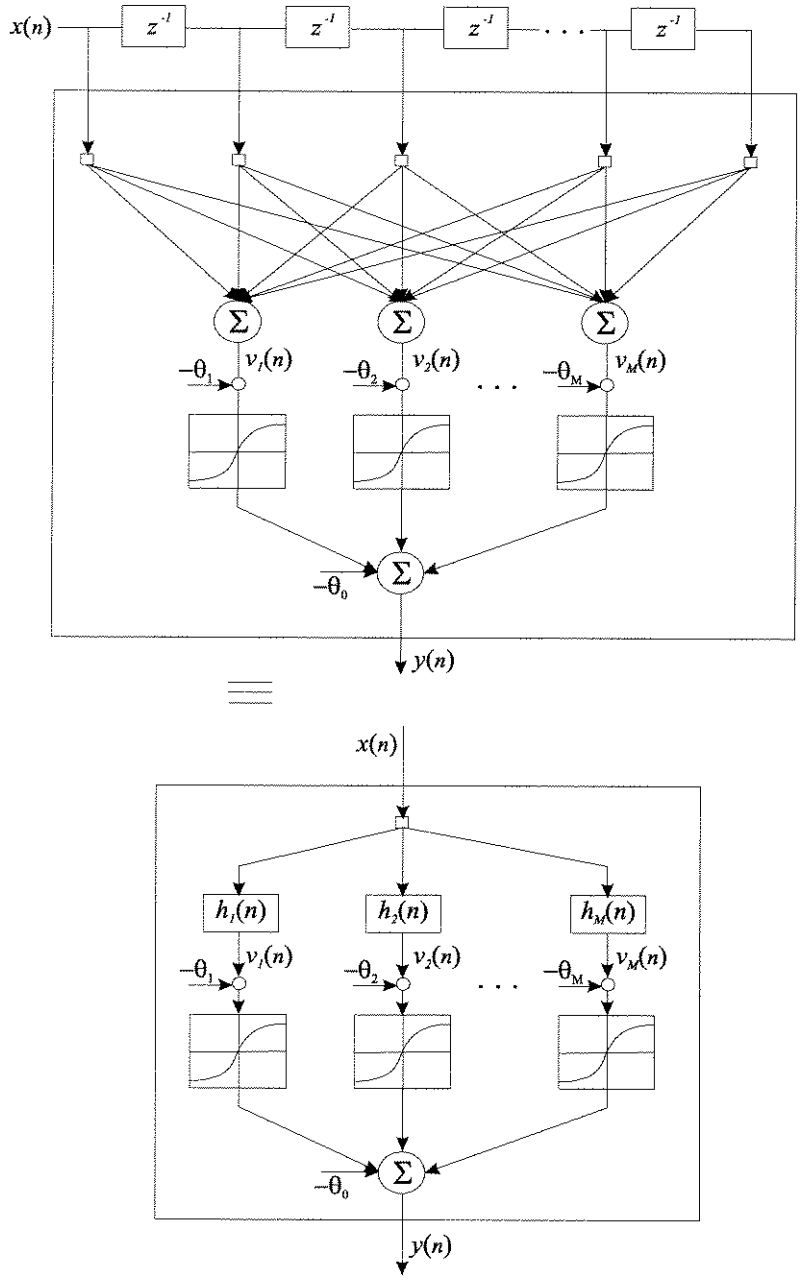

Figura 3.13: Duas representações de um mesmo filtro não linear implementado a partir de uma rede neural multicamadas. Observe que $v_1(n)$, $v_2(n)$, ..., $v_M(n)$, correspondem a saídas de filtro FIR.

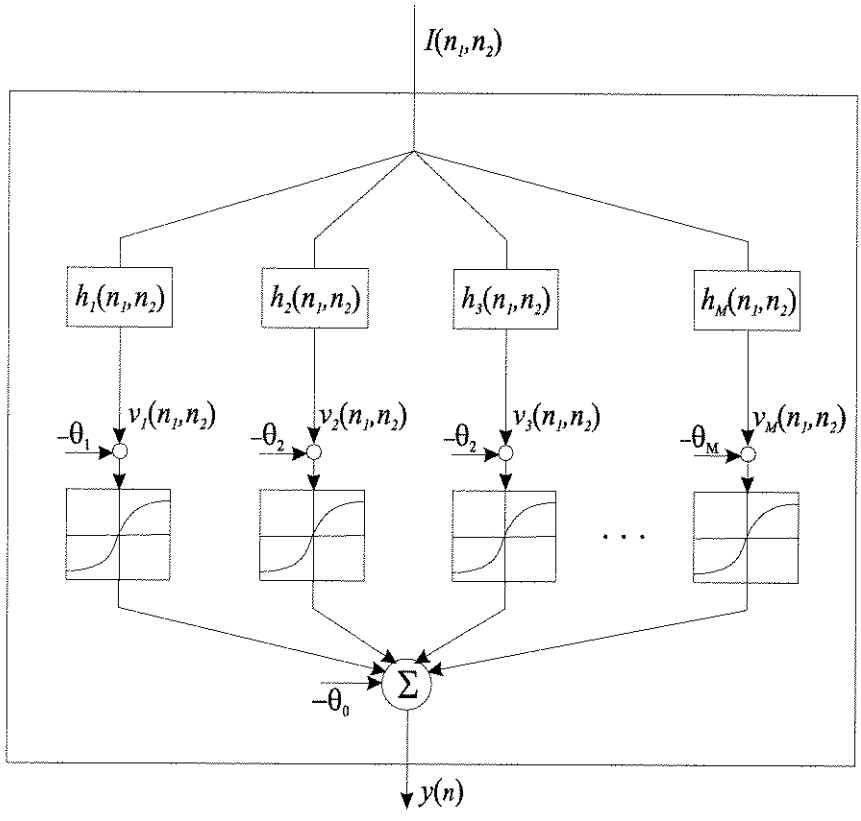

Figura 3.14: Generalização do filtro não-linear unidimensional para o caso bidimensional.

fase, ou seja, evitar que cada pixel tenha um deslocamento espacial diferente com relação aos demais pixels.

Até aqui, verificamos que cada neurônio da primeira camada escondida pode ser visto como um filtro FIR unidimensional, seguido de uma função de ativação. Se pensarmos nestes filtros como tendo dois componentes, generalizamos facilmente o filtro não-linear para o caso bidimensional, como é mostrado na Figura 3.14.

Cada filtro $h_j(n_1, n_2)$ pode ter sua dimensão fixada em $N \times N$, de modo que o número de sinápses, incluindo-se as polarizações, associado a esta rede corresponderá a $M(N^2+1)+1$. O número de sinápses pode ser bastante reduzido se impormos simetria octal aos filtros $h_j(n_1, n_2)$, $j = 1, 2, \cdots, M$, isto é, se

$$h_j(n_1, n_2) = h_j(-n_1, n_2) = h_j(n_1, -n_2) = h_j(-n_1, -n_2) = h_j(n_2, n_1). \tag{3.12}$$

O número de sinápses por filtro $h_j(n_1, n_2)$, com tal simetria, pode ser obtido mais facilmente com o auxilio da Figura 3.15. Assim, observe que o triângulo inferior do qua-

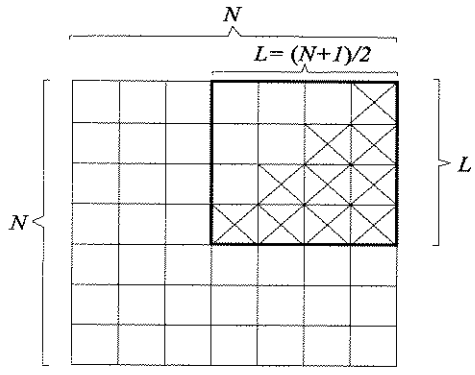

Figura 3.15: Simetria octal : cada um dos coeficientes que não estão hachurados tem um correspondente coeficiente hachurado.

drado $L \times L$ corresponde ao quadrado todo, menos a diagonal, dividido em duas partes. Assim, o número de coeficientes por filtro corresponde a

$$\frac{L^2 - L}{2} + L = \frac{L(L+1)}{2} \tag{3.13}$$

e o número de coeficientes total reduz-se para

$$M\left(\frac{L(L+1)}{2} + 2\right) + 1, \tag{3.14}$$

onde

$$L = \frac{N+1}{2} \tag{3.15}$$

e N será suposto ímpar. O tamanho da janela foi escolhido como sendo ímpar para que a janela tenha um coeficiente central. Ao deslizarmos esta janela sobre a imagem, implementando a convolução, este ponto central estará sobreposto ao pixel da imagem que será alterado. Visto que tal ponto é o eixo de simetria do filtro, então o mesmo terá fase zero.

Além da fase zero e da redução no número de coeficientes do filtro não-linear, o uso da simetria octal dará ao filtro não-linear uma característica mais próxima da geometria circular na freqüência. Tal característica é interessante em interpolação porque significa que o tratamento dado à imagem, no domínio da freqüência, será o mesmo em todas as direções.

A complexidade do filtro não-linear pode ser aumentada ainda mais, adicionando-se

outras camadas escondidas à rede, que seriam constituídas por neurônios simples. Mais ainda, elementos de memória (atrasos) também podem ser adicionados nas camadas escondidas, de modo que os neurônios nestas camadas sejam filtros FIR bidimensionais. Neste caso, teríamos uma rede neural dinâmica. Apesar da rede neural, mostrada na Figura 3.14, ter filtros bidimensionais na primeira camada escondida, isto não significa que ela seja uma rede dinâmica. Lembre-se que utilizamos apenas o artifício de passar os atrasos externos para dentro da rede.

3.6. A Rede Perceptron Multicamadas vista como uma Função de Transferência Não-Linear

O objetivo desta seção é mostrar que uma rede neural nada mais é do que uma função de transferência não-linear, especialmente construída durante uma etapa de treinamento, para resolver um determinado problema.

Suponha uma rede neural contendo, além dos nós de entrada e da camada de saída, uma camada escondida, como aquela mostrada na Figura 3.13. A partir desta figura, podemos escrever a saída da rede neural como uma combinação de sigmóides, aplicadas aos filtros lineares da camada escondida mais um nível DC, isto é,

$$y(n) = \sum_{i=1}^{M} \tanh\left(v_i(n) - \theta_i\right) + \theta_0. \tag{3.16}$$

Sabemos ainda que $v_i(n)$ pode ser visto como o produto interno entre o vetor de entrada e o vetor de pesos sinápticos, isto é,

$$v_i(n) = \mathbf{w}_i^T(n)\,\mathbf{x}(n). \tag{3.17}$$

Dessa forma,

$$y(n) = \sum_{i=1}^{M} \tanh\left(\mathbf{w}_i^T(n)\,\mathbf{x}(n) - \theta_i\right) + \theta_0. \tag{3.18}$$

Assim, a saída está relacionada com a entrada através de uma função de transferência não-linear, obtida através da combinação de sigmóides, sendo que a escala das sigmóides é determinada pelos pesos sinápticos, a fase pelos limiares da primeira camada e o nível DC pelo limiar da camada de saída. Por este motivo, uma rede neural também pode ser

utilizada como uma aproximadora de funções, visto que a função a ser aproximada seria obtida, após um treinamento, como a própria função de transferência representada pela rede.

Este resultado é interessante porque mostra que poderíamos substituir a função de ativação por uma outra função, por exemplo, uma Gaussiana para melhor resolver um determinado problema de classificação, reconhecimento, filtragem, etc.. Neste caso, o limiar θ_i estaria relacionado com a média da i-ésima Gaussiana e o vetor de pesos sinápticos com o desvio padrão.

3.7. Algoritmos de Otimização Tradicionais

Um dos algoritimos de treinamento mais populares é o "back-propagation." Tal método, que é baseado no gradiente, consiste basicamente em propagar o sinal de entrada pela rede, calcular o erro em sua saída e, em seguida, voltar propagando este erro pelos neurônios das camadas intermediárias, até chegar na camada de entrada. À medida que o erro é propagado para "trás," corrige-se os pesos sinápticos. Entretanto, este método, além de ser de busca localizada, é também muito lento em termos de convergência. Além disso, devido à aplicação aqui desenvolvida, é melhor olhar o treinamento do ponto de vista da otimização de funções. Com isto em mente, apresentamos três algoritmos de otimização tradicionais : o método da Descida do Gradiente, o método do Gradiente Conjugado e o método de Newton. Nosso objetivo é utilizá-los em algoritmos híbridos, em conjunto com algoritmos genéticos.

No método de primeira ordem, conhecido como descida do gradiente, o vetor de direção é simplesmente o negativo do vetor gradiente. Por este motivo, o algoritmo converge para o mínimo através de um caminho em zig-zag. O método do gradiente conjugado em geral evita este problema, incorporando a derivada segunda no cálculo do vetor direção. Este método garante encontrar o mínimo de uma função quadrática qualquer, de N variáveis, em no máximo N passos. A seguir, apresentamos o método do gradiente conjugado aplicado à rede Perceptron Multicamadas.

Definido $p(n)$ como sendo o vetor direção na n-ésima interação do algoritmo, então o vetor de pesos da rede é atualizado de acordo com a seguinte regra

$$w(n+1) = w(n) + \eta(n)\, p(n), \tag{3.19}$$

sendo $\eta(n)$ o parâmetro taxa de aprendizado. A direção inicial é fixada como o negativo do vetor gradiente, isto é,

$$p(0) = -g(0). \tag{3.20}$$

Cada vetor direção sucessivo é então calculado como uma combinação linear do vetor gradiente atual e do vetor direção anterior, na forma

$$p(n+1) = -g(n+1) + \beta(n)\, p(n), \tag{3.21}$$

sendo $\beta(n)$ um parâmetro variável com o tempo.

Existem várias regras [21] para a determinação de $\beta(n)$ em termos dos vetores gradiente $g(n)$ e $g(n+1)$. Duas possíveis regras são mostradas a seguir :

- A fórmula de Fletcher-Reeves [21]

$$\beta(n) = \frac{g^T(n+1)\, g(n+1)}{g^T(n)\, g(n)} \tag{3.22}$$

- A fórmula de Polak-Ribière [21]

$$\beta(n) = \frac{g^T(n+1)\,[g(n+1) - g(n)]}{g^T(n)\, g(n)} \tag{3.23}$$

Tais regras foram obtidas a partir da utilização da derivada segunda no cálculo do vetor direção. Por este motivo, ambas as regras se reduzem à mesma forma no caso de uma função quadrática.

O cálculo da taxa de aprendizado na Equação 3.19 envolve uma busca em linha. O objetivo é obter um valor particular de η que minimize a função custo $E_{av}(w(n) + \eta p(n))$ fixados $w(n)$ e $p(n)$, onde E_{av} pode ser, por exemplo, o erro médio quadrático. Assim, podemos calcular $\eta(n)$ por

$$\eta(n) = \arg\min_{\eta} \{E_{av}(w(n) + \eta p(n))\}. \tag{3.24}$$

A precisão com que a busca em linha é realizada tem uma grande influência no desempenho do método do Gradiente Conjugado.

Já o método de Newton é de segunda ordem, visto que é obtido a partir da expansão em série de Taylor do incremento da função custo $E_{av}(w)$, em função de Δw, até o termo

de segunda ordem. O resultado final é que os pesos sinápticos passam a ser atualizados de acordo com a regra $w(n+1) = w(n) - H^{-1}g$, onde H é a matriz Hessiana, ou seja,

$$H = \frac{\partial^2 E_{av}}{\partial w^2}.$$

O problema na utilização deste método reside na complexidade computacional, visto que a matriz H é quase sempre de posto incompleto (singular), onde então é necessária a decomposição por valores singulares.

3.8. Otimização Via AG

Nesta seção apresentamos como aplicar os AGs na otimização dos coeficientes de um filtro FIR não-linear, baseado em uma rede perceptron multicamadas. Inicialmente, apresentamos uma abordagem matemática para a solução do problema de otimização de tal filtro via AGs. Em seguida, damos uma visão geral sobre os elementos do AG e apresentamos o algoritmo. Finalmente, mostramos que tal algoritmo converge para o ótimo global.

3.8.1. Formalização do Problema

Considere a rede Perceptron representada pela função não-linear

$$y(n) = f(W, X), \tag{3.25}$$

onde f é uma função contínua em cada ponto, $W = [w_1, w_2, \cdots, w_h]$ é o conjunto de h pesos sinápticos e $X = [x(n), x(n-1), \cdots, x(n-m)]$ é o conjunto das m entradas passadas e da entrada presente.

O filtro FIR não-linear representado pela Equação (3.25) pode ser estimado por

$$\widehat{y}(n) = f\left(\widehat{W}, X\right), \tag{3.26}$$

sendo $\widehat{W} = [\widehat{w}_1, \cdots, \widehat{w}_h]$ o conjunto de pesos sinápticos estimado e X o vetor definido acima.

Para aplicar um AG na otimização do filtro, cada peso $\widehat{w}_i$ é codificado como uma "string" de números binários, conhecidos como *genes*. Os genes são então cascateados para

formar uma "string" mais longa $\widehat{W}$, chamada *cromossomo*. Cada possível combinação dos parâmetros (pesos) estimados é, deste modo, representado por um cromossomo. A estratégia de otimização é aplicar o AG para procurar pelo melhor cromossomo $\widehat{W}$, de modo que $\widehat{y}(n)$ convirja para a saída ótima $y(n)$. O erro médio associado ao j-ésimo cromossomo na i-ésima geração, é definido por

$$e_j^i = \frac{1}{d}\sum_{k=1}^{d}\left(y(k) - \widehat{y}_{ji}(k)\right)^2, \tag{3.27}$$

sendo d o tamanho da janela sobre o qual os erros serão acumulados, e $\widehat{y}_{ji}(\cdot)$ é a saída estimada associada ao j-ésimo cromossomo dos pesos estimados para a i-ésima geração.

Assim, uma coleção de N cromossomos dos pesos estimados são examinados em cada geração i, na procura por um erro de estimação mínimo,

$$e_{\min}^i = \min\left(e_j^i\right) \quad , \quad \text{todo } j \in [1 \cdots N], \tag{3.28}$$

sobre o espaço total de pesos e força $e_{\min}^i$ a convergir para zero nas gerações sucessivas.

3.8.2. Características do AG

Esta subseção descreve os vários elementos do AG, incluindo o mecanismo de codificação, o método de inicialização, a regra de seleção dos cromossomos pais, detalhes no procedimento de cruzamento, a introdução da mutação como um modo de evitar mínimos locais, bem como a estratégia de extinção e imigração criada por Yao [1].

1) **Codificação** : Cada parâmetro estimado é codificado em genes, como uma "string" de dígitos binários representados por um inteiro sem sinal. Optamos por utilizar um inteiro sem sinal de 64 bits como sendo o número máximo de bits necessários para representar cada gene. Na verdade, são utilizados apenas $B_k \leq 64$ bits mais significativos, sendo que B_k é o comprimento, em bits, do k-ésimo gene. Desta forma, um cromossomo nada mais é do que um vetor de inteiros sem sinal. Assumindo que os parâmetros se encontram no intervalo $[-\eta_k, \eta_k]$, isto é,

$$|w_k| \leq \eta_k \text{ para } k = 1, 2, \cdots, h, \tag{3.29}$$

então, o comprimento B_k do k-ésimo gene pode ser calculado a partir de η_k e da precisão

desejada, δ_k, como

$$B_k = \left\lceil \log_2 \left(\frac{2\eta_k}{\delta_k} \right) \right\rceil,$$

de modo que o comprimento efetivo total de bits, utilizados em um cromossomo, é dado por

$$\sum_{k=1}^{h} B_k.$$

2) **Inicialização**. Os valores iniciais dos parâmetros estimados são aleatoriamente atribuídos. Portanto, no início do processo de estimação, Nh inteiros sem sinal, de 64 bits, são gerados para formar N vetores (cromossomos) com h genes, sendo que cada gene utilizará apenas B_k bits, dos 64 bits possíveis.

3) **Seleção**. A seleção de cromossomos pais é baseada na noção de "adequabilidade," que governa a extensão em que um indivíduo pode influenciar gerações futuras. No AG, D cromossomos são selecionados, para cruzamento, com base na sua adequabilidade em relação à adequabilidade total de todos os cromossomos na geração. Nesta dissertação, a adequabilidade para o j-ésimo cromossomo na i-ésima geração é definida como $e^i_{\max} - e_{ji}$, sendo que $e^i_{\max}$ é o erro de estimação máximo na i-ésima geração. Além disso, escalonamos, como descrito no Capítulo 2, esta medida de adequabilidade para atenuar os problemas relativos à pressão seletiva. Assim, a probabilidade do j-ésimo cromossomo ser selecionado para cruzamento na próxima geração é dada por

$$P^i_j = \frac{\alpha \left(e^i_{\max} - e_{ji} \right) + \beta}{\sum_{k=1}^{i} \left[\alpha \left(e^i_{\max} - e_{ki} \right) + \beta \right]} \tag{3.30}$$

Visto que o critério de seleção é probabilístico, cromossomos representando conjuntos de parâmetros que estão próximos a valores ótimos podem ser descartados, resultando em uma convergência lenta. Sabendo-se que a melhora do erro de estimação requer que os melhores cromossomos estejam presentes na população, o AG é modificado para garantir que estes cromossomos sejam preservados em toda geração. Assim, adotando-se uma estratégia elitista, em uma população de N indivíduos, D são selecionados para cruzamento e $\rho D (< D < N)$ melhores cromossomos entre esses N indivíduos são mantidos na geração seguinte.

Após a seleção dos D cromossomos pais, aplica-se o operador cruzamento para gerar $2D$ filhos. Como são repassados os ρD melhores indivíduos de uma geração para a outra,

então, após o cruzamento, haverão $2D + \rho D$ indivíduos, sendo que este número deverá corresponder ao tamanho original N da população, de modo que

$$N = 2D + \rho D,$$

ou seja,

$$\rho = \frac{N - 2D}{D}.$$

Visto então que os ρD são preservados para a próxima geração, o erro de estimação mínimo na geração atual será sempre menor ou igual ao erro de estimação mínimo na geração anterior.

4) **Cruzamento**. Entre os D possíveis pais, são gerados $(N - \rho D)/2$ possíveis pares de cromossomos. Cada um desses pares irá gerar dois outros cromossomos via o procedimento de recombinação, que imita o cruzamento biológico. Primeiro, um ponto de corte é aleatoriamente atribuído ao longo dos cromossomos :

Pai1	*xxxxxxxxxxxxxx*
Pai2	*yyyyyyyyyyyyyy*
	↑ ponto de corte

Segundo, gera-se dois cromossomos filhos, sendo que o Filho 1 contém a parte inicial de Pai 1 e a final de Pai 2, enquanto que o Filho 2 contém a parte inicial de Pai 2 e a final de Pai 1 :

Filho1	*xxxxyyyyyyyyyy*
Filho2	*yyyyyxxxxxxxxx*

Supondo que o esquema xxxxx********* corresponda a um baixo erro de estimação, onde * representa um valor qualquer, então o Filho 1 poderia corresponder a um erro de estimação mais baixo do que qualquer outro cromossomo pai. É este procedimento (biologicamente inspirado) de cruzamento, bem como a eliminação dos cromossomos não "adequados," que diferenciam os AGs de algoritmos cujo estilo de busca é puramente aleatório.

Neste trabalho, a recombinação de material genético é realizada por parâmetro (gene) e não por cromossomo, porque o número de possíveis filhos é maior no primeiro do que no segundo caso, aumentando-se, assim, o tamanho da região de busca realizada pela recombinação.

5) **Mutação**. Geralmente, ao longo de um período de várias gerações, a população tende a se tornar mais e mais estagnada quando um gene (parâmetro) começa a dominar. A mutação é freqüentemente introduzida para evitar convergência prematura para uma solução não ótima. A mutação altera aleatoriamente um determinado bit de um gene, de 0 para 1 ou de 1 para 0, com probabilidade P_m. O objetivo da mutação é introduzir perturbações ocasionais nos parâmetros estimados, para garantir que todos os pontos favoráveis no espaço de busca possam ser alcançados.

6) **Extinção e Imigração**. Pode-se mostrar que o número de cromossomos, em uma população, associados com um erro de estimação pequeno, cresce exponencialmente. Portanto, depois de algumas gerações os D cromossomos pais escolhidos para cruzamento são muito semelhantes entre si. É óbvio que se dois pais são muito parecidos, seus filhos também serão, e nenhuma nova informação é obtida. A estimação tende deste modo a estagnar, restando apenas a mutação como mecanismo para gerar melhores cromossomos. Visto que P_m é geralmente pequena ($\leq 0,01$), a probabilidade de redução adicional do erro de estimação é muito pequena, especialmente para cromossomos longos. Yao [1] propôs uma técnica drástica, chamada extinção e imigração, para vencer esta dificuldade. Extinção elimina todos os cromossomos na geração atual, exceto o cromossomo correspondente ao erro de estimação mínimo. $N-1$ cromossomos são então aleatoriamente gerados para substituir os cromossomos eliminados (uma imigração em massa). $D = (N-1)/2$ cromossomos, entre estes imigrantes, são então selecionados como pais. Estes, juntamente com os cromossomos sobreviventes, são permitidos cruzar para formar a próxima geração. Por conveniência, dizemos então que outra *era* se inicia. Extinção e Imigração em conjunto com a mutação é análogo a uma taxa de mutação variante com o tempo, na qual a probabilida de mutação de bit resultante seria próxima a $1/2$ no início de cada era (pois novos indivíduos são gerados aleatoriamente) e, então, se reduz para P_m (por exemplo, 0,01) nas gerações restantes dentro da era.

A extinção e imigração deve ocorrer quando nenhum decréscimo adicional no erro de estimação mínimo for detectado em, digamos, L_e gerações.

3.8.3. Estabelecendo o Algoritmo

Há vários parâmetros reguladores no algoritmo que devem ser atribuídos antes de executar o AG. Estes parâmetros são os seguintes :

N : número de cromossomos em cada geração.

D : número de cromossomos escolhidos como pais para cruzamento.

L_t : número de gerações toleradas caso nenhum progresso no erro de estimação mínimo seja observado antes do AG terminar.

L_e : número de gerações toleradas caso nenhuma melhora seja observada no erro de estimação mínimo antes que o operador extinção e imigração seja aplicado. Note que $L_e \ll L_t$.

P_m : probabilidade de mutação.

ρ : fração de pais permitidos sobreviver na próxima geração.

ξ : precisão desejada da estimação (em bits).

Duas variantes do AG são aqui estabelecidas (AG1 e AG2) sem e com o operador extinção e imigração, respectivamente. O Algoritmo AG2 é ilustrado na Figura 3.16 e resumido como segue :

1. Fixe os parâmetros reguladores como descrito acima. Codifique os pesos sinápticos a serem estimados em cromossomos. Faça $i = 0$, $k = 0$, e $m = 0$.
2. Inicialize N cromossomos, faça $i = i + 1$, $k = 0$, e $m = 0$.
3. Decodifique os cromossomos e calcule o erro de estimação e_{ij} para cada j-ésimo cromossomo na i-ésima geração. Faça $e^i_{\min} = \min_j \left(e^i_j\right)$.
4. Passe ρD melhores pais para a próxima geração.
5. Selecione D pais e gere $N - \rho D$ filhos. Invoque mutação com o procedimento de recombinação.
6. Se $e^i_{\min} = e^{i-1}_{\min}$, então

   $k = k + 1$ e $m = m + 1$;

   caso contrário, faça $k = 0$ e $m = 0$.

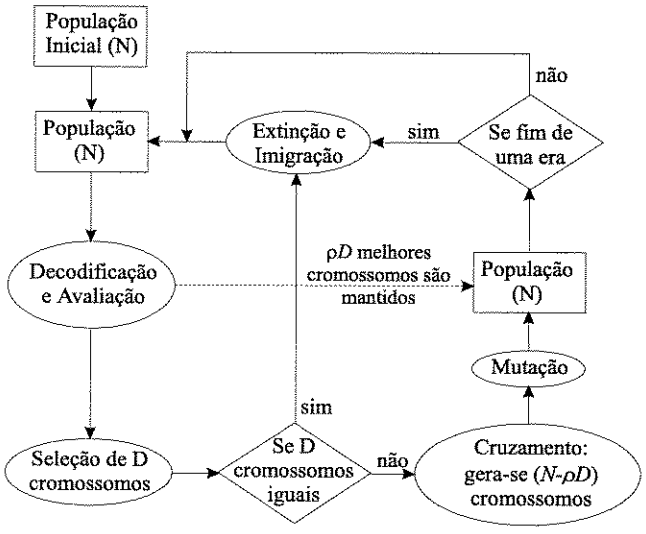

Figura 3.16: Algoritmo genético (AG2). O fim de uma era ocorre quando todos os D pais selecionados para cruzamento são iguais ou não há decréscimo no erro de estimação mínimo para L_e gerações.

7. Se D pais são todos idênticos, aplique a operação de extinção e imigração retornando ao passo (1) e salve somente o cromossomo correspondente a $e^i_{\min}$. Faça $k = 0$.
8. Se $k = L_e$, então volte ao passo (7).
9. Se $m = L_t$ (ou $e^i_{\min} < \xi$), então termine o algoritmo; caso contrário, vá ao passo (2).

Note que se a variância do ruído de quantização é conhecida a priori, então o AG é terminado se o erro de estimação é menor que ξ. Entretanto, se nenhuma informação estatística da medida do ruído é conhecida a priori, o AG é terminado quando nenhuma melhora no erro de estimação for detectada para L_t gerações. O algoritmo AG1 é basicamente o mesmo que AG2, exceto que os passos (6) e (7) para o operador extinção e imigração são eliminados.

3.8.4. Convergência do AG

Sabemos que, em cada geração, o indivíduo correspondente ao erro mínimo de estimação é sempre mantido na geração seguinte. Isto implica que a curva do erro mínimo é

decrescente e deve convergir, visto que tal curva é limitada inferiormente. Nesta seção é mostrado que o erro de estimação converge em probabilidade para zero.

Antes de prosseguir, é necessário definir o que é chamado de *casamento ideal*. Nosso objetivo é a estimação de parâmetros de uma rede não-linear através do emprego de AG. Isto implica que o conjunto de parâmetros da rede tem uma representação cromossômica, sendo que cada parâmetro corresponde a um gene, que por sua vez tem uma representação binária com uma quantidade finita de bits. Por este motivo, os parâmetros estimados, representados por estes genes, são realmente valores quantizados. Se o valor ideal do parâmetro estimado é w^* e sua representação quantizada é $[w^*]_Q$, o erro de quantização será

$$\left| w^* - [w^*]_Q \right| \leq \delta_k,\ k = 1, \cdots, h, \tag{3.31}$$

sendo que δ_k é o menor "quantum" para cada parâmetro estimado, que será tão menor quanto maior for a quantidade de bits alocada para o parâmetro. Portanto, diz-se que ocorreu *casamento ideal* quando, mesmo utilizando uma quantidade finita de bits na representação dos parâmentros, coincidiu-se do parâmetro estimado ter sido quantizado sem erro.

A prova de que o erro de estimação converge, em probabilidade, para zero, é dividida em duas proposições, um teorema e um corolário. Nas duas proposições são obtidos resultados que serão utilizados no teorema. Este teorema [1] mostra que o erro mínimo de estimação converge em probabilidade para zero, a uma taxa específica, sob a hipótese de casamento ideal. Esta taxa será proporcional a $\sum \theta_i$, sendo θ_i o número de todos os possíveis cromossomos na geração $i+1$ que correspondem a erros de estimação menores do que $e^i_{\min}$. Se a hipótese de casamento ideal falha, então o erro de estimação convergirá para uma constante.

Este teorema Yao [1] prova a convergência em probabilidade apenas para o AG sem extinção e imigração. O corolário deste teorema provará tal convergência, em probabilidade, para o caso em que o operador *Extinção e Imigração* for utilizado.

Proposição 3.1 *Suponha que D cromossomos pais são selecionados da geração atual. De todos os possíveis filhos que podem ser obtidos a partir do conjunto de D pais, através da recombinação, a probabilidade de se gerar um dado cromossomo filho, assumindo que*

não há mutação ($P_m = 0$), *é*

$$P = \frac{1}{D\left(D-1\right)} \prod_{k=1}^{h} \frac{1}{2\left(B_k - 1\right)}. \tag{3.32}$$

Prova : Seja (i, j) um possível par de cromossomos pais dentre os D selecionados para cruzamento. Seja $s_1, \cdots, s_h$ pontos de corte que ocorreram para os parâmetros $\widehat{w}_1, \cdots, \widehat{w}_h$, respectivamente, sendo $s_k \in [1 \cdots B_k - 1], k = 1 \cdots h$, onde B_k representa o número de bits do k-ésimo parâmetro. Há $D\left(D-1\right)$ formas de escolher um par de cromossomos para cruzamento, de modo que a probabilidade de selecionarmos um par de cromossomos (i, j) é

$$P_{ij} = \frac{1}{D\left(D-1\right)}.$$

Visto que existe um ponto de corte independente para cada parâmetro, que são $B_k - 1$ possíveis posições de corte, e que para cada posição de corte são gerados dois filhos, então a quantidade de cromossomos filhos gerados, devido ao cruzamento envolvendo apenas dois genes (cada um vindo de um cromossomo pai), é $2\left(B_k - 1\right)$ e o número de possíveis filhos devido ao cruzamento envolvendo todos os genes é

$$\prod_{k=1}^{h} 2\left(B_k - 1\right).$$

Desta forma, a probabilidade de se gerar um cromossomo filho a partir do cruzamento de dois cromossomos pais quaisquer, é dada por

$$P_{s_1 \cdots s_h} = \prod_{k=1}^{h} P_{s_k} = \frac{1}{\prod_{k=1}^{h} 2\left(B_k - 1\right)}.$$

Portanto, a probabilidade de se gerar um cromossomo filho a partir do cruzamento de um par de cromossomos escolhidos dentre D pais é

$$P_{ijs_1 \cdots s_h} = P_{ij} P_{s_1 \cdots s_h} = \frac{1}{D\left(D-1\right)} \frac{1}{\prod_{k=1}^{h} 2\left(B_k - 1\right)}. \tag{c.q.d.}$$

Observe que as probabilidades puderam ser multiplicadas porque a escolha dos pais é feita independente da recombinação.

Note que para cromossomos idênticos, a probabilidade de se gerar cada um dos possíveis filhos idênticos também é dada pela Equação 3.32.

Proposição 3.2 *Dados dois cromossomos arbitrários C_i e C_j com erros de estimação e_i e e_j respectivamente, existe uma constante $\widetilde{\alpha} > 0$ tal que se $e_i > e_j$ então*

$$e_i - \widetilde{\alpha} \geq e_j. \tag{3.33}$$

Prova : Faça $\alpha_{ij} = |e_i - e_j|$, para todo i, j. Visto que cada cromossoomo é representado por um número finito de bits, existe somente um número finito de cromossomos e, conseqüentemente, um número finito de possíveis α_{ij}. Faça $\widetilde{\alpha}$ igual ao menor α_{ij} não nulo.

Teorema 3.3 *Faça $e^g_{\min}$ igual ao erro de estimação na geração g, como em (3.28). Se a condição de casamento ideal é satisfeita para GA1, então*

$$E\left(e^g_{\min}\right) \leq E\left(e^0_{\min}\right) - c\sum_{i=0}^{g-1} E\left(\theta_i\right) \qquad \forall g \geq 1, \tag{3.34}$$

sendo c uma constante. Mais ainda,

$$P\left(e^g_{\min} \geq \widetilde{\alpha}\right) \to 0 \quad \textit{quando} \quad g \to \infty, \tag{3.35}$$

onde $\widetilde{\alpha}$ é definida como na Proposição 3.2.

Prova :

Antes de iniciarmos a prova deste teorema, descrevemos os principais símbolos utilizados :

Φ	Conjunto de todas as possíveis subpopulações de D cromossomos escolhidas a partir da população para cruzamento na geração g.
ϕ	Um dado conjunto de D cromossomos pais.
B_k	Todo k-ésimo parâmetro de qualquer cromossomo tem a mesma quantidade de bits B_k.
s_k	Corresponde a um ponto de corte para crossover no parâmetro $\widehat{w}_{ik}$ do cromossomo i com $\widehat{w}_{jk}$ do cromossomo j, sendo que $s_k \in [1 \cdots B_k - 1]$ para $k = 1 \cdots h$ e (i, j) é o par de cromossomos escolhido para cruzamento.
P_m	Probabilidade de mutação, ou seja, inverter um bit.
U	Conjunto de todos os possíveis padrões de mutação incluindo o padrão de não mutação.
$e^{g+1}_{ijs_1 \cdots s_h u}$	É o erro de estimação na geração $g+1$ associado ao cromossomo que foi obtido por cruzamento do par (i, j) de cromossomos (escolhidos para cruzamento na geração anterior) com pontos de corte em $s_1, \cdots, s_h$ e que sofreu mutação seguindo o padrão u.
θ_g	É o número de todos os cromossomos possíveis na geração $g+1$ que correspondem a erros de estimação menores do que o menor erro de estimação da geração anterior ($e^g_{\min}$).

Iniciamos a demonstração calculando o valor esperado de $e^{g+1}_{\min}$, dado que os D cromossomos ϕ foram selecionados na geração g, isto é,

$$
\begin{aligned}
E\left(e^{g+1}_{\min} \,|\phi\right) &= \sum_{i \in D} \sum_{\substack{j \in D \\ j \neq i}} \sum_{s_1} \cdots \sum_{s_h} \sum_{u \in U} P_{ijs_1 \cdots s_h u} \min\left(e^g_{\min}, e^{g+1}_{ijs_1 \cdots s_h u}\right) \\
&= \sum_{ijs_1 \cdots s_h u} P_{ijs_1 \cdots s_h} P_u \min\left(e^g_{\min}, e^{g+1}_{ijs_1 \cdots s_h u}\right) \\
&= P \sum_{ijs_1 \cdots s_h u} P_u \min\left(e^g_{\min}, e^{g+1}_{ijs_1 \cdots s_h u}\right) \\
&= P \sum_{\substack{ijs_1 \cdots s_h u \\ e^{g+1}_{ijs_1 \cdots s_h u} \geq e^g_{\min}}} P_u \min\left(e^g_{\min}, e^{g+1}_{ijs_1 \cdots s_h u}\right)
\end{aligned}
$$

$$
\begin{aligned}
&+P \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u \min\left(e^g_{\min}, e^{g+1}_{ijs_1\cdots s_h u}\right) \\
&= P \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} \geq e^g_{\min}}} P_u e^g_{\min} + P \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u e^{g+1}_{ijs_1\cdots s_h u} \qquad (3.36)
\end{aligned}
$$

O próximo passo é escrever $E\left(e^{g+1}_{\min}|\phi\right)$ em função de $e^g_{\min}$ e uma constante. Para isto, definamos

$$\alpha^g_{ijs_1\cdots s_h u} = e^g_{\min} - e^{g+1}_{ijs_1\cdots s_h u} > 0.$$

Então,

$$
\begin{aligned}
E\left(e^{g+1}_{\min}|\phi\right) &= P \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} \geq e^g_{\min}}} P_u e^g_{\min} + P \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u e^g_{\min} \\
&\quad -P \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u \alpha^g_{ijs_1\cdots s_h u}.
\end{aligned}
$$

Juntando os dois primeiros somatórios, obtemos

$$E\left(e^{g+1}_{\min}|\phi\right) = P \sum_{ijs_1\cdots s_h u} P_u e^g_{\min} - P \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u \alpha^g_{ijs_1\cdots s_h u}.$$

Definindo,

$$F_g = \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u \alpha^g_{ijs_1\cdots s_h u}, \qquad (3.37)$$

De modo que obtemos,

$$
\begin{aligned}
E\left(e^{g+1}_{\min}|\phi\right) &= P \sum_{ijs_1\cdots s_h u} P_u e^g_{\min} - PF_g \\
&= e^g_{\min} P \sum_{ijs_1\cdots s_h u} P_u - PF_g \\
&= e^g_{\min} \sum_{ijs_1\cdots s_h} P \sum_{u\in U} P_u - PF_g \\
&= e^g_{\min} \sum_{ijs_1\cdots s_h} P_{ijs_1\cdots s_h} \sum_{u\in U} P_u - PF_g \\
&= e^g_{\min} - PF_g. \qquad (3.38)
\end{aligned}
$$

O próximo passo é desenvolver a Equação 3.37 escrevendo P_u, que é a probabilidade de ocorrer um determinado padrão de mutação (quais bits no cromossomo serão mutados), em função da probabilidade de mutação de um bit, P_m, que é uma constante, e a partir daí obter uma desigualdade envolvendo F_g, de modo a obter a desigualdade (3.34). Com este objetivo, definamos $\overline{P}_J$ como sendo a probabilidade de mutação de J bits no cromossomo, isto é,

$$\overline{P}_J = (P_m)^J (1 - P_m)^{\left(\sum_{k=1}^{h} B_k\right) - J}, \quad J = 0, 1, \cdots \tag{3.39}$$

Observe que $\sum_{k=1}^{h} B_k$ corresponde ao número total de bits no cromossomo. Observe também que $\overline{P}_J$ está diretamente associado com P_u. Por exemplo, suponha um cromossomo com 5 bits e um padrão de mutação $u = 10101$, sendo que 1 significa mutação e 0 significa não mutação do bit correspondente no cromossomo. Então $P_u = P_{10101} = \overline{P}_3$ e, mais ainda, todas as probabilidades dos $\binom{5}{2} = 20$ padrões de mutação que correspondem a mutação de três bits no cromossomo, são iguais entre si e iguais a $\overline{P}_3$. Ou seja, $P_{00111} = P_{01011} = P_{10011} = \cdots = P_{11100} = \overline{P}_3$. Portanto, se u_i representa o i-ésimo bit do padrão u, então

$$P_u = \overline{P}_J, \qquad J = \sum_i u_i. \tag{3.40}$$

Definamos também $\widetilde{P}$ como sendo a probabilidade de mutação de todos os bits no cromossomo, isto é,

$$\widetilde{P} = (P_m)^{\left(\sum_{k=1}^{h} B_k\right)}. \tag{3.41}$$

Sabendo que P_m é escolhida como sendo menor que $0,1$, obtemos, a partir de (3.39) e (3.41),

$$\overline{P}_J \geq \widetilde{P}, \quad J = 0, 1, \cdots. \tag{3.42}$$

Deste modo, a partir de (3.40) e (3.42), obtemos

$$P_u \geq \widetilde{P}, \quad \text{para todo } u \in U. \tag{3.43}$$

Voltando a (3.37), temos

$$F_g = \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u \alpha^g_{ijs_1\cdots s_h u} \geq \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u \min\left(\alpha^g_{ijs_1\cdots s_h u}\right). \tag{3.44}$$

Substituindo (3.43) em (3.44), obtemos

$$\begin{aligned} F_g &\geq \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} P_u \min\left(\alpha^g_{ijs_1\cdots s_h u}\right) \geq \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} \widetilde{P} \min\left(\alpha^g_{ijs_1\cdots s_h u}\right) = \\ &= \widetilde{P} \min\left(\alpha^g_{ijs_1\cdots s_h u}\right) \sum_{\substack{ijs_1\cdots s_h u \\ e^{g+1}_{ijs_1\cdots s_h u} < e^g_{\min}}} 1 \\ &= \widetilde{P} \min\left(\alpha^g_{ijs_1\cdots s_h u}\right) \theta_g, \end{aligned}$$

ou seja,

$$F_g \geq \widetilde{P} \min\left(\alpha^g_{ijs_1\cdots s_h u}\right) \theta_g. \tag{3.45}$$

Observe que recaímos em θ_g, que é o número de todos os possíveis cromossomos na geração $g+1$ que correspondem a um erro de estimação menor que $e^g_{\min}$. Aplicando a Proposição 2 em (3.45), obtemos que

$$F_g \geq \widetilde{P} \min\left(\alpha^g_{ijs_1\cdots s_h u}\right) \theta_g \geq \widetilde{P}\widetilde{\alpha}\theta_g. \tag{3.46}$$

Substituindo (3.46) em (3.38), obtemos

$$E\left(e^{g+1}_{\min} | \phi\right) \leq e^g_{\min} - P\widetilde{P}\widetilde{\alpha}\theta_g. \tag{3.47}$$

Pelo teorema da esperança total,

$$E\left(E\left(e^{g+1}_{\min} | \phi\right)\right) = E\left(e^{g+1}_{\min}\right).$$

Conseqüentemente, de (3.47),

$$E\left(e^{g+1}_{\min}\right) \leq \sum_{\phi \in \Phi} P_\phi e^g_{\min} - P\widetilde{P}\widetilde{\alpha} \sum_{\phi \in \Phi} P_\phi \theta_g,$$

ou seja,

$$E\left(e^{g+1}_{\min}\right) \leq E\left(e^g_{\min}\right) - P\widetilde{P}\widetilde{\alpha} E\left(\theta_g\right) \tag{3.48}$$

Usando (3.48) recursivamente, obtemos

$$\begin{aligned} E\left(e_{\min}^{g}\right) &\leq E\left(e_{\min}^{g-1}\right) - P\widetilde{P}\widetilde{\alpha}E\left(\theta_{g-1}\right) \\ &\leq E\left(e_{\min}^{g-2}\right) - P\widetilde{P}\widetilde{\alpha}\left[E\left(\theta_{g-1}\right) + E\left(\theta_{g-2}\right)\right] \\ &\vdots \\ &\leq E\left(e_{\min}^{0}\right) - P\widetilde{P}\widetilde{\alpha}\sum_{i=0}^{g-1} E\left(\theta_i\right), \text{ para todo } g \geq 1. \end{aligned}$$

Portanto,

$$E\left(e_{\min}^{g}\right) \leq E\left(e_{\min}^{0}\right) - c\sum_{i=0}^{g-1} E\left(\theta_i\right), \tag{3.49}$$

onde $c = P\widetilde{P}\widetilde{\alpha}$ é uma constante.

Visto que $\theta_i \geq 1$ para todo $i \geq 0$, $\lim_{g\to\infty} E\left(e_{\min}^{g}\right) = 0$. Pela extensão da desigualdade de Chebychev,

$$\lim_{g\to\infty} P\left(e_{\min}^{g} \geq \widetilde{\alpha}\right) \leq \lim_{g\to\infty} \frac{E\left(e_{\min}^{g}\right)}{\widetilde{\alpha}} = 0. \tag{3.50}$$

Desta forma, está demonstrado o Teorema 1 que prova que o AG converge supondo casamento ideal. O próximo passo é provar a convergência do AG quando utilizado com o operador Extinção e Imigração. Isto é feito no corolário seguinte.

Corolário 3.1 *Faça cada era* $1, 2, \cdots, m$ *consistir de* $T_1, T_2, \cdots, T_m$ *gerações, respectivamente, e defina* g *como sendo a* j*-ésima geração após* m *eras, isto é,*

$$g = T_1 + T_2 + \cdots + T_m + j$$

Se a hipótese de casamento ideal é satisfeita para AG2, então (3.34) e (3.35) são ambas mantidas.

Prova : Podemos imaginar cada era como sendo um AG independente que na sua população inicial contém o melhor cromossomo da era anterior. Com este raciocínio, a Equação 3.34 é válida se considerarmos cada era isoladamente. Se mostrarmos que tal equação é válida para uma geração qualquer, independentemente da era à qual pertença, a convergência estará provada para o AG2.

Sejam g gerações que podem ser divididas em m eras mais j gerações na era $m+1$. Assim sendo, podemos desenvolver as $m+1$ equações abaixo.

Como a primeira era vai de 0 a T_1, então de (3.49) obtemos

$$E\left(e_{\min}^{T_1}\right) \leq E\left(e_{\min}^{0}\right) - c\sum_{i=0}^{T_1-1} E\left(\theta_i\right). \tag{3.51}$$

A segunda era vai de T_1 a T_1+T_2, então de (3.49) obtemos

$$E\left(e_{\min}^{T_1+T_2}\right) \leq E\left(e_{\min}^{T_1}\right) - c\sum_{i=T_1}^{T_1+T_2-1} E\left(\theta_i\right). \tag{3.52}$$

$$\vdots$$

E assim sucessivamente até a m-ésima era que vai de $T_1+\cdots+T_{m-1}$ a $T_1+\cdots+T_m$,

$$E\left(e_{\min}^{T_1+\cdots+T_m}\right) \leq E\left(e_{\min}^{T_1+\cdots+T_{m-1}}\right) - c\sum_{i=T_1+\cdots+T_{m-1}}^{T_1+\cdots+T_m-1} E\left(\theta_i\right) \tag{3.53}$$

Finalmente, com j gerações na era $m+1$, obtemos

$$E\left(e_{\min}^{T_1+\cdots+T_m+j}\right) \leq E\left(e_{\min}^{T_1+\cdots+T_m}\right) - c\sum_{i=T_1+\cdots+T_m}^{T_1+\cdots+T_m+j-1} E\left(\theta_i\right). \tag{3.54}$$

Substituindo (3.51) em (3.52) e assim sucessivamente ao longo das $m+1$ equações até (3.54), obtemos

$$E\left(e_{\min}^{T_1+\cdots+T_m+j}\right) \leq E\left(e_{\min}^{0}\right) - c\sum_{i=0}^{T_1+\cdots+T_m+j-1} E\left(\theta_i\right), \tag{3.55}$$

ou seja,

$$E\left(e_{\min}^{g}\right) \leq E\left(e_{\min}^{0}\right) - c\sum_{i=0}^{g-1} E\left(\theta_i\right). \tag{3.56}$$

Conseqüentemente, (3.35) também é válida se utilizarmos o AG2 e o corolário está provado.

(c.q.d.)

O teorema aqui demonstrado e seu corolário implicam que se a condição de casamento ideal é satisfeita, então o erro de estimação converge em probabilidade para zero. Digamos que os valores ideais dos coeficientes estimados para os quais a rede deva convergir sejam $w_1^*,\ldots,w_h^*$ e $y^*(n)$ seja a saída da rede para tais valores. De fato, se a condição de casamento ideal for mantida, a rede convergirá para tais coeficientes. Entretanto, se tal

condição não se mantiver, então a rede convergirá para $[w_1^*]_Q, \ldots, [w_h^*]_Q$ e a saída será $[y^*(n)]_Q$. Neste caso, podemos definir o erro de estimação quantizado como sendo

$$[e^*_{\min}]_Q = \frac{1}{d}\sum_{k=1}^{d}\left(y^*(k) - [y^*(k)]_Q\right)^2. \tag{3.57}$$

Desta forma, a partir de (3.31) vemos que o erro $[e^*_{\min}]_Q$ será uma constante que dependerá do erro de quantização δ_k, do número de nós na camada de entrada e do ganho do sistema. Como o erro $[e^*_{\min}]_Q$ é uma constante, então $\widehat{y}(n)$ de fato converge em probabilidade para $[y^*(n)]_Q$ quando $g \to \infty$.

3.9. Esquemas Híbridos AG-Gradiente

O espaço de busca de um AG pode ser contínuo ou discreto, côncavo ou convexo e, além disso, a função objetivo pode ter muitos picos e vales. Mais ainda, a função objetivo definida neste espaço pode ser derivável ou não. Daí a robustez dos AGs. Entretanto, quando a função objetivo é contínua, derivável e tem apenas um ponto estacionário, os métodos baseados no gradiente convergem bem mais rapidamente e com um custo computacional menor do que os AGs. Isto revela que, apesar do AG ter a capacidade de escapar de ótimos locais, dentro de uma determinada região, dominada por um ponto estacionário, a eficiência dos métodos baseados no gradiente é maior. Surge então a idéia de integrar um algoritmo eficiente de busca global com outro algoritmo eficiente de busca local, utilizando ambos o AG e o método do gradiente.

Existem inúmeras formas distintas de unir os dois métodos para compor um algoritmo híbrido. Neste trabalho, são sugeridos três algoritmos híbridos. No primeiro, mostrado na Figura 3.17, cascateamos o AG com o GC (Gradiente Conjugado). A idéia é utilizar o AG para encontrar uma condição inicial para o GC, que esteja dentro de uma região do espaço de busca dominada pelo ótimo global. Chamaremos esta estratégia de **Hibrido-1**.

No segundo, mostrado na Figura 3.18, ambos os algoritmos, AG e DG (Descida do Gradiente), rodam paralelamente. Só existe interação entre os dois algoritmos quando ocorre extinção e imigração no AG. Neste caso, se o melhor indivíduo no AG for menos adequado do que o resultado parcial do DG, substituímos tal indivíduo pela solução temporária do DG e vice-versa. Chamaremos esta estratégia de **Híbrido-2**.

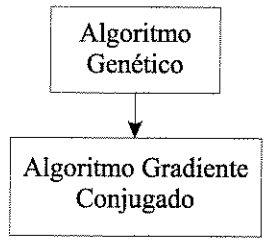

Figura 3.17: Esquema Híbrido-1.

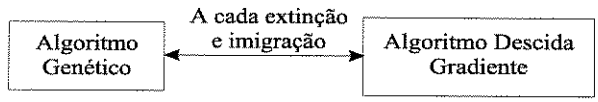

Figura 3.18: Esquema Híbrido-2.

No terceiro método, mostrado na Figura 3.19, criamos outro operador, chamado *mutaçãoG* que, ao ser aplicado à população, altera o melhor indivíduo com base no gradiente. A esta estratégia damos o nome de **Híbrido-3**.

3.10. O Sistema Implementado

O objetivo desta seção é apresentar o sistema implementado de interpolação não-linear. Tal sistema envolve dizimação espacial, otimização de parâmetros, interpolação convencional e filtragem não-linear, conforme apresentado na Figura 3.20.

Antes de prosseguirmos, é interessante adotar uma notação para referenciar as diferentes escalas da imagem a ser interpolada. Assim, definimos I_1 como sendo a imagem de referência a ser interpolada, I_2 como sendo a ampliação de I_1 por 2, I_4 a ampliação de I_1 por 4, e assim por diante. Definimos também $I_{1/2}$ como sendo a versão dizimada de I_1 por 2.

Utilizamos uma rede perceptron multicamadas com duas camadas escondidas e um neurônio na camada de saída. O número de nós de entrada vai depender da ordem dos filtros bidimensionais utilizados. Por exemplo, se forem utilizados filtros de ordem N, então serão N^2 nós de entrada distribuídos em uma janela $N \times N$. Tais filtros têm simetria octal, tornando o número de sinapses, por filtro, bem menor do que as $N^2 + 1$ (o termo +1 é

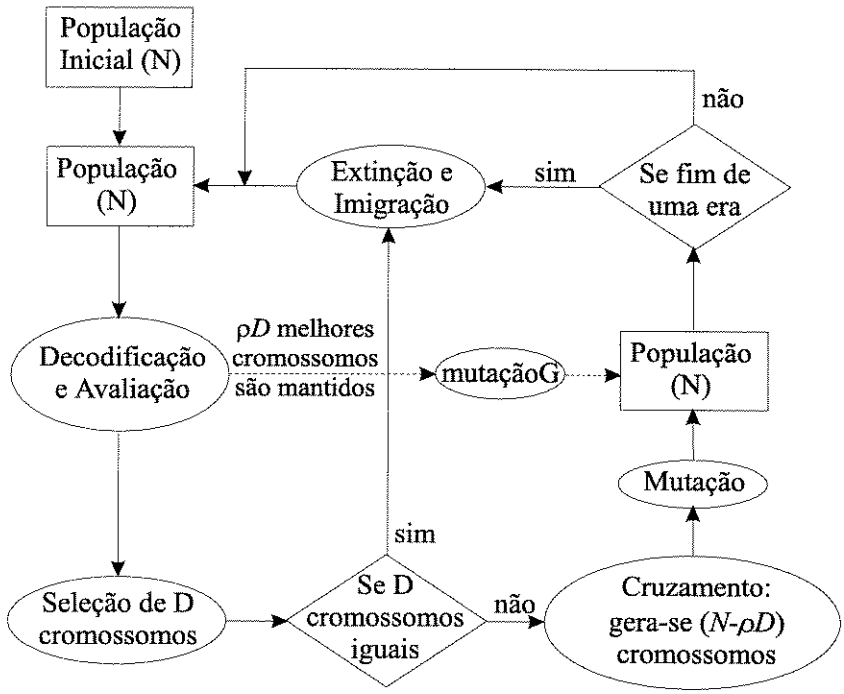

Figura 3.19: Esquema Híbrido-3.

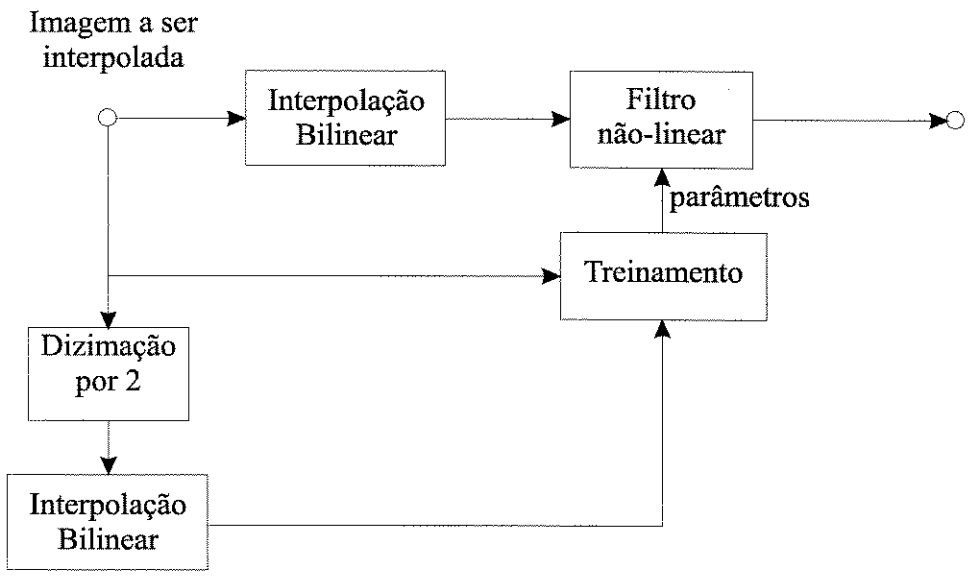

Figura 3.20: Sistema implementado para a interpolação não-linear de imagens.

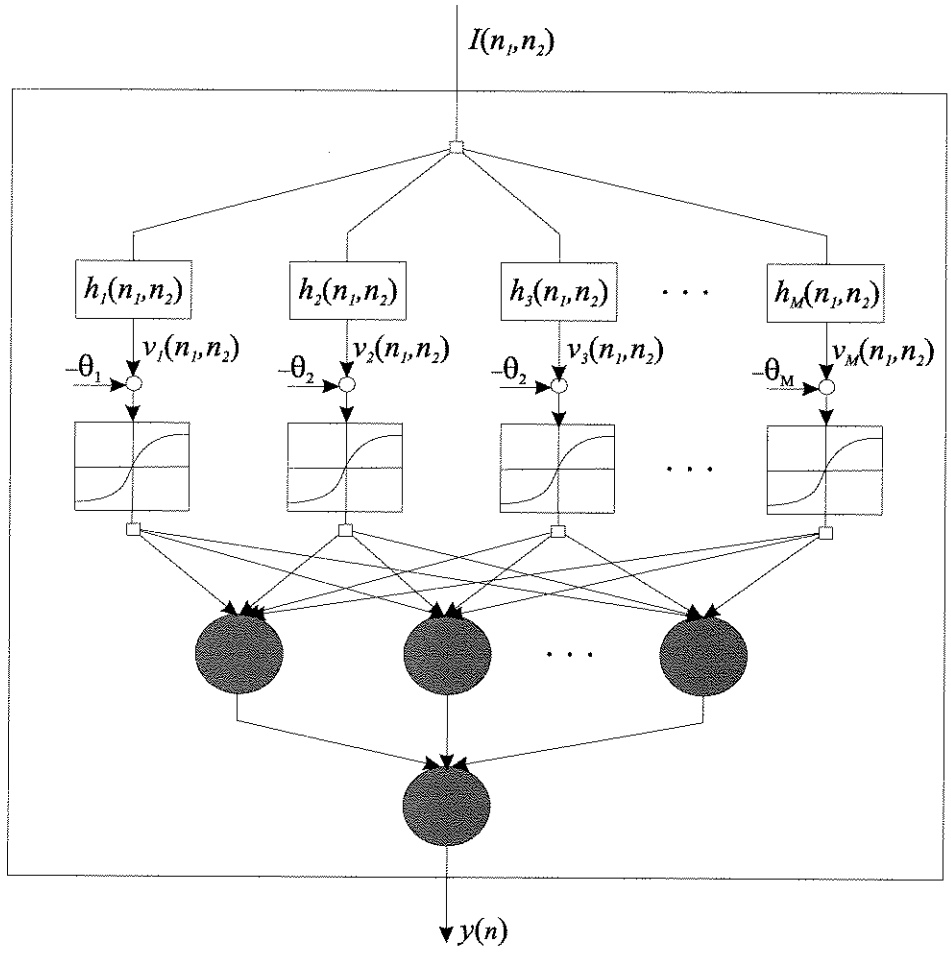

Figura 3.21: Arquitetura do filtro FIR não linear implementado. Os círculos cheios representam neurônios como os mostrados na Figura 3.6.

devido ao limiar de cada neurônio) caso não fosse utilizada simetria alguma. Por exemplo, de acordo com as Equações 3.13 e 3.15, o número de sinapses efetivas para um filtro de ordem 5, é de apenas $6 + 1 = 7$, que é bem menor do que as 26 sinapses que deveriam ser consideradas se não houvesse simetria alguma. A Figura 3.21 mostra a arquitetura da rede utilizada. Trata-se de uma rede $N : M_1 : M_2 : 1$, sendo N a ordem dos filtros bidimensionais, M_1 o número de filtros bidimensionais utilizados na primeira camada escondida e M_2 o número de neurônios na segunda camada escondida.

A utilização da interpolação bilinear no sistema tem dupla função. Primeiro, evitar que os filtros lineares bidimensionais da rede neural tornem-se, após o treinamento, em meros filtros passa-baixas com freqüência de corte em $\pi/2$. Segundo, realizar o ajuste de

escala da imagem, de modo que a informação original seja preservada. Isto significa que, ao realizar a interpolação bilinear, estamos dando um grau de liberdade maior para a rede neural. Mais ainda, a função da rede neural passa a ser a de enriquecimento da informação original, com base no modelo de construção obtido na etapa de treinamento.

Visto que o treinamento é supervisionado, precisamos de uma imagem de saída, com maior resolução espacial, e de uma imagem de entrada, com menor resolução espacial, sendo que ambas devem ser conhecidas. Com isto em mente, utilizamos como imagem de saída, para o treinamento, a imagem I_1 e, como entrada, a sua versão dizimada $I_{1/2}$.

Entretanto, esta estatégia dá margem a dúvidas quanto à validade deste modelo. Afinal, será que este modelo descreveria apenas a forma de obtenção de I_1 a partir de $I_{1/2}$? Este modelo poderia ser estendido para obter-se I_2 ou I_4 a partir de I_1? A resposta a estas perguntas tem por base a similaridade visual existente entre as diferentes escalas de I_1. De fato, certas estruturas de I_1, como por exemplo, boca, cabelos, nariz, etc., continuam presentes em $I_{1/2}$, I_2 e I_4. Para ser mais específico, os contornos (bordas) que determinam estes elementos são preservados. Portanto, é de se esperar que a rede aprenda como construir tais bordas em uma imagem de mais alta resolução a partir de uma imagem de referência. De fato, pudemos comprovar esta hipótese através de diversas simulações.

Obviamente, há detalhes específicos que vão sendo acrescentados à medida que se aumenta a resolução da imagem. Desta forma, o modelo de construção será tão melhor quanto menor for o fator de ampliação. Este resultado também é comprovado a partir das simulações.

O espaço de busca do AG, para este caso, é extremamente grande. Por exemplo, para uma rede 3 :4 :1 :1, o número de pesos sinápticos é 23. Se utilizarmos 16 bits para representar cada parâmetro, serão 65536 pontos por parâmetro, de modo que o número total de pontos no espaço de busca é $65536^{23} = 6.012269011901e + 110$.

3.11. Interpolação Adaptativa com Preservação de Bordas

O objetivo desta seção é apresentar um algoritmo simples para interpolação de imagens, visando à máxima preservação possível das bordas. Uma variante deste algoritmo

também é implementada, com o objetivo de melhorar a qualidade subjetiva final, permitindo-se uma certa degradação das bordas. Este algoritmo não utiliza Algoritmos Genéticos para a otimização do interpolador.

Nas seções anteriores, mostramos um sistema complexo de interpolação não-linear de imagens. Um modelo específico de como estimar informações inexistentes de alta freqüência, ou perdidas devido ao processo de dizimação, a partir da informação conhecida de mais baixa freqüência é obtido para a imagem a ser interpolada. Daí a complexidade do método.

Em vez de deixar a cargo do filtro não-linear e do AG a construção deste modelo que, além disso, é específico à imagem que se deseja interpolar, sugerimos nesta seção um modelo mais simples, que irá preservar ao máximo as bordas da imagem original. Mais ainda, este modelo é genérico e, por este motivo, não necessita de uma etapa de treinamento para extrair um modelo de multiresolução específico para a imagem a ser interpolada.

3.11.1. Interpolação Adaptativa com Máxima Preservação de Bordas

Uma vez detectadas as bordas da imagem, o algoritmo é bastante simples. A idéia básica é calcular os pontos de interesse através da definição de uma região cujas fronteiras são as bordas, onde os pontos conhecidos e os pontos a serem determinados se encontram. Esta análise é feita dentro de uma janela de pequenas dimensões.

Na Figura 3.22, mostramos o sistema implementado. Inicialmente, utilizamos um detector de bordas que vai explicitar os contornos a serem preservados. Em seguida, intercalamos as amostras de ambas as imagens com zeros, de modo a dobrar o tamanho delas. Finalmente, realizamos a sua convolução com um filtro cujos coeficientes vão variar localmente, de acordo com a posição do pixel a ser calculado e com a presença de bordas, bem como com o modo pelo qual essas bordas interceptam a janela de convolução.

No modelo de interpolação aqui proposto, a posição do pixel a ser calculado é importante. Como é mostrado na Figura 3.23, dentro de uma janela de convolução 3×3, aqui adotada, o pixel a ser calculado pode se encontrar em um ponto conhecido (ponto 0), entre dois pontos conhecidos (pontos 1 e 2) ou entre quatro pontos conhecidos (ponto 3).

O algoritmo foi desenvolvido para trabalhar apenas com uma janela de dimensão 3×3,

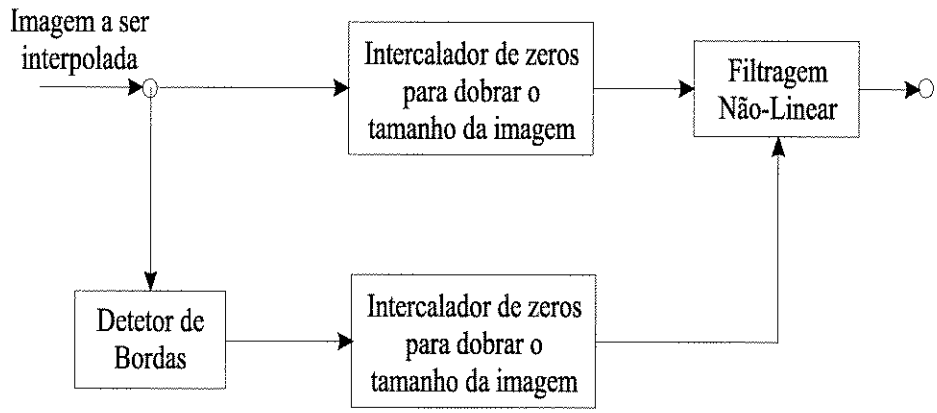

Figura 3.22: Sistema que implementa o interpolador adaptativo que explora a configuração local de bordas da imagem.

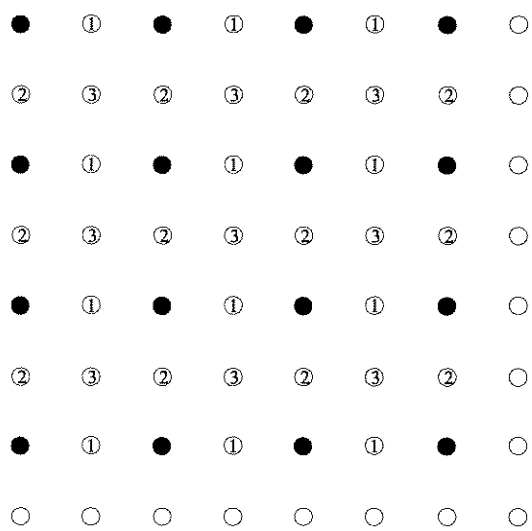

Figura 3.23: Tipo da posição do pixel na imagem intercalada com zeros. As bolas cheias correspondem aos pixels conhecidos e as bolas vazias aos pixels que devem ser determinados pelo interpolador não-linear.

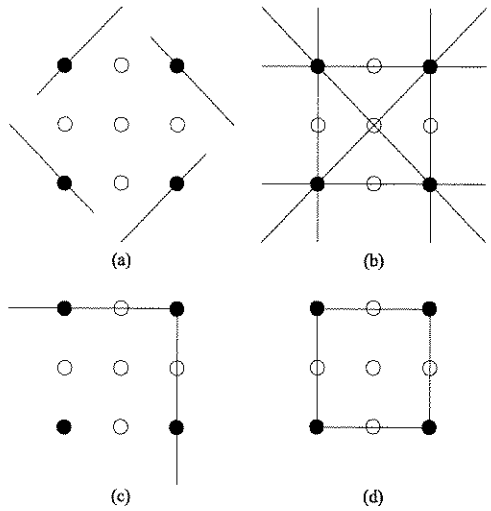

Figura 3.24: Modelo de disposição de uma borda na janela de convolução.

mas pode ser generalizado. Como esta janela é pequena, é razoável supor que apenas uma borda pode interceptar a janela de convolução. Neste caso, apresentamos na Figura 3.24 todas as possíveis situações em que a janela é interceptada por uma borda.

No cálculo dos pontos 1 e 2, independentemente do número de vezes que uma borda interceptar a janela de convolução, sempre haverá apenas três casos a serem analisados. Em dois deles, o caso em que os dois pontos mais próximos são interceptados pela borda e o caso em que ambos os pontos não são interceptados, os pontos tipo 1 e 2 são calculados como a média aritmética dos dois pontos mais próximos. Quando apenas um dos dois pontos mais próximos é interceptado, os pontos 1 e 2 são feitos iguais ao ponto mais próximo não interceptado.

No cálculo do Ponto 3, são quatro os pontos conhecidos mais próximos dentro da janela 3×3 e, dependendo de quais pontos são interceptados, este cálculo será realizado de uma forma diferente.

No caso mais simples, quando os pontos conhecidos dentro da janela não são interceptados, ou quando todos os pontos são interceptados, o Ponto 3 é calculado como a média aritmética dos quatro pontos conhecidos.

Quando apenas um ponto conhecido é interceptado, como na Figura 3.24(a), o Ponto 3 é calculado como a média dos pontos da diagonal que não foram interceptados pela borda.

Quando a janela é interceptada em dois pontos conhecidos, como na Figura 3.24(b), são dois os casos a seremem analisados no cálculo do Ponto 3. Se os dois pontos interceptados não são os da diagonal, então o Ponto 3 será a média dos outros dois pontos. Caso contrário, será a média dos dois pontos interceptados.

Quando a borda intercepta três pontos conhecidos, como na Figura 3.24(c), o Ponto 3 é calculado como sendo a média dos dois pontos da diagonal interceptados pela borda.

3.11.2. Interpolação Adaptativa com Degradação Controlada das Bordas

No algoritmo apresentado anteriormente, dividiu-se a região da imagem, que está dentro da janela 3×3, em duas outras regiões distintas através de uma certa configuração de borda dada pela Figura 3.24. Desta forma, dependendo de qual região o pixel a ser calculado se encontra, é utilizado no seu cálculo apenas a informação de sua região. A estrutura do algoritmo aqui proposto é a mesma do algoritmo anterior, ou seja, também é considerada a posição do pixel a ser calculado relativa a dos pixels conhecidos, assim como a configuração local de bordas (inclui-se aí a direção tomada por elas para evitar serrilhamento). Deste modo, a modificação realizada consistiu em permitir uma certa interpenetração das duas regiões no cálculo de um determinado pixel. Isto é feito ponderando-se as informações a serem utilizadas de cada uma das duas regiões consideradas, sendo que é dado um peso maior para aquela região que contém o pixel alvo. Assim, quanto menor for a diferença entre os pesos dados para cada uma das regiões, maior é a degradação local de bordas.

Capítulo 4

Resultados de Interpolação

4.1. Introdução

No capítulo anterior, desenvolvemos uma série de algoritmos tanto para interpolação não-linear quanto para otimização de parâmetros. Neste capítulo, apresentamos resultados quantitativos e qualitativos, obtidos para o filtro não-linear, cuja estrutura é neural, e para filtros adaptativos que exploram a estrutura da imagem em bordas e regiões.

Também, estabelecemos a hipótese de que a similaridade entre as diferentes resoluções da imagem, poderia ser "aprendida" por um filtro não-linear, a partir do conhecimento de apenas dois níveis de resolução. Com base nos resultados obtidos, verificamos que esta hipótese é válida.

Visto que foram desenvolvidos e implementados diferentes métodos de otimização, uma análise da eficiência destes algoritmos é aqui realizada.

4.2. Eficiência dos Algoritmos Desenvolvidos

O objetivo desta seção é definir qual dos algoritmos de otimização, apresentados no capítulo anterior, é o melhor. Uma vez definido o melhor algoritmo, ele será empregado na otimização do interpolador não-linear. Ao invés de utilizar o próprio interpolador nesta decisão, optamos pela mesma função objetivo multimodal, usada no Capítulo 2, pois neste

caso a velocidade de processamento é muito maior. Assim,

$$f(x) = -\frac{\sin(4\pi x)}{4\pi x},$$

que deverá ser minimizada no intervalo $-10 < x < 10$. Esta função é mostrada na Figura 2.13. A representação cromossômica foi obtida a partir da quantização do intervalo de interesse com 32 bits, que foram representados computacionalmente como *unsigned int.*

A Tabela 4.1 compara três dos algoritmos implementados : o AG puro na primeira linha, o H3 na segunda linha e o H2 na terceira linha. Esta tabela também ilustra o efeito do tamanho da população N e do número máximo L_e em que não há melhora alguma na população, resultando em extinção e imigração.

A partir dos resultados obtidos na Tabela 4.1, concluímos que o melhor algorítmo de otimização para $f(x)$ é o H3, ou seja, o algoritmo que utiliza, além dos operadores usuais do AG, um operador que implementa uma mutação baseada no método do gradiente. Isto é equivalente a implementar o AG e o gradiente numa mesma geração, com a diferença que o gradiente é aplicado apenas no melhor indivíduo.

Já o algoritmo H2 não apresentou vantagens com relação ao AG. Neste caso, o AG roda paralelamente ao gradiente e a interação entre os dois só ocorre quando não há melhora na população em L_e gerações. Entretanto, devido à multimodalidade da função objetivo, o gradiente fica preso, em poucas gerações, em um ótimo local, de modo que, ao interagirem, o AG já estará em um ponto melhor do que o gradiente. É provável que o desempenho não tenha sido melhor por causa da definição do ponto em que os dois algoritmos devem interagir. Em vez de defini-lo como sendo a geração em que ocorre a extinção e imigração, poderíamos defini-lo como sendo a geração em que há melhora no gradiente, ou seja, quando o gradiente está preso em um ótimo local.

4.3. Exemplos de Curvas de Erro Obtidas para o Filtro Não-Linear

O objetivo desta seção é mostrar alguns exemplos de curvas de erro, que foram obtidas pelos algoritmos implementados durante a etapa de treinamento do filtro não-linear. A imagem de treinamento utilizada é um trecho de 64×64 que engloba o olho da Lenna.

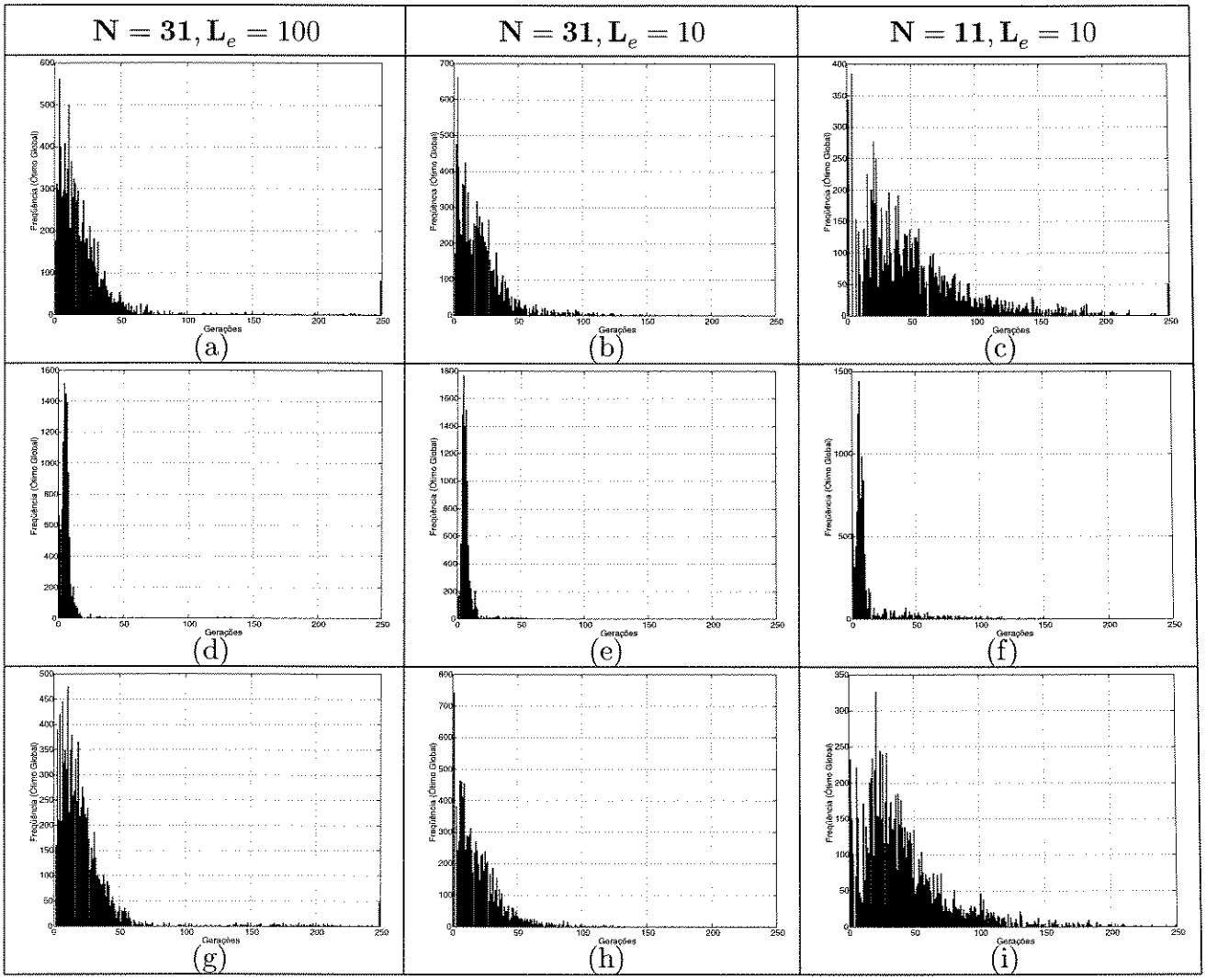

Tabela 4.1: Tabela de desempenho dos algoritmos implementados. As curvas apresentadas relacionam o número de vezes em que o algoritmo convergiu numa determinada geração, sendo que as curvas (a), (b) e (c) são resultados obtidos para o AG puro, as curvas (d), (e) e (f) para o H3 e, finalmente, (g), (h) e (i) para o H2.

Estas curvas são interessantes porque refletem a capacidade dos algoritmos de otimização, aqui desenvolvidos, de escapar de mínimos locais.

Inicialmente, podemos observar na Figura 4.1 que o GC (Gradiente Conjugado) fica preso em um mínimo local. Sabemos que é um mínimo local porque tanto o H3 quanto o H2 encontraram pontos de mínimo inferiores ao ponto obtido pelo GC. Além disso, a curva é suave, porque o GC segue um vetor direção que sempre aponta para a direção em que a função objetivo decresce. Em outras palavras, o GC sempre desce continuamente, de modo que se o ponto inicial estiver em uma região dominada por um ótimo local, o GC encontrará este ponto e ficará preso nele.

O algoritmo híbrido H2 é uma mistura do AG com o Gradiente, sendo que ambos rodam paralelamente e interagem nas gerações em que ocorrem extinção e imigração. Observe na Figura 4.2 que o AG decresce rapidamente, enquanto o Gradiente fica preso em um mínimo local. Quando a população do AG deixa de evoluir, ocorrendo extinção e imigração, o melhor indivíduo do AG é passado para o Gradiente, o que explica a queda abrupta do Gradiente. Isto ocorre sempre, de modo que a função efetiva do Gradiente neste algoritmo é apenas de refinar, nas gerações finais, o ponto encontrado pelo AG. Por este motivo, o desempenho do H2 deve ser bem parecido com o desempenho do AG, como foi mostrado na seção anterior.

A curva do EQM do algoritmo H3, mostrada na Figura 4.3 é bastante interessante. Observe que ela é uma curva suave, mas com descidas íngremes, refletindo características das curvas do Gradiente e do AG. De fato, o H3 pode ser visto como a ligação em série do AG com o gradiente, visto que em cada iteração, uma nova população será gerada, a partir da antiga, pelo AG e, em seguida, o melhor indivíduo é mutado de modo a seguir a descida do gradiente. Observe ainda, após comparar o número de gerações necessárias para a convergência de ambos H3 e H2, que o H3 é realmente mais eficiente do que o H2 e, portanto, mais eficiente do que o AG e o Gradiente usados individualmente. Dessa forma, podemos supor que o H3 é realmente mais eficiente e robusto, pois seu desempenho é superior tanto para o caso de $f(x)$ quanto para os filtros não-lineares.

Na seção seguinte , apresentamos imagens interpoladas pelo filtro não-linear e analisamos os resultados tanto quantitativamente, através do EQM, quanto qualitativamente, através da qualidade visual subjetiva das imagens obtidas em monitores e impressoras.

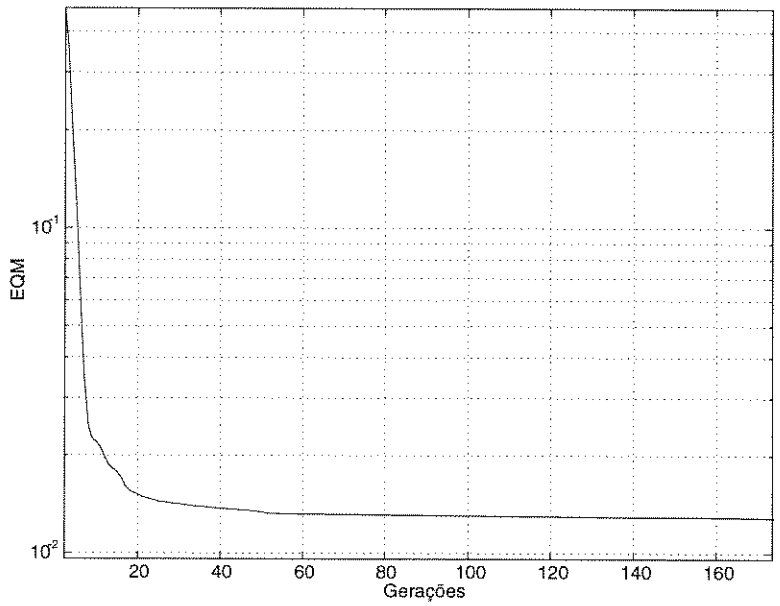

Figura 4.1: Curva do EQM para o algoritmo Gradiente Conjugado.

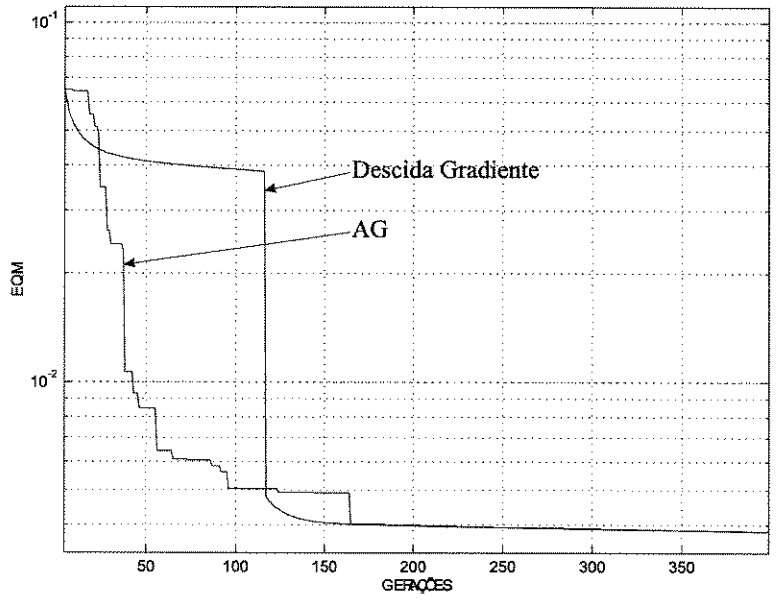

Figura 4.2: Evolução, em relação ao EQM, da população do AG paralelamente com a descida do Gradiente.

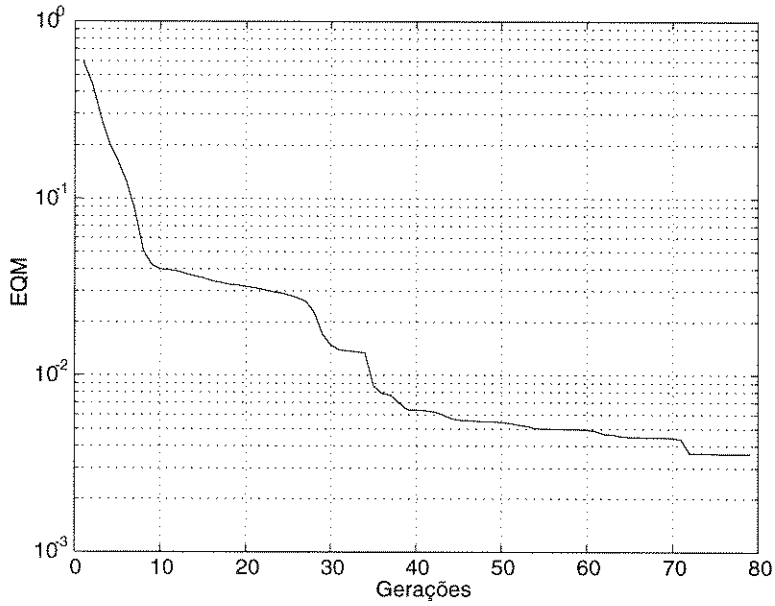

Figura 4.3: Curva do EQM, geração por geração, até a convergência do algoritmo híbrido H3.

4.4. Validação do Modelo de Estimação Representado pela Rede Neural

O objetivo desta seção é mostrar que o modelo de estimação, representado pela rede neural bidimensional após a fase de treinamento, consegue estimar com sucesso uma imagem de maior resolução, a partir de sua versão com menor resolução. Além disso, mostramos qualitativamente e quantitativamente que a interpolação pelo filtro não-linear apresenta resultados melhores do que o da interpolação bilinear. Neste trabalho, usamos dizimações e ampliações apenas por fatores de potência de dois.

Foram utilizadas cinco imagens, sendo que quatro delas são versões dizimadas da imagem Lenna. Definimos I_1 como sendo a imagem original Lenna e $I_{1/2}$, $I_{1/4}$, $I_{1/8}$ e $I_{1/16}$ suas versões dizimadas por 2, 4, 8 e 16, respectivamente.

Utilizamos um filtro não-linear com estrutura 5 :4 :1 :1 e o otimizamos com as imagens $I_{1/8}$ e $I_{1/4}$, sendo $I_{1/8}$ a imagem de entrada e $I_{1/4}$ a imagem desejada na saída do filtro. Após o treinamento, foram realizadas interpolações entre os diversos níveis de resolução da imagem I_1, com o objetivo de verificar a validade do modelo de estimação. As mesmas

	EQM(não-linear)	**EQM(bilinear)**	**Ganho**
$I_{1/2} \rightarrow I_1$	0,0020	0,0039	1,9196
$I_{1/4} \rightarrow I_{1/2}$	0,0036	0,0060	1,6620
$I_{1/8} \rightarrow I_{1/4}$	0,0076	0,0126	1,6549
$I_{1/16} \rightarrow I_{1/8}$	0,0264	0,0376	1,4242
$I_{1/4} \rightarrow I_1$	0,0070	0,0110	1,5826
$I_{1/8} \rightarrow I_1$	0,0220	0,0222	1,0091

Tabela 4.2: Interpolações relacionando diferentes níveis de resolução da imagem Lenna.

imagens que foram interpoladas pelo filtro não-linear, também foram interpoladas pelo bilinear. Visto que a saída ideal é conhecida, podemos calcular o erro quadrático médio (EQM) das duas técnicas empregadas. Em seguida, medimos o ganho, em termos do EQM, obtido pelo filtro não-linear em relação ao bilinear. As imagens obtidas pelas duas técnicas, também são apresentadas e comparadas.

Nas Figuras 4.5 a 4.10 é visualmente nítida a melhoria da resolução da imagem interpolada pelo filtro não-linear, quando comparada com o resultado obtido pela interpolação bilinear. Observe que as imagens obtidas pela interpolação bilinear estão mais borradas e apresentam menos detalhes que aquelas obtidas pelo filtro não-linear. Com relação ao erro quadrático médio, como é mostrado na Tabela 4.2, também constatamos uma melhora na qualidade da imagem interpolada de forma não-linear.

O interpolador não-linear utiliza a interpolação bilinear, em uma etapa inicial, com o objetivo de dobrar o tamanho da imagem. Em seguida, aplica-se o filtro não-linear propriamente dito. Podemos afirmar então que o filtro não-linear "aprendeu", na etapa de treinamento, como enriquecer a imagem obtida pela interpolação bilinear, sendo que este aprendizado se deu apenas através de dois níveis de resolução da imagem a ser interpolada. Por exemplo, suponha que a imagem a ser interpolada seja $I_{1/4}$. O treinamento do filtro não-linear é feito a partir dos dois níveis de resolução $I_{1/8}$ e $I_{1/4}$, sendo que $I_{1/8}$ é obtida por dizimação de $I_{1/4}$. Após o treinamento, constatamos que o filtro não-linear aprendeu como enriquecer a interpolação bilinear de $I_{1/8}$, visto que o ganho em relação ao EQM obtido pela interpolação bilinear foi de 1.6549 (linha 3 da Tabela 4.2). Mais ainda, o filtro conseguiu generalizar o modo de enriquecer a interpolação bilinear para obter outros níveis de resolução de $I_{1/4}$ para obter $I_{1/2}$ e I_1, sendo que pela Tabela 4.2, os ganhos com relação

a interpolação bilinear foram 1,662 e 1,9196 respectivamente.

Como já foi dito, o interpolador não-linear obteve resultados melhores, em termos de resolução, do que a interpolação bilinear. Apesar do filtro não-linear ter realçado as bordas, ele não conseguiu reduzir o serrilhamento que a interpolação bilinear inseriu nas bordas da imagem. Isto ocorreu por causa da medida utilizada para otimizar tal filtro não-linear. De fato, erros nas bordas se manifestam muito pouco no EQM da imagem, de modo que outra medida teria que ser utilizada. Uma idéia seria dividir a imagem em regiões de borda, regiões de textura e regiões planas, calcular o EQM em cada uma destas regiões separadamente e, finalmente, ponderar estas medidas dando um peso maior para o EQM de bordas. É muito provável que isto reduzirá bastante o serrilhamento, mas esta hipótese precisa ser validada.

4.5. Interpolação Adaptativa com Preservação de Bordas

Esta seção tem por objetivo apresentar os resultados do filtro não-linear, localmente adaptativo, para a preservação das bordas da imagem. Dois algoritmos foram implementados.

O primeiro, interpola uma imagem considerando em que região, determinada pela borda, o ponto a ser calculado se encontra, de modo que estas bordas sejam preservadas ao máximo. De fato, este objetivo (preservação máxima das bordas) foi alcançado com sucesso, como pode ser visto a partir das Figuras 4.13 e 4.15. Entretanto, este algoritmo dá um aspecto muito artificial à imagem, como pode ser visto na Figura 4.14. Parece que ela foi desenhada. Por este motivo, desenvolvemos, a partir deste algoritmo, um outro, que permite uma certa degradação (controlada) das bordas. Isto é feito considerando as duas regiões, localmente divididas por uma borda, no cálculo do ponto de interesse. Entretanto, dá-se um peso maior para a região na qual este ponto está situado. O resultado é mostrado na Figura 4.16. Observe por este resultado, que a imagem perdeu o aspecto artificial que tinha antes e, ao comparar este resultado com o da Figura 4.18, obtida por interpolação bilinear, observamos e constatamos, a partir das Figuras 4.17 e 4.19, que este novo algoritmo preserva melhor as bordas do que um algoritmo convencional (interpolação bilinear). Além disso, a resolução da imagem, obtida por este novo algoritmo, possui

Figura 4.4: Interpolação bilinear da imagem $I_{1/2}$ com ampliação por um fator de 2.

Figura 4.5: Interpolação não-linear da imagem $I_{1/2}$ com ampliação por um fator de 2.

Figura 4.6: Interpolação bilinear da imagem $I_{1/4}$ com ampliação por um fator de 2.

Figura 4.7: Interpolação não-linear da imagem $I_{1/4}$ com ampliação por um fator de 2.

Figura 4.8: Interpolação bilinear da imagem $I_{1/8}$ com ampliação por um fator de 2.

Figura 4.9: Interpolação não-linear da imagem $I_{1/8}$ com ampliação por um fator de 2.

uma qualidade melhor do que a da imagem obtida pela interpolação bilinear. Todavia, a pesar do resultado ter sido visualmente melhor, observamos que o EQM de uma imagem interpolada por esta técnica, quando comparado com o EQM de uma imagem obtida por interpolação bilinear, é maior. Isto mostra que o EQM não é uma boa medida no que se refere à qualidade subjetiva da imagem. Seria necessária uma medida que refletisse melhor os aspectos subjetivos, tais como preservação de borda e surgimento de artefatos indesejados na imagem. Se uma medida como esta tivesse sido utilizada, o resultado do interpolador adaptativo provavelmente seria melhor, visto que o interpolador foi projetado para obter uma imagem interpolada agradável subjetivamente, por exemplo, levou-se em conta a diferença entre duas regiões separadas por bordas assim como a direção tomada pela borda, de modo a evitar efeito de serrilhamento de borda.

Concluindo, foram implementados dois interpoladores adaptativos, sendo que o segundo foi uma evolução do primeiro, com o objetivo de fornecer resultados melhores do que os das correspondentes técnicas lineares, no que se refere aos aspectos subjetivos da análise psicovisual humana.

Figura 4.10: Interpolação bilinear da imagem $I_{1/4}$ com ampliação por um fator de 4.

Figura 4.11: Interpolação não-linear da imagem $I_{1/4}$ com ampliação por um fator de 4.

Figura 4.12: Imagem a ser interpolada.

Figura 4.13: Bordas da imagem a ser interpolada.

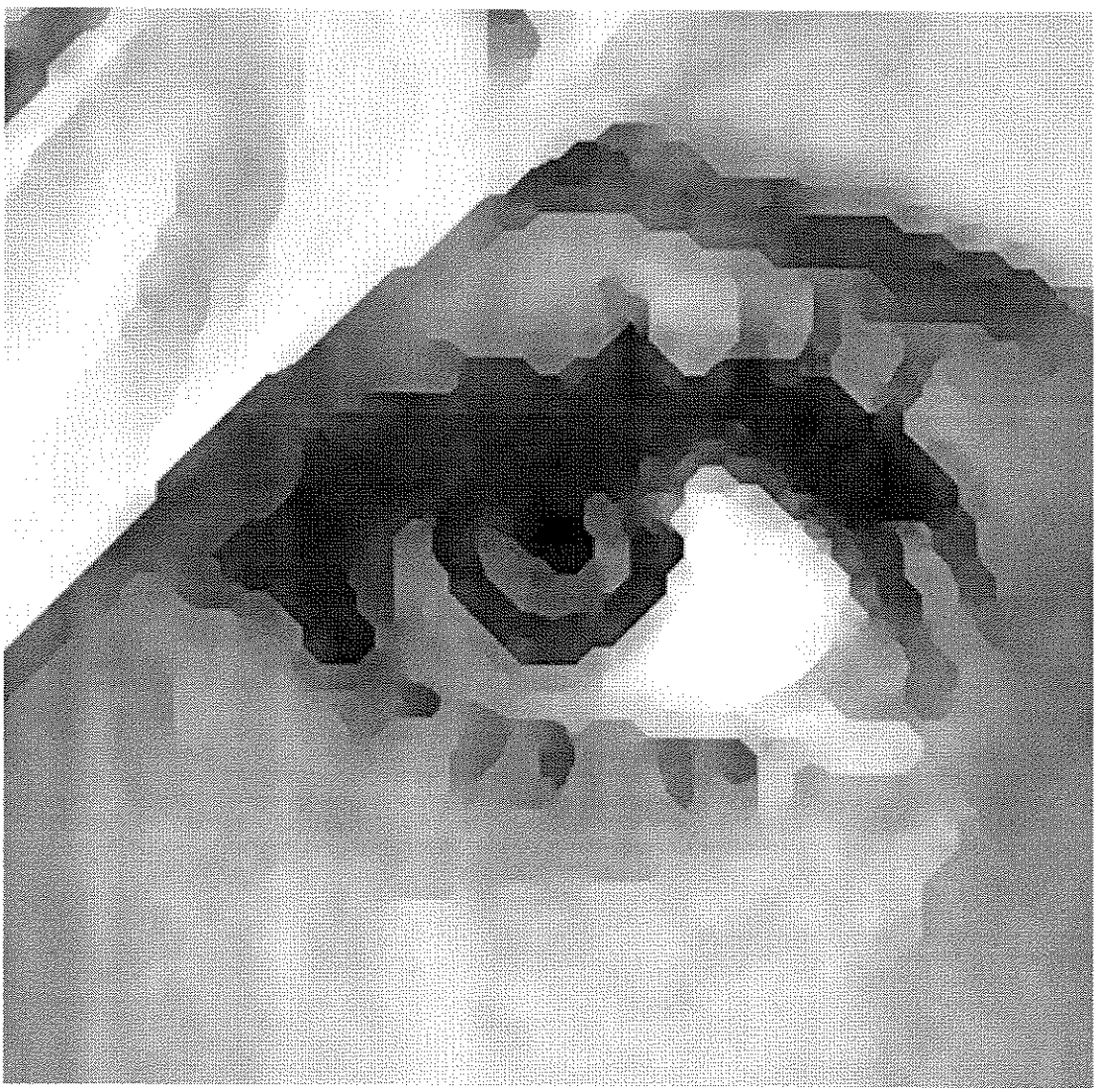

Figura 4.14: Imagem interpolada por um filtro não-linear adaptativo, de acordo com a disposição local das bordas da imagem para obter máxima preservação das mesmas.

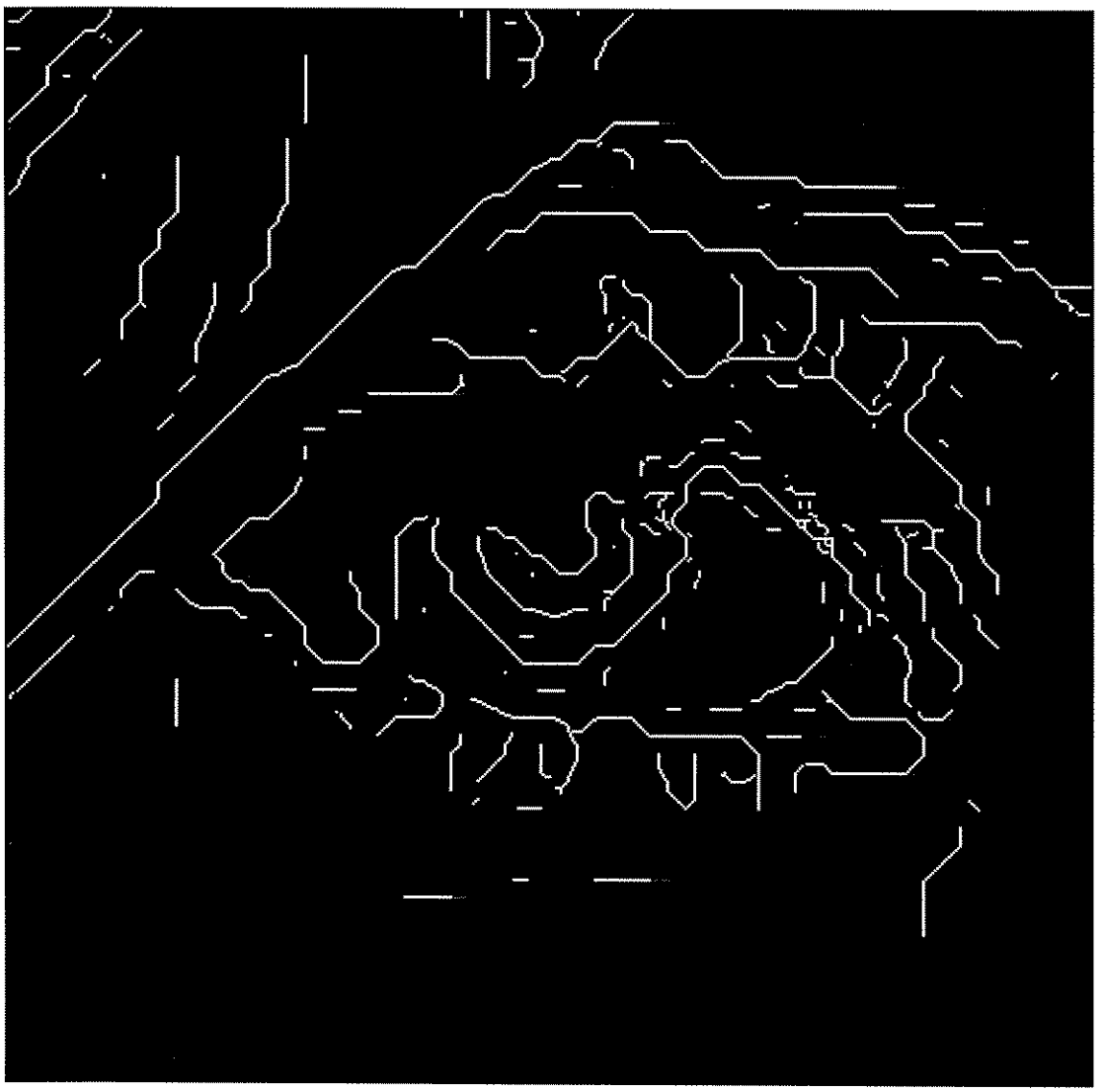

Figura 4.15: Bordas da imagem interpolada por um fator de 8.

Figura 4.16: Imagem interpolada pelo filtro não-linear adaptativo *modificado* com ampliação de 8.

Figura 4.17: Bordas da imagem interpolada pelo filtro adaptativo modificado.

Figura 4.18: Interpolação bilinear com ampliação de oito vezes.

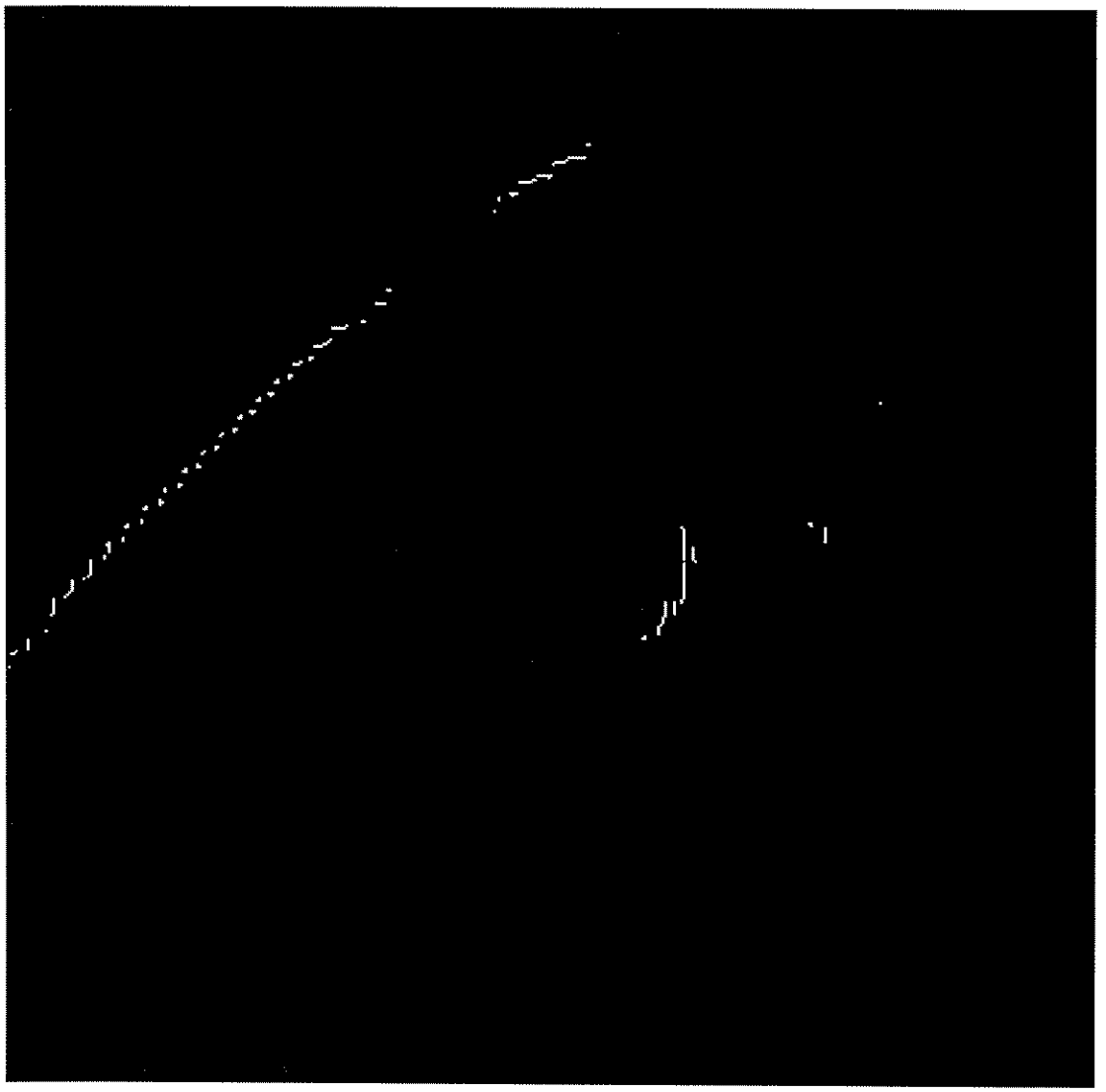

Figura 4.19: Bordas da imagem ampliada de oito vezes pelo interpolador bilinear.

Capítulo 5

Decodificação de Códigos Lineares

5.1. Introdução

O objetivo deste Capítulo é estudar a aplicação de Algoritmos Genéticos ao problema da decodificação de códigos corretores de erros [24]. As operações básicas de cruzamento, mutação e seleção natural são escolhidas de forma apropriada ao problema da decodificação de códigos lineares, sem a necessidade do conhecimento de qualquer estrutura algébrica. Através de vários resultados de simulação, constatamos que essas técnicas evolucionárias podem fornecer interessantes resultados para este difícil problema. Os resultados que serão apresentados foram obtidos para códigos de bloco binários e decodificação com decisão abrupta, mas os algoritmos propostos podem ser generalizados para decisão suave e para outros tipos de códigos lineares.

Um código de bloco linear pode ser caracterizado como segue. Seja $\mathbf{V}_n$ o conjunto de todas as n-uplas sobre $\mathbf{GF(2)}$, o campo de Galois de dois elementos. $\mathbf{C}$ é um código de bloco linear binário (n, k) se ele for um subespaço de $\mathbf{V}_n$ de dimensão k. $\mathbf{C}$ pode ser compactamente representado pela sua matriz geradora $\mathbf{G}$ ou pela correspondente matriz de paridade $\mathbf{H}$. Qualquer conjunto de k vetores linearmente independentes em $\mathbf{C}$ pode ser usado como linhas de $\mathbf{G}$, e qualquer conjunto de $n-k$ vetores linearmente independentes de $\mathbf{C}^{\perp}$, o espaço nulo de $\mathbf{C}$, pode ser usado como linhas de $\mathbf{H}$. Um vetor $\mathbf{V}_n$ é uma palavra-código em $\mathbf{C}$ se e somente se ele for uma combinação linear sobre $\mathbf{GF(2)}$ das linhas de $\mathbf{G}$. Portanto, uma palavra-código em $\mathbf{C}$ pode ser escrita como $\mathbf{c} = \mathbf{u} \cdot \mathbf{G}$, onde u é uma k-upla sobre $\mathbf{GF(2)}$. Como $\mathbf{C}$ e $\mathbf{C}^{\perp}$ são subespaços ortogonais, qualquer vetor

$\mathbf{v} \in \mathbf{V_n}$ é uma palavra-código de $\mathbf{C}$ se e somente se ele for ortogonal a toda linha de $\mathbf{H}$, isto é, $\mathbf{v} \cdot \mathbf{H^T} = \mathbf{0}$.

Seja $\mathbf{c} = (c_0, c_1, ..., c_{n-1})$ uma palavra-código de $\mathbf{C}$ transmitida sobre o Canal Binário Simétrico e seja $\mathbf{r} = (r_0, r_1, ..., r_{n-1})$ o vetor binário recebido. Um *decodificador ótimo* produz uma palavra-código $\mathbf{c}$ que é mais próxima de $\mathbf{r}$ no sentido da métrica de Hamming.

Acontece que este é um problema NP-completo, para o qual ainda não se encontrou uma solução definitiva e satisfatória (se é que existe) para códigos longos [25]. Para códigos com ricas estruturas algébricas, existem excelentes soluções para este problema [26], e a chamada *explosão combinatorial* característica dos problemas NP-completos, fica bastante suavizada. Quando a distância mínima e a distribuição de pesos do código são conhecidas, algoritmos eficientes baseados em Inteligência Artificial podem ser construídos [25]. Outros algoritmos baseados em treliças podem ser usados, mas apenas para códigos curtos ou com taxas muito baixas ou muito altas [25].

Apresentamos aqui uma técnica de decodificação baseada em Algoritmos Genéticos (AGs) para o caso mais geral onde se conhece apenas a matriz geradora $\mathbf{G}$. O método utiliza os conjuntos de informação como um substrato natural para a representação cromossômica, sobre o qual a evolução pode ser realizada sobre o espaço de busca altamente restrito. Nosso objetivo é investigar se um algoritmo evolucionário pode prover soluções eficientes para este problema, visto que ele tem sido usado com muito sucesso em outros problemas combinatoriais de difícil solução[27], [28], [7]. Não estaremos interessados na decodificação ótima propriamente dita, mas na aplicação dos AGs em decodificadores sub-ótimos, mas eficientes.

5.2. Considerações Preliminares

A idéia por trás dos AGs utilizados é converter o problema da decodificação em um problema de busca em torno do vetor recebido $\mathbf{r} = (r_0, r_1, ..., r_{n-1})$. Tal busca será realizada a partir de uma população de possíveis soluções que está sujeita a operadores probabilísticos. Estes irão alterar a população de tal modo que, em um número finito de gerações, o indivíduo mais adequado seja a solução desejada (ou bem próxima dela).

Como vimos nos capítulos anteriores, os AGs se baseiam no encadeamento de três

operações fundamentais : **cruzamento, mutação e seleção natural**. Quatro tipos de cruzamentos são estudados para este problema de decodificação. O primeiro deles corresponde à forma mais simples encontrada na literatura, baseando-se na simples troca de fragmentos entre pares de cromossomos. O segundo tipo é conhecido como cruzamento uniforme e também é bastmante empregado em AGs. Os dois últimos cruzamentos exploram características particulares do problema em estudo, de modo que a busca seja realizada na região do espaço em torno do vetor recebido, onde é mais provável de se encontrar a palavra ótima a ser decodificada. Além disso, dois tipos de mutação são implementados. O primeiro é o mais comumente utilizado, conforme já apresentado nos capítulos anteriores. Já o segundo explora características particulares do problema da decodificação, de modo a acelerar a convergência [3].

O ciclo básico do algoritmo proposto é mostrado na Figura 5.1. Ele consiste basicamente da mesma estrutura apresentada anteriormente, mas com algumas pequenas diferenças A primeira etapa consiste em gerar aleatoriamente um conjunto finito (N) de possíveis soluções, de acordo com a representação cromossômica escolhida. Tal conjunto de possíveis soluções caracteriza a População Inicial. A cada ciclo em que a população é modificada, de acordo com as operações probabilísticas a serem definidas, diz-se que se passou uma geração. Neste trabalho, um indivíduo da população será sempre uma palavra-código.

No passo seguinte do algoritmo, geram-se $2N$ filhos através de cruzamentos. Mantêm-se então pais e filhos juntos, em uma população intermediária com $3N$ indivíduos, que serão avaliados em seguida. Neste problema de decodificação, tal avaliação é feita com base na distância de Hamming de cada indivíduo com relação ao vetor recebido ($\mathbf{r}$). A partir da avaliação, tem-se uma medida da adequabilidade de cada indivíduo. Em seguida, guardam-se os M indivíduos mais adequados, ou seja, aqueles que mais se aproximam da solução desejada. Aplica-se então o operador Mutação à população intermediária de $3N$ indivíduos, mas de tal forma a se manter inalterado o número de indivíduos na população corrente. Avalia-se novamente tal população e escolhem-se os M melhores, que são então comparados com os M indivíduos que foram guardados anteriormente. A partir destes $2M$ indivíduos guardam-se os M melhores.

O passo seguinte consiste em eliminar indivíduos da população, através da Seleção Natural, de tal modo que ela seja reduzida novamente para N indivíduos. Conforme

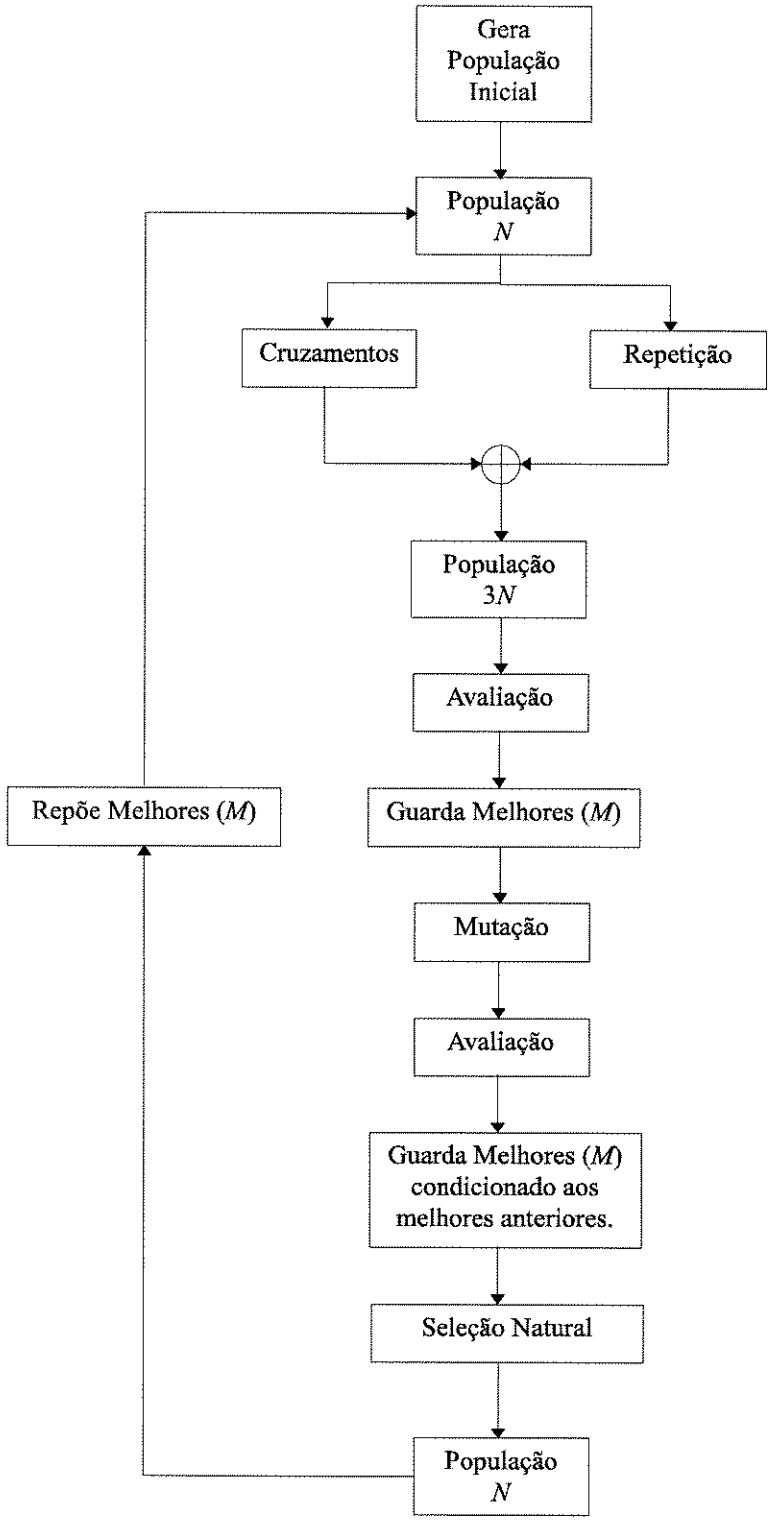

Figura 5.1: Estrutura básica do algoritmo genético utilizado como decodificador.

já discutido, a Seleção Natural é uma operação probabilística que tende a eliminar os indivíduos menos adequados. Entretanto, tal operação também pode eliminar indivíduos com boa adequabilidade, ou seja, os melhores podem ser eliminados da solução, o que compromete a convergência do algoritmo. Para se evitar este problema, adota-se uma estratégia de seleção, denominada de *elitista,* que consiste em substituir M indivíduos da população pelos M melhores salvos anteriormente. Este critério é fundamental para a convergência do algoritmo.

5.3. Representação Cromossômica

Um dos aspectos mais importantes nos Algoritmos Genéticos (mas pouco considerado na literatura) é a chamada *representação cromossômica*, isto é, o modo com que as soluções factíveis (ou candidatas) do problema são representadas vetorialmente, em geral na forma binária, sob a forma de cromossomos. Para o problema da decodificação, poder-se-iam utilizar, por exemplo, os k dígitos de informação de um código sistemático para representar um cromossomo de tamanho k. Os $n - k$ dígitos restantes seriam calculados pelas equações de paridade. Dessa forma, *um indivíduo* de tamanho n (palavra-código) possuiria *um cromossomo* de tamanho k. Embora aparentemente razoável, esta solução leva a desempenhos decepcionantes. Em nossas simulações, tal representação resultou até em casos de não-convergência do algoritmo, mesmo para códigos relativamente curtos.

Considerando então a importância deste problema de representação, decidimos investigar novas formas de "cromossomizar" os indivíduos. Optamos pela representação de *um indivíduo* por *dois cromossomos*, sendo um destes necessariamente um *conjunto de informação (CI)* do código e, o outro, uma palavra qualquer com n bits pertencente ou não ao código e que chamaremos de *cromossomo X*. O objetivo é guiar a busca, inserindo a estrutura do problema no algoritmo genético. Isto melhora significativamente a eficiência do algoritmo, que é específico para códigos lineares.

Em um código de blocos (n, k), define-se *conjunto de informação* como sendo um conjunto de k posições das palavras-código que podem ser especificadas independentemente. As demais $n - k$ posições são referidas como bits de paridade. Desta forma, se ruído tiver sido introduzido pelo canal no vetor recebido, alterando alguns bits da pa-

lavra originalmente transmitida, e se for possível encontrar um conjunto de informação de modo que estes bits errados sejam bits de paridade, então este vetor é decodificado corretamente.

Nesta tese, um conjunto de informação será representado por um vetor n-dimensional, em que k posições são iguais a 1 e as demais são iguais a 0. Assim, as suas posições não nulas indicam que as correspondentes colunas da matriz G são linearmente independentes. Tal vetor conjunto de informação é representado neste artigo pelo símbolo $\mathbf{p}$. Como ilustração, consideremos a matriz $\mathbf{G}$ de um código de bloco binário $(7, 4)$, não sistemático, dada por

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

Um possível vetor conjunto de informação para este código, seria

$$\mathbf{p} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

Os vetores

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix},$$

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

e

$$\begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

também são exemplos de conjuntos de informação, ao passo que

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

e

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

não correspondem a conjuntos de informação, como pode ser observado facilmente da matriz $\mathbf{G}$. Nota-se destes exemplos que as posições não nulas dos vetores associados aos conjuntos de informação, correspondem a colunas na matriz $\mathbf{G}$ que juntas são linearmente independentes.

Aqui existirá uma diferença importante entre cromossomo e indivíduo. A idéia que propomos neste trabalho é representar um indivíduo da população (palavra-código) no AG por um vetor qualquer $\mathbf{x}_i$ n-dimensional (que não é necessariamente uma palavra-código), juntamente com um vetor conjunto de informação $\mathbf{p}_i$. Se tomarmos as posições de $\mathbf{x}_i$ que correspondem às componentes não nulas de $\mathbf{p}_i$, podemos gerar uma palavra código $\mathbf{c}_i$ utilizando as equações de paridade obtidas da matriz $\mathbf{G}$. Assim, embora $\mathbf{x}_i$ não seja uma palavra código, o indivíduo $\mathbf{c}_i$ representado pelo par $(\mathbf{x}_i, \mathbf{p}_i)$ é garantidamente uma palavra código. Esta representação possibilitará a definição de operações de cruzamento e mutação sobre *conjuntos de informação*, de forma a se garantir soluções que são sempre factíveis. Isto é, a finalidade do vetor conjunto de informação é garantir que os indivíduos gerados a partir dos operadores probabilísticos continuem sendo palavras código. Resumindo, $(\mathbf{x}_i, \mathbf{p}_i)$ será o par de cromossomos do indivíduo $\mathbf{c}_i$.

5.4. Modelos de População

Definida a representação cromossômica, três modelos populacionais são propostas neste trabalho. No primeiro modelo, os cromossomos CI dos filhos são gerados aleatoriamente e os operadores genéticos atuam apenas nos cromossomos X dos pais. Com isto, o AG, a partir de uma determinada função de adequabilidade, deve evoluir estes cromossomos X, investigando vetores $\mathbf{x}_i$, numa vizinhança do vetor recebido, de modo que sejam corrigidos apenas os bits errados de $\mathbf{r}$ que são bits de informação.

Uma evolução natural desta estratégia é a utilização de um *modelo poligâmico* para a população. Este modelo corresponde na natureza ao caso em que há um indivíduo macho para várias fêmeas. Neste modelo, o macho é o indivíduo $(\mathbf{r}, \mathbf{p}_0)$ e as fêmeas são os indivíduos $(\mathbf{x}_i, \mathbf{p}_i)$. A idéia é utilizar o material genético do vetor recebido como referência para analisar com mais eficiência a sua vizinhança.

No terceiro modelo, os indivíduos da população são inicialmente do tipo $(\mathbf{r}, \mathbf{p}_i)$. Isto significa que todos os cromossomos tipo X são, inicialmente, o próprio vetor recebido. Ao longo das gerações, o operador genético cruzamento deve atuar apenas nos cromossomos CI e o operador mutação no cromossomo X. A idéia é encontrar um vetor $\mathbf{p}_i$, por cruzamento, de modo que os bits de informação do cromossomo X estejam todos corretos.

Além disso, deseja-se também corrigir, por mutação, os bits errados do cromossomo X, sejam eles bits de informação ou não.

5.5. População Inicial

De acordo com a representação cromossômica adotada na seção anterior, a população inicial poderia então ser obtida a partir de pares $(\mathbf{x}_i, \mathbf{p}_i)$ gerando-se, aleatoriamente, N vetores associados a conjuntos de informação. A população inicial é obtida a partir da geração aleatória de vetores $\mathbf{x}_i$ e $\mathbf{p}_i$ de acordo com a estratégia adotada. Por exemplo, para $N = 4$, teríamos :

- Modelo 1 : $\{(\mathbf{x}_0, \mathbf{p}_0), (\mathbf{x}_1, \mathbf{p}_1), (\mathbf{x}_2, \mathbf{p}_2), (\mathbf{x}_3, \mathbf{p}_3)\}$;
- Modelo 2 (poligâmico) : $\{(\mathbf{x}_0, \mathbf{p}_0), (\mathbf{x}_1, \mathbf{p}_1), (\mathbf{x}_2, \mathbf{p}_2), (\mathbf{r}, \mathbf{p}_3)\}$, sendo os três primeiros indivíduos fêmeas e o último macho;
- Modelo 3 : $\{(\mathbf{r}, \mathbf{p}_0), (\mathbf{r}, \mathbf{p}_1), (\mathbf{r}, \mathbf{p}_2), (\mathbf{r}, \mathbf{p}_3)\}$.

5.6. Avaliação, Adequabilidade e Seleção

Consideremos o indivíduo (palavra-código) $\mathbf{c}_i = (\mathbf{x}, \mathbf{p}_i)$. Neste problema de decodificação, a função adequabilidade deve ser definida a partir da distância de Hamming entre uma palavra código $\mathbf{c}_i$, candidata, e o vetor recebido $\mathbf{r}$. Seja $D_H(\mathbf{c}_i, \mathbf{r})$ esta distância. Aqui a adequabilidade de uma palavra código $\mathbf{c}_i$ será então definida por

$$a(\mathbf{c}_i) = \frac{1}{[1 + D_H(\mathbf{c}_i, \mathbf{r})]}. \tag{5.1}$$

A partir desta função, pode-se definir a probabilidade de seleção de uma palavra $\mathbf{c}_i$ por

$$p(\mathbf{c}_i) = \frac{a(\mathbf{c}_i)}{\sum_{j=0}^{N-1} a(\mathbf{c}_j)}. \tag{5.2}$$

Assim, a seleção natural tem por base a probabilidade dada pela equação acima. A idéia é montar uma roleta com base nas probabilidades de seleção de cada indivíduo da

população. Em outras palavras, gera-se um número aleatório ξ entre 0 e 1 e calcula-se

$$\sum_{i=0}^{n} p(\mathbf{c}_i) < \xi \leq \sum_{i=0}^{n+1} p(\mathbf{c}_i), \tag{5.3}$$

onde $p(\mathbf{c}_N) = 0$ (por convenção), para se decidir pela seleção do n-ésimo indivíduo. Entretanto, tal roleta poderia privilegiar demasiadamente os indivíduos mais aptos, selecionando-os mais de uma vez. Isto acarretaria em *perda de diversidade* da população. Uma alteração que garantiria a não repetição do indivíduo selecionado, seria zerar a sua respectiva probabilidade de seleção e na próxima seleção gerar um número entre 0 e $1 - \sum_{j \in J} p(\mathbf{c}_j)$, onde J é o conjunto de índices dos indivíduos já selecionados.

5.7. Cruzamentos utilizados

Sejam os indivíduos $\mathbf{c}_i = (\mathbf{x}, \mathbf{p}_i)$ e $\mathbf{c}_k = (\mathbf{y}, \mathbf{p}_k)$, $i, k \in \{0, 1, \cdots, N-1\}$, e $\mathbf{x}$ e $\mathbf{y}$ vetores binários quaisquer n-dimensionais. Uma maneira de se gerar novos indivíduos é através da combinação de $\mathbf{x}$ e $\mathbf{y}$ de modo que dois filhos sejam gerados. Dos quatro tipos de cruzamento implementados, três utilizam este tipo de combinação.

Outro forma de combinação possível utiliza indivíduos do tipo $\mathbf{c}_i = (\mathbf{r}, \mathbf{p}_i)$ e $\mathbf{c}_k = (\mathbf{r}, \mathbf{p}_k)$, $i, k \in \{0, 1, \cdots, N-1\}$ e os filhos gerados são obtidos pela combinação de $\mathbf{p}_i$ com $\mathbf{p}_k$. Um dos cruzamentos implementados utiliza este tipo de combinação.

5.7.1. Cruzamento Tipo 1

Este implementa a combinação de $\mathbf{x}$ com $\mathbf{y}$. Neste caso, o AG é implementado para utilizar o primeiro modelo populacional. Esta recombinação cromossômica é realizada da forma mais simples possível, ou seja, consiste em sortear uma posição de corte nos cromossomos paternos de modo que o material genético de ambos possa ser trocado para gerar dois outros cromossomos. Entretanto, os filhos só serão gerados quando for sorteado um conjunto de informação para cada cromossomo. Tal sorteio é feito com base num grupo de N conjuntos de informação utilizados para gerar a população inicial. A Figura 5.2 ilustra o que foi dito. O procedimento que descreve como implementar este cruzamento é descrito abaixo :

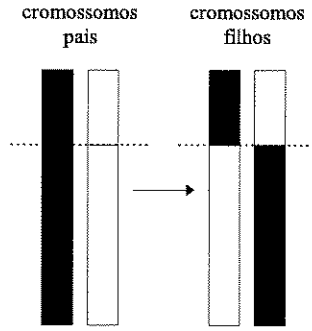

Figura 5.2: A operação cruzamento 1 em um algoritmo genético. A linha tracejada corresponde à posição de corte que foi sorteada aleatoriamente. Pela troca de material genético entre os cromossomos paternos obtém-se os cromossomos filhos.

1. Escolhem-se na população dois indivíduos aleatoriamente, sendo que estes dois indivíduos podem ser iguais. Suponha que os dois indivíduos escolhidos tenham sido $\mathbf{c}_i = (\mathbf{x}, \mathbf{p}_i)$ e $\mathbf{c}_k = (\mathbf{y}, \mathbf{p}_k)$.
2. Pareia-se os cromossomos $\mathbf{x}$ e $\mathbf{y}$. Em seguida, gera-se uma posição de corte com distribuição de probabilidade uniforme para estes cromossomos.
3. Realiza-se a recombinação descrita pela Figura 5.2 nos cromossomos $\mathbf{x}$ e $\mathbf{y}$ para se gerar os cromossomos $\mathbf{u}$ e $\mathbf{v}$. Após este passo, obtém-se os filhos $\mathbf{f}_i = (\mathbf{u}, ?)$ e $\mathbf{f}_k = (\mathbf{v}, ?)$. Neste ponto, os cromossomos CI de $\mathbf{f}_i$ e $\mathbf{f}_k$ estão indefinidos.
4. Finalmente, são gerados aleatoriamente dois CIs, um para cada filho, $\mathbf{p}'_i$ e $\mathbf{p}'_k$. Feito isto, os indivíduos filhos estão formados e correspondem a $\mathbf{f}_i = (\mathbf{u}, \mathbf{p}'_i)$ e $\mathbf{f}_k = (\mathbf{v}, \mathbf{p}'_k)$.

Repetindo este procedimento N vezes, geram-se $2N$ filhos.

5.7.2. Cruzamento Tipo 2

Equivalente ao Cruzamento 1 exceto que a combinação é uniforme. Neste caso o AG também é implementado para utilizar o modelo 1 de população. O método de recombinação consiste em lançar uma moeda para cada bit do casal de pais selecionados para decidir qual bit vai para qual filho. A Figura 5.3 ilustra o que foi dito para a formação do cromossomo de um único filho, sendo que os bits restantes irão formar o cromossomo do outro filho.

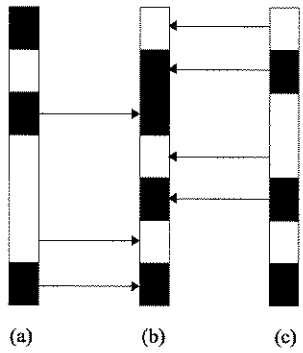

Figura 5.3: Cruzamento uniforme entre o cromossomo do pai (a) e o da mãe (c) para gerar um dos filhos (b).

5.7.3. Cruzamento Tipo 3

Este tipo de cruzamento também envolve a combinação dos cromossomos $\mathbf{x}$ e $\mathbf{y}$. Entretanto, utiliza-se o modelo populacional poligâmico. Isto significa que no procedimento descrito no Cruzamento 1, o cromossomo $\mathbf{y}$ é sempre igual ao vetor recebido $\mathbf{r}$. O método de recombinação cromossômica é o mesmo do Cruzamento 2, ou seja, o cruzamento uniforme, com a diferença que um dos pais é sempre o mesmo ($\mathbf{r}$). A idéia é restringir o espaço de busca, pois aqui todos os indivíduos gerados serão de certa forma semelhantes ao vetor recebido. Não se deve esquecer que os filhos só serão gerados depois que cada cromossomo tiver o seu conjunto de informação.

5.7.4. Cruzamento Tipo 4

A utilização deste tipo de cruzamento pressupõe a utilização do terceiro modelo populacional aqui proposto. Neste cruzamento, o cromossomo X é fixo e corresponde ao vetor recebido. A idéia consiste em combinar inteligentemente os conjuntos de informação (CI) dos pais (seu segundo cromossomo) para se obter outros dois conjuntos de informação que, juntamente com o cromossomo $\mathbf{r}$, gerarão os filhos. Aqui tenta-se cruzar dois conjuntos de informação para se obter um outro CI que seja *tão diferente quanto possível* dos CIs paternos. Por este motivo, podemos chamar este cruzamento de *Aberrativo.* Isto é feito na tentativa de escolher k bits de informação no vetor recebido isentos de erro pois, se o

algoritmo ainda não convergiu para a solução desejada, significa que algumas das posições do vetor recebido, que fazem parte dos CIs utilizados, estão erradas. O procedimento é descrito abaixo :

1. O primeiro passo consiste em escolher dois indivíduos aleatoriamente, por exemplo, $\mathbf{c}_i = (\mathbf{r}', \mathbf{p}_i)$ e $\mathbf{c}_k = (\mathbf{r}, \mathbf{p}_k)$.
2. Pareia-se os cromossomos $\mathbf{p}_i$ e $\mathbf{p}_k$ para gerar um cromossomo $\mathbf{q}_i$, como descrito adiante.
3. Inicializa-se as n posições de cromossomo $\mathbf{q}_i$ com zeros.
4. Para l variando de 1 a n bits, faça :
   (a) Se $\mathbf{p}_i[l] = \mathbf{p}_k[l] = 0$ faça :
   $$\mathbf{q}_i[l] \leftarrow 1;$$
   (b) Se as colunas da matriz geradora $\mathbf{G}$, correspondentes às posições de $\mathbf{q}_i$ iguais a um, forem linearmente dependentes, faça :
   $$\mathbf{q}_i[l] \leftarrow 0;$$
5. fim de Para.
6. Se o código é (n, k) e foram encontradas $m < k$ posições em $\mathbf{q}_i$ iguais a um, então distribui-se $k - m$ elementos iguais a 1 nas posições restantes de $\mathbf{q}_i$ de modo que as k correspondentes colunas da matriz $\mathbf{G}$ sejam linearmente independentes.
7. Finalmente, sorteia-se, entre os cromossomos X dos pais, aquele que será herdado pelo filho. Por exemplo, suponha que $\mathbf{r}'$. Se isto ocorre, então o filho gerado é $\mathbf{f}_i = (\mathbf{r}', \mathbf{q}_i)$, cujos pais são $\mathbf{c}_i$ e $\mathbf{c}_k$.

Este procedimento deve ser repetido $2N$ vezes para se gerar $2N$ filhos.

5.8. Mutações utilizadas

Os métodos de mutação aqui desenvolvidos devem operar apenas nos cromossomos X dos indivíduos, qualquer que seja o modelo populacional adotado.

5.8.1. Mutação Tipo 1

Este operador altera alguns indivíduos da população simplesmente invertendo um ou mais bits de seu primeiro cromossomo $\mathbf{x}$. Salienta-se que tal alteração é aleatória.

5.8.2. Mutação Tipo 2

Este operador também altera alguns indivíduos da população por inversão de bit. Entretanto, tal inversão é realizada apenas em um bit, e sempre no sentido de melhorar a adequabilidade do indivíduo. Se não for possível melhorar a adequabilidade, então tal indivíduo não é alterado. A idéia é novamente direcionar e supervisionar a busca, de modo a acelerar a convergência do algoritmo.

5.9. Resultados

Quatro tipos de códigos de bloco lineares binários foram ulizados para testar o decodificador genético : o código de Golay (23,12), um código (30,15), um código (40,20) e o código de resíduos quadráticos extendido (48,24). Os resultados obtidos são apresentados em forma de histograma, onde o eixo das abcissas corresponde ao número de gerações pelas quais a população evoluiu até que um dos indivíduos, o mais adequado que surgisse na população, fosse a solução desejada. Em outras palavras, correponde ao número de iterações necessárias que o algoritmo precisou para convergir. Já o eixo das ordenadas corresponde à freqüência com que o AG convergiu para um dado número de gerações.

Estaremos quase sempre interessados em situações de pior caso, onde o vetor recebido apresenta muitos erros e pode se localizar na fronteira das regiões de decisão (onde a decodificação é muito mais difícil). De antemão, escolheremos sempre um vetor recebido para o qual conhecemos pelo menos uma palavra código mais próxima. Dessa forma, saberemos quando o algoritmo convergiu e o número de gerações utilizadas. Isto nos permite avaliar a eficiência do método e projetar parâmetros para decodificação normal.

Algumas combinações de operadores são testadas neste trabalho. A ordem em que tais resultados são apresentados corresponde à evolução cronológica de tais operadores ao

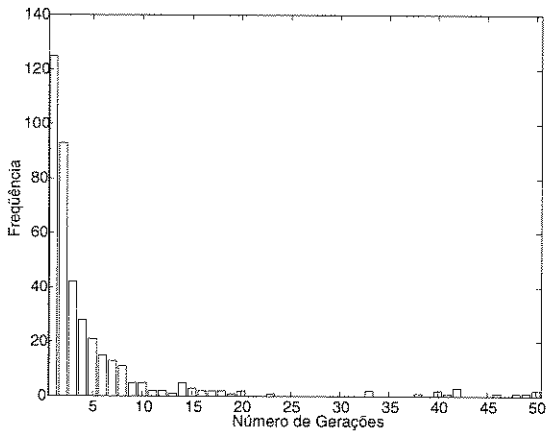

Figura 5.4: Histograma do número de gerações necessárias para que o AG implementado com os operadores Cruzamento 1 e Mutação 1 convirja. O código utilizado foi o de Golay (23,12) e o tamanho da população inicial foi dez.

longo da nossa pesquisa. O resultado apresentado na Figura 5.4 corresponde à forma mais simples possível de AG, devido à simplicidade dos operadores cruzamento e mutação. O algoritmo foi executado quatrocentas vezes, sendo que em alguns casos levou-se mais de 50 gerações para convergir. O vetor recebido neste caso possuía 3 bits errados, que é a capacidade máxima do código de Golay, e o desempenho foi apenas razoável.

Na Figura 5.5 mudou-se ambos os operadores cruzamento e mutação. O cruzamento passa a ser do tipo uniforme e a mutação a ser do tipo 2. A diferença de desempenho é bastante significativa, principalmente se se considerar que o código utilizado é mais longo e difícil de decodificar. Tal ganho de desempenho é devido principalmente à técnica de mutação empregada pois, embora o resultado não seja apresentado, verificou-se que existe um ganho de desempenho inexpressivo se apenas o operador cruzamento for substituído.

Sabemos que a função do operador cruzamento é realizar uma busca local ao longo do espaço. Os operadores cruzamento tradicionais realizam esta função. Ao substituirmos o método de cruzamento uniforme pelo método tipo 3, estamos restringindo tal busca a uma região próxima ao vetor recebido. Na prática, o ganho não foi tão significativo assim. Como pode ser visto na Figura 5.6, houve um pequeno aumento do número de convergências nas primeiras gerações e um espalhamento para gerações mais elevadas. Todavia, este é um resultado que deve ser verificado com mais cuidado, executando-se o algoritmo um número maior de vezes.

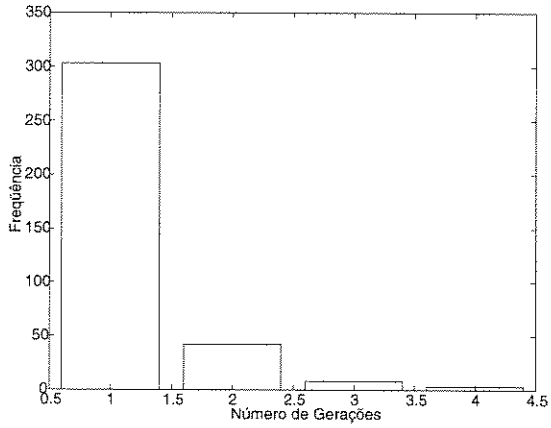

Figura 5.5: Histograma do número de gerações necessárias para que o AG implementado com Cruzamento 2 e Mutação 2 convirja. O código utilizado é o (30,15), a palavra transmitida é um vetor de zeros e a recebida um vetor com distância de Hamming 5 com relação à solução desejada.

Finalmente, substituiu-se o operador Cruzamento 3 pelo Cruzamento 4 e verificou-se que a combinação dos operadores Cruzamento 4 e Mutação 2 é a ideal. O ganho no desempenho com esta combinação de operadores foi muito grande quando comparada com as anteriores. Alguns resultados utilizando esta combinação de operadores são mostrados nas Figuras 5.7 e 5.8. Além dos resultados apresentados, diversas outras simulações foram feitas mostrando que o algoritmo aparentemente sempre converge para uma solução válida. Mais ainda, o algoritmo tem uma característica interessante, que é a indiferença com relação às soluções possíveis. Por exemplo, para o vetor recebido

$$\mathbf{r} = 000000000000000000000000000000000000000001111111,$$

as 1000 execuções representadas pelo histograma da Figura 5.8 levaram às soluções mostradas na Figura 5.9, todas elas igualmente válidas. Tal resultado mostra que o vetor $\mathbf{r}$ estava equidistante de nove soluções possíveis.

Deve-se observar que o algoritmo trabalhou com um número muito pequeno de amostras do espaço de busca. Por exemplo, o espaço de busca do código $(48, 24)$ consiste de $2^{24} = 16.777.216$ palavras, enquanto que para os exemplos apresentados utilizou-se apenas 4 indivíduos na população. À medida que o tamanho da população cresce, o desempenho melhora significativamente.

É importante enfatizar que a representação cromossômica desenvolvida neste trabalho

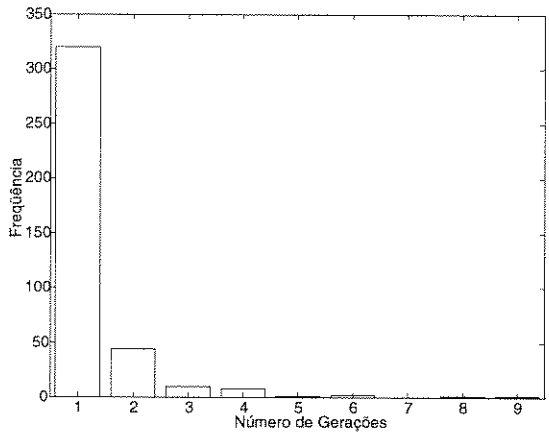

Figura 5.6: GA implementado com Cruzamento 3 e Mutação 2. O código utilizado é o (30,15) e a distância de Hamming do vetor recebido corresponde a cinco bits em relação à palavra transmitida.

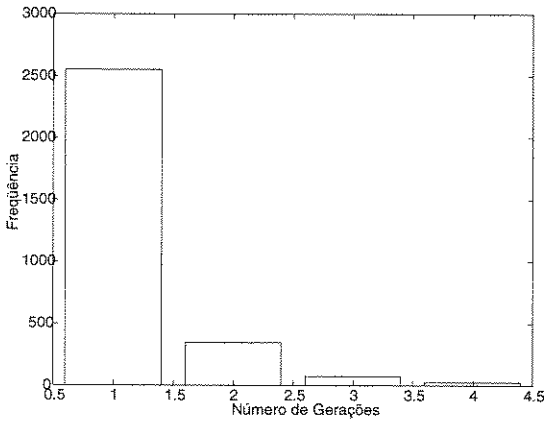

Figura 5.7: Histograma do número de gerações necessárias para que o AG implementado com Cruzamento 4 e Mutação 2 convirja. O código utilizado foi o (40,20). O vetor recebido tem distância 5 da palavra transmitida e o tamanho da população é de apenas 4.

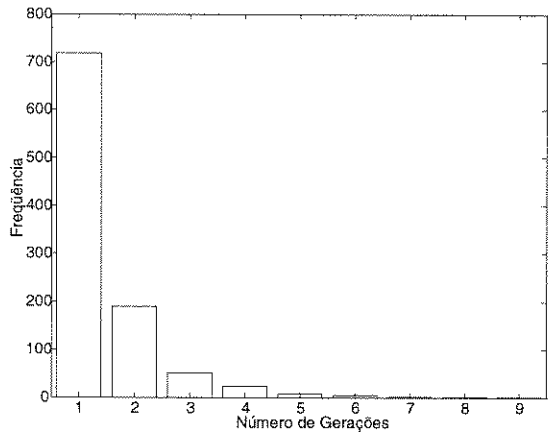

Figura 5.8: Histograma do número de gerações necessárias para que o AG implementado com Cruzamento 4 e Mutação 2 convirja. O código utilizado foi o (48,24). O vetor recebido tem distância 7 da palavra transmitida e o tamanho da população é de apenas 4.

Vetor recebido:
000000000000000000000000000000000000000001111111
Soluções:
000000000000000000000000000000000000000000000000
000000000000001010000010100000100000000101110111
000000010000000000100000011000000001000100111111
100010000100000000000000100000000001100001111011
000100000100010001000000000000010000000101101111
000100001000000000000001000000000011000011110111
000000100000000001000000110000000010001001111101
001000000000000010010000001000010010000001111011
000010010000011000000100000000000010000001111110

Figura 5.9: Soluções encontradas para o código (48,24) pelo AG.

foi fundamental para contornar o problema da geração de palavras não válidas, quando a população é submetida a um dos operadores probabilísticos dos AGs.

A eficiência do algoritmo, no que se refere ao pequeno número de amostras necessárias no espaço de busca, foi confirmada por intensa simulação de pior caso e para códigos relativamente complexos. Desenvolveu-se com êxito um par de operadores probabilísticos cruzamento-mutação, que realizam uma busca eficiente e rápida no espaço do código. A representação cromossômica escolhida, sem dúvida contribuiu decisivamente para a eficiência do algoritmo. Deve-se observar, todavia, que o algoritmo proposto é basicamente *probabilístico*, isto é, as melhores soluções intermediárias fornecidas pelo algoritmo podem não convergir para a solução ótima em um tempo finito pré-determinado.

Capítulo 6

Conclusões

6.1. Introdução

Conforme visto, esta tese está centrada na aplicação de algoritmos genéticos (AGs) na construção de modelos para interpolação não-linear de imagens, assim como na decodificação de códigos lineares. Na primeira parte do trabalho, os AGs foram empregados para otimizar uma rede neural bidimensional, visando à construção de um modelo de interpolação, específico à imagem a ser expandida, para enriquecer o espectro da imagem original através da predição de informações de alta freqüência. A partir deste método de interpolação, surgiu a idéia de se construir um modelo mais robusto de interpolação, que pudesse ser empregado a qualquer imagem sem a necessidade de treinamento. Neste segundo modelo, é realizada uma interpolação adaptativa de acordo com a configuração local das bordas da imagem, visando atender aspectos psicovisuais, tais como preservação e continuidade das bordas, assim como a diferença entre as distintas regiões da imagem que são separadas por bordas.

Na segunda parte do trabalho, são desenvolvidos esquemas de decodificação cuja estrutura é o próprio AG. Os esquemas de decodificação são implementados a partir do AG padrão, e são aperfeiçoados, a partir do projeto de operadores genéticos específicos, para o problema da decodificação.

Neste capítulo, as principais conclusões são destacadas. Vantagens e limitações destas técnicas são também discutidas e comparadas. Finalmente, são discutidas as propostas futuras de trabalho.

6.2. Principais Conclusões deste Trabalho

6.2.1. Interpolação Não-Linear de Imagens

A correlação existente entre os diferentes níveis de resolução de uma imagem, sugeria a possibilidade de se extrair um modelo de formação para se obter uma imagem de mais alta resolução a partir da imagem original. Sabendo da capacidade que as redes neurais têm em realizar um tipo de mapeamento não-linear, que é determinado durante a etapa de treinamento, surgiu a idéia de que tal rede poderia ser utilizada para representar o modelo de formação desejado. Entretanto, esta rede teria que ser adaptada, de modo a operar sobre os dados de entrada em todas as possíveis direções de um espaço bidimensional. A forma encontrada nesta tese, para realizar esta adaptação, foi encarar o problema do ponto de vista de filtros digitais unidimensionais. Nesta abordagem, a primeira camada escondida de uma rede neural tradicional, com alimentação direta, é vista como um banco de filtros FIR lineares unidimensionais, seguidos por uma função de transferência não-linear específica. Desta forma, foi fácil adaptá-la para uma entrada bidimensional, substituindo os filtros unidimensionais por filtros bidimensionais. Além disso, o treinamento da rede passou a ser visto como um problema de otimização de parâmetros.

De fato, constatamos que tal filtro não-linear, otimizado a partir de dois níveis de resolução conhecidos da imagem original, é capaz de caracterizar um modelo de formação que resultou em uma imagem interpolada com qualidade superior a de técnicas lineares convencionais. Com este resultado, também comprovamos a validade da hipótese de similaridade entre os diferentes níveis de resolução da imagem. Mais ainda, verificamos que o filtro não-linear é capaz de captar tal similaridade.

Quanto aos métodos de otimização empregados, foram desenvolvidos e implementados dois esquemas híbridos envolvendo gradiente e AG. Comparamos estes esquemas híbridos com o AG puro, constatando a melhor eficiência do esquema AG-Gradiente que utiliza, além dos operadores tradicionais, um operador de mutação baseado no gradiente.

Já o modelo utilizado no método de interpolação adaptativa, utiliza as bordas, previamente detectadas, como uma fronteira bem definida entre duas regiões bastante distintas da imagem original. Mais ainda, o modelo considera a direção tomada pelas bordas para evitar efeito de bloco. Inicialmente, o algoritmo desenvolvido não permitia interpenetração

das duas regiões, preservando ao máximo as bordas da imagem. Entretanto, este modelo de interpolação resultou uma imagem um tanto artificial, ou seja, com aspecto de desenho. A partir deste resultado, o algoritmo foi modificado para permitir interpenetração das duas regiões. Isto resultou em uma suavização das bordas, mas eliminou o problema da artificialidade observada na imagem expandida. Além disso, esta técnica de interpolação, quando comparada com o método de interpolação bilinear, gerou uma imagem expandida em que tanto o contraste quanto as bordas eram melhor realçados.

6.2.2. Decodificação de Códigos Lineares

Em códigos lineares, operações elementares definidas sobre um campo de Galois, tais como multiplicação por um escalar e adição de um múltiplo de uma palavra à outra, realizadas sobre palavras código, geram outras palavras que ainda pertencem ao código. Entretanto, as operações genéticas não são lineares e, conseqüentemente, podem gerar palavras que não pertencem ao código. Neste contexto, a representação cromossômica, desenvolvida neste trabalho, foi fundamental para contornar o problema de se gerar palavras não válidas a partir dos operadores probabilísticos (cruzamento e mutação) aqui implementados.

Já a eficiência do algoritmo, na decodificação de códigos corretores de erros, foi confirmada por intensa simulação de pior caso e para códigos relativamente complexos. Mais ainda, a partir desta análise de eficiência, constatou-se o êxito em se projetar pares de operadores cruzamento-mutação específicos ao problema tratado.

6.3. Vantagens e Limitações dos Métodos Desenvolvidos

Podemos enfatizar as seguintes vantagens dos métodos desenvolvidos neste trabalho :

- Todas as técnicas de interpolação aqui desenvolvidas, apresentaram uma imagem interpolada final de melhor qualidade subjetiva do que as técnicas convencionais.
- O filtro não-linear bidimensional, baseado no perceptron multicamadas, foi a técnica que efetivamente adicionou informações à imagem final, enriquecendo-a e aumentando sua resolução. Além disso, sua implementação pode ser realizada com técnicas

paralelas.

- Já a interpolação adaptativa foi a técnica que melhor preservou as bordas da imagem, assim como manteve a continuidade das mesmas sem a inserção de artefatos.
- Na decodificação de códigos lineares, desenvolveu-se operadores genéticos que, juntamente com a representação cromossômica adotada, conseguiu decodificar, com sucesso, vetores recebidos (de pior caso) a partir da inspeção de um número muito pequeno de amostras do espaço de busca. Por exemplo, para o código (48,24), conseguiu-se, para a maioria dos casos, decodificação ótima inspecionado apenas 72 pontos de um espaço composto por 2^{24} palavras.
- Baixa complexidade de implementação no caso da decodificação de códigos.

É importante enfatizar também algumas limitações dos métodos :

- O filtro não-linear não conseguiu evitar o efeito de serrilhamento nas bordas da imagem interpolada. Isto ocorreu devido à função de custo utilizada (EQM), que não enfatiza o erro ocorrido nas bordas. Pelo contrário, o EQM torna este erro desprezível ao considerar a imagem como um todo.
- Quando comparado com as técnicas lineares, o filtro não-linear é computacionalmente despendioso devido à etapa de treinamento.
- O filtro adaptativo não enriquece a imagem final como um todo, mas preserva a informação original, enfatizando e mantendo a continuidade das bordas.
- O decodificador de códigos lineares, aqui implementado, necessita do conhecimento da distância mínima do código para melhorar a eficiência da decodificação. Caso esta informação não esteja disponível, o algoritmo pode convergir para uma solução sub-ótima devido à adoção de um critério de parada baseado na não modificação do melhor indivíduo ao longo de um determinado número de gerações.

6.4. Limitações dos Experimentos Realizados

Todos os algoritmos aqui desenvolvidos foram implementados utilizando linguagem C, sendo compilados e executados em estações de trabalho baseadas no sistema Unix (Solaris 2.5). Neste ambiente, os experimentos realizados tiveram as seguintes limitações :

- As imagens interpoladas são avaliadas pela comparação de seus valores de EQM, o que impõe a necessidade de se ter uma imagem de referência para o cálculo do erro. Neste caso, a imagem a ser interpolada é uma versão dizimada da imagem de referência. Também são utilizados critérios subjetivos informais na avaliação de tais imagens. Nos casos em que a imagem de referência é desconhecida, analisamos, a partir de uma comparação subjetiva, as imagens obtidas pelas técnicas aqui desenvolvidas com aquelas obtidas por interpolação bilinear. Métodos mais efetivos de avaliação subjetiva necessitam ser introduzidos para uma validação mais completa.
- Todas as imagens aqui obtidas, foram comparadas apenas com a interpolação bilinear, que representava o conjunto de técnicas convencionais. Não foram feitas comparações destes resultados com aqueles obtidos por outras técnicas não-lineares.
- No caso do decodificador de códigos lineares, não foi feita uma análise comparativa, tanto em eficiência quanto em complexidade, do algoritmo aqui desenvolvido com outros decodificadores disponíveis na literatura.

6.5. Trabalhos Futuros

A principal limitação do filtro não linear é a qualidade das bordas da imagem interpolada, devido ao efeito de bloco. Já antecipamos que a causa deste efeito é a função objetivo utilizada. A partir deste problema, poderíamos sugerir duas soluções, para trabalhos futuros, que não são de difícil implementação.

Primeiro, poderíamos utilizar mais de uma função objetivo na etapa de treinamento, sendo que cada uma delas representaria um aspecto da imagem que deve ser melhorado. Por exemplo, poderíamos extrair, a partir da imagem desejada, uma máscara que determinaria qual a região de bordas desta imagem e, conseqüentemente, qual a região de bordas da imagem interpolada. Desta forma, estaríamos aptos a calcular, além do EQM global, o EQM de bordas, de modo que, durante a etapa de treinamento, tanto o EQM global quanto o de bordas sejam minimizados. Da mesma forma, este raciocínio pode ser estendido para as regiões planas da imagem. Mais ainda, podemos realizar um treinamento mais rigoroso nas regiões planas e de bordas e sermos mais flexíveis nas regiões de textura.

Segundo, podemos otimizar dois filtros não-lineares. Um para operar nas regiões de bordas e outro nas regiões de textura da imagem. Nas regiões planas, pode-se utilizar a interpolação bilinear. Entretanto, o treinamento destes dois filtros deve ser diferente. Sugerimos que o filtro de textura seja otimizado da mesma forma como foi feito nesta tese. Já o filtro de bordas, deve ser otimizado apenas uma única vez, para um conjunto de imagens, pois acreditamos na possibilidade de se obter um filtro de bordas genérico.

O método de interpolação adaptativa foi desenvolvido para uma janela de dimensão 3×3, porque as possíveis configurações de bordas, dentro desta janela, eram pequenas, o que tornou viável a realização de uma operação específica para cada uma destas possíveis configurações. Falta ainda generalizar este método para janelas de maior dimensão. A idéia é utilizar um algoritmo inteligente, que seja capaz de levantar e classificar todas as possíveis configurações de bordas para, a partir dessa informação, decidir qual operação realizar. Obviamente, existe o perigo da explosão combinatorial para janelas grandes. Mas, neste caso, AGs poderiam ser utilizados, a partir de dois níveis de resolução da imagem original, para decidir qual a melhor operação a ser realizada para uma dada configuração de bordas.

Na decodificação de códigos lineares, é necessário adaptar o algoritmo para realizar decodificação suave. Além disso, o algoritmo também deve ser generalizado para outros tipos de código.

Bibliografia

[1] Leehter Yao e William A. Sethares. *Nonlinear Parameter Estimation via the Genetic Algorithm.* IEEE Transaction on Signal Processing, vol 42, no. 4, pp. 927-935, abril, 1994.

[2] S.C. NG, S.H. Leung, C.Y. Chung, A. Luk e W.H. Lau. *The Genetic Search Approach : a new Learning Algorithm for Adaptativ IIR Filtering.* IEEE Signal Processing Magazine, pp. 38-46, novembro, 1996.

[3] K.S. Tang, K.F. Man, S. Kwong e Q. HE. *Genetic Algorithms and their Applications.* EEE Signal Processing Magazine, pp. 22-37, novembro, 1996.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* New York : Addison-Wesley, 1989.

[5] Kenneth Price e Rainer Storn. *Differential Evolution : A simple evolution strategy for fast optimization.* Dr. Dobb's Journal, pp. 18-24, abril, 1997.

[6] Hamid R. Rabiee e R. L. Kashyap. *GMLOS : A New Robust Nonlinear Filter for Image Processing Aplications.*

[7] Julio Tanomaru. *Motivação, Fundamentos e Aplicações de Algoritmos Genéticos.* II Congresso Brasileiro de Redes Neurais. Curitiba, novembro de 1995.

[8] M. A. Sid-Ahmed, *Image Processing : Theory, Algorithms, and Architectures.* McGraw-Hill, Inc. New York, 1995.

[9] K. P. Hong, J. K. Paik, H. J. Kim e C. H. Lee, *An Edge-Preserving Image Interpolation System for a Digital Camcorder.* IEEE Trans. on Consumer Eletronics, vol. 42, no. 3. Agosto, 1996.

[10] K. Jensen e D. Anastassious, *Subpixel Edge Localization and the Interpolation of Still Image.* IEEE Transaction on Image Processing, vol. 4, no. 3. Março, 1995.

[11] N. Herodotou e A. N. Venetsanopoulos. *Colour Image Interpolation for High REsolution Acquisition and Display Devices.* IEEE Trans. on Consumer Eletronics, vol. 41, no. 4. Novembro, 1995.

[12] Y. I. Wong, *Nonlinear Scale-Space Filtering and Multiresolution System.* IEEE Trans. on Image Processing, vol. 4, no. 6. Junho, 1995.

[13] N. Herodotou, L. Onural, A. N. Venetsanopoulos. *Image Interpolation Using a Simple Gibbs Random Field Model.* ICIP 95, Washington D.C. Outubro, 1995.

[14] R. R. Schultz, R. L. Stevenson, *A Bayesian Approach to Image Expansion for Improved Definition.* IEEE Trans. on Image Processing, vol. 3, no. 3, pp. 233-242. Maio, 1994.

[15] S. Lakshmanan, A. K. Jain, Y. Zhong, *Multi-Resolution Image Represntation Using Markov Random Fields.* IEEE Proc. ICIP, Vol. I, pp. 855-860, 1994.

[16] B. Zeng e A. N. Venetsanopoulos. *A Comparative Study of Several Nonlinear Image Interpolation Schemes.* Proceedings SPIE : Visual Communications and Image Processing, Vol. 1818, pp. 21-29. Novembro, 1992.

[17] Y. Wang e S. K. Mitra, *Motion/pattern adaptive interpolation of interleced video sequences.* Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (Toronto, Canada). pp. 2829-2832, Maio, 1991.

[18] D. De Vleeschauwer e I. Bruyland, *Nonlinear interpolators in Compatible HDTV Image Coding.* Signal Processing of HDTV (L. Chiariglione, Ed.). Amsterdam : Elsevier (North Holland), 1988.

[19] D. M. Martinez e J. S. Lim, *Spatial Interpolation of Interlaced Television Pictures.* Proc. IEEE Int. Conf. Acoust. Speech, Signal Processing (Scotland), pp. 1886-1889. Abril, 1989.

[20] A. V. Oppenheim e R. W. Schafer, *Discrete-Time Signal Processing.* Prentice-Hall, 1989.

[21] S. Haykin, *Neural Networks : A Comprehensive Foundation,* Prentice Hall, 1994.

[22] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," IEEE Trans. Signal Processing, vol. 41, pp. 3345-3462, December 1993.

[23] A. Said and W. A. Pearlman, "An Image Multiresolution Representation for Lossless and Lossy Image Compression," IEEE Trans. on Image Processing, Sept. 1996.

[24] Fabbryccio A. C. M. Cardoso e Dalton S. Arantes, "Decodificação de Códigos Lineares com Algoritmos Genéticos", XV Simpósio Brasileiro de Telecomunicações, SBT´97, Recife, Pe, Setembro, 1997.

[25] Y. S. Han, C. R. P. Hartmann and C. C. Chen, "Efficient Priority-First Search Maximum-Likelihood Soft-Decision Decoding of Linear Block Codes", IEEE Trans. on Information Theory, Vol. 39, N. 5, September, 1993.

[26] G. C. Clark and J. B. Cain, "Error-Correction Coding for Digital Communications", Plenum Press, New York, 1981.

[27] J. N. Amaral, "Genetic Algorithms and Evolutionary Neural Computation", invited lecture in Sian Ka'an International Conference : The First Joint Mexico-US International Workshop on Neural Networks and Neurocontrol, September, 1995.

[28] J. N. Amaral, K. Tumer and J. Ghosh, "Designing Genetic Algorithms for the State Assignment Problem", IEEE Transactions on Systems, Man and Cybernetics, pp.687-694, Vol. 25, No. 4, April 1995.